UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

Inteligência Computacional Distribuída: Arquitetura, Especificação Formal e Aplicação

Por : Gilberto Shigueo Nakamiti

Orientador: Prof. Dr. Fernando Antônio Campos Gomide

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Unicamp como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica e de Computação.

Campinas, abril de 1996.

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

N145i

Nakamiti, Gilberto Shigueo
Inteligência computacional distribuida : arquitetura, especificação formal e aplicação / Gilberto Shigueo Nakamiti.--Campinas, SP: [s.n.], 1996.

Orientador: Fernando Antonio Campos Gomide.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Inteligência artificial. 2. Engenharia de sistemas. 3. Conjuntos nebulosos. 4. Fluxo de tráfego. 5. Tráfego urbano. I. Gomide, Fernando Antonio Campos. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

À minha mãe Olimpia

Agradecimentos

Agradeço especialmente ao meu orientador, Prof. Dr. Fernando Antônio Campos Gomide, pela confiança e valiosa orientação em todas as fases do trabalho.

Agradeço ao secretário dos transportes do município de Campinas, Dr. Jurandir Fernandes, pela possibilidade de ter contado com o apoio técnico de sua secretaria. Agradeço ao engenheiro de tráfego Wagner Bonetti por suas valiosas considerações quanto às questões do tráfego urbano.

Agradeço ao amigo Renato Vitor Selleio pelo inestimável apoio e amizade nos períodos de maior dificuldade. Agradeço também aos amigos Ivan Rizzo Guilherme, Ricardo Luís de Freitas, Marco Antônio Leonel Caetano e demais amigos do Departamento de Estatística, Matemática Aplicada e Computacional da Unesp, campus de Rio Claro, bem como ao trabalho técnico do desenhista Gilberto Girello e de Eduardo Murai Soares.

Agradeço à Profa. Dra. Heloísa Vieira da Rocha do Instituto de Computação da Unicamp pelo apoio e incentivo.

Agradeço à minha mãe Olimpia pelo amor e apoio.

Agradeço ainda à Capes pelo suporte financeiro.

Sumário

A inteligência artificial distribuída tem como objetivo a criação de modelos flexíveis para a resolução de problemas através de um conjunto de agentes inteligentes. Cada um dos agentes, embora possuidor de conhecimento incompleto, incerto ou eventualmente inconsistente, deve interagir com seus pares, procurando resolver cooperativamente um problema comum.

A teoria dos conjuntos nebulosos tem sido foco de intensa pesquisa, tanto no campo teórico quanto na área de aplicações nos últimos anos. Sua utilização tem se estendido pelos mecanismos de tratamento de incertezas e a flexibilidade que provê aos sistemas.

Os sistemas baseados em casos provêem mecanismos que mimetizam características importantes do raciocínio humano na tomada de decisões, utilizando modelos e experiências de decisões anteriores para aplicá-las a situações similares novas. Permitem a tomada de decisões complexas, reaproveitando o esforço computacional anterior.

Os algoritmos genéticos constituem-se em uma ferramenta para a resolução de problemas e tomada de decisões em ambientes pouco estruturados. Propiciam adaptação e aumento de desempenho nesses ambientes, evitando máximos ou mínimos locais.

Este trabalho decorre da simbiose entre a inteligência artificial distribuída, a teoria dos conjuntos nebulosos, os sistemas baseados em casos e os algoritmos genéticos. Seu objetivo é o de propor uma abordagem única e coesa que apresente características de resolução distribuída de problemas, cooperação, manipulação de incertezas, flexibilidade, utilização de experiências anteriores e adaptação a novas situações. A abordagem proposta, denominada Inteligência Computacional Distribuída, fornece base para o desenvolvimento de um sistema distribuído de controle de tráfego urbano. Vários testes de desempenho são apresentados, comparando-se seus resultados com os de estratégias tradicionais de controle de tráfego urbano.

Abstract

Distributed artificial intelligence aims at creating flexible models for problem solving through intelligent agents. Agents have to interact with one another seeking for cooperation, despite their incomplete, uncertain or even inconsistent knowledge.

Fuzzy sets theory has been a focus of intense research in the last years, both in the theoretical as well as in the application fields. Its use has spread over for its mechanisms for handling uncertainties and for the flexibility it provides.

Case-based systems provides mechanisms to mimic human behavior in decision making through the use of past decisions models in similar situations. They allow complex decision making, re-using previous computational effort.

Genetic algorithms are often used for problem solving and decision making in new environments. They lead to adaptation and performance increase, avoiding local maxima or minima.

This work constitutes a symbiosis among distributed artificial intelligence, fuzzy sets theory, case-based systems, and genetic algorithms. It aims at introducing an approach that provides distributed problem solving, cooperation, uncertainties handling, flexibility, use of past experiences, and adaptation to new situations. The Distributed Computational Intelligence approach is a basis for the development of a distributed traffic control system. Several performance tests are presented, and their results are compared to conventional traffic control strategies.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Formulação do Problema

A distribuição do esforço computacional para a resolução de problemas vem sendo objeto de muita pesquisa nos últimos anos. Isso tem ocorrido não só pela redução de preço dos equipamentos, mas também pela possibilidade de aumentar a confiabilidade, a eficiência e a modularidade dos sistemas computacionais.

Na área da inteligência artificial, distingue-se a inteligência artificial distribuída, a qual procura unir os conceitos e técnicas da inteligência artificial com o processamento distribuído. Dentre os objetivos dessa abordagem, podemos incluir as tentativas de reproduzir e entender os mecanismos de raciocínio e interação humanas quando em sociedade, buscando a cooperação entre seus indivíduos.

Um problema básico da inteligência artificial distribuída são as diferentes representações internas de cada um dos indivíduos (mais comumente denominados agentes), que podem potencialmente conduzir os sistemas a soluções inadequadas. A mani-

pulação de informações incertas, incompletas ou inconsistentes é, assim, um fator crucial para a utilização de sistemas distribuídos de inteligência artificial em aplicações complexas, envolvendo muitos agentes e interações externas. Associada à questão do dinamismo ambiental, constata-se a necessidade de sistemas que se adaptem a tais variações, como forma de aumentar sua flexibilidade e viabilizar sua efetiva aplicação, sem aumentos significativos nos custos computacionais. Assim, um sistema de inteligência artificial distribuída deve ser capaz de manipular informações incertas e de adaptar-se às variações do meio, aprendendo sobre as novas situações com que se depara, dinamicamente.

Uma abordagem para o desenvolvimento de sistemas distribuídos inteligentes, que manipulem incertezas e adaptem-se ao seu meio ambiente, constitui-se, sem dúvida, em um importante tópico de pesquisa. Como exemplo de dificuldade para o entendimento e expansão das abordagens de inteligência artificial distribuída, podemos citar a sua representação. Grande parte dos sistemas de inteligência artificial possuem representações descritivas, o que dificulta o seu entendimento e agrega margens de interpretações quanto ao que se desejou exprimir. Várias delas apresentam representações procedimentais ou mesmo em alguma linguagem formal, embora isso não seja freqüente para os sistemas de inteligência artificial distribuída. Uma arquitetura genérica para o desenvolvimento de sistemas distribuídos inteligentes e adaptativos deveria utilizar uma linguagem formal ou semi-formal de representação.

Finalmente, um grande desafio para uma arquitetura genérica dessa natureza seria o de facilitar o desenvolvimento de aplicações complexas, em domínios dinâmicos, onde pudessem ocorrer muitas formas de interação internas e externas ao sistema, como na previsão do tempo distribuída, no controle distribuído de chãos de fábrica e no controle distribuído de tráfego.

1.2 Objetivos

O objetivo básico deste trabalho foi o de viabilizar uma arquitetura desse tipo. Para isso, é proposta uma nova abordagem no contexto da inteligência artificial distribuída, denominada Inteligência Computacional Distribuída (ICD). São agregadas construções da teoria dos conjuntos nebulosos, conhecida por sua capacidade de representação e manipulação de informações incertas à inteligência artificial distribuída. É proposto ainda um mecanismo evolutivo baseado em casos como forma de prover as características de adaptabilidade necessárias ao modelo. Para isso, propõe-se que os casos sejam representados como uma cadeia de genes, onde cada gene é a representação nebulosa de uma das características de um caso. Através de algoritmos genéticos, são propostos mecanismos de aprendizagem e derivação de situações e ações. O mecanismo nebuloso básico coexiste com o mecanismo baseado em casos, provendo maior flexibilidade ao modelo. A arquitetura é definida em termos de uma linguagem formal de especificação, baseada em lógica proposicional temporal. Esse tipo de lógica provê uma linguagem abstrata para caracterizar o tempo e para especificar as ações e escolhas tomadas pelos diferentes agentes.

Com o objetivo de validar a arquitetura, foi ainda proposto e desenvolvido um protótipo de um sistema distribuído para o controle de tráfego urbano. Trata-se de uma aplicação típica, mas que envolve inúmeras variáveis de fluxo, interações entre os diversos agentes, e eventos imprevistos, tais como acidentes.

1.3 Organização do Trabalho

No Capítulo 2, são revistos os princípios da inteligência artificial distribuída, a sua evolução, a classificação de sistemas, e as principais abordagens desenvolvidas. No Capítulo 3, são fornecidos os conceitos básicos da lógica nebulosa, dos sistemas distribuídos, dos sistemas baseados em casos e dos algoritmos genéticos. Por tratar-se

de um trabalho que envolve diversas áreas, procurou-se fornecer os conceitos básicos de cada uma delas, visando fornecer subsídios para o entendimento do trabalho, e não uma revisão completa e exaustiva de cada área. Para isso, as referências bibliográficas mais relevantes são citadas. O Capítulo 4 apresenta a proposta para a abordagem Inteligência Computacional Distribuída (ICD), bem como sua especificação formal. No Capítulo 5, é apresentado um sistema distribuído de controle de tráfego urbano, desenvolvido através dessa nova abordagem. Incluem-se ainda os resultados de simulação do sistema e a comparação com abordagens tradicionais de controle de tráfego urbano. A conclusão deste trabalho, suas contribuições e trabalhos futuros são apresentados no Capítulo 6. Finalmente, os apêndices incluem a relação completa dos gráficos de desempenho para todos os testes realizados, os cálculos efetuados para a obtenção dos tempos recomendados para a malha de trânsito utilizada, bem como os elementos da teoria de tráfego necessários ao desenvolvimento da aplicação.

Capítulo 2

Inteligência Artificial Distribuída

2.1 Introdução

Frequentemente, a pesquisa em inteligência artificial enfatiza abordagens onde um sistema único centralizado deve resolver seus próprios problemas, com pouco ou nenhum auxílio de sistemas adicionais. Esse tipo de enfoque, no entanto, encontra limitações para resolver diversas classes de problemas onde é usualmente necessária a existência e cooperação de sistemas diversos, em ambiente dinâmico. Outras limitações de sistemas centralizados incluem uma menor capacidade de processamento face ao seu custo, dificuldade de modelar e gerenciar todo o conhecimento necessário em um sistema único, e menor tolerância a falhas.

A Inteligência Artificial Distribuída (IAD) é uma área da ciência da computação que une os conceitos e técnicas da inteligência artificial com aquelas relativas ao processamento distribuído, visando tratar as limitações das abordagens tradicionais. A IAD pode ser dividida em inteligência artificial paralela, a abordagem conexionista de

distribuição de dados e de controle, e sistemas distribuídos inteligentes, abordagem de maior granularidade [16].

Um sistema de IAD pode ser visto como um grupo de entidades inteligentes, denominadas agentes, que interagem entre si. Um *agente*, segundo Bond [5], é um processo computacional com escopo bem definido e com controle local único. A interação entre os agentes visa possibilitar que o conjunto do seu conhecimento e de suas capacidades supere suas limitações individuais, conduzindo potencialmente todo o sistema a objetivos comuns. Assim sendo, cada agente, possuidor de uma visão parcial e imprecisa do estado e objetivo global do sistema, procura entender, prever e influenciar o estado global, segundo sua visão local [46]. Através de comunicação, os agentes podem trocar soluções, tarefas, objetivos e estimativas do estado global do problema.

Várias questões emergem do estudo da IAD, principalmente quanto às relações e a comunicação entre os diversos agentes. A seguir, serão apresentados os principais conceitos relativos à inteligência artificial distribuída, com o objetivo de prover a fundamentação para o entendimento de trabalhos na área. Serão apresentadas também as principais abordagens propostas e os principais sistemas desenvolvidos, com o intuito de ilustrar o seu uso e de refletir seu estado atual, bem como as potencialidades da pesquisa na área.

2.2 Histórico

As primeiras idéias sobre processamento concorrente utilizando inteligência artificial surgiram na década de 1950. Elas incluem as teorias de Minsky sobre sociedades de agentes cooperativos [64] e a conceituação de modelos de redes neurais [90, 94].

No início da década de 1970 foram propostos os sistemas de produção, que se propunham a modelar o processamento computacional a partir do raciocínio humano [83]. Essa abordagem alcançou grande sucesso, e difundiu o uso das *regras de produção*,

conhecidas pela forma *SE <condição> ENTÃO <ação>*, bastante utilizadas em sistemas especialistas. De sua evolução, surgiram os sistemas *blackboard*, através do projeto Hearsay-II [21, 22, 23, 62]. Os sistemas *blackboard* podem ser vistos como uma evolução dos sistemas especialistas clássicos, onde o conhecimento é segmentado em módulos, e a cada módulo é associada uma máquina de inferência. A comunicação entre os módulos dá-se através de uma memória comum denominada *blackboard*.

No início da década de 1980 foram propostos os sistemas Funcionalmente Corretos e Cooperativos (FA/C - *Functionally Accurate, Cooperative Systems*) [58, 61], onde os vários agentes trocavam soluções parciais para resolver inconsistências. Foram ainda propostos sistemas onde a alocação de tarefas era baseada em contratos, ou negociação entre agentes [1], como no sistema *Contract Net* [99]. Outro foco de estudo foram os atores e sistemas abertos de Hewitt [40], que procuravam identificar os mecanismos de comunicação existentes em uma sociedade de especialistas para a resolução de problemas.

Durante a década de 1980 foram refinados os modelos de ação da teoria de multi-agentes [29] ; de sistemas *blackboard*, dando origem a arquiteturas mais genéricas, como o sistema AGE (*Attempt to Generalize*) [84] e com agentes dotados de maior autonomia, como o DVMT (*Distributed Vehicle Monitoring Testbed*) [59] ; bem como a teoria de negociação e contratos [9].

Mais recentemente, importantes trabalhos vêm sendo desenvolvidos, os quais se agregam às abordagens anteriores, resolvendo muitos de seus problemas, como transmissão desnecessária de mensagens e tratamento de incertezas, aumentando seu escopo de atuação, ou propondo variações às estratégias propostas [77]. Como exemplos, temos a estratégia *Partial Global Planning*, [17], que é derivada dos trabalhos dos Sistemas Funcionalmente Corretos e Cooperativos, onde os nós devem utilizar seu conhecimento local para propor soluções viáveis para o problema. Tiveram também prosseguimento os trabalhos com sistemas abertos, com o objetivo de que conduzissem a sistemas abertos de computação distribuída ultra-concorrente [39, 41], além de trabalhos sobre

negociação entre agentes [9, 52], sistemas *blackboard* paralelos [89], orientados a objetos [119], aplicações em ambientes reativos [87, 88] e na automação industrial [25, 81].

2.3 Classificação

Algumas formas de classificação de sistemas de IAD foram propostas, como por exemplo em [7, 16]. Segundo [16], que apresenta uma forma simples, mas bastante abrangente de classificação, a inteligência artificial distribuída pode ser classificada segundo as seguintes características:

- o nível de decomposição (granularidade) dos processos;
- a distribuição do conhecimento;
- os métodos para obter controle distribuído;
- o processo de comunicação.

Quanto à decomposição, a IAD pode ser dividida em inteligência artificial paralela, a abordagem conexionista (decomposição em grãos mais finos), e sistemas distribuídos inteligentes (decomposição em grãos maiores). Em geral, a decomposição em grãos maiores leva a sistemas cognitivamente orientados. A inteligência artificial paralela, por ter características próprias, provindas, por exemplo, da influência de estudos neurofisiológicos, costuma ser estudada separadamente dos sistemas distribuídos inteligentes e compõe sub-áreas distintas, como a das redes neurais. A ênfase deste trabalho é relativa aos sistemas distribuídos inteligentes, de maior granularidade.

Os sistemas distribuídos inteligentes podem ser compostos por agentes que possuem conhecimento restrito (de curto espectro) ou conhecimento mais abrangente. No primeiro caso, compõem sistemas com fontes de conhecimento distribuídas e no segundo caso, os sistemas para resolução distribuída de problemas. A Figura 2.1, extraída de [16] ilustra essa classificação.

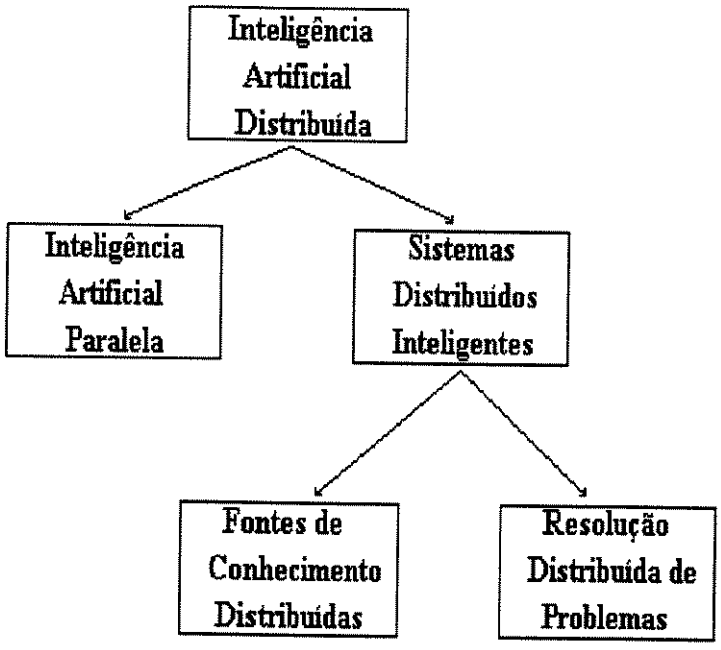


Figura 2.1: Classificação de sistemas em IAD

Os métodos para obter controle distribuído podem ser vistos em função de três características: a cooperação entre os agentes ; a sua organização, visto que cooperam de alguma forma ; e como a cooperação é obtida. *Cooperação*, segundo [100] é um método a ser utilizado quando os agentes possuem conhecimento distinto, a fim de minimizar a comunicação, permitir o balanceamento de carga de processamento e distribuir o controle, mantendo o comportamento coerente. Em [61], é sugerido que a cooperação se deve à necessidade de manipular incertezas no controle, que ocorrem quando a distribuição da informação de controle difere daquela onde as decisões de controle são tomadas. A *organização* entre os agentes de um sistema pode variar desde grupos totalmente livres (que podem também ser chamados de times ou comitês), onde não existe hierarquia explícita de controle, até relações do tipo mestre-escravo. Associado à organização, existe também o conceito de *coerência*, que indica o quão adequadas são as ações individuais dos agentes aos objetivos globais do grupo. Finalmente, a *dinâmica* refere-se a como a cooperação é obtida. Ela pode ser obtida através de organizações estáticas, onde os problemas de coerência devem ser resolvidos através da comunicação, ou organizações dinâmicas, deixando que a estrutura de controle seja modificada du-

rante o processo de resolução de problemas, adequando-se às novas situações.

As formas de comunicação podem ser vistas a partir de três características: o paradigma de comunicação, o conteúdo da informação transmitida, e os protocolos utilizados para prover efetivamente a comunicação. Há dois paradigmas básicos que os agentes de um sistema podem utilizar para comunicação: a memória compartilhada e a transmissão de mensagens. O modelo de memória compartilhada mais comumente utilizado é o modelo *blackboard*. No caso da transmissão de mensagens, o *conteúdo* das mensagens transmitidas pode ser avaliado por sua correção, relevância, temporariedade e completeza. A correção refere-se à exatidão da informação ; a relevância, à consistência da informação em relação ao que já é conhecido ; a temporariedade refere-se a quanto o destinatário será afetado pela mensagem ; e a completeza indica o quanto da informação é enviado. Por fim, existem inúmeros *protocolos* de comunicação que podem ser utilizados: a comunicação pode ser feita, por exemplo, através de protocolos-padrão de redes, transações, sincronização, seletivamente (mensagens para destinatários específicos), sob demanda (somente com requisição de algum outro agente), e *broadcast* (para todos os agentes).

2.4 Abordagens

As abordagens propostas na área de IAD definem os vários caminhos e possibilidades de estudo, incluindo seus tópicos, desenvolvimentos e classes de aplicação que podem ser exploradas. Refletem, assim, o estado da arte na área, a partir do qual pode-se introduzir novos conhecimentos, ou desenvolver novos métodos, técnicas ou abordagens. A seguir serão apresentadas as abordagens mais significativas propostas para a inteligência artificial distribuída, e alguns dos principais sistemas desenvolvidos a partir delas.

2.4.1 Sistemas *Blackboard*

Blackboard é um paradigma de resolução de problemas em inteligência artificial distribuída capaz de utilizar o conhecimento de vários agentes, denominados *fontes de conhecimento*, para solucionar um problema comum.

Na arquitetura *blackboard*, ilustrada pela Figura 2.2, diversas fontes de conhecimento (FC) podem cooperar na formação e modificação do estado da solução do problema. O processo de formação da solução é incremental, sendo que ela e as estratégias para a sua formação podem ser avaliadas, trocadas ou abandonadas a qualquer momento. Além disso, as fontes de conhecimento contribuem oportunisticamente para a resolução do problema, pois elas têm acesso permanente ao estado da solução, podendo alterá-lo no momento mais oportuno, isto é, quando sua ação tiver maior impacto na temporariedade e completeza do estado da solução.

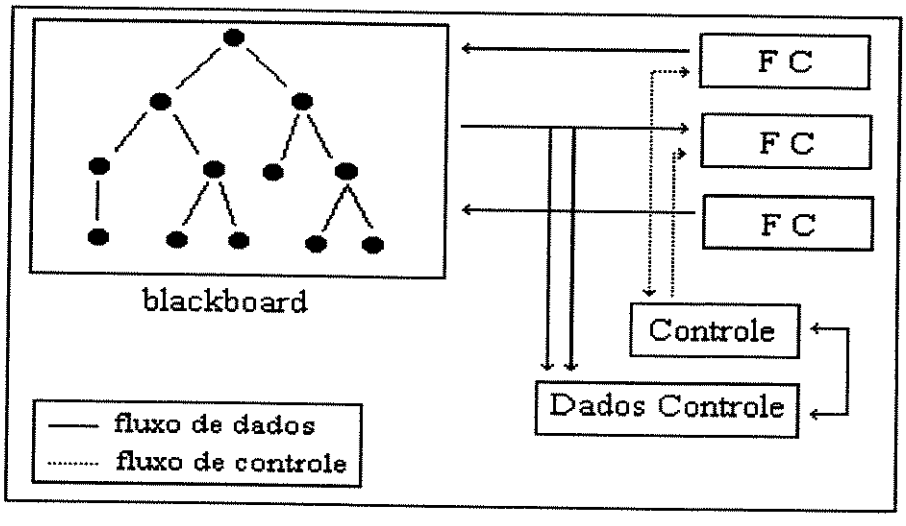


Figura 2.2: Exemplo de arquitetura *blackboard*

O modelo *blackboard* pode ser visto como uma evolução dos sistemas especialistas clássicos onde o conhecimento é segmentado em módulos, e a cada módulo é associada uma máquina de inferência. A comunicação entre os módulos se dá através de uma memória de trabalho comum, denominada *blackboard*.

Este modelo é composto pelas fontes de conhecimento e pelo *blackboard* propriamente dito. Desta forma, não há qualquer componente de controle nele especificado [20]. O modelo apenas determina a organização do domínio do conhecimento, as entradas e as soluções parciais necessárias para a solução do problema. O conhecimento a ser aplicado é baseado no estado da solução. As fontes de conhecimento atuam quando puderem contribuir em sua solução, dinamicamente.

O domínio do conhecimento necessário para a solução de um problema é particionado entre as fontes de conhecimento, que podem modificar o *blackboard* e as estruturas de controle (que também poderão estar no *blackboard*). O *blackboard* só é modificado pelas fontes de conhecimento, e todas as modificações são explícitas e visíveis. Na maioria dos sistemas, as fontes de conhecimento são auto-ativadas. Neste caso, elas devem conhecer as condições nas quais poderão contribuir para a solução do problema. Outra possibilidade é não haver pré-condições para a ativação, como no sistema *CRYSALIS* [104]. Neste caso, os módulos de controle são representados por fontes de conhecimento, que executam uma seleção para a ativação das fontes. Elas podem ainda ser representadas por conjuntos de regras de produção, como no sistema *AGE* [84] ; ou por uma única regra, como no sistema *SACAP* [65].

O *blackboard* é uma estrutura de dados que consiste de objetos (elementos) do espaço de soluções. Seu objetivo é servir como um meio através do qual as fontes de conhecimento interagem e conduzem à formação incremental de uma solução. Constitui-se de dados de entrada, soluções parciais, alternativas, soluções finais, e opcionalmente, dados de controle. Tradicionalmente é organizado de forma hierárquica.

Para monitorar e disciplinar o acesso paralelo ao *blackboard*, que é comum às diversas fontes de conhecimento, foi introduzida uma estrutura de controle. Essa entidade é responsável por gerar as decisões estratégicas que irão conduzir à solução do problema. Muitas vezes, pode resumir-se a um escalonador síncrono, ou ser composto por uma coleção de fontes de conhecimento, como no sistema *CRYSALIS*. O controle (ou por vezes, uma fonte de conhecimento) é responsável por reconhecer quando o processo de

resolução chegou ao final: uma solução satisfatória foi encontrada, ou o sistema pára por falta de conhecimento ou dados.

O sistema *blackboard HEARSAY-II* [21, 22, 62], da década de 1970, tinha o intuito de reconhecer e produzir sons da fala humana. O problema apresentava duas dificuldades principais: a falta de exatidão nos dados de entrada, isto é, o desvio entre a frase que se pretendeu dizer e a recebida pelo sistema, e a imprecisão das regras de compreensão da fala. O *blackboard* foi estruturado hierarquicamente, sendo que suas construções variavam das frases e palavras aos fonemas que as formariam. Além disso, armazenavam-se no *blackboard* diferentes hipóteses para as falas, e assim, as soluções alternativas encontradas. As fontes de conhecimento foram implementadas através de regras de produção, enquanto que o controle era efetuado através de um monitor que acompanhava as alterações feitas no *blackboard*, e um escalonador que calculava prioridades para as atividades. O sistema *HEARSAY-II* contava com três tipos de estratégias para gerar as soluções parciais: *top-down*, *bottom-up*, e teste de hipóteses. Segundo essa última estratégia, uma fonte de conhecimento testava as hipóteses geradas por outra fonte de conhecimento. Havia então um confronto entre essas hipóteses com elementos de uma base de conhecimento, determinando se as hipóteses em questão eram ou não aceitáveis para aquela situação. Com isso, pretendia-se evitar a explosão combinatória das hipóteses, que poderia ser gerada pelas estratégias *top-down* e *bottom-up*.

Outro sistema importante foi o sistema HASP [85]. Seu objetivo era o de identificar a presença de submarinos e navios em uma certa área do oceano, através de hidrofones colocados na área de observação e conhecimento sobre embarcações amigas, inimigas e comerciais. Além das estratégias *top-down* e *bottom-up*, foi utilizada uma estratégia *dirigida a ilhas* (de solução). Segundo essa estratégia, as soluções parciais mais seguras eram exploradas, dando origem a outras soluções seguras e mais complexas.

Dentre os muitos sistemas *blackboard* propostos, podemos destacar ainda o CHRYSALIS [104], cujo objetivo era identificar a estrutura molecular de proteínas. Esse sistema foi importante por introduzir duas estruturas de *blackboard* ao invés de uma

como nos projetos anteriores. O sistema AGE (*Attempt to Generalize*) [84] foi uma tentativa de produzir uma ferramenta para auxiliar no desenvolvimento de outros sistemas *blackboard*, assim como o HEARSAY-III [23], o BB1 [38] e o GBB (*Generic Blackboard*) [12], que o seguiram. Baseados nos sistemas anteriores, surgiram sistemas orientados a objetos, como o BLOBS [119], concorrentes como o POLIGON [89], e variações, como o ASTUTO [66, 67, 68, 69], que construía suas soluções utilizando um modelo relacional de dados.

2.4.2 Sistemas Funcionalmente Corretos e Cooperativos

A principal característica dos Sistemas Funcionalmente Corretos e Cooperativos (*Functionally Accurate, Cooperative*) - FA/C [61, 58] é que todo seu processo de resolução de problemas se dá na presença de incertezas. Esses sistemas lidam com dois tipos de incertezas: *incertezas nos dados* e *incertezas no controle*. As incertezas nos dados são relacionadas à correção, à completeza e à relevância dos resultados de cada agente em relação aos dados recebidos dos outros agentes. As incertezas no controle são relacionadas à escolha da tarefa mais apropriada para execução, dentre todas as possíveis alternativas. Para resolver inconsistências e obter soluções satisfatórias, a resolução de erros nesta abordagem é parte integrante do processo de resolução de problemas e, para isto, os agentes devem trocar esboços de resultados parciais.

O objetivo desta abordagem é que os agentes cooperem entre si, mesmo possuindo informações limitadas e inconsistentes sobre as atividades dos outros agentes, como critérios diferentes para a escolha de atividades a realizar, conhecimento conflitante sobre estratégias gerais de resolução do problema, e eventualmente, erros de *hardware* e *software*.

Este modelo foi desenvolvido visando limitar a comunicação necessária para suportar certas aplicações computacionais distribuídas, como no reconhecimento de aviões através de uma rede de sonares. O modelo seria útil para aplicações onde não fosse possível decompor o problema em uma série de subproblemas, com distribuição ade-

quada da informação, conhecimento, processamento e capacidade de comunicação entre os agentes. O principal desafio para esta abordagem era como estruturar as interações cooperativas entre os agentes, minimizando os custos de comunicação, e ainda assim gerar respostas aceitáveis em tempos razoáveis. Assim, as aplicações deveriam ser estruturadas da seguinte forma:

- Os agentes deveriam resolver subproblemas interdependentes, de grande granularidade;
- Os agentes deveriam ser capazes de gerar soluções parciais de alto nível, apesar da presença de informações incertas e incompletas;
- Os agentes deveriam resolver parcialmente as inconsistências e incertezas, baseados nas soluções parciais recebidas dos outros agentes.

Essa forma de resolução deveria produzir resultados parciais cada vez mais completos, resolvendo as incertezas e comparando as soluções alternativas. Desta forma, os erros introduzidos por informações incompletas, inconsistentes ou ultrapassadas, seriam resolvidos através das trocas assíncronas de resultados parciais entre os agentes. Essa ênfase na resolução de inconsistências aproxima muito este trabalho de outros trabalhos de negociação em resolução distribuída de problemas, tais como [9, 52], manutenção distribuída de consistência [14], e sistemas abertos [39], que serão apresentados na Seção 2.4.5.

O processo de resolução cooperativa é baseado em três tipos de interdependências entre as soluções parciais: soluções envolvendo áreas sobrepostas devem ser (ou tornarem-se) consistentes, hipóteses múltiplas que possam estender soluções de outros agentes devem também ser consistentes, e os agentes devem ser capazes de encontrar evidências externas quando suas hipóteses estiverem incertas e interagirem com hipóteses de outros agentes.

Alguns requisitos se mostram necessários para uma operação adequada do modelo:

- Operação assíncrona, onde não se possa assumir que toda informação necessária para completar uma computação esteja disponível sempre que necessário;
- Exploração de restrições de soluções parciais para resolver subproblemas com áreas sobrepostas. Isso pode envolver um processo de inferência potencialmente complexo, podendo englobar o uso de conhecimento sobre os agentes e sobre a comunicação realizada;
- Representação e raciocínio sobre incertezas nas soluções parciais. Isso pode envolver a manipulação de múltiplas soluções parciais alternativas para julgar a probabilidade das soluções parciais estarem corretas;
- Os agentes devem ser capazes de comunicar entre si, reconhecendo informações redundantes ou ultrapassadas, e serem capazes de distribuir adequadamente a carga de processamento entre eles de forma a minimizar sua ociosidade. Entretanto, algoritmos para controle cooperativo são muito complexos, porque os agentes têm visões incompletas, ultrapassadas ou incertas dos seus similares.

Os experimentos de validação para o modelo envolvem aplicações na área de interpretação distribuída, utilizando arquiteturas *blackboard*. A primeira aplicação importante envolveu uma implementação distribuída do sistema Hearsay . Haviam três agentes que eram sistemas Hearsay completos, e que recebiam continuamente dados acústicos. Eles trocavam soluções parciais envolvendo áreas sobrepostas, de forma a terem sempre soluções alternativas. Os primeiros experimentos mostraram que o sistema obtinha o mesmo desempenho da versão centralizada do sistema, mas os agentes não conseguiam comportamento coerente: os agentes transmitiam informações não relevantes ou em instantes inadequados, o que "distraia" os outros agentes. Esses problemas foram atribuídos aos agentes, por serem excessivamente auto-dirigidos, isto é, não haverem formas explícitas de cooperação.

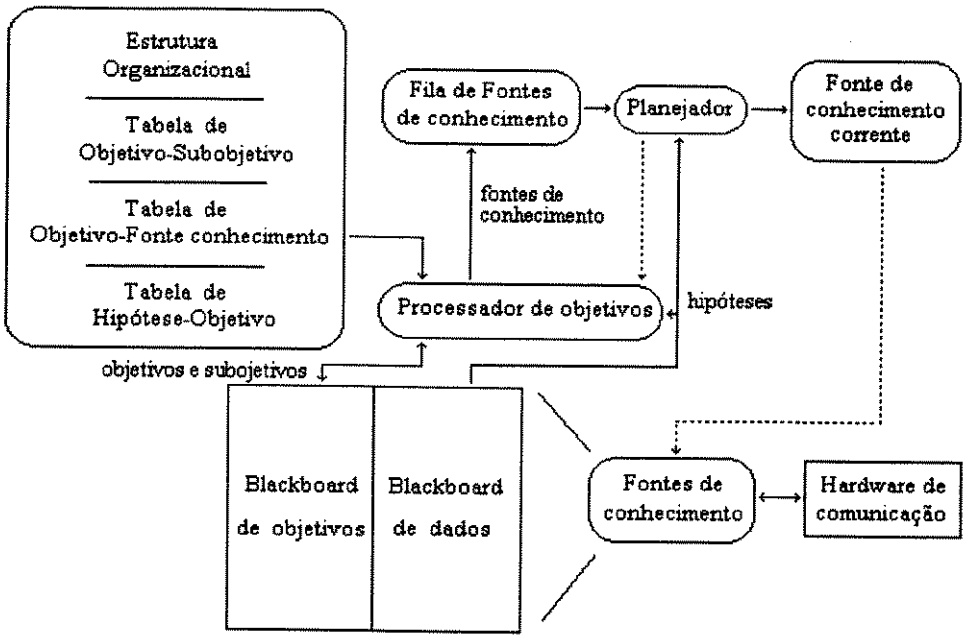


Figura 2.3: Arquitetura do sistema DVMT

A segunda aplicação importante foi com o sistema DVMT *(Distributed Vehicle Monitoring Testbed)* [60], ilustrado pela Figura 2.3. Esse sistema tornou-se o principal ambiente para as experiências realizadas com o modelo. Podem ser identificadas três fases na evolução dos mecanismos de controle. Na primeira fase, buscou-se a integração do controle dirigido a dados e a objetivos. Os objetivos foram representados explicitamente, e assim que os dados eram alterados, gerava-se um conjunto de objetivos que representava os resultados potenciais que poderiam ser obtidos a partir das novas informações. Os objetivos eram representados em cada agente através de uma estrutura de objetivos parciais, que representavam aqueles a serem alcançados a curto prazo, e eram transmitidos para outros agentes. Os agentes modificavam dinamicamente suas representações de objetivos, buscando obter maior coerência em suas atividades. Para a segunda fase, tentou-se eliminar as incertezas nas decisões de controle. No entanto, isso implicava que as decisões deveriam ser constantemente reavaliadas, devido às novas informações geradas durante o processo de resolução. Os custos para eliminar essas incertezas seriam altos demais, especialmente para sistemas grandes ou complexos. Optou-se então por uma estratégia com dois níveis de decisão: o primeiro, organiza-

cional, referindo-se às decisões que não necessitariam ser constantemente reavaliadas para obter desempenho razoável ; e o segundo, a nível de agente, referindo-se às decisões táticas, que deveriam ser constantemente reavaliadas. Para atingir um objetivo, os agentes negociavam com outros agentes conflitantes e que também poderiam atingir aquele objetivo, tentando resolver os seus conflitos. Na terceira fase, utilizou-se de conhecimento organizacional para melhorar a cooperação entre os agentes. Esse conhecimento era utilizado para definir estratégias de cooperação e organização, tal como organizar os agentes hierarquicamente. Desenvolveu-se uma técnica denominada *partial global planning* [17, 18], onde cada agente construía e mantinha uma visão de seus objetivos de curto prazo. Cada agente selecionava uma ordem para atingir seus objetivos, e estimava quanto tempo levaria para atingi-los, além da qualidade esperada para os resultados.

A abordagem evoluiu consideravelmente desde a sua proposta inicial de 1981, embora muitas questões ainda permaneçam abertas. Dentre estas questões, destaca-se a manipulação de informações para controle em tempo-real, crucial para sua aplicação prática.

2.4.3 Redes de Contrato

O protocolo de Redes de Contrato (*Contract Nets*) [99, 100] é um dos que mais influenciaram a inteligência artificial distribuída. Segundo este modelo, os agentes utilizam o protocolo para firmar contratos que estabelecem a alocação de tarefas no sistema. A contratação envolve trocas de informações entre as partes envolvidas, uma avaliação da informação por cada agente, baseada em sua própria perspectiva, e um acordo. Segundo esse protocolo, as partes discordantes podem retirar-se da negociação, ao contrário dos protocolos de votação, onde todos devem sujeitar-se às decisões da maioria.

Segundo o protocolo de Redes de Contrato, os agentes coordenam suas atividades através de contratos, para atingir seus objetivos. Os contratos são firmados segundo a estratégia *top-down*. Em cada estágio, um agente gerente decompõe seus contratos

em subcontratos, que devem ser cumpridos por agentes empreiteiros. Esse processo envolve um protocolo onde são realizadas ofertas para estabelecer quais agentes irão realizar tais subcontratos. Esse processo de decomposição continua até que os nós possam concluir seus contratos sem assistência. Como resultado, obtém-se uma rede de relações de controle entre gerentes e empreiteiros, distribuída pela rede.

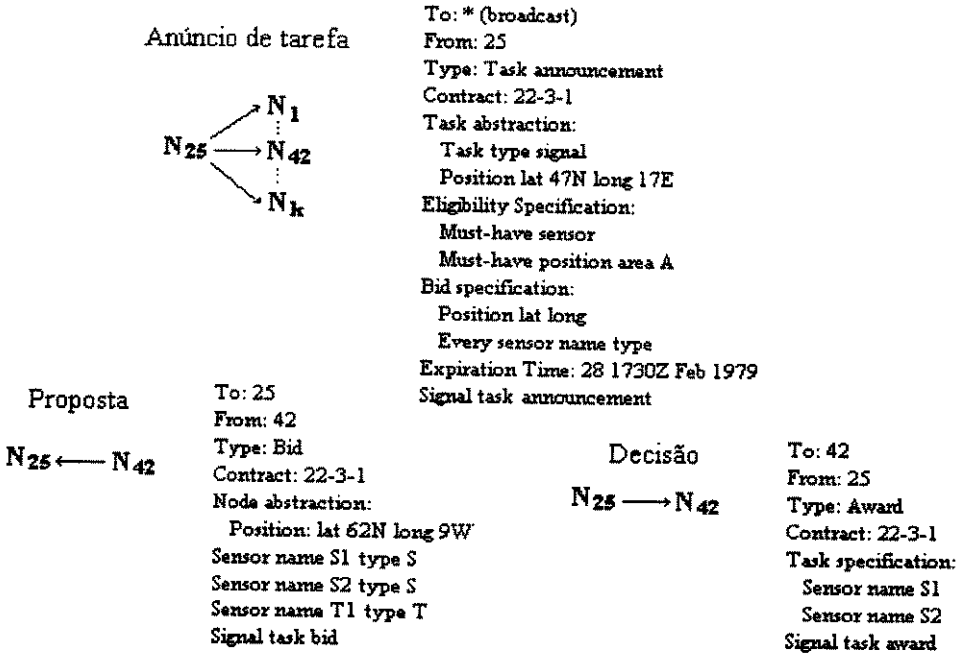


Figura 2.4: Exemplo de trocas de mensagens nas Redes de Contrato

Antes que uma contratação possa começar, um gerente deve identificar uma tarefa a ser alocada. Tipicamente, um gerente, ao receber uma tarefa complexa, a decompõe em pequenas tarefas de forma pré-determinada. A seguir, uma tarefa é anunciada na rede, e os nós que estiverem ociosos no momento recebem e avaliam o anúncio. Os nós com recursos, habilidades e conhecimento apropriados, enviam propostas ao gerente, indicando seu grau de adequação para realizar a tarefa. Após algum tempo, o gerente avalia as propostas recebidas e designa o nó que deverá realizá-la. Finalmente, o gerente e o empreiteiro trocam informações sobre a tarefa e o seu progresso até que o empreiteiro termine e envie os resultados da tarefa ao gerente. Um exemplo de trocas de mensagens realizadas para o anúncio, proposta e decisão do nó que irá realizar uma tarefa são mostradas pela Figura 2.4.

O protocolo de Redes de Contrato provê formatos padronizados para as mensagens e uma comunicação estruturada, isto é, os nós conhecem a ordem das mensagens que devem trocar para gerar os contratos. Contudo, um gerente deve utilizar conhecimento sobre uma determinada aplicação para decidir quais tarefas contratar, como descrever uma tarefa a ser anunciada, para onde enviar os anúncios, e como escolher os melhores empreiteiros. Os empreiteiros devem ser capazes de interpretar as exigências da tarefa a partir de seu anúncio, e de decidir se, e como vai enviar uma proposta para a tarefa. O protocolo não inclui o conhecimento dependente da aplicação necessário para tomar essas decisões, provendo apenas a linguagem a ser utilizada pelos nós para trocar informações.

2.4.4 Atores e Sistemas Abertos

O modelo de Atores foi desenvolvido com o intuito de encontrar uma linguagem com invocação de primitivas dirigida a padrões [41]. Em uma linguagem dirigida a padrões, um padrão representa o que se está buscando (estabelecer ou utilizar), e atua como uma notificação do processamento a ser realizado. As primeiras pesquisas consistiram em identificar primitivas para os atores, e especificar leis que poderiam ser válidas para as suas computações [40]. Essas leis estabelecem quais eventos podem ocorrer e como eles se relacionam. A computação no modelo é realizada por atores que enviam mensagens uns para os outros.

Para enviar uma mensagem a um ator, é necessário conhecer o seu *handle*, que é uma espécie de endereço lógico. Uma vez conhecido esse endereço, será sempre possível comunicar-se com aquele ator, mesmo que ele mude de endereço físico. Quando um ator recebe uma mensagem, suas ações são determinadas segundo o modelo de comportamento que estiver sendo utilizado por ele, podendo tomar decisões locais, criar novos atores, enviar mensagens ou alterar seu modelo de comportamento, conforme ilustra a Figura 2.5.

O modelo de atores constituiu uma base para a proposta dos *sistemas abertos*

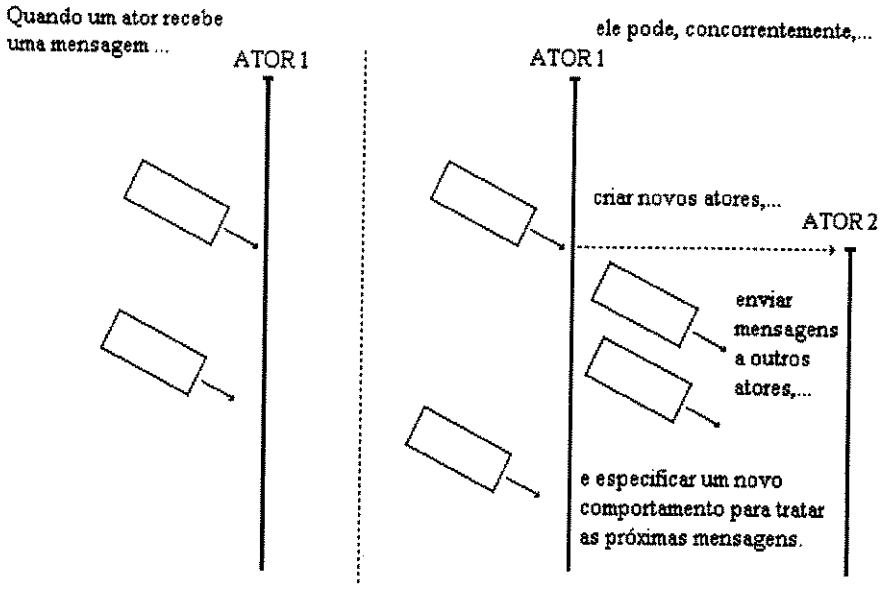


Figura 2.5: Respostas possíveis a uma mensagem

de computação distribuída ultra-concorrente [39, 41], que serviriam para as próximas gerações de hardware concorrente. Nesses sistemas, a informação dos diferentes atores pode ser inconsistente, incompleta ou incorreta. Eles estão sujeitos a resultados indeterminados de suas operações devido à concorrência e à possibilidade de chegada de novas informações a qualquer instante. Além disso, lidam com grandes quantidades de informação e, para tratá-las, devem explorar massivamente a concorrência. O modelo de atores é caracterizado pelo indeterminismo, assincronismo, controle descentralizado, presença de informações inconsistentes e operação contínua.

2.4.5 Outras Abordagens

Várias outras abordagens foram propostas para a inteligência artificial distribuída. Algumas delas obtiveram destaque por serem extensões ou variações das anteriormente apresentadas, enquanto outras, por apresentarem idéias ou conceitos novos.

- A Negociação Multi-Estágios [8, 9] é uma extensão do protocolo de Redes de Contrato, onde se propõe uma outra forma de negociação para alocação de tare-

fas. Considera-se uma classe de problemas de alocação de tarefas, denominada problema de satisfação distribuída de restrições. Nesses problemas, existe um conjunto coordenado de ações para atingir os objetivos da rede, mas cada nó possui recursos limitados e insuficientes para atingi-los sozinho. A combinação das restrições locais quanto aos recursos, e a necessidade de coordenação entre os nós dá origem a um conjunto complexo de restrições globais interdependentes. Os nós tentam determinar ações locais para alocar os recursos da rede, e trocam essas informações iterativamente. A cada iteração, os nós identificam se as suas ações escolhidas violariam as intenções dos outros nós. Os nós devem trocar informações suficientes para obter uma configuração de tarefas que satisfaça suas restrições. Um outro modelo de negociação multi-estágios é proposto em [101, 102]. Neste caso, utiliza-se um modelo de negociação fundamentado no raciocínio baseado em casos para dirigir o processo de negociação. Armazenam-se as negociações passadas para que sejam aplicadas nos casos futuros. Caso essa estratégia não funcione, aplica-se uma análise de preferência multi-atributos para a escolha da decisão mais aceitável para os outros agentes. Os agentes revisam dinamicamente os seus objetivos, armazenando um histórico com as decisões passadas.

- No modelo de Negociação para Controle de Tráfego Aéreo [6], o objetivo é o de permitir que cada nó (aeronave) construa um plano de vôo que o mantenha a distâncias apropriadas das outras aeronaves, além de satisfazer outras restrições de vôo (como dirigir-se ao local desejado com consumo mínimo de combustível). Destaca-se no modelo a política de centralização de tarefas. Segundo essa política, quando houverem aeronaves em situações potencialmente conflitantes, uma delas será escolhida para solucionar o conflito. Essa aeronave deve agir então como um planejador centralizado que especifica as ações concorrentes de todas as aeronaves. A presença da distribuição está no estabelecimento da aeronave que irá realizar o planejamento. Os resultados obtidos indicam que normalmente a melhor estratégia é escolher o nó com mais conhecimento, exceto em situações muito complexas, quando a melhor alternativa seria escolher a aeronave com maior flexibilidade para manobras.

- No Planejamento Multi-Agentes, os nós constróem um plano que especifica suas ações e interações futuras. O plano é construído através de trocas de informação através da rede de comunicação. Um ou mais agentes retêm a posse desse plano, que indica as ações e interações de todos os nós, por todo o período de atividade do sistema. Os agentes que detêm o plano identificam e evitam possíveis conflitos e duplicações de esforço, antes que eles possam ocorrer. Além disso, o plano especifica exatamente que ações os nós devem tomar e quando. Existem abordagens centralizadas e distribuídas. Em [30], propõe-se uma estratégia onde inicialmente são estabelecidos os planos individuais dos nós. Um nó centralizador coleta e analisa os planos individuais a fim de identificar possíveis interações e conflitos envolvendo recursos limitados. Ele também estabelece regiões críticas, ou seja, seqüências de ações que levam a conflitos, e insere comandos de comunicação nos planos, de forma que os nós possam sincronizar suas ações apropriadamente, evitando os conflitos. Essa estratégia é similar à política de centralização de tarefas do modelo de Negociação para Controle de Tráfego Aéreo. Nas abordagens distribuídas, a idéia básica é prover a cada agente os modelos dos planos dos outros agentes. Um planejador distribuído hierárquico é apresentado em [11], onde o planejamento se dá por níveis. Os nós constróem modelos dos outros nós, e constróem também planos locais para cada nível de detalhe, tentando resolver os conflitos. Uma outra abordagem distribuída permite que os nós intercalem planejamento e ações, ao invés de terem que estabelecer um plano completo antes de executar as ações. Nessa abordagem, chamada de *partial global planning* [18, 17], os nós tentam obter coordenação a partir de sua visão atual (e parcial) do sistema. Cada nó constrói um plano de curto prazo contendo os objetivos que espera atingir, além da qualidade da solução e dos prazos esperados para atingi-los. Os agentes trocam esses planos (também chamados de visões) e os combinam, obtendo planos de maior escopo e maior confiabilidade. Os planos são então delegados aos agentes através do modelo de Redes de Contrato.

- A estrutura organizacional reflete o padrão da informação e dos relacionamentos de controle, e a distribuição do conhecimento e habilidades entre os nós. Nos

trabalhos com os Sistemas Funcionalmente Corretos e Cooperativos, defende-se que não é realístico desenvolver políticas suficientemente flexíveis e eficientes, com comunicação restrita, e ao mesmo tempo tomar todas as decisões para cada nó da rede. Os nós devem decidir sobre suas próprias atividades, baseados em sua visão atual do sistema, utilizando conhecimento organizacional sobre o seu papel e o papel dos outros nós na rede para orientar suas decisões. As Redes de Contrato também provêem formas de estruturação, onde os nós organizam-se na forma gerente-empreiteiro para a resolução de problemas. A Metáfora da Comunidade Científica [56] apresenta um modelo onde os nós depositam questões (objetivos) e respostas (resultados) em um arquivo acessível a todos. Tenta-se organizar o sistema analogamente à estrutura da comunidade científica. A presença e o acesso às informações depositadas no arquivo permitem que vários nós processem questões já tratadas por outros nós, buscando novas e melhores soluções.

2.5 Resumo

A inteligência artificial distribuída é uma área vasta e complexa, compreendendo diversas formas de organização e interação entre agentes computacionais. A grande abrangência da IAD traz, *per si*, grande complexidade aos trabalhos da área, que por vezes são desenvolvidos ao longo de décadas, sem que consigam envolver toda as complexidades de um modelo de organização e interação entre entidades distintas.

Nesse capítulo, foram apresentados os conceitos básicos de IAD com uma revisão de sistemas da área, os quais, embora não contemplem toda a complexidade possível, provêem mecanismos para estudar sistematicamente os trabalhos existentes. Foram também apresentadas várias abordagens para sistemas de IAD, proporcionando uma idéia da abrangência do tema, da complexidade envolvida e do universo ainda a ser estudado.

No próximo capítulo será apresentada, sucintamente, uma introdução às outras áreas de conhecimento que compõem o presente trabalho.

Capítulo 3

Fundamentação Teórica

3.1 Introdução

Neste capítulo são apresentados os conceitos básicos da teoria dos conjuntos nebulosos, dos sistemas distribuídos, dos sistemas baseados em casos, e dos algoritmos genéticos. Não se pretende aqui apresentar uma revisão completa e exaustiva, mas introduzir de forma resumida seus aspectos mais importantes, utilizados na seqüência deste trabalho.

3.2 Teoria dos Conjuntos Nebulosos

A teoria dos conjuntos nebulosos teve início em 1965 com trabalhos de Lotfi Zadeh [112, 114], tendo sido posteriormente desenvolvida por vários pesquisadores. A teoria dos conjuntos nebulosos oferece uma alternativa formal para a representação do raciocínio humano, que envolve imprecisões e incertezas. Através da lógica convencional, acostumamo-nos a classificar os objetos através de categorias pré-determinadas,

induzindo à dicotomia do tipo falso ou verdadeiro, par ou ímpar, certo ou errado. Isso certamente não é suficiente para representar a forma como pensamos ou tomamos decisões. Na verdade, o pensamento humano, segundo [48], nem sempre é binário e talvez raramente o seja. Muitas vezes encontramos classes de objetos que não são definidas precisamente, como a dos homens altos, dos carros econômicos, das matrizes esparsas, dentre muitas outras. Tais conceitos, apesar de imprecisos, têm um significado óbvio considerando-se um determinado contexto; eles são utilizados em situações reais para o raciocínio, o planejamento e a tomada de decisões. Na teoria dos conjuntos nebulosos, a noção básica de conjunto é modificada, permitindo que os graus de pertinência ao conjunto tomem valores em um conjunto M [49].

Definição 1 *Sejam* E *um conjunto, enumerado ou não, e seja* x *um elemento de* E. *Então um conjunto nebuloso* A *de* E *é um conjunto de pares ordenados*

$$(x, \mu_A(x)), \forall x \in E,$$

onde $\mu_A(x) : E \rightarrow M$ *é uma função de pertinência (ou característica), com valores em um conjunto ordenado* M, *e indica o grau ou índice de pertinência de* x *em* A. M *é chamado de conjunto de pertinência.*

Se M=$\{0,1\}$, o conjunto nebuloso A é entendido como um conjunto convencional. As funções μ_A são, neste caso, funções com valores binários; $\mu_A(x) : E \rightarrow \{0, 1\}$. Freqüentemente M é considerado como o intervalo unitário.

Assim, a noção de conjunto nebuloso é ligada à noção clássica de conjunto, permitindo o estudo de conceitos imprecisos usando estruturas matemáticas precisas.

Muitas das propriedades das operações com conjuntos clássicos são também satisfeitas pelos conjuntos nebulosos, como as da comutatividade, associatividade, distributividade e idempotência.

Um tópico de grande interesse na teoria dos subconjuntos nebulosos é a seleção de operadores apropriados para a inclusão, interseção, união e complemento de subconjuntos, que representem uma generalização para o ambiente nebuloso dos operadores de conjuntos clássicos. Essas operações foram originariamente definidas como:

- **Inclusão.** Sejam E um conjunto e M seu conjunto de pertinência associado, e sejam A e B dois subconjuntos nebulosos de E. Diz-se que A *está contido em* B se

$$\forall x \in E : \mu_A(x) \leq \mu_B(x)$$

sendo denotado por

$$A \subseteq B.$$

- **Igualdade.** Sejam E um conjunto, M um conjunto de pertinência, e sejam A e B dois subconjuntos nebulosos de E. Diz-se que A e B são iguais se e somente se

$$\forall x \in E : \mu_A(x) = \mu_B(x)$$

sendo denotado por

$$A = B.$$

Se existir pelo menos um x tal que a igualdade $\mu_A(x) = \mu_B(x)$não for satisfeita, diz-se que A e B não são iguais, sendo denotado por

$$A \neq B.$$

- **Complementação.** Sejam E um conjunto, M=[0,1] um conjunto de pertinência, e sejam A e B dois subconjuntos nebulosos de E. Diz-se que A e B são complementares se

$$\forall x \in E : \mu_B(x) = 1 - \mu_A(x)$$

sendo denotado por

$$B = \overline{A} \text{ ou } A = \overline{B}.$$

Obviamente, temos que

$$\overline{\overline{A}} = A.$$

- **Interseção.** Sejam E um conjunto, M um conjunto de pertinência, e sejam A e B dois subconjuntos nebulosos de E. Define-se a interseção

$$A \cap B$$

como o maior subconjunto nebuloso contido em A e B ao mesmo tempo. Isto é,

$$\forall x \in E : \mu_{A \cap B} = MIN(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x).$$

- **União.** Sejam E um conjunto, M=[0,1] um conjunto de pertinência, e sejam A e B dois subconjuntos nebulosos de E. Define-se a união

$$A \cup B$$

como o menor subconjunto nebuloso contido em A e B ao mesmo tempo. Isto é,

$$\forall x \in E : \mu_{A \cup B} = MAX(\mu_A(x), \mu_B(x))$$

$$\text{ou } \forall x \in E : \mu_{A \cup B} = \mu_A(x) \vee \mu_B(x).$$

Para a determinação de outros operadores adequados para essas operações, além dos convencionais(min, max e complemento de um), foram definidas generalizações dos operadores de união e interseção, onde se destacam as T-normas e T-conormas [92].

Definição 2 *Um operador $N : [0,1] \mapsto [0,1]$ é chamado um operador de negação se*

i) $N(1) = 0$ *e* $N(0) = 1$

ii) é involutivo

$N(N(a)) = a$

iii) é de ordem reversa

se $a > b$ *então* $N(a) \leq N(b)$

É importante observar a diferença entre o operador de negação e de antônimo de um subconjunto nebuloso. O antônimo relaciona conceitos como alto-baixo, quente-frio, grande-pequeno, enquanto que a negação relaciona conceitos como alto-não_alto, baixo-não_baixo, quente-não_quente. Se A é um subconjunto nebuloso definido sobre o conjunto base X, e X é um conjunto ordenado

$X = \{x_1, x_2, ..., x_n\}$

onde $x_j > x_i$ quando $j > i$ então o antônimo $\hat{A}$ pode ser definido [109] por

$A(x_j) = A(x_{n-j+1})$.

3.2.1 Elementos da lógica nebulosa

A lógica nebulosa [113, 116] surgiu com o principal objetivo de modelar o raciocínio e a tomada de decisões racionais em ambientes com incertezas e imprecisões.

A lógica nebulosa pode ser vista como uma extensão da lógica multivalorada, embora manipule modelos aproximados ao invés de modelos precisos. Ela provê métodos para representar o significado de quantificadores nebulosos, como "a maioria", "alguns", "freqüentemente", e predicados modificadores nebulosos e não nebulosos, como

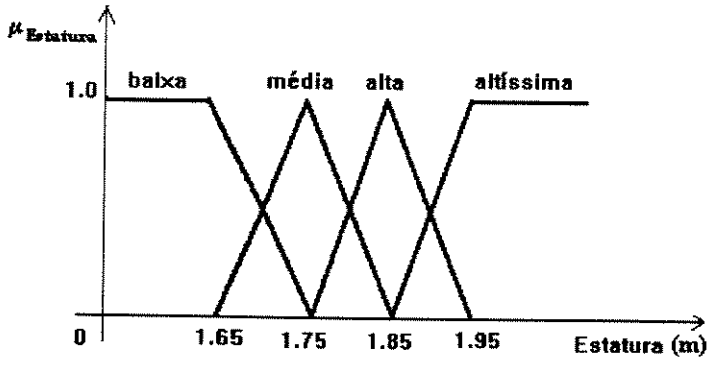


Figura 3.1: Representação da variável lingüística *Estatura*

"não", "muito", "mais ou menos", "um pouco", "extremamente", ... Isso introduz ao conceito de *variáveis lingüísticas*, cujos valores são palavras ou sentenças de uma linguagem natural. Por exemplo, "Estatura" é uma variável lingüística que pode assumir valores como "baixa", "média", "alta", e "altíssima", ilustrada graficamente pela Figura 3.1. Além disso, é possível introduzir qualificadores, como "não é bem verdade", "é quase impossível", e "é improvável".

Assim, as principais características que diferenciam a lógica nebulosa das lógicas tradicionais são [92, 113]:

- Os valores verdade podem ser subconjuntos nebulosos de um conjunto U, usualmente o intervalo [0,1], e denotados por variáveis lingüísticas, como *verdadeiro*, *mais ou menos verdadeiro*, *não muito falso*, etc.
- Os predicados podem ser precisos como na lógica clássica (*mortal, pai de*), ou imprecisos (*grande, muito jovem*).
- Podem existir vários tipos de quantificadores (*a maioria, vários, freqüentemente, cerca de, pelo menos*).
- Podem também ser representados modificadores de predicado (*não, muito, mais ou menos, extremamente*).

- As proposições podem ser quantificadas de três formas diferentes:
  - quantificação de verdade, como em

    (Maria é jovem) não é bem verdade.
  - quantificação de probabilidade, como em

    (Maria é jovem) é pouco provável.
  - quantificação de possibilidade, como em

    (Maria é jovem) é quase impossível.

Uma característica importante da lógica nebulosa é que ela fornece uma base para a teoria de raciocínio aproximado [118, 115, 110], utilizado na representação e raciocínio com informação imprecisa. Por exemplo, uma proposição do tipo:

"Os suecos são loiros"

pode ser interpretada como

"A maioria dos suecos é loira"

ou ainda

geralmente(um sueco é loiro)

Sua idéia básica é que uma proposição **p** em linguagem natural ou sintética pode ser vista como uma coleção de restrições $R_1, R_2, .., R_K$, que restringem os valores de uma coleção de variáveis $X = (X_1, X_2, ..., X_N)$. Em geral, as restrições são implícitas e a representação da semântica de **p** é, em essência, um processo de explicitação das restrições e variáveis de **p**. Na lógica nebulosa, isto é realizado representando **p** em sua *forma canônica.*

p $\rightarrow$ X é A

onde A é um predicado nebuloso ou equivalentemente, uma relação n-ária nebulosa em **U**, onde **U**=U_1xU_2x .. xU_n e $U_i, i = 1, .., n$ é o domínio de **X**.

Por exemplo, considerando a seguinte proposição:

p=João é ALTO

onde o símbolo = deve ser lido como *denota* ou é igual a, por definição. Neste caso X=Altura(João), A=ALTO, e a forma canônica de **p** é

Altura(João) é ALTO

(A relação nebulosa ALTO efetua papel de restrição na forma canônica).

Da forma canônica, segue-se a equivalência de possibilidades:

$Poss\ Altura(João) = u\ = \mu_{ALTO}(u)$

onde μ_{ALTO} é a função de pertinência de ALTO e $\mu_{ALTO}(u)$ é o grau de pertinência de u em ALTO, isto é, o grau que um valor de altura u satisfaz a restrição introduzida pela relação ALTO.

Podemos ter também proposições quantificadas, da forma

p = QA´s são B´s

por exemplo,

p = a maioria dos homens altos não é muito gorda

onde Q é um quantificador nebuloso e A e B são predicados nebulosos. A variável de restrição X é a proporção de B´s em A´s, com Q representando uma restrição elástica em X. Mais especificamente, se U é um conjunto finito $\{u_1, .., u_m\}$, a proporção de B´s em A´s é definida por

$$\sum Count(B/A) = \frac{\sum_j \mu_A(u_j) \wedge \mu_B(u_j)}{\sum_j \mu A(u_j)} j = 1, .., m$$

onde $\mu_A(u_j)$ e $\mu_B(u_j)$ denotam os graus de pertinência de u_j em A e B respectivamente. Na forma canônica, essa mesma expressão pode ser escrita como

$$\Sigma Count(B/A) \text{ é } Q$$

O conceito de uma forma canônica provê meios efetivos para formular problemas de inferência, onde fatos e regras podem ser qualificados probabilisticamente ou expressos em proposições qualificadas [113].

3.2.2 Regras Básicas de Inferência

Uma característica da lógica nebulosa é que as premissas e as conclusões nas regras de inferência são geralmente expressas na forma canônica. Isso evidencia o fato de que cada premissa é uma restrição de uma variável e que a conclusão é uma restrição induzida através do processo de propagação de restrições.

Temos, por exemplo, o princípio de herança:

X é A
A $\subset$ B

X é B

onde X é uma variável que assume valores no universo de discurso U, e A e B são subconjuntos nebulosos de U; e uma variação:

usualmente (X é A)
A $\subset$ B
---
usualmente (X é B)

No caso limite, onde *usualmente* torna-se *sempre*, a variação anterior reduz-se ao princípio de herança.

Outras regras:

- regra conjuntiva

  X é A
  X é B

  X é A ∩ B

- produto cartesiano

  X é A
  Y é B

  (X,Y) é AxB

  onde (X,Y) é uma variável composta e AxB é definida por

  $\mu_{AxB}(u,v) = \mu_A(u) \wedge \mu_B(v), u \in U, v \in V$

- regra de projeção

  (X,Y) é R

  X é ${}_xR$

  onde ${}_xR$ é a projeção da relação binária R no domínio de X, definida por

  $\mu_{{}_xR}(u) = sup_v \mu_R(u,v), u \in U, v \in V$

  onde $\mu_R(u,v)$ é a função de pertinência de R, tomando-se o máximo sobre $v \in V$.

- regra da composição

  X é A
  (X,Y) é R

  Y é $A \circ R$

  onde A ∘ R é a composição da relação unária A, onde a relação binária R é definida por

  $\mu_{A \circ R}(v) = sup_u(\mu_A(u) \wedge \mu_R(u,v))$

- princípio da extensão

  X é A

  $f(X)$ é $f(A)$

  onde a função de pertinência de $f(A)$ é definida por

  $\mu_{f(A)}(v) = sup_u \mu_A(v)$, sujeita à condição $v = f(u), u \in U, v \in V$.

- princípio da extensão disposicional

  usualmente (X é A)
  ---
  usualmente ($f(X)$ é $f(A)$)

  O princípio da extensão disposicional pode ter papel importante na inferência a partir de conhecimento de senso comum [111].

3.3 Sistemas Distribuídos

Um sistema distribuído é um sistema computacional que consiste de múltiplos processadores autônomos, que não compartilham memória primária, mas cooperam através do envio de mensagens por meio de uma rede de comunicação [2]. Cada um dos processadores é denominado *nó* do sistema. Cada nó de um sistema distribuído executa sua própria seqüência de instruções, e utiliza seus próprios dados.

Os sistemas distribuídos podem ser classificados quanto à sua rede de comunicação. A rede determina a velocidade e a confiabilidade da comunicação, além da distribuição espacial dos processadores. Tradicionalmente, denominam-se *sistemas fortemente acoplados* as arquiteturas distribuídas onde a comunicação é rápida e segura, e onde os processadores ficam fisicamente próximos uns dos outros. Denominam-se ainda *sistemas fracamente acoplados* aqueles sistemas com comunicação lenta, eventualmente menos confiável, e com os processadores fisicamente dispersos. Um exemplo de sistema distribuído fracamente acoplado é mostrado pela Figura 3.2.

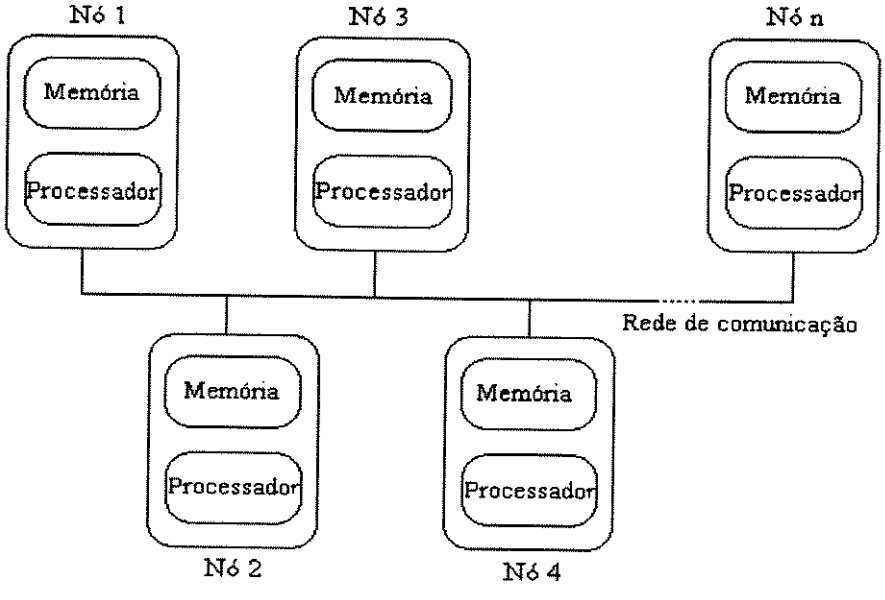


Figura 3.2: Exemplo de sistema distribuído fracamente acoplado

Uma das características que distinguem um sistema distribuído de uma rede de computadores tradicional, que também é constituída por um conjunto de processadores interligados por serviços de comunicação, é sua transparência [105]. Para ter acesso a um recurso ou serviço em uma rede de computadores, é necessário explicitar sua localização física, enquanto que em um sistema distribuído, essa localização deve ser realizada de forma transparente para o usuário. Os graus de transparência incluem acesso, localização, nome, controle, dados, execução e desempenho [33].

Tratar logicamente um conjunto de computadores interligados por uma rede de comunicação como um sistema de computação distribuído único e integrado, pode trazer uma série de benefícios que, juntos, facilitam o suporte a aplicações mais complexas. Por exemplo: [13, 57, 96]

- **Melhor desempenho.** Pode-se executar vários programas ou partes diferentes de um mesmo programa em diferentes processadores, em paralelo;
- **Segurança e tolerância a falhas.** A falha de um subconjunto dos componentes do sistema não causa necessariamente a interrupção de suas atividades, tornando-o mais seguro e confiável;
- **Uso de especialização funcional.** Pode-se utilizar de processadores dedicados a tarefas específicas do sistema, provendo maior velocidade e alta confiabilidade;
- **Compartilhamento de recursos.** Recursos disponíveis a um processador, como periféricos, podem também ser utilizados por outros processadores;
- **Acesso a dados geograficamente dispersos.** Permite, por exemplo, particionar as informações entre os diversos nós;
- **Expansibilidade.** Pode-se adicionar novos recursos ao sistema, de acordo com sua demanda por serviços e desempenho, sem que sejam necessárias mudanças significativas no sistema corrente.

Como desvantagens, temos, por exemplo, que os sistemas distribuídos normalmente requerem software mais complexo e são dependentes da capacidade e confiabilidade da rede [103].

A área de sistemas distribuídos pode ser dividida basicamente em duas sub-áreas: aquela envolvendo os sistemas propriamente ditos, e outra envolvendo aplicações. A primeira delas pode ainda ser subdividida em *hardware* e *software* [96].

A área de *hardware* para sistemas distribuídos inclui as redes (físicas) de computadores e engenharia de protocolos de comunicação, e *hardware* para permitir tolerância a falhas. Os mecanismos de tolerância a falhas envolvem normalmente redundância de recursos de *hardware*, além de *software* para fazer com que o sistema retorne até o último estado consistente possível. Os projetos de protocolos de comunicação têm impacto direto na eficiência e segurança do sistema, sendo que não incluem somente velocidades de comunicação do meio físico, mas também tipos de protocolos (estocásticos ou determinísticos), de topologias (em malha, anel, barramento, árvore), e as próprias ligações físicas.

Na área de *software* e algoritmos distribuídos, algumas questões caracterizam a distribuição e abrangência da área: exclusão mútua distribuída, consenso, escalonamento, tratamento de *deadlocks*, relógios lógicos e recuperação de erros, dentre outros. A sincronização de relógios lógicos, por exemplo, tenta compensar a ausência de um relógio único para o sistema. O escalonamento distribuído de processos utiliza técnicas para distribuir a carga de processamento e aumentar o desempenho do sistema. Isso pode ser realizado, por exemplo, através do particionamento e alocação das tarefas entre diversos nós, tentando minimizar a comunicação inter-nós necessária.

Na área de aplicações, é importante notar a presença de alguns modelos e decisões que balizam o seu desenvolvimento, como por exemplo, a a granularidade dos processos (que se refere ao número médio de ciclos de *CPU* para executar um processo) e a comunicação interprocessos. Quanto à granularidade, existe todo um espectro que vai desde os chamados processos de grão pequeno, ou fino, aos processos de grão grande, ou

grosso. A comunicação interprocessos pode ser realizada através de vários mecanismos como trocas de mensagens, caixas-postais, *rendez-vous*, e compartilhamento de dados. As trocas de mensagens são realizadas através da estrutura de comunicação, supondo a existência de endereços dos processos de origem e destino da informação. O mecanismo de caixas-postais é similar, podendo envolver endereços abstratos e selecionar tipos de mensagens que se deseja receber. O mecanismo de *rendez-vous* é normalmente embutido na linguagem de programação, e exige um aceite explícito da comunicação, servindo então também como mecanismo de sincronização. O compartilhamento de dados, por sua vez, permite o assincronismo entre o produtor e o consumidor da informação, permite uma definição mais precisa do estado do sistema, mas apresenta restrições quanto ao controle de acesso (concorrência) aos dados, e pode ser mais sensível à velocidade da rede de comunicação. Decisões quanto à granularidade dos processos e mecanismo de comunicação limitam ou expandem o escopo dos sistemas e aplicações distribuídas. Outras questões incluem as formas de interação interprocessos (através, por exemplo, do modelo cliente-servidor, *pipes* ou chamada de procedimentos remota).

3.4 Sistemas Baseados em Casos

Os sistemas baseados em casos são sistemas que utilizam experiências passadas para compreender e resolver novos problemas [55]. Assume-se que, em geral, resolver um problema pela segunda vez é mais fácil que pela primeira, porque é possível lembrar e repetir a solução encontrada anteriormente. É possível também lembrar os erros, e evitá-los ou corrigi-los.

Dada uma especificação de entrada, um sistema baseado em casos buscará em sua base de casos, um caso anterior equivalente a esta especificação. Se obtiver sucesso (e o sucesso neste caso depende do número de casos armazenados), obterá um caso anterior equivalente ao novo problema, possibilitando aplicar rapidamente as soluções já obtidas. Poderá ainda obter um caso similar à nova situação, mas que não proveja uma solução completa para o problema. Neste caso, o sistema deverá identificar as porções do caso obtido que não satisfazem as especificações de entrada e substituí-las. Esse processo de adaptação de um caso anterior gera a solução para um novo problema, que poderá também ser armazenado na base de casos para uso futuro. O aprendizado deste novo caso poderá minimizar o esforço computacional na resolução de casos similares. O funcionamento de um sistema baseado em casos é ilustrado pela Figura 3.3.

Assim, a qualidade das soluções desta abordagem depende dos seguintes elementos:

- a experiência adquirida;
- a habilidade de entender as novas situações em relação às experiências anteriores;
- a capacidade de adaptação;
- a capacidade de avaliação.

As áreas de aplicação desta abordagem incluem: *o planejamento*, onde utilizam-se planos anteriormente utilizados e que atingiram plena ou parcialmente os objetivos

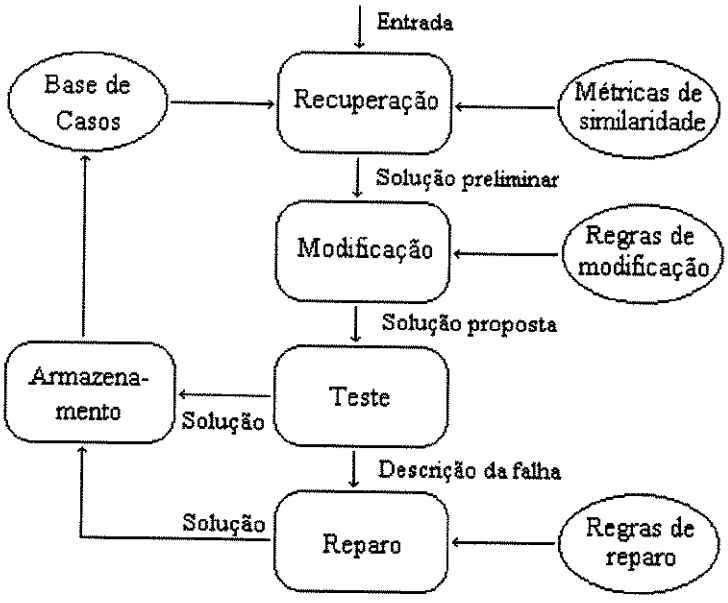


Figura 3.3: Exemplo de funcionamento de sistema baseado em casos

atuais [26, 25, 37]; *projeto* (e outras aplicações onde as soluções dificilmente podem ser decompostas), onde as restrições de projeto provêem informações que conduzem à solução dos novos problemas [42]; *diagnose*, onde os sintomas explicam ou caracterizam os diagnósticos [53]; e *explicação*, onde explicações sobre fenômenos anteriores servem de base para prover explicações sobre fenômenos parecidos [93].

A partir de sua aplicabilidade, identificam-se duas formas principais de sistemas baseados em casos: aqueles que buscam a resolução de problemas e aqueles que buscam a interpretação. Para a resolução de problemas, as soluções anteriores provêem a base para a solução dos novos problemas, além de servirem de alerta contra erros potenciais. Para a interpretação, as novas situações são interpretadas no contexto de situações passadas. Um sistema baseado em casos deve levar em conta cinco importantes funções básicas:

1. representação dos casos;

2. indexação;

3. armazenamento e recuperação;

4. adaptação;

5. aprendizagem e generalização.

A *representação* dos casos pode ser realizada através de *frames*, listas de características ou através de conjuntos conectados de sub-casos, por exemplo. Eles são compostos por três partes principais, não necessariamente preenchidas: a descrição da situação do problema, a solução especificada, e o resultado da aplicação dessa solução. A representação envolve a definição da terminologia do domínio, e a coleção de exemplos (casos) representativos do domínio [3, 54].

A *indexação* de casos serve para permitir a sua recuperação sempre que puderem ser úteis, embora prever todas as situações onde serão úteis nem sempre é possível. Nesse caso, pode-se recuperá-los baseando-se somente em similaridade. A indexação pode ainda ser realizada de três outras formas: através de indexação indutiva, guiada por conhecimento, ou uma combinação das anteriores. Indexando os casos por similaridade, a busca é baseada em uma soma ponderada das características do novo problema que equivalem às características de cada caso armazenado. A abordagem indutiva tenta determinar quais características dos casos os levarão a obter melhor desempenho, necessitando, portanto, de uma grande quantidade de casos e conhecimento sobre o domínio de aplicação. A abordagem guiada por conhecimento tenta aplicar o conhecimento existente nos casos para determinar quais são suas características mais importantes, e exige conhecimento explanatório disponível sobre os casos.

O *armazenamento* e *recuperação* dos casos depende da forma como eles foram indexados. As abordagens encontram-se em um espectro entre as associativas (utilizadas, por exemplo, pela indexação por similaridade), e as hierárquicas. A organização dos casos depende ainda da quantidade de conhecimento existente para classificar os casos, e da flexibilidade desejada (as organizações hierárquicas podem oferecer recuperações de ordem logarítmica, mas podem também tornar a estrutura mais rígida).

O processo de *adaptação* toma o caso que mais satisfaz as características do novo problema, e o modifica de forma a atender todos os seus requisitos. Este é o processo mais dependente do domínio de aplicação.

A *aprendizagem* em sistemas baseados em casos pode ser realizada pelo simples armazenamento de novos casos. Com o acúmulo destes, é possível ainda definir casos protótipos, isto é, casos que possuem as características genéricas de grupos de casos específicos, podendo melhorar sua representação e aumentar o conhecimento sobre o domínio.

A abordagem baseada em casos é particularmente útil para domínios onde o conhecimento seja incompleto, ou as evidências sejam esparsas. Os sistemas baseados em casos tentam suprir essa falta de conhecimento, baseando-se em sua experiência. As soluções geradas, portanto, nem sempre serão ótimas ou mesmo corretas. Trazem, no entanto, outras vantagens, como o reaproveitamento do esforço computacional anterior em resolver problemas, a possibilidade de propor soluções em domínios não completamente conhecidos, e meios de prevenir-se contra possíveis repetições de falhas. Além disso, eles facilitam a aquisição de conhecimento sobre o domínio da aplicação, e propiciam uma memória dos problemas e decisões tomadas [55, 98].

3.5 Algoritmos Genéticos

A aplicação de algoritmos genéticos para a resolução de problemas iniciou-se com trabalhos de J.H. Holland [43]. Os algoritmos genéticos são modelados para simular computacionalmente o processo de evolução segundo a teoria de seleção natural proposta por Darwin. Assim, são algoritmos de busca baseados nos mecanismos de seleção natural e genética, onde o ambiente pode sofrer variações no tempo, e os indivíduos podem existir somente se adaptarem-se ao ambiente. Os algoritmos genéticos propõem a sobrevivência de estruturas (*strings*), através da troca estruturada e ao mesmo tempo randômica de suas informações. A cada geração, novos *strings* são criadas a partir das informações dos melhores *strings* anteriores, constituindo uma *população.* Apesar dessa troca randômica, eles não constituem um mecanismo puramente randômico, por explorarem informações históricas e manipularem mecanismos de busca que focam sua atenção na melhoria do desempenho [31]. Seu funcionamento pode ser ilustrado pela Figura 3.4.

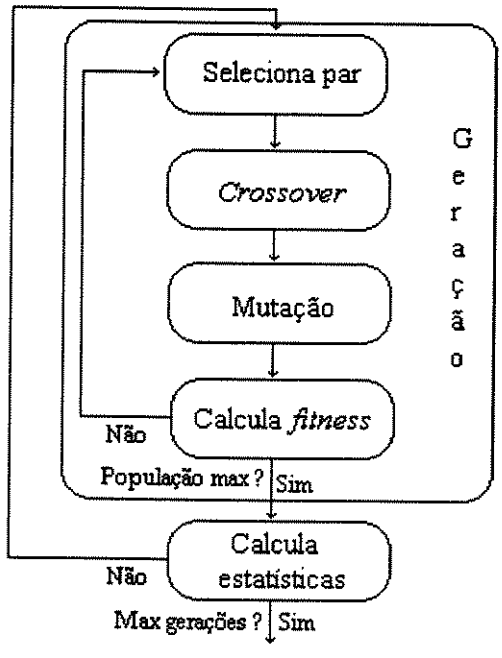


Figura 3.4: Esquema de funcionamento de algoritmo genético

São dois os objetivos desta abordagem: abstrair e explicar os processos adaptativos dos sistemas naturais, e projetar sistemas artificiais que simulem os mecanismos dos sistemas naturais.

O tema central da pesquisa em algoritmos genéticos tem sido a *robustez*, que permite a sobrevivência de *strings* em ambientes diferentes e dinâmicos. Para os sistemas de computação, a robustez implica em menores custos de manutenção. Além disso, caso se obtenha razoável adaptabilidade dos sistemas, eles poderão ser utilizados por mais tempo e com melhor desempenho.

Apesar de conseguirem proporcionar tais características altamente desejáveis, os algoritmos genéticos envolvem pouca complexidade computacional. Eles são compostos por três operadores básicos:

1. Reprodução;
2. *Crossover*;
3. Mutação.

Reprodução é o processo pelo qual os indivíduos (*strings*) são copiados, de acordo com suas aptidões (funções-objetivo). As aptidões ou funções-objetivo representam as características que se pretende maximizar ou minimizar. Ao tomar-se os *strings* de acordo com suas funções-objetivo (ou os indivíduos de acordo com suas aptidões), está se proporcionando mais chance aos *strings* com valores mais altos/baixos, de que contribuam para as próximas gerações.

O processo de *crossover*, ou combinação, pode ser dividido em duas fases. Na primeira, os *strings* selecionados para reprodução são agrupados aos pares. A segunda fase consiste na seleção (randômica) de uma posição k, entre 1 e o tamanho do *string* (l) menos 1 ([1, l-1]). Dois novos *strings* são criados pela troca de todos os caracteres dos *strings* entre as posições $k+1$ e l. Por exemplo, considere a combinação entre os *strings* A_1 e A_2 a seguir:

A_1= 0 1 1 0 | 1 1

A_2= 1 1 0 0 | 0 0

Suponha que na seleção randômica de um inteiro entre 1 e 5, tenhamos obtido $k = 4$, representado acima pelo separador "|". O *crossover* resulta em dois novos indivíduos (*strings*) A'_1 e A'_2:

A'_1= 0 1 1 0 0 0

A'_2= 1 1 0 0 1 1

Os mecanismos de reprodução e *crossover* são bastante simples, envolvendo apenas geração randômica de números, cópias de *strings*, e trocas parciais de *strings*. No entanto, essa computação simples provê aos algoritmos genéticos muito de seu poder. Hadamard [35] tenta explicar o poder de combinações como estas:

> "Nós veremos que a possibilidade de atribuir descobertas à sorte é sempre excluída... Pelo contrário, há sempre uma intervenção da sorte, mas também um trabalho da inconsciência, a última implicando e não contradizendo a primeira... Na verdade, é óbvio que a invenção ou a descoberta, seja na matemática ou em qualquer área, acontece quando se combinam idéias."

A *mutação* é o terceiro operador necessário aos algoritmos genéticos. Isso porque, somente com a reprodução e o *crossover*, algumas combinações dos *strings* poderiam ser perdidas (alguns valores em determinadas posições). Nos sistemas genéticos artificiais, o papel das mutações é o de proteger os sistemas contra perdas irrecuperáveis. O processo de mutação consiste em uma alteração randômica, de baixa probabilidade, capaz de alterar o valor de uma posição no *string*.

Algumas abordagens baseadas nos algoritmos genéticos surgiram, dentre as quais se destaca o aprendizado de máquinas baseado em genética e seus *sistemas classificadores* [44].

Um sistema classificador é um sistema que aprende sintaticamente regras simples, denominadas *classificadores*, que dirigirão sua atuação em um determinado ambiente. Compõem-se de três componentes principais:

1. Sistema de mensagens e regras;
2. Sistema de atribuição de crédito;
3. Algoritmo genético.

Grande parte dos sistemas classificadores é baseados em sistemas de produção [15]. Nos sistemas classificadores, as regras têm tamanho e representação restritos, sendo transmitidos e recebidos pelo sistema de mensagens e regras. Restringindo-se a representação das regras, permite-se que todos os *strings* dentro de um alfabeto, sejam sintaticamente compreensíveis. Além disso, o seu tamanho fixo permite a aplicação de operadores baseados nos algoritmos genéticos para efetuar combinações entre as regras. Em um sistema classificador, os valores relativos das regras a serem aplicadas são aprendidos pelo sistema, ao contrário dos sistemas de produção convencionais, onde esses valores são fixados a priori. Esse aprendizado se dá por competição. Às regras que geram melhores resultados, são atribuídos os maiores valores pelo sistema de atribuição de crédito, permitindo que as melhores regras tenham maior probabilidade de gerarem novas regras.

Recentemente, os algoritmos genéticos têm sido empregados para expressar modelos de adaptação ambiental de sistemas biológicos, resolução de problemas combinatoriais de otimização, problemas de busca, classificação e aprendizagem, dentre outras [10, 47, 91, 106]. Segundo [106], o uso dos algoritmos genéticos deve-se, dentre outros, à possibilidade de otimização independentemente da formulação das funções-objetivo. A

razão disso é o emprego de um algoritmo probabilístico de busca, e assim, o fato de a função-objetivo ser bem comportada, não afeta a otimização.

3.6 Resumo

Nesse capítulo, foram introduzidos conceitos básicos da teoria dos conjuntos nebulosos, dos sistemas distribuídos, dos sistemas baseados em casos, e dos algoritmos genéticos. Juntamente com a inteligência artificial distribuída, esses temas constituem as áreas de pesquisa sobre as quais este trabalho foi constituído.

No capítulo seguinte, será apresentada uma nova abordagem derivada da inteligência artificial distribuída, denominada Inteligência Computacional Distribuída. Essa nova abordagem se caracteriza pela agregação de elementos das várias áreas apresentadas, visando dar um novo enfoque à resolução distribuída de problemas, provendo flexibilidade e adaptabilidade e permitindo o desenvolvimento de sistemas reais complexos.

Capítulo 4

Inteligência Computacional Distribuída

4.1 Introdução

A inteligência computacional refere-se à "cognição de baixo nível no estilo da mente", segundo [4]. Segundo sua visão, um sistema é computacionalmente inteligente quando:

- manipula somente dados numéricos;
- possui um componente de reconhecimento de padrões;
- não utiliza conhecimento no sentido da inteligência artificial;
- exibe ou começa a exibir adaptabilidade computacional;
- possui alguma forma de tolerância a falhas;
- apresenta tempos de resposta próximos aos humanos;
- apresenta taxas de erros próximos ao desempenho humano.

Segundo [120], os limites entre a inteligência artificial e a computação inteligente não são claros. Segundo sua visão, a inteligência computacional inclui a computação evolutiva, a vida artificial, tópicos dos sistemas nebulosos e as redes neurais.

Neste capítulo é introduzida uma nova abordagem no âmbito dos sistemas inteligentes, denominada **Inteligência Computacional Distribuída - ICD**, que une e estende as visões apresentadas. A Figura 4.1 apresenta um diagrama geral para a Inteligência Computacional Distribuída.

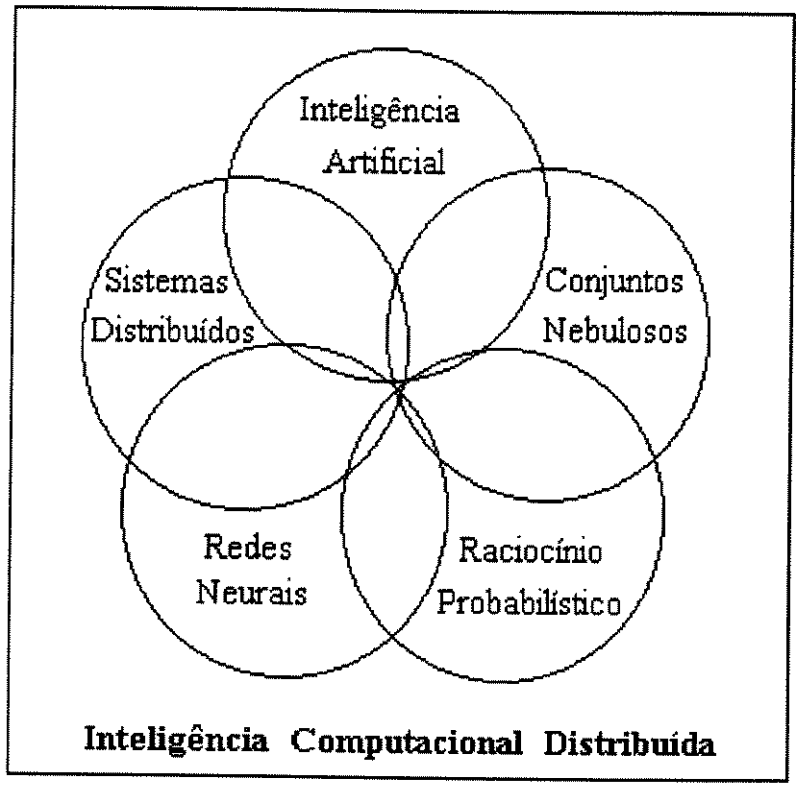


Figura 4.1: Diagrama geral para a Inteligência Computacional Distribuída

Através da simbiose entre os sistemas distribuídos de inteligência artificial, a teoria dos conjuntos nebulosos, os sistemas baseados em casos e algoritmos genéticos, são propostos mecanismos para prover flexibilidade, eficiência e adaptabilidade a sistemas distribuídos inteligentes.

O objetivo desta abordagem é prover meios para a definição, implementação e aplicação de sistemas distribuídos inteligentes reais. Propõem-se mecanismos que pos-

sam ser eficazes no tratamento de problemas reais, sujeitos a diferentes eventos e incertezas. A formalização do conhecimento sobre tais problemas é, freqüentemente, incompleta e imprecisa, devido à complexidade envolvida. Um sistema real deve considerar essas questões e propiciar que o sistema possa se adaptar a situações não previstas (modeladas), gerando ainda assim soluções aceitáveis.

Uma questão relacionada aos sistemas distribuídos inteligentes é a dificuldade em estender as abordagens atuais e desenvolver novos sistemas baseados nelas. Isso é acentuado por suas representações geralmente discursivas, causadas em parte pela falta de linguagens formais e semi-formais adequadas para a especificação de sistemas inteligentes e distribuídos complexos. Outras dificuldades incluem a representação do tempo e de informações temporais (tais como conectivos e advérbios temporais), e a necessidade de modelos abstratos adequados para representar concorrência e comunicação.

A maioria das tentativas utilizam representações procedimentais, que acrescentam certo rigor às especificações, mas são ainda difíceis de estender para novos domínios, de caracterizar formalmente, e tornam difícil uma comparação mais apurada entre seus diferentes agentes [95].

As representações declarativas, embora sejam mais difíceis de desenvolver, tornam explícitos o conhecimento e as habilidades dos agentes que permitem a sua interação e cooperação. Além disso, elas permitem o desenvolvimento de modelos formais dos agentes. Apesar da complexidade envolvida, esse tipo de representação é de grande importância para sistemas que empregam raciocínio temporal [28], e algumas tentativas têm sido realizadas para prover linguagens declarativas de especificação para sistemas de inteligência artificial distribuída, como em [95, 27].

A seguir, será apresentada a arquitetura **ICD**: sua descrição, seus componentes e seu funcionamento, além de sua especificação formal em uma linguagem declarativa de especificação baseada em [95].

4.2 Descrição

Um sistema ICD [70, 71] consiste de uma rede de nós de processamento, capazes de raciocinar e agir autonomamente. Embora os nós sejam autônomos, eles devem cooperar para atingir objetivos globais, como uma comunidade. Na realidade, em um sistema ICD, os nós tomam os maiores benefícios globais como prioridade para o seu próprio processamento. Apesar disso, eles são sempre os responsáveis pelo controle de seus dispositivos locais. Para obter cooperação, eles devem conhecer a estrutura e o estado do sistema, além de seus objetivos e restrições. Assim, os nós devem comunicar-se entre si para manter suas informações atualizadas, sem obstruir o meio de comunicação.

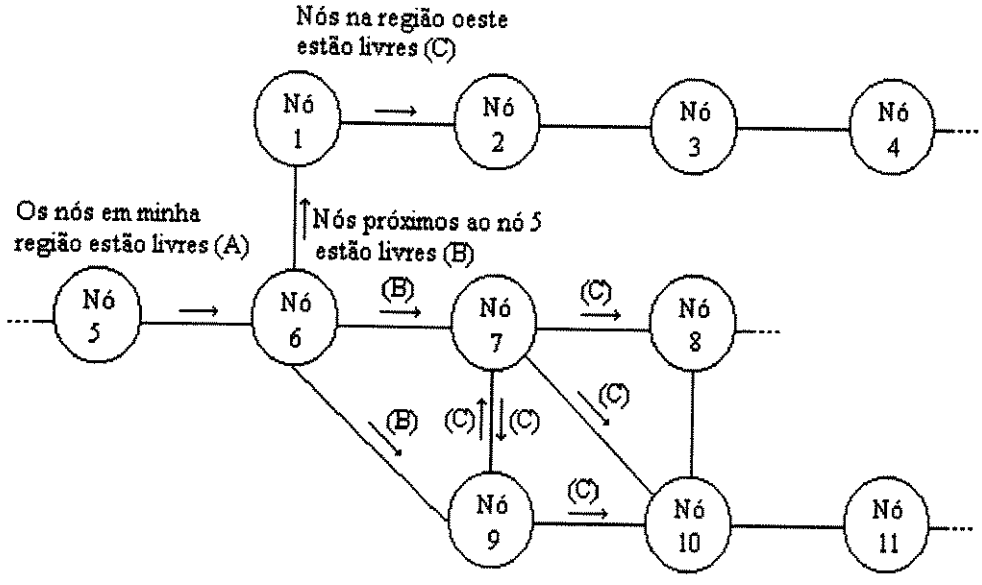


Figura 4.2: Exemplo de uma rede ICD

Os nós comunicam-se enviando mensagens de alto nível, capazes de encapsular bastante significado, permitindo ainda a comunicação de incertezas no sistema. Na Figura 4.2, o *nó 5*, por exemplo, informa o *nó 6* que existem outros nós próximos desocupados (mensagem *A*). Essa informação é propagada pelo *nó 6* (através da mensagem *B*) e por seus vizinhos (mensagem *C*), somente para nós que possam estar interessados nela, evitando saturar o meio de transmissão com mensagens desnecessárias.

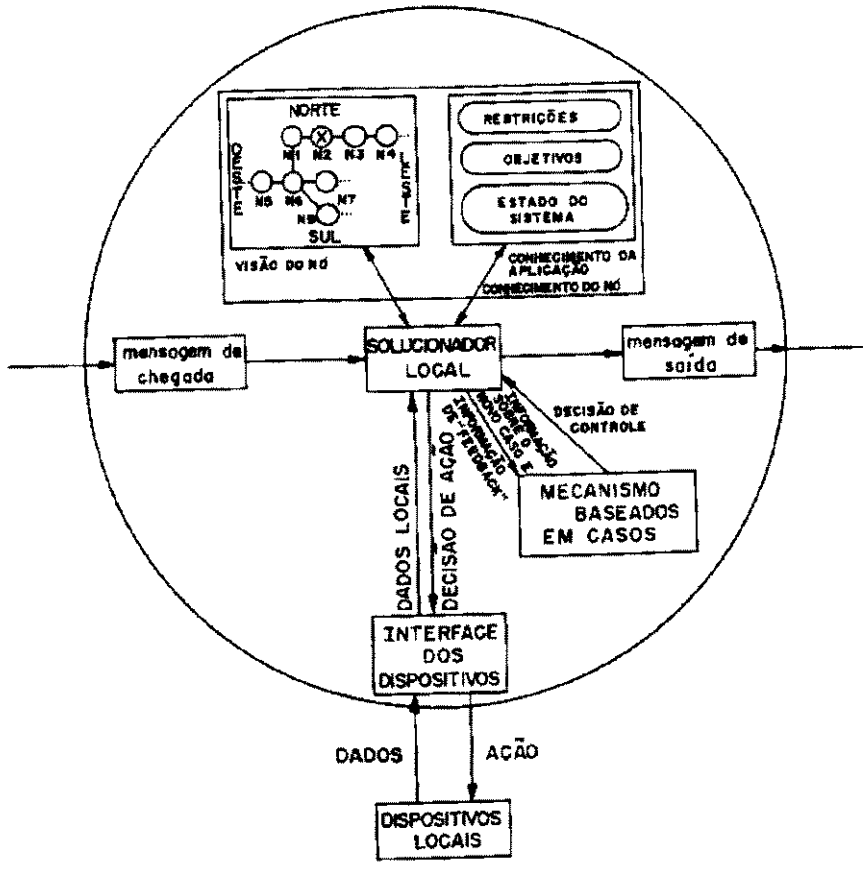


Figura 4.3: Estrutura de um nó

Cada nó conhece a estrutura da rede: conhece sua vizinhança em detalhes e tem uma visão geral do sistema. Além disso, ele é capaz de raciocinar sobre o problema global e como ele pode cooperar na resolução do problema, pois possui também conhecimento sobre a aplicação. Esse conhecimento do nó, representado na parte superior da Figura 4.3, contém as informações necessárias para permitir que os nós cooperem entre si.

O mecanismo de funcionamento é bastante simples: um nó recebe informações de seus dispositivos locais e dos nós adjacentes. Essas informações são tratadas pelo Solucionador Local que, de posse do Conhecimento do nó, pode, por exemplo, alterar sua visão do Estado do Sistema. Pode também produzir uma Decisão de Ação ou uma mensagem a ser enviada a outro(s) nó(s). Assim, as decisões podem ser tomadas por

consenso, através da troca interativa de mensagens, ou pelo nó mais próximo ou mais bem informado. Um Mecanismo Baseado em Casos (MBC) auxilia o Solucionador Local. Ele analisa a situação corrente e verifica casos anteriores que sejam similares. As decisões tomadas nos casos anteriores são adaptadas à nova situação com o objetivo de melhorá-las, gerando então uma decisão de controle. O Mecanismo Baseado em Casos observará os resultados dessa decisão, e os utilizará para sua aprendizagem, possibilitando melhorias futuras de desempenho.

Os esquemas básicos de funcionamento do Solucionador Local e do Mecanismo Baseado em Casos, detalhados na Seção 4.5 e exemplificados na Seção 5.4, são os seguintes:

Procedure: Solucionador Local

Begin

Recebe_rede(informação_1);

Recebe_sensores(informação_2);

Atualiza_visão_do_sistema;

Envia_vizinhos(visão);

if não_capaz(Tomar_decisão(visão,regras_ou_heurísticas))

then

Envia_MBC(situação);

Recebe_MBC(decisão);

Executa(decisão);

Envia_vizinhos(decisão);

Envia_MBC(desempenho_da_decisão)

End

Procedure: **Mecanismo Baseado em Casos**

Begin

Recebe_Solucionador(situação);

Procura_casos_similares(situação);

Seleciona_casos_bem_sucedidos;

Combina_ações;

Envia_Solucionador(ações);

Recebe_Solucionador(desempenho);

Decide_armazenamento(situação,ações,desempenho)

End

4.3 Arquitetura

4.3.1 O Solucionador Local

O Solucionador Local é o responsável por toda a interface do nó: localmente, comunicando-se com eventuais dispositivos; e globalmente, comunicando-se com os outros nós da rede. Pode ser composto por regras de produção, heurísticas, regras nebulosas, modelos de redes neurais, ou uma combinação entre elas. É o responsável por tomar as decisões rotineiras, que são aquelas que lidam com situações que puderam ser modeladas e incorporadas previamente ao sistema, eventualmente contando com soluções ótimas.

O Solucionador Local recebe informações de seus sensores e dos nós vizinhos. Essas informações podem envolver diversas imprecisões, o que pode tornar o processo de decisão mais difícil. Além das incertezas relacionadas aos dados dos sensores (devidos à sua própria imprecisão ou ao ruído, por exemplo), existem as imprecisões recebidas através da rede. As incertezas são propagadas quando os processadores informam seus vizinhos sobre suas condições locais. Essas imprecisões podem acumular-se ao difundirem-se as informações [79].

Desta forma, cada nó deve tomar conhecimento, raciocinar e aprender sobre o estado do problema, disperso por toda a rede, para tomar decisões razoáveis em tempo-real. Além disso, os nós mantêm registros de diferentes situações passadas. Essas informações baseadas em casos passados podem ajudar no processo de decisão, aumentando o grau de certeza e a velocidade na tomada de decisão, e melhorando o desempenho global do sistema.

Troca de mensagens

Para obter comportamento mais coerente, os nós trocam mensagens próximas à linguagem natural. Utilizando construções nebulosas, como *variáveis lingüísticas* [113], é possível inferir e tomar decisões a partir de proposições construídas com elas, como será mostrado proximamente. A função essencial das variáveis lingüísticas é a da granulação das variáveis, atuando na compressão dos dados transmitidos e reproduzindo as habilidades humanas de sintetizar os dados e concentrar as decisões nas informações relevantes [117].

Os agentes comunicam-se entre si através de mensagens como:

"Processadores ociosos no Departamento de Matemática."

"Problemas na Área de Administração."

"Não ocupe os processadores do Departamento de Economia; ali há sérios problemas."

"Tente desocupar os processadores do Departamento de Estatística."

Como os nós conhecem a estrutura e o estado dos processadores de sua vizinhança e têm a visão geral de toda a rede, eles podem decidir quais informações são importantes para quem. Além disso, eles podem trocar planos de ação, baseados nas informações recebidas e nos casos passados. Assim, minimizam a troca de mensagens irrelevantes.

As mensagens circulam apenas na região onde são de interesse e somente enquanto forem relevantes.

O conhecimento sobre a estrutura da rede, particularidades da aplicação, casos passados, e conhecimento sobre os outros nós provêem os meios necessários para permitir que os agentes raciocinem de forma dinâmica e efetiva, tomando decisões satisfatórias em tempo-real. Quando os nós tornam-se mais sofisticados, podem ser capazes de explorar melhor as informações disponíveis, escolhendo as informações mais importantes para transmitir aos outros nós e evitando o tratamento de muitas mensagens e o congestionamento do meio de transmissão.

Tomada de Decisão com Incertezas

O processo de tomada de decisão em um ambiente de incertezas pode ser tratado segundo várias abordagens, com um ou vários estágios, uma ou mais entidades tomando parte do processo de decisão, através de uma simples otimização de uma função-objetivo, otimização com restrições, ou com critérios múltiplos [80].

Em um sistema ICD, cada nó toma localmente suas decisões, buscando a melhor decisão para o sistema, e utiliza para isso seu conhecimento e as informações recebidas. Em caso de indeterminação ou falta de tempo (para aplicações tempo-real), o nó age sobre seus dispositivos locais, levando em consideração que casos passados poderão potencialmente conduzir a decisões acertadas. Dependendo do sucesso das decisões anteriores e da disponibilidade de tempo, as prioridades podem ser alteradas. Por exemplo, se um nó precisa tomar uma decisão rapidamente e tem recebido boas informações de realimentação, ele pode continuar com a mesma estratégia de controle. Se sua realimentação não for boa, uma boa e rápida alternativa poderia ser tomar uma decisão de sucesso já pronta, a partir de um caso similar anterior.

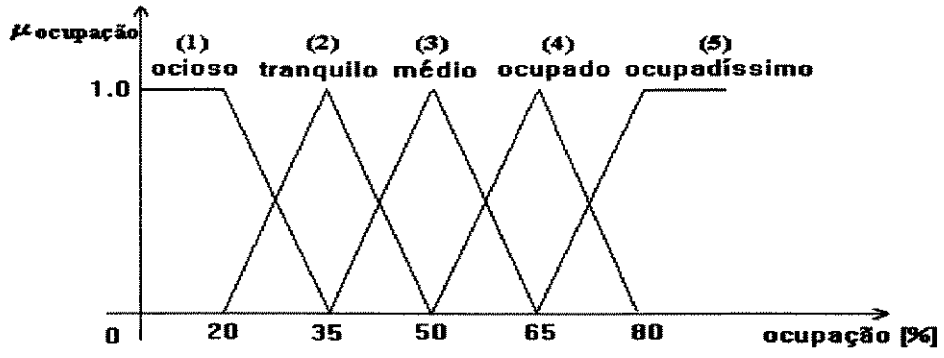


Figura 4.4: Funções de pertinência associadas aos termos de uma variável lingüística

As mensagens são tratadas em tempo-real, de forma que os nós tenham sempre visões "quase atualizadas" do estado do sistema. As inferências são derivadas de regras e de conhecimento, transmitido através de mensagens.

Por exemplo:

Considere a variável lingüística apresentada na Figura 4.4. Considere também a seguinte regra geral:

SE (processadores ocupadíssimos no Departamento *DEPARTAMENTO*)
ENTÃO (tente diminuir a ocupação do Departamento *DEPARTAMENTO*) (1)

E a seguinte regra específica:

SE ((tente diminuir a ocupação do Departamento de_Computação)
e (Departamento de_Matemática está ocioso))
ENTÃO (incentive a ocupação do Departamento de_Matemática) (2)

Considere ainda que o nó possua o seguinte conhecimento ou mensagens recebidas:

a) Processadores ocupadíssimos no Departamento de_Computação.

b) Departamento de_Matemática está ocioso.

O nó será capaz de inferir, a partir da regra *1* e do conhecimento *a)*, *(tente diminuir a ocupação do Departamento de_Computação)*.

De forma similar, a partir desse conhecimento, da função de pertinência, da regra *2* e do conhecimento *b)*, ele será capaz de inferir *(incentive a ocupação do Departamento de_ Matemática)*, tentando balancear melhor a carga de processamento.

Essa abordagem nebulosa provê um instrumento formal para manipular incertezas e tomar decisões, dando maior flexibilidade e permitindo meios para incorporar raciocínio de senso comum [111].

Caso o Solucionador Local seja implementado através de regras de controle nebulosas, o esquema de funcionamento de sua tomada de decisão será o seguinte:

Procedure: Tomar_decisão(situação_corrente,regras)

Begin

if modelada(situação_corrente) then

Converte_para_variáveis_lingüísticas(situação_corrente);

Dispara_regras_nebulosas(regras);

decisão = Calcula_centro_de_área;

else

decisão = NULL

End

4.3.2 O Conhecimento do Nó

Cada nó deve possuir conhecimento sobre a estrutura física do sistema e sua posição nessa estrutura (caso se trate de uma malha, como ilustrado pela Figura 4.2), além de conhecimento dependente da aplicação para que possa contribuir na resolução do problema. Dessa forma, seu conhecimento pode variar bastante em função da topologia da rede, da aplicação e dos mecanismos de resolução adotados pelo Solucionador Local. De forma geral, deve incluir:

- **Visão do nó**. O nó deve conhecer sua posição física (e eventualmente lógica) na rede. Deve conhecer seus nós vizinhos e nós especiais de supervisão, caso existam. Deve ainda ter uma visão geral de toda a rede, como por exemplo, regiões com nós dedicados ou especializados em determinadas tarefas. Dessa forma, pode reconhecer e endereçar mensagens para qualquer nó do sistema;

- **Conhecimento da aplicação**. O nó deve também conhecer os objetivos do sistema, as restrições que devem ser respeitadas, além de seu estado global. Outras formas de conhecimento que dirijam o processo de resolução, tal como heurísticas, podem estar incluídos no Solucionador Local, permitindo eventual substituição modular do processo de resolução.

4.3.3 O Mecanismo Baseado em Casos

O objetivo do Mecanismo Baseado em Casos é prover formas simples de aquisição de conhecimento, tomar decisões rápidas e eficientes aproveitando o esforço computacional anterior, aumentar a robustez, e permitir melhor desempenho com o passar do tempo. Não se pode limitar sempre as decisões em aplicações tempo-real a resultados teóricos ou heurísticas com eficiência. Decisões anteriores, que obtiveram sucesso ou não, podem contribuir também para aplicações dinâmicas e complexas.

O Mecanismo Baseado em Casos básico proposto realiza quatro funções principais: a Busca, a Adaptação, a Tomada de Decisão e a Aquisição de Conhecimento (Aprendizagem), mostradas na Figura 4.5.

A Busca toma as informações dos casos passados, armazenadas na Base de Casos, que é estruturada hierarquicamente. A recuperação dos casos é feitas com base na Nova Situação (novo caso), recebida pelo Solucionador Local. A busca dos casos na árvore, através de seus atributos mais genéricos, reduz o espaço de casos a serem examinados para identificar quais os mais similares, representados na Figura 4.5 por *Situação 1* a *Situação i*).

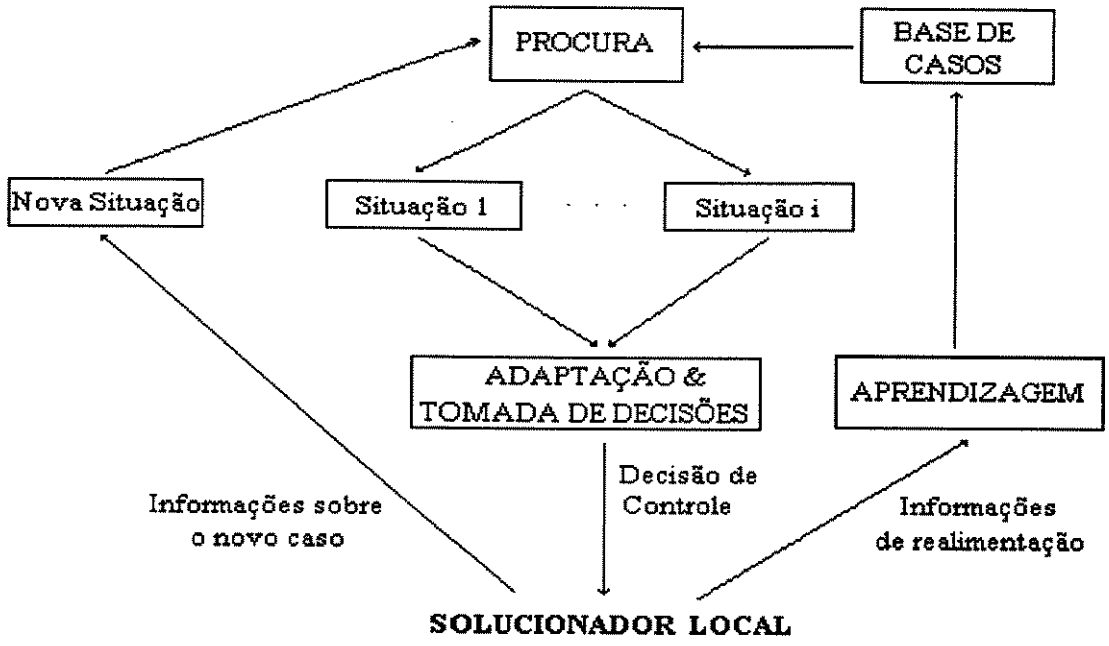


Figura 4.5: O Mecanismo Baseado em Casos

Os casos mais similares e que obtiveram os melhores resultados são adaptados à nova situação, e uma decisão nebulosa como "Incentive a ocupação do Departamento de_Matemática" é gerada.

Após o Solucionador Local executar a decisão, o Mecanismo Baseado em Casos irá observar se a decisão gerada foi apropriada, verificando o desempenho resultante. Este novo caso e seus resultados irão alimentar a Base de Casos, permitindo que o desempenho do sistema aumente com o tempo.

Mecanismos Evolutivos

Para identificar e recuperar os casos mais similares, o Mecanismo Baseado em Casos percorre uma base de casos estruturada como uma árvore. Ele manipula as experiências, que são divididas em uma parte de atributos, uma parte de ações, uma parte de resultados, e uma parte de informações complementares (ver Figura 4.6). Cada uma dessas partes é composta por itens, que especificam cada porção de in-

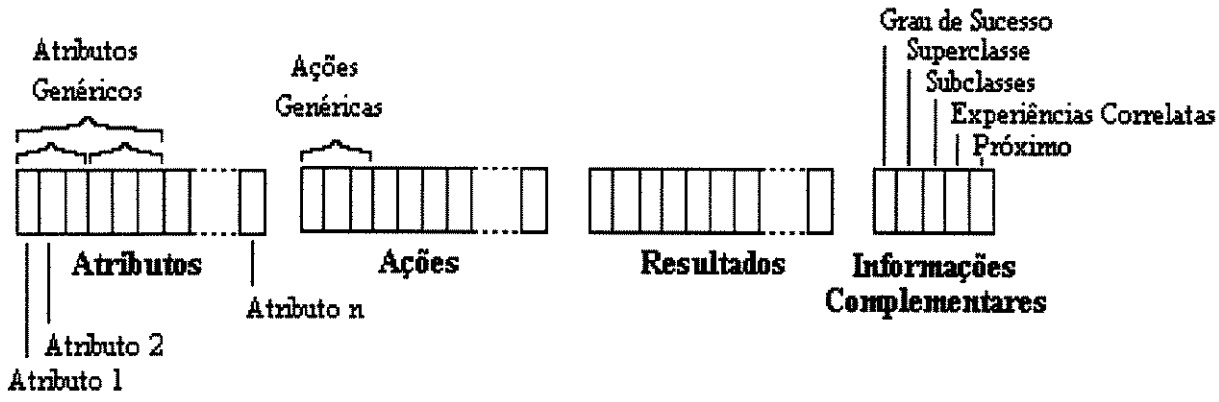


Figura 4.6: Estrutura das experiências

formação referente ao caso. A estrutura de árvore permite, além de um menor tempo de busca, que os casos sejam classificados segundo seus atributos mais importantes, de forma que os casos do mesmo tipo permaneçam nas mesmas subárvores.

Para identificar os casos mais similares, é necessário comparar os atributos do caso atual com os dos casos armazenados na mesma subárvore (isto é, casos do mesmo tipo). Para cada atributo, temos funções de pertinência associadas, que definem o domínio dos valores que o atributo pode assumir. Por exemplo, para a variável *Ocupação* da Figura 4.4, teríamos como domínio os números entre 1 e 5, além de "X" e "?", onde "X" representa "não importa" e "?" representa "indeterminado".

Consideremos as seguintes partes de atributos para o caso atual e um dos casos armazenados:

Atributos do caso atual (AT):	4	X	1	5	2	?	3	2	X	3	5	3

Atributos do caso armazenado (AR):	5	1	1	3	?	3	X	0	X	5	1	3

Muito poucos atributos dos dois casos coincidem perfeitamente, embora as experiências possam ser bastante parecidas. É necessário comparar cada atributo e estabelecer um *grau de proximidade* entre o caso atual e os casos armazenados, conforme mostra a equação 4.1:

$$grau\ de\ proximidade = \frac{\sum_{i=1}^{n} w_i s_i}{n}, \tag{4.1}$$

$$\text{onde} \begin{cases} s_i = 1 \text{ se } atributo_i(AT) = X \text{ ou } atributo_i(AR) = X \\ \quad \text{(ou se o atributo tiver somente um valor possível)} \\ \quad \text{senão} \\ s_i = 1 - \frac{|atributo_i(AT) - atributo_i(AR)|}{\text{escopo de conversão do alfabeto do atributo}} \end{cases}$$

onde n é o número de atributos sem '?', w_i é o peso do atributo i, $atributo_i(X)$ corresponde ao i-ésimo atributo de X diferente de '?', e escopo corresponde ao número de termos lingüísticos da variável lingüística menos 1.

Para o exemplo apresentado, considerando um peso 1 e um escopo 5 para todos os atributos, teríamos um grau de similaridade de 0.78[1].

Obviamente, ao invés realizar esse teste de proximidade com todos os casos armazenados, é selecionada inicialmente uma (ou um pequeno grupo de) subárvores que contém os casos do mesmo tipo. O teste de similaridade é realizado com os casos armazenados nessa(s) subárvore(s), e com outros casos correlatos obtidos através de ponteiros a partir deles. Para cada caso testado, obtém-se o grau de proximidade e recupera-se o seu grau de sucesso/fracasso armazenado em sua parte de resultados.

[1]Para o atributo referente à variável *Ocupação* da Figura 4.4, que possui 5 termos lingüísticos - ocioso, tranqüilo, médio, ocupado e ocupadíssimo - o escopo seria 4.

Esses valores permitem selecionar os casos mais similares, de melhor resultado, ou através de suas funções-objetivo.

Após recuperar os casos mais similares através de sua parte de atributos, o Mecanismo Baseado em Casos combina a parte de ações dos casos selecionados através de algoritmos genéticos. Há três estratégias possíveis:

1. tomar as ações por atributos genéricos. Como ilustração, poderiam ser tomados os olhos de um indivíduo (tamanho, formato e cor), o nariz de um outro, e assim por diante. Essa é uma boa estratégia para ações inter-relacionadas ou de granularidade fina.

   Parte de ações da Experiência 1:

2	1	4	X	5	?	3	1	X	1	5	6

   Parte de ações da Experiência 2:

6	X	X	3	1	4	X	2	1	6	2	2

   Parte de ações resultante:

2	1	4	3	1	4	X	2	1	1	5	6

2. tomar as ações individualmente, "gene a gene". Quando os valores das duas ações forem iguais, simplesmente toma-se seu valor. Quando os valores forem diferentes, toma-se um dos dois, evitando 'X' e '?'.

   Parte de ações da Experiência 1:

2	1	4	X	5	?	3	1	X	1	5	6

   Parte de ações da Experiência 2:

6	X	X	3	1	4	X	2	1	6	2	2

   Parte de ações resultante:

6	1	4	3	5	4	3	1	1	1	2	2

3. tomar as ações individualmente, escolhendo qualquer valor situado entre os valores das duas experiências.

Parte de ações da Experiência 1:	2	1	4	X	5	?	3	1	X	1	5	6

Parte de ações da Experiência 2:	6	X	X	3	1	4	X	2	1	6	2	2

Parte de ações resultante:	4	1	4	3	5	4	3	1	1	4	2	3

Esses mecanismos permitem que se trabalhe com as experiências de maior grau de similaridade e as mais bem sucedidas, combinando-as antes de uma possível adaptação. Entretanto, mecanismos desse tipo, embora possam melhorar decisões passadas, evitam a tomada de ações completamente diferentes, pois só adaptam decisões anteriores, e isso pode conduzir a máximos locais [106]. Alguns valores ou combinações de valores podem ser perdidos durante o processo de combinações sucessivas. Para evitar essa limitação, foi introduzida, com baixa probabilidade, uma mudança randômica dos valores das ações.

Após ter gerado a decisão, o Mecanismo Baseado em Casos a envia para o Solucionador Local e aguarda seus resultados.

A parte de resultados do caso é preenchida pelo exame da efetividade das ações tomadas. Por exemplo, se um medicamento for ministrado para baixar a febre de um paciente de 39.5ºC para 37.5ºC, e a tiver baixado para 39ºC, podemos considerar isso como um resultado *ruim*. Um *bom* resultado, por exemplo, teria sido a febre ter baixado para 38ºC. Após obter-se todos os dados na parte de resultados da experiência, calcula-se o seu grau geral de sucesso através de uma variação da equação do grau de similaridade. Maiores detalhes serão mostrados na Seção 5.4.2.

Experiências médias (isto é, experiências com grau de sucesso regulares) são armazenadas na base de casos somente se houverem poucos casos do mesmo tipo (na mesma subárvore), ou por pequenos períodos de tempo. As melhores experiências

(aquelas mais bem sucedidas) são sempre armazenadas, e tendem a permanecer na base de casos por bastante tempo. Isso porque quando a subárvore atinge o seu tamanho máximo, são retirados os casos de pior desempenho e o caso mais antigo (à exceção do melhor caso dentre todos). Dessa forma, mesmo os casos de bom desempenho acabam sendo eliminados da árvore, para evitar que os novos casos possam acabar sendo sempre gerados a partir dos mesmos casos armazenados. Isso leva a um maior dinamismo da base de casos, evitando a degradação dos resultados com o passar do tempo.

A aplicação desse processo de combinar experiências passadas, analisar seus resultados e utilizar as melhores decisões passadas em casos futuros seleciona as melhores soluções e faz com que o sistema melhore seu desempenho, mantendo-se apto a adaptar-se a novas situações.

A seguir, serão apresentados esquemas de funcionamento mais detalhados para o Mecanismo Baseado em Casos. Esses esquemas serão especificados formalmente na Seção 4.5 e exemplificados na Seção 5.4.

Procedure: Gera_decisão_MBC

Begin

Recebe_Solucionador(atributos);

Seleciona_subárvore(atributos);

Calcula_grau_proximidade(atributos,casos_armazenados);

Recupera_grau_sucessso(casos_armazenados);

Seleciona_por_média_ponderada(grau_proximidade,grau_sucesso,casos_armazenados);

Combina_ações(casos_selecionados);

Envia_Solucionador(ações_resultantes)

End

Procedure: **Analisa_desempenho_MBC**

Begin

Recebe_Solucionador(situação_atual);

Identifica_caso;

Calcula_desempenhos_por_atributo(caso,situação_atual);

Calcula_desempenho_geral(caso);

Identifica_subárvore(caso);

Decide_armazenamento(caso,desempenho,casos_armazenados)

End

4.4 Linguagem de Especificação

Para a especificação do sistema, foi escolhida uma linguagem de especificação baseada em lógica proposicional temporal linear (LPTL) [19]. Este tipo de lógica provê uma linguagem abstrata para caracterizar o tempo e para especificar as ações e escolhas tomadas pelos diversos agentes. Ela permite, assim, que as especificações formais possam ser usadas para representar o seu raciocínio. Além disso, permite incluir a presença de incontáveis agentes na sua representação, o que é uma característica fundamental para a especificação de sistemas em IAD.

Para descrever sistemas de IAD de diferentes domínios de aplicação, uma linguagem de especificação deve ser capaz de expressar [95]:

1. O estado do sistema e as partes dos estados dos agentes que são importantes para o sistema;
2. As transições legais do sistema;
3. As restrições do estado do sistema que devem ser respeitadas;
4. As ações legais dos agentes;
5. As habilidades de raciocínio dos agentes;

6. As habilidades de percepção dos agentes;

7. As habilidades de comunicação dos agentes.

Para representar essas características de um sistema, é necessária uma linguagem formal com uma semântica bem definida para descrevê-las. A seguir, serão apresentadas a linguagem formal de especificação utilizada neste trabalho e sua semântica.

4.4.1 A Linguagem Formal

Uma fórmula em LPTL pode ser definida pela seguinte gramática:

1. $< cond >::=< cond\_atômica >$
2. $< cond >::= \neg < cond >$
3. $< cond >::=< cond > \vee < cond >$
4. $< cond >::=< cond > \wedge < cond >$
5. $< cond >::=< cond > \rightarrow < cond >$
6. $< cond >::= X < cond >$
7. $< cond >::= F < cond >$
8. $< cond >::= G < cond >$
9. $< cond >::=< cond > U < cond >$

A semântica para esta linguagem formal refere-se a um modelo composto por estruturas temporais. Uma estrutura temporal linear é uma seqüência de estados que o mundo pode assumir, onde cada estado sucede o outro no tempo, estabelecendo como o sistema pode evoluir ao longo do mesmo. Uma condição atômica, como proposto em [95], é verdadeira em um dado instante se e somente se foi estipulado que ela seria verdadeira naquele instante. As condições $\neg, \vee, \wedge, e \rightarrow$ são entendidas da maneira

convencional. A condição temporal "X *cond*" é verdadeira em algum instante na estrutura se e somente se a condição "*cond*" for verdadeira no próximo instante de tempo naquela estrutura. A condição "F *cond*" é verdadeira em algum instante na estrutura se e somente se "*cond*" for verdadeira em algum instante de tempo posterior naquela estrutura. A condição "G *cond*" é verdadeira em algum instante na estrutura se e somente se "*cond*" for sempre verdadeira no futuro naquela estrutura. "c_1 U c_2" é verdadeira em algum instante se e somente se a condição c_2 é satisfeita em algum instante t_2 no futuro de t_1, e a condição c_1 é satisfeita continuamente de t_1 a t_2.

Descrições de ações descrevem ações que podem ser realizadas por um agente, e assim são tipos de ações. Algumas ações podem ter nomes (como *virar-à-direita* ou *mover-bloco*) e serem descritas diretamente. Outras ações são descritas em termos das condições com as quais elas são relacionadas:

1. $< descrição\_ação >::=< nome\_ação >$
2. $< descrição\_ação >::= Achieve(< cond >)$
3. $< descrição\_ação >::= Maintain(< cond >)$

A semântica de um nome de ação é um conjunto de estruturas e instantes de tempo onde sua condição de definição é satisfeita. Cada nome de ação possui uma condição temporal que a define, chamada *condição de ação.* A semântica de "*Achieve(cond)*" é o conjunto de estruturas e instantes de tempo onde "F *cond*" é satisfeita, isto é, a partir de que instante "*cond*" pode ser satisfeita no futuro. Similarmente, "*Maintain(cond)*" é o conjunto de estruturas e instantes de tempo onde "G *cond*" é satisfeita, isto é, quando "*cond*" será sempre satisfeita no futuro. A diferença entre F e G, e *Achieve* e *Maintain*, é que no primeiro caso, eles se referem a condições, enquanto que no segundo caso, eles se referem a descrições de ações.

4.4.2 Semântica Formal

Define-se um modelo $\mathcal{M}$ como o conjunto de estruturas e uma atribuição de fórmulas atômicas a índices de tempo naquelas estruturas, sendo denotado por $\mathcal{M}$ = <**S**,**P**>. Cada estrutura $S \in \mathbf{S}$ é uma seqüência linear de índices de tempo $<t_0, t_1, ..>$. A função **P** atribui a cada índice de tempo o conjunto de fórmulas atômicas que são verdadeiras nele (os índices em estruturas diferentes são distintos). Denotamos que "a fórmula p é satisfeita na estrutura S no índice de tempo t " por $S,t \models p$. A fórmula p para a qual esta estrutura e índice de tempo podem ser encontrados, é denominada *satisfazível.* Uma fórmula que é satisfeita em todas as estruturas e em todos os índices de tempo é chamada *válida.* Podem então ser obtidas as seguintes definições [95][2]:

- $S,t \models p$, onde p é uma fórmula atômica, sse $p \in \mathbf{P}(t)$;
- $S,t \models \neg\ p$, sse $S,t \not\models$ p;
- $S,t \models p \vee q$ sse $S,t \models$ p ou $S,t \models$ q;
- $S,t \models p \wedge q$ sse $S,t \models$ p e $S,t \models$ q;
- $S,t \models p \rightarrow q$ sse $S,t \not\models$ p ou $S,t \models$ q;
- $S,t \models Xp$ sse $S,t' \models$ p, onde t' é o sucessor de t na estrutura S;
- $S,t \models Fp$ sse para algum t': $S,t' \models$ p, onde $t' > t$;
- $S,t \models Gp$ sse para todo t': $S,t' \models$ p, onde $t' > t$;
- $S,t \models pUq$ sse para algum t'': $S,t'' \models$ q, onde $t'' > t$, e para todo t': $t < t' < t''$: $S,t' \models p$.

Utilizando as descrições de ações apresentadas anteriormente, pode-se prover as seguintes condições que satisfazem as ações:

[2]sse = se e somente se.

- $S,t \models A$, onde A é um nome de ação, sse $S,t \models$ *condição_de_ação(A)*;
- $S,t \models$ *Achieve(p)* sse $S,t \models Fp$;
- $S,t \models$ *Maintain(p)* sse $S,t \models Gp$.

Essas condições de ações podem ser estendidas para a transmissão de mensagens. Podemos dizer que uma estrutura temporal linear satisfaz o envio de uma mensagem segundo um dado protocolo se e somente se aquela mensagem é efetivamente enviada naquele momento. O momento em que a mensagem chegará, e se ela chegará ao seu destino, dependerá das condições nas quais ela foi enviada e do protocolo utilizado. A recepção da mensagem imporá a capacidade de recepção dos agentes e a eventual ativação de uma rotina de processamento.

Na especificação a seguir, será utilizada ainda a notação $H(condição_1 \rightarrow condição_2)$ para expressar o estado interno do agente antes e depois da invocação de suas habilidades, isto é, antes e depois do processamento de suas ações.

4.5 Especificação da Arquitetura

4.5.1 Algoritmo

O algoritmo básico de processamento para esta arquitetura é o seguinte [72]:

1. Cada agente recebe informações através do meio de comunicação, que o atualizam sobre o estado do sistema, e em particular, sobre sua vizinhança;
2. Cada agente recebe informações de seus dispositivos locais (caso hajam), informando-o sobre o estado atual de sua área;
3. O Solucionador Local atualiza sua visão sobre o sistema, e informa os agentes vizinhos sobre seu estado e sua visão atuais, se necessário;

4. O Solucionador Local busca uma solução satisfatória para o sistema (ou, se possível, uma decisão globalmente ótima), baseado no estado do sistema e no conhecimento sobre o domínio da aplicação;

5. Se o Solucionador Local não estiver apto a tomar uma decisão razoável com o conhecimento que possui, solicita ajuda a um mecanismo auxiliar, como um Mecanismo Baseado em Casos;

6. O Mecanismo Baseado em Casos analisa a situação atual, e procura por casos passados similares. As decisões tomadas nesses casos são adaptadas à nova situação através de algoritmos genéticos, gerando uma decisão de controle;

7. Baseado nessa decisão, o Solucionador Local produz uma ação, podendo atuar sobre seus dispositivos locais, mandar mensagens ou requisitar ações dos outros agentes, por exemplo;

8. O Solucionador Local avalia o estado do mundo após a aplicação da decisão do Mecanismo Baseado em Casos. Ele informa o Mecanismo sobre os resultados da decisão, para que esse último possa aprender com ela e melhorar o desempenho do sistema ao longo do tempo.

Conhecimento e habilidades dos Agentes

1. Os agentes devem ser capazes de enviar e receber mensagens de outros agentes;

2. Os agentes devem ser capazes de receber informações e atuar sobre seus dispositivos locais, caso existam;

3. Os agentes devem ser capazes de armazenar e recuperar informações sobre o estado do sistema;

4. Os agentes devem ser capazes de determinar ao menos uma solução satisfatória para o problema, baseados no estado do sistema e no conhecimento sobre o domínio da aplicação;

5. Os agentes devem ser capazes de interagir com o Mecanismo Baseado em Casos e de analisar o desempenho dos casos;

6. O Mecanismo Baseado em Casos deve ser capaz de armazenar e recuperar casos, tomar decisões baseado nesses casos e analisar informações sobre desempenho, recebidas do Solucionador Local.

4.5.2 A estrutura da rede e os protocolos de comunicação

A seguir, são descritas a topologia da rede de comunicação e os enlaces que os agentes (nós) podem utilizar para seus propósitos de comunicação (ilustrado pela Figura 4.2, página 53). Os protocolos de comunicação estabelecem como os agentes podem comunicar-se, e de uma forma mais ampla, a organização existente entre os agentes, e a natureza e extensão da interação permitida entre eles.

Definimos $\mathcal{X} = \{N_1, N_2, .., N_k\}$ o conjunto de agentes do sistema, e $\mathcal{L}$=$\{(x_{i_1}, x_{j_1}), (x_{i_2}, x_{j_2}), .. (x_{i_m}, x_{j_m})\}$, $i_k, j_k \in \{1, 2, .., n\}$ o conjunto de enlaces de comunicação, definindo a topologia do sistema.

Os protocolos na arquitetura ICD serão especificados somente em termos da linguagem de comunicação permitida e dos agentes que a utilizam para comunicação. Outras propriedades, como atrasos nos canais de transmissão ou possíveis erros introduzidos nas mensagens, além da informação que é transmitida, são especificados através do uso dos predicados *Sends* e *Receives*, que correspondem respectivamente à ocorrência de ações *Send* e *Receive*. Isto é exatamente como o proposto em [95], com a única diferença que aqui são introduzidos, para propósito de transmissão em *multicast* e *broadcast*, o símbolo $\mathcal{N}$, indicando os vizinhos do nó que envia a mensagem, o símbolo $\mathcal{A}$, indicando todos os agentes do sistema, e o símbolo $\mathcal{LD}$, indicando os dispositivos locais controlados pelo nó. É introduzido ainda, para propósito de supervisão do sistema, o símbolo $\mathcal{S}$, que corresponde a um eventual nó de supervisão do sistema, que pode receber mensagens e enviar comandos a outros nós, se necessário. Dessa

forma, o predicado $Sends(N_1, \mathcal{A}, P, mensagem_1)$ avaliado no instante t_1 significa que o nó $N_1 \in \mathcal{X}$ enviou a mensagem "$mensagem_1$" no instante t_1 para todos os agentes do sistema ($\mathcal{A}$), segundo as regras do protocolo P. Os agentes capazes de entender o protocolo P recebem "$mensagem_2$" no instante t_2, onde "$mensagem_2$" é a mensagem recebida após o ruído do canal de comunicação, e $(t_2 - t_1)$ é o atraso no mesmo. As outras primitivas básicas de ação são dadas na Tabela 4.1.

PRIMITIVA	**DESCRIÇÃO DA AÇÃO**
N.atualiza(novo_valor,N.antigo_valor,P.slot_afetado)	Atualiza o "slot_afetado" no nó N com o novo valor.
N.executa(função,N.estado,N.restrições)	Executa a função determinada, dependendo do estado atual e das restrições existentes.
N.escalona(função,N.estado,N.prioridades)	Escalona a função determinada para aplicação futura, dependendo do estado e das prioridades do nó.
Ótimo(parâmetros_de_estado)	Determina e aplica uma solução satisfatória ou ótima para o domínio de aplicação do sistema.
Nova_decisão(objetivos_&_restrições,estado,novo_estado)	Define uma nova estratégia de resolução para o problema.
Observa(desempenho_MBC)	Observa o desempenho das decisões do Mecanismo Baseado em Casos, comparando o desempenho resultante ao anterior à aplicação das decisões.

Tabela 4.1: Primitivas básicas de ação do Solucionador Local

Podem ser transmitidos os seguintes tipos de mensagens:

- **mensagens de dados.** As mensagens de dados contêm informações sobre uma determinada parte do mundo, sobre o estado do nó remetente, ou sobre alguma decisão que ele tenha tomado. Esse tipo de mensagem pode alterar o estado do nó receptor, ou a sua visão do sistema.

  Por exemplo, um nó N_1 envia a mensagem de dados "$mensagem_1$" ao nó N_2, de acordo com o protocolo P: $Send(N_1,N_2,P,mensagem_1)$. O nó receptor N_2 recebe a mensagem e altera a sua visão do sistema:

$$H(Receive(N_2,N_1,P,mensagem_2)$$
$$\rightarrow X([N_2.atualiza]\ (mensagem_2,N_2.estado\_local,P.slot\_afetado))).$$

  Nesse exemplo, após o nó N_2 identificar "$mensagem_2$" como uma mensagem de dados, ele aplicará a função ($N_2.atualizar$), que modificará sua visão do mundo. P.slot_afetado é o seu dado que terá o valor modificado pela informação recebida. Essa atualização ocorrerá no próximo índice de tempo.

  Este tipo de mensagem de dados é o mesmo que os nós recebem de seus dispositivos locais.

- **mensagens de comando.** As mensagens de comando são descrições de ações que o nó receptor deve realizar. Embora não exista hierarquia entre os agentes (à possível exceção do nó supervisor), esse tipo de mensagem provê meios para controlá-los, tratando eventos especiais, como emergências, em alguns domínios de aplicação. Esse tipo de mensagem pode também ser aceito condicionalmente, respeitando restrições locais e da aplicação. Por exemplo, o nó N_1 envia uma mensagem de comando ao nó N_2: $Send(N_1,N_2,P,mensagem_1)$. O nó N_2 recebe a mensagem de comando e a executa:

$$H(Receive(N_2,N_1,P,mensagem_2)$$
$$\rightarrow X([N_2.executa]\ (mensagem_2,N_2.estado\_local,N_2.restrições))).$$

O nó N_2 identifica a mensagem de comando "$messagem_2$" e a executa, a menos que existam restrições à sua aplicação.

Os nós enviam comandos aos seus dispositivos locais através de mensagens de comando. Naturalmente, os protocolos devem variar de acordo com os dispositivos específicos existentes.

- **mensagens de requisição.** As mensagens de requisição contêm requisições de ações, sem a garantia de que sejam realizadas. O nó receptor decidirá, baseado em suas próprias prioridades, estratégias e estado, se e quando irá realizar a ação requisitada. Por exemplo, o nó N_1 envia a mensagem de requisição "$mensagem_1$" ao nó N_2: $Send(N_1,N_2,P,mensagem_1)$. O nó N_2 recebe a mensagem de requisição e escalona sua execução:

$$H(Receive(N_2,N_1,P,mensagem_2)$$
$$\rightarrow X([N_2.escalona]\ (mensagem_2,N_2.estado\_local,N_2.prioridades))).$$

O nó N_2 recebe a mensagem de requisição "$mensagem_2$" e pode decidir executá-la, baseado em sua próprias prioridades.

4.5.3 Habilidades dos nós

A seguir, será apresentada a especificação das habilidades dos nós: as tarefas do Solucionador Local, interface com dispositivos locais, visão do mundo, e interação com o Mecanismo Baseado em Casos (ilustrado pela Figura 4.3, página 54).

O Solucionador Local

O Solucionador Local recebe informações através dos enlaces da rede e de seus dispositivos locais. Ele pode decidir modificar sua visão do sistema ou mandar mensagens para outros nós, baseado nessas informações. Além disso, ele deve ser capaz de determinar uma solução satisfatória, e se possível, uma solução ótima para o problema. Deve

produzir decisões para controlar seus dispositivos locais, e informar os outros nós sobre seu estado atual, se necessário. Deve ainda ser capaz de identificar situações anormais, isto é, situações onde o nó não possui conhecimento suficiente para gerar uma solução satisfatória. Nesses casos, ele deve pedir ajuda a um mecanismo auxiliar, como um Mecanismo Baseado em Casos. Finalmente, deve ser capaz de receber decisões de controle do Mecanismo Baseado em Casos e prover informações de realimentação sobre o desempenho das decisões. Assim, o nó deve ser capaz de realizar as seguintes tarefas:

- **Atualização da visão do sistema.** O Solucionador Local recebe mensagens de dados informando o estado atual do sistema. Se uma porção do ambiente tiver se modificado significativamente desde a última atualização, ele atualiza a sua visão sobre o sistema, informando ainda seus nós vizinhos ($\mathcal{N}$) sobre o estado atual do sistema:

$$H(Receive\ (N_2,N_1,P,mensagem),$$
$$((—N_2.estado\_local - mensagem.novo\_valor—) > valor\_limite))$$
$$\rightarrow N_2.ações =$$
$$[N_2.atualiza]\ (mensagem,N_2.estado\_local,P.slot\_afetado),$$
$$(Send(N_2,\mathcal{N},P,mensagem))).$$

- **Determinação e aplicação de uma solução satisfatória.** A tarefa de determinar e aplicar uma solução ótima ou satisfatória é a mais difícil que o Solucionador Local deve realizar. Esta tarefa é dependente do domínio da aplicação, e cada uma das tarefas abaixo pode representar um grupo de tarefas, dependendo do domínio:

  - O Solucionador Local identifica que uma condição de término foi alcançada, e informa seus vizinhos e o eventual nó supervisor ($\mathcal{S}$) sobre isto:

$$H(Ótimo\_1([condição\_de\_término\_alcançada])$$
$$\rightarrow N.ações =$$
$$(Send(N,\mathcal{A},\ P,mensagem\_de\_condição\_de\_término)),$$

$$(Send(N,\mathcal{S},P,N.estado\_local))).$$

- O Solucionador Local identifica uma mudança significativa no ambiente, e decide alterar sua estratégia de resolução do problema. Ele faz com que seus dispositivos locais ($\mathcal{LD}$) sigam a nova estratégia adotada, e informa os nós vizinhos e o eventual nó supervisor sobre as mudanças:

$$H(Ótimo\_2([(—N.antigo\_valor - novo\_valor— > valor\_limite)])$$
$$\rightarrow N.ações =$$
$$(Achieve(Nova\_decisão\ (aplicação.conjunto\_de\_objetivos\_\&\_restrições,$$
$$N.antigo\_estado,\ novo\_estado)),$$
$$(Send(N,\mathcal{LD},P,N.nova\_decisão\_de\_controle)),$$
$$(Send(N,\mathcal{N},P,\ N.nova\_decisão\_de\_controle))$$
$$(Send(N,\mathcal{S},P,\ N.nova\_decisão\_de\_controle))).$$

- O Solucionador Local reconhece uma situação anormal e pede ajuda ao Mecanismo Baseado em Casos. O Solucionador Local recebe a nova decisão de controle, informa seus vizinhos e faz com que seus dispositivos locais apliquem a decisão. Além disso, ele observará o desempenho das decisões tomadas, informando o Mecanismo Baseado em Casos sobre elas, para que este último possa gerar soluções cada vez melhores:

$$H(Ótimo\_3([situação\_anormal(novo\_estado)])$$
$$\rightarrow N.ações =$$
$$(Send(N,\mathcal{MBC},PL,novo\_estado)),\ ^{3}$$
$$(Receive(N,\mathcal{MBC},P,nova\_decisão\_de\_controle)),$$
$$(Send(N,\mathcal{LD},P,N.nova\_decisão\_de\_controle)),$$
$$(Send(N,\mathcal{N},P,\ N.nova\_decisão\_de\_controle)),$$
$$(Achieve(Observa(desempenho\_MBC))),$$
$$(Send(N,\mathcal{MBC},P,desempenho\_MBC))).$$

[3]Embora o Mecanismo Baseado em Casos integre o nó, ele possui suas próprias habilidades. Assim, para o propósito de especificação, a comunicação entre o Solucionador Local e o Mecanismo Baseado em Casos foi modelada através de um protocolo local de comunicação (PL).

4.5.4 O Mecanismo Baseado em Casos

Embora o Mecanismo Baseado em Casos integre o nó, ele possui suas próprias habilidades. Assim, para o propósito de especificação, ele será considerado um módulo independente.

No Mecanismo Baseado em Casos proposto, utiliza-se uma base de casos estruturada como uma árvore, que classifica os casos de acordo com seus atributos. Cada atributo é visto como um gene em uma cadeia de DNA, e possui um alfabeto específico associado, que descreve os valores nebulosos que ele pode assumir (seu domínio). Cada caso é dividido em três partes: uma parte de atributos que identifica o caso, uma parte de ações que define as decisões tomadas, e uma parte de resultados que define o desempenho obtido por essas decisões [73].

Quando o Mecanismo Baseado em Casos encontra uma nova situação, busca na base de casos os casos mais similares à nova situação, baseado em seus atributos. Combina então as ações dos (dois) casos mais similares, dos casos que tenham obtido os melhores resultados, ou iterativamente, através de algoritmos genéticos, quando for possível calcular a aptidão dos *strings* (ou seja, as funções-objetivo para os indivíduos de cada geração). Como resultado, é gerada a parte de ações para a nova situação. Após a decisão ter sido aplicada, o Mecanismo Baseado em Casos obtém informações sobre o desempenho dessa decisão através do Solucionador Local. O Mecanismo Baseado em Casos preenche então a parte de resultados para o caso. Se houverem poucos casos do mesmo tipo (na mesma subárvore), ou se o desempenho do caso for bom em relação aos dos casos armazenados, o Mecanismo Baseado em Casos irá armazenar o novo caso. Caso hajam muitos casos na subárvore, ele irá ainda remover os casos mais antigos e de pior desempenho, visando melhorar continuamente o desempenho do sistema [78].

A seguir, será descrito o esquema básico de processamento para o Mecanismo Baseado em Casos evolutivo proposto, suas habilidades necessárias, sua comunicação com o Solucionador Local, e as tarefas que deve realizar.

O Algoritmo

O esquema básico de processamento para o Mecanismo Baseado em Casos é o seguinte:

1. O Mecanismo Baseado em Casos recebe informações do Solucionador Local, que identifica situações anormais;
2. O Mecanismo Baseado em Casos busca na base de casos os casos passados mais similares e que tenham obtido melhor desempenho. Essa busca é baseada nos atributos dos casos;
3. Combina então as ações tomadas nos casos selecionados através de algoritmos genéticos, para gerar um conjunto de decisões de ação para o novo caso;
4. Ele envia a decisão gerada ao Solucionador Local, e aguarda por informações sobre seu desempenho;
5. O Mecanismo Baseado em Casos recebe do Solucionador Local, informações sobre seu desempenho;
6. Ele analisa o desempenho obtido. Se o desempenho do caso em questão for bom em relação aos desempenhos dos casos armazenados, ou se houverem poucos casos similares na base de casos, irá armazenar o novo caso para permitir um melhor desempenho em suas futuras decisões;
7. Caso hajam muitos casos similares armazenados na base de casos, irá remover os casos mais antigos e de pior desempenho, para permitir uma reciclagem dos mesmos.

Conhecimento e habilidades necessárias

1. O Mecanismo Baseado em Casos deve ser capaz de enviar e de receber informações do Solucionador Local;

2. Deve ser capaz de armazenar e recuperar casos da base de casos, baseado em seus atributos;

3. Deve ser capaz de combinar situações passadas através de algoritmos genéticos e de gerar decisões para os casos;

4. Deve ainda ser capaz de analisar informações sobre desempenho, de compará-las com os dados armazenados, e de decidir quais casos armazenar e quais remover.

A Comunicação

A comunicação entre o Mecanismo Baseado em Casos e o Solucionador Local dá-se através de mensagens de comando (recebimento de um novo caso para tratar), mensagens de dados (envio de decisões), e mensagens de requisição (recebimento de resultados sobre desempenho para serem analisados), como definido na subseção 4.5.2.

As tarefas

As primitivas básicas de ação para o Mecanismo Baseado em Casos são mostradas na Tabela 4.2. As tarefas do Mecanismo Baseado em Casos são as seguintes:

- **Recuperação de casos similares.** O Mecanismo Baseado em Casos recebe informações sobre o estado atual do nó e do sistema, que identificam um novo caso. Para recuperar as experiências anteriores mais similares ou de maior sucesso, o Mecanismo Baseado em Casos primeiro seleciona uma subárvore ou um pequeno grupo de subárvores. Essa subárvore (ou grupo de subárvores) é aquela que

armazena os casos do mesmo tipo do atual. Esses casos são testados para identificar os mais similares ou de maior grau de sucesso. Sua representação interna é a seguinte:

$$H(Receive(\mathcal{MBC},\mathcal{LPS},P,novo\_caso)$$
$$\rightarrow MBC.ações =$$
$$(Seleciona\_subárvore(novo\_caso)),$$
$$(Calcula\_grau\_de\_proximidade\ (subárvores\_selecionadas,\ novo\_caso)),$$
$$(Seleciona\_melhores\_casos\ (casos\_mais\_similares,\ novo\_caso))).$$

- **Combinação das experiências selecionadas, gerando decisões de controle.** Após recuperar as experiências mais similares ou de maior sucesso, o Mecanismo Baseado em Casos combina-as através de algoritmos genéticos, gerando as decisões. Essas decisões são enviadas ao Solucionador Local para que sejam aplicadas. A representação interna do Mecanismo Baseado em Casos é a seguinte:

$$\rightarrow MBC.ações =$$
$$(Combina\_dependendo\_da\_estratégia\ (casos\_selecionados)),$$
$$(Send(\mathcal{MBC},\mathcal{LPS},P,decisão\_de\_controle)).$$

- **Informação sobre desempenho.** O Mecanismo Baseado em Casos deve utilizar informações sobre desempenho não só para escolher entre as experiências passadas, mas também para decidir se o novo caso vai ser armazenado e por quanto tempo. Primeiramente, recebe as informações sobre desempenho do Solucionador Local, e identifica a que caso essas informações referem-se. Calcula então o grau geral de desempenho para o caso, e busca a subárvore que armazena os casos do mesmo tipo. Baseado no número de casos armazenados naquela subárvore e no desempenho dos casos, decide se o novo caso será armazenado e por quanto tempo. O Mecanismo Baseado em Casos deve decidir ainda se existem casos para serem removidos, tal como casos muito antigos ou de baixo desempenho:

$$H(Receive\ (\mathcal{MBC}, \mathcal{LPS}, P, dados\_sobre\_desempenho),$$
$$\rightarrow MBC.ações =$$
$$(Identifica\_caso(dados\_sobre\_desempenho)),$$
$$(Calcula\_grau\_de\_desempenho(caso)),$$
$$(Seleciona\_subárvore(caso)),$$
$$(Decide\_se\_armazena(subárvore, caso)),$$
$$(Decide\_se\_remove(subárvore, casos\_armazenados))).$$

4.6 Questões de Sistemas Distribuídos

Embora não seja do escopo deste trabalho realizar uma análise formal da arquitetura do ponto de vista dos sistemas distribuídos, podem ser feitas algumas considerações:

- **Quanto ao acoplamento.** Um sistema ICD é um sistema fracamente acoplado, com dispersão física de seus processadores e comunicação eventualmente lenta;
- **Quanto à comunicação.** Os agentes de um sistema ICD comunicam-se através de mensagens, prevendo-se a existência de mensagens entre pares de agentes, *multicast* e *broadcast*. Obviamente, outras formas de comunicação poderiam ser também incorporadas à arquitetura;
- **Quanto à interação interprocessos.** Os processos são autônomos, sendo previstas formas de interação através de mensagens de comando;
- **Quanto à transparência.** Vários graus de transparência [33] podem ser providos, dependendo de cada aplicação. A arquitetura não fixa, *a priori*, as formas de transparência que devem existir nas aplicações;
- **Quanto à granularidade dos processos.** Por incluírem mecanismos de resolução de problemas a nível de aplicação e mecanismo baseado em casos, poderão haver processos de grão médio a grande.

- **Quanto à segurança e tolerância a falhas.** Questões de segurança e tolerância a falhas poderão ser melhor observadas em cada aplicação, pelo aumento de complexidade e custo computacional que envolvem. É provável que várias das questões envolvendo falhas de processadores, violações de tempo e falhas de informação possam ser resolvidas ou minimizadas através da inserção de conhecimento no Solucionador Local e através do Mecanismo Baseado em Casos.

4.7 Resumo

Neste capítulo foi apresentado um novo paradigma para o desenvolvimento de sistemas inteligentes distribuídos, denominado Inteligência Computacional Distribuída (**ICD**). Além de sua descrição global, da descrição de seus componentes e funcionamento, foi apresentada também sua especificação formal em linguagem declarativa de especificação.

A abordagem apresentada instrumenta-se com a teoria dos conjuntos nebulosos, os sistemas baseados em casos e algoritmos genéticos, podendo resolver problemas reais complexos, onde seja necessário manipular informações e conhecimento incertos, prover eficiência computacional, flexibilidade e adaptabilidade.

No próximo capítulo, será apresentado um exemplo de aplicação para essa arquitetura, envolvendo o controle de um sistema viário urbano, onde será exemplificado o funcionamento dos mecanismos aqui propostos em um sistema real.

PRIMITIVA	DESCRIÇÃO DA AÇÃO
Seleciona_subárvore(caso)	Seleciona a subárvore que contém os casos do mesmo tipo do caso especificado.
Calcula_grau_de_proximidade(subárvore, caso)	Avalia o quão similar é o caso novo com relação aos casos armazenados.
Seleciona_melhores_casos(casos_similares, caso)	Seleciona os casos mais similares, de maior sucesso ou através de suas funções-objetivo para combinação.
Combina_dependendo_da_estratégia(casos)	Combina os casos selecionados, tomando-os por atributos genéricos ou "gene a gene", gerando as decisões.
Identifica_caso(desempenho_do_caso)	Identifica o caso ao qual os resultados de desempenho referem-se.
Calcula_grau_de_desempenho(caso)	Calcula o grau geral de desempenho para aquele caso.
Decide_se_armazena(subárvore,caso)	Decide se armazena o caso na subárvore, dependendo do seu desempenho e do número de casos na subárvore.
Decide_se_remove(subárvore,caso)	Decide se remove o caso da subárvore, dependendo do seu desempenho, tempo de permanência e do número de casos na subárvore.

Tabela 4.2: Primitivas básicas de ação do Mecanismo Baseado em Casos

Capítulo 5

Controle de Tráfego Urbano

5.1 Introdução

O problema do controle de tráfego urbano é cada vez mais crítico, especialmente nas grandes cidades. Além da perda de tempo e desperdício de combustível, existem ainda o aumento da poluição atmosférica causada pela emissão de gases, estresse e irritação, diminuindo a segurança de motoristas e pedestres, dentre muitos outros. O problema envolve diversos fatores logísticos e humanos, tornando-o bastante complexo. Quaisquer melhorias de desempenho e uma maior adaptabilidade dos sistemas de controle aos inevitáveis e (quase) imprevisíveis acidentes e congestionamentos, além de novas estratégias de controle, podem contribuir muito para esse problema que envolve milhões de pessoas todos os dias.

O Sistema Distribuído de Controle de Tráfego Urbano aqui apresentado tem por objetivo mostrar a aplicabilidade e o desempenho potencial de um sistema ICD na resolução de problemas distribuídos e dinâmicos. O sistema consiste de um proces-

sador situado em cada cruzamento com semáforos, comunicando-se somente com os processadores vizinhos, e decidindo as ações dos semáforos desse cruzamento. Assume-se a existência de sensores, que fornecem dados relativos aos tamanhos das filas e os fluxos de tráfego em cada via incidente no cruzamento. Esses sensores enviam os dados coletados ao processador local. O objetivo do sistema é otimizar o fluxo do tráfego, reduzindo os tempos de espera dos veículos e os tamanhos das filas, melhorando a qualidade do tráfego urbano [74, 76].

O mapeamento da aplicação na arquitetura proposta é bastante simples: os processadores correspondem aos nós do sistema, as ligações com os vizinhos estabeleçam a topologia do sistema, os sensores e os semáforos correspondem aos dispositivos locais, e a tarefa de melhorar a qualidade do tráfego será realizada pelo Solucionador Local e o Mecanismo Baseado em Casos.

Para estudar esse problema no âmbito da ICD, considerou-se a área central da cidade de Campinas - SP, a partir de sua rede de cruzamentos com semáforos, mostrada na Figura 5.1.

Uma vez que a especificação da arquitetura ICD e dos protocolos de comunicação são independentes do domínio de aplicação, será enfatizada aqui apenas a aplicação propriamente dita: a representação do conhecimento nesse domínio e as tarefas dos nós que são específicas para esse problema. Também a especificação do Mecanismo Baseado em Casos segue àquela apresentada na subseção 4.5.4. Os detalhes de sua implementação serão discutidos na Seção 5.4.

A seguir, serão apresentados o problema propriamente dito, o conhecimento utilizado em sua resolução, as tarefas específicas dos nós, detalhes de implementação, os resultados obtidos pelo protótipo implementado, e a análise dos resultados.

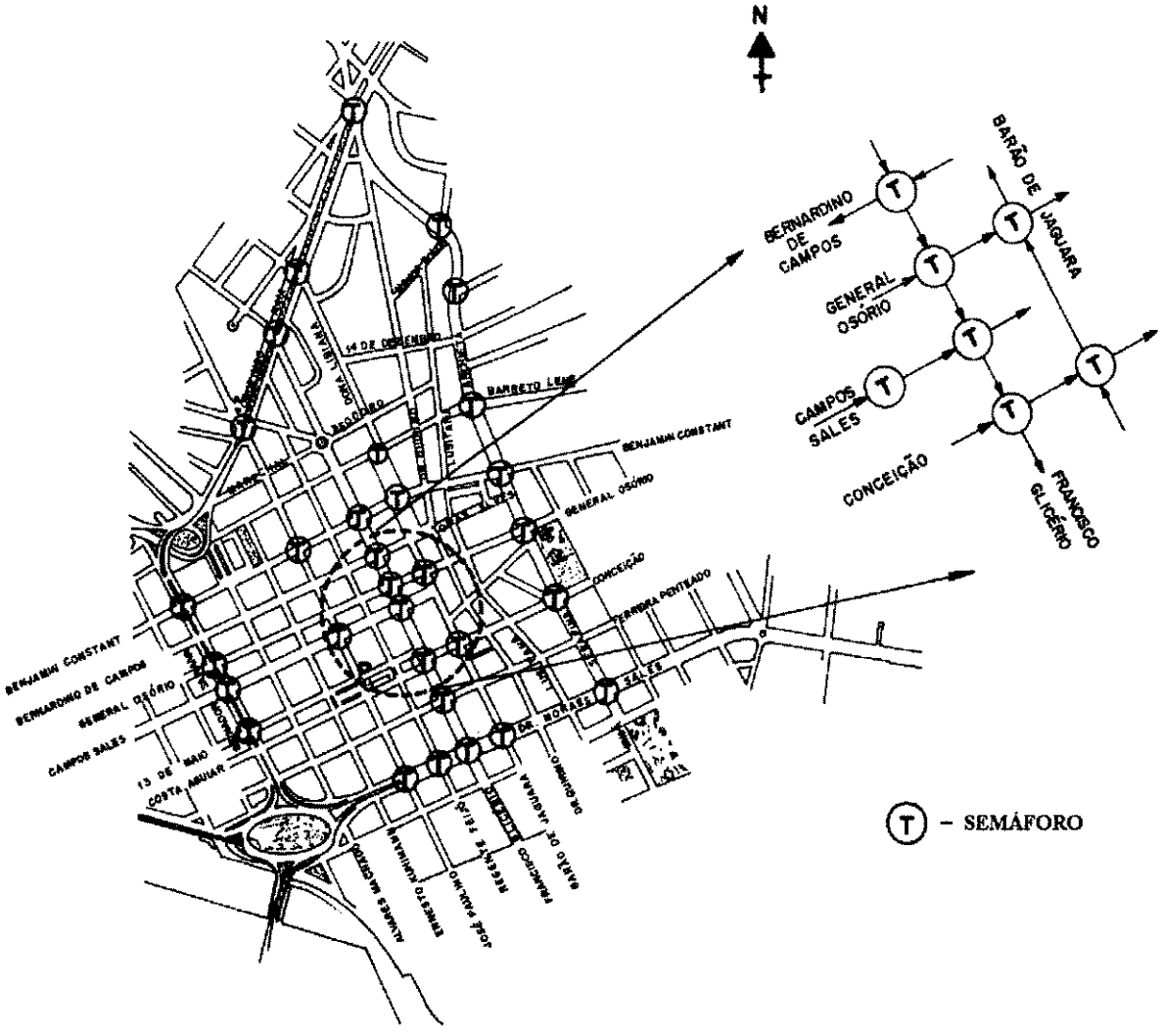


Figura 5.1: Área central de Campinas-SP e sua rede de cruzamentos com semáforos

5.2 Especificação do Problema de Aplicação

- **Estado Inicial.**

  Inicialmente, todos os semáforos piscam em amarelo (esta é uma restrição de segurança internacionalmente aceita - e exigida - para os sistemas que controlam semáforos). Da mesma forma, sempre que haja uma falha e os semáforos tenham que reinicializar sua operação, voltam a piscar em amarelo.

[*(amarelo,piscante)] [1]

- **Transições Legais de Estado.**

  1. [(amarelo,piscante)] → [(verde,normal)].
  2. [(amarelo,piscante)] → [(vermelho,normal)].
  3. [(verde,normal)] → [(amarelo,normal)].
  4. [(verde,normal)] → [(amarelo,piscante)].
  5. [(verde,normal)] → [(vermelho,normal)].
  6. [(verde,normal)] → [(vermelho,piscante)].[2]
  7. [(amarelo,normal)] → [(verde,normal)].
  8. [(amarelo,normal)] → [(vermelho,normal)].
  9. [(amarelo,normal)] → [(amarelo,piscante)].
  10. [(vermelho,normal)] → [(verde,normal)].
  11. [(vermelho,normal)] → [(amarelo,normal)].
  12. [(vermelho,normal)] → [(amarelo,piscante)].

- **Restrições.**

  Nas restrições a seguir, denotamos N^c o semáforo N que controla o fluxo de carros e N^p o semáforo que controla o fluxo de pedestres.

  1. Apenas um semáforo pode estar em verde em cada cruzamento.

$$\nexists N_i^c, N_j^c \in \{\text{semáforos no mesmo cruzamento } \} \mid N_i : (verde, \_) \wedge N_j : (verde, \_), i \neq j.$$

[1] O símbolo '*' representa "repetição", e o símbolo '_' representa "não importa".

[2] Para pedestres.

2. Se o semáforo para carros estiver em verde, seu semáforo para pedestres associado (se existir) não poderá estar em verde.

$$N_i^c : (verde, \_) \Rightarrow N_i^p : \neg(verde, \_).$$

3. Um semáforo não pode permanecer indefinidamente em verde. O tempo de máximo de verde dependerá de cada semáforo e cruzamento específicos.

$$(N_i : (verde, \_)) \rightarrow Achieve(N_i : (vermelho, \_)) \vee Achieve(N_i : (amarelo, \_)).$$

4. Similarmente para o vermelho:

$$(N_i : (vermelho, \_)) \rightarrow Achieve(N_i : (verde, \_)) \vee Achieve(N_i : (amarelo, \_)).$$

5. Similarmente para o amarelo:

$$(N_i : (amarelo, \_)) \rightarrow Achieve(N_i : (verde, \_)) \vee Achieve(N_i : (vermelho, \_)).$$

6. Uma vez que tenha passado para o estado de verde, o semáforo deve permanecer em verde por pelo menos t segundos. O tempo de mínimo de verde dependerá de cada semáforo e de cada cruzamento específicos.

$$(N_i : (verde, \_)) \rightarrow Maintain_t(N_i : (verde, \_)).^3$$

7. Similarmente para o vermelho:

$$(N_i : (vermelho, \_)) \rightarrow Maintain_t(N_i : (vermelho, \_)).$$

8. Similarmente para o amarelo:

$$(N_i : (amarelo, \_)) \rightarrow Maintain_t(N_i : (amarelo, \_)).$$

[3] O uso de *Maintain* foi estendido, de forma que $Maintain_t$ corresponde a *Maintain* por t segundos, e $Maintain_e$ corresponde a *Maintain* até a ocorrência do evento e.

9. Para cada cruzamento, deve haver um grupo de seqüências de estados para o operação dos semáforos, tal como a seqüência abaixo: [4]

$$\{(N_1 : (verde, normal), N_2 : (vermelho, normal))\} \rightarrow$$
$$\{Achieve(N_1 : (amarelo, normal), N_2 : (vermelho, normal))\} \rightarrow$$
$$\{Achieve(N_1 : (vermelho, normal), N_2 : (verde, normal))\} \rightarrow$$
$$\{Achieve(N_1 : (vermelho, normal), N_2 : (amarelo, normal))\} \rightarrow$$
$$\{Achieve(N_1 : (verde, normal), N_2 : (vermelho, normal))\}.$$

- **Comunicação.**

  A comunicação dá-se da forma estabelecida na subseção 4.5.2. Ressalta-se a presença de um nó supervisor ($\mathcal{S}$), que recebe informações de todos os outros nós, e mantém uma visão razoavelmente atualizada do sistema. Através desse nó, é possível supervisionar o sistema, enviando mensagens e comandos para os nós remotos, e eventualmente enviando pessoal técnico para as regiões com problemas.

- **Objetivos.**

  Os objetivos do sistema são minimizar os tamanhos médios das filas de veículos e maximizar o fluxo de tráfego na área urbana, melhorando a sua qualidade.

A seguir, é apresentado um modelo do conhecimento que os nós devem possuir para realizar suas tarefas de controle em consonância com os objetivos estabelecidos.

5.2.1 Conhecimento sobre Direção.

O conhecimento sobre direção permite que os nós realizem suas tarefas com mais eficiência, comparando direções e raciocinando sobre o sentido de fluxo.

[4] Os semáforos para pedestres associados foram omitidos para maior clareza.

- mesma_direção(d,d).
- direção_similar {(N,NE), (N,NW), (S,SE), (S,SW), (E,NE), (E,SE), (W,NW), (W,SW) (NE,N), (NW,N), (SE,S), (SW,S), (NE,E), (SE,E), (NW,W), (SW,W)}.
- direção_oposta {(N,S), (E,W), (NE,SW), (SE,NW) (S,N), (W,E), (SW,NE), (NW,SE)}.
- direção_quase_oposta {(N,SE), (N,SW), (S,NE), (S,NW), (E,NW), (E,SW), (W,NE), (W,SE) (SE,N), (SW,N), (NE,S), (NW,S), (NW,E), (SW,E), (NE,W), (SE,W)}.
- direção_lateral_direita {(N,W), (S,E), (E,N), (W,S), (SE,NE), (NE,NW), (NW,SW), (SW,SE) }.
- direção_lateral_esquerda {(E,S), (W,N), (S,W), (N,E), (SW,NW), (SE,SW), (NE,SE), (NW,NE) }.
- direção_quase_lateral_direita {(N,NW), (N,SW), (S,NE), (S,SE), (E,NW), (E,NE), (W,SW), (W,SE), (SE,E), (SE,N), (NE,W), (NE,N), (NW,S), (NW,W), (SW,E), (SW,S)}.
- direção_quase_lateral_esquerda {(N,NE), (N,SE), (S,NW), (S,SW), (E,SW), (E,SE), (W,NW), (W,NE), (SE,W), (SE,S), (NE,E), (NE,S), (NW,N), (NW,E), (SW,W), (SW,N) }.

5.2.2 Visão do nó

Cada nó possui visão detalhada e atualizada sobre seu cruzamento, recebendo as informações dos sensores locais. Possui também visão detalhada sobre sua vizinhança e sobre eventuais outros nós cujas ações interajam diretamente às do nó local. Esse tipo de interação ocorre, por exemplo, em avenidas e corredores de tráfego. Finalmente, os nós mantêm também uma visão geral do sistema e sua posição nessa estrutura, além de conhecerem o endereço do nó supervisor ($\mathcal{S}$), de forma a poder enviar-lhe mensagens e identificar mensagens onde ele é o remetente.

1. **Estado Local.**

   { *(*{ ruas, direção, tamanho_atual_da_fila, fluxo_de_tráfego_atual, *condição_anormal}* [5], *dia_da_semana, horário, *evento_especial) ≡ (*{ *nome_rua, valor_direção, tamanho_fila, fluxo_tráfego, *cond_anormal}, dia_semana, valor_hora, *evento_esp)* }

   onde:

   nome_rua ∈ { Rua_Alvares_Machado, Av_Anchieta,.. }

   valor_direção ∈ { N, S, E, W, NE, NW, SE, SW }

   tamanho_fila ∈ { muito_pequena, pequena, normal, grande, muito_grande }

   fluxo_tráfego ∈ { livre, leve, normal, pesado, muito_pesado }

   cond_anormal ∈ { falha_de_semáforo, falha_de_processador, acidente,.. }

   dia_semana ∈ { domingo, segunda, terça, quarta, quinta, sexta, sábado }

   valor_hora ∈ { 0.00.00 .. 23.59.99 }

   evento_esp ∈ { feriado, férias, jogo, desfile, .. }

2. **Nó Supervisor.**

   O endereço de comunicação dos nós, e em particular do nó supervisor, é definido de acordo com os tipos de acesso permitidos à rede e à linguagem de programação utilizada, como por exemplo, através do seu endereço *(host)* e a porta de comunicação: $< Host, Port >$

3. **Nós vizinhos e relacionados.**

   { *(*{ vizinho, ruas, direção, tamanho_atual_da_fila, fluxo_de_tráfego_atual, *condição_anormal, *prioridade }) ≡ *{ endereço_vizinho, *nome_rua, valor_direção, tamanho_fila, fluxo_tráfego, *cond_anormal, *valor_prioridade })* }

   onde:

   endereço_vizinho: $(< nome\_ruaXnome\_rua >, < Host, Port >)$

   valor_prioridade ∈ { sem_prioridade,baixa_prioridade, média_prioridade, alta_prioridade}

[5]Para cada direção de fluxo no cruzamento.

4. **Visão da cidade.**

   { *(região, direção_da_região, condição_de_tráfego, *evento_especial, *prioridade)* ≡ *(nome_região, valor_direção, fluxo_tráfego, *evento_esp, *valor_prioridade)* }

   onde:

   nome_região ∈ { Guanabara, Prefeitura, Maternidade .. }

5. **Estrutura da Rede**

   { **(vizinho_ou_nó_relacionado, posição_relativa, região, direção_região)* ≡ **(endereço_vizinho, valor_direção, nome_região, valor_direção)* }

5.3 Tarefas dos Nós

As tarefas genéricas que os nós devem realizar estão descritas na subseção 4.5.3. As tarefas específicas para o domínio de aplicação incluem:

Direcionamento do tráfego de áreas mais congestionadas para áreas mais livres.

1. Na mesma direção. Se o tráfego local estiver mais pesado que o tráfego de sua vizinhança, e o sentido do tráfego for do nó local para os vizinhos com tráfego mais leve, o tempo de verde local será aumentado para incentivar a saída dos veículos. A decisão tomada será informada aos vizinhos e ao nó supervisor.

*H(N:(ruas, direção_fluxo, fila, fluxo_tráfego, *condição_anormal),*

Vizinhança:(vizinho, *rua_2, posição, direção_fluxo, fila_2, fluxo_tráfego_2, *condição_anormal_2),*

*(fluxo_de_tráfego > *fluxo_tráfego_2),*

(posição = direção_fluxo)

→ N.ações =

(calcula_incremento_verde),

$(Maintain_{t+incremento}(N{:}(verde,normal))),$

$(Send(N,\mathcal{LD},P,nova\_decisão\_controle)),$

$(Send(N,\mathcal{N},P,mensagem)),$

$(Send(N,\mathcal{S},P,mensagem))).$

2. Em direções parecidas ou corredores de tráfego. Similar ao anterior, mas englobando também direções similares:

$H(N{:}(ruas,\ direção\_fluxo,\ fila,\ fluxo\_tráfego,\ *condição\_anormal),$

$Vizinhança{:}*(vizinho,\ *rua\_2,\ posição,\ direção\_fluxo\_2,\ fila\_2,\ fluxo\_tráfego\_2,$
$*condição\_anormal\_2),$

$direção\_similar(direção\_fluxo,\ direção\_fluxo\_2) \vee corredor\_tráfego(rua,$
$direção\_fluxo,\ rua\_2,\ direção\_fluxo\_2)$

$(fluxo\_tráfego > *fluxo\_tráfego\_2)$

$\rightarrow N.ações =$

$(calcula\_incremento\_verde),$

$(Maintain_{t+incremento}\ (N{:}(verde,normal))),$

$(Send(N,\mathcal{LD},P,nova\_decisão\_controle)),$

$(Send(N,\mathcal{N},P,mensagem)),$

$(Send(N,\mathcal{S},P,mensagem))).$

3. Antecipar o tráfego que chega. Se o fluxo de tráfego deve aumentar (vindo dos nós vizinhos), o tempo de verde local pode ser aumentado para minimizar as futuras filas de veículos, antecipando o fluxo de tráfego que chega. Os nós vizinhos e o nó supervisor são informados:

$H(N{:}(ruas,\ direção\_fluxo,\ fila,\ fluxo\_tráfego,\ *condição\_anormal),$

$Vizinhança{:}*(vizinho,\ *rua\_2,\ posição,\ direção\_fluxo\_2,\ fila\_2,\ fluxo\_tráfego\_2,$
$*condição\_anormal\_2),$

direção_oposta(posição, direção_fluxo_2) $\vee$ *direção_quase_oposta (posição, direção_fluxo_2)* $\vee$ *corredor_tráfego(ruat_2,direção_fluxo_2, rua, direção_fluxo),*

(fluxo_tráfego_2 $>$ *fluxo_tráfego)*

$\rightarrow$ *N.ações* $=$

(calcula_incremento_de_verde),

$(Maintain_{t+incremento}$ *(N:(verde,normal))),*

(Send(N,$\mathcal{LD}$*,P,nova_decisão_controle)),*

(Send(N,$\mathcal{N}$*,P,mensagem)),*

(Send(N,$\mathcal{S}$*,P,mensagem))).*

4. Direciona o tráfego das laterais. Se o tráfego nas laterais é mais intenso e deve incidir no cruzamento, o tempo de verde na direção principal é diminuído (e conseqüentemente, o tempo de verde da lateral é aumentado), para suportar melhor o tráfego incidente:

*H(N:(ruas, direção_fluxo, fila, fluxo_tráfego, *condição_anormal),*

*Vizinhança: *(vizinho, rua_2, posição, direção_fluxo_2, fila_2, fluxo_tráfego_2, *condição_anormal_2)* ,

direção_lateral_direita(direção_fluxo, direção_fluxo_2) $\vee$
direção_quase_lateral_direita (direção_fluxo, direção_fluxo_2) $\vee$
direção_lateral_esquerda(direção_fluxo, direção_fluxo_2) $\vee$
direção_quase_lateral_esquerda (direção_fluxo, direção_fluxo_2),

(fluxo_tráfego_2 $>$ *fluxo_tráfego)*

$\rightarrow$ *N.ações* $=$

(calcula_decremento_de_verde),

$(Maintain_{t-decremento}$ *(N:(verde,_))),*

(Send(N,$\mathcal{LD}$*,P,nova_decisão_controle)),*

$(Send(N,\mathcal{N},P,mensagem)),$

$(Send(N,\mathcal{S},P,mensagem))).$

5. Na falta de conhecimento, solicita auxílio ao Mecanismo Baseado em Casos. Quando o Solucionador Local identifica uma situação anormal, solicita auxílio ao Mecanismo Baseado em Casos. O Solucionador Local o informa sobre as condições locais de tráfego e recebe as novas decisões de controle. Ele aplica as novas decisões nos semáforos locais, informa seus vizinhos e o nó supervisor sobre as novas decisões. Observa então o desempenho dessas decisões para enviar ao Mecanismo Baseado em Casos, para que esse último possa melhorar o desempenho de suas decisões ao longo do tempo:

$H(N{:}(rua,\ direção\_fluxo,\ fila,\ fluxo\_tráfego,\ {*}condição\ anormal:\ [acidente,..]))$

$\rightarrow N.ações =$

$(Send(N,\mathcal{MBC},P,mensagem)),$

$(Receive(N,\mathcal{MBC},P,nova\_decisão\_controle)),$

$(Send(N,\mathcal{LD},P,nova\_decisão\_controle)),$

$(Send(N,\mathcal{N},P,mensagem\_2)),$

$(Send(N,\mathcal{S},P,mensagem\_3)),$

$(Achieve(Observa(desempenho\_decisão\_cbm))),$

$(Send(N,\mathcal{MBC},P,desempenho\_decisão\_cbm))).$

5.4 Implementação

Um protótipo para o Sistema de Controle de Tráfego foi desenvolvido em microcomputador do tipo PC-486, em linguagem de programação *C* [51].

Foram implementadas também as estratégias clássicas de ciclo fixo e ciclo fixo progressivo (seqüencial), além da abordagem ICD para propósitos comparativos. Em todos

os casos, foi utilizada a mesma rede de cruzamentos e os mesmos parâmetros, tais como o fluxo de veículos e movimentos de conversão, baseados em [63] e em observações de tráfego na região central da cidade de Campinas-SP (ver Apêndice B).

Nesta seção, serão descritos aspectos específicos da implementação, necessários para o entendimento do sistema sob o ponto de vista de aplicações.

5.4.1 Solucionador Local

O propósito do Solucionador Local nesta aplicação é controlar o fluxo de tráfego em seu cruzamento. Basicamente, ele compara os fluxos de entrada e as filas locais e dos cruzamentos relacionados a ele, tais como dos cruzamentos vizinhos. Essa informação é obtida através de seus sensores e de informações recebidas através da rede. Baseado nessa informação, o Solucionador Local deve decidir sobre a extensão do seu tempo de verde e avisar os processadores vizinhos sobre a sua decisão. Essa estratégia é utilizada sempre que as condições do tráfego forem similares àquelas utilizadas para ajustar os semáforos, de acordo com [108]. Nesses casos, o Solucionador Local ajusta os tempos previamente estabelecidos.

Após obter os dados dos sensores locais e receber informações de outros nós, o Solucionador Local converte eventuais valores numéricos recebidos em valores lingüísticos. Exemplos para as variáveis lingüísticas *Fluxo de Tráfego*, *Fila*, *Tempo de Espera* e *Extensão* são mostrados nas Figuras 5.2 a 5.5. De posse das novas informações, o Solucionador Local pode alterar a sua visão do sistema. Caso não receba novas informações, trabalhará com sua visão mais recente do mesmo.

O Solucionador Local utiliza informações sobre o fluxo de tráfego incidente em cada cruzamento, além dos tamanhos de fila das laterais e do cruzamento posterior para calcular a variação do tempo de verde [75]. O tráfego incidente é classificado em "livre", "leve", "normal", "pesado" e "congestionado". Como exemplo, vamos considerar as regras de controle nebulosas ilustradas pela Tabela 5.1, referente a uma

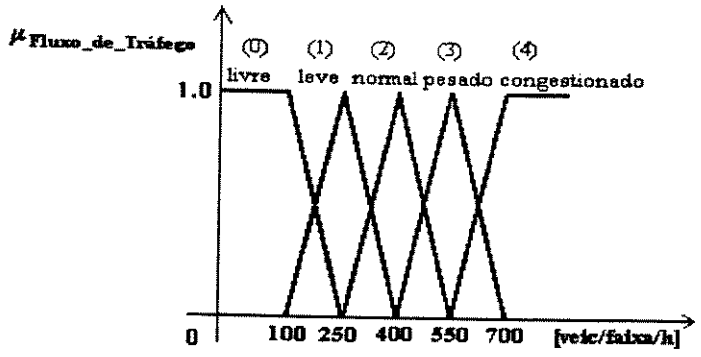


Figura 5.2: Valores da variável lingüística *Fluxo de Tráfego*

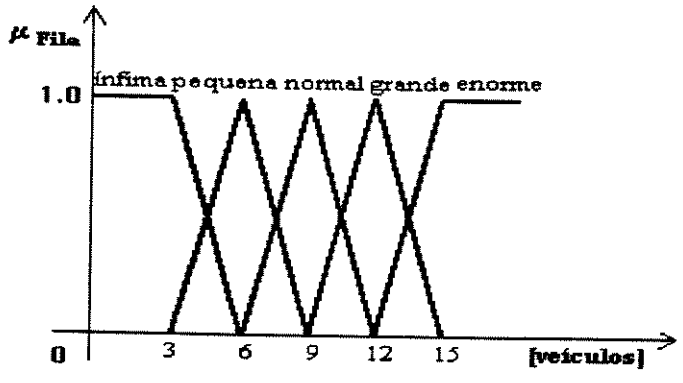


Figura 5.3: Valores da variável lingüística *Fila*

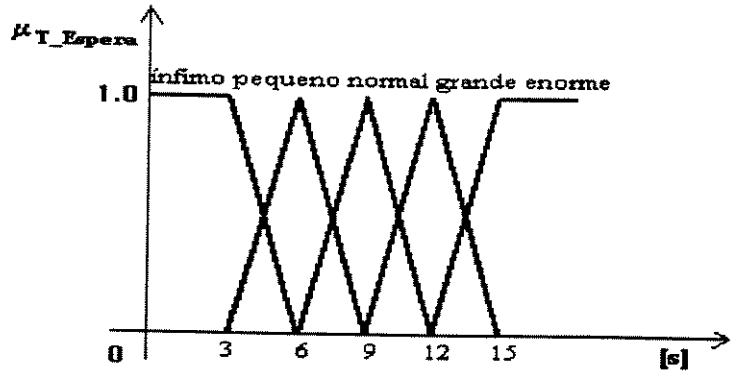


Figura 5.4: Valores da variável lingüística *Tempo de Espera*

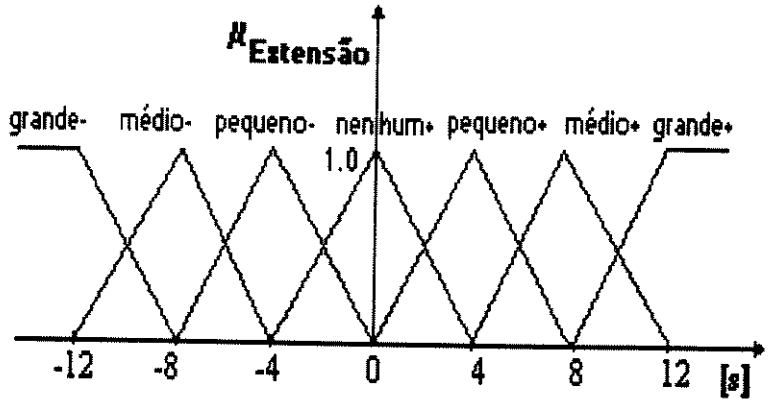


Figura 5.5: Valores da variável lingüística *Extensão.*

fila "normal" e a um fluxo de tráfego "leve" incidindo no cruzamento, para a via principal.

Para a tomada de decisão, os conectivos *e* e *ou* são interpretados como operadores *min* e *max*, respectivamente. Para a *defuzificação* e obtenção da extensão do tempo de verde, foram tomadas as funções de pertinência da variável "Extensão" (do tempo de verde), e calculado o centro de sua área.

O Solucionador Local adota a estratégia da Tabela 5.1 quando puder gerar uma solução eficaz para a situação do tráfego, isto é, quando observar estados similares aos esperados, como filas médias nos semáforos. Quando as condições de tráfego forem

Fila Lateral / **Fila Cruzamento Posterior**	ÍNFIMA	PEQUENA	NORMAL	GRANDE	ENORME
ÍNFIMA	médio_inc	pequeno_inc	pequeno_inc	pequeno_decr	médio_decr
PEQUENA	médio_inc	pequeno_inc	nenhum	pequeno_decr	médio_decr
NORMAL	pequeno_inc	pequeno_inc	nenhum	pequeno_decr	médio_decr
GRANDE	pequeno_inc	nenhum	nenhum	pequeno_decr	médio_decr
ENORME	nenhum	nenhum	pequeno_decr	médio_decr	médio_decr

Tabela 5.1: Incremento/decremento do tempo de verde em uma via principal

diferentes das esperadas, como em casos de tráfego congestionado ou filas grandes ou enormes, o Mecanismo Baseado em Casos é acionado para tomar uma decisão de controle.

Exemplo de Operação

Para exemplificar a operação do Solucionador Local, consideremos:

- Fila local: 9 veículos;
- Fluxo de tráfego local: 250 veículos/faixa/hora;
- Fila lateral: 6 veículos;
- Fila do cruzamento posterior: 10 veículos.

A partir das Figuras 5.3 e 5.2, observamos que uma fila local de 9 veículos pode ser considerada uma fila "normal" com grau de pertinência igual a 1 ($\mu_{Normal}(Fila\_Local = 9) = 1$), e que o fluxo de 250 veículos/faixa/hora pode ser considerado um fluxo "leve" com grau de pertinência igual a 1 ($\mu_{Leve}(Fluxo\_Tráfego\_Local = 250) = 1$). Isso nos leva a considerar a Tabela 5.1 para determinar a extensão do tempo de verde local. Regras de outras tabelas (para filas "grandes" ou "enormes", por exemplo) não afetarão a saída do sistema e não serão mostradas para maior clareza.

A Tabela 5.1 possui vinte e cinco regras na forma:

Regra 1: *SE Fila_Lateral é Ínfima e Fila_Cruzamento_Posterior é Ínfima ENTÃO Extensão é médio_incremento;*

Regra 2: *SE Fila_Lateral é Ínfima e Fila_Cruzamento_Posterior é Pequena ENTÃO Extensão é médio_incremento;*

Regra 3: *SE Fila_Lateral é Ínfima e Fila_Cruzamento_Posterior é Normal ENTÃO Extensão é pequeno_incremento;*

:

Regra 8: *SE Fila_Lateral é Pequena e Fila_Cruzamento_Posterior é Normal ENTÃO Extensão é pequeno_incremento;*

Regra 9: *SE Fila_Lateral é Pequena e Fila_Cruzamento_Posterior é Grande ENTÃO Extensão é nenhum_incremento;*

:

Regra 25: *SE Fila_Lateral é Enorme e Fila_Cruzamento_Posterior é Enorme ENTÃO Extensão é médio_decremento.*

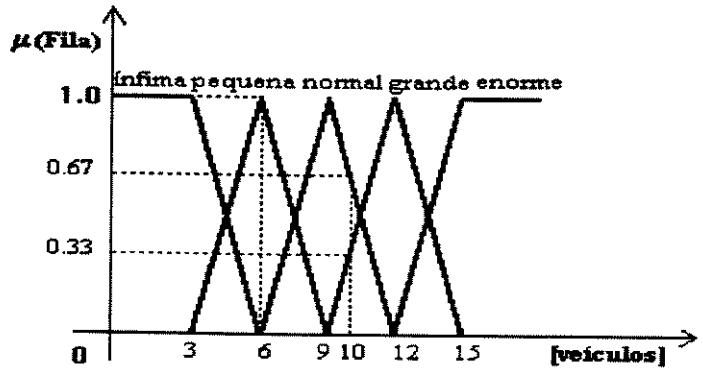


Figura 5.6: Graus de pertinência dos valores de *Fila*

Considerando ainda os graus de pertinência das filas lateral e lateral posterior para a aplicação das regras, temos a seguinte classificação das variáveis de entrada com relação às respectivas variáveis lingüísticas: $(\mu_{Pequena}(Fila\_Lateral = 6) = 1)$, $(\mu_{Normal}(Fila\_Cruzamento\_Posterior = 10) = 0.67)$ e $(\mu_{Grande}(Fila\_Cruzamento\_Posterior = 10) = 0.33)$, conforme ilustra a Figura 5.6.

Feita essa classificação, o procedimento de inferência (lógica de decisão) deve avaliar as regras, identificando aquelas que se aplicam à situação [32]. Para Fila_Lateral=6 e Fila_Cruzamento_Posterior=10, as regras que se aplicam são as seguintes:

Regra 8: SE Fila_Lateral é Pequena e Fila_Cruzamento_Posterior é Normal
ENTÃO Extensão é pequeno_incremento;

Regra 9: SE Fila_Lateral é Pequena e Fila_Cruzamento_Posterior é Grande
ENTÃO Extensão é nenhum_incremento;

Como as regras possuem dois antecedentes relacionados pelo conectivo *e* (intersecção), definindo-se o operador intersecção como sendo $\wedge$ (min), obtém-se:

Regra 8: $\min(\mu_{Pequena}(Fila\_Lateral = 6), \mu_{Normal}(Fila\_Cruzamento\_Posterior = 10)) = (1) \wedge (0.67) = 0.67$

Regra 9: $\min(\mu_{Pequena}(Fila\_Lateral = 6), \mu_{Grande}(Fila\_Cruzamento\_Posterior = 10)) = (1) \wedge (0.33) = 0.33$

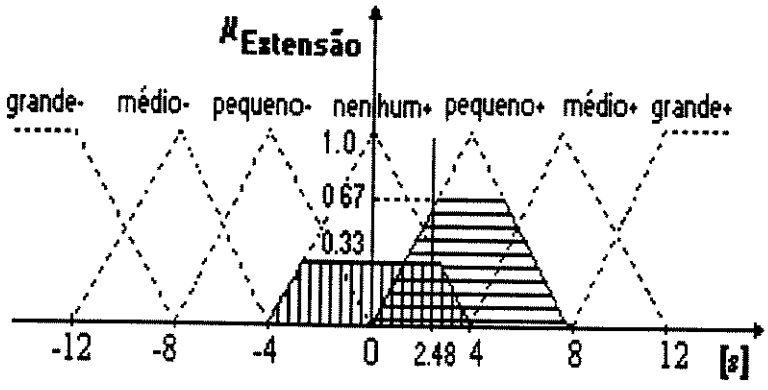


Figura 5.7: Defuzificação por centro de área

Para cada regra, o grau de ativação da ação de controle é calculada de acordo com o resultado da combinação de antecedentes. Se a regra de inferência é *max-min*, o resultado da inferência, para uma regra, é obtido pelo mínimo entre o grau de combinação dos antecedentes e conseqüentes. A ação de controle é calculada a partir da união das contribuições proporcionadas por cada regra ativada. Se a união é definida como sendo o operador de agregação *max*, o resultado será o mostrado pela área hachurada da Figura 5.7. O último passo consiste na determinação da extensão do tempo de verde. Pelo método do centro de área da figura obtida, resulta-se uma extensão de 2.48 segundos, conforme ilustra a Figura 5.7.

5.4.2 Mecanismo Baseado em Casos

O Solucionador Local recorre ao Mecanismo Baseado em Casos quando não dispuser de uma solução eficaz para a situação de tráfego presente, isto e', quando as condições do tráfego forem diferentes das esperadas, como por exemplo com filas enormes ou tempos de espera ínfimos.

O Mecanismo Baseado em Casos faz uso de uma base de casos estruturada em forma de árvore, mostrada parcialmente pela Figura 5.8. O propósito desse tipo de estrutura é diminuir o espaço de casos a serem testados para similaridade, classificá-los, e permitir uma gerência mais fácil e eficiente da base de casos.

A estrutura dos casos é mostrada na Figura 5.9. Sua parte de atributos descreve:

- Fila local: 4 (grande);
- Fila lateral: 2 (pequena);
- Fila do próximo cruzamento: 3 (média);
- Fila do cruzamento anterior: 5 (enorme);
- Fila da próxima lateral: 2 (pequena);

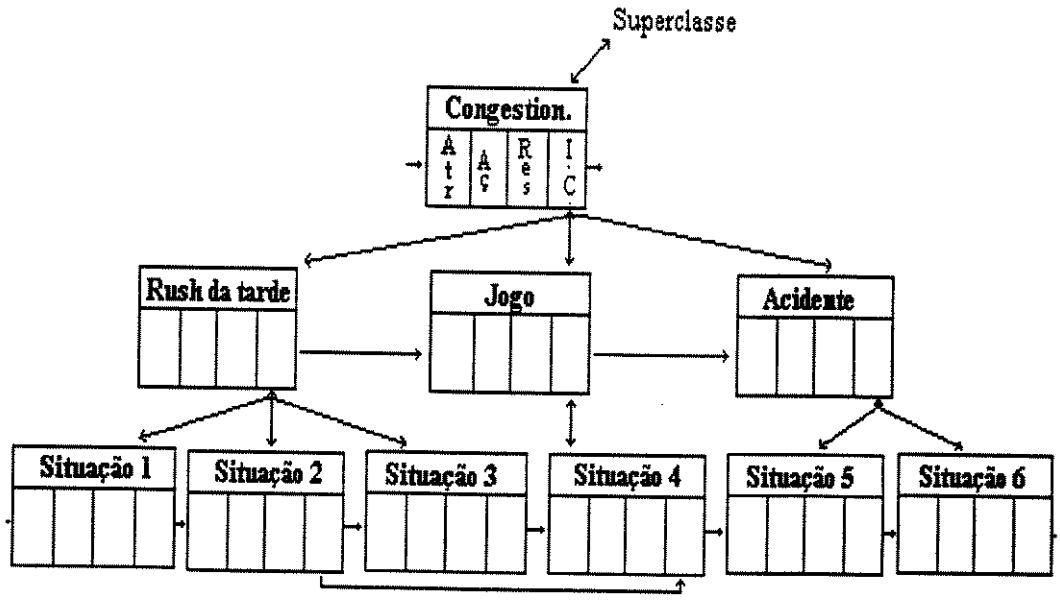


Figura 5.8: Estrutura da base de casos

- Fila da lateral anterior: 1 (ínfima);
- Tempo de espera local: 5 (enorme);
- Tempo de espera do próximo cruzamento: 4 (grande);
- Tempo de espera do cruzamento anterior: 4 (grande);
- Tempo de espera da lateral: 3 (médio);
- Tempo de espera da próxima lateral: 2 (pequeno).

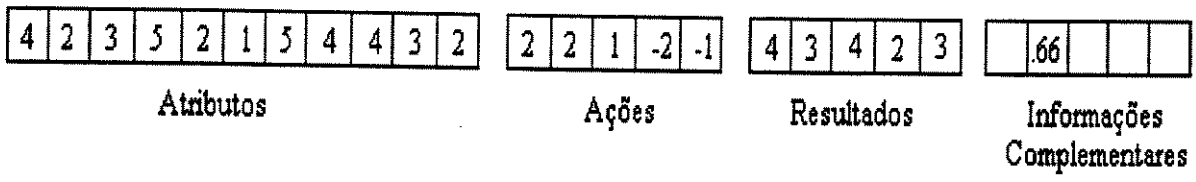


Figura 5.9: Exemplo de caso

A parte de ações especifica:

- Extensão de verde local: 2 (médio incremento);
- Extensão de verde do próximo cruzamento: 2 (médio incremento);
- Extensão de verde do cruzamento anterior: 1 (pequeno incremento);
- Extensão de verde da próxima lateral: -2 (médio decremento);
- Extensão de verde da lateral anterior: -1 (pequeno decremento).

A parte de resultados descreve:

- Fila local: 4 (bom);
- Fila do próximo cruzamento: 3 (médio);
- Fila do cruzamento anterior: 4 (bom);
- Fila lateral: 2 (ruim);
- Fila da próxima lateral: 3 (médio).

O grau de sucesso para o caso, considerando pesos de 1.5, 1.0, 1.0, 1.0 e 0.5 para os campos da parte de resultados, e escopo 5 para a variável lingüística *Resultado* é 0.66. Os cálculos para se chegar a esse resultado serão mostrados a seguir.

Exemplo de Operação

A seguir será apresentado um exemplo de operação do Mecanismo Baseado em Casos, envolvendo todas as suas etapas: a identificação de casos similares, a seleção dos casos, a combinação das ações dos casos selecionados, gerando uma decisão de controle, a avaliação do desempenho da decisão e o armazenamento do caso.

1) Identificação de casos similares.

A partir da especificação dos atributos de uma nova situação, recebida do Solucionador Local, o Mecanismo Baseado em Casos irá identificar os casos mais similares e de melhor desempenho, selecionado dois deles para o processo de combinação.

Atributos da nova situação, descrevendo tamanhos das filas e tempos de espera, conforme a Figura 5.9:

5	2	3	4	2	2	5	5	3	2	2

Casos Armazenados:

Caso 1:

4	2	3	5	2	1	5	4	4	3	2	3	3	1	-2	-1	4	3	4	2	3	.66				

Caso 2:

3	3	4	4	2	3	3	3	4	1	2	2	2	2	0	-2	4	3	3	2	2	.60				

Caso 3:

2	4	3	2	3	4	3	2	3	4	4	1	-1	-1	1	1	2	1	2	3	4	.44				

Aplicando a equação do grau de proximidade (ver página 64) entre os atributos da nova situação e dos casos armazenados, obtemos[6]:

Para o Caso 1:

grau_de_proximidade =

$$\frac{4*(3/4)+1*(1)+2*(1)+2*(3/4)+1*(1)+1*(3/4)+4*(1)+2*(3/4)+2*(3/4)+1*(3/4)+1*(1)}{11} \approx \boxed{1.636}$$

[6]Considerando pesos 4, 1, 2, 2, 1, 1, 4, 2, 2, 1 e 1 para os campos da parte de atributos.

Para o Caso 2:

grau_de_proximidade =

$$\frac{4*(2/4)+1*(3/4)+2*(3/4)+2*(1)+1*(1)+1*(3/4)+4*(2/4)+2*(2/4)+2*(3/4)+1*(3/4)+1*(1)}{11} \approx$$

$\boxed{1.295}$

Para o Caso 3:

grau_de_proximidade =

$$\frac{4*(1/4)+1*(2/4)+2*(1)+2*(2/4)+1*(3/4)+1*(2/4)+4*(2/4)+2*(1/4)+2*(1)+1*(2/4)+1*(2/4)}{11} \approx$$

$\boxed{1.023}$

2) Seleção dos casos.

Nesta implementação em particular, a seleção dos casos a serem combinados foi resultado de uma média ponderada entre o grau de proximidade (peso 1) e o grau de sucesso (peso 2):

Caso 1: $\frac{1.636+2*0.66}{3} \approx \boxed{0.985}$

Caso 2: $\frac{1.295+2*0.60}{3} \approx \boxed{0.832}$

Caso 3: $\frac{1.023+2*0.44}{3} \approx \boxed{0.634}$

Nesse exemplo, seriam selecionados os casos 1 e 2 para combinação.

3) Combinação das ações dos casos selecionados.

Através da escolha aleatória das ações dos casos 1 e 2, geraríamos, por exemplo uma decisão como:

Ações do caso 1:

3	3	1	-2	-1

Ações do caso 2:

2	2	2	0	-2

Ações geradas:

3	2	1	0	-1

Assim, para uma situação com atributos descrevendo:

- Fila local: 5 (enorme);
- Fila lateral: 2 (pequena);
- Fila do próximo cruzamento: 3 (média);
- Fila do cruzamento anterior: 4 (grande);
- Fila da próxima lateral: 2 (pequena);
- Fila da lateral anterior: 2 (pequena);
- Tempo de espera local: 5 (enorme);
- Tempo de espera do próximo cruzamento: 5 (enorme);
- Tempo de espera do cruzamento anterior: 3 (médio);
- Tempo de espera da lateral: 2 (pequeno);
- Tempo de espera da próxima lateral: 2 (pequeno).

o Mecanismo Baseado em Casos teria gerado uma decisão de controle especificando:

- Extensão de verde local: 3 (grande incremento);
- Extensão de verde do próximo cruzamento: 2 (médio incremento);
- Extensão de verde do cruzamento anterior: 1 (pequeno incremento);

- Extensão de verde da próxima lateral: 0 (nenhum incremento);
- Extensão de verde da lateral anterior: -1 (pequeno decremento).

Essa decisão seria transmitida para o Solucionador Local para que este pudesse executar as ações localmente[7] e transmitir as decisões para os semáforos vizinhos. O Mecanismo Baseado em Casos aguarda o resultado dessas ações para que possa alimentar sua base de casos e permitir a reutilização das decisões tomadas.

4) Avaliação do desempenho e armazenamento do caso.

Após um certo período de tempo (um ciclo), o Solucionador Local envia a situação das filas nos diversos cruzamentos, como por exemplo:

- Fila local: 3 (média);
- Fila do próximo cruzamento: 2 (pequena);
- Fila do cruzamento anterior: 3 (média);
- Fila lateral: 3 (média);
- Fila da próxima lateral: 2 (pequena).

Comparando com a situação anterior à aplicação da decisão:

- Fila local: 5 (enorme);
- Fila do próximo cruzamento: 3 (média);
- Fila do cruzamento anterior: 4 (grande);

[7]Localmente seria computada uma média ponderada entre essa decisão gerada e as decisões recebidas dos nós vizinhos, com peso maior para a decisão local.

- Fila lateral: 2 (pequena);
- Fila da próxima lateral: 2 (pequena).

podem ser obtidos resultados sobre cada uma das filas:

- Fila local: 5 (ótimo);
- Fila do próximo cruzamento: 4 (bom);
- Fila do cruzamento anterior: 2 (ruim);
- Fila lateral: 2 (ruim);
- Fila da próxima lateral: 3 (médio).

No protótipo implementado foram utilizadas variáveis lingüísticas com muitos termos lingüísticos (13 termos para a variável lingüística *Resultados*), para maior variabilidade genética dos casos. A conversão dos valores das filas para obtenção dos resultados acima deu-se em função das diferenças entre seus valores. Assim, ao passar a fila local de *enorme* (5) para *média* (3), atribuímos ilustrativamente um valor *ótimo* (5) para o resultado.

Considerando pesos de 1.5, 1.0, 1.0, 1.0 e 0.5 para os campos da parte de resultados, e escopo 5 para a variável lingüística *Resultado* (excelente, ótimo, bom, médio, ruim e péssimo), podemos calcular um grau geral de desempenho, baseados na equação do grau de proximidade:

$$grau\ de\ desempenho = \frac{\sum_{i=1}^{n} w_i \frac{desemp_i}{escopo}}{n} \tag{5.1}$$

onde n é o número de atributos, w_i é o peso do atributo i, $desemp_i$ é o desempenho em relação ao atributo i, e $escopo$ é o escopo da variável de desempenho.

$$grau\ de\ desempenho = \frac{1.5*(5/5)+1.0*(4/5)+1.0*(2/5)+1.0*(2/5)+0.5*(3/5)}{5} = \boxed{0.68}$$

Uma vez que o grau de desempenho do novo caso é superior ao dos casos armazenados, o novo caso será também armazenado na base de casos. Quando sua subárvore tiver atingido seu tamanho máximo, serão retirados os casos de pior desempenho e o caso mais antigo, independentemente do desempenho. Esse mecanismo mantém a base de casos em dimensões limitadas, preservando os tempos de acesso e de busca. Não permite também que casos isolados de alto desempenho possam reproduzir-se demasiadamente, reduzindo a variabilidade genética da base de casos e assim as possibilidades de adaptação do sistema a situações novas.

5.5 Resultados

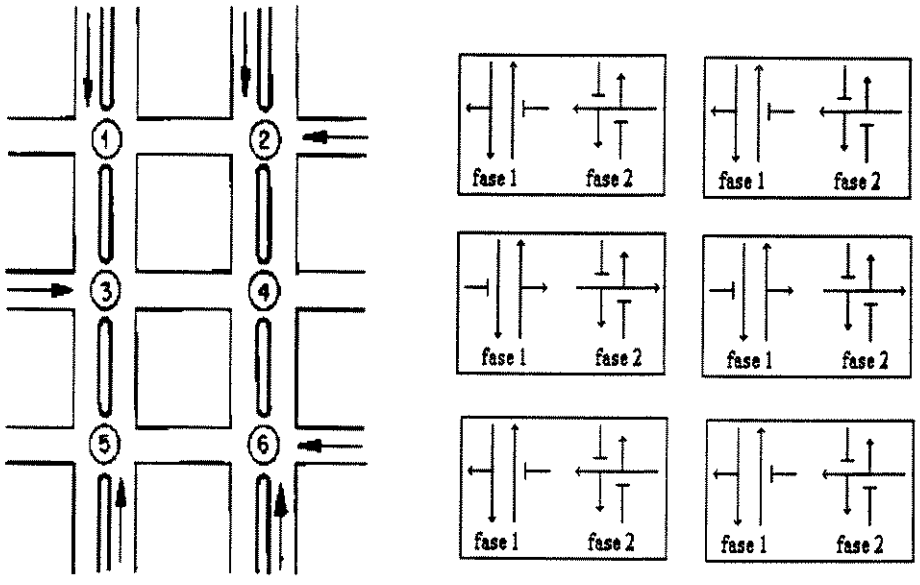


Figura 5.10: Rede de cruzamentos

A seguir, são mostrados os resultados obtidos pelo protótipo do sistema. A rede de cruzamentos utilizada, baseada em cruzamentos reais da cidade de Campinas - SP, compreende seis cruzamentos com três semáforos por cruzamento, conforme ilustra a Figura 5.10. O Apêndice B apresenta os parâmetros e os cálculos efetuados a partir

Fase	Direção de Movimento	Fase de Vermelho[s]	Fase de verde		Ciclo Total[s]	Fluxo [veic/h]	Número de Vias
			Verde[s]	Amarelo[s]			
1	F.Glicério	55	32	3	90	1000	2
2	O. Maia	35	52	3	90	1500	3

Tabela 5.2: Tabela de valores utilizados no cruzamento

Fase	Direção de Movimento	Fase de Vermelho[s]	Fase de verde		Ciclo Total[s]	Fluxo [veic/h]	Número de Vias
			Verde[s]	Amar.[s]			
1	F.Glicério	29	17	4	50	1000	2
2	O. Maia	21	24	5	50	1500	3
1	F.Glicério	53	33	4	90	1000	2
2	O. Maia	37	48	5	90	1500	3

Tabela 5.3: Tabela de valores calculados

de medições de tráfego realizadas em cruzamentos da Avenida Orozimbo Maia, para a obtenção dos tempos de verde para os cruzamentos, os quais balizaram as simulações. As taxas de chegada de veículos que balizaram os vários testes variaram a partir de 1500 veículos por hora nas avenidas e 1000 veículos por hora nas ruas transversais. Esses foram os valores obtidos no cruzamento mais crítico medido, o da Avenida Orozimbo Maia com a Rua Francisco Glicério, em horário de pico. As tabelas 5.2 e 5.3 mostram os principais valores utilizados.

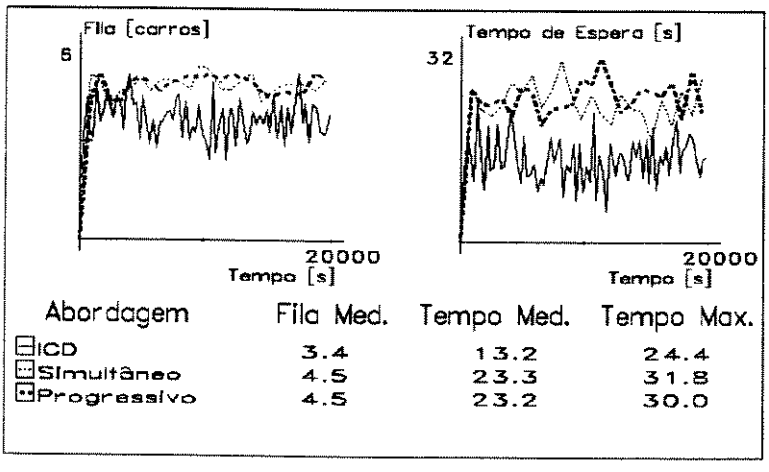
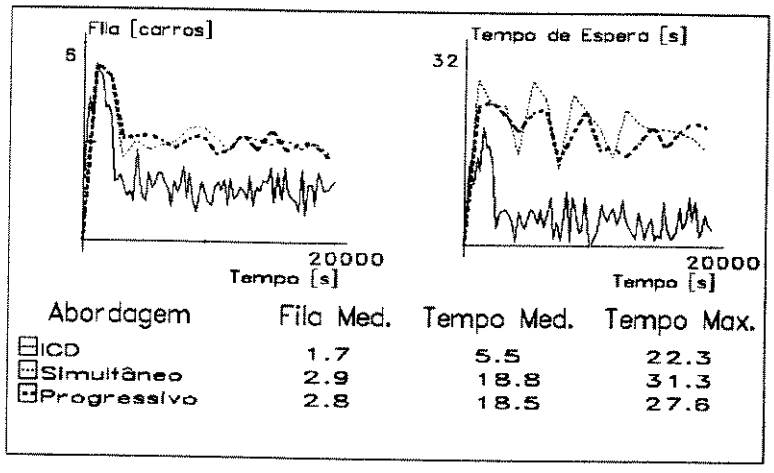
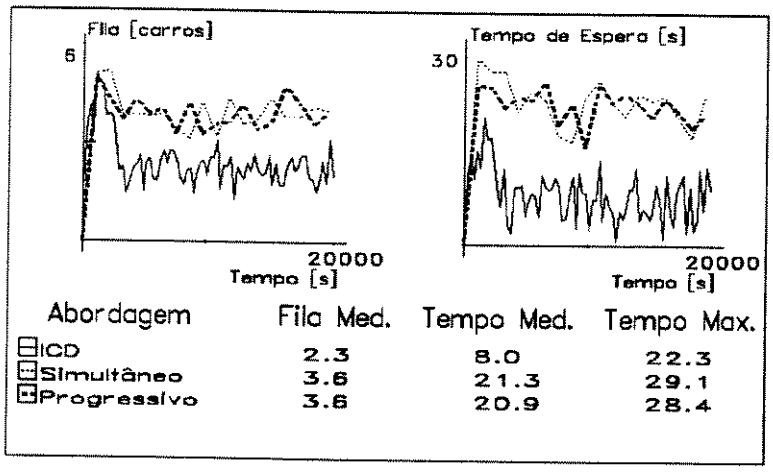


Figura 5.11: Gráfico para o fluxo-base



Abordagem	Fila Med.	Tempo Med.	Tempo Max.
ICD	1.7	5.5	22.3
Simultâneo	2.9	18.8	31.3
Progressivo	2.8	18.5	27.6

Figura 5.12: Gráfico para 60% do fluxo de entrada



Abordagem	Fila Med.	Tempo Med.	Tempo Max.
ICD	2.3	8.0	22.3
Simultâneo	3.6	21.3	29.1
Progressivo	3.6	20.9	28.4

Figura 5.13: Gráfico para 80% do fluxo de entrada

A Figura 5.11 mostra os valores médios de fila e tempo de espera para os dezoito semáforos da rede, utilizando as taxas básicas de chegada de veículos (taxas medidas nos cruzamentos). Essa figura apresenta os valores obtidos nas simulações de semáforos de ciclo-fixo simultâneo, ciclo-fixo progressivo (seqüencial) e ICD. Para as abordagens convencionais, foram utilizados os valores de ciclo observados nos cruzamentos, que são resultado de cálculos similares e ajuste de engenheiros de tráfego nos cruzamentos. Foi simulado o comportamento das abordagens por 20.000 segundos, exceto quando as taxas de chegada de veículos foram aumentados em 50%, quando não houve memória suficiente no microcomputador utilizado.

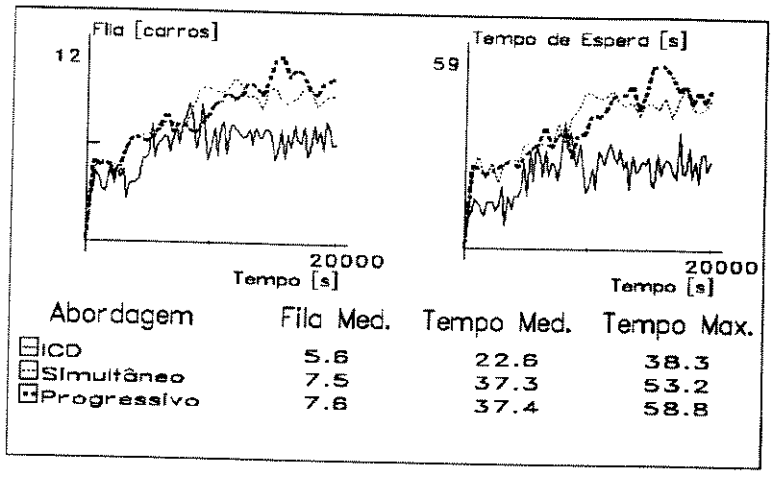


Figura 5.14: Gráfico para um aumento de 20% no fluxo de entrada

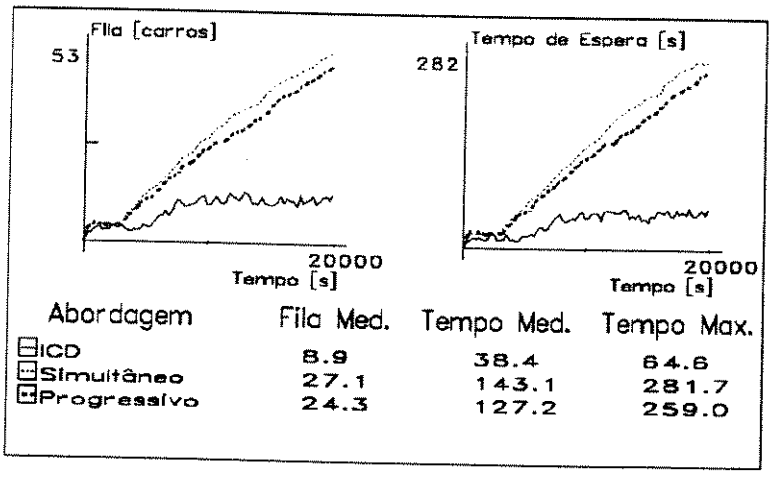


Figura 5.15: Gráfico para um aumento de 30% no fluxo de entrada

As Figuras 5.12 a 5.16 mostram os resultados obtidos variando-se os fluxos de entrada, a partir do instante 3000s. Elas mostram o comportamento das abordagens quando os fluxos de entrada deixam de ser aqueles esperados (isto é, quando diferem dos valores previamente fixados). A Figura 5.17 mostra a média das filas e atraso dos veículos em uma simulação de acidente em uma das vias, também ocorrido no instante 3000s, e a Figura 5.18 mostra os dados do cruzamento onde o acidente ocorreu. A Figura 5.19 mostra os resultados de simulação quando foram utilizados os tempos de ciclo calculados segundo [108], sem os ajustes de engenheiros de tráfego. Finalmente, as Figuras 5.21 a 5.23 mostram simulações onde foram utilizadas duas estratégias para os semáforos convencionais: uma correspondendo a 100% do fluxo-base e outra cor-

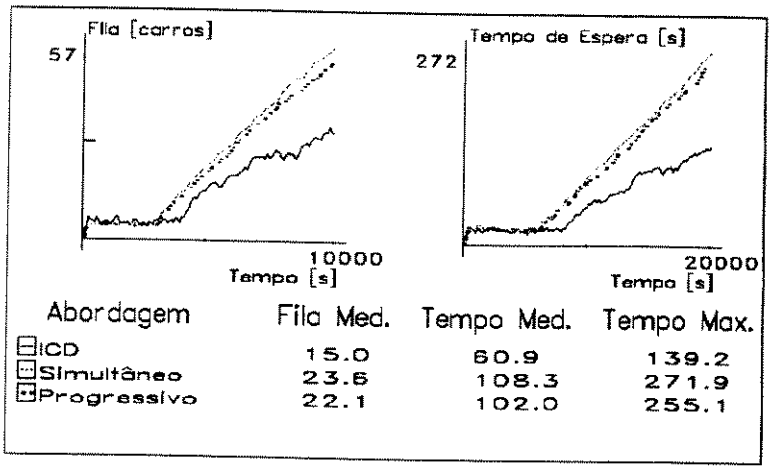


Figura 5.16: Gráfico para um aumento de 50% no fluxo de entrada

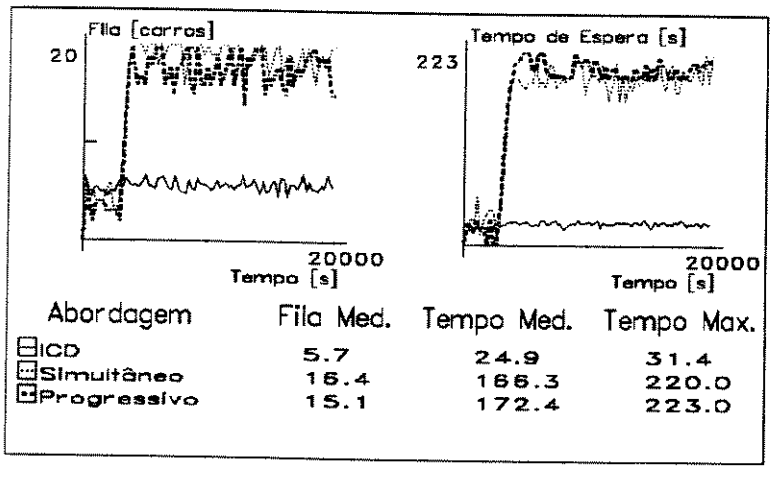


Figura 5.17: Gráfico cruzamento com acidente de trânsito (média)

respondendo a 120% do fluxo-base, ajustados conforme [108]. Os gráfico do fluxo de tráfego para essas duas simulações são mostrados nas Figuras 5.20 e 5.22, respectivamente. Em todos os casos, utilizou-se distribuição gaussiana, com desvio padrão igual a 20% da média. A relação completa dos gráficos de todos os cruzamentos, para todas as variações de fluxo e acidente encontram-se no Apêndice A.

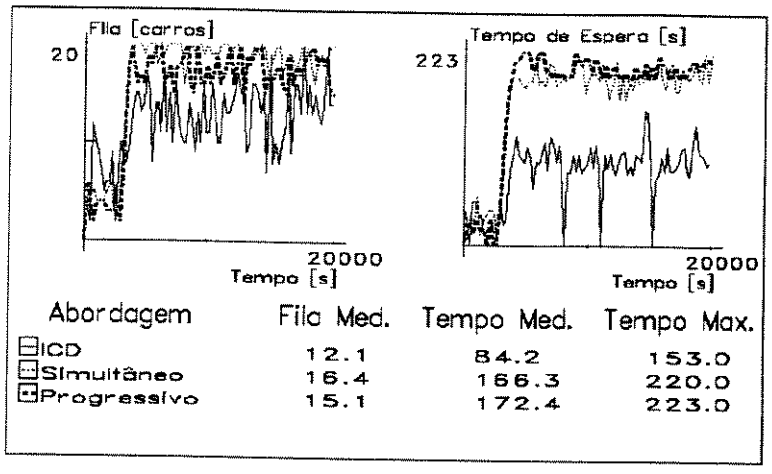


Figura 5.18: Gráfico cruzamento com acidente de trânsito

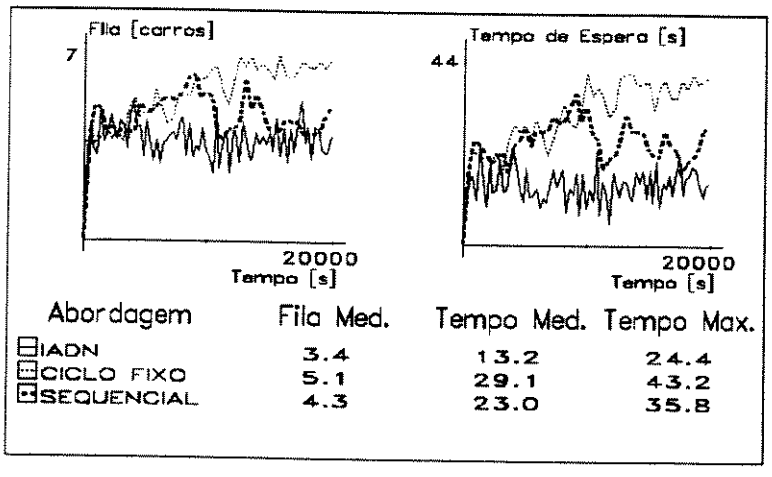


Figura 5.19: Gráfico para o fluxo-base com ciclo calculado

5.6 Análise dos Resultados

No primeiro exemplo, mostrado pela Figura 5.11, tomou-se como fluxos de entrada os valores medidos nos cruzamentos. Tomou-se também os ciclos obtidos matematicamente para as abordagens de ciclo-fixo simultâneo e ciclo-fixo progressivo. Esse exemplo compara o desempenho das abordagens para condições ideais de tráfego, e mostra uma vantagem da abordagem ICD sobre as abordagens convencionais. Embora os semáforos convencionais tenham sido ajustados para aquele padrão de fluxo, as equações que calculam esses tempos não levam em conta as interações entre os diversos

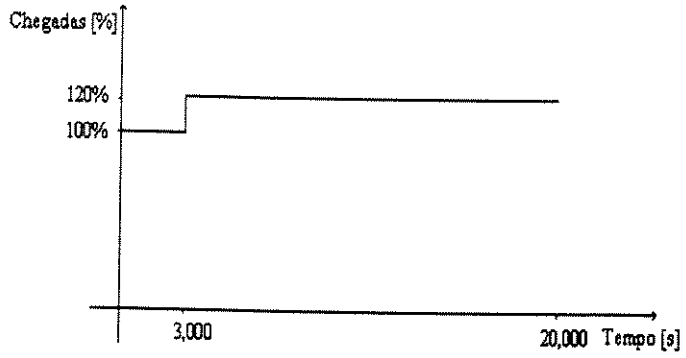


Figura 5.20: Gráfico de fluxo para aumento em degrau

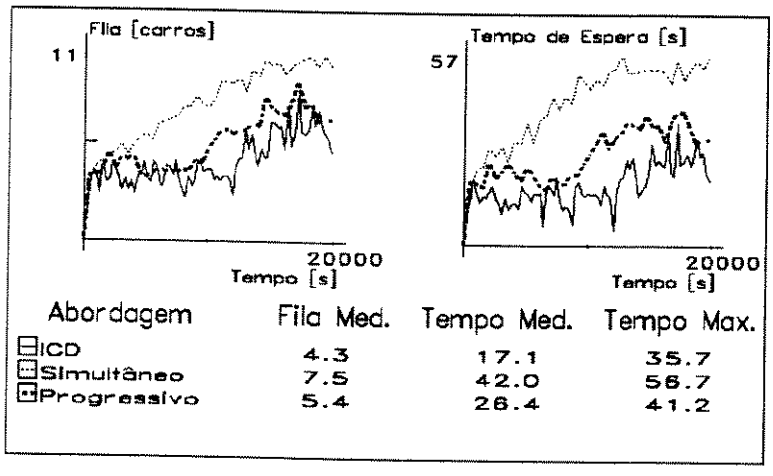


Figura 5.21: Gráfico para aumento de fluxo em degrau

cruzamentos de uma malha urbana. A abordagem ICD apresentou filas cerca de 24% menores (3.4 veículos por ciclo contra 4.5 veículos, em média), e tempos de espera cerca de 43% menores (vantagem de cerca de 10 segundos).

Quando as taxas de chegada diferem das taxas de chegada preestabelecidas, a vantagem da abordagem ICD acentua-se.

As Figuras 5.12 e 5.13 mostram os desempenhos quando as taxas de chegada passam, no instante 3000s, a 60 e 80% das taxas esperadas, respectivamente. No primeiro caso, as filas são da ordem de 40% menores para a abordagem ICD (1.7 contra 2.8 veículos), e os tempos médios de espera foram menores que um terço das convencionais (5.5s contra 18.5s). No segundo caso, as filas são da ordem de 36% menores e os tempos

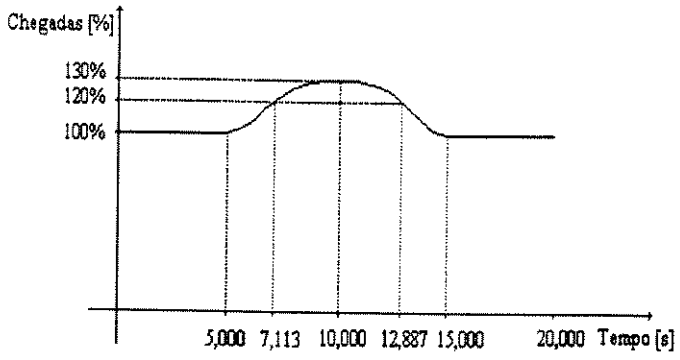


Figura 5.22: Gráfico de fluxo para pico de tráfego

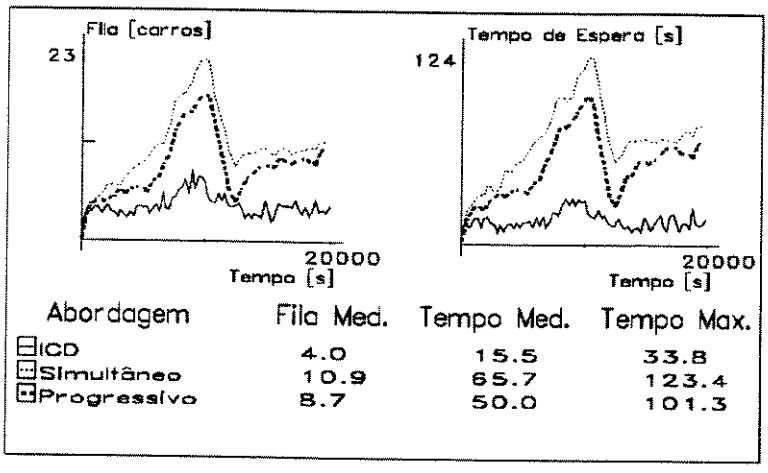


Figura 5.23: Gráfico para pico de tráfego

de espera da ordem de 62% menores.

Nas situações críticas, quando a taxa de chegada de veículos é maior que o esperado ou quando ocorre uma situação crítica imprevista, o desempenho da abordagem ICD também mostrou-se bastante superior. Na Figura 5.14, observamos que após ter sido aumentado o fluxo de veículos (no instante 3000), o desempenho das abordagens convencionais piora consideravelmente (da ordem de 4.5 para 7.5 carros, e de 23 para 37 segundos de tempo médio de espera). O desempenho da abordagem ICD passa de 3.4 para 5.6 veículos, e de 13.2 para 22.6 segundos de espera para cada motorista no trânsito. Assim, durante todo o período de 20.000s, a abordagem ICD foi cerca de 25% melhor quanto aos tamanhos das filas, e cerca de 39% melhor quanto aos tempos de

espera. Ao aumentar mais a taxa de chegada de carros, a diferença de desempenho acentua-se. A Figura 5.15 mostra um aumento de 30% a partir do instante 3000s. As filas e o tempo de espera dos veículos aumentam incessantemente nas abordagens convencionais, enquanto que a abordagem ICD consegue adaptar-se à nova situação. Após 20.000s, as filas permanecem em torno de 8.9 veículos e o tempo de espera em 38.4s, enquanto que nas abordagens convencionais já chegam a atingir pelo menos 24.3 carros e 127.2s de tempo de espera, com tendência a continuar aumentando sempre mais. Aumentando-se a taxa de chegada de veículos em 50%, mostrado na Figura 5.16, o desempenho de todas as abordagens piora muito. No entanto, a abordagem ICD adapta-se melhor ao grande fluxo de veículos, obtendo médias de 15.0 veículos e 60.9s contra pelo menos 22.1 veículos e 102s nas abordagens convencionais.

Na simulação de acidente de trânsito em um dos cruzamentos, ilustrada pelas Figuras 5.17 e 5.18, a abordagem ICD mostra também sua capacidade de adaptação. A Figura 5.17 mostra as médias obtidas pelas três abordagens. A abordagem ICD apresentou filas cerca de 62% menores e tempos de espera cerca de 85% menores. Considerando o desempenho apenas do cruzamento onde ocorre o acidente, ilustrado pela Figura 5.18, observamos filas cerca de 20% menores (5.7 veículos contra 15.1 veículos) e tempos médios de espera cerca de 50% menores (82.4 contra 166.3s).

Utilizando os ciclos de controle calculados, observamos também vantagem da abordagem ICD. Na Figura 5.19, observamos uma vantagem de aproximadamente 29% em relação às filas (3.7 contra 5.2 veículos) e de aproximadamente 47% em relação aos tempos de espera (15.9 contra 29.8s). Na simulação seguinte, as taxas de chegada foram aumentadas de 20% após 10.000s, conforme ilustra a Figura 5.21. Nesse mesmo instante foram alteradas as estratégias de controle das abordagens convencionais para a nova situação. A abordagem ICD apresentou filas médias pelo menos 20% menores (4.3 contra 5.4 veículos) e tempos médios de espera pelo menos 35% menores (17.1 contra 26.4s), ilustrado pela Figura 5.21. Finalmente, alteramos as taxas de chegada conforme ilustra a Figura 5.23. Entre os instantes 7.113 e 12.887s as abordagens convencionais foram ajustadas para 120% do fluxo básico. A Figura 5.23 ilustra os resultados de

simulação obtidos. O desempenho da abordagem ICD foi aproximadamente 54% melhor em relação às filas (4.0 contra 8.7 veículos) e 69% melhor em relação aos tempos médios de espera (15.5 contra 50.0s).

Dessa forma, o protótipo implementado obteve sempre melhores resultados que as abordagens de ciclo-fixo simultâneo e ciclo-fixo progressivo, tendo sido capaz de adaptar-se a diferentes situações, obtendo resultados bastante bons. Os resultados obtidos, bastante expressivos, mostram o grande potencial da abordagem para a resolução de problemas, e em particular, para o controle de tráfego em malha urbana.

5.7 Resumo

Nesse capítulo foi evidenciada a aplicação da arquitetura ICD em resolução de problemas, particularmente o de controle de tráfego urbano. Explicitou-se a especificação do problema e do conhecimento, e as tarefas relativas aos nós do sistema, para que este atinja seus objetivos. Foram mostrados detalhes da implementação do protótipo e apresentados os resultados obtidos por ele, comparando-os com os de abordagens convencionais de controle de tráfego. Mostrou-se que uma abordagem adaptativa dessa natureza pode ser de grande utilidade para o controle de sistemas de tráfego urbano reais, sujeitos a várias perturbações que os fazem atuar sob condições não ideais.

Capítulo 6

Conclusão

Neste trabalho, foi apresentada uma nova abordagem para o desenvolvimento de sistemas inteligentes distribuídos, denominada Inteligência Computacional Distribuída. Essa nova abordagem une características de várias técnicas para prover flexibilidade, adaptabilidade, aproveitamento do esforço computacional anterior e aprendizagem. A abordagem foi apresentada não apenas descritivamente, mas também através de uma linguagem formal de especificação de sistemas distribuídos, baseada em lógica proposicional temporal. Por ter sido especificada em uma linguagem formal, a arquitetura proposta poderá ser mais facilmente compreendida, implementada e estendida do que um sistema apenas descrito em linguagem natural.

A abordagem proposta pôde ser testada e validada através de um sistema de controle de tráfego urbano. O sistema apresentou as características desejadas de manipulação de informações incertas e incompletas, adaptação a novas situações e aprendizagem. Além disso, obteve desempenho superior a abordagens convencionais de controle de tráfego, indicando a cooperação obtida pelos agentes.

6.1 Contribuições

Este trabalho decorre da simbiose entre conceitos e métodos de várias áreas da ciência da computação, derivando-se uma abordagem única e coesa, caracterizada pelas seguintes contribuições:

1. Agregação da teoria dos conjuntos nebulosos à inteligência artificial distribuída. As características de representação, tratamento de incertezas e mecanismos de inferência da teoria dos conjuntos nebulosos foram pela primeira vez utilizadas dentro do contexto da inteligência artificial distribuída. Essa integração, onde os diversos agentes de um sistema podem manipular informações incompletas, incertas ou inconsistentes, une a flexibilidade à cooperação entre agentes;

2. Mecanismo Baseado em Casos evolutivo. A visão de um caso passado como uma cadeia de genes, codificados através de variáveis lingüísticas, permitiu a união de várias características: a reutilização do esforço computacional anterior mais efetivo, adaptação do sistema, evitando máximos locais (algoritmos genéticos) e aprendizagem. Dessa forma, obteve-se um mecanismo eficiente, flexível e adaptativo;

3. Especificação formal da arquitetura genérica. A especificação da arquitetura através de uma linguagem formal a torna suscetível a novas implementações e extensões, que poderão reproduzir ou aumentar sua aplicabilidade a outros domínios, em aplicações potencialmente complexas, dinâmicas e distribuídas;

4. Sistema de controle de tráfego urbano. A arquitetura proposta foi utilizada em um sistema de controle de tráfego urbano composto por uma malha de dezoito semáforos, em seis cruzamentos, podendo ser facilmente estendida para outras configurações. A aplicação apresentada pode ser considerada bastante complexa (vários cruzamentos e interações entre eles) e abrangente (vários testes realizados). Nesta área, têm sido apresentados sistemas que controlam semáforos de um único cruzamento ou de dois cruzamentos consecutivos na mesma via. O sistema

obteve resultados significativamente superiores aos das abordagens convencionais, utilizando dados reais, em todos os casos testados.

6.2 Trabalhos Futuros

Por tratar-se de uma nova abordagem e por reunir várias áreas de pesquisa, inúmeros trabalhos podem ser identificados a partir deste, dentre os quais incluem-se:

1. Extensões à arquitetura, agregando mecanismos como de contratação de tarefas (como redes de contrato) e resolução de inconsistências (como negociação);

2. Tratamento explícito de falhas, que se referem não somente a falhas de processadores, mas também a violações de tempo, problemas de interação entre os nós, e falhas de informação.

3. Mecanismos para relacionar ações e resultados, no Mecanismo Baseado em Casos. O desenvolvimento de mecanismos capazes de entender ou associar como as ações produzem respostas (in)desejadas, poderá conduzir a uma melhor compreensão dos domínios de aplicação, e obter melhores mecanismos de raciocínio;

4. Estabelecimento de mecanismos de prova para a linguagem de especificação utilizada. Seria desejável desenvolver mecanismos para prova formal de sistemas distribuídos inteligentes, que incluíssem sua completeza e correção.

5. Implantação real do sistema de controle de tráfego. Dados os resultados de simulação do sistema, implantá-lo inicialmente em uma pequena malha urbana. Isso sem dúvida faria com que a abordagem e o sistema se desenvolvessem rapidamente, levantaria novas questões como a eventual influência de atrasos de transmissão, e traria potencialmente melhorias aos sistemas de controle de tráfego urbano utilizados atualmente.

Apêndice A

Gráficos de desempenho dos cruzamentos

A seguir, serão apresentados os gráficos de todos os semáforos, para cada um dos testes realizados. A identificação dos semáforos, introduzida para maior clareza, é constituída basicamente por uma seqüência de três números inteiros: o primeiro indicando a linha; o segundo indicando a coluna; e o terceiro indicando qual o semáforo do cruzamento, conforme ilustra a Figura A.1. Os semáforos dos cruzamentos [1] e [2] pertencem à linha *1*, os semáforos dos cruzamentos [3] e [4] pertencem à linha *2*, e os semáforos dos cruzamentos [5] e [6] pertencem à linha *3*. Quanto à coluna, pertencem à coluna *1* os semáforos dos cruzamentos [1], [3] e [5], e os semáforos dos cruzamentos [2], [4] e [6] pertencem à coluna *2*. Dentro de cada cruzamento, os semáforos das vias horizontais são denotados por *0*, os semáforos das vias verticais em sentido N-S são denotados por *1* e os semáforos em sentido S-N são denotados por *2*. Assim, o semáforo com fluxo de entrada de carros do cruzamento [6] é denotado por *322*, e o semáforo com fluxo de entrada de carros do cruzamento [4] é denotado por *220*.

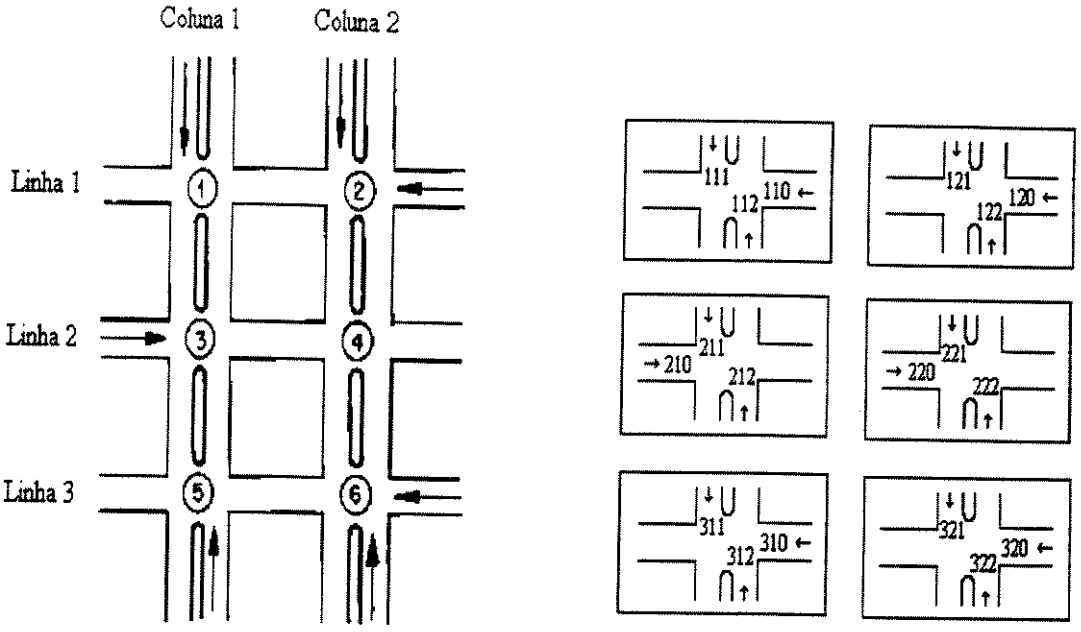
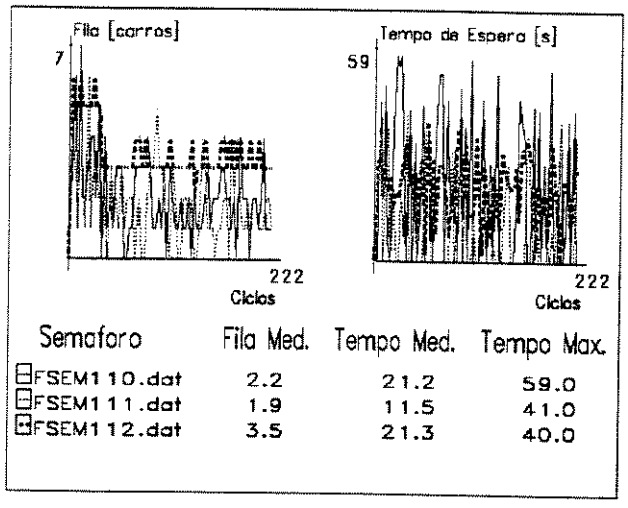


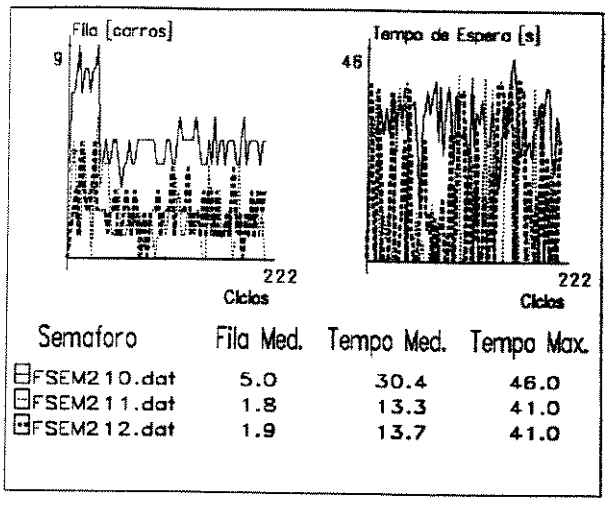
Figura A.1: Identificação dos semáforos

As figuras A.2 a A.31 mostram gráficos de desempenho de cada um dos semáforos, relativos às simulações apresentadas na Seção 5.5. As figuras A.2 a A.19 mostram os resultados obtidos nas simulações onde variamos as taxas de chegada de veículos a partir de 3000 segundos. As figuras A.20 a A.22 mostram os resultados para uma situação de acidente no instante 3000 segundos. As figuras A.23 a A.25 mostram os resultados de cada semáforo quando utilizamos os ciclos calculados segundo [108], sem levar em conta os ajustes realizados por engenheiros de tráfego. As figuras A.26 a A.31 mostram os resultados das simulações quando alteramos as taxas de chegada de veículos e as estratégias de controle para os semáforos, segundo [108].



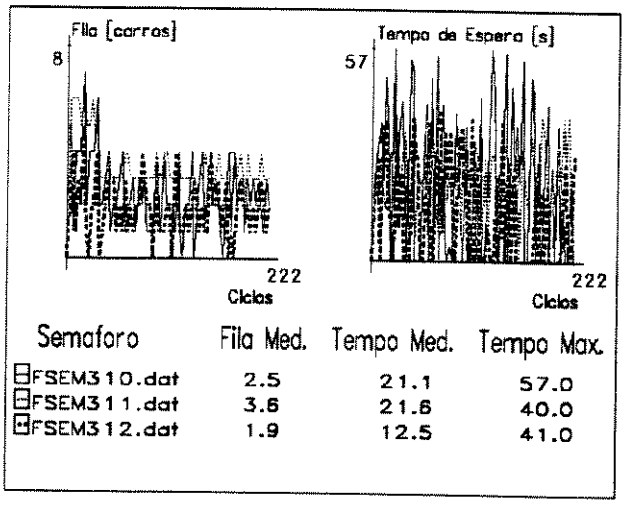
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	2.2	21.2	59.0
FSEM111.dat	1.9	11.5	41.0
FSEM112.dat	3.5	21.3	40.0

(a)



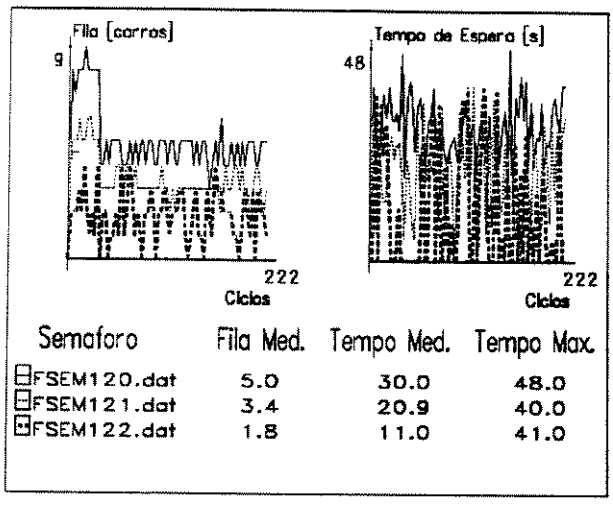
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	5.0	30.4	46.0
FSEM211.dat	1.8	13.3	41.0
FSEM212.dat	1.9	13.7	41.0

(b)



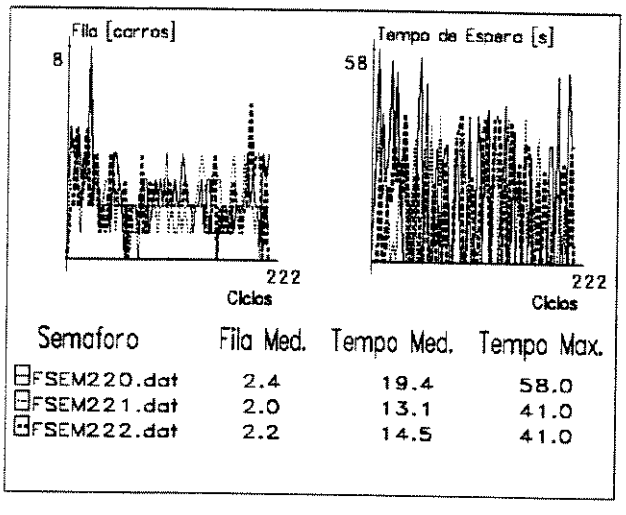
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	2.5	21.1	57.0
FSEM311.dat	3.6	21.6	40.0
FSEM312.dat	1.9	12.5	41.0

(c)



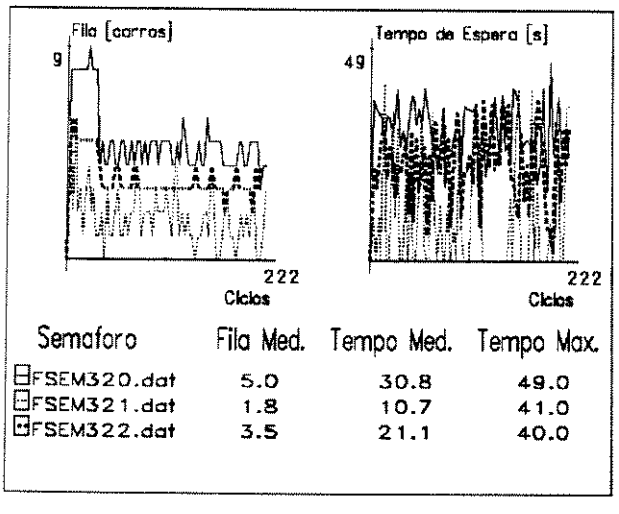
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	5.0	30.0	48.0
FSEM121.dat	3.4	20.9	40.0
FSEM122.dat	1.8	11.0	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	2.4	19.4	58.0
FSEM221.dat	2.0	13.1	41.0
FSEM222.dat	2.2	14.5	41.0

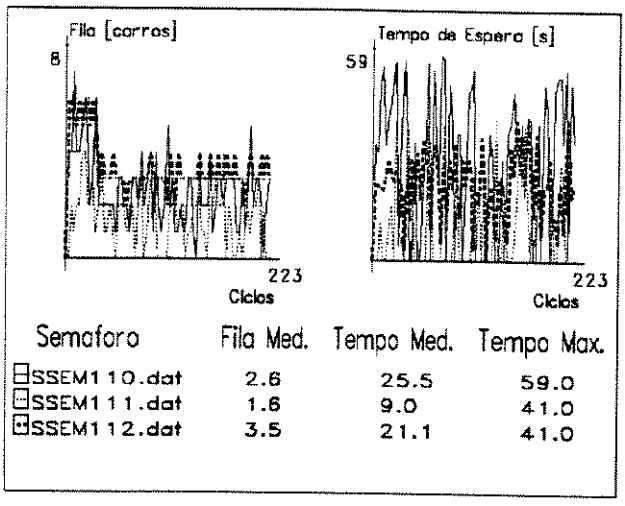
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	5.0	30.8	49.0
FSEM321.dat	1.8	10.7	41.0
FSEM322.dat	3.5	21.1	40.0

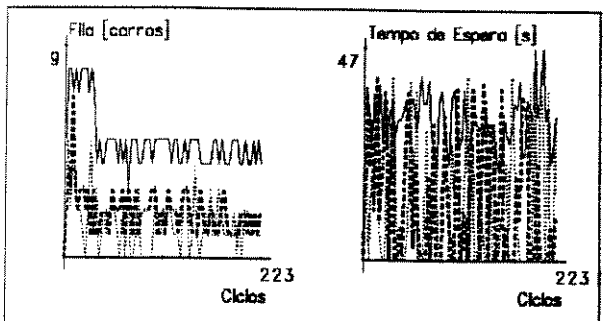
(f)

Figura A.2: Simultâneo: 60% do fluxo de entrada



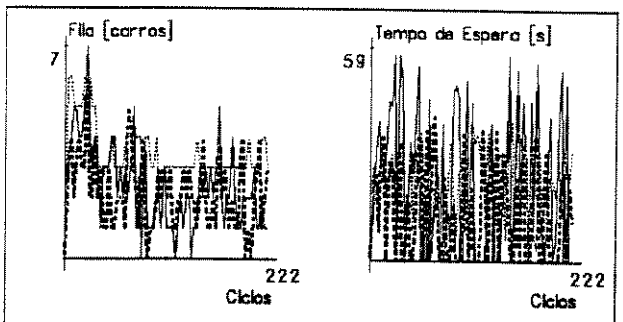
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	2.6	25.5	59.0
SSEM111.dat	1.6	9.0	41.0
SSEM112.dat	3.5	21.1	41.0

(a)



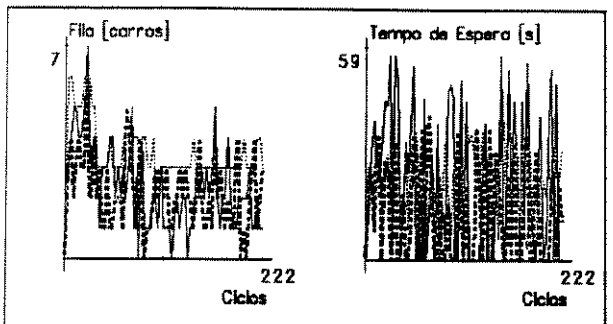
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	4.9	30.5	47.0
SSEM211.dat	1.5	10.3	41.0
SSEM212.dat	2.1	14.4	41.0

(b)



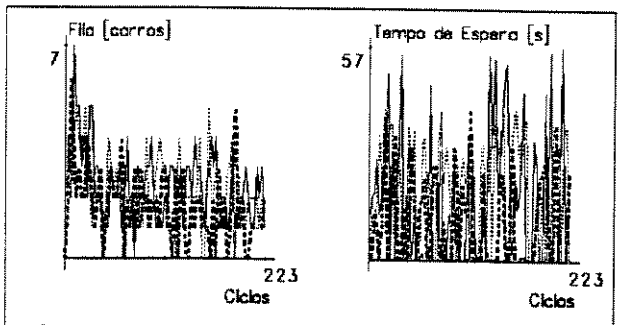
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	2.4	21.9	59.0
SSEM311.dat	3.5	21.9	41.0
SSEM312.dat	2.2	13.7	41.0

(c)



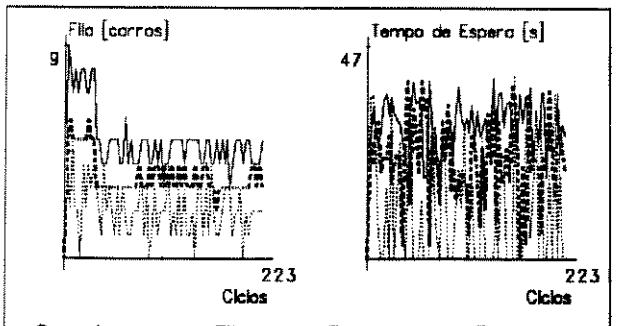
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	2.4	21.9	59.0
SSEM311.dat	3.5	21.9	41.0
SSEM312.dat	2.2	13.7	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	2.2	16.1	57.0
SSEM221.dat	2.2	14.8	41.0
SSEM222.dat	1.8	8.9	41.0

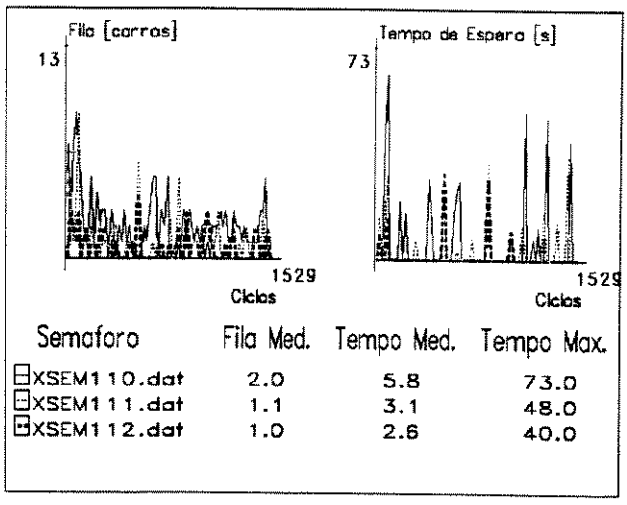
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	4.9	30.0	47.0
SSEM321.dat	2.0	12.3	41.0
SSEM322.dat	3.5	21.6	40.0

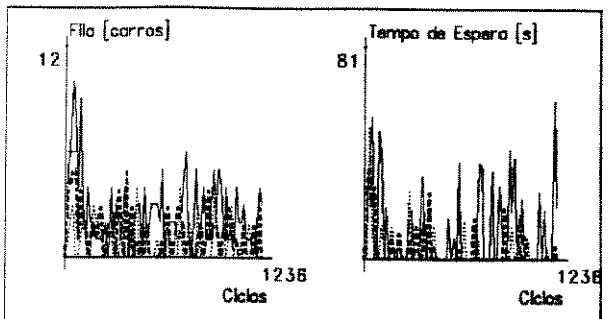
(f)

Figura A.3: Progressivo: 60% do fluxo de entrada



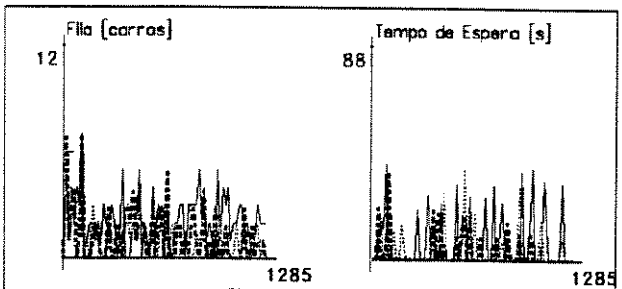
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	2.0	5.8	73.0
XSEM111.dat	1.1	3.1	48.0
XSEM112.dat	1.0	2.6	40.0

(a)



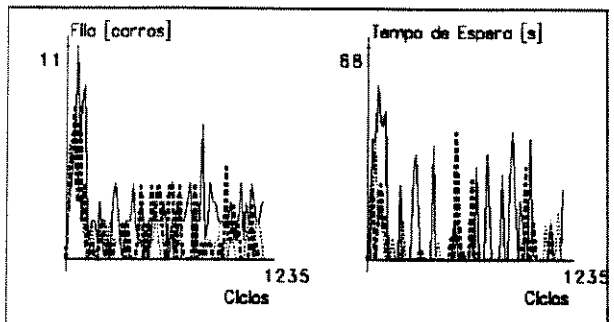
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	2.5	8.6	81.0
XSEM211.dat	1.1	3.5	55.0
XSEM212.dat	1.4	4.3	59.0

(b)



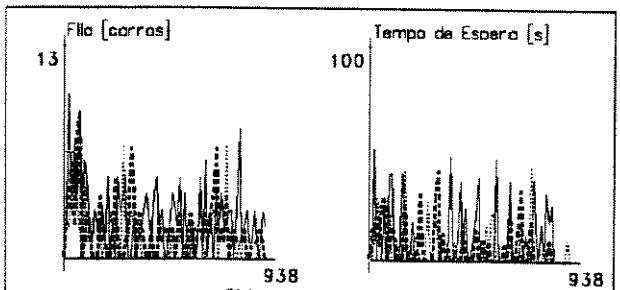
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	2.1	7.3	88.0
XSEM311.dat	1.0	2.9	40.0
XSEM312.dat	1.2	3.5	44.0

(c)



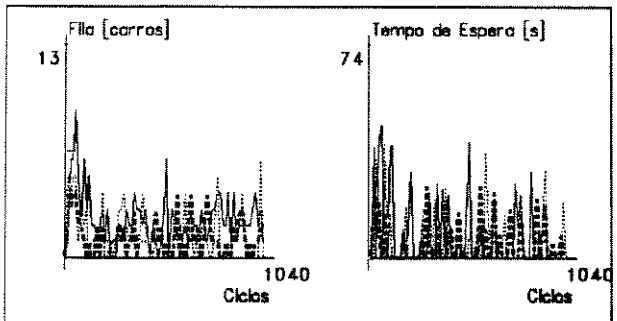
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	2.4	7.8	88.0
XSEM121.dat	1.2	3.5	42.0
XSEM122.dat	1.5	4.2	49.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	2.4	8.7	100.0
XSEM221.dat	1.2	4.1	45.0
XSEM222.dat	1.6	6.0	47.0

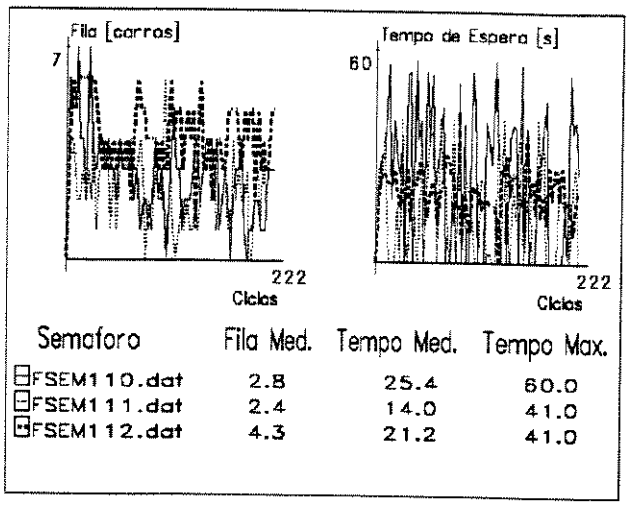
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	2.5	8.8	74.0
XSEM321.dat	1.5	5.1	53.5
XSEM322.dat	1.2	3.9	52.0

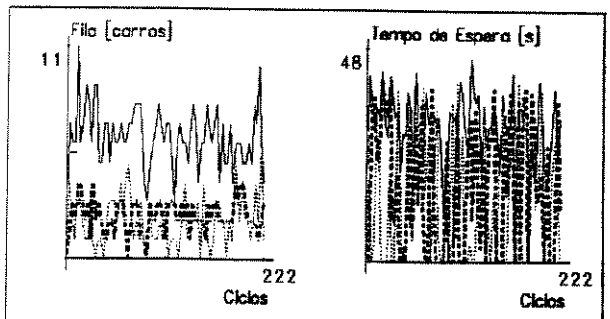
(f)

Figura A.4: ICD: 60% do fluxo de entrada



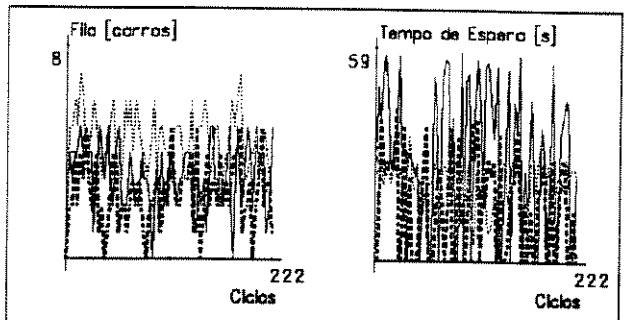
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	2.8	25.4	60.0
FSEM111.dat	2.4	14.0	41.0
FSEM112.dat	4.3	21.2	41.0

(a)



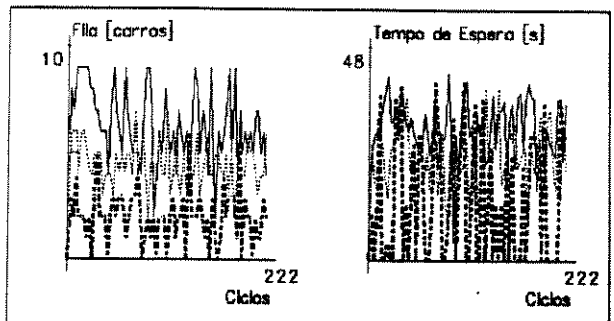
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	6.4	31.0	48.0
FSEM211.dat	2.1	15.0	41.0
FSEM212.dat	2.4	17.4	41.0

(b)



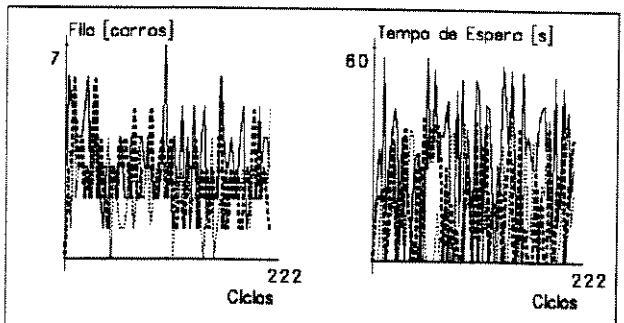
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	3.1	27.0	59.0
FSEM311.dat	4.5	22.2	41.0
FSEM312.dat	2.4	15.4	43.0

(c)



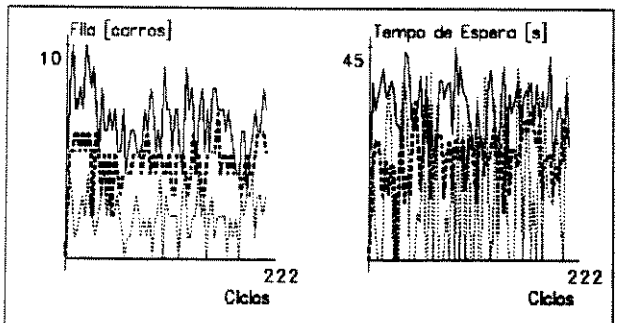
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	6.2	30.9	48.0
FSEM121.dat	4.4	22.5	39.0
FSEM122.dat	2.1	14.6	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	3.1	29.0	60.0
FSEM221.dat	2.4	15.3	41.0
FSEM222.dat	2.7	16.4	41.0

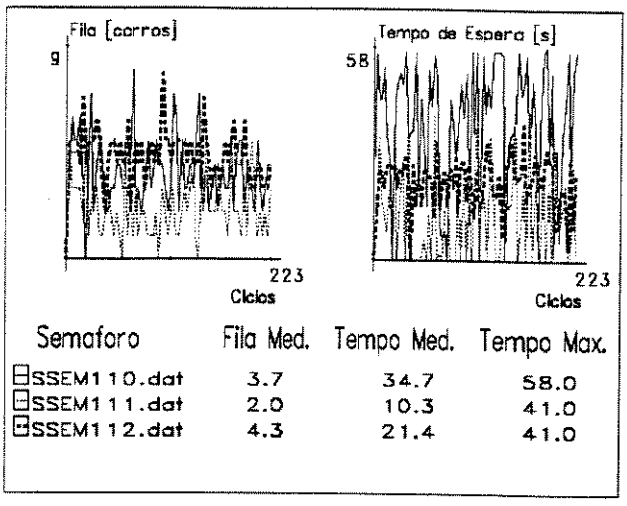
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	6.3	30.8	45.0
FSEM321.dat	2.0	13.1	41.0
FSEM322.dat	4.4	21.8	35.0

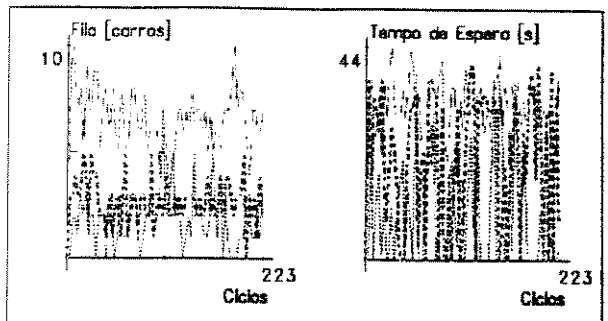
(f)

Figura A.5: Simultâneo: 80% do fluxo de entrada



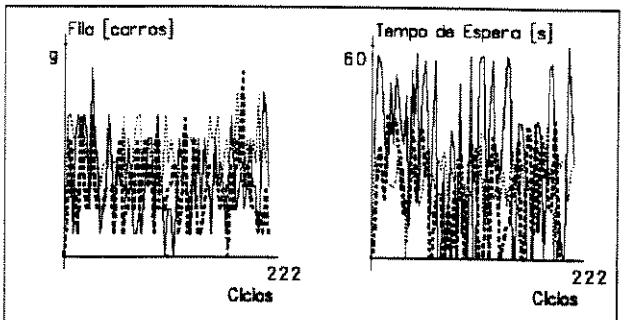
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	3.7	34.7	58.0
SSEM111.dat	2.0	10.3	41.0
SSEM112.dat	4.3	21.4	41.0

(a)



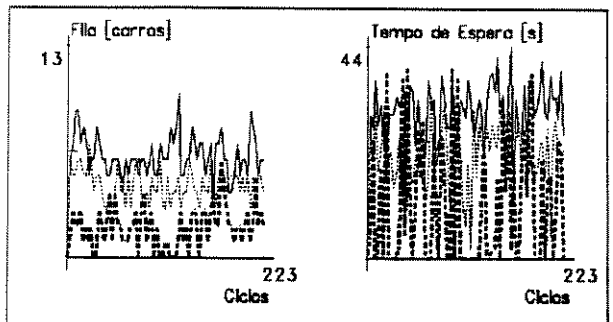
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	6.4	30.4	44.0
SSEM211.dat	1.9	10.0	41.0
SSEM212.dat	2.6	19.6	41.0

(b)



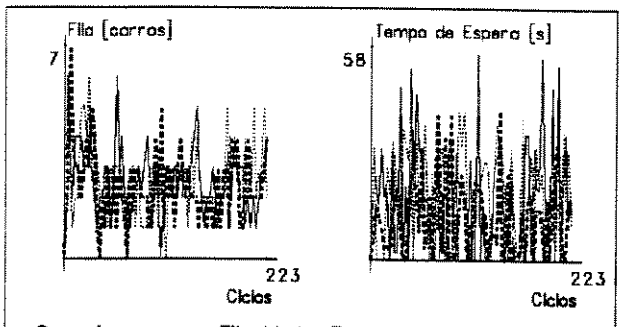
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	3.4	30.2	60.0
SSEM311.dat	4.5	22.2	38.0
SSEM312.dat	2.7	16.3	41.0

(c)



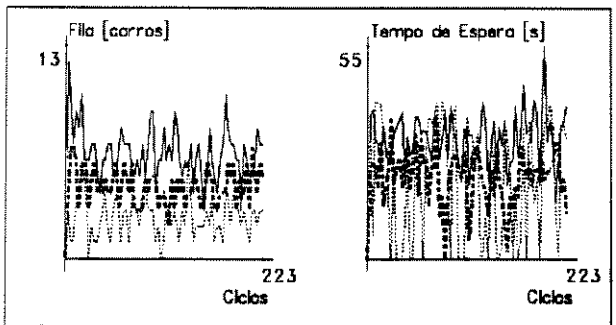
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	6.3	30.2	44.0
SSEM121.dat	4.4	21.7	40.0
SSEM122.dat	1.9	10.7	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	2.6	18.3	58.0
SSEM221.dat	2.7	18.7	41.0
SSEM222.dat	2.4	12.0	41.0

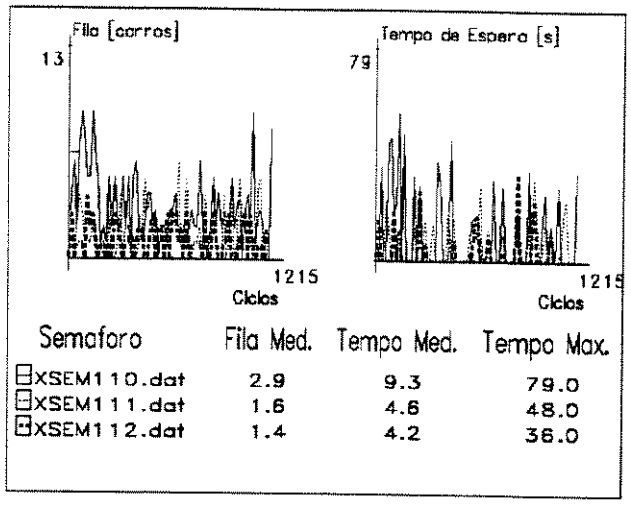
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	6.5	31.2	55.0
SSEM321.dat	2.5	17.5	41.0
SSEM322.dat	4.4	21.7	39.0

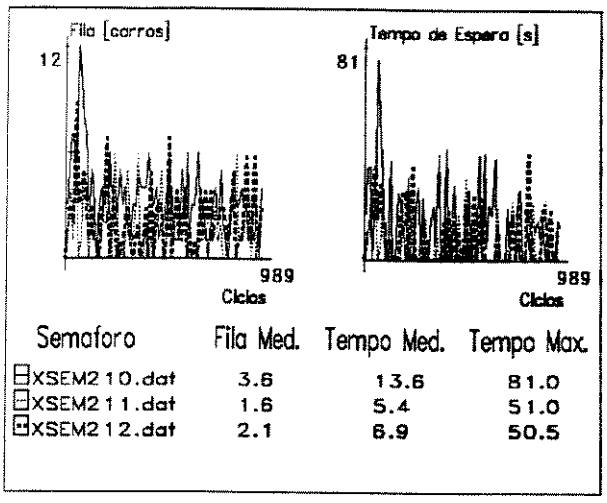
(f)

Figura A.6: Progressivo: 80% do fluxo de entrada



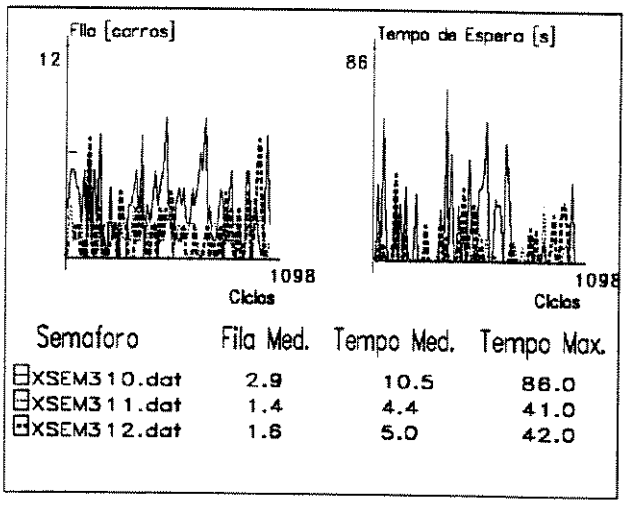
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	2.9	9.3	79.0
XSEM111.dat	1.6	4.6	48.0
XSEM112.dat	1.4	4.2	36.0

(a)



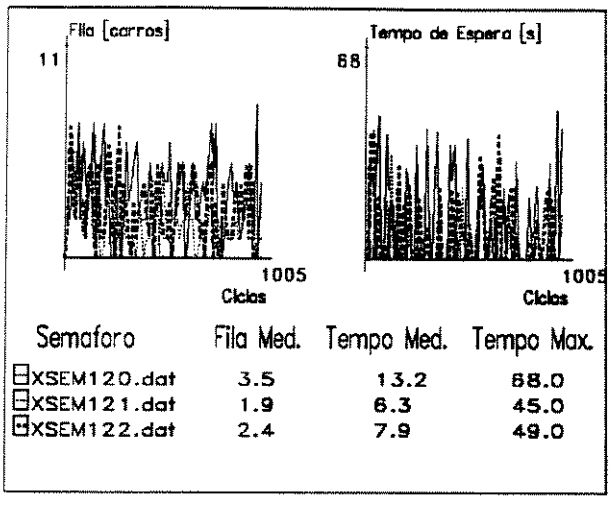
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	3.6	13.6	81.0
XSEM211.dat	1.6	5.4	51.0
XSEM212.dat	2.1	6.9	50.5

(b)



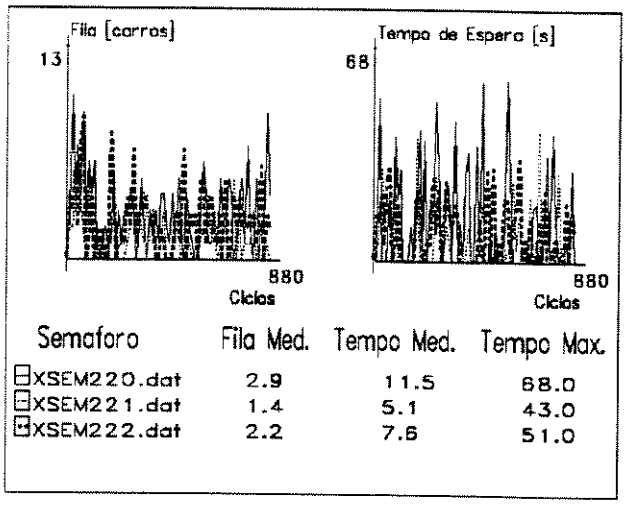
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	2.9	10.5	86.0
XSEM311.dat	1.4	4.4	41.0
XSEM312.dat	1.6	5.0	42.0

(c)



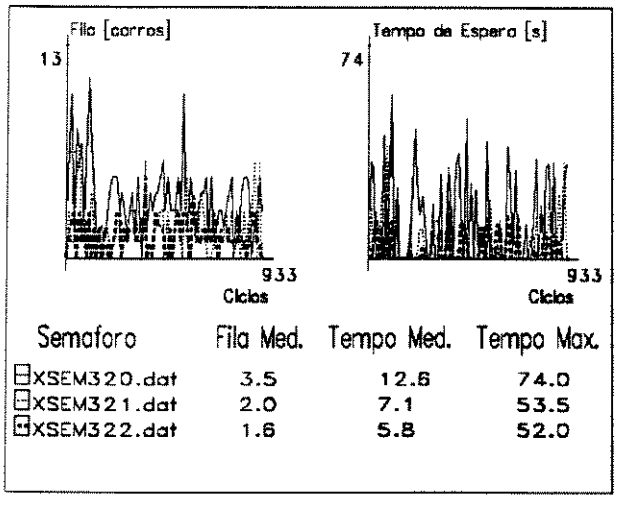
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	3.5	13.2	88.0
XSEM121.dat	1.9	6.3	45.0
XSEM122.dat	2.4	7.9	49.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	2.9	11.5	68.0
XSEM221.dat	1.4	5.1	43.0
XSEM222.dat	2.2	7.6	51.0

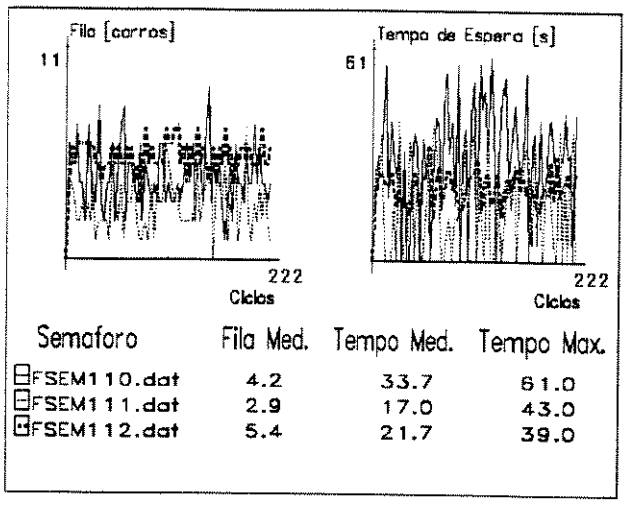
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	3.5	12.6	74.0
XSEM321.dat	2.0	7.1	53.5
XSEM322.dat	1.6	5.8	52.0

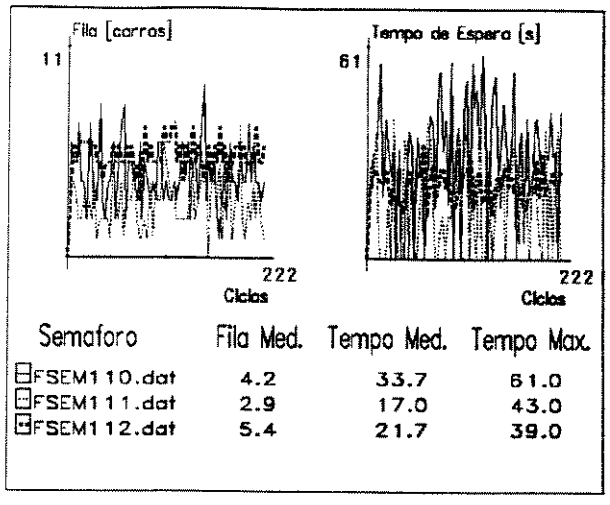
(f)

Figura A.7: ICD: 80% do fluxo de entrada



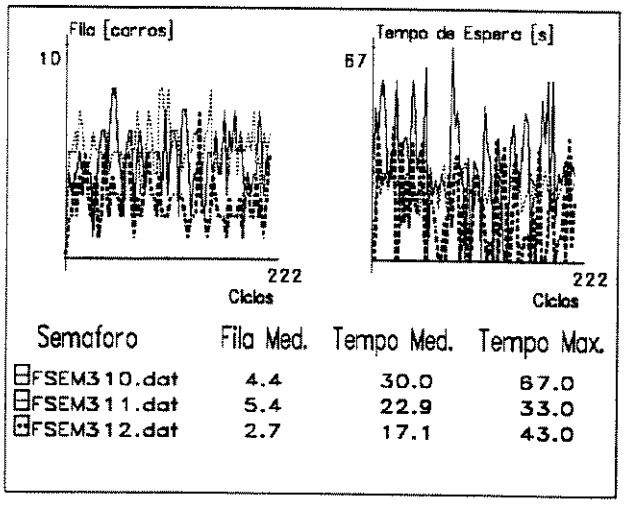
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	4.2	33.7	61.0
FSEM111.dat	2.9	17.0	43.0
FSEM112.dat	5.4	21.7	39.0

(a)



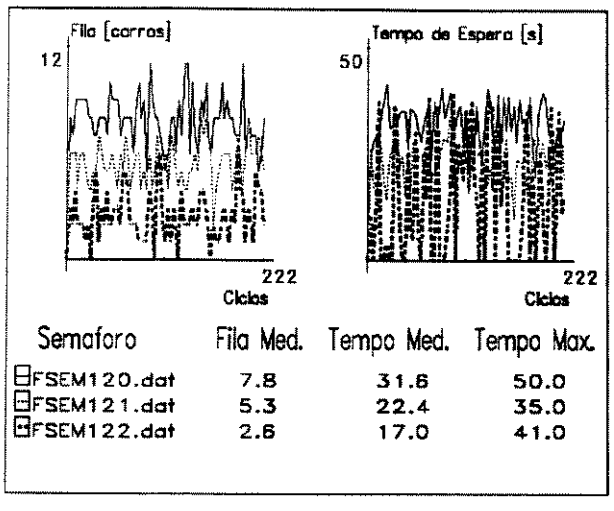
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	4.2	33.7	61.0
FSEM111.dat	2.9	17.0	43.0
FSEM112.dat	5.4	21.7	39.0

(b)



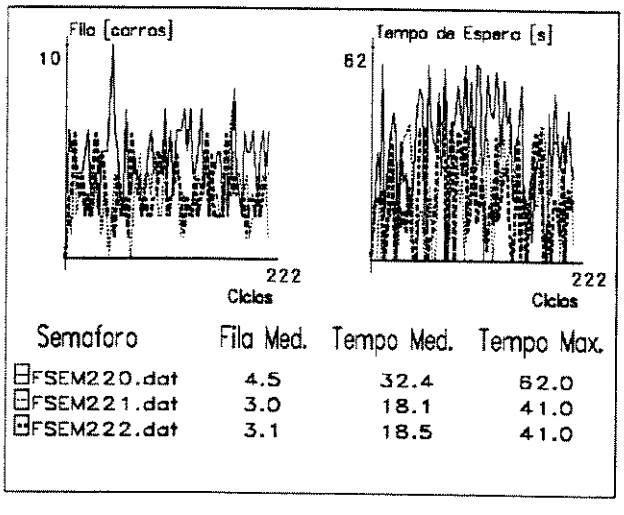
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	4.4	30.0	67.0
FSEM311.dat	5.4	22.9	33.0
FSEM312.dat	2.7	17.1	43.0

(c)



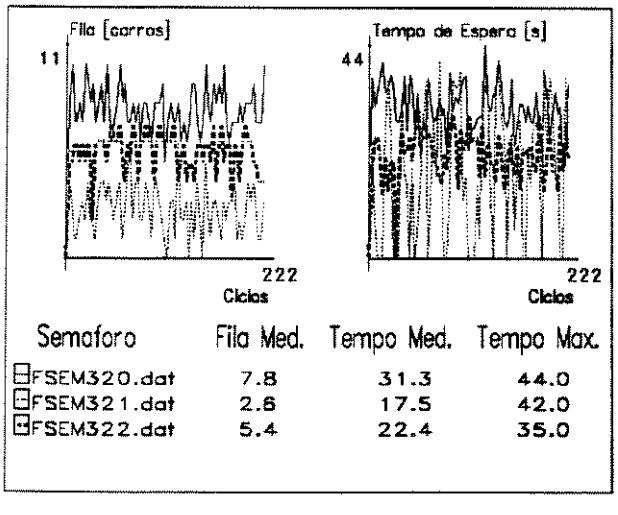
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	7.8	31.6	50.0
FSEM121.dat	5.3	22.4	35.0
FSEM122.dat	2.6	17.0	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	4.5	32.4	62.0
FSEM221.dat	3.0	18.1	41.0
FSEM222.dat	3.1	18.5	41.0

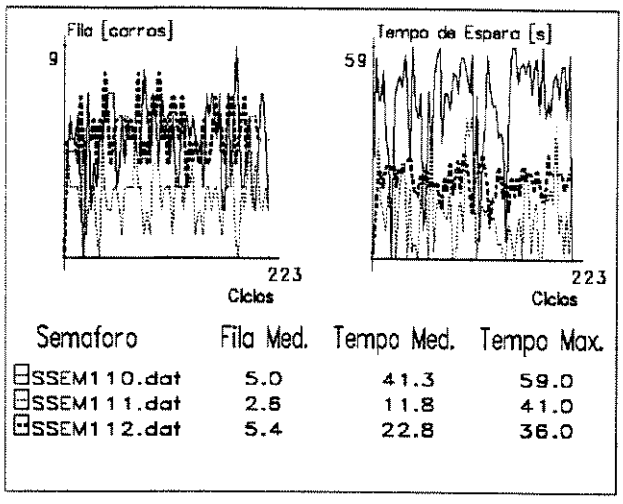
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	7.8	31.3	44.0
FSEM321.dat	2.6	17.5	42.0
FSEM322.dat	5.4	22.4	35.0

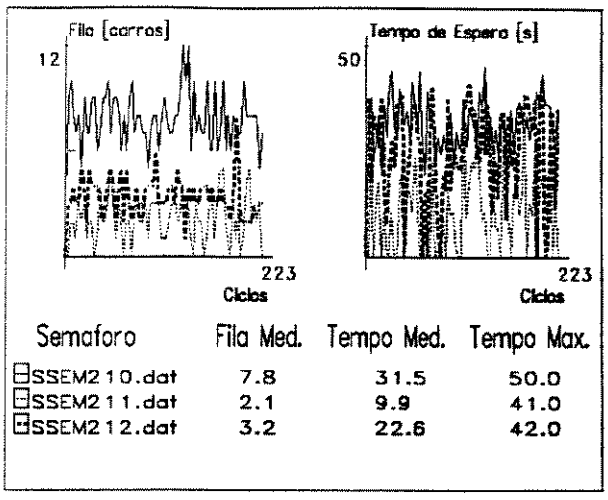
(f)

Figura A.8: Simultâneo: 100% do fluxo de entrada



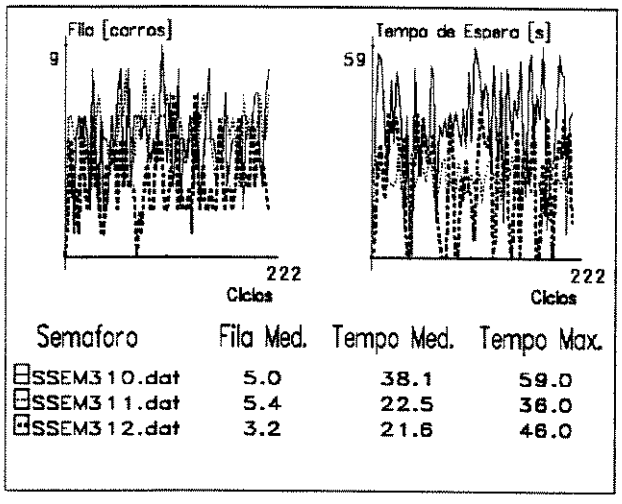
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	5.0	41.3	59.0
SSEM111.dat	2.6	11.8	41.0
SSEM112.dat	5.4	22.8	36.0

(a)



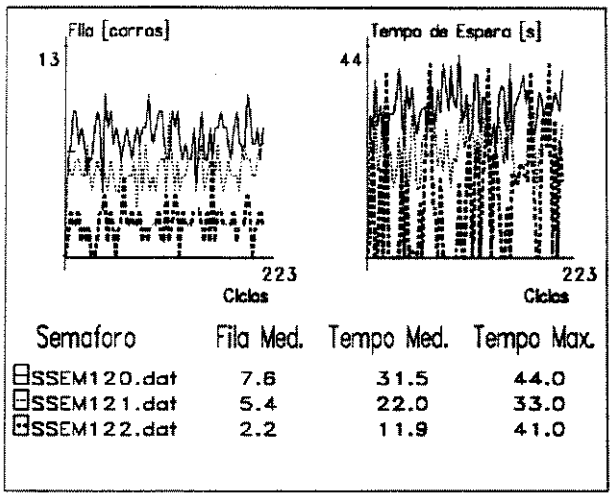
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	7.8	31.5	50.0
SSEM211.dat	2.1	9.9	41.0
SSEM212.dat	3.2	22.6	42.0

(b)



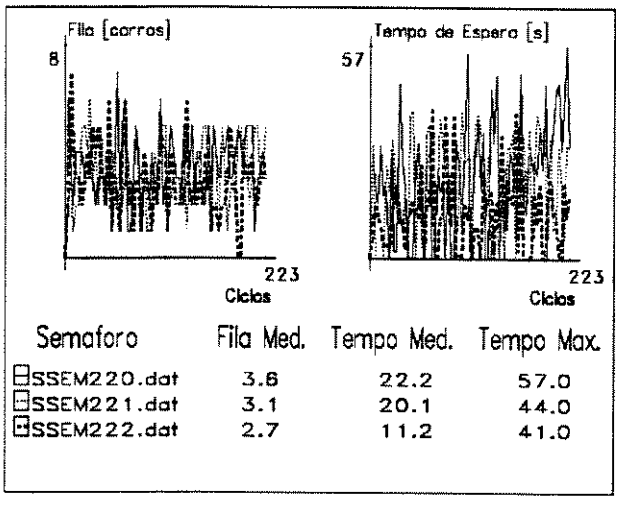
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	5.0	38.1	59.0
SSEM311.dat	5.4	22.5	36.0
SSEM312.dat	3.2	21.6	46.0

(c)



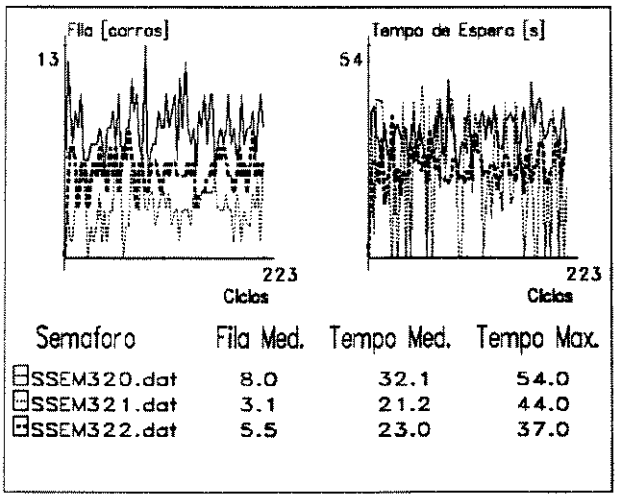
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	7.6	31.5	44.0
SSEM121.dat	5.4	22.0	33.0
SSEM122.dat	2.2	11.9	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	3.8	22.2	57.0
SSEM221.dat	3.1	20.1	44.0
SSEM222.dat	2.7	11.2	41.0

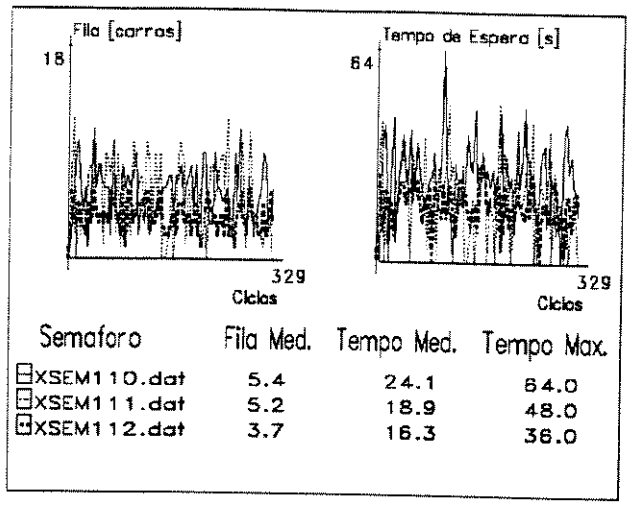
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	8.0	32.1	54.0
SSEM321.dat	3.1	21.2	44.0
SSEM322.dat	5.5	23.0	37.0

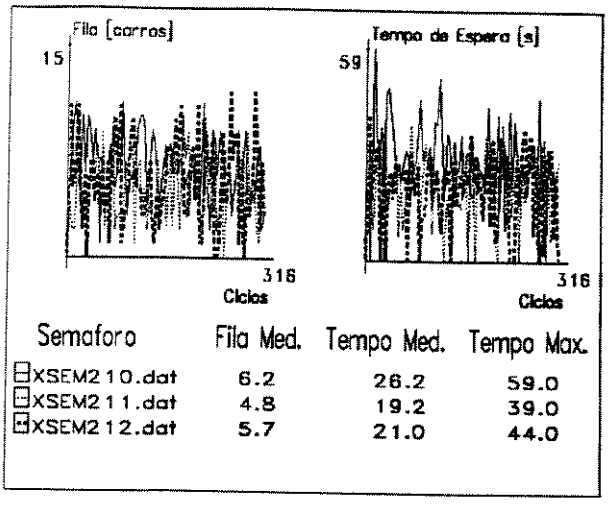
(f)

Figura A.9: Progressivo: 100% do fluxo de entrada



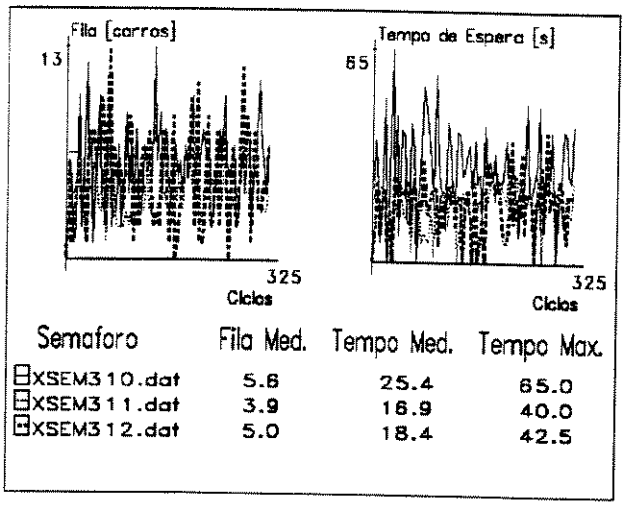
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	5.4	24.1	64.0
XSEM111.dat	5.2	18.9	48.0
XSEM112.dat	3.7	16.3	36.0

(a)



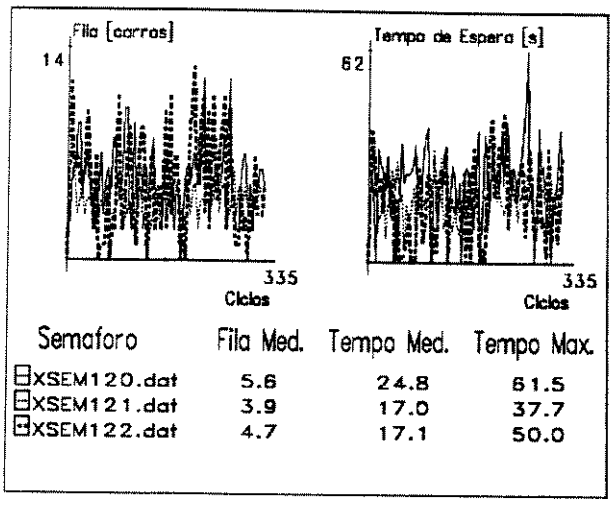
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	6.2	26.2	59.0
XSEM211.dat	4.8	19.2	39.0
XSEM212.dat	5.7	21.0	44.0

(b)



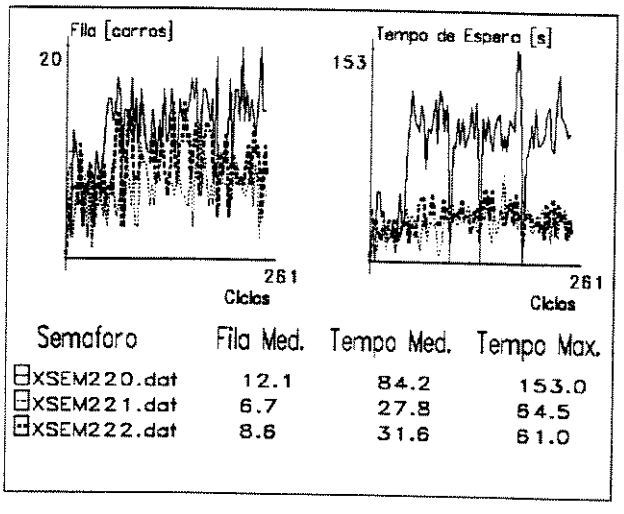
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	5.6	25.4	65.0
XSEM311.dat	3.9	16.9	40.0
XSEM312.dat	5.0	18.4	42.5

(c)



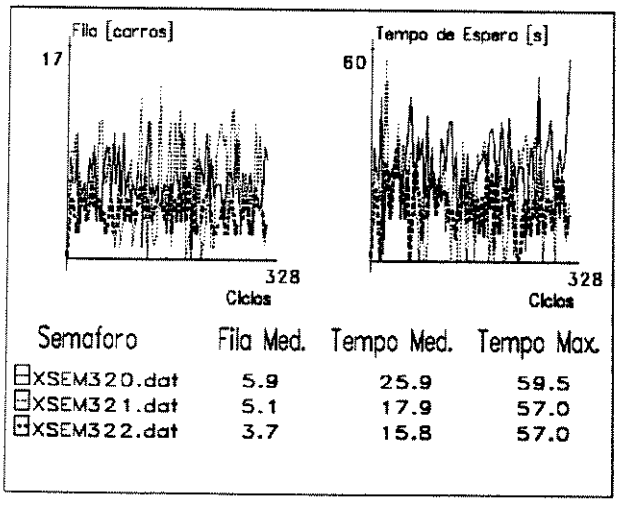
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	5.6	24.8	61.5
XSEM121.dat	3.9	17.0	37.7
XSEM122.dat	4.7	17.1	50.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	12.1	84.2	153.0
XSEM221.dat	6.7	27.8	64.5
XSEM222.dat	8.6	31.6	61.0

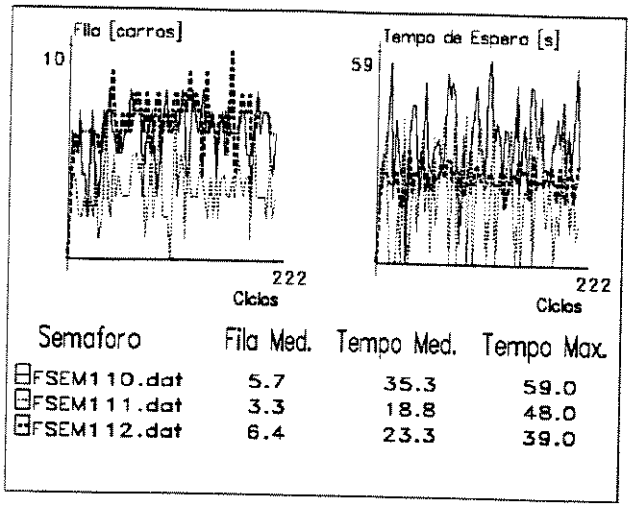
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	5.9	25.9	59.5
XSEM321.dat	5.1	17.9	57.0
XSEM322.dat	3.7	15.8	57.0

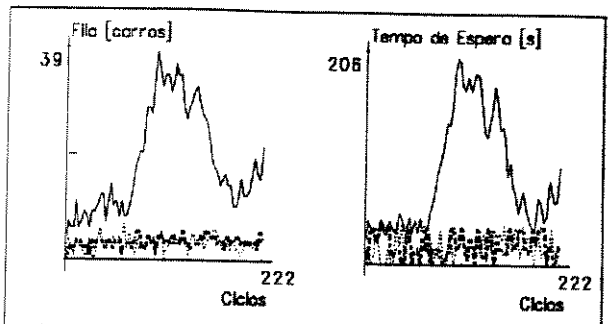
(f)

Figura A.10: ICD: 100% do fluxo de entrada



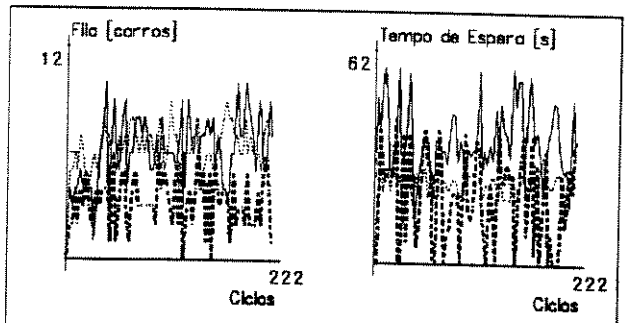
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	5.7	35.3	59.0
FSEM111.dat	3.3	18.8	48.0
FSEM112.dat	6.4	23.3	39.0

(a)



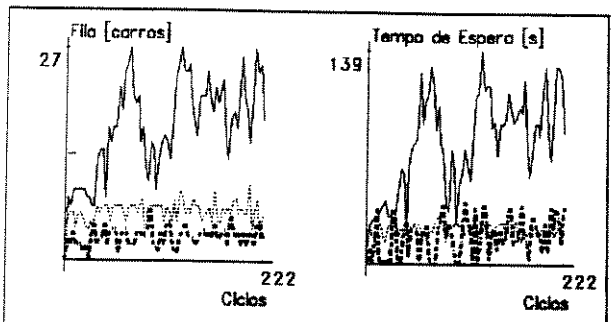
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	18.1	85.6	205.8
FSEM211.dat	3.0	19.9	41.0
FSEM212.dat	3.4	20.5	47.0

(b)



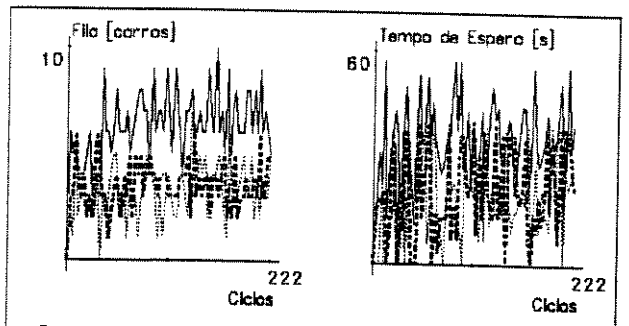
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	6.1	34.6	62.0
FSEM311.dat	6.4	22.6	32.0
FSEM312.dat	3.2	18.0	43.0

(c)



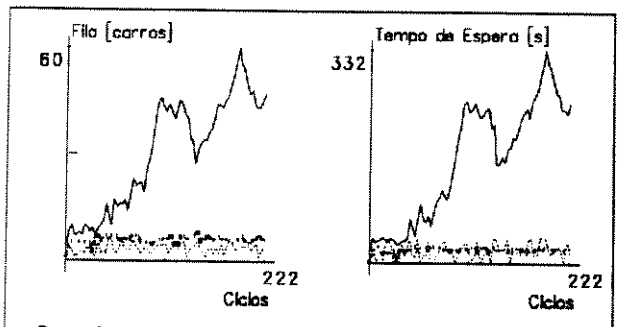
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	17.1	78.2	139.0
FSEM121.dat	6.2	22.6	32.7
FSEM122.dat	3.2	19.1	43.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	6.2	33.9	60.0
FSEM221.dat	3.1	18.5	41.0
FSEM222.dat	3.5	19.9	41.0

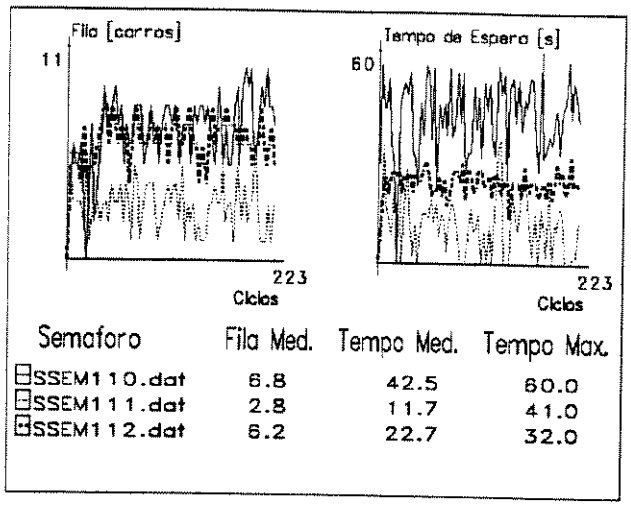
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	30.6	159.3	332.3
FSEM321.dat	3.0	19.2	43.0
FSEM322.dat	6.3	22.9	33.0

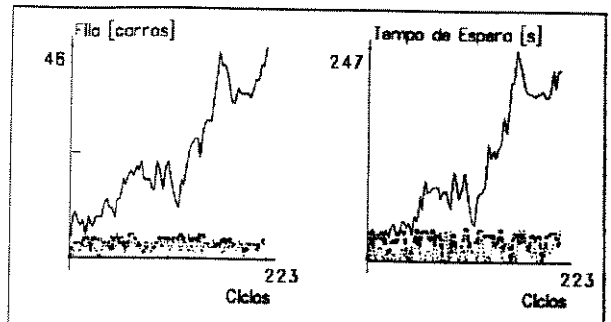
(f)

Figura A.11: Simultâneo: aumento de 20% no fluxo de entrada



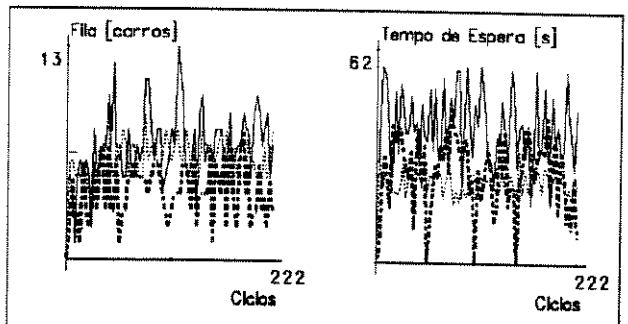
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	6.8	42.5	60.0
SSEM111.dat	2.8	11.7	41.0
SSEM112.dat	6.2	22.7	32.0

(a)



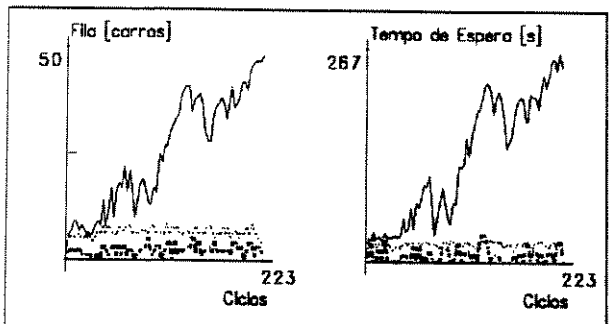
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	22.4	109.5	247.2
SSEM211.dat	2.7	13.0	41.0
SSEM212.dat	3.8	24.2	45.0

(b)



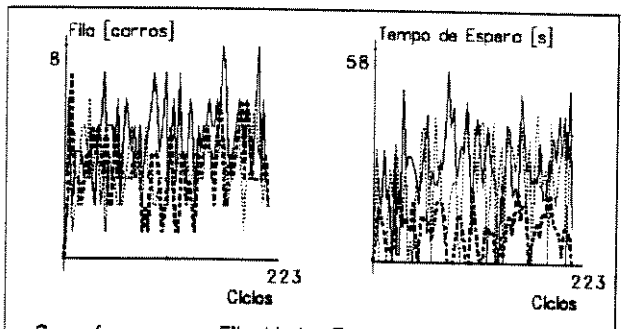
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	7.1	39.9	62.0
SSEM311.dat	6.4	23.6	34.0
SSEM312.dat	4.0	25.6	46.0

(c)



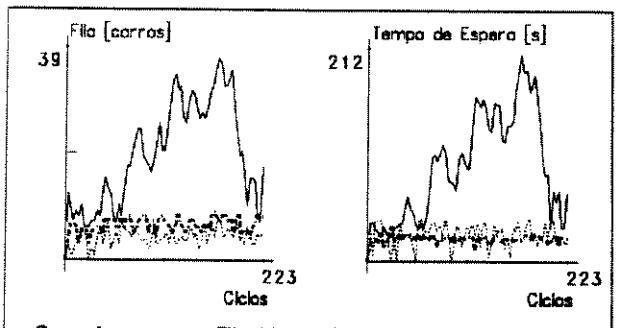
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	25.5	128.4	267.2
SSEM121.dat	6.4	22.6	32.0
SSEM122.dat	2.5	10.9	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	4.5	25.2	58.0
SSEM221.dat	3.5	22.0	41.0
SSEM222.dat	3.1	10.6	41.0

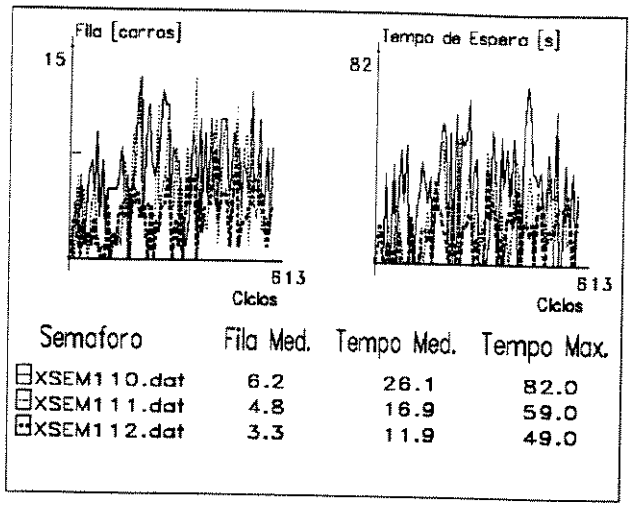
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	19.6	94.3	212.2
SSEM321.dat	4.1	25.5	43.0
SSEM322.dat	6.4	22.8	37.0

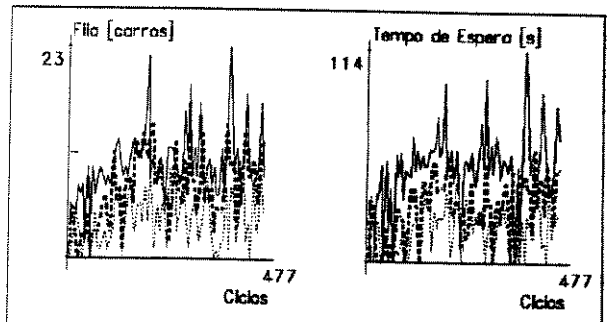
(f)

Figura A.12: Progressivo: aumento de 20% no fluxo de entrada



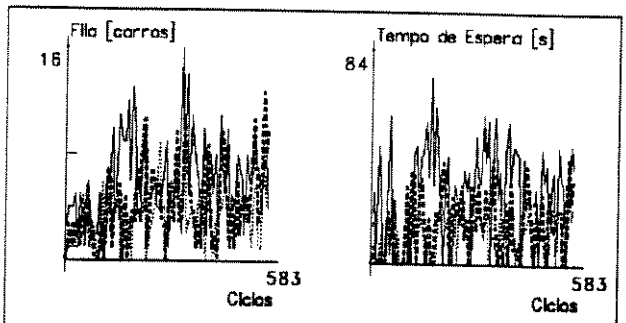
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	6.2	26.1	82.0
XSEM111.dat	4.8	16.9	59.0
XSEM112.dat	3.3	11.9	49.0

(a)



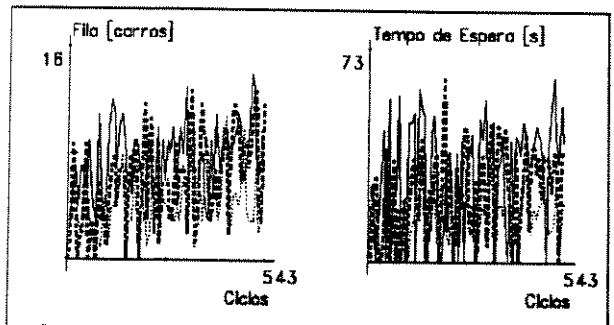
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	10.1	46.6	113.3
XSEM211.dat	4.3	17.2	60.0
XSEM212.dat	7.2	28.0	70.3

(b)



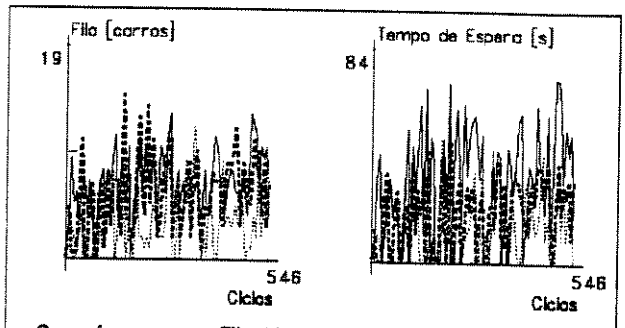
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	6.2	28.0	83.5
XSEM311.dat	3.5	12.5	42.7
XSEM312.dat	4.3	16.0	58.3

(c)



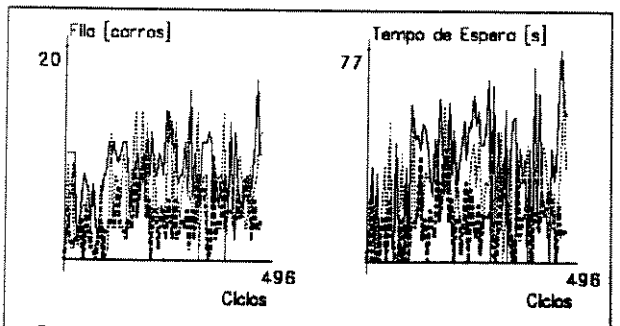
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	7.6	32.3	72.5
XSEM121.dat	4.2	15.6	48.0
XSEM122.dat	6.0	24.2	71.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	6.9	32.6	83.5
XSEM221.dat	3.5	12.8	54.0
XSEM222.dat	5.9	21.6	56.2

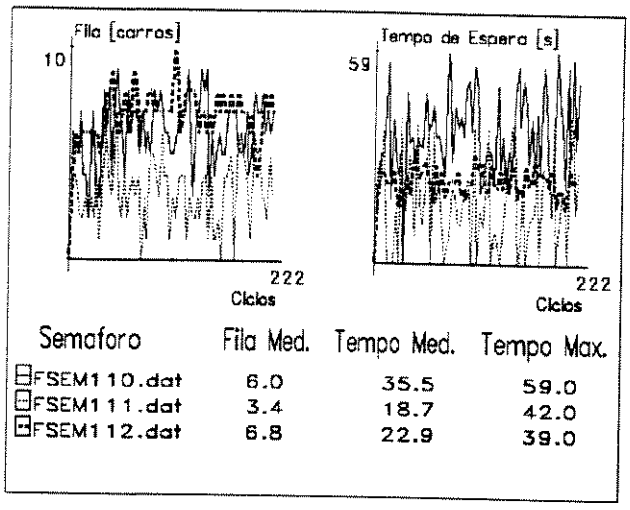
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	8.1	34.0	77.0
XSEM321.dat	6.2	22.7	64.0
XSEM322.dat	4.0	15.3	52.0

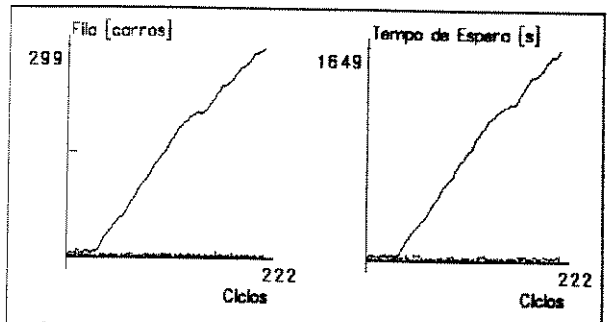
(f)

Figura A.13: ICD: aumento de 20% no fluxo de entrada



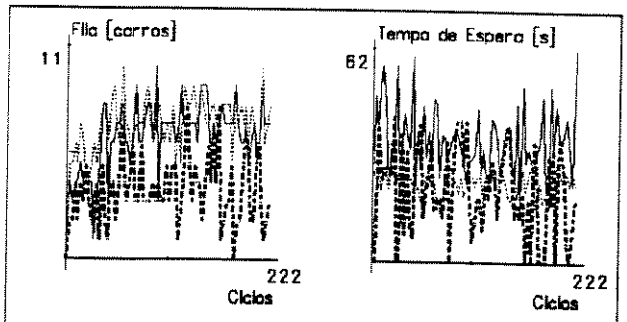
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	6.0	35.5	59.0
FSEM111.dat	3.4	18.7	42.0
FSEM112.dat	6.8	22.9	39.0

(a)



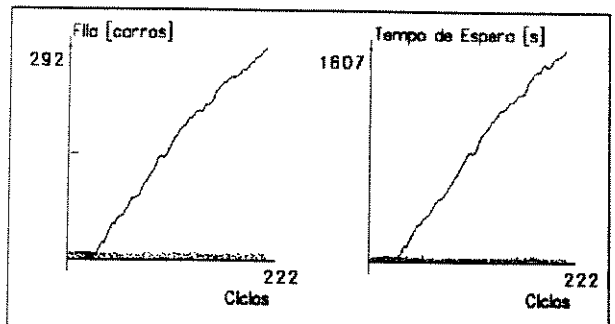
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	143.8	769.0	1649.0
FSEM211.dat	3.1	21.1	44.0
FSEM212.dat	3.6	21.5	49.0

(b)



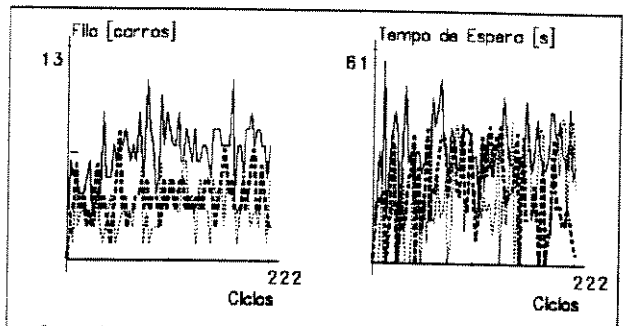
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	6.2	33.9	62.0
FSEM311.dat	7.0	22.9	32.0
FSEM312.dat	3.5	19.7	43.0

(c)



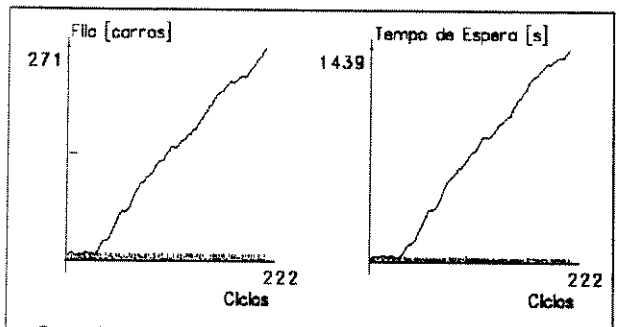
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	143.3	769.5	1607.0
FSEM121.dat	6.8	22.9	32.7
FSEM122.dat	3.3	19.4	43.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	6.4	34.2	60.5
FSEM221.dat	3.2	19.2	42.0
FSEM222.dat	3.6	20.1	41.0

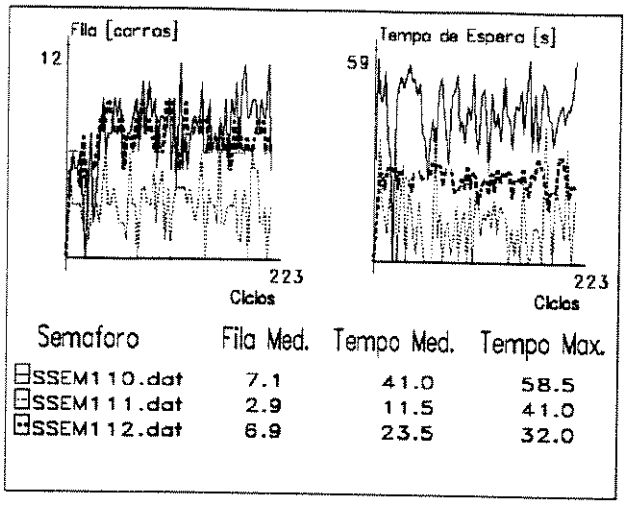
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	127.1	684.0	1438.8
FSEM321.dat	3.3	19.8	43.0
FSEM322.dat	6.8	22.5	34.0

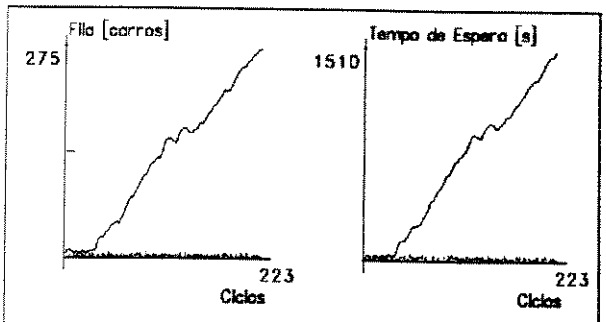
(f)

Figura A.14: Simultâneo: aumento de 30% no fluxo de entrada



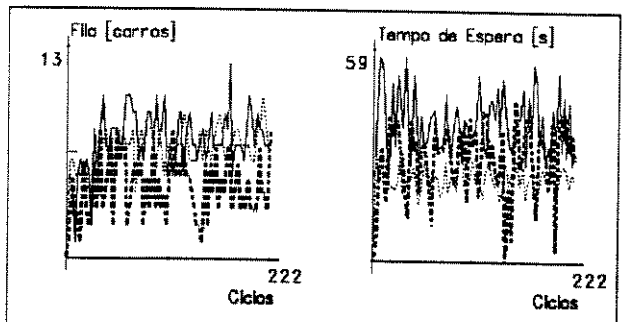
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	7.1	41.0	58.5
SSEM111.dat	2.9	11.5	41.0
SSEM112.dat	6.9	23.5	32.0

(a)



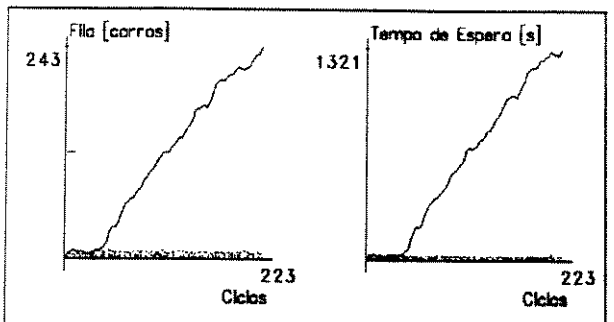
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	126.9	679.6	1509.8
SSEM211.dat	2.6	12.5	41.0
SSEM212.dat	4.3	26.8	45.0

(b)



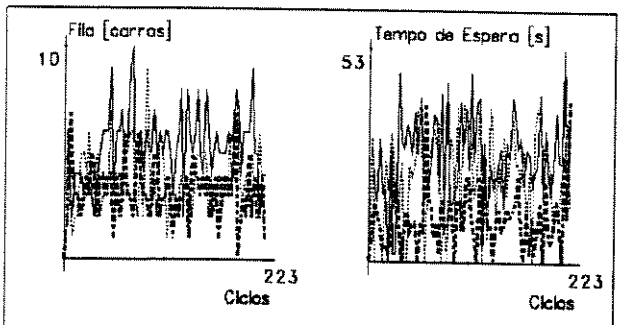
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	7.0	38.3	59.0
SSEM311.dat	6.9	23.2	31.0
SSEM312.dat	4.5	27.7	45.5

(c)



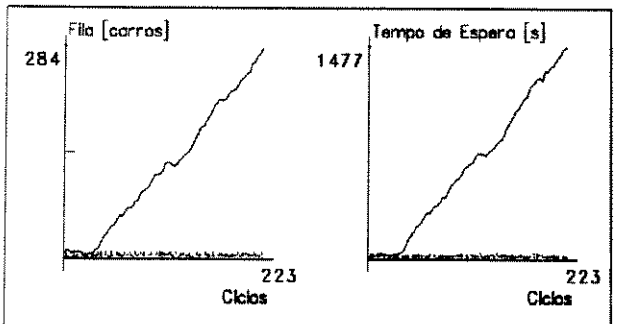
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	115.5	624.6	1320.5
SSEM121.dat	6.9	23.1	31.0
SSEM122.dat	2.7	13.5	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	5.2	25.3	53.0
SSEM221.dat	3.7	22.1	44.0
SSEM222.dat	3.0	10.4	41.0

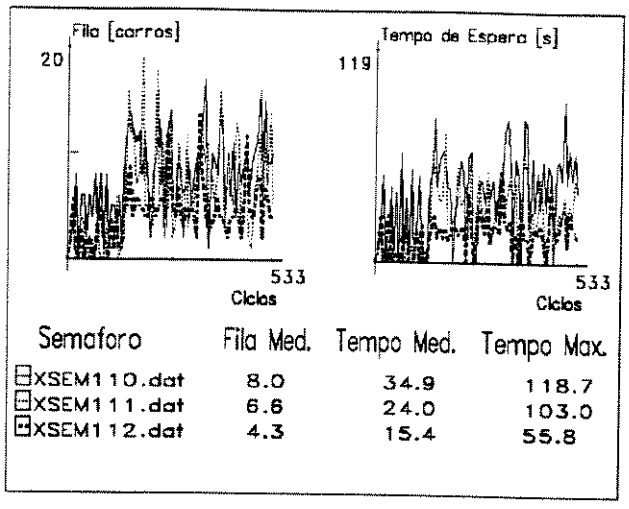
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	121.8	646.9	1477.3
SSEM321.dat	4.8	27.6	44.0
SSEM322.dat	6.9	23.2	37.0

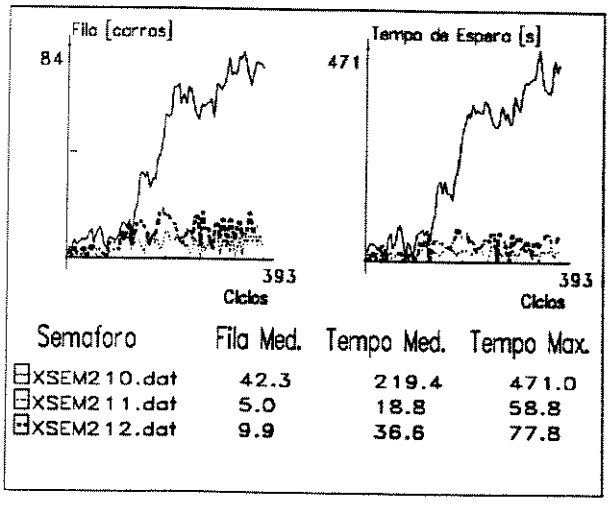
(f)

Figura A.15: Progressivo: aumento de 30% no fluxo de entrada



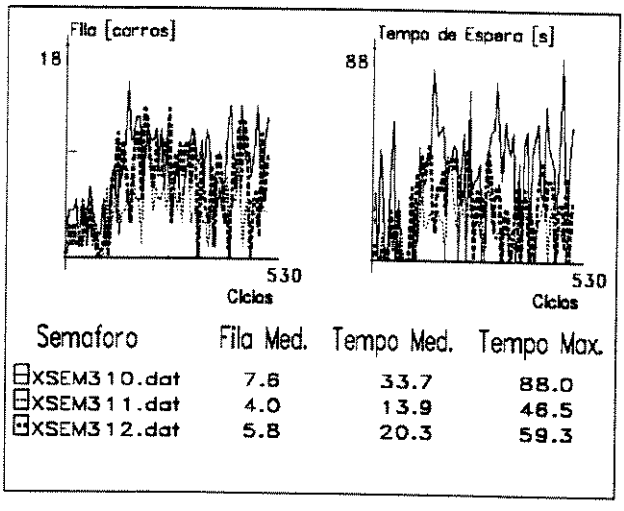
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	8.0	34.9	118.7
XSEM111.dat	6.6	24.0	103.0
XSEM112.dat	4.3	15.4	55.8

(a)



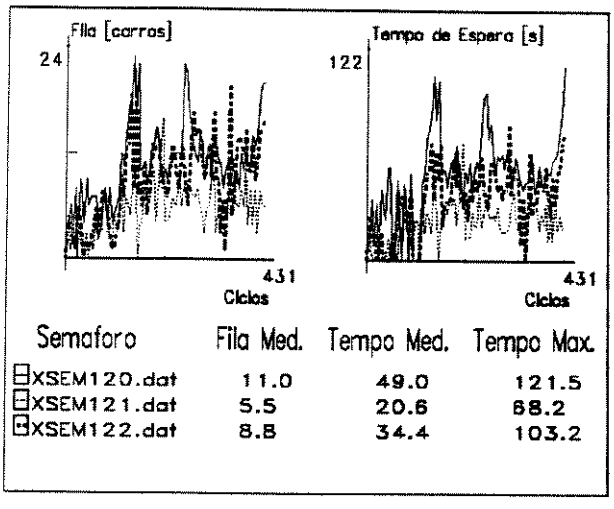
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	42.3	219.4	471.0
XSEM211.dat	5.0	18.8	58.8
XSEM212.dat	9.9	36.6	77.8

(b)



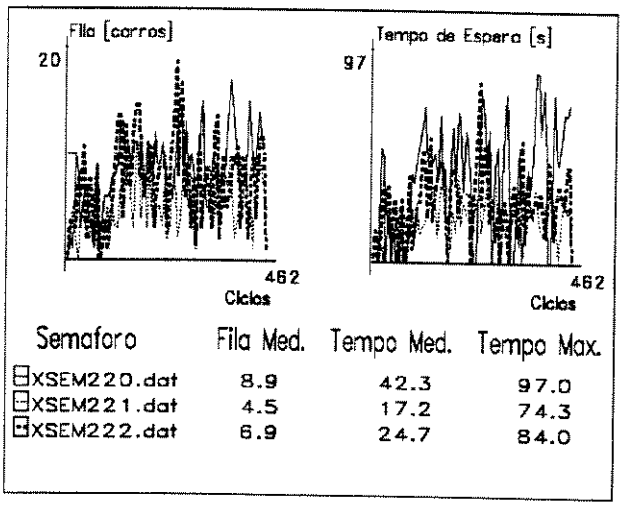
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	7.6	33.7	88.0
XSEM311.dat	4.0	13.9	46.5
XSEM312.dat	5.8	20.3	59.3

(c)



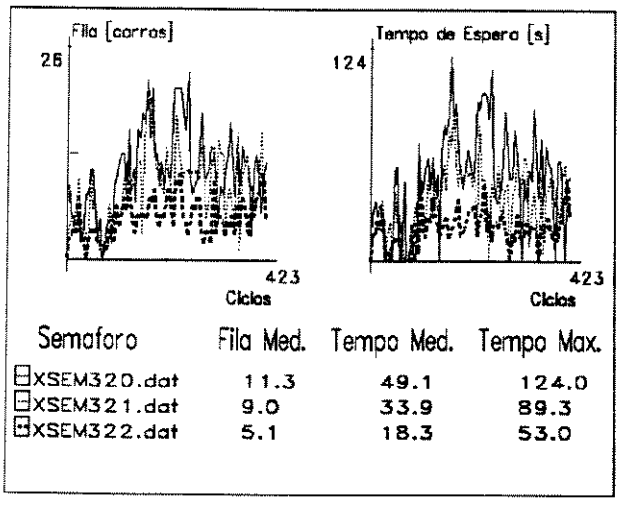
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	11.0	49.0	121.5
XSEM121.dat	5.5	20.6	68.2
XSEM122.dat	8.8	34.4	103.2

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	8.9	42.3	97.0
XSEM221.dat	4.5	17.2	74.3
XSEM222.dat	6.9	24.7	84.0

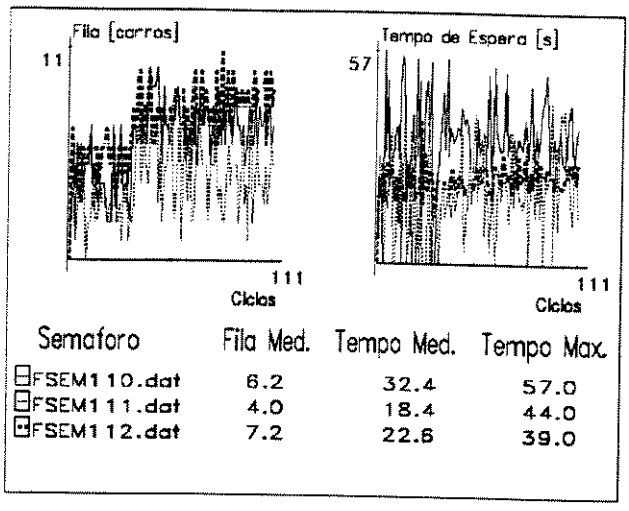
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	11.3	49.1	124.0
XSEM321.dat	9.0	33.9	89.3
XSEM322.dat	5.1	18.3	53.0

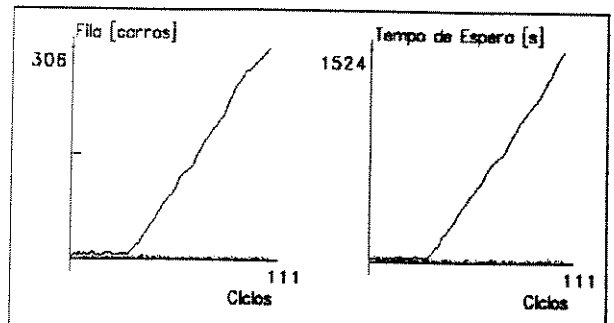
(f)

Figura A.16: ICD: aumento de 30% no fluxo de entrada



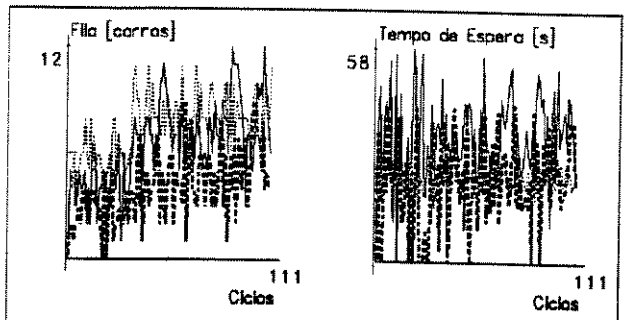
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	6.2	32.4	57.0
FSEM111.dat	4.0	18.4	44.0
FSEM112.dat	7.2	22.6	39.0

(a)



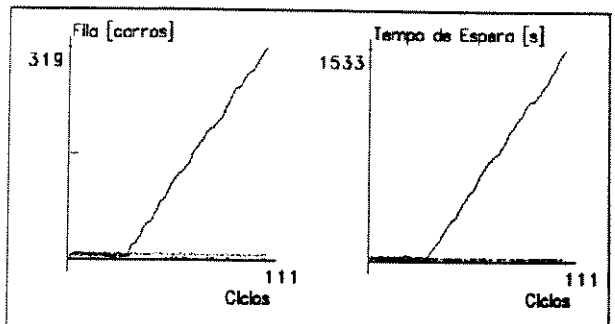
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	114.5	531.3	1523.8
FSEM211.dat	3.2	19.2	41.0
FSEM212.dat	4.1	18.5	42.0

(b)



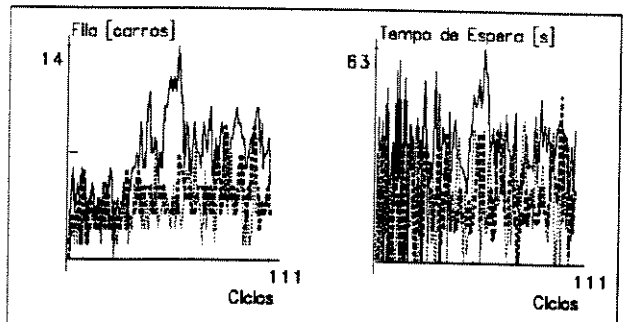
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	6.4	32.9	58.0
FSEM311.dat	7.3	23.2	32.0
FSEM312.dat	4.0	22.2	44.0

(c)



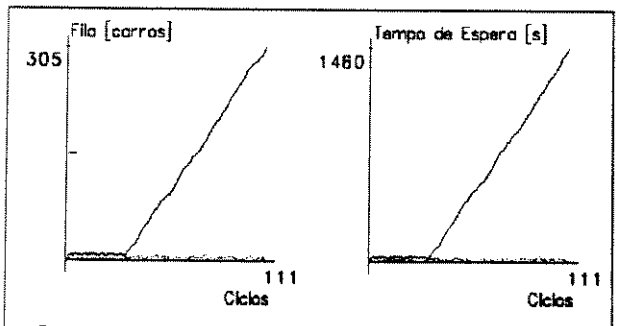
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	119.3	553.8	1533.2
FSEM121.dat	7.4	23.1	32.7
FSEM122.dat	4.1	21.2	43.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	6.6	33.8	63.3
FSEM221.dat	3.4	18.0	43.0
FSEM222.dat	3.9	19.0	51.0

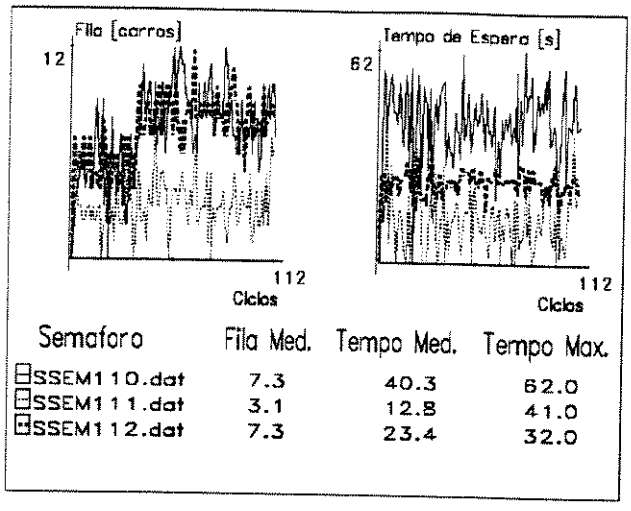
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	111.3	517.2	1460.2
FSEM321.dat	4.0	19.2	44.0
FSEM322.dat	7.3	22.7	33.0

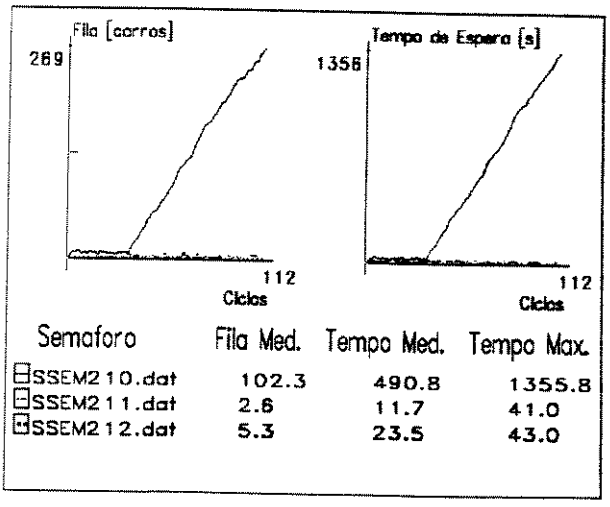
(f)

Figura A.17: Simultâneo: aumento de 50% no fluxo de entrada



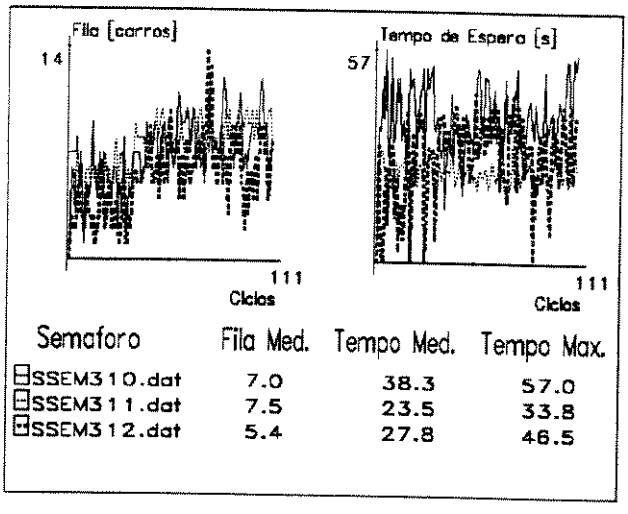
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	7.3	40.3	62.0
SSEM111.dat	3.1	12.8	41.0
SSEM112.dat	7.3	23.4	32.0

(a)



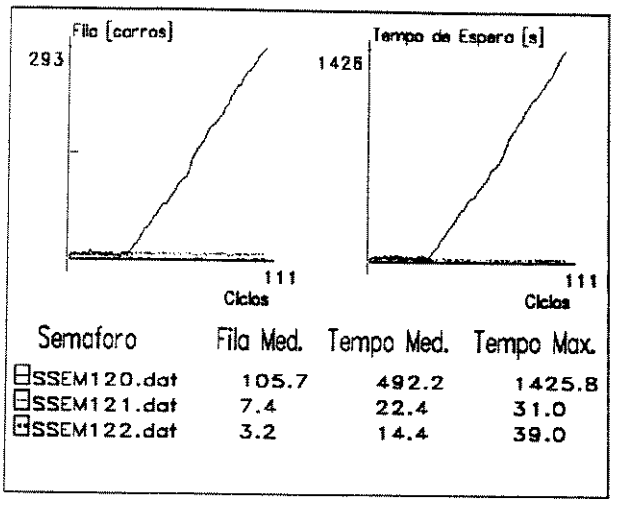
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	102.3	490.8	1355.8
SSEM211.dat	2.8	11.7	41.0
SSEM212.dat	5.3	23.5	43.0

(b)



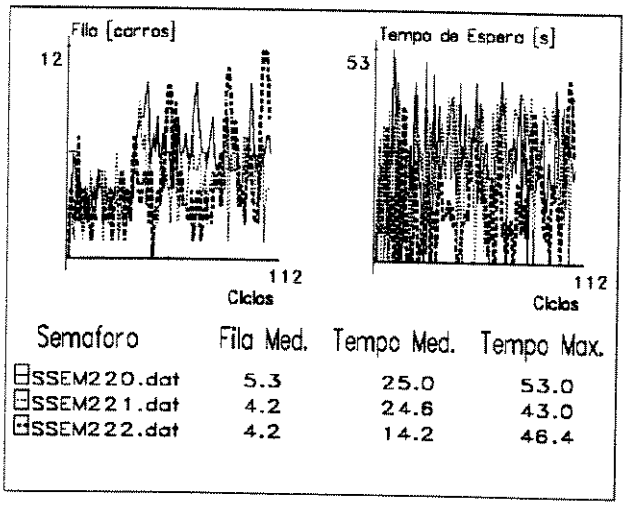
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	7.0	38.3	57.0
SSEM311.dat	7.5	23.5	33.8
SSEM312.dat	5.4	27.8	46.5

(c)



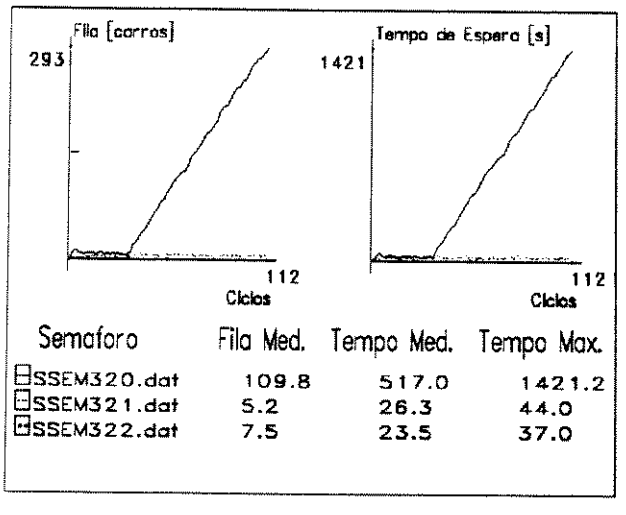
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	105.7	492.2	1425.8
SSEM121.dat	7.4	22.4	31.0
SSEM122.dat	3.2	14.4	39.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	5.3	25.0	53.0
SSEM221.dat	4.2	24.6	43.0
SSEM222.dat	4.2	14.2	46.4

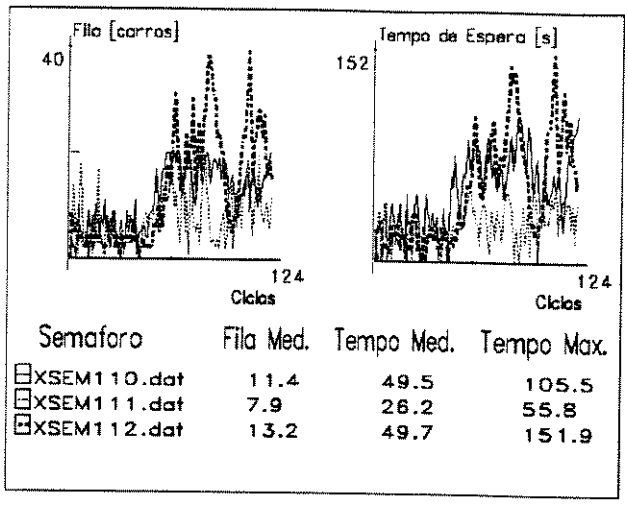
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	109.8	517.0	1421.2
SSEM321.dat	5.2	26.3	44.0
SSEM322.dat	7.5	23.5	37.0

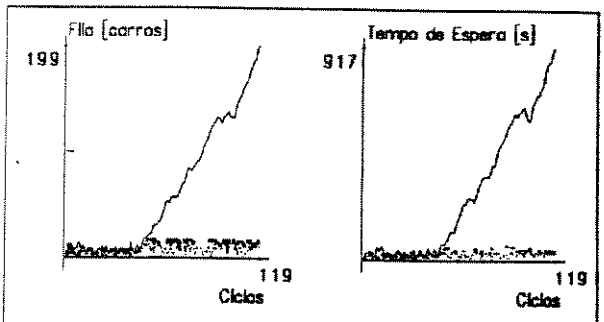
(f)

Figura A.18: Progressivo: aumento de 50% no fluxo de entrada



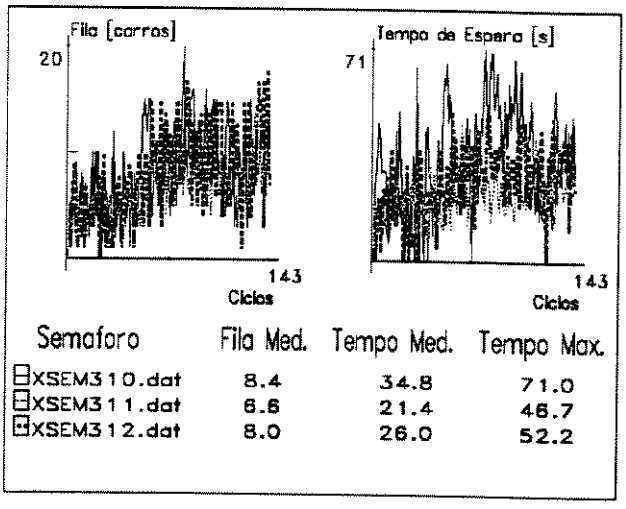
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	11.4	49.5	105.5
XSEM111.dat	7.9	26.2	55.8
XSEM112.dat	13.2	49.7	151.9

(a)



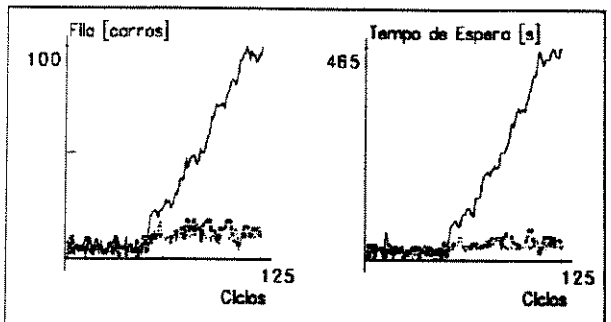
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	61.5	282.9	916.8
XSEM211.dat	7.2	25.0	59.4
XSEM212.dat	11.7	39.4	80.1

(b)



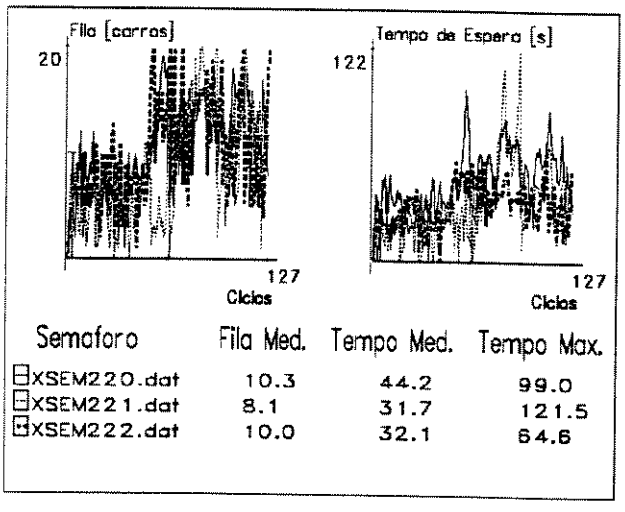
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	8.4	34.8	71.0
XSEM311.dat	6.6	21.4	46.7
XSEM312.dat	8.0	26.0	52.2

(c)



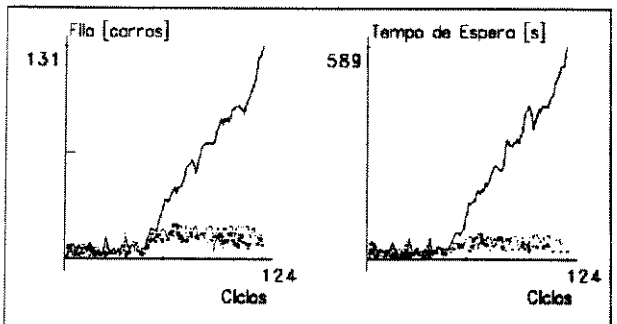
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	36.1	162.3	464.8
XSEM121.dat	8.1	25.8	55.2
XSEM122.dat	9.8	33.2	75.7

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	10.3	44.2	99.0
XSEM221.dat	8.1	31.7	121.5
XSEM222.dat	10.0	32.1	64.6

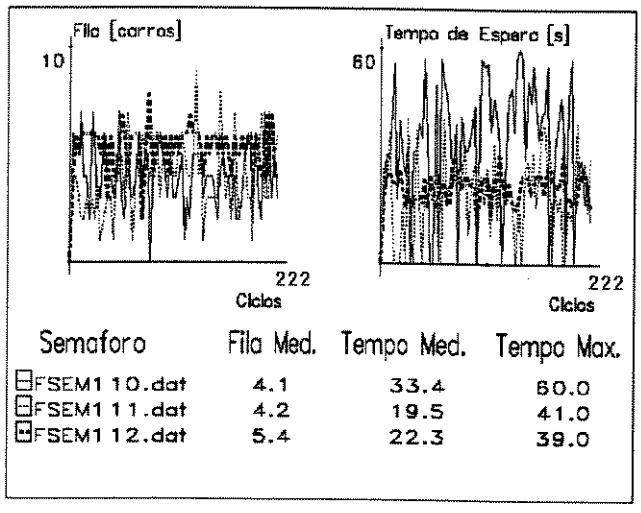
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	41.7	185.9	589.0
XSEM321.dat	11.8	39.3	78.2
XSEM322.dat	8.7	28.7	79.2

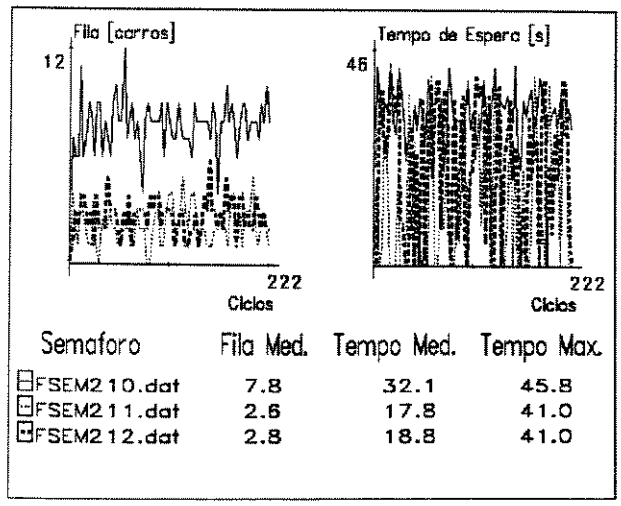
(f)

Figura A.19: ICD: aumento de 50% no fluxo de entrada



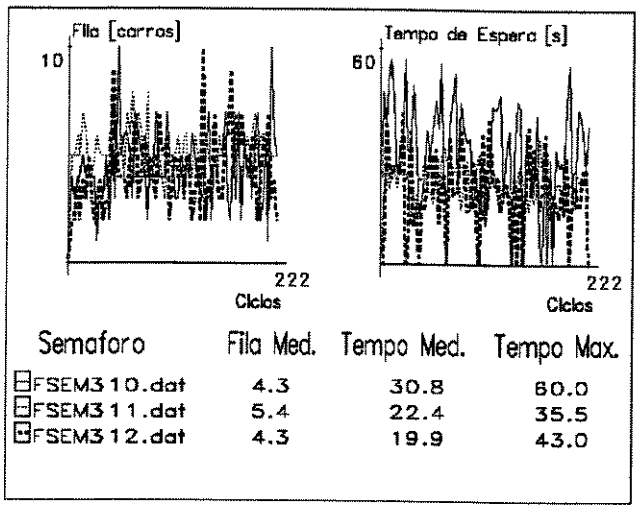
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	4.1	33.4	60.0
FSEM111.dat	4.2	19.5	41.0
FSEM112.dat	5.4	22.3	39.0

(a)



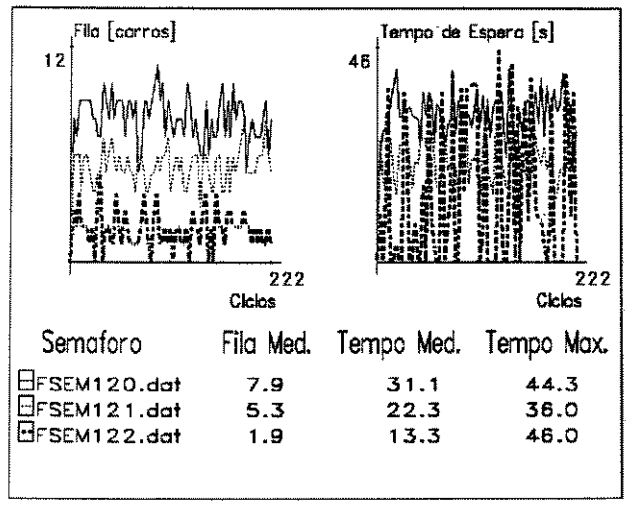
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	7.8	32.1	45.8
FSEM211.dat	2.6	17.8	41.0
FSEM212.dat	2.8	18.8	41.0

(b)



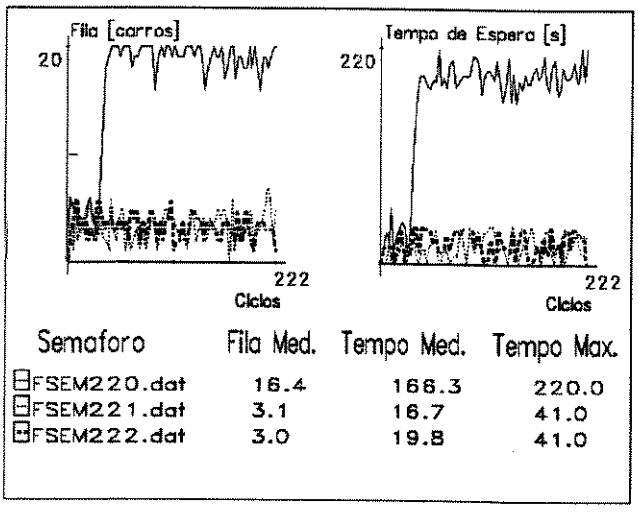
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	4.3	30.8	60.0
FSEM311.dat	5.4	22.4	35.5
FSEM312.dat	4.3	19.9	43.0

(c)



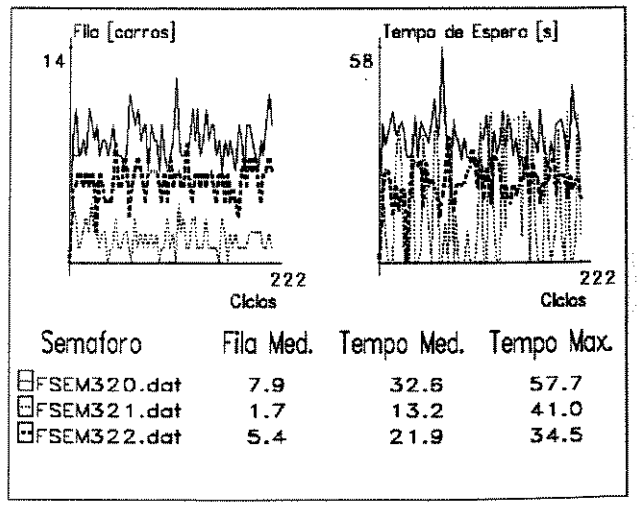
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	7.9	31.1	44.3
FSEM121.dat	5.3	22.3	36.0
FSEM122.dat	1.9	13.3	46.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	16.4	166.3	220.0
FSEM221.dat	3.1	16.7	41.0
FSEM222.dat	3.0	19.8	41.0

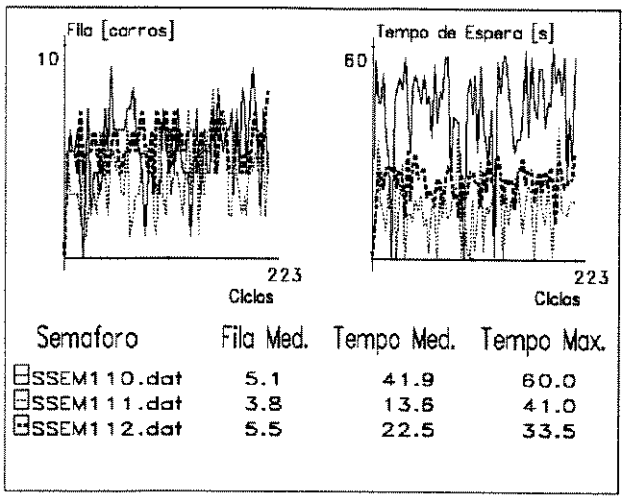
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	7.9	32.6	57.7
FSEM321.dat	1.7	13.2	41.0
FSEM322.dat	5.4	21.9	34.5

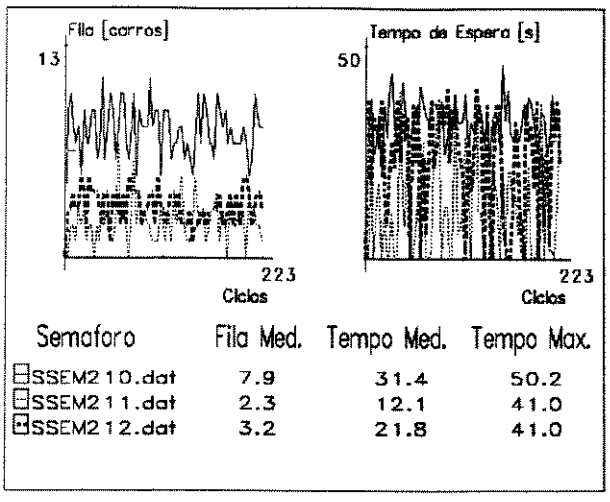
(f)

Figura A.20: Simultâneo: acidente de trânsito.



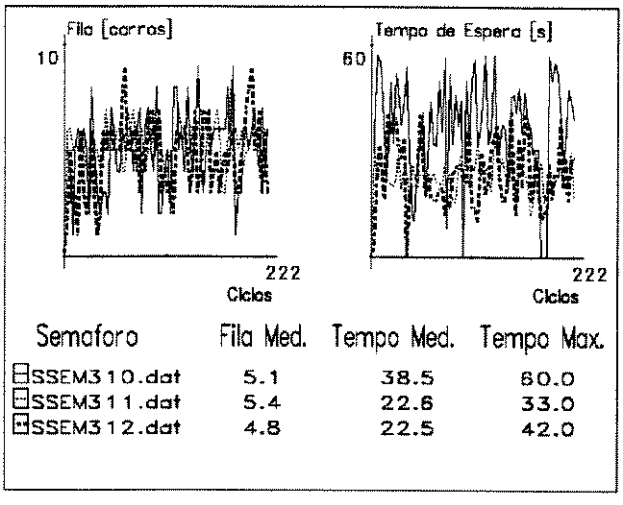
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	5.1	41.9	60.0
SSEM111.dat	3.8	13.6	41.0
SSEM112.dat	5.5	22.5	33.5

(a)



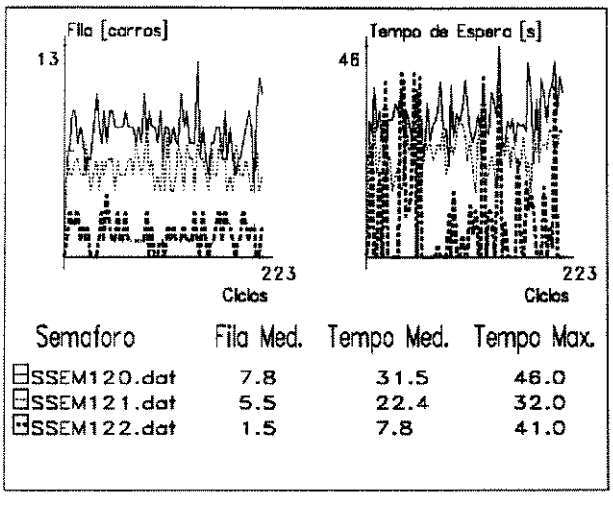
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	7.9	31.4	50.2
SSEM211.dat	2.3	12.1	41.0
SSEM212.dat	3.2	21.8	41.0

(b)



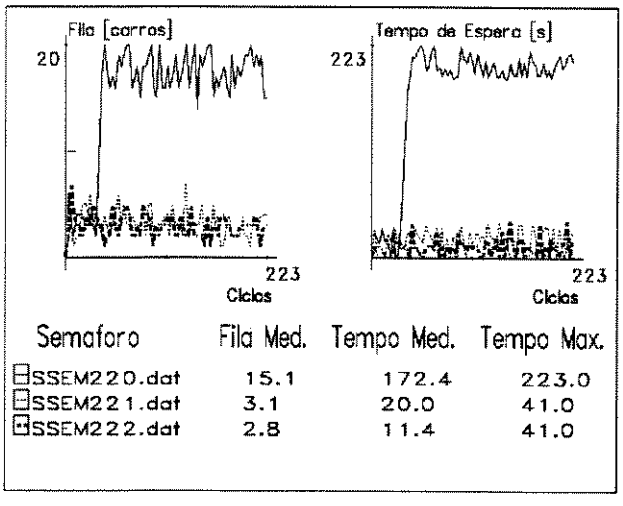
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	5.1	38.5	60.0
SSEM311.dat	5.4	22.6	33.0
SSEM312.dat	4.8	22.5	42.0

(c)



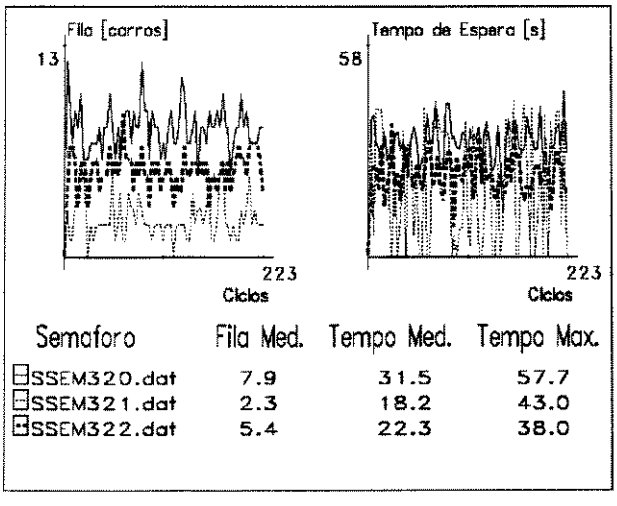
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	7.8	31.5	46.0
SSEM121.dat	5.5	22.4	32.0
SSEM122.dat	1.5	7.8	41.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	15.1	172.4	223.0
SSEM221.dat	3.1	20.0	41.0
SSEM222.dat	2.8	11.4	41.0

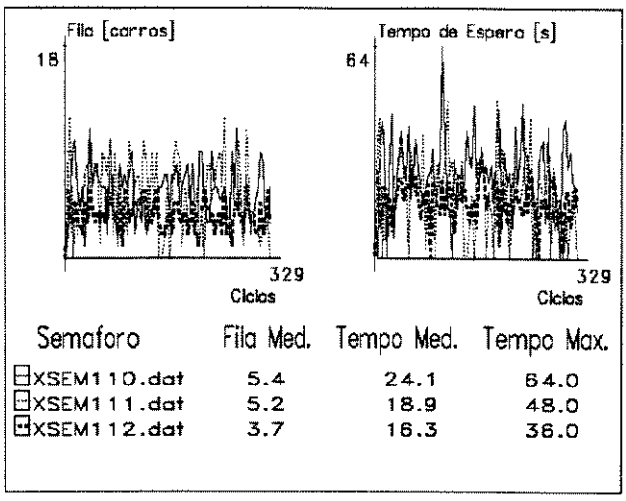
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	7.9	31.5	57.7
SSEM321.dat	2.3	18.2	43.0
SSEM322.dat	5.4	22.3	38.0

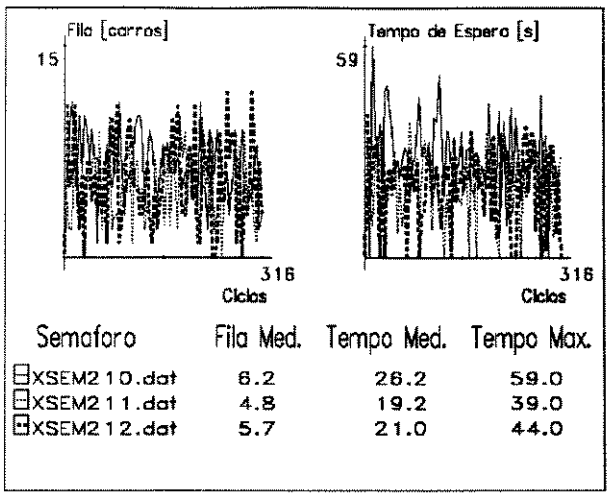
(f)

Figura A.21: Progressivo: acidente de trânsito.



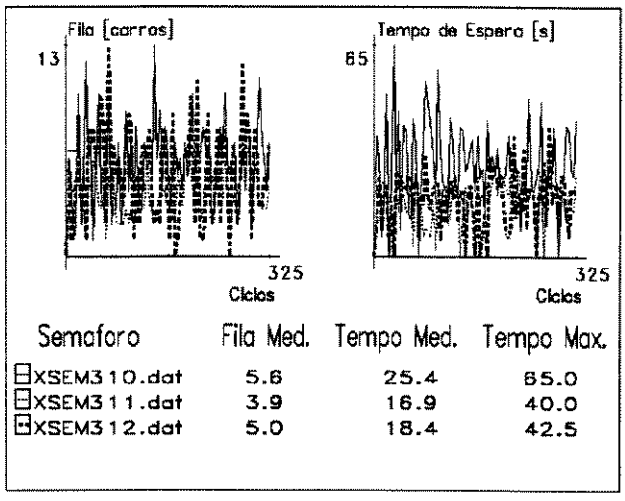
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	5.4	24.1	64.0
XSEM111.dat	5.2	18.9	48.0
XSEM112.dat	3.7	16.3	36.0

(a)



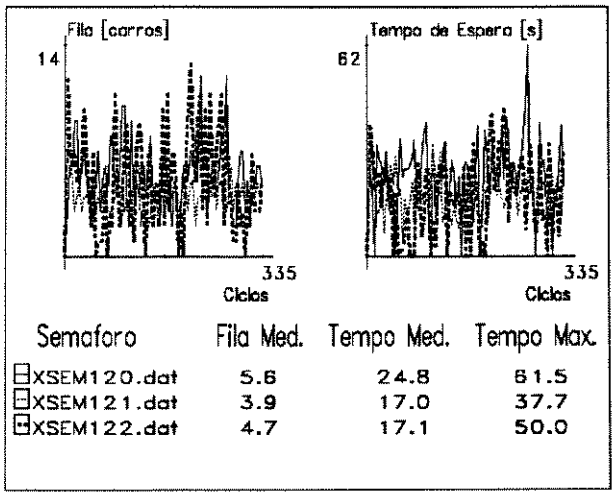
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	6.2	26.2	59.0
XSEM211.dat	4.8	19.2	39.0
XSEM212.dat	5.7	21.0	44.0

(b)



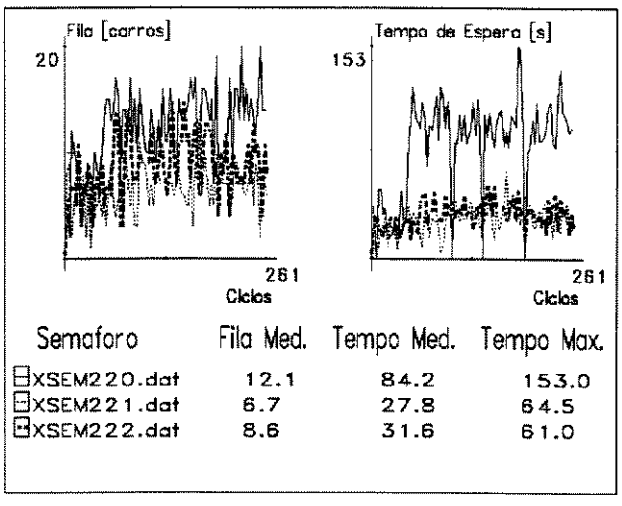
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	5.6	25.4	65.0
XSEM311.dat	3.9	16.9	40.0
XSEM312.dat	5.0	18.4	42.5

(c)



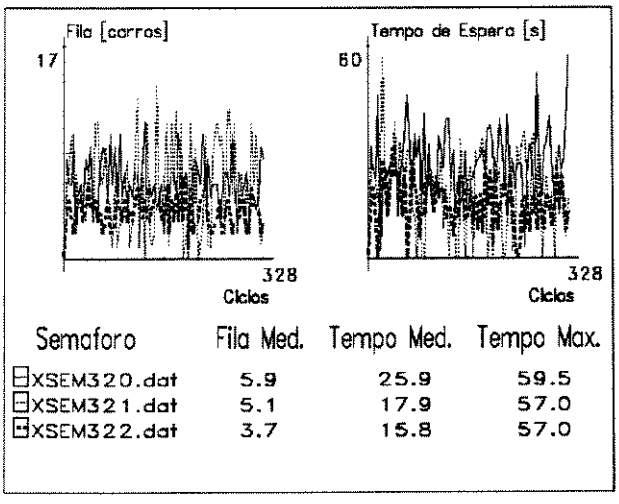
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	5.6	24.8	61.5
XSEM121.dat	3.9	17.0	37.7
XSEM122.dat	4.7	17.1	50.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	12.1	84.2	153.0
XSEM221.dat	6.7	27.8	64.5
XSEM222.dat	8.6	31.6	61.0

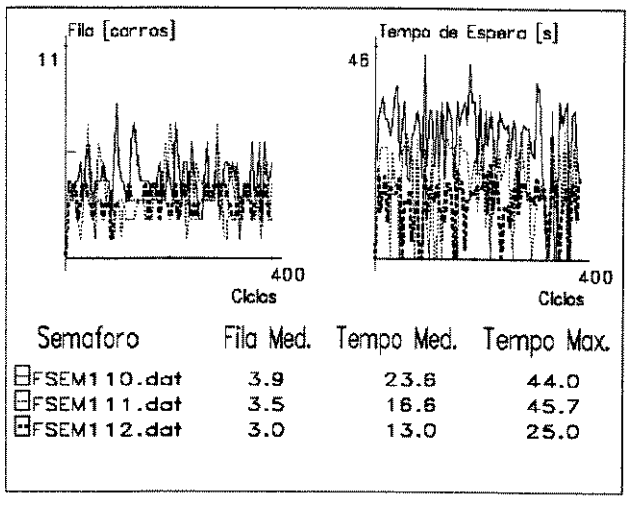
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	5.9	25.9	59.5
XSEM321.dat	5.1	17.9	57.0
XSEM322.dat	3.7	15.8	57.0

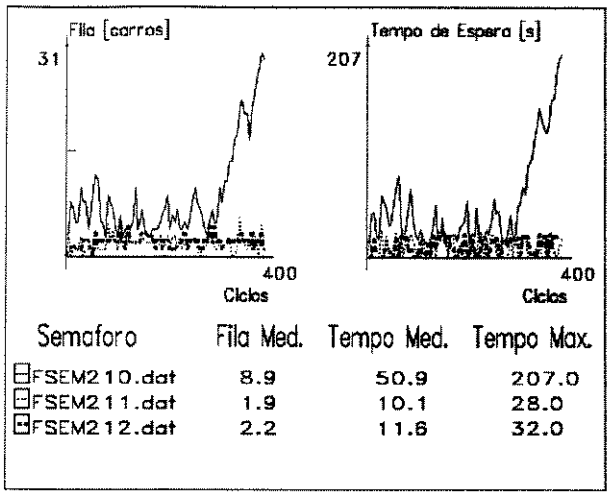
(f)

Figura A.22: ICD: acidente de trânsito



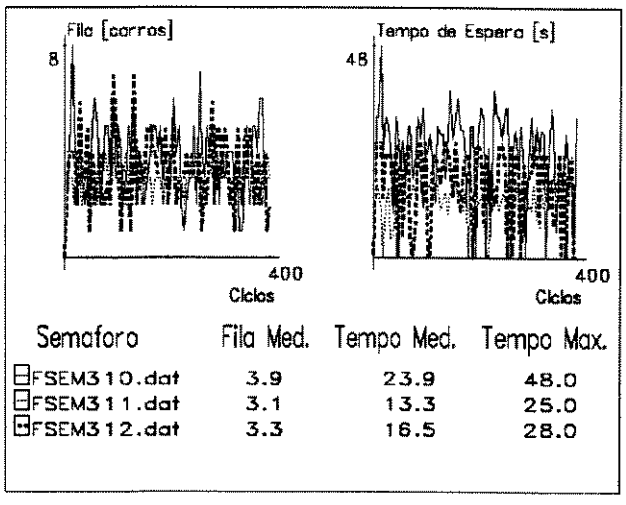
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	3.9	23.6	44.0
FSEM111.dat	3.5	16.6	45.7
FSEM112.dat	3.0	13.0	25.0

(a)



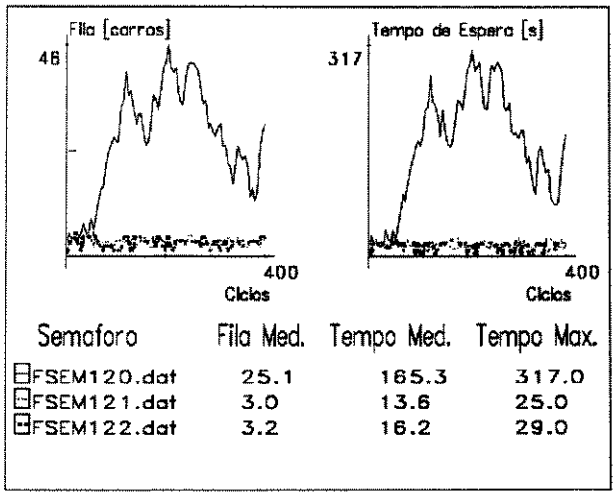
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	8.9	50.9	207.0
FSEM211.dat	1.9	10.1	28.0
FSEM212.dat	2.2	11.6	32.0

(b)



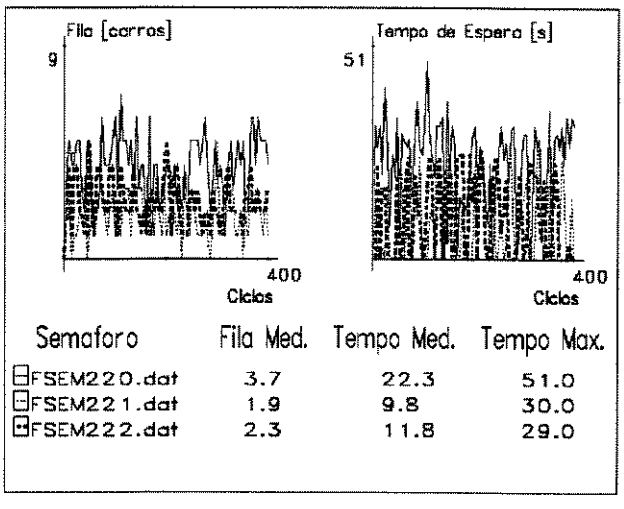
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	3.9	23.9	48.0
FSEM311.dat	3.1	13.3	25.0
FSEM312.dat	3.3	16.5	28.0

(c)



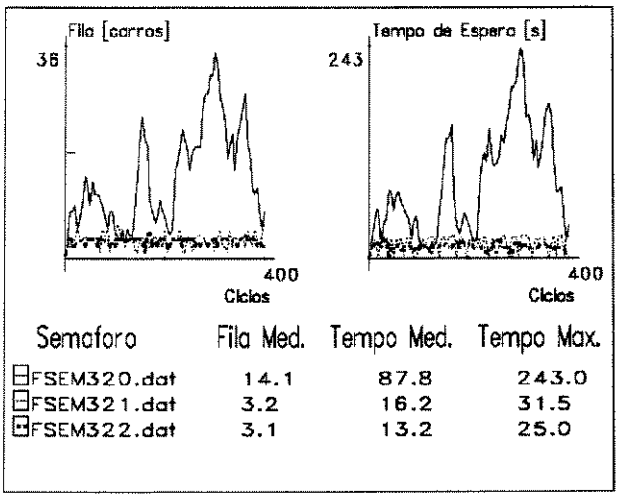
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	25.1	165.3	317.0
FSEM121.dat	3.0	13.6	25.0
FSEM122.dat	3.2	16.2	29.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	3.7	22.3	51.0
FSEM221.dat	1.9	9.8	30.0
FSEM222.dat	2.3	11.8	29.0

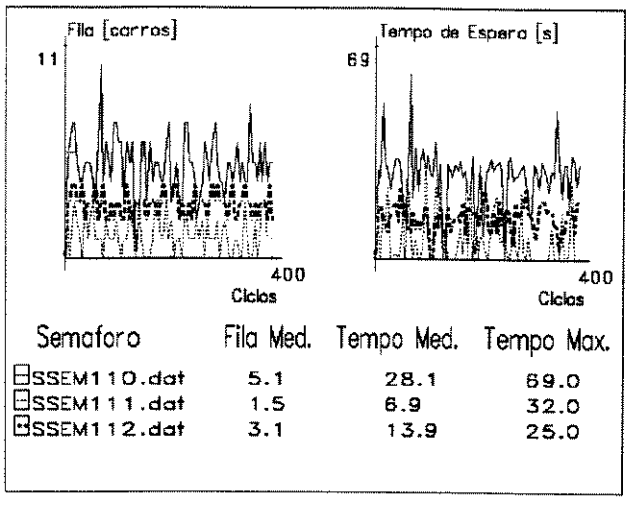
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	14.1	87.8	243.0
FSEM321.dat	3.2	16.2	31.5
FSEM322.dat	3.1	13.2	25.0

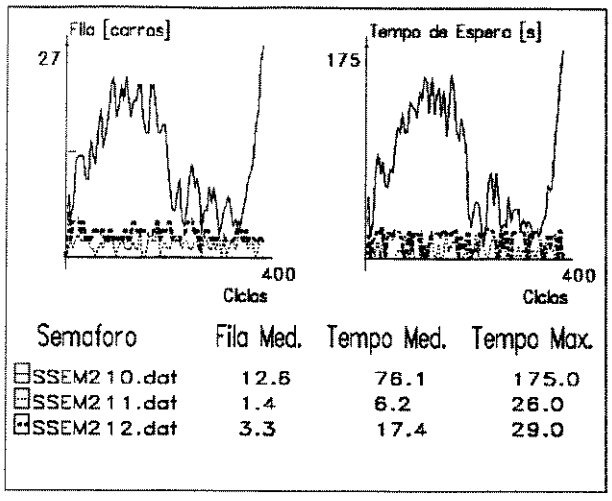
(f)

Figura A.23: Simultâneo: 100% do fluxo de entrada com ciclo calculado.



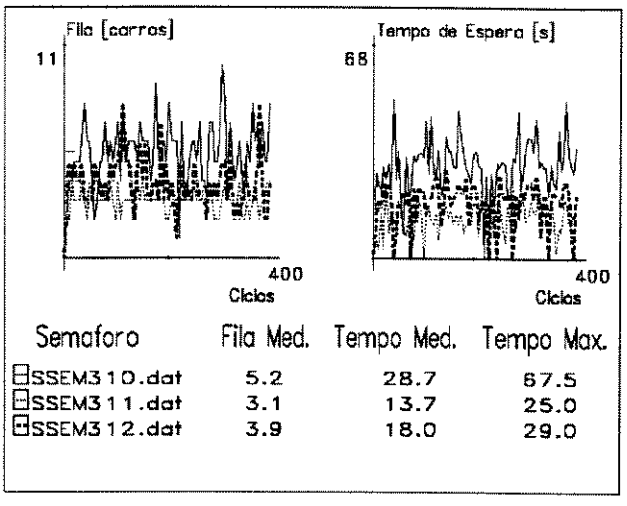
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	5.1	28.1	69.0
SSEM111.dat	1.5	6.9	32.0
SSEM112.dat	3.1	13.9	25.0

(a)



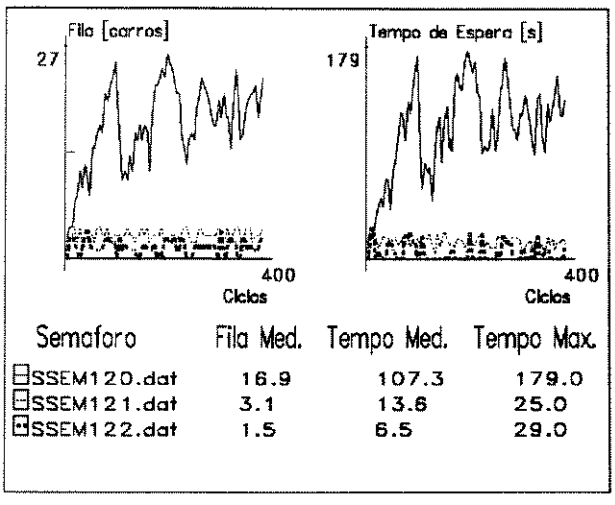
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	12.6	76.1	175.0
SSEM211.dat	1.4	6.2	26.0
SSEM212.dat	3.3	17.4	29.0

(b)



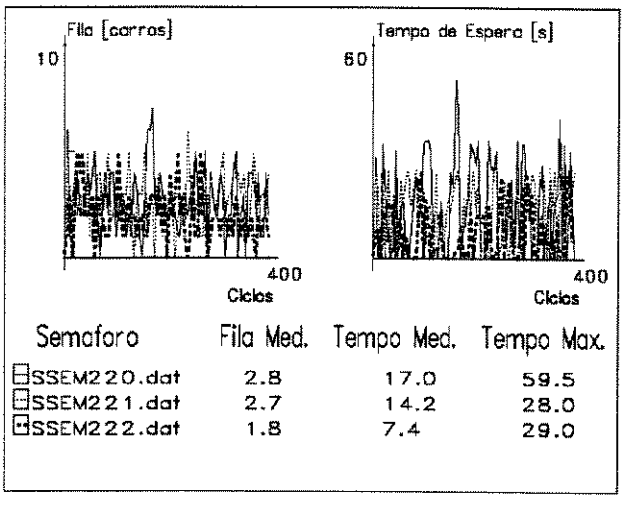
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	5.2	28.7	67.5
SSEM311.dat	3.1	13.7	25.0
SSEM312.dat	3.9	18.0	29.0

(c)



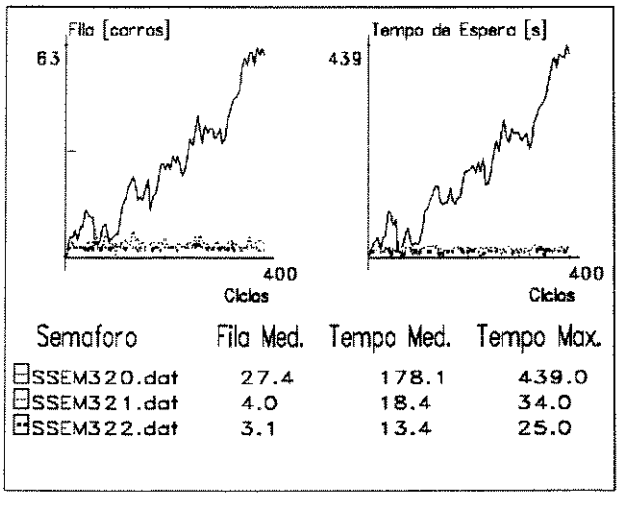
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	16.9	107.3	179.0
SSEM121.dat	3.1	13.6	25.0
SSEM122.dat	1.5	6.5	29.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	2.8	17.0	59.5
SSEM221.dat	2.7	14.2	28.0
SSEM222.dat	1.8	7.4	29.0

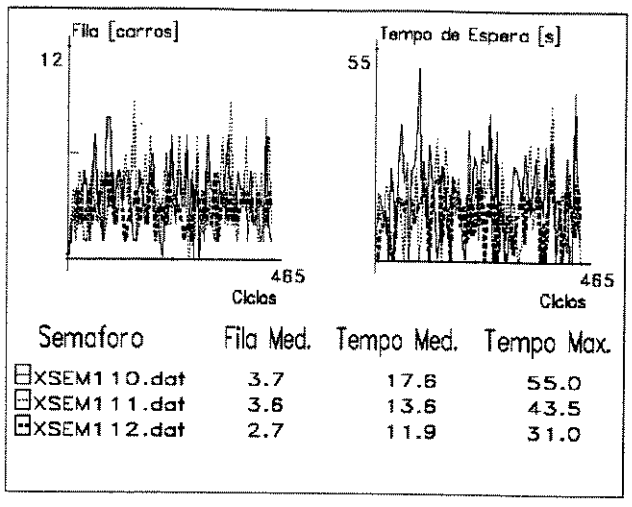
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	27.4	178.1	439.0
SSEM321.dat	4.0	18.4	34.0
SSEM322.dat	3.1	13.4	25.0

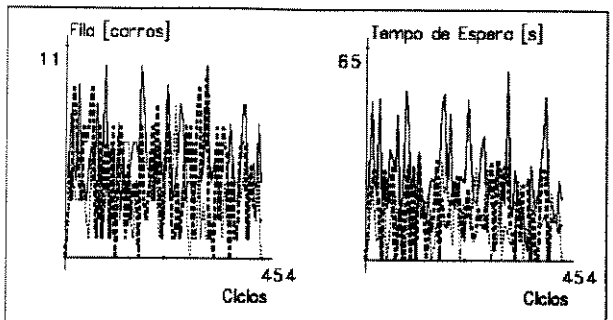
(f)

Figura A.24: Progressivo: 100% do fluxo de entrada com ciclo calculado.



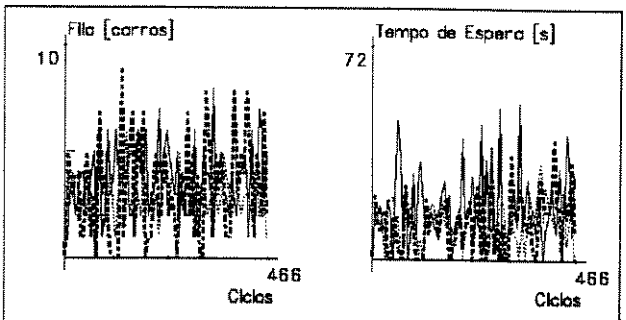
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	3.7	17.6	55.0
XSEM111.dat	3.6	13.6	43.5
XSEM112.dat	2.7	11.9	31.0

(a)



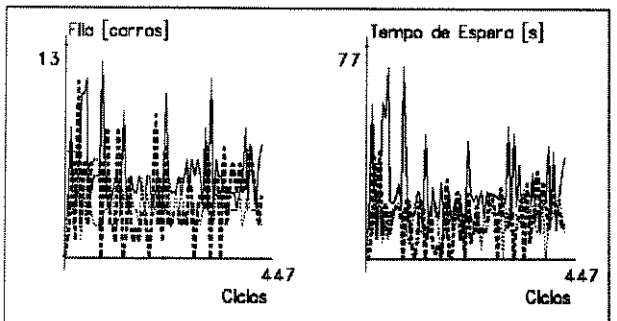
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	5.2	25.8	65.0
XSEM211.dat	3.4	13.3	41.5
XSEM212.dat	4.2	15.9	41.3

(b)



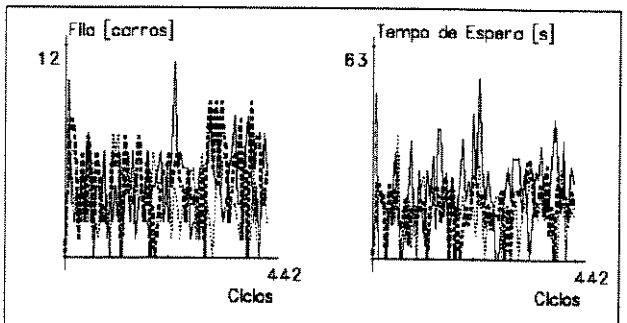
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	3.8	18.0	72.0
XSEM311.dat	2.8	12.2	36.0
XSEM312.dat	3.3	12.7	42.0

(c)



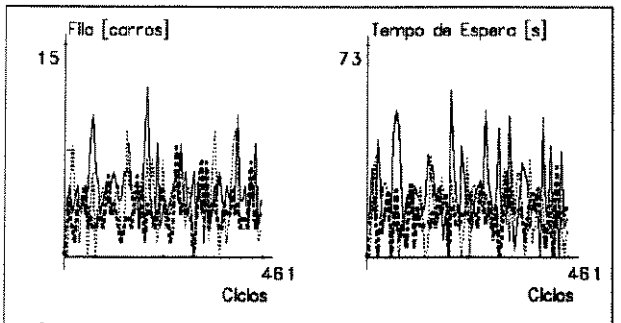
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	4.6	21.9	77.0
XSEM121.dat	3.0	13.0	34.5
XSEM122.dat	3.6	14.3	45.3

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	3.9	18.2	62.5
XSEM221.dat	3.3	13.1	39.0
XSEM222.dat	4.3	15.7	34.3

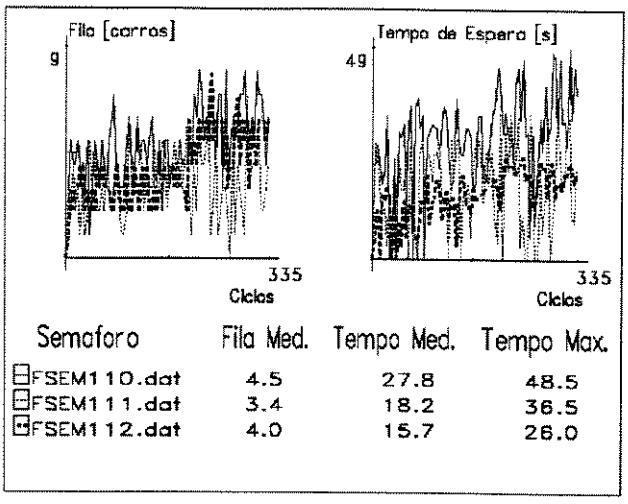
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	4.6	22.0	72.5
XSEM321.dat	3.8	14.9	47.0
XSEM322.dat	3.0	12.4	34.5

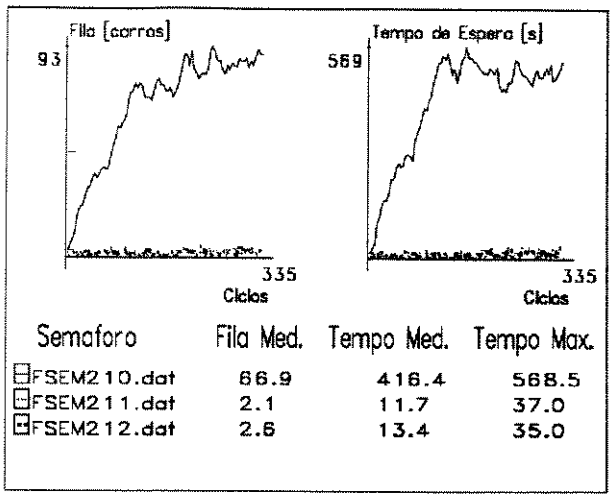
(f)

Figura A.25: ICD: 100% do fluxo de entrada com ciclo calculado.



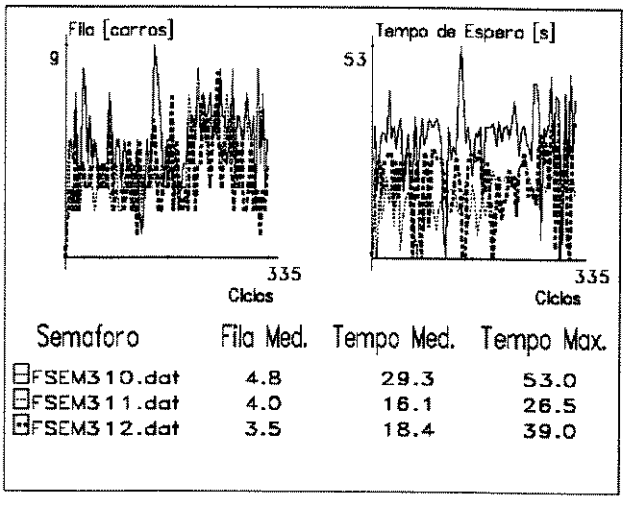
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	4.5	27.8	48.5
FSEM111.dat	3.4	18.2	36.5
FSEM112.dat	4.0	15.7	26.0

(a)



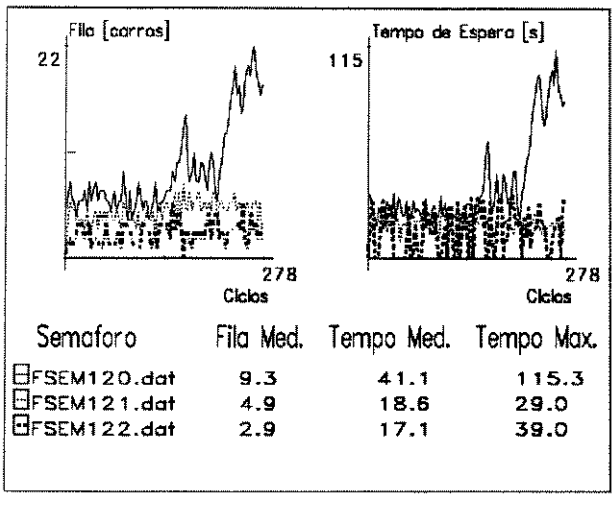
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	66.9	416.4	568.5
FSEM211.dat	2.1	11.7	37.0
FSEM212.dat	2.6	13.4	35.0

(b)



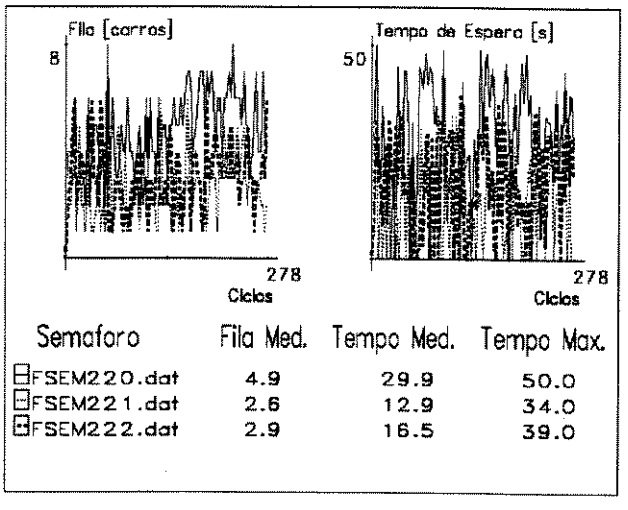
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	4.8	29.3	53.0
FSEM311.dat	4.0	16.1	26.5
FSEM312.dat	3.5	18.4	39.0

(c)



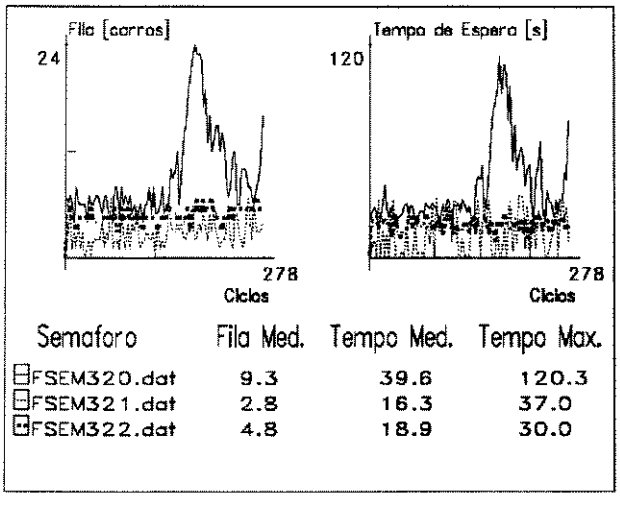
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	9.3	41.1	115.3
FSEM121.dat	4.9	18.6	29.0
FSEM122.dat	2.9	17.1	39.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	4.9	29.9	50.0
FSEM221.dat	2.6	12.9	34.0
FSEM222.dat	2.9	16.5	39.0

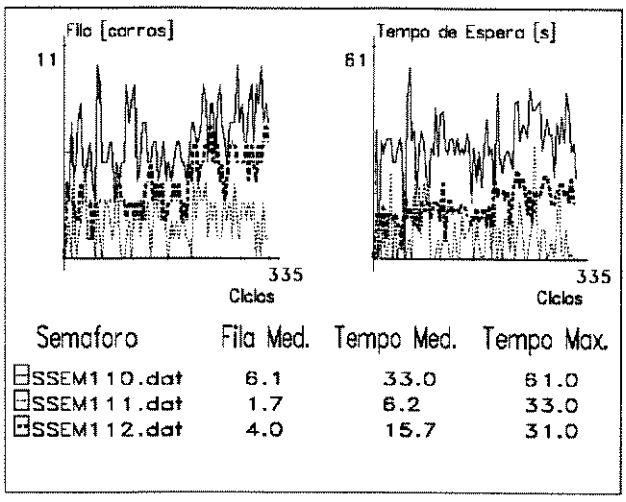
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	9.3	39.6	120.3
FSEM321.dat	2.8	16.3	37.0
FSEM322.dat	4.8	18.9	30.0

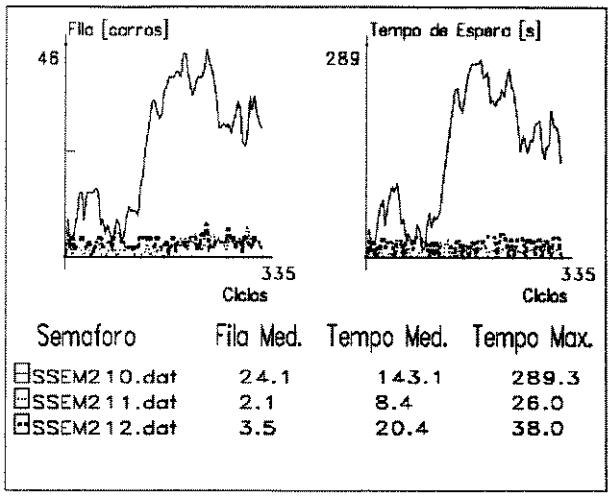
(f)

Figura A.26: Simultâneo: fluxo em degrau



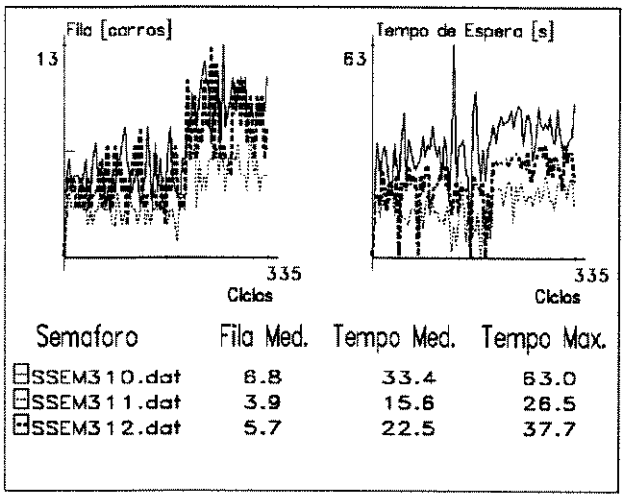
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	6.1	33.0	61.0
SSEM111.dat	1.7	6.2	33.0
SSEM112.dat	4.0	15.7	31.0

(a)



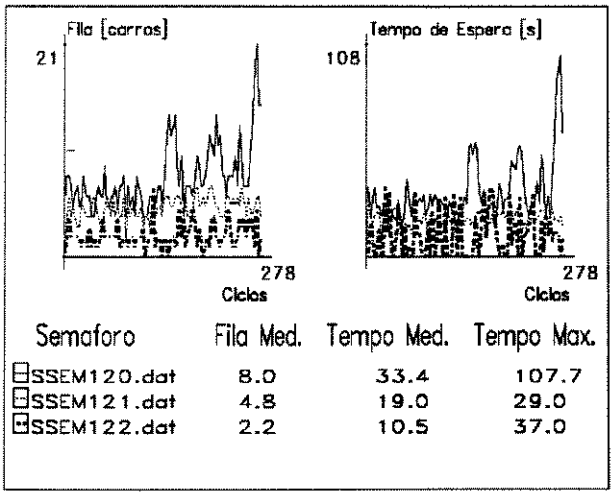
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	24.1	143.1	289.3
SSEM211.dat	2.1	8.4	26.0
SSEM212.dat	3.5	20.4	38.0

(b)



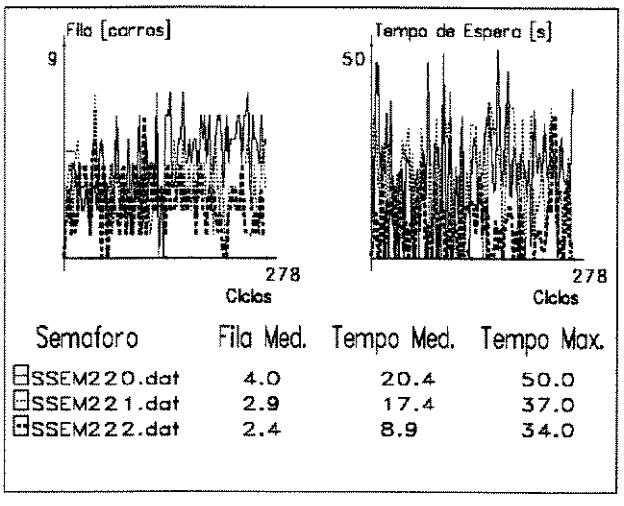
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	6.8	33.4	63.0
SSEM311.dat	3.9	15.6	26.5
SSEM312.dat	5.7	22.5	37.7

(c)



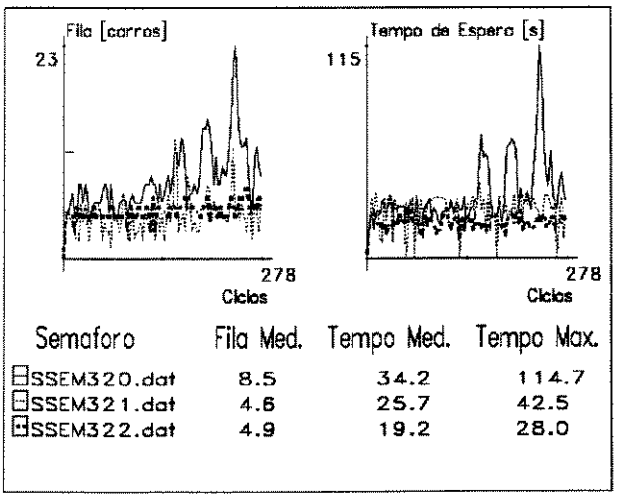
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	8.0	33.4	107.7
SSEM121.dat	4.8	19.0	29.0
SSEM122.dat	2.2	10.5	37.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	4.0	20.4	50.0
SSEM221.dat	2.9	17.4	37.0
SSEM222.dat	2.4	8.9	34.0

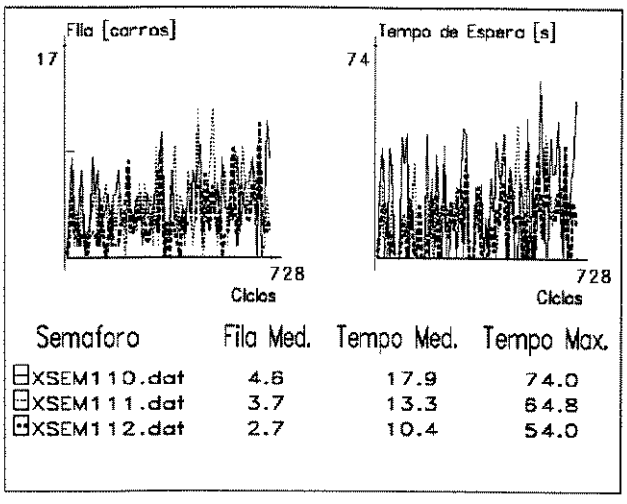
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	8.5	34.2	114.7
SSEM321.dat	4.6	25.7	42.5
SSEM322.dat	4.9	19.2	28.0

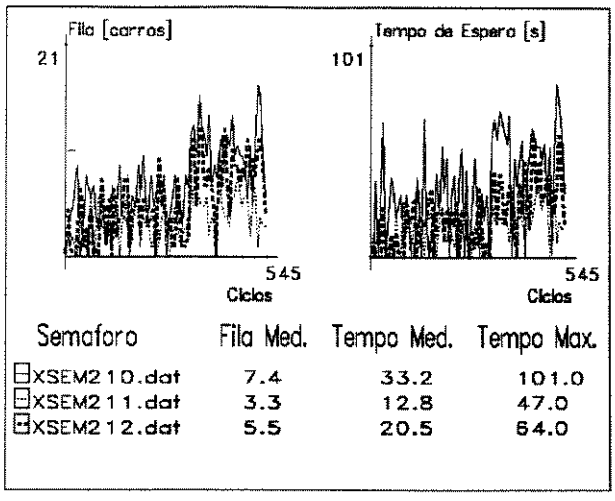
(f)

Figura A.27: Progressivo: fluxo em degrau



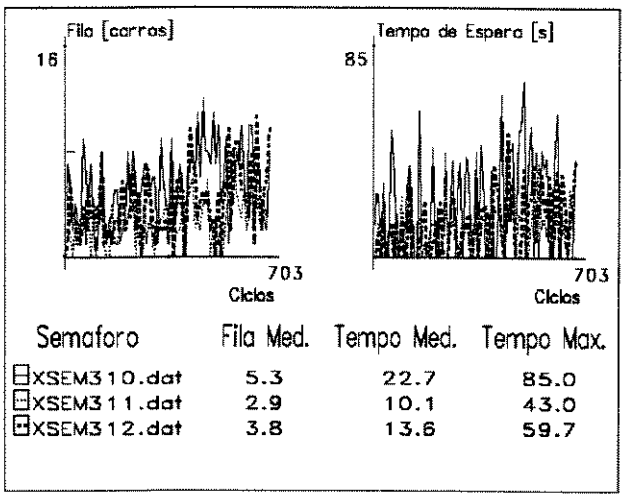
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	4.6	17.9	74.0
XSEM111.dat	3.7	13.3	64.8
XSEM112.dat	2.7	10.4	54.0

(a)



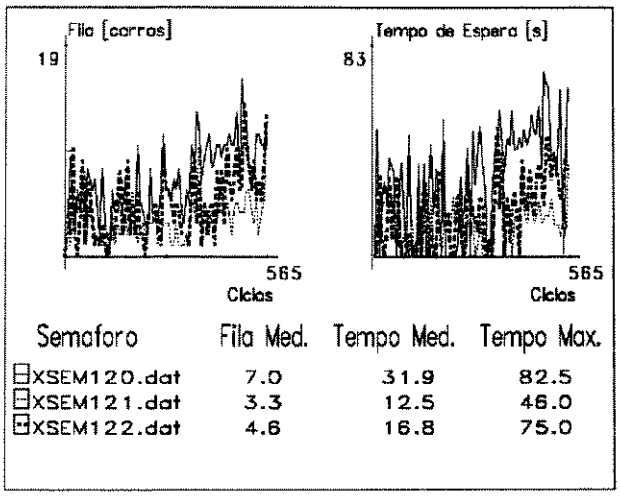
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	7.4	33.2	101.0
XSEM211.dat	3.3	12.8	47.0
XSEM212.dat	5.5	20.5	64.0

(b)



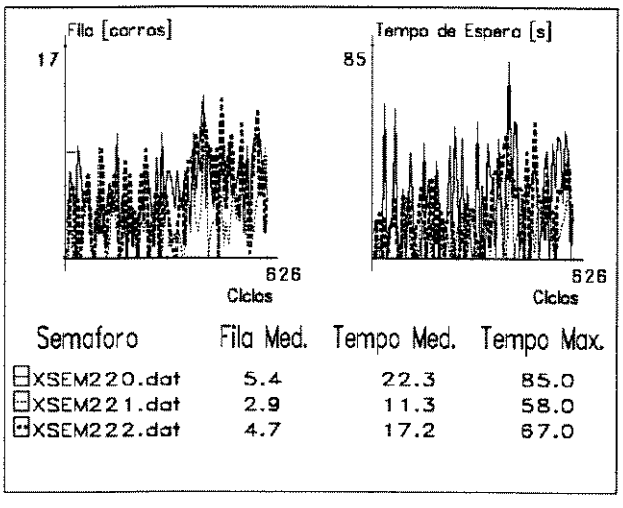
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	5.3	22.7	85.0
XSEM311.dat	2.9	10.1	43.0
XSEM312.dat	3.8	13.6	59.7

(c)



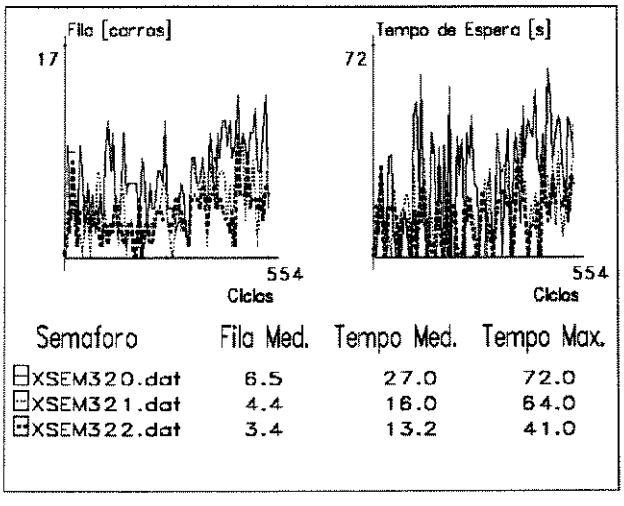
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	7.0	31.9	82.5
XSEM121.dat	3.3	12.5	46.0
XSEM122.dat	4.6	16.8	75.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	5.4	22.3	85.0
XSEM221.dat	2.9	11.3	58.0
XSEM222.dat	4.7	17.2	67.0

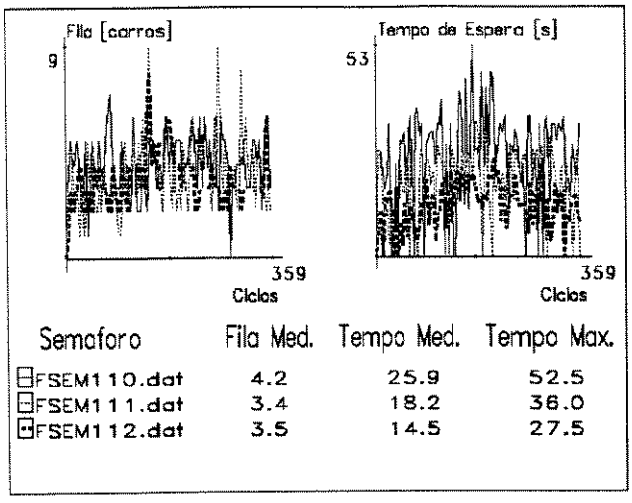
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	6.5	27.0	72.0
XSEM321.dat	4.4	16.0	64.0
XSEM322.dat	3.4	13.2	41.0

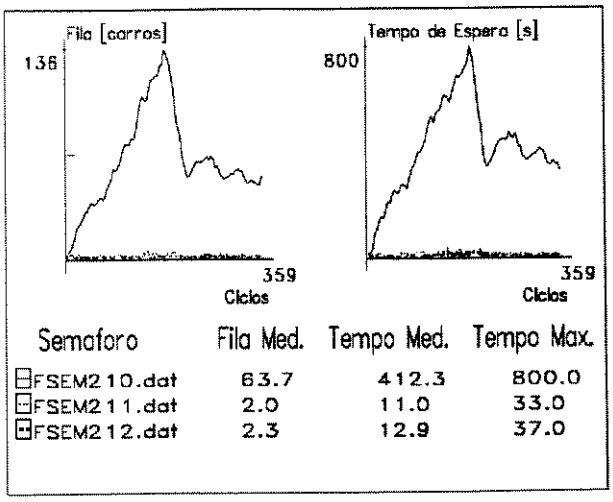
(f)

Figura A.28: ICD: fluxo em degrau



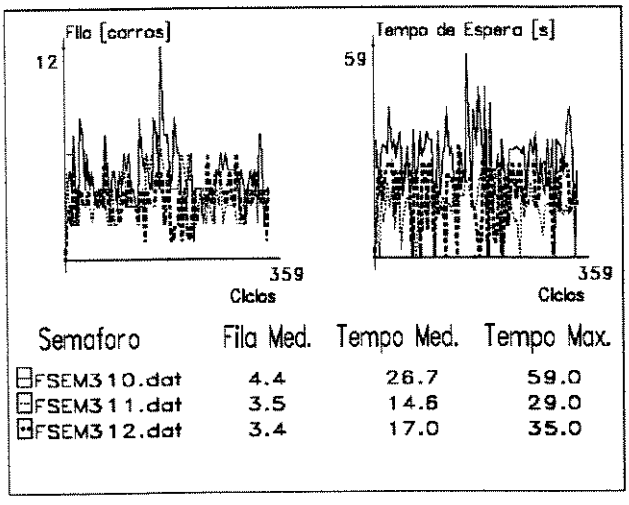
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM110.dat	4.2	25.9	52.5
FSEM111.dat	3.4	18.2	36.0
FSEM112.dat	3.5	14.5	27.5

(a)



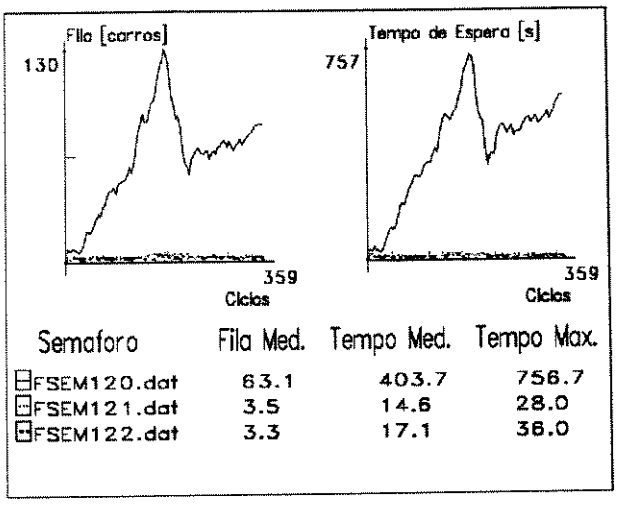
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM210.dat	63.7	412.3	800.0
FSEM211.dat	2.0	11.0	33.0
FSEM212.dat	2.3	12.9	37.0

(b)



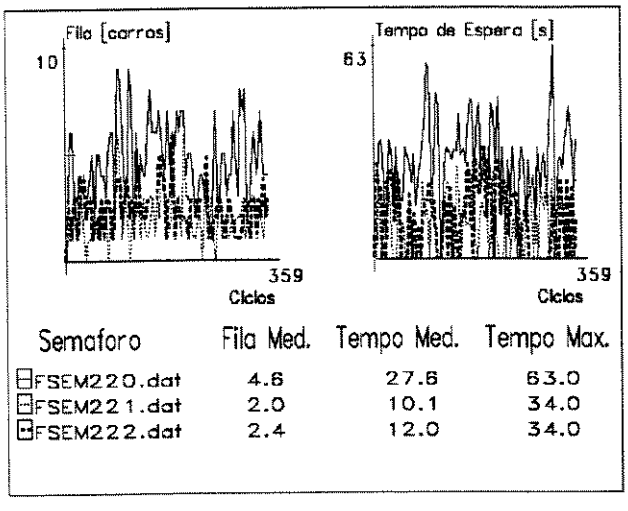
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM310.dat	4.4	26.7	59.0
FSEM311.dat	3.5	14.6	29.0
FSEM312.dat	3.4	17.0	35.0

(c)



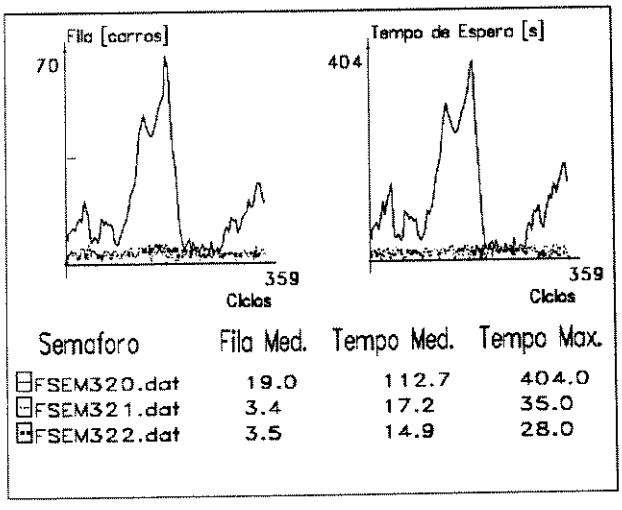
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM120.dat	63.1	403.7	756.7
FSEM121.dat	3.5	14.6	28.0
FSEM122.dat	3.3	17.1	36.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM220.dat	4.6	27.6	63.0
FSEM221.dat	2.0	10.1	34.0
FSEM222.dat	2.4	12.0	34.0

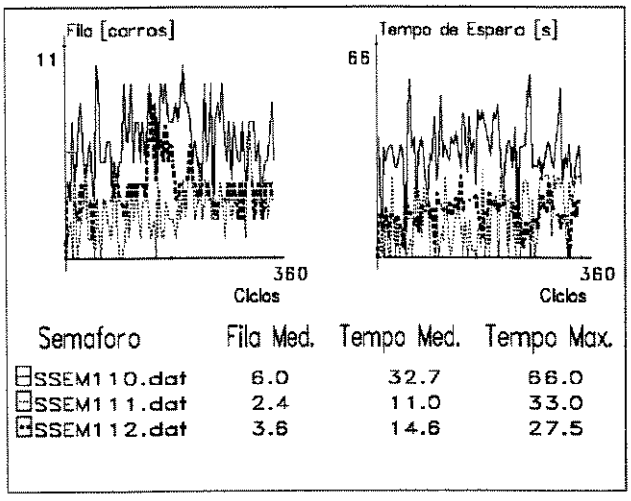
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
FSEM320.dat	19.0	112.7	404.0
FSEM321.dat	3.4	17.2	35.0
FSEM322.dat	3.5	14.9	28.0

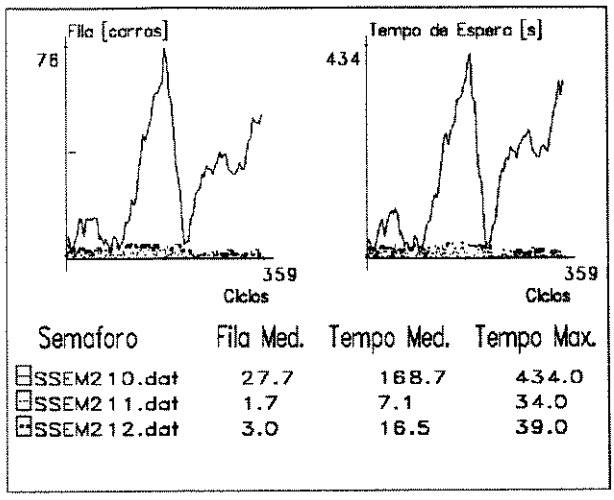
(f)

Figura A.29: Simultâneo: pico de tráfego



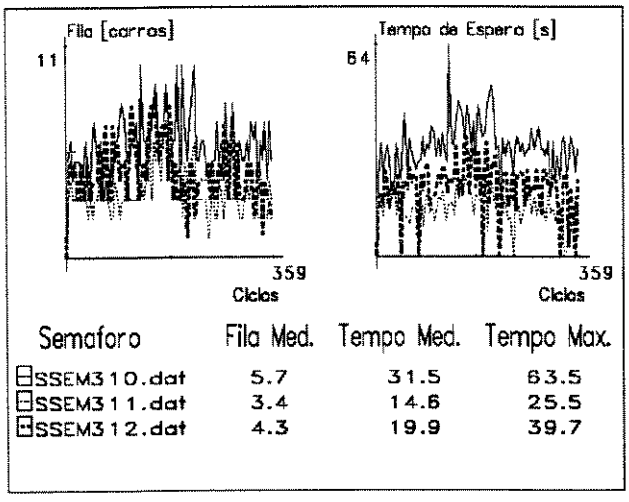
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM110.dat	6.0	32.7	66.0
SSEM111.dat	2.4	11.0	33.0
SSEM112.dat	3.6	14.6	27.5

(a)



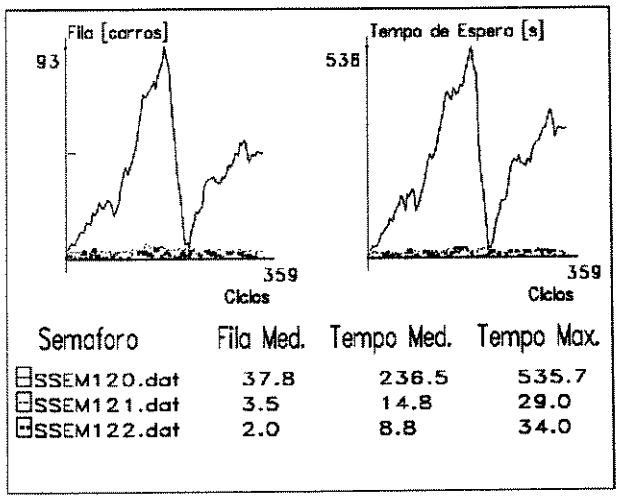
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM210.dat	27.7	168.7	434.0
SSEM211.dat	1.7	7.1	34.0
SSEM212.dat	3.0	16.5	39.0

(b)



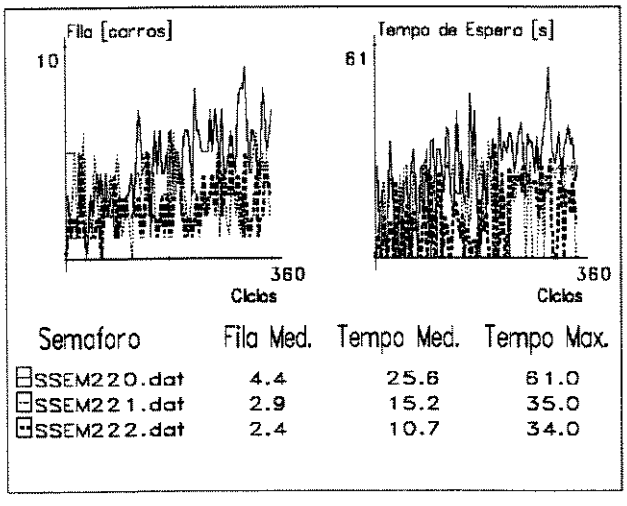
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM310.dat	5.7	31.5	63.5
SSEM311.dat	3.4	14.6	25.5
SSEM312.dat	4.3	19.9	39.7

(c)



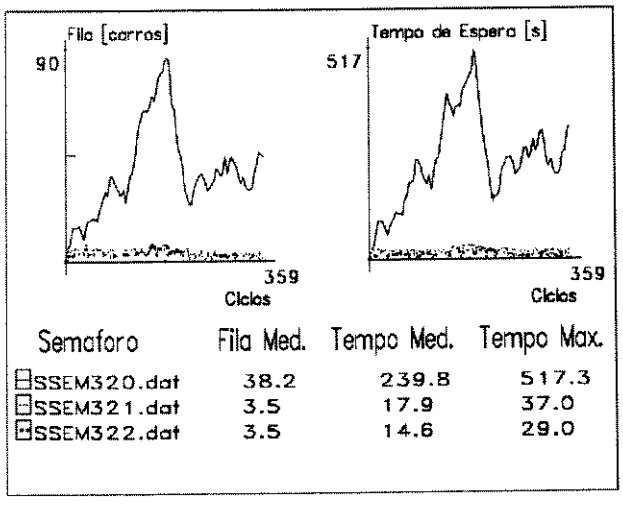
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM120.dat	37.8	236.5	535.7
SSEM121.dat	3.5	14.8	29.0
SSEM122.dat	2.0	8.8	34.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM220.dat	4.4	25.6	61.0
SSEM221.dat	2.9	15.2	35.0
SSEM222.dat	2.4	10.7	34.0

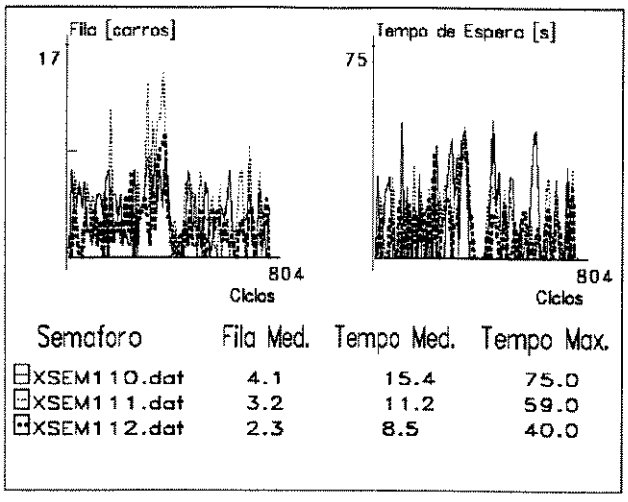
(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
SSEM320.dat	38.2	239.8	517.3
SSEM321.dat	3.5	17.9	37.0
SSEM322.dat	3.5	14.6	29.0

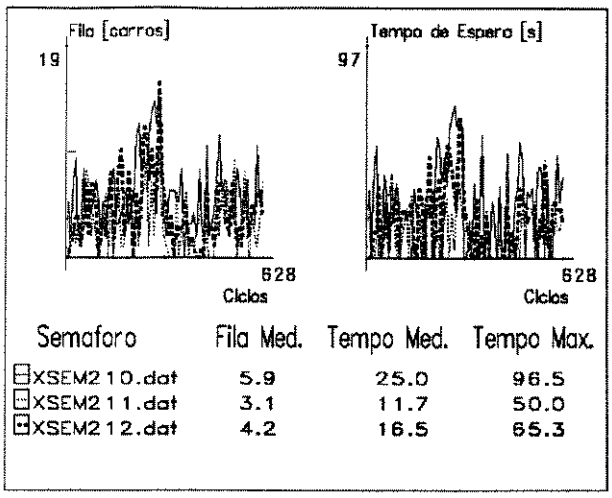
(f)

Figura A.30: Progressivo: pico de tráfego



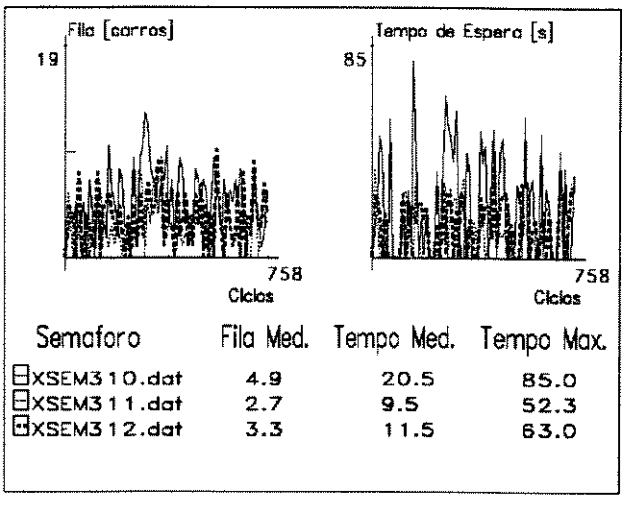
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM110.dat	4.1	15.4	75.0
XSEM111.dat	3.2	11.2	59.0
XSEM112.dat	2.3	8.5	40.0

(a)



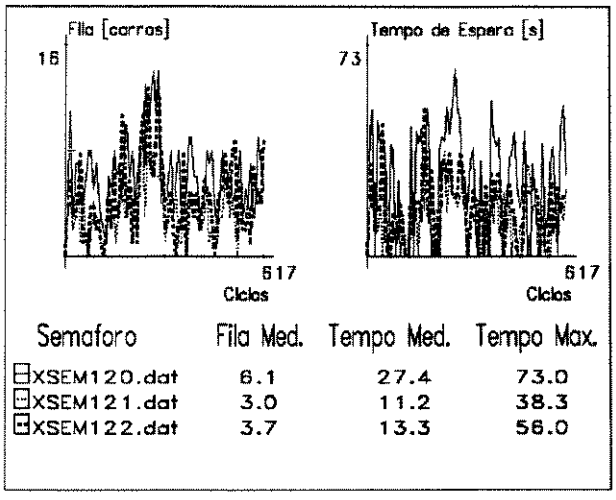
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM210.dat	5.9	25.0	96.5
XSEM211.dat	3.1	11.7	50.0
XSEM212.dat	4.2	16.5	65.3

(b)



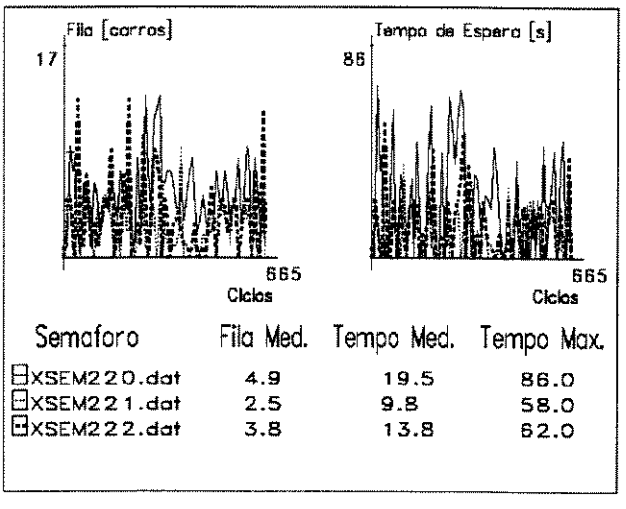
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM310.dat	4.9	20.5	85.0
XSEM311.dat	2.7	9.5	52.3
XSEM312.dat	3.3	11.5	63.0

(c)



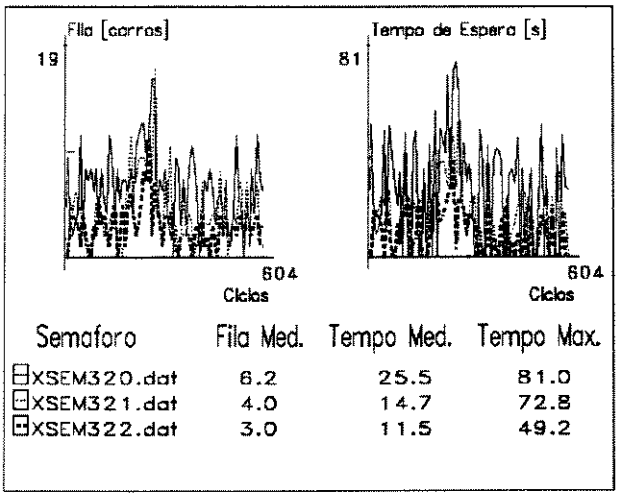
Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM120.dat	6.1	27.4	73.0
XSEM121.dat	3.0	11.2	38.3
XSEM122.dat	3.7	13.3	56.0

(d)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM220.dat	4.9	19.5	86.0
XSEM221.dat	2.5	9.8	58.0
XSEM222.dat	3.8	13.8	62.0

(e)



Semaforo	Fila Med.	Tempo Med.	Tempo Max.
XSEM320.dat	6.2	25.5	81.0
XSEM321.dat	4.0	14.7	72.8
XSEM322.dat	3.0	11.5	49.2

(f)

Figura A.31: ICD: pico de tráfego

Apêndice B

Cálculo dos tempos dos semáforos

A seguir, serão apresentados os dados coletados em cruzamentos selecionados da cidade de Campinas - SP, além dos cálculos para estabelecer os tempos para os ciclos dos semáforos, de acordo com as considerações de [108]. Serão apresentados inicialmente os cálculos teóricos para a determinação do tempo de ciclo, ciclos de verde , atrasos e filas esperadas. Em seguida, serão apresentados cálculos similares que utilizam o tempo de ciclo observado no cruzamento, ajustado por engenheiros de tráfego segundo as condições locais. As medições e os cálculos realizados não tiveram por objetivo traçar um padrão exato das condições de tráfego na cidade, mas apenas prover indicações das grandezas envolvidas. Os valores medidos e os resultados teóricos obtidos foram utilizados para validar os resultados de simulação, permitindo uma comparação entre as estratégias de controle implementadas.

As aferições foram realizadas nos três cruzamentos da avenida Orozimbo Maia, entre as ruas José Paulino e Sacramento. Em cada cruzamento, foram realizadas pelo menos dez contagens para chegadas de veículos em cada sentido do fluxo, e das filas que se formavam em cada ciclo. As contagens foram sempre consecutivas, com duração de

um minuto cada, às quintas-feiras dos meses de maio e junho de 1995, entre as dezoito e as dezenove horas, horário considerado crítico. O cruzamento da avenida Orozimbo Maia e rua Francisco Glicério foi o que apresentou maiores taxas de chegada e maiores filas de veículos, tendo sido assim o cruzamento que balizou os cálculos. Os valores calculados foram bastante semelhantes aos observados nos cruzamentos, à exceção dos tamanhos das filas. Os tamanhos das filas observados foram superiores aos tamanhos teóricos, o que pode ser explicado pelo grande número de motoristas que estacionavam seus veículos e realizavam conversões a partir das faixas direitas, retardando o fluxo de tráfego.

Os valores aferidos e utilizados nos cálculos e comparações são os seguintes:

- **Fluxo de veículos.**

  Rua Francisco Glicério: $DV_1 = 1000 veic./h$
  Avenida Orozimbo Maia (sentido mais crítico): $DV_2 = 1500 veic./h$
  Total do cruzamento: $DV = 2500 veic/h$

- **Tempo de Ciclo:**

  90 s

- **Tempos de Verde:**

  rua Francisco Glicério: 35 s
  avenida Orozimbo Maia: 55 s

- **Filas médias no início da fase de verde:**

  rua Francisco Glicério: 15.6
  avenida Orozimbo Maia: 10.6

- **Diferença de fase entre semáforos consecutivos.**

  avenida Orozimbo Maia: 6 s

Além desses valores, vários outros são necessários para o cálculo das fases em um cruzamento. Muitos deles demandariam equipamentos específicos para que fossem determinados com precisão. Tais valores foram extraídos de sugestões presentes em trabalhos cujo objetivo era o determinar as condições de como o tráfego ocorria em cruzamentos, e estabelecer equações para modelar e estabelecer valores apropriados para esses tempos, como em [108]. Os valores utilizados são os seguintes: [1]

- **Tempo de reação.**

  $t_d = 1$ s

- **Comprimento do veículo-padrão.**

  L=17 ft ($\approx$ 5.18 m)

- **Aceleração.**

  $a_2 = 12\ ft/s^2$ ($\approx 3.66\ m/s^2$)

- Velocidade.

  F. Glicério: $U = 25mph(36.7ft/s)$ ($\approx$ 40 Km/h)

  O. Maia: $U = 30mph(44.0ft/s)$ ($\approx$ 48 Km/h)

- **Largura do cruzamento.**

  F. Glicério: $W = 36ft$ ($\approx$ 11 m)

  O. Maia: $W = 68ft$ ($\approx$ 20.7 m)

A seguir, são apresentados os cálculos para estabelecer os tempos de ciclo, de amarelo, e os atrasos esperados por veículo e filas médias esperadas no início da fase de verde:

[1] A unidade de comprimento é dada em pés (*ft*) para a utilização das equações presentes na literatura.

- **Tempo de Amarelo:**

$$t_a = t_d + \frac{U}{2a_2} + \frac{W+L}{U}$$

Para a fase 1 (rua Francisco Glicério):

$$t_a = 1 + \frac{36.7}{2\text{x}12} + \frac{36+17}{36.7} \approx \boxed{4\text{s}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$t_a = 1 + \frac{44.0}{2\text{x}12} + \frac{68+17}{44.0} \approx 4.8 \approx \boxed{5\text{s}}$$

- **Tempo Perdido:**

$K_i = k_{1_i} + k_{2_i} = 3.7 + \frac{W_i+L_i}{U_i}$

Para a fase 1 (rua Francisco Glicério):

$$K_1 = 3.7 + \frac{36+17}{36.7} \approx \boxed{5.1\text{s}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$K_2 = 3.7 + \frac{68+17}{44.0} \approx \boxed{5.6\text{s}}$$

B.1 Cálculo do ciclo e dos tempos de verde

A seguir, serão apresentados os cálculos para o tempo de ciclo total, bem como dos tempos de verde e os valores esperados para atrasos e filas.

- **Tempo de Ciclo:**

$$y = \sum_{i=1}^{p} \frac{DV_i}{s} = \frac{500}{1714.29} + \frac{500}{1714.29} \approx 0.58 \qquad \text{(B.1)}$$

$$C_o = \frac{1.5k + 5}{1 - y} = \frac{1.5 * (5.1 + 5.6) + 5}{1 - 0.58} \approx \boxed{50s} \tag{B.2}$$

onde

C_o= duração do ciclo [s];

k = tempo perdido em todas as fases [s];

y = soma das razões de fluxo crítico por faixa, pelo grau de saturação.

- Tempo de Verde Efetivo:

$$G_E = C - \sum_{i=1}^{2} K_i = 50 - 5.1 - 5.6 = \boxed{39.3s}$$

O tempo de verde e amarelo para a i-ésima fase é dado por:

$$G_i = \frac{DV_i}{DV}.G_E + K_i$$

Para a fase 1 (rua Francisco Glicério):

$$G_1 = \frac{1000}{2500}.39.3 + 5.1 \approx \boxed{21s}$$

Para a fase 2 (avenida Orozimbo Maia):

$$G_2 = \frac{1500}{2500}.39.3 + 5.6 \approx \boxed{29s}$$

Outros Cálculos :

- **Atraso por Veículo:**

$$d_j = \frac{C(1 - \lambda_i)^2}{2(1 - \lambda_i x_j)} + \frac{x_j^2}{2V_j(1 - x_j)} - 0.65(\frac{C}{V_j^2})^{\frac{1}{3}}.x_j^{2+5\lambda_i}$$

Para a fase 1 (rua Francisco Glicério):

$$\lambda_i = \frac{G_i - K_i}{C} = \frac{21 - 5.1}{50} \approx 0.318$$

$$S = \frac{1}{h} = \frac{1}{2.1} = \approx 0.476$$

$$V_1 = \frac{500}{3600} \approx 0.139$$

$$x_1 = \frac{V_1}{\lambda_i S} \approx \frac{0.139}{0.318\text{x}0.476} \approx 0.918$$

$$d_1 \approx \frac{50(1-0.318)^2}{2(1-0.318\text{x}0.918)} + \frac{0.918^2}{2\text{x}0.139(1-0.918)} - 0.65\left(\frac{50}{0.139^2}\right)^{\frac{1}{3}}.0.918^{2+5\text{X}0.318} \approx \boxed{45.14\text{s}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$\lambda_i = \frac{G_i - K_i}{C} = \frac{29 - 5.6}{50} \approx 0.468$$

$$S = \frac{1}{h} = \frac{1}{2.1} = \approx 0.476$$

$$V_2 = \frac{500}{3600} \approx 0.139$$

$$x_2 = \frac{V_1}{\lambda_i S} \approx \frac{0.139}{0.468\text{x}0.476} \approx 0.624$$

$$d_1 \approx \frac{50(1-0.468)^2}{2(1-0.468\text{x}0.624)} + \frac{0.624^2}{2\text{x}0.139(1-0.624)} - 0.65\left(\frac{50}{0.139^2}\right)^{\frac{1}{3}}.0.624^{2+5\text{X}0.468} \approx \boxed{12.57\text{s}}$$

- **Filas (no início do período de verde):**

$$n_j = \frac{V_j R_i}{2} + V_j d_j$$

ou

$$V_j R_i$$

, o que for maior

Para a fase 1 (rua Francisco Glicério):

$$n_1 \approx \frac{0.139\text{x}29}{2} + 0.139\text{x}45.14 \approx 8.3$$

ou

$$n_1 = 0.139\text{x}29 \approx 4.0$$

Portanto:

$$n_1 \approx \boxed{8.3 \text{ veíc.}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$n_2 \approx \frac{0.139\text{x}21}{2} + 0.139\text{x}12.57 \approx 3.2$$

ou

$$n_2 = 0.139\text{x}21 \approx 2.9$$

Portanto:

$$n_2 \approx \boxed{3.2 \text{ veíc.}}$$

B.2 Cálculos para o ciclo observado

A seguir, serão apresentados os cálculos para os tempos de verde, atrasos e fila levando em conta o tempo de ciclo total observado no cruzamento.

- **Tempo de Ciclo:**

  Ciclo de 90s.

- Tempo de Verde Efetivo:

$$G_E = C - \sum_{i=1}^{2} K_i = 90 - 5.1 - 5.6 = \boxed{79.3\text{s}}$$

  O tempo de verde e amarelo para a i-ésima fase é dado por:

$$G_i = \frac{DV_i}{DV}.G_E + K_i$$

  Para a fase 1 (rua Francisco Glicério):

$$G_1 = \frac{1000}{2500}.79.3 + 5.1 \approx \boxed{37\text{s}}$$

  Para a fase 2 (avenida Orozimbo Maia):

$$G_2 = \frac{1500}{2500}.79.3 + 5.6 \approx \boxed{53\text{s}}$$

Outros Cálculos :

- **Atraso por Veículo:**

$$d_j = \frac{C(1-\lambda_i)^2}{2(1-\lambda_i x_j)} + \frac{x_j^2}{2V_j(1-x_j)} - 0.65(\frac{C}{V_j^2})^{\frac{1}{3}}.x_j^{2+5\lambda_i}$$

Para a fase 1 (rua Francisco Glicério):

$$\lambda_i = \frac{G_i - K_i}{C} = \frac{37 - 5.1}{90} \approx 0.354$$

$$S = \frac{1}{h} = \frac{1}{2.1} = \approx 0.476$$

$$V_1 = \frac{500}{3600} \approx 0.139$$

$$x_1 = \frac{V_1}{\lambda_i S} \approx \frac{0.139}{0.354\text{x}0.476} \approx 0.825$$

$$d_1 \approx \frac{90(1-0.354)^2}{2(1-0.354\text{x}0.825)} + \frac{0.825^2}{2\text{x}0.139(1-0.825)} - 0.65\left(\frac{90}{0.139^2}\right)^{\frac{1}{3}}.0.825^{2+5\text{X}0.354} \approx \boxed{35.26\text{s}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$\lambda_i = \frac{G_i - K_i}{C} = \frac{53 - 5.6}{90} \approx 0.527$$

$$S = \frac{1}{h} = \frac{1}{2.1} = \approx 0.476$$

$$V_2 = \frac{500}{3600} \approx 0.139$$

$$x_2 = \frac{V_1}{\lambda_i S} \approx \frac{0.139}{0.527\text{x}0.476} \approx 0.554$$

$$d_1 \approx \frac{90(1-0.527)^2}{2(1-0.527\text{x}0.554)} + \frac{0.554^2}{2\text{x}0.139(1-0.554)} - 0.65\left(\frac{90}{0.139^2}\right)^{\frac{1}{3}}.0.554^{2+5\text{X}0.527} \approx \boxed{15.99\text{s}}$$

- **Filas (no início do período de verde):**

$$n_j = \frac{V_j R_i}{2} + V_j d_j$$

ou

$$V_j R_i$$

, o que for maior

Para a fase 1 (rua Francisco Glicério):

$$n_1 \approx \frac{0.139\text{x}53}{2} + 0.139\text{x}35.26 \approx 8.6$$

ou

$$n_1 = 0.139\text{x}53 \approx 7.4$$

Portanto:

$$n_1 \approx \boxed{8.6 \text{ veíc.}}$$

Para a fase 2 (avenida Orozimbo Maia):

$$n_2 \approx \frac{0.139\text{x}37}{2} + 0.139\text{x}15.99 \approx 4.8$$

ou

$$n_2 = 0.139\text{x}37 \approx 5.1$$

Portanto:

$$n_2 \approx \boxed{5.1 \text{ veíc.}}$$

Dessa forma, temos:

a) Valores Observados no cruzamento:

Fase	Direção de Movimento	Fase de Vermelho[s]	Fase de verde		Ciclo Total[s]	Fila [veic]	Fluxo [veic/h]	Número de Vias
			Verde[s]	Amarelo[s]				
1	F.Glicério	55	32	3	90	15.6	1000	2
2	O. Maia	35	52	3	90	10.6	1500	3

b) Valores Calculados:

Fase	Direção de Movimento	Fase de Verm.[s]	Fase de verde		Ciclo Total[s]	Tempo de Espera[s]	Fila [veic]	Fluxo [veic/h]	Num. Vias
			Verde[s]	Amar.[s]					
1	F.Glicério	29	17	4	50	45.14	8.3	1000	2
2	O. Maia	21	24	5	50	12.57	3.2	1500	3
1	F.Glicério	53	33	4	90	35.26	8.6	1000	2
2	O. Maia	37	48	5	90	15.99	5.1	1500	3

Apêndice C

Fundamentos da Teoria de Tráfego Urbano

A área de Engenharia de Tráfego, e em particular, de Tráfego Urbano, tem por objetivo a aplicação de princípios científicos, ferramentas, métodos e técnicas para possibilitar o movimento de pessoas e de mercadorias, de maneira rápida, segura, confortável, conveniente e econômica [63, 107, 108]. Para descrever os componentes do tráfego, consideram-se:

- **Composição:** refere-se ao tipo de tráfego e suas características. É constituída basicamente por duas classes de veículos: uma para o movimento de pessoas e outra para o movimento de cargas;
- **Volume:** diz respeito à quantidade de veículos que está se movendo;
- **Origem e Destino:** de onde vem e para onde se dirige o fluxo de tráfego;

- **Qualidade:** como o tráfego se move. Pode incluir aspectos como velocidade de movimento, conveniência, conforto, segurança e privacidade;
- **Custo:** o quão caro é manter uma determinada qualidade de tráfego.

A Teoria de Fluxo de Tráfego leva em consideração três elementos básicos: a composição, o volume e a qualidade. Além desses elementos, algumas definições e relações são necessárias para seu entendimento.

Considera-se, por exemplo, como sendo uma **via**, um conjunto de **faixas** de fluxos, que podem se mover em várias direções (ver Figura C.1).

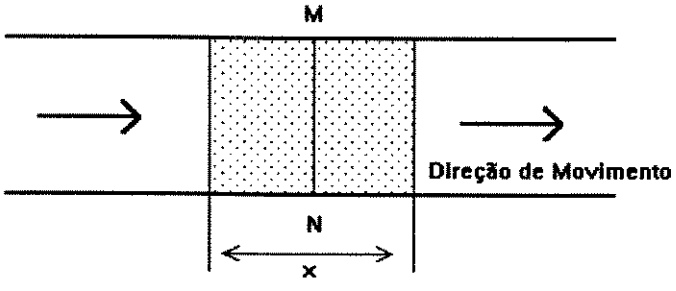


Figura C.1: Exemplo de via de trânsito

Sendo **n** o número de veículos que cruzam o segmento MN durante um intervalo de tempo **T**, pode-se ainda definir:

- **Fluxo** ou **Volume (V):** número de veículos que passam por um dado ponto numa unidade de tempo [veículos/hora]:

$$V = \frac{n}{t} \tag{C.1}$$

- **Densidade** ou **Concentração (D):** numero médio de veículos que trafegam, por unidade de comprimento da via [veículos/Km]:

$$D = \frac{\sum_{i=1}^{n} \frac{t_i}{T}}{x} \tag{C.2}$$

onde:

t_i= tempo para o i-ésimo veículo se mover por uma distância x;

$\sum_{i=1}^{n} \frac{t_i}{T}$= número médio de veículos que passam por x.

- **Espaço Velocidade Média ou Velocidade Média** ($\overline{U}s$:) média das velocidades dos veículos que trafegam por um dado comprimento de via, ponderado de acordo com o tempo gasto para atravessar este comprimento [Km/hora]:

$$\overline{U}s = \frac{x}{\frac{1}{n} \cdot \sum_{i=1}^{n} t_i} \tag{C.3}$$

ou

$$\overline{U}s = \frac{V}{D} \tag{C.4}$$

- **Velocidade Média Livre:** velocidade alcançada quando as vias estiverem vazias, ou sem interferência de outros veículos;

- **Distância de *Headway* ou Espaço (s):** distância entre a frente de um veículo e frente do próximo [m];

- **Tempo de *Headway* (h):** tempo entre a chegada da frente de um veículo e a chegada da frente do próximo [s];

- **Unidade de Tempo de Viagem (m):** tempo de viagem por unidade de distância [s];

- **Capacidade de uma Via:** fluxo máximo de veículos que pode passar através de uma secção da via;

- **Capacidade de um Cruzamento ou Grau de Saturação:** fluxo máximo que pode atravessar um cruzamento, durante uma fase de verde.

Uma vez definidos os conceitos acima, é possível estabelecer algumas relações entre eles. Uma relação bastante importante é a existente entre o fluxo e a densidade de

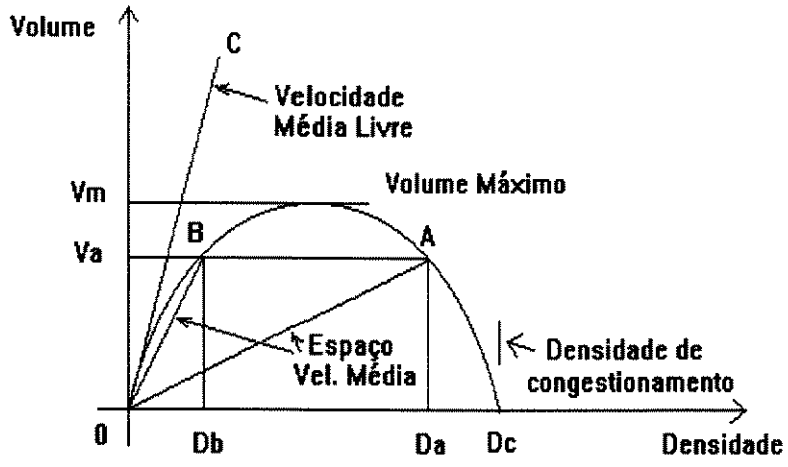


Figura C.2: Diagrama fundamental de tráfego

tráfego. Ela constitui-se no Diagrama Fundamental de Tráfego em Rodovias [36], ilustrado pela Figura C.2.

Este diagrama é importante porque ilustra as possíveis situações de tráfego:

- O volume é zero quando a densidade é zero ou máxima (Dc), o que define a densidade de congestionamento;
- O volume começa a diminuir antes que a densidade seja máxima;
- Sendo a velocidade média a razão entre o volume e a densidade, o segmento $\overline{OA}$ representa o espaço velocidade média para o volume VA e densidade DA. Para o volume VA, existe uma outra velocidade possível, definida pela inclinação do segmento $\overline{OB}$, correspondendo à densidade DB;
- A velocidade média livre é a velocidade que os motoristas assumem quando a rodovia está praticamente livre, ou seja, não há interferência de outros veículos. Ela acontece quando o fluxo e a densidade tendem a zero.

C.1 Tipos de Cruzamentos

Os cruzamentos em vias, expressas ou não, lidam com fluxos de tráfego em direções e sentidos diferentes, levando à espera e acidentes. A operação dos cruzamentos é crucial para a determinação da capacidade e desempenho da malha viária. Cada cruzamento pode apresentar características diferentes quanto ao desenho (*layout*) físico, fluxo de veículos, movimentos de conversão e movimentos de pedestres, dentre outros.

Segundo o tipo de controle, os cruzamentos podem ser classificados em quatro categorias:

- **Cruzamentos não controlados:** quando as vias concorrentes tiverem importância similar, com fluxos baixos de tráfego;
- **Cruzamentos com prioridade:** quando uma das vias tiver sempre prioridade sobre outra. A via menos prioritária é controlada com um sinal, como o de **Pare**. A via principal praticamente não sofre atrasos;
- **Cruzamentos compartilhados por espaço:** quando o objetivo for dar ampla, mas igual prioridade, e permitir movimento contínuo para todos os fluxos. Um exemplo típico são as rotatórias;
- **Cruzamentos compartilhados por tempo:** quando existem vários fluxos importantes entrando em um cruzamento, é atribuída prioridade a cada um dos fluxos, em diferentes instantes de tempo. O controle neste tipo de cruzamento é realizado normalmente por semáforos.

C.2 Cruzamentos controlados por semáforos

Os cruzamentos controlados por semáforos constituem a melhor alternativa de controle dentre as descritas anteriormente, conforme os fluxos de tráfego aumentam. Hoje, é

claramente impensável controlar razoavelmente o trânsito de grandes cidades sem a instalação de semáforos.

Para um bom entendimento dos sistemas controlados por semáforos, é importante destacar:

- **Estado:** constitui-se de uma de uma determinada indicação (cor) para uma determinada via do cruzamento;
- **Fase:** Constitui-se de um conjunto de estados em um determinado instante;
- **Ciclo:** é o tempo necessário para que um semáforo execute uma seqüência completa de indicações, isto é, que todas as vias tenham direito de movimento por pelo menos uma vez.

Considerando um único cruzamento, os semáforos podem ser classificados quanto à sua flexibilidade em termos do seu ciclo e fase, em:

- **Semáforos de Ciclo Fixo:** o ciclo total e a duração de cada fase são pré-ajustados, e permanecem fixos durante períodos do dia ou durante todo o dia;
- **Semáforos Semi-Atuados:** o ciclo total e a duração das fases variam de acordo com as chegadas nas vias menos prioritárias, onde devem existir detectores de veículos;
- **Semáforos Totalmente Atuados:** é utilizado somente em cruzamentos entre duas vias importantes, com alto fluxo de veículos. Detectores são instalados em todas as vias, e a duração do ciclo e da fase são determinados de acordo com o fluxo em cada via.

Além disso, para vias urbanas, é necessário considerar os conjuntos de vias como parte de um sistema único de trânsito. Neste caso, existem ainda as seguintes variações para o controle do tráfego:

- **Sistemas Simultâneos:** todos os semáforos ao longo de uma via exibem a mesma cor ao mesmo tempo. O ciclo é idêntico para todos os seus cruzamentos. Esse tipo de sistema oferece melhores resultados quando as quadras forem curtas, ou quando as vias suportarem altas velocidades;

- **Sistemas Alternados:** um grupo (ou seqüência) de sinais estará vermelho, o próximo grupo, verde, e assim por diante, ao longo de uma via;

- **Sistemas Progressivos:** cruzamentos sucessivos têm o mesmo ciclo, mas as fases de verde e vermelho podem variar, de acordo com as condições locais de tráfego. A temporização de um semáforo em relação ao próximo é feita de forma a permitir um movimento contínuo de veículos ao longo da via (onda verde);

- **Sistemas Controlados por Computador:** os sistemas controlados por computador manipulam grande quantidade de dados, no intuito de prover melhor desempenho aos cruzamentos, através de filas menores e menores atrasos aos veículos. Eventualmente podem tentar integrar porções maiores da malha viária.

C.3 Análise de Desempenho

Para a análise de desempenho de um cruzamento, é necessário levar em conta diversos fatores. Um deles é o tempo necessário para que os veículos entrem no cruzamento após receberem um sinal verde, uma vez que o movimento não é iniciado instantaneamente. Esse estudo foi realizado por [34], e é mostrado na Tabela C.1. Além disso, normalmente utiliza-se uma *unidade padrão de veículos.* Com isso, pode-se aumentar o número real de veículos na análise de desempenho, a fim de compensar atrasos como o de veículos comerciais maiores (que correspondem de 1 a 5 veículos-padrão), e veículos que farão conversão (as conversões à esquerda correspondem a 1.6 veículos-padrão, e as à direita correspondem a 1.4 veículos-padrão, segundo [108]).

Posição do Veículo	Tempo para entrar no cruzamento [s]
1	3.8
2	3.1
3	2.7
4	2.4
5	2.2
6	2.1
..	..
n	2.1

Tabela C.1: Tempos para entrada no cruzamento

Para a análise de desempenho, são ainda considerados:

- **Atraso Médio:** é o tempo médio de espera sofrido por um veículo ao passar pela j-ésima via durante a i-ésima fase, sendo dado por:

$$d_j = \frac{C(1-\lambda_i)^2}{2(1-\lambda_i x_j)} + \frac{x_j^2}{2V_j(1-x_j)} - 0.65\frac{C}{V_j^2}^{\frac{1}{2}} .x_j^{2+5\lambda_i} \qquad \text{(C.5)}$$

onde

d_j= atraso médio por veículo na j-ésima via durante a i-ésima fase [s];

C= duração do ciclo [s];

λ_i= proporção de verde efetivo no ciclo durante a i-ésima fase;

V_j= volume real na j-ésima via, durante a i-ésima fase [veíc./faixa/s];

x_j= grau de saturação da j-ésima via (razão entre o fluxo real e o fluxo máximo que pode passar pelo cruzamento) $(\frac{V_j}{\lambda_i S})$;

S = fluxo de saturação [veíc./faixa/s].

- **Tamanho Médio da Fila:** o tamanho médio da fila no início do estado de verde para a j-ésima faixa, na i-ésima fase, é dada pelo máximo entre:

$$n_j = \frac{V_j R_i}{2} + V_j d_j \tag{C.6}$$

e

$$n_j = V_j R_i \tag{C.7}$$

onde

n_j= número médio de veículos esperando na fila na j-ésima faixa, no início do estado de verde;

V_j= taxa de fluxo na j-ésima faixa [veículos/s];

R_i= tempo da i-ésima fase em vermelho [s];

d_j= atraso médio por veículo na j-ésima faixa [s].

C.4 Controle de Tráfego

A maneira mais simples e mais utilizada atualmente para o controle de semáforos, é a determinação prévia de cada fase e do ciclo total através de dados estatísticos. Estes são os chamados Semáforos de Ciclo Fixo.

Em algumas cidades, existem sistemas mais avançados, que dividem os dias em períodos, e os controlam de acordo com os fluxos em cada período. Neste caso, são feitas estatísticas de fluxo para cada um dos períodos, e estabelecidos os seus respectivos ciclos. Assim, os semáforos são de ciclo fixo, mas variam ao longo do dia, isto é, possuem vários *planos* de ciclos.

Existem equações para a medida de desempenho de semáforos, e também modelos analíticos para calcular o ciclo ótimo que minimiza o atraso total que o cruzamento

causa [107]. Esses modelos utilizam como principal parâmetro de entrada os fluxos de cada uma das vias. O método que otimiza o desempenho dos semáforos consiste nos seguintes passos:

1. determinação do tempo de amarelo:

$$t_a = t_d + \frac{U}{2a_2} + \frac{W + L}{U} \tag{C.8}$$

onde

t_a= tempo de amarelo [s];

t_d= tempo de reação e tomada de decisão do motorista [s];

U = velocidade de aproximação do veículo [m/s];

W = largura do cruzamento [m];

L = comprimento do veículo [m];

a_2= desaceleração do veículo $[m/s^2]$.

2. consideração do desempenho do veículo. Nesta fase, todos os veículos são transformados em veículos-padrão, segundo as convenções apresentadas anteriormente.

3. determinação da proporção do tempo de verde. Antes que a duração do ciclo total seja determinado, cada uma das fases de verde devem ser calculadas. O tempo total de verde disponível para o movimento de veículos é dado pelo ciclo total menos o tempo perdido. Esse tempo perdido deve-se ao tempo gasto para que a fila se coloque em movimento, e ao período de amarelo, enquanto se espera que os veículos deixem o cruzamento. Assim, para cada fase, tem-se:

$$k = k_1 + k_2 \tag{C.9}$$

onde

k = tempo total perdido [s];

k_1= tempo perdido para que a fila se coloque em movimento - aproximadamente 3.7 segundos por fase;

k_2= tempo perdido enquanto o último veículo passa pelo cruzamento. Essa parcela depende da velocidade de aproximação e do comprimento do veículo, além da largura do cruzamento. Normalmente varia de 1 a 2 segundos por fase.

Em geral, o tempo de verde efetivo é dado por:

$$G_E = C - \sum_{i=1}^{p} k_i \tag{C.10}$$

onde

G_E= tempo total de verde efetivo [s];

k_i= tempo total perdido na fase i [s];

p = número de fases do semáforo;

C = ciclo total [s].

O tempo total de verde efetivo para movimentação de veículos pode ser distribuído entre as fases proporcionalmente ao volume projetado de veículos/via no cruzamento. Em geral, considera-se como volume projetado do cruzamento o volume da via mais saturada. No entanto, quando uma fase tem um volume de veículos significativamente maior (por exemplo, com mais vias movimentando-se), consegue-se reduzir o atraso total do cruzamento aumentando-se ligeiramente a proporção de verde em favor dessa fase. A distribuição proporcional de verde efetivo, incluindo o tempo de amarelo pode ser obtido da seguinte forma:

$$G_i = \frac{DV_i}{DV}.G_E + K_i \tag{C.11}$$

onde

G_i= duração dos estados de verde e amarelo durante a i-ésima fase [s];

DV_i= volume projetado durante a i-ésima fase;

DV = somatória dos volumes projetados durante todas as fases.

Para se determinar a fase de verde real, basta subtrair o tempo perdido.

4. a determinação do atraso médio;

5. a determinação do tamanho médio da fila;

Sendo assim, o ciclo ótimo que minimiza o atraso médio em um cruzamento é dado por:

$$C_o = \frac{1.5k + 5}{1 - y} \tag{C.12}$$

onde

C_o= duração do ciclo ótimo [s];

k = tempo perdido em todas as fases [s];

y = soma das razões de fluxo crítico por faixa, pelo grau de saturação, que é dado por:

$$y = \sum_{i=1}^{p} \frac{DV_i}{s} \tag{C.13}$$

Em sistemas mais simples, esses cálculos são feitos uma única vez, determinando um ciclo que é aplicado durante todo o dia, ao longo dos dias. Nos sistemas de vários planos, os cálculos são realizados para cada plano que é acionado durante o dia, podendo também variar conforme a semana e feriados, por exemplo.

A grande vantagem desses sistemas é sua simplicidade, facilidade de implementação e baixo custo. Porém, tornam-se obsoletos à medida em que as condições de tráfego variam, dependendo do dia e hora de ação. Além disso, os semáforos de vários planos exigem revisões periódicas para a sua atualização.

A maior parte da pesquisa em controle de tráfego urbano na área de lógica nebulosa ainda se restringe ao estudo de um cruzamento isolado. Em [86], é descrito um controlador para um cruzamento entre duas ruas de mão única, onde se decide, a cada dez segundos, a extensão do tempo de verde atual. Em [50] é apresentado um sistema nebuloso de controle para um cruzamento não restrito a vias de mão única, mas ainda incapaz de adaptar-se a certas mudanças no ambiente, como mudanças nos fluxos de todas as vias. Em [82], é apresentado um sistema de controle para dois

cruzamentos sucessivos de uma via arterial de mão única. Outros trabalhos apresentam sistemas capazes de adaptar-se a mudanças nos fluxos, considerando um só cruzamento [24, 45, 63, 97].

Bibliografia

[1] Agha G. *Actors: A Model of Concurrent Computation in Distributed Systems.* MIT Press, Cambridge - MA, 1986.

[2] Bal H.E., Steiner J.G., e Tanenbaum A.S. Programming languages for distributed computing systems. *ACM Computing Surveys*, v.21, n.3, p.261–321, 1989.

[3] Barletta R. An introduction to case-based reasoning. *AI Expert*, 1991.

[4] Bezdek J.C. What is computational intelligence? Zurada J.M., Marks II R.J., e Robinson C.J., eds., *Computational Intelligence - Imitating Life*, p.1–12, IEEE Press, 1994.

[5] Bond A.H. e Gasser L. *Readings in Distributed Artificial Intelligence.* Morgan Kaufmann Publ., San Mateo - CA, 1988.

[6] Cammarata S., McArthur D., e Steeb R. Strategies for cooperation in distributed problem solving. Bond A. e Gasser L., eds., *Readings in Distributed Artificial Intelligence*, Morgan Kaufman, 1988.

[7] Chaib-Draa B., Moulin B., Mandiau R., e Millot P. Trends in distributed artificial intelligence. *Artificial Intelligence Review*, v.6, 1992.

[8] Conry S., Meyer R., e Lesser V. Mutistage negotiation in distributed planning. Bond A. e Gasser L., eds., *Readings in Distributed Artificial Intelligence*, p.367–384, Morgan Kaufman, 1988.

[9] Conry S.E., Kuwabara K., Lesser V.R., e Meyer R.A. Mutistage negotiation for distributed constraint satisfaction. *IEEE Transactions on Systems, Man, and Cybernetics*, v.21, 1991.

[10] Cooper M. e Vidal J. Genetic design of fuzzy controllers: the cart and jointed-pole problem. *Anais do III IEEE International Conference on Fuzzy Systems - IEEE World Congress on Computational Intelligence*, p.1332–1337, Orlando - FL, EUA, Junho de 1994.

[11] Corkill D. Hierarquical planning in a distributed environment. *Anais do X National Conference on Artificial Intelligence*, p.168–179, Cambridge, 1979.

[12] Corkill D., Gallagher K., e Murray K. Gbb: a generic blackboard development system. *Anais do National Conference on Artificial Intelligence*, p.1008–1014, Filadelphia - EUA, 1986.

[13] Coulouris G.F. e Dollimore J. *Distributed Systems: Concepts and Design.* Addison-Wesley Publ. Co., 1988.

[14] Courand G. Cooperation via consensus formation. *Anais do Distribited AI Workshop*, 1990. Chapter 10.

[15] Davis R. e King J. An overview of production systems. Elcock E.W. e Michie D., eds., *Machine Intelligence 8*, p.300–332, Wiley & Sons, 1976.

[16] Decker K.S. Distributed problem-solving techniques: a survey. *IEEE Transactions on Systems, Man, and Cybernetics*, v.SMC-17, n.5, p.729–740, 1987.

[17] Durfee E. e Lesser V. Partial global planning: a coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, v.21, 1991.

[18] Durfee E.H. e Lesser V.R. Using partial global plans to coordinate distributed problem solvers. Bond A. e Gasser L., eds., *Readings in Distributed Artificial Intelligence*, Morgan Kaufman, 1988.

[19] Emerson E.A. *Temporal and modal logic.* North-Holland Publ. Co., 1989.

[20] Engelmore R.S. *Blackboard Systems.* Addison-Wesley Publ. Co., 1988.

[21] Erman L.D., Hayes-Roth F., Lesser V.R., e Reddy D.R. The hearsay-ii speach understanding system: integrating knowledge to resolve uncertainty. *Computing Surveys*, v.12, 1980.

[22] Erman L.D., Hayes-Roth F., Lesser V.R., e Reddy D.R. The hearsay-ii speech-understanding system: integrating knowledge to resolve uncertainty. *Blackboard Systems*, p.31–86, Addison-Wesley Publ. Co., 1988.

[23] Erman L.D., London P.E., e Fickas S.F. The design and an example use of hearsay-ii. *Blackboard Systems*, p.281–295, Addison-Wesley Publ. Co., 1988.

[24] Favilla J., Machion A., e Gomide F. Fuzzy traffic control: adaptative strategies. *Anais do II IEEE International Conference on Fuzzy Systems*, p.506–511, S o Francisco - CA, EUA, 1993.

[25] Freitas R., Prado J., Nakamiti G., e Policastro C. Plancel: um planejador para células flexíveis de manufatura. *Anais do X Congresso Brasileiro de Automática / VI Congresso Latino-Americano de Controle Automático*, p.127–132, Rio de Janeiro - RJ, Setembro de 1994.

[26] Freitas R., Prado J., Nakamiti G., Policastro C., e Rillo M. Raciocínio baseado em casos: uma visão geral. *Anais do XVII Congresso Nacional de Informática*, Salvador - BA, 1994.

[27] Gabbay D. *Modal and Logic Programming*, cap.6, p.197–237. Academic Press, 1987.

[28] Galton A. *Temporal Logic and Computer Science: An Overview*, cap.1, p.1–52. Academic Press, 1987.

[29] Georgeff M. Actions, processes and causality - reasoning about actions and plans. *Anais do 1986 Workshop*, p.99–122, Morgan Kaufmann Publ., San Mateo - CA, 1987.

[30] Georgeff M. Communication and interaction in multi-agent planning. Bond A. e Gasser L., eds., *Readings in Distributed Artificial Intelligence*, p.200–204, Morgan Kaufman, 1988.

[31] Goldberg D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publ. Co., 1989.

[32] Gomide F., Gudwin R., e Tanscheit R. Conceitos fundamentais da teoria de conjuntos fuzzy, lógica fuzzy e aplicações. *Sixth International Fuzzy Systems Association World Congress (Tutoriais)*, p.1–38, Unicamp, 1995.

[33] Goscinski A. *Distributed Operating Systems - The Logical Design.* Addison-Wesley Publ. Co., 1992.

[34] Greenshields B., Schapiro D., e Ericksen E. *Traffic Performance at Urban Intersections.* Relatório Técnico, Yale Bureau of Highway Traffic, New Haven - Conn., 1947.

[35] Hadamard J. *The psychology of invention in the mathematical field.* Princeton University Press, 1949.

[36] Haight F. *Mathematical Theories of Traffic-Flows.* Academic Press Inc., 1963.

[37] Hammond K. *Case-Based Planning: Viewing Planning as a Memory Task.* Academic Press Inc., 1989.

[38] Hayes-Roth B. e Hewett M. Bb1: an implementation of the blackboard control architecture. *Blackboard Systems*, p.297–313, Addison-Wesley Publ. Co., 1988.

[39] Hewitt C.E. Open information system semantics for distributed artificial intelligence. *Artificial Intelligence*, v.47, 1991.

[40] Hewitt C.E. Viewing contro structures as patterns of passing messages. *Artificial Intelligence*, v.8, n.3, p.323–364, 1977.

[41] Hewitt C.E. e Inman J. Dai betwixt and between: from "intelligent agents" to open sytems science. *IEEE Transactions on Systems, Man, and Cybernetics*, v.21, n.6, p.1409–1419, 1991.

[42] Hinrichs T. Strategies for adaptation and recovery in a design problem solver. *Anais do Case-Based Reasoning Workshop*, Morgan Kaufmann Publ. Inc., 1989.

[43] Holland J.H. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, 1975.

[44] Holland J.H. e Reitman J.S. Cognitive systems based on adaptative algorithms. Waterman D.A. e Hayes-Roth F., eds., *Pattern Directed Inference Systems*, p.313–329, Academic Press, 1978.

[45] Hoyer R. e Jumar U. Fuzzy control of traffic-lights. *Anais do III IEEE International Conference on Fuzzy Systems - IEEE World Congress on Computational Intelligence*, p.1526–1531, Orlando - FL, EUA, Junho de 1994.

[46] Huhns M.N. *Distributed Artificial Intelligence.* Pitman Morgan Publ., San Mateo - CA, 1987.

[47] Ishibushi H., Nozaki K., Yamamoto N., e Tanaka H. Acquisition of fuzzy classification knowledge using genetic algorithms. *Anais do III IEEE International Conference on Fuzzy Systems - IEEE World Congress on Computational Intelligence*, p.1963–1968, Orlando - FL, EUA, Junho de 1994.

[48] Jones A., Kaufmann A, e Zimmermann H., eds. *Fuzzy Sets Theory and Applications.* D. Reidel Publ. Co., 1986.

[49] Kaufmann A. *Introduction to the Theory of Fuzzy Subsets.* Academic Press, 1975.

[50] Kelsey R., Bisset K., e Jamshidi M. A simulation environment for fuzzy control of traffic systems. *Anais do XII IFAC - World Congress*, p.553–556, Sydney, Australia, 1993.

[51] Kernighan B. e Ritchie D. *The C Programming Language*. Prentice-Hall, 1978.

[52] Klein M. Supporting conflict resolution in cooperative design systems. *IEEE Transactions on Systems, Man, and Cybernetics*, v.21, 1991.

[53] Kolodner J. e Kolodner R. Using experience in clinical problem solving. *IEEE Trans. Syst., Man, and Cybern.*, v.17, n.3, p.420–431, 1987.

[54] Kolodner J.L. Improving human decision making through case-based decision aiding. *AI Magazine*, v.12, n.2, p.52–68, 1991.

[55] Kolodner J.L. An introduction to case-based reasoning. *Artificial Intelligence*, v.6, n.1, p.3–34, 1992.

[56] Kornfeld W. e Hewitt C. The scientific community metaphor. *IEEE Transactions on Systems, Man, and Cybernetics*, v.SMC-11, 1981.

[57] Lampson B.W., Paul M., e Siegert H.J, eds. *Distributed Systems: Architecture and Implementation*. Springer-Verlag, 1983.

[58] Lesser V.R. A retrospective view of fa/c distributed problem solving. *IEEE Transactions on Systems, Man, and Cybernetics*, v.21, n.6, p.1347–1362, 1991.

[59] Lesser V.R. e Corkill D.D. The distributed vehicle monitoring testbed: a tool for investigating distributed problem solving networks. *Blackboard Systems*, p.353–386, Addison-Wesley Publ. Co., 1988.

[60] Lesser V.R. e Corkill D.D. The distributed vehicle monitoring testbed: a tool for investigating problem-solving networks. *AI Magazine*, v.4, n.3, p.15–33, 1983.

[61] Lesser V.R. e Corkill D.D. Functionally accurate, cooperative distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, v.SMC-11, n.1, p.81–99, 1981.

[62] Lesser V.R. e Erman L.D. A retrospective view of the hearsay-ii architecture. *Blackboard Systems*, p.87–122, Addison-Wesley Publ. Co., 1988.

[63] Machion A. *Um Controle Adaptativo Baseado em Lógica Nebulosa para Tráfego Urbano.* Tese de Mestrado, Faculdade de Engenharia Elétrica - FEE/UNICAMP, Campinas - SP, 1993.

[64] Minsky M. The society theory of thinking. *Artificial Intelligence: an MIT Perspective*, p.423–450, MIT Press, Cambridge - MA, 1979.

[65] Nagao M., Matsuyama T., e Mori H. Structural analysis of complex aerial photographs. *Blackboard Systems*, p.219–230, Addison-Wesley Publ. Co., 1988.

[66] Nakamiti G. *Modelo de Utilização do Paradigma Blackboard.* Tese de Mestrado, Faculdade de Engenharia Elétrica - FEE/UNICAMP, Campinas - SP, Outubro de 1991.

[67] Nakamiti G. e Daltrini B. Astuto - uma arquitetura para o desenvolvimento de sistemas blackboard em prolog. *Anais do XVII Conferência Latino-Americana de Informática*, p.607–621, Caracas, Venezuela, Julho de 1991.

[68] Nakamiti G. e Daltrini B. *Especificação e Implementação do Sistema ASTUTO.* Relatório Técnico RT-DCA 018/91, Departamento de Engenharia de Computação e Automação Industrial - DCA/FEE/UNICAMP, Campinas - SP, Junho de 1991.

[69] Nakamiti G., Daltrini B., e Gomide F. Resolução de um problema de planejamento multi-tarefas utilizando inteligência artificial. *Anais do I Workshop Brasileiro em Fenômenos Não Lineares, Caóticos e Estocásticos*, p.106–107, Rio Claro - SP, Novembro de 1993.

[70] Nakamiti G., Freitas R., Prado J., e Gomide F. Fuzzy distributed artificial intelligence systems. *Anais do III IEEE International Conference on Fuzzy Systems - FUZZ'IEEE 94 - IEEE World Congress on Computational Intelligence*, p.462–467, Orlando - FL, EUA, Junho de 1994.

[71] Nakamiti G., Freitas R., Prado J., e Gomide F. Inteligência artificial distribuída nebulosa. *Anais do X Congresso Brasileiro de Automática / VI Congresso Latino-Americano de Controle Automático*, p.245–250, Rio de Janeiro - RJ, Setembro de 1994.

[72] Nakamiti G. e Gomide F. An architecture for distributed problem solving: specification and implementation. 1996. submetido.

[73] Nakamiti G. e Gomide F. An evolutive fuzzy mechanism based on past experiences. *Anais do II European Congress on Intelligent Techniques and Soft Computing - EUFIT 94*, p.1211–1217, Aachen, Alemanha, Setembro de 1994.

[74] Nakamiti G. e Gomide F. Fuzzy sets in distributed traffic control. *Anais do V IEEE International Conference on Fuzzy Systems - FUZZ'IEEE 96*, New Orleans - LA, EUA, Setembro de 1996.

[75] Nakamiti G. e Gomide F. Incorporating fuzzy sets in intelligent distributed systems. *Anais do Biennial Conference of the North American Fuzzy Information Processing Society - NAFIPS 96*, Berkeley - CA, EUA, Junho de 1996.

[76] Nakamiti G. e Gomide F. Intelligent distributed traffic control. 1996. submetido.

[77] Nakamiti G. e Gomide F. *A Pesquisa em Inteligência Artificial Distribuída.* Relatório Técnico RT-DCA 009/92, Departamento de Engenharia de Computação e Automação Industrial - DCA/FEE/UNICAMP, Campinas - SP, Maio de 1992.

[78] Nakamiti G. e Gomide F. *Remembering and Forgetting - An Approach for a Fuzzy Case-Based Mechanism.* Relatório Técnico RT-DCA 001/94, Departamento de Engenharia de Computação e Automação Industrial - DCA/FEE/UNICAMP, Campinas - SP, Janeiro de 1994.

[79] Nakamiti G. e Gomide F. *Sistemas Fuzzy e Aplicações.* Relatório Técnico RT-DCA 019/93, Departamento de Engenharia de Computação e Automação Industrial - DCA/FEE/UNICAMP, Campinas - SP, 1993.

[80] Nakamiti G. e Gomide F. *Utilização de Conhecimento em Sistemas de Banco de Dados.* Relatório Técnico RT-DCA 017/92, Departamento de Engenharia de Computação e Automação Industrial - DCA/FEE/UNICAMP, Campinas - SP, Setembro de 1992.

[81] Nakamiti G., Guilherme I., Prado J., e Freitas R., eds. *I Simpósio Brasileiro de Automação Inteligente - I SBAI*, Rio Claro - SP, Setembro de 1993.

[82] Nakatsuyama M., Nagahashi H., e Nishizura N. Fuzzy logic controller for a traffic junction in the one-way arterial road. *Anais do IX IFAC - World Congress*, p.13–18, Budapeste, Hungria, 1984.

[83] Newell A. e Simon H.A. *Human Problem Solving.* Prentice-Hall, Englewood Cliffs - NJ, 1972.

[84] Nii H.P. e Aiello N. Age (attempt to generalize): a knowledge-based program for building knowledge-based programs. *Blackboard Systems*, p.251–280, Addison-Wesley Publ. Co., 1988.

[85] Nii H.P., Feigenbaun E.A., Anton J.J., e Rockmore A.J. Signal to signal transformation: hasp/siap case study. *Blackboard Systems*, p.135–157, Addison-Wesley Publ. Co., 1988.

[86] Papis C. e Mamdani E. A fuzzy logic controller for a traffic junction. *IEEE Trans. Syst., Man, and Cybern.*, v.7, 1977.

[87] Prado J., Freitas R., Nakamiti G., e Batista G. Inteligência artificial distribuída no controle de células flexíveis de manufatura. *Anais do I Workshop Brasileiro em Fenômenos Não Lineares, Caóticos e Estocásticos*, p.132–133, Rio Claro - SP, Novembro de 1993. Resumo.

[88] Prado J., Freitas R., Nakamiti G., Guilherme I., e Rillo M. Inteligência artificial distribuída em ambientes reativos. *Anais do I Simpósio Brasileiro de Automação Inteligente*, p.327–338, Rio Claro - SP, Setembro de 1993.

[89] Rice J.P. Problems with problem solving in parallel: the poligon system. Windman L.E., Loparo K.A., e Norman N.R., eds., *Artificial Intelligence, Simulation & Modeling*, 1989.

[90] Rosenblatt F. *Principles of Neurodynamics and the Theory of Brain Mechanisms.* Spartan Books, Washington - DC, 1962.

[91] Sakurai M., Kurihara Y., e Karasawa S. Color classification using fuzzy inference and genetic algorithm. *Anais do III IEEE International Conference on Fuzzy Systems - IEEE World Congress on Computational Intelligence*, p.1975–1978, Orlando - FL, EUA, Junho de 1994.

[92] Scarpelli H. *Modelagem, Projeto e Verificação de Bases de Regras Nebulosas Via Teoria de Redes.* Tese de Doutoramento, Faculdade de Engenharia Elétrica - FEE/Unicamp, Campinas - SP, 1993.

[93] Schank R., ed. *Explanation Patterns.* Lawrence Erlbaum Assoc. Inc., 1986.

[94] Selfridge O.G. Pandemonium: a paradigm for learning. *Anais do* Blake D. e Uttley A., eds., *Symposium on Mechanisation of Thought Processes*, Londres, 1959.

[95] Singh M.P., Huhns M.N., e Stephens L.M. Declarative representations of multiagent systems. *IEEE Transactions on Knowledge and Data Engineering*, v.5, n.5, p.721–739, 1993.

[96] Singhal M. e Casavant T.L. Distributed computing systems. *IEEE Computer*, v.24, n.8, p.12–15, 1991.

[97] Skowronski W. e Shaw L. Self-learning fuzzy traffic controller for a traffic junction. *Anais do I European Congress on Intelligent Techniques and Soft Computing - EUFIT 93*, p.751–761, Aachen, Alemanha, 1993.

[98] Slade S. Case-based reasoning: a research paradigm. *AI Magazine*, v.12, n.1, p.42–55, 1991.

[99] Smith R.G. The contract net protocol: high level communication and control in distributed problem solver. *IEEE Transactions on Computers*, v.C-29, n.12, p.1104–1113, 1980.

[100] Smith R.G. e Davis R. Frameworks for cooperation in distributed problem solving. *IEEE Trans. on Syst., Man, Cybern.*, v.SMC-11, 1981.

[101] Sycara K. Argumentation: planning other agent's plans. *Anais do XI International Joint Conference on Artificial Intelligence*, p.517–523, 1989.

[102] Sycara K. Resolving goal conflicts via negociation. *Anais do National Conference on Artificial Intelligence*, p.245–250, 1988.

[103] Tanenbaum A.S. *Modern Operating Systems.* Prentice-Hall Inc., 1992.

[104] Terry A. *The CHRYSALIS Project: Hierarquical Control of Production Systems.* Relatório Técnico HPP-83-19, Stanford University, 1983. Heuristic Programming Project.

[105] Traiger I.L., Gray J., Galtieri C.A., e Lindsay B.G. Transactions and consistency in distributed database systems. *ACM Transactions on Database Systems*, v.7, n.3, p.323–342, 1982.

[106] Watada J. e Yabuuchi Y. Fuzzy robust regression analysis. *Anais do III IEEE International Conference on Fuzzy Systems - IEEE World Congress on Computational Intelligence*, p.1370–1376, Orlando - FL, EUA, Junho de 1994.

[107] Webster F.V. *Traffic Signal Settings.* Relatório Técnico 39, Road Research Laboratory, Londres, 1958.

[108] Wohl M. e Martin B.V. *Traffic System Analysis for Engineers and Planners.* McGraw-Hill Book Co., 1967.

[109] Yager R.R. Connectives and quantifiers in fuzzy sets. *Fuzzy Sets and Systems*, v.40, 1991.

[110] Yager R.R. Deductive approximate reasoning systems. *IEEE Transactions on Knowledge and Data Engineering*, v.3, n.4, p.399–414, 1991.

[111] Zadeh L.A. Commonsense and fuzzy logic. *The Knowledge Frontier - Essays in The Representation Knowledge*, p.103–136, Springer-Verlag, 1983.

[112] Zadeh L.A. Fuzzy algorithms. *Information and Control*, v.12, 1968.

[113] Zadeh L.A. Fuzzy logic. *IEEE Computer*, 1988.

[114] Zadeh L.A. Fuzzy sets. *Information and Control*, v.8, 1965.

[115] Zadeh L.A. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, v.1, 1978.

[116] Zadeh L.A. Knowledge representation in fuzzy logic. *IEEE Transactions on Knowledge and Data Engineering*, v.1, n.1, p.89–100, 1989.

[117] Zadeh L.A. Soft computing and fuzzy logic. *IEEE Software*, Novembro de 1994.

[118] Zadeh L.A. A theory of approximate reasoning. *Machine Intelligence*, v.9, 1979.

[119] Zanconato R. Blobs - an object-oriented blackboard system framework for reasoning in time. *Blackboard Systems*, p.335–345, Addison-Wesley Publ. Co., 1988.

[120] Zurada J.M., Marks II R.J., e Robinson C.J. Introduction. Zurada J.M., Marks II R.J., e Robinson C.J., eds., *Computational Intelligence - Imitating Life*, p.1–12, IEEE Press, 1994.