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

RAFAEL PINTO COSTA

Análise do Atraso Fim-a-Fim Observado por Diferentes Classes de Tráfego em Redes *Ethernet* Micro-Segmentadas

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. João Cesar Netto
Orientador

Porto Alegre, abril de 2004

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Costa, Rafael Pinto

Análise do Atraso Fim-a-Fim Observado por Diferentes Classes de Tráfego em Redes *Ethernet* Micro-Segmentadas / Rafael Pinto Costa. – Porto Alegre: PPGC da UFRGS, 2004.

95 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2004. Orientador: João Cesar Netto.

1. Switched ethernet. 2. Tempo real. 3. Requisitos temporais. 4. Automação. 5. 802.1D/Q/p. I. Netto, João Cesar. II. Título.

“Aprender é a única coisa de que a mente nunca se cansa,
nunca tem medo e nunca se arrepende.”
— LEONARDO DA VINCI

AGRADECIMENTOS

Durante o desenvolvimento desta pesquisa recebi ajuda de muitas pessoas: professores, amigos, colegas e familiares. A todas estas, devo pelo menos um sincero muito obrigado.

Aos professores, agradeço os preciosos conhecimentos repassados que serviram de base para a conclusão deste trabalho. Em especial a meu orientador, pelas inúmeras contribuições, compreensão e definição dos rumos da pesquisa.

Aos colegas e amigos, pelo interesse, disponibilidade e auxílio. Um abraço especial a meus colegas de trabalho, tanto de Porto Alegre quanto de Brasília, por me incentivarem a concluir esta jornada. Mesmo correndo o risco de cometer um erro grave por esquecimento, não posso deixar de citar alguns: Luis Otávio, Doris, Mello, Léo, Quintiliano, Norma e Mendes.

A minha família, pelas palavras e gestos de incentivo e apoio. Meus pais e meus irmãos saibam que suas contribuições psicológicas e práticas foram decisivas. Que sorte eu tive por já chegar ao mundo rodeado de pessoas como vocês. Também à minha sogra, pela energia positiva.

A minha esposa Raquel, sempre ao meu lado auxiliando de todas maneiras possíveis e acumulando os papéis de amiga, colega, professora e acima de tudo, amada.

Em resumo, obrigado Deus.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
ATM	Asynchronous Transfer Mode
CBR	Constant Bit Rate
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
FCS	Frame Check Sequence
FDDI	Fiber Distributed Data Interface
FIFO	First In First Out
ICIR	ICSI Center for Internet Research
ICSI	International Computer Science Institute
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
NCS	Network Control System
OSI	Open Systems Interconnection
OTcl	Object Tcl (MIT)
QoS	Quality of Service
RED	Random Early Detection
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VLAN	Virtual Bridged Local Area Networks
VID	VLAN Identifier

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A tecnologia de rede dominante em ambientes locais é *ethernet*. Sua ampla utilização e o aumento crescente nos requisitos impostos pelas aplicações são as principais razões para a evolução constante presenciada desde o surgimento deste tipo de rede simples e barata. Na busca por maior desempenho, configurações baseadas na disputa por um meio de transmissão compartilhado foram substituídas por topologias organizadas ao redor de *switches* e micro-segmentadas, ou seja, com canais de transmissão individuais para cada estação. Neste mesmo sentido, destacam-se os dispositivos de qualidade de serviço padronizados a partir de 1998, que permitem a diferenciação de tráfego prioritário.

O uso desta tecnologia em ambientes de automação sempre foi refreado devido à imprevisibilidade do protocolo CSMA/CD. Entretanto, o uso de *switched ethernet* em conjunto com a priorização de tráfego representa um cenário bastante promissor para estas aplicações, pois ataca as duas fontes principais de indeterminismo: colisões e enfileiramento. Este trabalho procura avaliar esta estrutura em ambientes de controle distribuído, através de uma análise temporal determinística. Como resultado, propõe-se um modelo analítico de cálculo capaz de prever os valores máximos, médios e mínimos do atraso fim-a-fim observado nas comunicações de diferentes classes de tráfego. Durante a análise, outras métricas relevantes são investigadas como a variação no atraso (*jitter*) e a interferência entre classes de prioridades distintas.

Palavras-chave: Switched ethernet, tempo real, requisitos temporais, automação, 802.1D/Q/p.

End-to-End Delay Analysis for Different Traffic Classes on Switched Ethernets

ABSTRACT

Ethernet has been the main network technology in local environments. Its wide use and the growing increase of application requirements are the main reasons for the constant evolution witnessed since the advent of such a simple and cheap network. In search for better performance, shared media configurations were replaced by switched and micro-segmented topologies, i.e., with individual transmission channels for each station. Likewise, the quality of service mechanisms, which reached a standard in 1998, allows for priority traffic differentiation.

The use of such technology in automation systems has always been contained, due to unpredictability of the CSMA/CD protocol. However, the use of switched Ethernet, combined with traffic prioritization represents a quite promising scenario for these applications, for they tackle the two main sources of non-determinism: collisions and queuing. The present work aims at evaluating that structure in distributed control systems through a deterministic time analysis. As a result, an analytic model calculus is proposed to predict the maximum, average and minimum values of end-to-end delay observed in different traffic communication classes. During the analysis, other relevant metrics are investigated such as jitter (delay variation) and interference among classes of different priorities.

Keywords: switched ethernet, real-time, temporal requirements, automation, 802.1D/Q/p.

1 INTRODUÇÃO

A tecnologia de tempo real tem se destacado notadamente nas últimas décadas por sua relação com uma gama bastante ampla de aplicações importantes para a vida moderna, incluindo sistemas de controle de processos industriais, automação de modo geral, multimídia e realidade virtual, medicina, além de tantas outras. Em particular, quase todos sistemas críticos e a maioria dos sistemas de computação embarcada são sistemas de tempo real (STANKOVIC et al., 1996).

A tônica destes ambientes é sua forte dependência em relação à variável temporal. A característica básica das aplicações desta natureza é que o funcionamento correto do sistema depende não apenas de resultados lógicos, mas também do tempo no qual estes resultados são produzidos.

Anos de desenvolvimento prático e de pesquisa demonstram que o fornecimento de garantias para o atendimento aos requisitos temporais que emergem destes cenários não é uma tarefa trivial (STANKOVIC, 1988). Sob a ótica da computação, o projeto destes sistemas, especialmente os críticos, requer que os conceitos previsibilidade e determinismo temporal sejam considerados em todas camadas da arquitetura empregada.

Qualquer ponto nesta estrutura que não apresente um projeto adequado de atendimento aos requisitos temporais, sejam aplicações finais, protocolos, serviços, dispositivos em hardware ou qualquer outro componente envolvido pode tornar inviável a construção de sistemas específicos, ou pior ainda, provocar a falha de sistemas projetados sobre premissas equivocadas. Problemas potenciais geralmente residem na disputa e alocação de recursos (memória, disco, rede, processador, ...), dada a variabilidade e o não determinismo possivelmente observados nos tempos de contenção inseridos. Este fato ressalta a importância na distribuição dos recursos e, conseqüentemente, dos algoritmos de escalonamento adotados.

Preocupações desta natureza justificam o desenvolvimento de soluções específicas para a área de computação em tempo real, como por exemplo, o caso de sistemas operacionais dedicados (e.g. MicroC/OS II) ou de extensões de linguagens de programação de alto nível (e.g. Java-RT) (BOLLELLA et al., 2000).

As redes de comunicação são figuras extremamente relevantes neste contexto, pois muitos dos sistemas envolvidos englobam a comunicação entre entidades espacialmente separadas no ambiente local. Este é o caso geral na área de automação, industrial ou predial/residencial por exemplo, com a utilização de sistemas de controle em rede (NCS) (WALSH; HONG, 2001; CHOW; TIPSUWAN, 2001) para coordenação de sensores e atuadores.

Algumas das métricas de desempenho que impactam diretamente na aplicação de tecnologias de rede em sistemas de controle incluem: tempo de acesso, tempo de transmissão, tempo de resposta, tempo de mensagem, percentual de colisões, vazão e tamanho

das mensagens, percentual de utilização da rede e determinismo temporal. A atenção a estas métricas pode ser resumida em duas características fundamentais: atraso temporal limitado para as comunicações e transmissão sem perdas (LIAN; MOYNE; TILBURY, 2001). Isto significa que as mensagens devem ser corretamente transmitidas dentro de um limite temporal previsível. Transmissões sem sucesso ou com tempo muito elevado entre sensores e atuadores podem, por exemplo, degradar o desempenho de um sistema ou levá-lo à instabilidade.

Neste sentido, diversas tecnologias dedicadas de rede foram desenvolvidas com o objetivo de atender a estas exigências, basicamente através do suporte à troca freqüente de pequenas quantidades de dados (THOMESSE, 1999). Genericamente conhecidas como *fieldbuses*, os exemplos mais conhecidos são: TTP/C, CAN, DeviceNet, ProfiBus, WorldFip, P-Net e Foundation FieldBus (IEC, 2003; ISO, 1992).

Embora adequadas em muitos casos, estas tecnologias apresentam algumas desvantagens como: alto custo, dificuldade de integração com outras tecnologias de rede, incompatibilidades entre dispositivos de fabricantes distintos e escassez na banda disponível, tipicamente entre 1 e 5 Mbps (DECOTIGNIE, 2001).

Devido a estas limitações, alguns protocolos de uso geral passaram a ser cogitados como alternativas para aplicação em sistemas com características temporais, como FDDI e ATM (SONG, 2001) e, principalmente, a tecnologia mais utilizada no âmbito local: redes tipo *ethernet* (SHIMOKAWA; SHIOBARA, 1985; COURT, 1992; CHIUEH; VENKATRAMANI, 1994; KWEON; SHIN; WORKMAN, 2000; BELLO et al., 2000; BELLO; MIRABELLA, 2001; STEFFEN; ZELLER; KNORR, 2003).

Esta tendência é fortemente presenciada no controle de processos industriais, com o emprego massivo de redes *ethernet* observado nos últimos anos (KAPLAN, 2001; CARO, 2001). Como argumentos favoráveis à utilização de redes *ethernet* em aplicações de controle, Decotignie apresenta em (DECOTIGNIE, 2001):

- Baixo custo;
- Fácil integração com a Internet e com protocolos de níveis superiores;
- Evolução constante na taxa de transmissão disponível;
- Facilidade de manutenção e de testes;
- Estabilidade da tecnologia.

Entretanto, inicialmente baseadas na filosofia de compartilhamento e disputa do meio de comunicação, a dificuldade histórica para o emprego de redes *ethernet* em ambientes distribuídos de automação é a imprevisibilidade característica de seu mecanismo de arbitragem (colisões) - protocolo CSMA/CD (WHEELIS, 1993; KHANNA; SINGH, 1994).

Por esta razão, muitas soluções propostas são fundamentadas no controle de tráfego, na inserção de camadas superiores de abstração, em mecanismos de passagem de *tokens* ou na alteração direta do protocolo CSMA/CD. Pedreiras, em (PEDREIRAS; ALMEIDA, 2003), identifica os principais trabalhos na área, apresentando uma visão geral da pesquisa na comunicação tempo real em redes ethernet.

Nesta mesma direção, a alternativa mais adotada e que tem atraído o interesse da pesquisa científica é a utilização de redes *ethernet* totalmente comutadas (*switched ethernet*). Seu surgimento é resultado natural do aumento na demanda por desempenho (aplicações)

aliado aos baixos custos envolvidos, e ao aproveitamento, em muitos casos, da infra-estrutura existente. Hoje, topologias totalmente construídas ao redor de *switches* com apenas uma estação conectada a cada porta são realidade. Cada nó possui um canal exclusivo para transmissão e para recepção (full-duplex), formando uma série de conexões ponto-a-ponto entre estações e *switches*. Neste cenário, a possibilidade de colisões é abolida e a perda de pacotes se dá principalmente pelo enfileiramento nas interfaces dos comutadores.

Embora a ausência intrínseca de colisões, característica de redes micro-segmentadas e totalmente comutadas, represente o ataque ao principal ponto de indeterminismo temporal, não garante por si só o atendimento a todos requisitos de sistemas tempo dependentes. Por exemplo, a possibilidade de enfileiramento pode agregar ainda grande variabilidade ao tempo gasto nas comunicações. Entre os trabalhos desenvolvidos na análise e na adequação desta estrutura estão: (LEE; LEE, 2002), (HOANG et al., 2002), (HOANG; JONSSON, 2003), (JASPERNEITE; NEUMANN, 2001), (JASPERNEITE et al., 2002), (MOLDOVANSKY, 2002) e (SONG, 2001).

O presente trabalho se enquadra neste contexto, sendo seu objetivo principal a avaliação de um modelo que contemple soluções para os dois problemas básicos da tecnologia *ethernet*:

- ausência de colisões com o uso de *switched ethernet*; e
- previsibilidade no enfileiramento a partir do emprego dos mecanismos de qualidade de serviço previstos na padronização IEEE de redes locais (IEEE 802.1D/Q). As bases destes mecanismos são a diferenciação de tráfego prioritário e o escalonamento de prioridade absoluta entre filas distintas.

Como aplicação de estudo considera-se um sistema distribuído que agrega tráfego de controle (NCS), caracterizado por transmissões cíclicas, e tráfego sem requisitos temporais (rede de dados).

Baseado neste cenário é elaborada uma análise determinística, fundamentada na previsão do número de quadros prioritários presentes a cada período na rede, sobre o atraso fim-a-fim observado entre as entidades de controle. O modelo analítico proposto permite a obtenção do perfil temporal das comunicações nas diferentes classes de quadros, através da previsão de valores mínimos, médios e máximos.

1.1 Organização do Texto

O capítulo 2 apresenta uma síntese dos mecanismos de qualidade de serviço disponíveis na camada de enlace, fundamentais para a análise desenvolvida. O capítulo 3 é a base do trabalho e apresenta a dedução do modelo analítico de cálculo, ilustrando sua aplicação em dois casos de estudo específicos. No capítulo 4 apresenta-se a técnica de validação do modelo utilizada - simulação. Os resultados dos experimentos efetuados são expostos no capítulo 5. Por último, o capítulo 6 exibe as conclusões e considerações finais a respeito da análise desenvolvida.

Nos dois anexos seguem os *scripts* de simulação utilizados para modelagem dos casos de aplicação do modelo, empregados para confronto entre resultados teóricos e experimentais.

2 MECANISMOS DE NÍVEL DE ENLACE DISPONÍVEIS - NORMA IEEE 802.1D/Q

Este capítulo tem o objetivo de apresentar os mecanismos de qualidade de serviço previstos nas normas IEEE 802.1D e Q (IEEE, 1998, 2003). Estes recursos são baseados na diferenciação de tráfego, e foram projetados com o intuito de dotar o ambiente local do suporte necessário a integração com os sistemas de QoS desenvolvidos nos níveis superiores.

Ênfase é dada à tecnologia *ethernet* (IEEE 802.3 (IEEE, 2002))[1], apresentando a arquitetura de redes locais virtuais (VLANs) que, neste caso, atua como base para classificação e priorização dos quadros.

2.1 Família de Protocolos IEEE 802

A preocupação com a transmissão de informações críticas em relação a variável temporal foi introduzida na família de protocolos para redes locais IEEE (Figura 2.1) através da especificação 802.1p, incorporada na edição de 1998 da norma IEEE Std 802.1D (IEEE, 1998). Este documento foi adotado pela ISO (*International Organization for Standardization*) sob a identificação ISO/IEC 15802-3:1998.

Entre outras especificações, a norma 802.1D apresentou a idéia de classificação do tráfego e definiu, de modo independente da tecnologia de nível físico empregada, o comportamento esperado das entidades intermediárias [2]. Embora baseada na diferenciação por sinalização explícita, não apresentou novos formatos para os quadros de tecnologias que não possuíssem a capacidade de carregar informações de prioridade, como por exemplo, *ethernet*.

O complemento necessário veio também em 1998 através do padrão IEEE 802.1Q (IEEE, 2003) que estendeu o conceito dos comutadores, adicionando funcionalidades para o suporte e gerenciamento de redes locais virtuais (VLANs). Entre as definições apresentadas estava um novo formato de quadro capaz de carregar informações de identificação de VLAN e de prioridade de usuário. No caso de redes tipo *ethernet*, estas informações foram incluídas em um novo campo de cabeçalho inserido imediatamente após os campos de endereços origem e destino do quadro original.

A figura 2.1 apresenta o relacionamento entre os diversos protocolos que compõem a

[1]Embora originalmente a denominação *Ethernet* tenha sido escolhida para designar o protocolo desenvolvido pela Xerox em 1973, durante o texto e em acordo com sua ampla utilização, o termo será empregado em referência ao padrão IEEE 802.3.

[2]entidades de nível 2 capazes de interligar segmentos distintos de rede, repassando adequadamente os quadros. Como o interesse de estudo recai sobre redes comutadas, os termos *switches* ou comutadores são utilizados como sinônimos ao longo do texto.

Fonte: IEEE 802.1Q - pág. III.

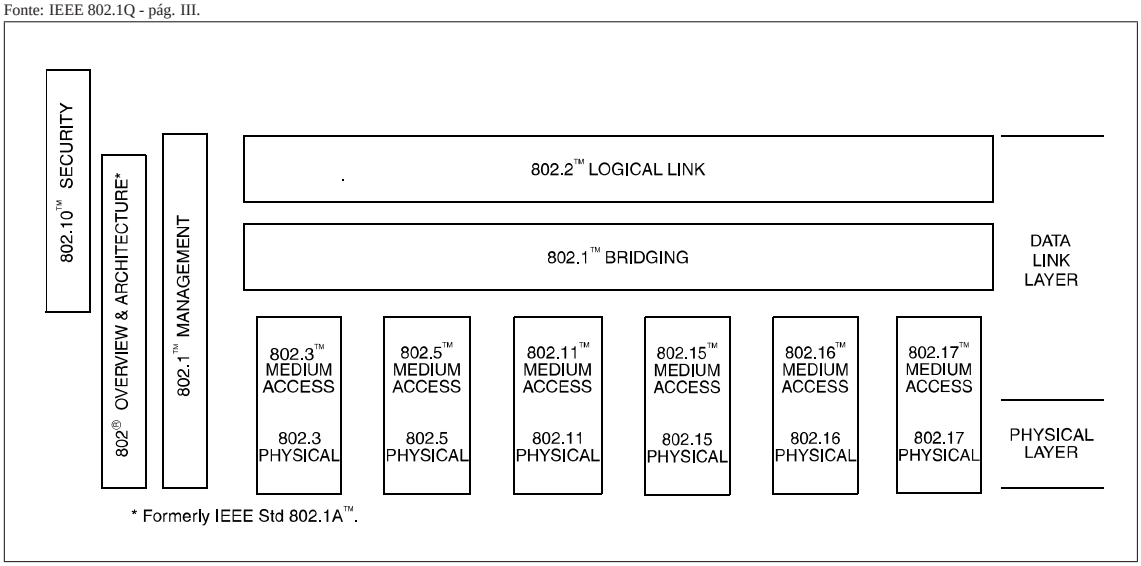


Figura 2.1: Família de protocolos IEEE

família IEEE de padronização em redes locais. Os padrões listados lidam com os níveis físico e de enlace definidos na arquitetura introduzida pelo modelo básico de referência OSI (*Open Systems Interconnection Basic Reference Model* (ZIMMERMANN, 1980)) da Organização Internacional de Padronização (ISO). São observados vários padrões desenvolvidos para o acesso a diferentes tecnologias de camada física. A tabela 2.1 identifica os documentos relacionados aos principais protocolos da família IEEE.

Tabela 2.1: Protocolos da família IEEE 802

Documento	**Título / Descrição**
IEEE Std 802^{TM}	*Overview and Architecture*. Visão geral dos protocolos da família de padrões IEEE 802.
IEEE Std $802.1D^{TM}$ [ISO/IEC 15802-3]	*Media Access Control (MAC) Bridges*. Arquitetura de interconexão de LANs IEEE 802.
IEEE Std $802.1Q^{TM}$	*Virtual Bridged Local Area Networks*. Arquitetura de VLANs.
IEEE Std 802.3^{TM} [ISO/IEC 8802-3]	*CSMA/CD Access Method and Physical Layer Specifications*. Tecnologia de rede local tipo *ethernet*.
IEEE Std 802.5^{TM} [ISO/IEC 8802-5]	*Token Ring Access Method and Physical Layer Specifications*. Tecnologia de rede local tipo anel.
IEEE Std 802.11^{TM} [ISO/IEC 8802-11]	*Wireless LAN Medium Access Control (MAC) Sublayer and Physical Layer Specifications*. Tecnologia de rede local sem fio.
IEEE Std 802.15^{TM}	*Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Network*.

2.2 Diretivas de Qualidade de Serviço

Segundo as definições presentes nas normas IEEE, alguns parâmetros são essenciais a provisão de QoS e devem ser considerados.

- **Disponibilidade de Serviço**: medida como a fração do tempo total em que o serviço MAC está disponível. A orientação presente na norma define que as entidades de nível 2 devem ser dotadas de funções de reconfiguração automática da rede, minimizando o tempo de indisponibilidade de serviços quando, por exemplo, ocorrem falhas físicas nos equipamentos ou conexões.
- **Perda**: não está previsto um serviço de entrega garantido no nível 2, entretanto a probabilidade de perda é muito baixa. As únicas situações que possibilitam o descarte de quadros ocorrem quando:
  - Determinado quadro não pôde ser transmitido dentro do limite máximo de tempo definido para transmissão de quadros. Este limite é estabelecido para garantir o funcionamento dos níveis superiores da arquitetura de rede;
  - A capacidade máxima de armazenamento das filas é atingida;
  - O tamanho do quadro excede o tamanho máximo permitido para o segmento ao qual está destinado;
  - Alterações na topologia provocam a reconfiguração automática da rede e o descarte pode ser adotado como medida para assegurar a rápida estabilização do processo;
  - O repasse do quadro é impedido por regras de filtragem configuradas nos comutadores.
- **Entrega desordenada**: dado um par emissor e receptor, não existe a possibilidade de que quadros pertencentes a mesma classe de serviço sejam entregues fora de ordem.
- **Duplicação**: os serviços MAC não admitem a duplicação de quadros.
- **Atraso dos quadros**: tempo compreendido entre o pedido de transmissão de um quadro e sua chegada ao nó destino. Além do tempo gasto com enfileiramento, injeção e propagação dos quadros nos canais de transmissão, os equipamentos intermediários são responsáveis pela adição de uma pequena parcela do atraso global nas comunicações. Esta parcela diz respeito ao tempo de processamento gasto, por exemplo, com o cálculo de integridade dos quadros (FCS) efetuado a cada comutador ao longo do caminho.
- **Tempo de vida do quadro**: é assegurada a existência de um limite máximo para o atraso global observado por determinado quadro. Isto se deve ao fato de existir um tempo máximo de permanência dos quadros em cada comutador. Este tempo é definido em uma grandeza de segundos e, quando ultrapassado, provoca o descarte do quadro.
- **Taxa de erros não detectados**: como a integridade dos quadros é avaliada na estação destino e a cada ponto intermediário, a taxa de erros não percebidos é extremamente baixa.

- **Prioridade**: o atraso causado pelo enfileiramento de quadros nos comutadores pode ser gerenciado através da associação de prioridades e do conseqüente tratamento diferenciado dispensado a classes distintas de quadros.

2.3 Redes Locais Virtuais (VLANs - Virtual Bridged Local Area Networks)

Com o passar do tempo a utilização e, conseqüentemente, os requisitos impostos sobre as redes locais aumentaram de forma extremamente rápida e intensa. Uma das causas determinantes foi o aumento progressivo nas necessidades do nível de aplicação, incluindo exigências temporais que se acentuaram, por exemplo, com a difusão de sistemas multimídia. Durante este processo, algumas limitações características do ambiente local passaram a ser relevantes na medida em que se tornaram entraves nesta busca pelo maior desempenho possível.

Em redes *ethernet*, um dos problemas principais está relacionado à disputa por um meio compartilhado de transmissão e à possibilidade de colisões – bases de funcionamento do protocolo CSMA/CD. Em redes com alta carga de tráfego, este mecanismo constitui um gargalo para as comunicações, uma vez que acima de determinado patamar, a taxa efetiva de transmissão apenas diminui enquanto se aumenta a carga oferecida (fenômeno conhecido como *trashing*) (CARREIRO; FONSECA; PEDREIRAS, 2003).

Este ambiente impulsionou a pesquisa e o projeto de soluções capazes de sustentar o desenvolvimento observado. Um dos pontos principais de ataque ao protocolo CSMA/CD é a diminuição na probabilidade de colisões com a segmentação da rede em diversos domínios de difusão. Entre as alternativas propostas neste contexto destacam-se: a utilização de redes totalmente baseadas em *switches*, as redes locais virtuais (VLANS) e os mecanismos de qualidade de serviço.

2.3.1 Conceito

O padrão IEEE 802.1Q define a arquitetura para as LANs virtuais, apresentando os serviços disponíveis, protocolos e algoritmos envolvidos na construção destes serviços. As redes locais virtuais constituem uma ferramenta capaz de micro-segmentar o ambiente local, facilitando sua administração e reduzindo a necessidade de alterações na configuração física da rede. Entre os objetivos principais de seu desenvolvimento estão:

- disponibilidade em todos protocolos IEEE 802 de acesso ao meio, podendo ser utilizadas tanto em redes locais de meio compartilhado, bem como em redes ponto-a-ponto;

- facilidade para administração de grupos lógicos de estações;

- restrição do tráfego trocado no ambiente local. O tráfego de determinada rede virtual deve ser distribuído apenas nos segmentos que participam da VLAN específica;

Uma VLAN é, em resumo, um grupo de estações, servidores ou outros recursos de rede que se comportam como conectados a um único segmento de rede, mesmo não estando. Um exemplo de aplicação possível para esta ferramenta seria o agrupamento de estações afins que estejam espalhadas ao longo de um prédio.

2.4 Método de Sinalização

Um aspecto importante que deve ser considerado quando redes virtuais são utilizadas diz respeito ao método de sinalização empregado na rede. Uma classificação bastante aceita define duas formas para indicação da VLAN a que pertencem os quadros: implícita ou explícita.

2.4.1 Identificação Implícita

Chama-se identificação implícita quando o mapeamento entre quadro e rede virtual está baseado em informações já carregadas no quadro ou, prontamente derivadas dele. De uma forma geral, existem três modelos básicos de sinalização implícita:

2.4.1.1 VLANs Baseadas em Portas

Cada porta do comutador é associada a uma VLAN específica. A segmentação da rede se dá através do agrupamento das portas. Neste modelo, a VLAN de cada quadro é determinada diretamente através de sua porta de entrada (Figura 2.2).

Embora apresente a vantagem de ser transparente para as estações envolvidas, o modelo possui duas limitações importantes:

- mudanças na disposição das estações exigem reconfiguração do comutador;
- apenas uma VLAN pode ser associada a determinada porta, impondo que todas estações conectadas a ela pertençam a mesma rede virtual.

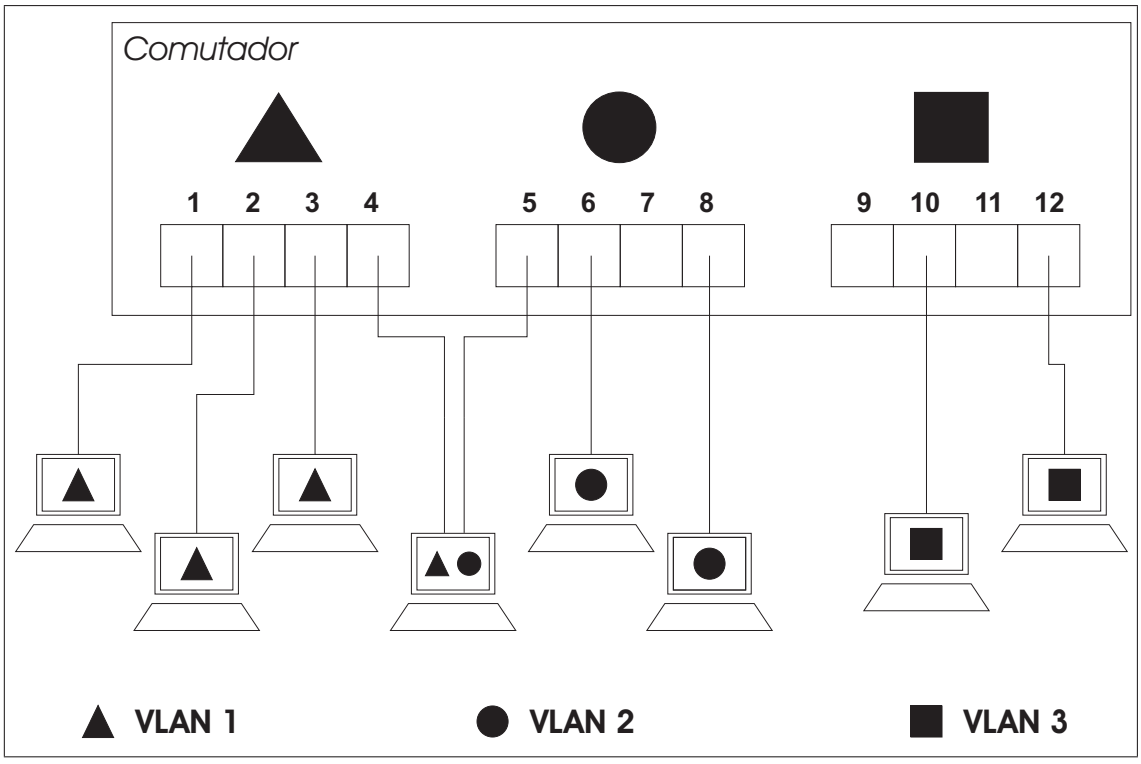


Figura 2.2: Ilustração - VLANS baseadas em portas

2.4.1.2 VLANS Baseadas em Endereços MAC

O relacionamento entre quadro e VLAN é estabelecido através do endereço *ethernet* do quadro. Cabe ao comutador manter uma tabela de mapeamentos entre endereços e VLANs correspondentes.

Neste modelo não há necessidade de reconfiguração do comutador em razão de redistribuição das estações entre suas portas.

A principal desvantagem reside na dificuldade de configuração e gerenciamento da tabela mantida nos comutadores. Além disso, geralmente determinado endereço *ethernet* pode estar relacionado com apenas uma VLAN.

2.4.1.3 VLANs Baseadas em Protocolo

A amarração entre quadro e rede virtual está baseada no tipo de protocolo superior carregado (IP, IPX, ...) ou no endereço de rede do pacote (camada 3 - arquitetura OSI). Esta implementação freqüentemente impõe requisitos maiores sobre os comutadores, pois envolve a inspeção de conteúdo do cabeçalho da camada superior da arquitetura. Normalmente é o modelo mais flexível.

2.4.2 Identificação Explícita

Já a identificação explícita está atrelada à adição de informações no quadro, exclusivamente com o objetivo de identificar a rede virtual envolvida.

Este é o principal método de sinalização previsto na norma IEEE e requer a alteração no formato dos quadros utilizado, conforme seção seguinte.

2.5 Formato de Quadro Estendido

Uma das alterações significativas introduzida pela arquitetura de redes locais virtuais é a extensão do cabeçalho dos quadros. Este novo formato, conhecido como *Tagged Frame*, foi projetado com o objetivo de permitir:

- o suporte padronizado de VLANs entre tecnologias distintas de nível físico;
- que os quadros carreguem a identificação da VLAN a que pertencem;
- a diferenciação de tráfego através da sinalização de prioridade a cada quadro transmitido (conhecida como prioridade de usuário);
- que os quadros indiquem o formato do endereço MAC carregado;

A figura 2.3 apresenta o novo formato de quadro definido para a codificação *ethernet*. As informações adicionadas ao cabeçalho dos quadros são:

- TPID (Tagged Protocol Identifier) - capaz de identificar o formato do quadro em questão (*tagged frame*): codificação *ethernet*, FDDI ou Token Ring, e:
- TCI (*Tag Control Information*) - composto pelos seguintes elementos:
    1. Prioridade de Usuário (*User Priority*);
    2. CFI (*Canonical Format Indicator*) - utilizado para indicar o formato do endereço MAC codificado;

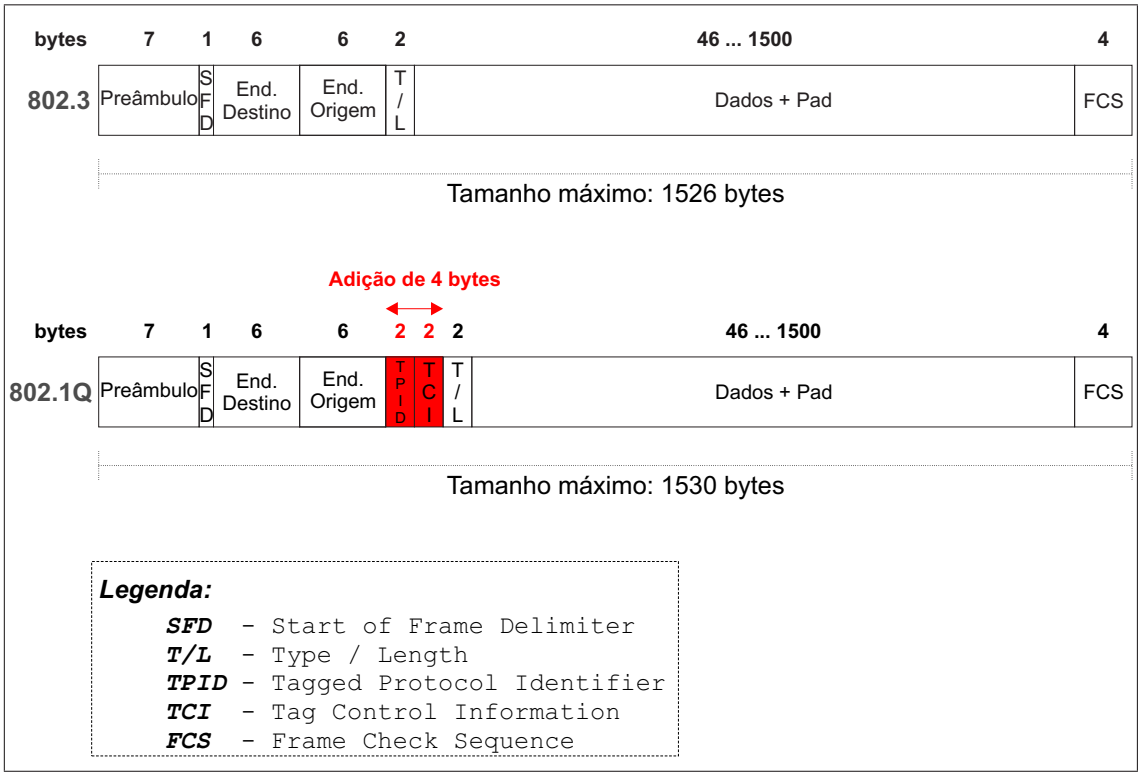


Figura 2.3: *Tagged Frame*

3. VID (*VLAN Identifier*)- campo que identifica de maneira única a VLAN a qual o quadro pertence.

Como o objeto de interesse são as redes IEEE 802.3/Ethernet, a codificação dos campos mencionados acima será apresentada apenas para este caso, negligenciando o formato adotado quando outras tecnologias de nível físico são utilizadas.

2.5.1 Formato do Campo TPID

Um quadro 802.3 com cabeçalho estendido (*Tagged Frame*) carrega em seu campo TPID um valor fixo em dois bytes que identifica a utilização de VLANs:

$$\textit{802.1Q Tag Protocol Type} = \text{81-00 (hex)}$$

2.5.2 Formato do Campo TCI

A distribuição dos componentes do campo TCI é apresentada na figura 2.4.

Fonte: IEEE 802.1Q – pág. 85

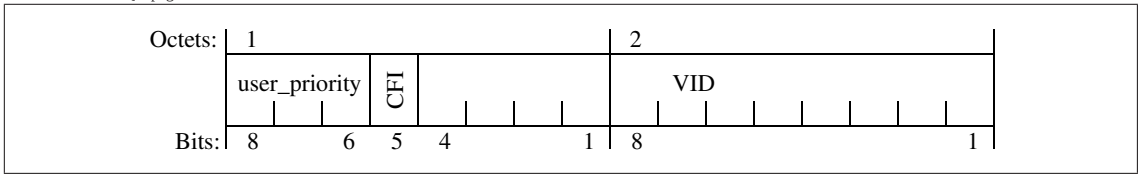


Figura 2.4: Formato do campo TCI

O campo que define a prioridade do usuário possui três bits, interpretados como um

número binário. Desta forma, existe a possibilidade de diferenciação na rede em até oito níveis de prioridade.

O bit CFI funciona como um simples sinalizador. Quando está com valor 0 indica que toda informação sobre endereçamento MAC está no formato canônico. Já quando está em alto (valor 1) possui vários significados, podendo servir por exemplo, para indicar a utilização de roteamento na origem (E-RIF - ieee 802.1D C.3.3.2.).

O campo VID possui 12 bits que identificam a VLAN a qual pertence o quadro em questão. Cada VLAN possui um identificador único na rede, sendo alguns valores reservados, conforme tabela 2.2.

Tabela 2.2: Valores de VID reservados

VID (hex)	**Significado**
000	O código nulo indica que a etiqueta de cabeçalho carrega apenas a informação de prioridade de usuário (*Priority Tagged Frame*).
001	Valor padrão configurado nas portas dos comutadores para identificar a VLAN a que pertence a porta. Quando se utiliza a sinalização implícita de VLAN, através de portas por exemplo, um pacote é classificado de acordo com sua porta de entrada.
FFF	Reservado para testes e implementação.

Cabe aos comutadores efetuar o controle de quais segmentos diretamente conectados possuem capacidade de transmitir quadros com cabeçalho estendido. Desta forma, admite-se a convivência na rede de ambos formatos, recaindo sobre os comutadores a responsabilidade de realizar as conversões necessárias. Por exemplo, quadros recebidos que não possuam informação de prioridade são associados a um valor padrão, previamente configurado.

2.6 Operação dos Comutadores - Repasse e Chaveamento de quadros

O funcionamento dos comutadores neste ambiente de redes virtuais e de diferenciação de tráfego se baseia em três módulos principais:

- repasse e chaveamento de quadros;
- manutenção da informação de suporte às decisões de repasse e chaveamento;
- gerenciamento desta estrutura.

Em relação à manutenção de suporte às decisões de repasse e chaveamento de quadros, um aspecto importante é a execução, entre as entidades da rede, do protocolo conhecido por *Spanning Tree*. Trata-se da execução de um algoritmo distribuído que tem por finalidade evitar o repasse cíclico de mensagens, transformando a visão natural da topologia de rede (grafo) em uma distribuição lógica na forma de árvore. São características deste protocolo a rápida convergência e a reconfiguração automática em caso de falhas.

Fonte: IEEE 802.1Q - pág. 37

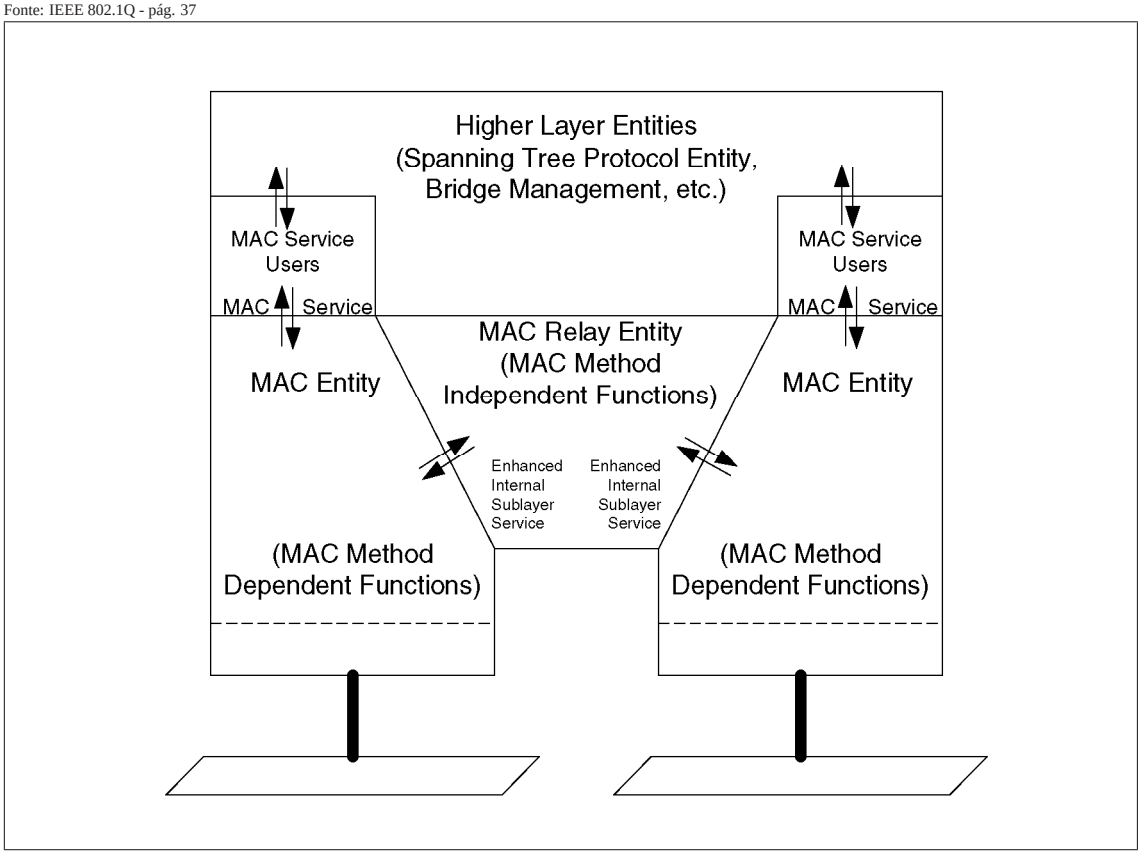


Figura 2.5: Arquitetura de um comutador

A figura 2.5 ilustra a arquitetura geral de um comutador.

O processo de encaminhamento de determinado quadro através de um comutador pode ser descrito por uma série de etapas seqüenciais:

1. Recepção do quadro;
2. Descarte caso o quadro tenha sido recebido com erro (FCS);
3. Regeneração da prioridade do quadro, caso necessário.
4. Aplicação das regras de entrada de redes virtuais com objetivo de associar cada quadro recebido a uma VLAN particular;
5. Descarte caso o quadro não tenha sido associado a qualquer rede virtual, ou caso sua porta de entrada não aceite quadros daquela VLAN;
6. Aplicação das regras de filtragem definidas que implementam o descarte seletivo com base nas informações de VID associado ao quadro e endereço MAC destino;
7. Descarte do quadro caso exceda o tamanho máximo permitido;
8. Consulta a tabela de chaveamento;
9. Repasse dos quadros recebidos para demais portas do comutador;
10. Aplicação das regras de saída de redes virtuais;

11. Seleção da classe de tráfego a que pertence o quadro, tomando por base o valor de prioridade carregado no quadro e as regras de filtragem que podem ser definidas no comutador;

12. Enfileiramento dos quadros de acordo com sua classe;

13. Descarte caso o tempo máximo definido para o trânsito de um quadro no comutador tenha sido ultrapassado;

14. Seleção dos quadros enfileirados para transmissão;

15. Recálculo do FCS, caso necessário (troca de formato do quadro, por exemplo);

16. Transmissão do quadro;

A figura 2.6 procura detalhar as fases e o relacionamento entre os diversos blocos funcionais envolvidos no processo de repasse de quadros.

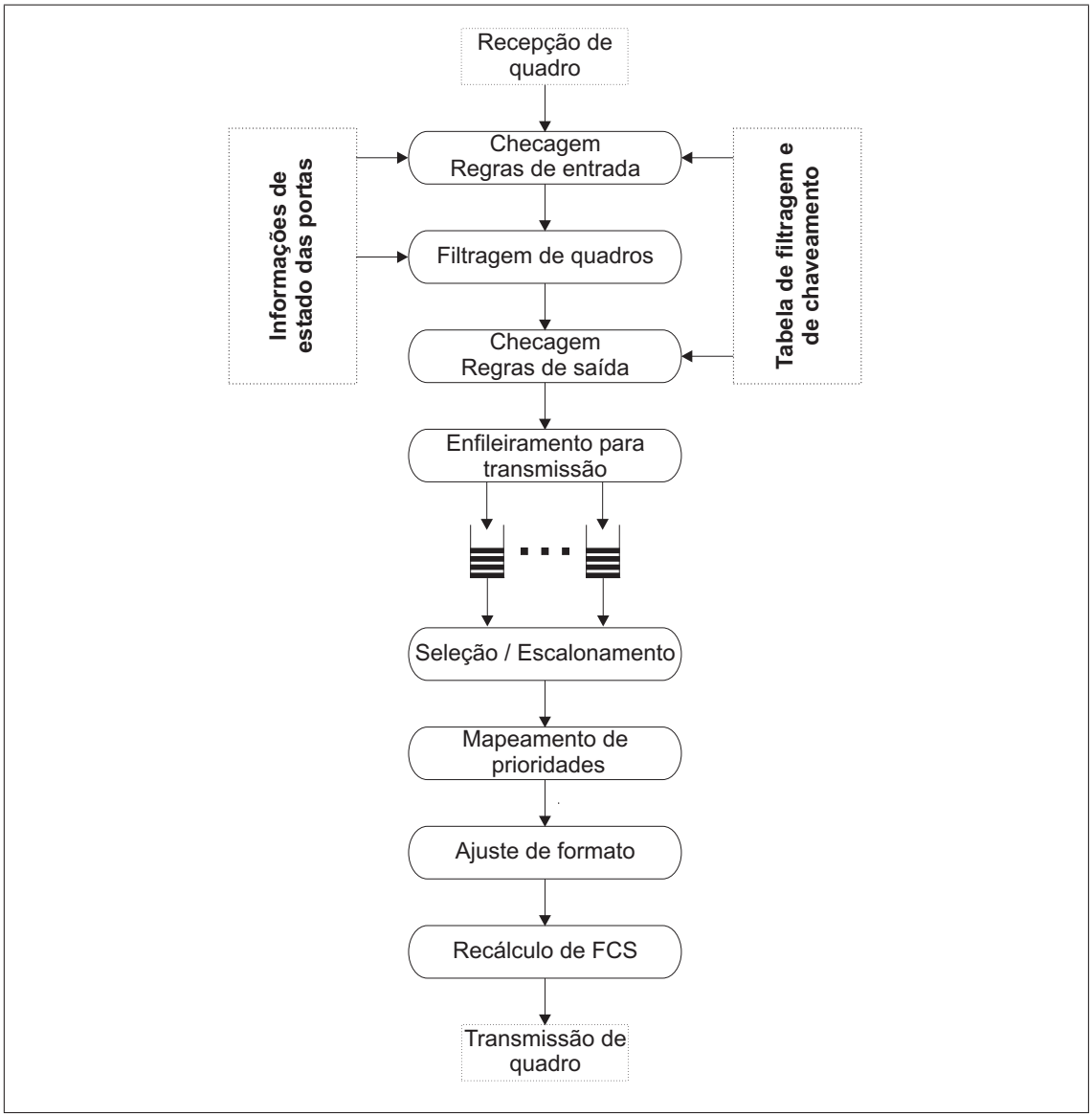


Figura 2.6: Fases envolvidas no processo de encaminhamento de quadros

Algumas das etapas percorridas durante o processo de encaminhamento dos quadros são fundamentais, conforme exposto nas subseções seguintes.

2.6.1 Regeneração do Valor de Prioridade

Uma das características definidas para os comutadores é a possibilidade de alteração no valor de prioridade dos quadros que encaminham. Esta atividade é regulada através de consultas a tabelas de mapeamento que são definidas para cada porta.

Através desta funcionalidade é possível, por exemplo, atribuir valores específicos de prioridade a quadros que ingressem por determinada porta ou que estejam destinados a um segmento de rede definido (porta de saída).

Como resultado, através de uma configuração adequada, os comutadores podem ser utilizados para marcação do tráfego na rede. Por se tratar de equipamentos confinados no ambiente local, teoricamente sob unidade administrativa, não existe previsão de nenhum protocolo de negociação a ser estabelecido entre comutadores distintos. A coerência que deve ser mantida entre os comutadores de uma mesma rede é deixada a cargo do administrador.

O mapeamento de prioridades padrão implementado nos comutadores não causa alteração na prioridade dos quadros, preservando o valor sinalizado durante o repasse do quadro.

2.6.2 Aplicação de Regras de Entrada - Redes Virtuais

Cada quadro recebido é associado a apenas uma rede virtual. Esta classificação se dá da seguinte forma:

- Se o parâmetro VID carregado pelo quadro for nulo, a rede virtual associada ao quadro será determinada através de sua porta de entrada ou do protocolo de nível superior utilizado, caso habilitado e configurado no comutador.

- Caso contrário, o VID associado ao quadro será aquele sinalizado em seu cabeçalho;

2.6.3 Enfileiramento dos Quadros

Outro ponto que merece destaque é o enfileiramento dos quadros.

O requisito básico é de que o ordenamento seja mantido entre quadros de uma única prioridade, recebidos em uma mesma porta e com o mesmo par de endereços origem e destino. O processo de enfileiramento pode ser divido nas seguintes etapas:

1. A prioridade do quadro em questão é determinada(*user_priority* $\rightarrow$ campo TCI)

2. Este valor é utilizado como índice na tabela de mapeamento entre prioridades e classes, mantida para cada porta do comutador;

3. O valor resgatado da tabela representa a classe de tráfego a qual o quadro é associado;

4. Como cada classe de tráfego possui apenas uma fila associada a si, o quadro é colocado no final da fila correspondente.

O relacionamento entre filas e classes de tráfego é de um para um, existindo portanto, uma fila distinta para cada classe de tráfego definida. Um máximo de oito classes de tráfego podem ser reconhecidas/implementadas pelos comutadores, correspondendo a existência de filas distintas para cada nível de prioridade possível (3 bits).

Esta diferenciação entre classes de serviços e prioridades dos quadros representa uma liberdade de implementação fornecida aos fabricantes de equipamentos de nível 2. Os comutadores podem ser construídos para suportar um número x ($1 \leqslant x \leqslant 8$) de filas distintas, mapeando mais de um nível de prioridade para a mesma classe de tráfego quando necessário ($x < 8$).

A tabela 2.3 apresenta o mapeamento entre classes e prioridades recomendado pelos padrões IEEE. Uma conseqüência do mapeamento apresentado é que quadros carregando valor padrão de prioridade (0) recebem tratamento preferencial em relação as prioridades 1 e 2, em comutadores com quatro ou mais classes de tráfego implementadas.

Tabela 2.3: Mapeamento prioridades x classes de tráfego

		Número de classes de tráfego disponíveis							
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Prioridades	**0**	0	0	0	1	1	1	1	2
	1	0	0	0	0	0	0	0	0
	2	0	0	0	0	0	0	0	1
	3	0	0	0	1	1	2	2	3
	4	0	1	1	2	2	3	3	4
	5	0	1	1	2	3	4	4	5
	6	0	1	2	3	4	5	5	6
	7	0	1	2	3	4	5	6	7

2.6.4 Escalonamento dos Quadros para Envio

A seleção dos quadros para efetiva transmissão segue o seguinte algoritmo:

- Para cada porta, quadros são selecionados para transmissão com base nas classes de tráfego reconhecidas pela porta. Para determinada classe, quadros são selecionados da fila correspondente, se e somente se, todas filas prioritárias estejam vazias no momento da seleção;

- Em cada fila, a ordem de chegada dos quadros é respeitada no momento de transmissão (FIFO);

A figura 2.7 apresenta uma implementação conceitual, descrita em linguagem C, para este algoritmo.

```
for(porta=0;porta < numero_portas_comutador; porta++)
  for(fila=nro_classes(porta); fila >= 0; fila=fila-1)
    if (tamanho(fila,porta) > 0)
      transmita_quadro(fila,porta);
```

Figura 2.7: Seleção de quadros para transmissão

3 MODELO ANALÍTICO PARA CÁLCULO DO ATRASO NAS COMUNICAÇÕES EM REDES *ETHERNET*

Historicamente é reconhecida a inadequação do emprego de redes *ethernet* em ambientes com requisitos temporais estritos (KHANNA; SINGH, 1994)(PARK; YOON, 1998). Isto ocorre em razão das características originais de disputa e acesso múltiplo ao meio de transmissão e da conseqüente imprevisibilidade advinda das colisões. Com o passar do tempo, diversas alternativas foram propostas para amenizar estas limitações, destacando a grande utilização de redes baseadas em *switches*.

Este capítulo procura demonstrar que o uso de redes completamente comutadas, aliado aos mecanismos de qualidade de serviço apresentados no capítulo anterior, permite a utilização desta tecnologia simples e barata em sistemas com aplicações tempo dependentes. Apresenta-se um modelo analítico, desenvolvido sobre esta estrutura, capaz de fornecer limites para o atraso fim-a-fim observado por quadros de diferentes classes. Ao final, o modelo deduzido é aplicado sobre dois casos hipotéticos, nos quais se considera a convivência na rede entre aplicações de controle e aplicações que compõem a chamada rede de dados, e não apresentam relação crítica com a variável temporal.

3.1 Premissas

O ambiente de análise é o de uma rede local *ethernet* compatível com a padronização IEEE 802. Com o objetivo de definir o cenário de análise, a seguir são apresentadas diversas considerações a respeito da arquitetura de rede e do modelo de tráfego empregados.

3.1.1 Topologia

Em sincronia com a exigência cada vez maior por desempenho e sua ampla utilização, assume-se que as redes estão totalmente estruturadas ao redor *switches* (comutadores). A interconexão das n estações presentes na rede se dá no maior grau de segmentação possível, ou seja, cada porta de um comutador possui apenas uma estação conectada ou outro *switch* encadeado (Figura 3.1).

Esta topologia proporciona que todo nó (estação) possua um canal exclusivo para transmissão e para recepção de dados (full-duplex), formando uma série de conexões ponto-a-ponto entre estações e comutadores. A idéia de disputa por um meio de transmissão compartilhado é abandonada e a possibilidade de colisões inexiste em razão da extrema segregação dos canais.

Com a ausência de colisões, a perda de pacotes se dá principalmente pelo descarte devido ao enfileiramento nas interfaces dos *switches*.

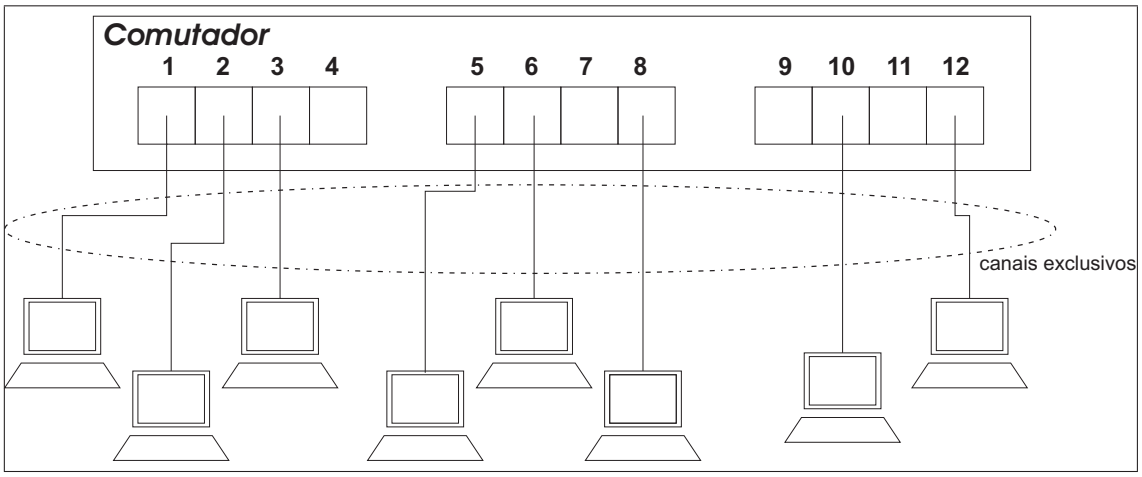


Figura 3.1: Topologia – modelo ponto-a-ponto

3.1.2 *Switches*

Existem dois modelos de chaveamento principais utilizados na construção dos switches comumente empregados em redes locais: *store-and-forward* e *cut-through*.

No modelo *store-and-forward* os quadros são encaminhados apenas após completamente recebidos e armazenados na memória do comutador. Como desvantagens principais podem ser citadas o aumento na latência dos quadros e na quantidade de memória necessária. Já o chaveamento *cut-through* admite o encaminhamento de quadros parcialmente recebidos, ou seja, assim que seus cabeçalhos sejam completamente interpretados. Isto reduz o tempo consumido em cada ponto intermediário e ameniza os requisitos sobre a capacidade de armazenamento.

Store-and-Forward Switches podem evitar o desperdício de banda da rede com a propagação de erros, uma vez que é possível recalcular o valor de FCS e identificar quadros corrompidos. Por outro lado, num ambiente local com taxas de erros muito baixas, a menor latência de comutadores *cut-through* pode ser fundamental para aplicações sensíveis ao tempo.

As análises apresentadas posteriormente consideram a utilização de *switches* do tipo *store-and-forward* por se tratar de um modelo amplamente utilizado e, principalmente, pela simplicidade de análise do ponto de vista temporal, facilitando a dedução de um modelo analítico didático e objetivo. Uma discussão mais detalhada dos principais aspectos envolvidos na construção de *switches* pode ser encontrada em (NI; QIAO; YANG, 1997).

Neste contexto, as seguintes características são consideradas para *switches store-and-forward* (compatíveis com as normas IEEE 802.1D e Q):

- compatibilidade com a utilização de redes virtuais;
- enfileiramento nas portas de saída;
- filas independentes para classes de tráfego distintas;
- escalonamento de prioridade absoluta entre as filas, o que significa dizer que determinada fila só será atendida, se e somente se, as filas de prioridade superior estiverem vazias;
- compatibilidade com o protocolo *Spanning Tree*;

O formato das mensagens que trafegam entre as diversas entidades na rede é aquele apresentado na figura 2.3 - *Tagged Frame*. A diferenciação entre as classes de tráfego é derivada da sinalização de prioridade em cada quadro transmitido, compreendendo o máximo de oito níveis distintos (3 bits).

O emprego do algoritmo *Spanning Tree* (IEEE, 1998) entre os diversos comutadores que integram a rede faz com que, do ponto de vista lógico funcional, os *switches* percebam topologias arbitrárias sob a forma de árvores. Pela teoria dos grafos *spanning tree* é uma árvore de arcos que se estende por um grafo, mantendo sua conectividade sem conter caminhos fechados. Sua utilização garante a existência de no máximo uma rota entre duas estações quaisquer, eliminando eventuais ciclos (*loops*). Este é um fator fundamental que viabiliza a análise temporal das comunicações, pois na ausência de falhas físicas os quadros percorrerão sempre o mesmo caminho ao transitarem entre duas estações específicas.

3.1.3 Modelo de tráfego

A estrutura de rede é compartilhada por dois grupos distintos de entidades: rede de controle (requisitos temporais – *real-time*) e a rede de dados (não prioritária).

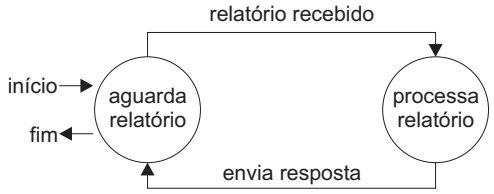
Esta divisão de funções impõe a coexistência de duas categorias básicas de comunicações com requisitos completamente diferentes. Enquanto nas transmissões *real-time* a maior preocupação está no tempo gasto para atingir o destino, os requisitos de quadros da rede de dados estão ligados a entrega sem erros ou duplicações. A diferença significativa está na tolerância aos valores e oscilações das variáveis temporais. O tráfego dito tempo dependente possui limites estritos que devem ser respeitados sob pena de, por exemplo, inutilidade da transmissão.

O padrão de geração de quadros adotado para a rede de controle é periódico e com comunicações baseadas no modelo mestre-escravo. A adoção de um modelo mestre-escravo implica na presença na rede de pelo menos uma estação controladora ou mestra, responsável por receber os quadros com relatórios de estado enviados pelas estações escravas. Desta forma, o controlador mantém uma visão global da rede de controle, intervindo através do envio de comandos específicos (sinais de controle).

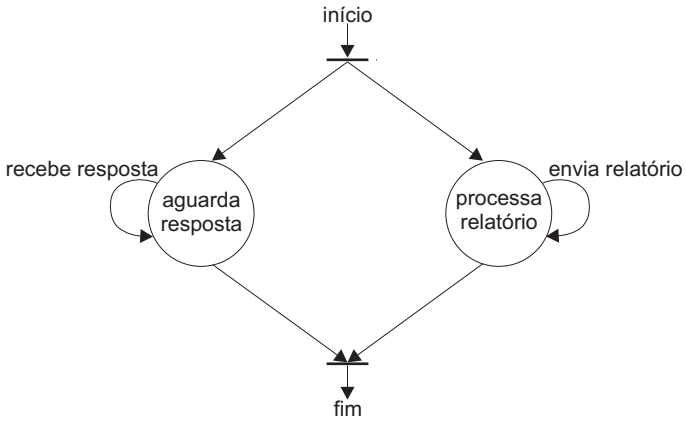
As figuras 3.2(a) e 3.2(b) demonstram o diagrama de estados representativo da comunicação entre estações (escravos) e controlador (mestre). Cabe ao controlador receber os quadros de relatório (“aguarda relatório”) gerados periodicamente pelas estações (*plan output*), e responder através do envio de comandos de controle (“envia resposta”). As estações ou escravos são representados por dois processos independentes, responsáveis pela geração periódica de relatórios de estado (“processa relatório”) e recepção de comandos do controlador (“aguarda resposta”). Este formato de comunicação é coerente com o perfil de redes de automação, sendo utilizado freqüentemente na coordenação de sensores e atuadores (CHOW; TIPSUWAN, 2001).

Já em relação à rede de dados, nenhuma restrição é imposta sobre o perfil de tráfego utilizado.

Para facilitar a compreensão das análises, o valor zero é utilizado como topo da escala, representando a prioridade mais elevada na rede ($p0$). Invariavelmente o tráfego de dados é considerado como o de prioridade mais baixa, recebendo tratamento tradicional de melhor esforço (*best effort*). Além disso, o mapeamento admitido entre prioridades e classes é um para um, significando que cada valor de prioridade possível na rede está associado a uma classe distinta das demais. Como resultado, os termos prioridade e classe de tráfego passam a representar sinônimos empregados indistintamente ao longo do texto.



(a) Mestre



(b) Escravo

Figura 3.2: Diagramas de estado – modelo mestre-escravo

3.2 Objetivos

Uma vez descrito o cenário de estudo, é possível estabelecer o objetivo específico do modelo analítico de cálculo proposto na seção seguinte: previsão dos limites extremos e valor médio para o atraso fim-a-fim observado por quadros da rede de controle no sentido estações $\rightarrow$ controlador. Tal métrica é definida na literatura como *"One-Way Delay (OWD)"* (ALMES; KALIDINDI; ZEKAUSKAS, 1999), e apresenta extrema importância para sistemas sensíveis ao tempo.

O objetivo geral é a avaliação do determinismo proporcionado por redes *ethernet* e da utilidade dos agregados mecanismos de qualidade de serviço.

3.3 Cálculo do Atraso Observado nas Comunicações

A operação estável de uma rede *ethernet* baseada em *switches* está atrelada a algumas condições de contorno, conforme apresentado em (LEE; LEE, 2002), fundamentais para validação da análise apresentada.

Em primeiro lugar há de se observar que o tráfego total presente na rede deve ser compatível com a capacidade dos comutadores. Isto é expresso pela relação 3.1, onde:

- $Capa_s(m)$ = capacidade do *switch* m em termos de quadros que ele é capaz de processar por unidade de tempo;
- n = o número de estações diretamente conectadas ao comutador m; e
- $Quadros_i$ = número máximo de quadros produzidos pela estação i por unidade de tempo.

$$Capa_s(m) \geq \sum_{i=0}^{n} Quadros_i, \qquad \text{para todo m} \tag{3.1}$$

A principal conseqüência desta relação é que cada comutador deve ser capaz de processar o número máximo de quadros gerados pelas entidades vizinhas em uma unidade de tempo.

Uma segunda condição importante diz respeito a capacidade das linhas de transmissão. A soma do tráfego endereçado a determinada estação, a cada período, deve ser menor que a capacidade da linha de recepção desta mesma estação, conforme apresentado na relação 3.2.

$$Capa_e(j) \geq \sum_{i=0}^{n} Quadros_i(j), \qquad \text{para todo j} \tag{3.2}$$

onde:

- $Capa_e(j)$ = taxa da linha de transmissão entre o *switch* e a estação j;
- $Quadros_i(j)$ = número de bits a serem transmitidos, numa unidade de tempo, da estação i para estação j.

Esta condição extrapola para cada canal o exposto na relação 3.1. Caso não seja satisfeita, a fila formada no comutador para o canal de saída de uma estação específica pode crescer indefinidamente, mesmo que a capacidade do *switch* seja suficiente para processar todo o tráfego.

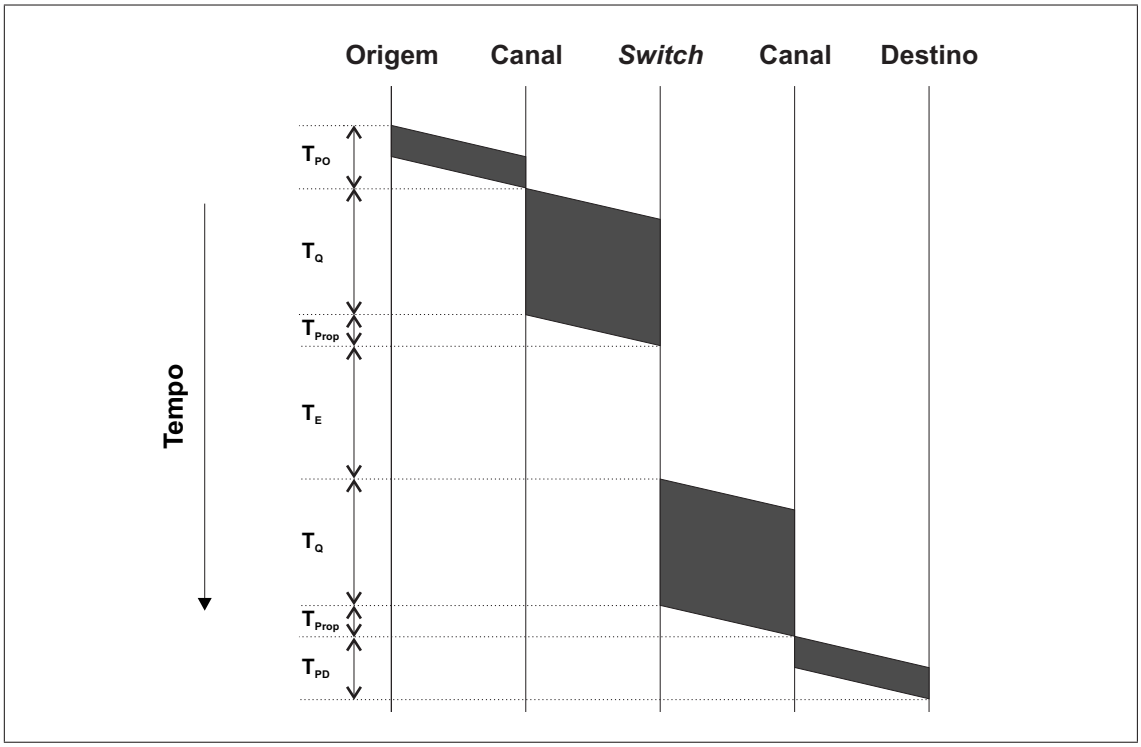


Figura 3.3: Diagrama temporal – componentes de A_T

Com a rede em operação estável, satisfeitas a relações 3.1 e 3.2, o atraso total fim-a-fim (A_T) experimentado por quadros de prioridade p ao transitarem entre duas estações

distintas quaisquer, pode ser definido como a soma de várias parcelas (Figura 3.3) – equação 3.3.

$$A_T(p) = T_{PO} + T_Q + T_{Prop} + T_E(p) + T_{PD} \tag{3.3}$$

onde:

- T_{PO} e T_{PD} = tempos de processamento consumidos nas estações origem e destino, respectivamente;
- T_Q = tempo de quadro, representa o tempo necessário para injeção dos quadros em cada linha de transmissão percorrida;
- T_{Prop} = tempo de propagação dos quadros através das linhas. Está diretamente relacionado com a velocidade de propagação e com a distância entre as estações e *switches*;
- T_E = Tempo de enfileiramento que cada quadro aguarda nas filas possivelmente formadas ao longo do caminho (*switches*). Este tempo é função da prioridade/classe de cada quadro, pois esta informação determina sua fila de espera.

O tempo gasto com o processamento das mensagens, T_{PO} e T_{PD}, são tipicamente constantes e dependem exclusivamente da capacidade de processamento das estações. Não apresentam relação direta com as condições físicas da rede ou com peculiaridades do protocolo empregado e, portanto, são ignorados na discussão apresentada. Estes dois termos são suprimidos na equação 3.4.

$$A_T(p) = T_Q + T_{Prop} + T_E(p) \tag{3.4}$$

As demais parcelas restantes que compõem o cálculo de A_T dependem fundamentalmente de fatores e condições da própria rede de comunicação, conforme demonstrado nas equações seguintes.

O cálculo de T_Q compreende a soma do tempo gasto com a inserção do quadro em cada um dos k canais percorridos em determinada comunicação, como exposto pela equação 3.5.

$$T_Q = \sum_{c=1}^{k} \frac{Tam_q}{Taxa(c)} \tag{3.5}$$

onde Tam_q é o tamanho em bits do quadro considerado, k o número de canais percorridos e $Taxa(c)$ a taxa de transferência característica do canal c, expressa em bits/segundo (bps).

Da mesma forma é preciso considerar o tempo gasto por cada bit durante seu trânsito (T_{Prop}) através de toda extensão dos canais – equação (3.6).

$$T_{Prop} = \sum_{c=1}^{k} \frac{Comp(c)}{C} \tag{3.6}$$

em que:

- $Comp(c)$ = comprimento em metros de cada canal;

- C = velocidade de propagação do sinal no meio.

A velocidade das ondas eletromagnéticas no vácuo é de aproximadamente $3 \times 10^8 m/s$ (velocidade da luz). Entretanto, em meios como cobre (par trançado) a velocidade cai para cerca de 2/3 deste valor (TANENBAUM, 1996), levando C para algo em torno de $2 \times 10^8 m/s$.

Resta portanto a definição da parcela correspondente ao tempo gasto com o enfileiramento nos *switches*. Esta parcela pode agregar grande variação ao valor final de A_T, fato que ressalta sua importância em sistemas tempo dependentes.

Em um ambiente com filas isoladas para cada uma das classes de tráfego, o tempo de contenção de um quadro com prioridade p, em cada comutador, depende do número de mensagens (N_Q) na sua própria fila e também nas filas de maior prioridade, como definido na equação 3.7.

$$T_E(p) = \sum_{i=0}^{p} (N_Q(i) \times T_{trans}) \tag{3.7}$$

O índice i representa a inspeção da fila p e de todas prioritárias a esta ($0 \leq i \leq p$), lembrando que a maior prioridade no sistema é a de valor 0. O tempo de transmissão gasto com cada quadro presente nestas filas (T_{trans}) é dado pela equação 3.8.

$$T_{trans} = \frac{Tam_q}{Taxa(c)} + T_{EQ} \tag{3.8}$$

T_{EQ} representa o tempo mínimo que deve ser mantido sem transmissão entre quadros consecutivos. Este tempo é definido no padrão como o tempo necessário para injeção de 96 bits no canal e constitui o intervalo necessário para recuperação dos níveis inferiores da arquitetura, além de impedir, no caso de acesso múltiplo, o monopólio sobre o meio de transmissão. Com uma taxa de operação de 10 Mbps $T_{EQ} = 9,6\mu s$, e para 100 Mbps $T_{EQ} = 0,96\mu s$.

A melhor hipótese possível na busca pelo menor valor de T_E é que não ocorra enfileiramento ao longo do caminho. Para um quadro de prioridade p isto significa que o tamanho da própria fila de prioridade p e das filas de maior prioridade seja nulo, isto é:

$$N_Q(i) = 0, \qquad \text{para todo i} \qquad 0 \leq i \leq p \tag{3.9}$$

Entretanto, é importante observar que ainda assim, sem enfileiramento, dependendo da carga na rede e da topologia adotada, o tempo de contenção nos *switches* pode não tender a zero. Isto ocorre porque existe a possibilidade de que um quadro, mesmo de prioridade inferior, já esteja em processo de transmissão quando uma mensagem prioritária alcança o comutador. Como não existe previsão de preempção deste processo, um quadro qualquer pode ser obrigado a aguardar, a cada *switch*, o final da transmissão de uma outra mensagem de tamanho arbitrário. Em uma rede padrão IEEE 802.3 o menor quadro possível é de 576 bits (72 bytes) e o maior 12240 bits (1530 bytes), incluindo os bits de preâmbulo e sinalizador de início de quadro.

Trata-se de um caso possível de inversão de prioridades cujos efeitos sobre o valor final de A_T não são desprezíveis. Esta situação exige que a equação 3.7 seja redefinida de modo a considerar este tempo residual (T_{Res}), possivelmente gasto com uma transmissão já em andamento.

$$T_E(i) = \sum_{p=0}^{i} (N_Q(p) \times T_{trans}) + T_{Res} \tag{3.10}$$

À medida que a carga na rede aumenta, mais relevante pode se tornar o valor de T_{Res}. Havendo disputa por canais entre classes distintas (concentração em linhas de saída), maior será a probabilidade de que um quadro de prioridade inferior já esteja em processo de transmissão. Redes com alta carga de tráfego são o foco da análise aqui apresentada, pois estudos prévios (WHEELIS, 1993)(LIAN; MOYNE; TILBURY, 2001) já demonstraram a possibilidade do emprego de redes *ethernet* em aplicações de controle, quando a carga total presente é mantida em um valor bem baixo.

3.4 Aplicação

Para exemplificar o emprego do modelo de cálculo apresentado, é necessário definir alguns parâmetros principais do ambiente. Com este objetivo foram descritos dois cenários de estudo. Em cada um destes casos, assume-se que todos canais de comunicação possuem as mesmas características básicas: 100 metros de comprimento e taxa de transmissão de 100 Mbps.

3.4.1 Caso 1: N Estações x 1 *Switch* - 2 Prioridades

Neste primeiro caso a rede está estruturada ao redor de apenas um comutador, com todas entidades diretamente conectadas a ele (Figura 3.4). As características do sistema são propostas em acordo com as premissas admitidas para o modelo de cálculo apresentado, ou seja:

- Diferenciação dos quadros entre dois níveis: controle (periódico e modelo mestre-escravo) e dados;
- Filas distintas no comutador para cada classe de tráfego, com a prioridade mais elevada atribuída às comunicações entre as entidades de controle (prioridade 0);
- escalonamento de prioridade absoluta entre as filas;
- aplicação do modelo analítico para o cálculo de A_T na rede de controle, no sentido estações $\rightarrow$ controlador.

A tabela 3.1 define os principais parâmetros adotados para a rede de controle.

Tabela 3.1: Parâmetros principais da rede de controle – Caso 1

Descrição	**Valor**
Número de entidades	15 estações e um controlador
Taxa amostragem	1000 quadros/s
Tamanho dos quadros	576 bits

O primeiro ponto a ser verificado é a operação estável da rede - relações 3.1 e 3.2. A transmissão periódica de 1000 quadros a cada segundo, efetivada por cada uma das

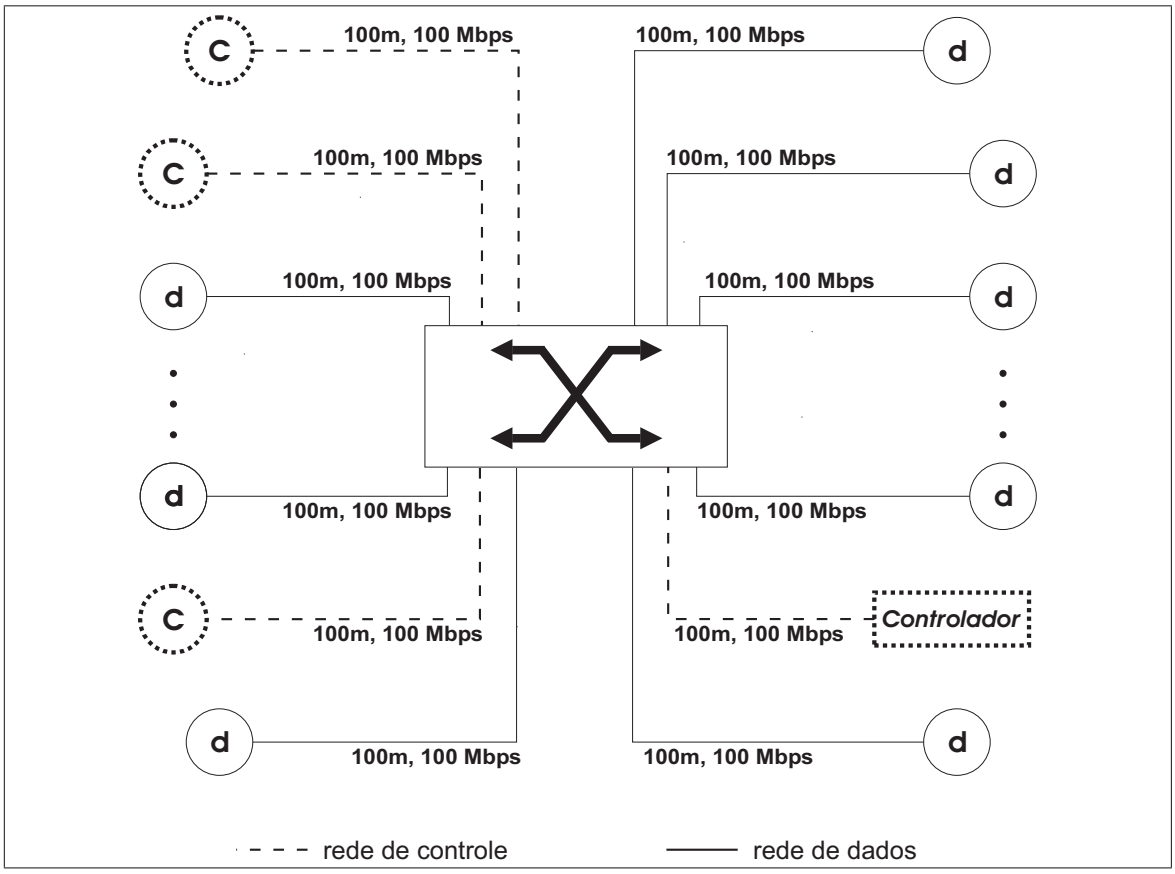


Figura 3.4: Caso 1 – Topologia

15 estações da rede de controle, corresponde a um período de amostragem de $1\ ms$ e produz um total de 15000 quadros por unidade de tempo. Esses quadros com tamanho de 576 bits (Tabela 3.1) geram o equivalente a uma taxa de $8.64\ Mbps$ ($576\ bits/quadro \times 1000\ quadros/s \times 15$ estações) destinada a estação de controle; um valor bem abaixo da taxa do canal ($100\ Mbps$) e, portanto, plenamente compatível com a relação 3.2. O período de $1ms$ adotado para amostragem na rede de controle satisfaz a grande maioria das aplicações (LEE; LEE, 2002).

O valor de 15000 quadros por segundo é muito inferior também a capacidade dos *switches* atuais que é da ordem de milhões de mensagens por segundo (3COM, 2003). Portanto, na verdade existe uma grande folga em relação a capacidade do comutador, não havendo problemas para suportar o tráfego gerado pelas outras estações pertencentes a rede de dados e satisfazendo facilmente a relação 3.1. Por exemplo, considerando uma taxa de geração de 5000 quadros/s nas estações da rede de dados, um valor cinco vezes maior do que na rede de controle, ainda seriam necessárias 197 estações ativas para transpor a barreira de $1.000.000$ quadros/s.

Como existe apenas um *switch* na rede, toda mensagem ao trafegar entre duas estações distintas quaisquer irá transpor duas linhas de transmissão idênticas: origem $\rightarrow$ *switch* e *switch* $\rightarrow$ destino. Por aplicação direta da equação 3.5, para os quadros da rede de controle:

$$T_Q = 2 \times \left(\frac{576\ bits}{100 \times 10^6\ bits/s} \right) = 11,52\mu s \tag{3.11}$$

De forma semelhante:

$$T_{Prop} = 2 \times \left(\frac{100\ m}{2 \times 10^8\ m/s} \right) = 1\mu s \tag{3.12}$$

Estes são valores fixos que dependem exclusivamente da configuração física da rede. Resta o cálculo da parcela variável que compõe o atraso, conseqüência direta do enfileiramento.

Neste sentido, é importante observar que a cada período de transmissão, o pior cenário para uma estação específica ocorre quando o seu quadro é o último a atingir o *switch*. Nesta situação e considerando a transmissão praticamente simultânea de 15 estações de mesma prioridade, o tamanho máximo da fila possivelmente encontrado por um quadro é de 14 mensagens ($N_Q(p0) = 14$).

O emprego da topologia apresentada na figura 3.4, onde o controlador não participa da rede de dados, impossibilita que um quadro com prioridade superior seja contido por uma transmissão já iniciada de um quadro com menor prioridade, independentemente da carga de tráfego presente. A inexistência de canais compartilhados entre as redes de dados e de controle, somada ao enfileiramento independente em cada porta do *switch*, faz com que estas duas classes de tráfego fiquem completamente isoladas, levando o valor do tempo de contenção residual para zero ($T_{Res} = 0$).

Com estes dados, as equações 3.8 e 3.10 fornecem, respectivamente:

$$\begin{aligned} T_{trans} &= \frac{576\ bits}{100 \times 10^6\ bits/s} + 0,96\ \mu s \\ T_{trans} &= 6,72\ \mu s \end{aligned} \tag{3.13}$$

e

$$\begin{aligned} T_{Emax}(p0) &= (14 \times T_{trans}) + T_{Res} \\ &= 14 \times 6,72\mu s + 0 \\ T_{Emax}(p0) &= 94,08\mu s \end{aligned} \tag{3.14}$$

Já o menor tempo de enfileiramento possível ocorre quando a fila está vazia ($N_Q(p0) = 0$), ou seja, o quadro de determinada estação é o primeiro a atingir o *switch* no período considerado. Nesta situação todas parcelas da equação 3.10 se anulam e:

$$T_{E_{min}}(p0) = 0 \tag{3.15}$$

Com estas considerações acerca da parcela variável que compõe o cálculo de A_T, através da aplicação da equação 3.4, são definidos os limites mínimo e máximo para o atraso total dos quadros da rede de controle.

$$\begin{aligned} A_{T_{max}}(p0) &= T_Q + T_{Prop} + T_{E_{max}}(0) \\ &= 11,52\mu s + 1\mu s + 94,08\mu s \\ &= 106,60\mu s \\ A_{Tmax}(p0) &= 0,106ms \end{aligned} \tag{3.16}$$

e

$$\begin{aligned} A_{Tmin}(p0) &= T_Q + T_{Prop} + T_{Emin}(0) \\ &= 11,52\ \mu s + 1\ \mu s \\ A_{Tmin}(p0) &= 12,52 \mu s \end{aligned} \tag{3.17}$$

De forma semelhante, é coerente supor que em média o quadro transmitido por determinada estação encontrará a fila no comutador com a metade de seu tamanho máximo $\left(\frac{N_Q(0)}{2} = \frac{14}{2} = 7\right)$ - equação 3.18.

$$\begin{aligned} T_{E_{med}}(p0) &= (7 \times T_{trans}) + T_{Res} \\ &= 7 \times 6,72 \mu s + 0 \\ T_{E_{max}}(p0) &= 47,04 \mu s \end{aligned} \tag{3.18}$$

E, portanto:

$$\begin{aligned} A_{T_{med}}(p0) &= T_Q + T_{Prop} + T_{E_{med}}(0) \\ &= 11,52 \mu s + 1 \mu s + 47,04 \mu s \\ A_{T_{med}}(p0) &= 59,56 \mu s \end{aligned} \tag{3.19}$$

O isolamento entre as duas classes de tráfego da rede se reflete nos cálculos apresentados que não demonstram a influência de parâmetros característicos da rede de dados. O gráfico apresentado na figura 3.5 mostra o comportamento de A_T durante o aumento gradual no número de estações da rede de controle, enquanto os demais parâmetros são mantidos constantes. A tabela 3.2 resume alguns dos valores calculados a partir da aplicação direta das equações apresentadas.

Tabela 3.2: Valores teóricos de A_T – fluxo de prioridade 0

Estações ($p0$)	$\mathbf{A_{T_{min}}(p0)}$	$\mathbf{A_{T_{med}}(p0)}$	$\mathbf{A_{T_{max}}(p0)}$
5	0,01252	0,02596	0,03940
10	0,01252	0,03940	0,07300
15	0,01252	0,05956	0,10660
25	0,01252	0,09316	0,17380
50	0,01252	0,17380	0,34180
70	0,01252	0,24100	0,47620
90	0,01252	0,30820	0,61060
110	0,01252	0,37540	0,74500
130	0,01252	0,44260	0,87940
148	0.01252	0.50308	1.00036
150	0,01252	0,50980	1,01380

Obs: *valores expressos em milissegundos (ms)*

Observa-se o aumento linear do atraso total à medida em que mais estações são inseridas na rede. Isto é conseqüência do aumento proporcional de quadros na fila do comutador, elevando o tempo de contenção ao qual os quadros são submetidos. O aumento de

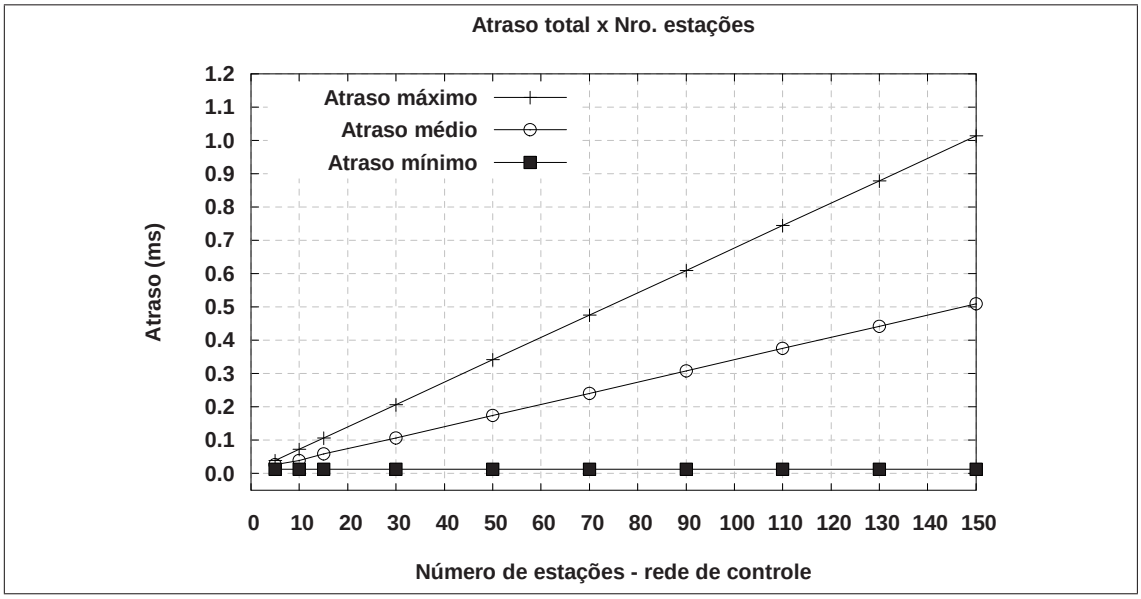


Figura 3.5: Caso 1 – Variação de A_T em função do aumento de estações na rede de controle

entidades na rede não se reflete em A_{Tmin} uma vez que, neste caso, a fila é considerada sempre vazia.

Em cada configuração, conforme equações anteriormente apresentadas, o valor máximo é obtido considerando o maior tamanho de fila que um quadro pode encontrar (Número de estações $-$ 1) e o valor médio, a metade deste limite ((Número de estações $-$ 1) / 2).

Quando a rede de controle é composta por aproximadamente 148 estações o valor de A_{Tmax} atinge o limiar de 1ms. Este é o período de transmissão das estações e representa o limite de validade do modelo de cálculo utilizado. A partir deste ponto o *switch* pode não ser capaz de despachar todas mensagens geradas em um período de transmissão. É possível portanto, que o fluxo de chegada de quadros ultrapasse o fluxo de saída, fazendo com que a fila tenda a um valor infinito. Nesta condição a previsão para os números máximo e médio de quadros na fila não é mais válida.

Conseqüentemente, a faixa de operação do cenário considerado é definida, impondo o valor limite de 148 estações para garantia de obediência aos limites de atraso calculados.

Já a figura 3.6 mostra o comportamento de A_T quando o número de estações é mantido constante (15), mas o tamanho dos quadros na rede de controle é gradativamente incrementado de 72 até 1530 *bytes*. Neste caso, o número de quadros em disputa por posições na fila é mantido sempre o mesmo, mas o tempo necessário para transmissão de cada um deles (T_{trans}) é proporcionalmente elevado em razão do maior número de bits em cada mensagem.

Novamente o limite de 1ms é ultrapassado, significando que nesta configuração adotada o maior quadro recomendável para a rede de controle possui cerca de 770 *bytes*. Acima deste valor, as garantias temporais são perdidas e o comportamento estável do sistema fica comprometido.

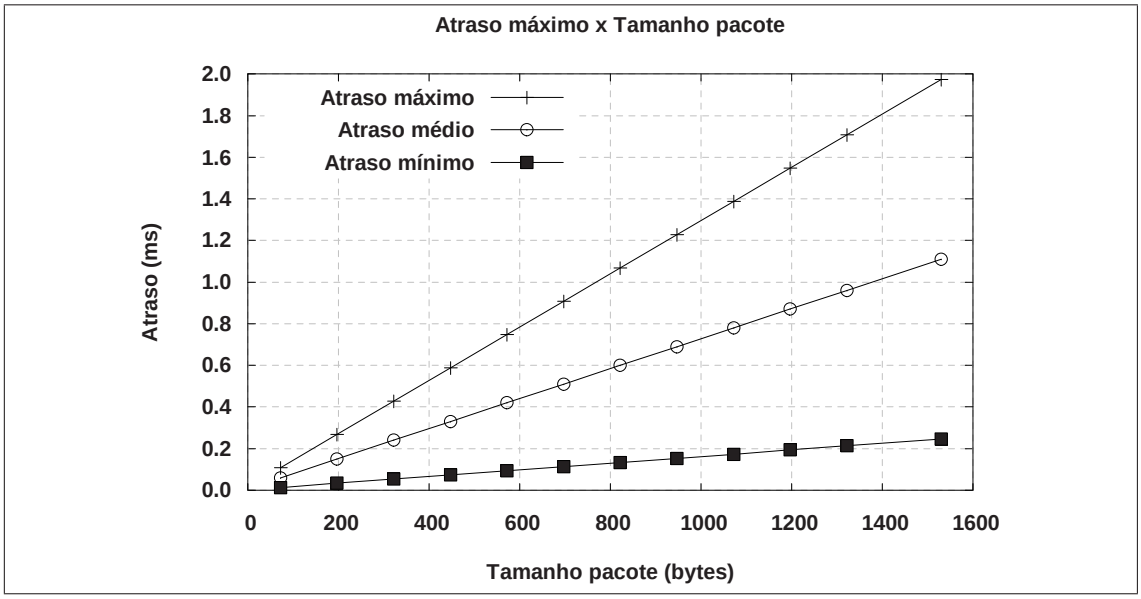


Figura 3.6: Caso 1 - Variação de A_T em função do aumento no tamanho dos quadros (15 estações na rede de controle)

3.4.2 Caso 2: N estações x 2 *Switches* - 3 Prioridades

No caso anterior, a ausência de disputa pelos recursos da rede, originada no isolamento criado entre as distintas classes de tráfego, impede a observação dos efeitos da interferência entre os fluxos de prioridades distintas. Com este objetivo, a figura 3.7 apresenta uma nova topologia em que se sugere a existência de um canal de concentração para o trânsito dos quadros na rede.

Outra diferença significativa é que os quadros da rede de controle são transmitidos sob duas prioridades distintas (0 e 1), conforme tabela 3.3. Novamente o tráfego de menor prioridade pertence às comunicações de dados ($p > 1$).

Em relação às comunicações, o tráfego de controle produz neste caso um total de 30.000 quadros a cada segundo. A taxa gerada por estas comunicações é de 19,20Mbps, e pode ser facilmente obtida pela soma das transmissões efetuadas sob cada umas das prioridades da rede de controle (0 e 1) a cada período, conforme equação 3.20.

$$\begin{aligned} Taxa_{controle} &= Taxa_{p0} + Taxa_{p1}; \\ &= 7,68Mpbs + 11,52Mbps \\ &= 19,20Mbps; \end{aligned} \tag{3.20}$$

onde:

$$\begin{aligned} Taxa_{p0} &= 768\ bits/quadro \times 1000\ quadros/seg. \times 10\ estacoes \\ &= 7,68Mbps \end{aligned} \tag{3.21}$$

e

$$\begin{aligned} Taxa_{p1} &= 576\ bits/quadro \times 1000\ quadros/seg. \times 20\ estacoes \\ &= 11,52Mbps \end{aligned} \tag{3.22}$$

Tabela 3.3: Parâmetros principais da rede de controle – Caso 2

Descrição	**Valor**
Prioridade 0 (p0)	
Número de entidades	10 estações
Taxa amostragem	1000 quadros/s
Tamanho dos quadros	768 bits
Período	1ms
Prioridade 1 (p1)	
Número de entidades	20 estações
Taxa amostragem	1000 quadros/s
Tamanho dos quadros	576 bits
Período	1ms

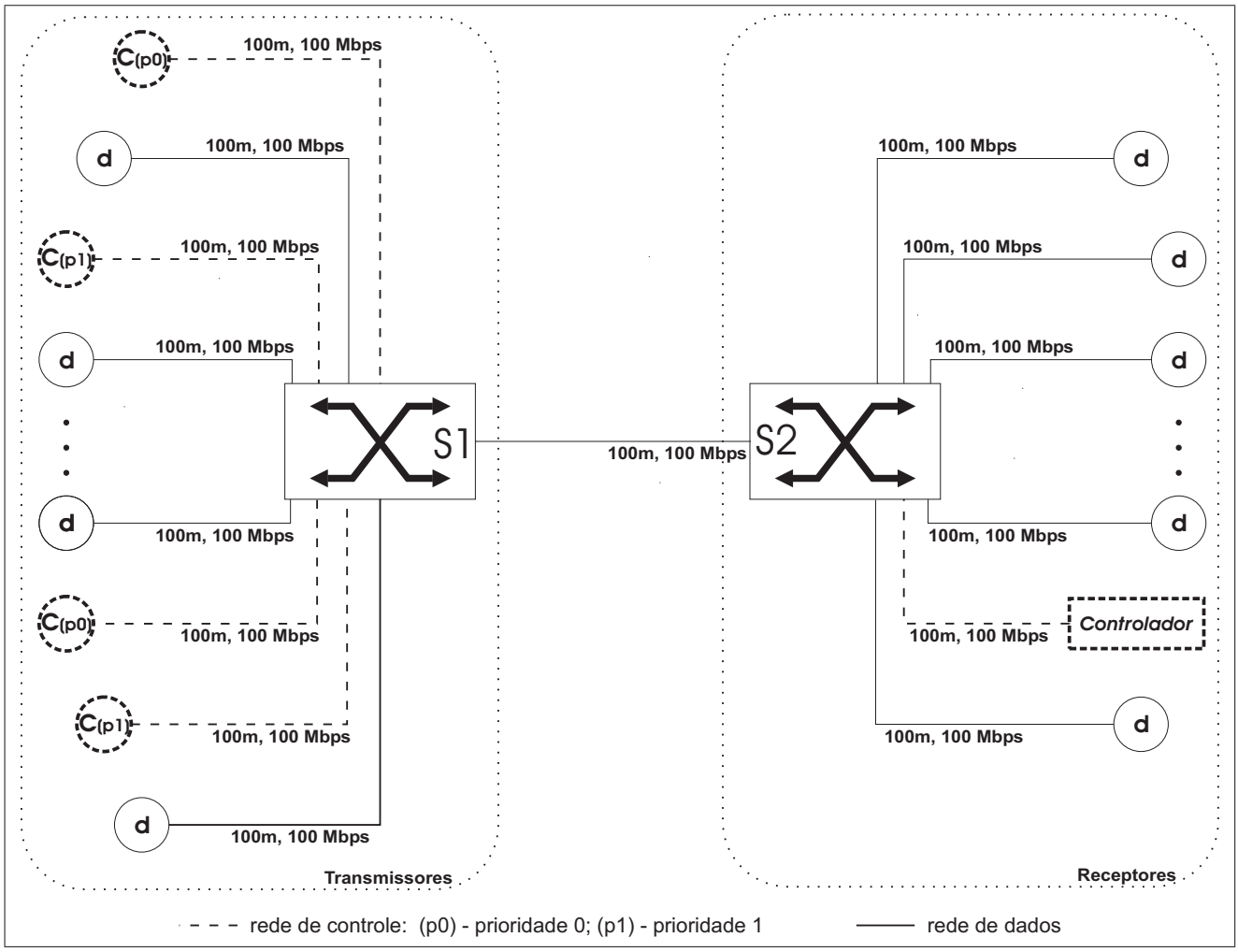


Figura 3.7: Caso 2 – Topologia

De maneira semelhante ao exemplo anterior, os valores de 30.000 quadros por segundo e taxa de transmissão de 19,20Mbps satisfazem plenamente as relações 3.2 e 3.1.

A seguir, os resultados da análise temporal sobre a rede de controle são apresentados isoladamente para cada classe de tráfego distinta.

3.4.2.1 Fluxos de Prioridade 0

A presença de dois comutadores encadeados na rede (Figura 3.7) faz com que o caminho percorrido por determinado quadro, quando em trânsito entre uma estação qualquer e o controlador, seja composto por três linhas de transmissão: origem→*switch*(S1), *switch*(S1)→*switch*(S2) e *switch*(S2)→controlador.

As equações 3.5 e 3.6 fornecem:

$$T_Q(p0) = 3 \times \left(\frac{768\ bits}{100 \times 10^6\ bits/s} \right) = 23,04\ \mu s \tag{3.23}$$

$$T_{Prop}(p0) = 3 \times \left(\frac{100\ m}{2 \times 10^8\ m/s} \right) = 1,5\ \mu s \tag{3.24}$$

A disposição em série dos dois *switches* na rede obriga que o tempo de enfileiramento seja estudado separadamente em cada um deles.

Como existem 10 estações com prioridade 0, o tamanho máximo enfrentado nas filas formadas no primeiro *switch* (S1 - figura 3.7), em cada ciclo de transmissão, é de 9 mensagens ($N_{Qmax}(p0, S1) = 9$). É importante observar que em S2, embora as mensagens da rede de controle possuam um destino comum, o fato de todos quadros alcançarem o *switch* através de um único canal (S1→S2), com taxa de transmissão igual a da linha de saída (S2→controlador), impede a formação de filas e assim, $N_{Qmax}(p0, S2) = 0$.

A disputa entre quadros de prioridades distintas pelo canal único entre os *switches* pode impor variações consideráveis ao valor do tempo residual de contenção (T_{Res}) em S1. Determinado quadro, mesmo prioritário, pode ser obrigado a aguardar a liberação do canal ocupado por uma transmissão já em andamento. Como o canal é compartilhado inclusive com o tráfego da rede de dados, que não tem restrições quanto ao tamanho ou quanto a taxa de transmissão, no pior caso isto pode significar o tempo de injeção do maior quadro possível na rede:

$$\begin{aligned} T_{Res_{max}}(S1, p0) &= \frac{12240\ bits}{100 \times 10^6\ bits/s} \\ T_{Res_{max}}(S1, p0) &= 122,40\ \mu s \end{aligned} \tag{3.25}$$

Já no segundo *switch* (S2) o tempo residual de contenção não é significativo, uma vez que apenas tráfego de controle, originado de uma mesma porta de entrada e naturalmente enfileirado, é direcionado ao controlador.

$$T_{Res_{max}}(S2, p0) = 0\ \mu s \tag{3.26}$$

Através das equações 3.8 e 3.10:

$$\begin{aligned} T_{trans}(p0) &= \frac{768\ bits}{100 \times 10^6\ bits/s} + 0,96\ \mu s \\ T_{trans}(p0) &= 8,64\ \mu s \end{aligned} \tag{3.27}$$

e

$$\begin{aligned} T_{E_{max}}(p0) &= T_{E_{max}}(p0,S1) + T_{E_{max}}(p0,S2) \\ T_{E_{max}}(p0) &= [(9 \times T_{trans}) + T_{Res_{max}}(S1)] + 0 \\ &= 9 \times 8,64\ \mu s + 122,40\ \mu s \\ &= 200,16\ \mu s \\ &= 0,2\ ms \end{aligned} \tag{3.28}$$

A busca pelo limite mínimo do tempo de enfileiramento deve considerar o melhor cenário possível, ou seja, fila vazias ($N_Q(p0) = 0$) e tempo residual nulo ($T_{Res_{min}} = 0$). Por aplicação direta da equação 3.10:

$$T_{E_{min}}(p0) = 0\ \mu s \tag{3.29}$$

Com os valores máximos e mínimos estabelecidos para o tempo de enfileiramento é possível calcular, através da relação 3.4, os limites para a variação do atraso total observado pelos quadros pertencentes aos fluxos de prioridade mais elevada na rede.

$$\begin{aligned} A_{T_{max}}(p0) &= T_Q(p0) + T_{Prop}(p0) + T_{E_{max}}(p0) \\ &= 23,04\mu s + 1,5\mu s + 200,16\mu s \\ &= 224,70\mu s \\ A_{T_{max}}(p0) &= 0,224ms \end{aligned} \tag{3.30}$$

$$\begin{aligned} A_{T_{min}}(p0) &= T_Q(p0) + T_{Prop}(p0) + T_{E_{min}}(p0) \\ &= 23,04\ \mu s + 1,5\ \mu s + 0 \\ A_{T_{min}}(p0) &= 24,54\mu s \end{aligned} \tag{3.31}$$

Considerando, a exemplo do efetuado anteriormente, que em média as estações encontram a fila em S1 com a metade de seu tamanho máximo ($N_{Q_{med}}(p0,S1) = \frac{N_{Q_{max}}(p0,S1)}{2} = 9/2 = 4quadros$), pode-se prever o valor médio de A_T.

$$\begin{aligned} A_{T_{med}}(p0) &= T_Q(p0) + T_{Prop}(p0) + T_{E_{med}}(p0) \\ &= 23,04\mu s + 1,5\mu s + 95,76\mu s \\ &= 120,30\mu s \\ A_{T_{med}}(p0) &= 0,120\ ms \end{aligned} \tag{3.32}$$

onde $T_{E_{med}}(p0)$ é dado por:

$$\begin{aligned} T_{E_{med}}(p0) &= T_{E_{med}}(p0,S1) + T_{E_{med}}(p0,S2) \\ T_{E_{med}}(p0) &= [(4 \times T_{trans}) + T_{Res_{med}}(S1)] + 0 \\ &= \left[(4 \times T_{trans}) + \frac{T_{Res_{max}}(S1)}{2}\right] \\ &= 4 \times 8,64\ \mu s + \frac{122,40}{2}\ \mu s \\ &= 95,76\ \mu s \\ &= 0,96\ ms \end{aligned} \tag{3.33}$$

É importante notar que o tempo de contenção residual médio ($T_{Res_{med}}$) a que os quadros são submetidos possui valor igual a metade de seu limite superior ($T_{Res_{med}} = \frac{T_{Res_{max}}}{2}$).

A figura 3.8 mostra a evolução linear da média e dos limites máximos e mínimos para o atraso total dos quadros de prioridade 0, enquanto um número cada vez maior de estações é incluído na rede. Os valores apresentados neste gráfico foram obtidos da seguinte forma:

- $A_{T_{max}}$ - tempo gasto por um quadro na pior condição possível, ou seja, tamanho máximo para fila e contenção pelo maior quadro possível na rede ($T_{Res_{max}}(S1, p0) = 122,40\ \mu s$ – equação 3.25);
- $A_{T_{med}}$ - fila com a metade do número máximo de quadros possível ($N_Q(S1, p0) =$ estações$/2$).
- $A_{T_{min}}$ - filas vazias e tempo nulo de contenção. Independe do aumento de estações na rede.

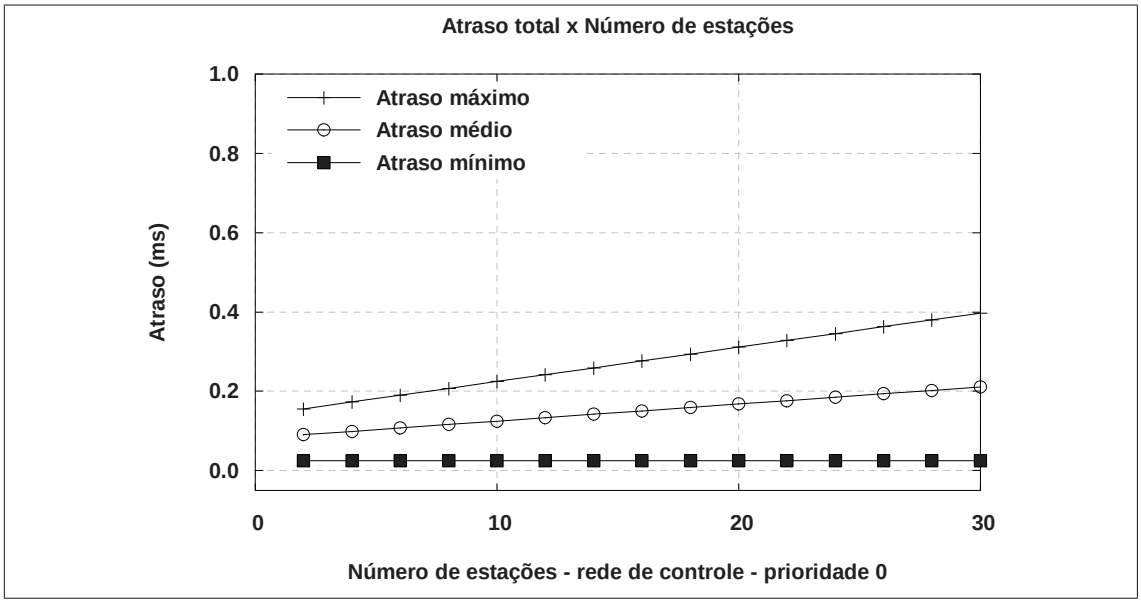


Figura 3.8: Caso 2 – Variação de A_T em função do aumento de estações na classe de prioridade 0 ($p0$)

3.4.2.2 Fluxos de Prioridade 1

Da mesma maneira empregada para a classe de prioridade superior, é possível calcular diretamente os valores do tempo de quadro (T_Q) e do tempo de propagação (T_{Prop}) também para os fluxos de quadros com prioridade 1.

$$T_Q(p1) = 3 \times \left(\frac{576\ bits}{100 \times 10^6\ bits/s} \right) = 17,28\ \mu s \tag{3.34}$$

$$T_{Prop}(p1) = 3 \times \left(\frac{100\ m}{2 \times 10^8\ m/s} \right) = 1,5\ \mu s \tag{3.35}$$

Note-se que o valor de T_{Prop} é o mesmo para os fluxos de ambas prioridades ($p0$ e $p1$). Esta coincidência ocorre porque o caminho percorrido pelos quadros é idêntico e seu cálculo independe de qualquer outro fator, senão da velocidade do sinal no meio e do comprimento dos canais.

Novamente, na busca pelo limite do tempo de enfileiramento é preciso considerar de forma isolada cada um dos *switches* da rede. Tanto o enfileiramento (T_E) quanto a contenção residual (T_{Res}) podem apresentar valores significativos em S1. Isto não acontece em S2 em razão da ausência de enfileiramento na porta de comunicação com o controlador.

A diferença é que a previsão para os tamanhos das filas enfrentadas por quadros desta classe deve considerar não apenas as 20 estações que transmitem sob a mesma prioridade (concorrem pela mesmas filas), mas também as 10 estações pertencentes à classe de prioridade mais elevada. Com isto, um quadro de prioridade 1 pode encontrar outros 19 quadros a sua frente na fila e mais 10 mensagens enfileiradas sob a prioridade 0, ou seja:

$$\begin{aligned} N_{Qmax}(p1) &= N_{Q_{max}}(p0) + N_{Q_{max}}(p1) \\ &= 10\ quadros_{p0} + 19\ quadros_{p1} \end{aligned} \tag{3.36}$$

Os tempos de contenção residual máximos possíveis nos dois comutadores para um quadro de prioridade 1 são exatamente os mesmos considerados para os fluxos de prioridade 0 (eqs. 3.25,3.26) , isto é:

$$\begin{aligned} T_{Res_{max}}(S1, p1) &= T_{Res_{max}}(S1, p0) \\ T_{Res_{max}}(S1, p1) &= 122,40\ \mu s \end{aligned} \tag{3.37}$$

e

$$\begin{aligned} T_{Res_{max}}(S2, p1) &= T_{Res_{max}}(S2, p0) \\ T_{Res_{max}}(S1, p1) &= 0\ \mu s \end{aligned} \tag{3.38}$$

Assim, para o cálculo do tempo gasto com o enfileiramento (eq. 3.10) resta a obtenção do tempo de transmissão (eq. 3.8):

$$\begin{aligned} T_{trans}(p1) &= \frac{576\ bits}{100 \times 10^6\ bits/s} + 0,96\ \mu s \\ T_{trans}(p1) &= 6,72\ \mu s \end{aligned} \tag{3.39}$$

Com isto,

$$\begin{aligned} T_{E_{max}}(p1) &= T_{E_{max}}(p1, S1) + T_{E_{max}}(p1, S2) \\ T_{E_{max}}(p1) &= [(19 \times T_{trans}(p1)) + (10 \times T_{trans}(p0)) + T_{Res_{max}}(S1)] + 0 \\ &= 19 \times 6,72\ \mu s + 10 \times 8,64\ \mu s + 122,40\ \mu s \\ &= 336,48\ \mu s \\ &= 0,336\ ms \end{aligned} \tag{3.40}$$

A amplitude máxima possível para o atraso total observado pelos quadros de prioridade 1 pode então ser calculada pela aplicação da equação 3.4 nos limites máximos e mínimos previamente obtidos:

$$\begin{aligned} A_{T_{max}}(p1) &= T_Q(p1) + T_{Prop}(p1) + T_{E_{max}}(p1) \\ &= 17,28\mu s + 1,5\mu s + 336,48\mu s \\ &= 355,26\mu s \\ A_{T_{max}}(p1) &= 0,355ms \end{aligned} \tag{3.41}$$

$$\begin{aligned} A_{Tmin}(p1) &= T_Q(p1) + T_{Prop}(p1) + T_{E_{min}}(p1) \\ &= 17,28\ \mu s + 1,5\ \mu s + 0;\mu s \\ A_{Tmin}(p1) &= 18,78\mu s \end{aligned} \tag{3.42}$$

O gráfico da figura 3.9 apresenta a variação dos limites de A_T enquanto o número de estações na rede de controle é incrementado.

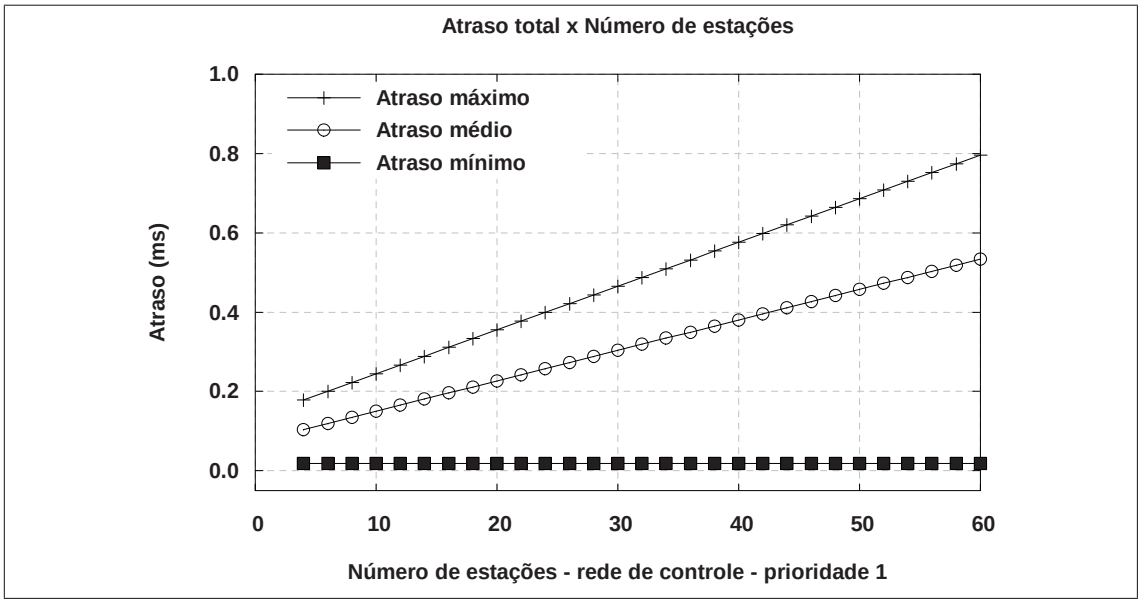


Figura 3.9: Caso 2 – Variação de A_T em função do aumento de estações na classe de prioridade 1 ($p1$)

Quadros de prioridade 1 podem encontrar filas vazias de ambas prioridades, desde que atinjam o comutador antes de qualquer outro quadro no período de transmissão considerado ($A_{T_{min}}$). Entretanto, caso o tamanho da fila de prioridade 0 não seja nulo, um quadro de prioridade 1 será obrigado a aguardar que ela esvazie. Durante este período de contenção, mais quadros prioritários podem atingir o comutador aumentando o tamanho da fila até seu valor máximo correspondente ao número de estações com prioridade 0 (N_{p0}).

Desta forma, num ambiente de transmissões periódicas e praticamente simultâneas entre as diversas estações, é esperado que um quadro de prioridade 1 seja obrigado a aguardar, caso não encontre a fila prioritária vazia, o tempo necessário para transmissão de todos quadros com prioridade superior e de metade dos quadros das demais estações de mesma prioridade. Tal previsão se reflete nas equações 3.43 e 3.44, e constitui a razão pela qual a linha representativa de $A_{T_{med}}$ aparece na figura 3.9 ligeiramente deslocada para cima.

$$
\begin{aligned}
T_{E_{med}}(p1) &= T_{E_{med}}(p1, S1) + T_{E_{med}}(p1, S2) \\
T_{E_{med}}(p1) &= \left[\left(\frac{N_{Q_{p1}} - 1}{2} \times T_{trans}(p1)\right) + \left(N_{Q_{p0}} \times T_{trans}(p0)\right) + T_{Res_{med}}(S1)\right] + 0 \\
&= [(9 \times T_{trans}(p1)) + (10 \times T_{trans}(p0)) + T_{Res_{med}}(S1)] + 0 \\
&= 9 \times 6,72\ \mu s + 10 \times 8,64\ \mu s + \frac{122,40}{2}\ \mu s \\
&= 208,08\ \mu s \\
&= 0,208\ ms
\end{aligned}
\tag{3.43}
$$

Mais uma vez, o valor médio para o tempo de contenção residual ($T_{Res_{med}}$) é considerado igual a metade de seu valor máximo possível ($T_{Res_{med}} = \frac{T_{Res_{max}}}{2}$).

$$
\begin{aligned}
A_{T_{med}}(p1) &= T_Q(p1) + T_{Prop}(p1) + T_{E_{med}}(p1) \\
&= 17,28\mu s + 1,5\mu s + 208,08\mu s \\
&= 226,86\mu s \\
A_{T_{med}}(p1) &= 0,226ms
\end{aligned}
\tag{3.44}
$$

Os valores apresentados no gráfico da figura 3.9 para os quadros de prioridade 1 não podem ser obtidos independentemente, pois guardam dependência direta com os parâmetros característicos da classe de prioridade 0. Em razão disto, durante a montagem do gráfico foi considerada a mesma proporção apresentada na tabela 3.3, na qual o número de estações transmitindo sob prioridade 1 é mantido igual ao dobro de estações de prioridade 0.

A tabela 3.4 resume alguns dos pontos calculados para ambos fluxos. Através da inspeção da coluna contendo a soma dos valores de $A_{T_{max}}$ é possível determinar o ponto a partir do qual o sistema atinge o limite de $1\ ms$. Com 24 e 48 estações transmitindo sob prioridades 0 e 1 respectivamente, a soma do tempo máximo de atraso total possível ultrapassa o período de geração de quadros, característico das entidades de controle. A partir deste ponto não é possível garantir que todos pacotes gerados em um período de transmissão sejam efetivamente entregues ao controlador. A taxa de entrada nas filas pode ultrapassar a taxa de saída, possibilitando inclusive a perda de quadros da rede de controle por escassez de recursos nos comutadores.

Este ponto limite para validade dos extremos temporais calculados é totalmente dependente da configuração do sistema, devendo ser recalculado a cada alteração nos parâmetros dos fluxos de quadros (Tabela 3.3) ou na topologia da rede (Figura 3.7).

Tabela 3.4: Valores teóricos de $A_{T_{max}}$ – fluxos de prioridades 0 e 1

Estações ($p0$)	$\mathbf{A_{T_{max}}(p0)}$	*Estações ($p1$)*	$\mathbf{A_{T_{max}}(p1)}$	$\mathbf{A_{T_{max}}(p0)+A_{T_{max}}(p1)}$
02	0,1556	04	0,1786	0,3342
04	0,1729	08	0,2228	0,3956
06	0,1901	12	0,2669	0,4571
08	0,2074	16	0,3111	0,5185
10	0,2247	20	0,3553	0,5800
12	0,2420	24	0,3994	0,6414
14	0,2593	28	0,4436	0,7028
16	0,2765	32	0,4877	0,7643
18	0,2938	36	0,5319	0,8257
20	0,3111	40	0,5761	0,8872
22	0,3284	44	0,6202	0,9486
24	**0,3457**	**48**	**0,6644**	**1,0100**
26	0,3629	52	0,7085	1,0715
28	0,3802	56	0,7527	1,1329
30	0,3975	60	0,7969	1,1944

Obs: *valores temporais expressos em milissegundos (ms)*

4 VALIDAÇÃO DO MODELO

A abordagem escolhida para validação do modelo de cálculo proposto é a comparação dos resultados teóricos com os obtidos através de simulação dos cenários escolhidos como estudos de caso. O simulador empregado é o *NS* ou *Network Simulator*, muito utilizado no meio científico.

Ao longo deste capítulo são apresentadas as principais características do simulador e, em linhas gerais, a técnica de modelagem adotada. O NS não dispõe de suporte para modelagem dos mecanismos de diferenciação de tráfego no nível de enlace (padrão IEEE 802.1D/Q). Por esta razão são traçadas algumas considerações que justificam a utilização do módulo de serviços diferenciados definido um nível acima na arquitetura. Além disso, são descritas algumas alterações efetuadas sobre a ferramenta.

4.1 A Necessidade de Simulação

Formalmente, simulação pode ser definida como (SHANNON, 1975):

> "... o processo de projetar um modelo de um sistema real e de conduzir experimentos com este modelo tendo o objetivo de compreender o comportamento do sistema ou de avaliar diversas estratégias (dentro de limites impostos por um critério ou conjunto de critérios) de operação do mesmo"

A partir desta definição é possível compreender suas principais vantagens:

- possibilidade de observar o comportamento, a sensibilidade e o melhor ajuste dos parâmetros de interesse do sistema, sem a necessidade de construir ou alterar configurações em ambientes reais;
- viabilidade para estudo do sistema em diferentes níveis de abstração através do emprego de técnicas de modelagem *top-down*, por exemplo.
- maior controle sobre os componentes envolvidos viabilizando a compreensão das relações e interferências existentes.

Apesar destas vantagens destacadas, a técnica de simulação apresenta limitações como: a construção de um bom simulador é tão ou mais complexa do que o sistema sendo simulado, e constituir uma técnica intrinsecamente imprecisa[1], já que os modelos são abstrações de sistemas reais.

[1] dada a subjetividade envolvida, a obtenção deste grau de imprecisão é extremamente difícil

A necessidade de um modelo flexível, aliada aos custos e às dificuldades envolvidos na construção e na análise de um sistema real, foram decisivas para escolha de um simulador consagrado no âmbito da pesquisa em redes computacionais - *Network Simulator - Version 2 - NS-2* (NS2, 2003a) (BAJAJ et al., 1999).

4.2 Network Simulator (NS-2)

NS é um simulador orientado a eventos discretos voltado para pesquisa em redes de computadores. Fornece um suporte substancial para simulação de protocolos de transporte, aplicação e roteamento sobre diversas configurações de redes: com ou sem fio, locais, com uso de satélites, etc.

Atualmente, o desenvolvimento do NS é apoiado pela DARPA (*Defense Advanced Research Projects Agency*) através do projeto SAMAN (*Simulation Augmented by Measurement and Analysis for Networks*) (SAMAN, 2003) e pela NSF - *National Science Foundation* com o projeto CONSER (*Collaborative Simulation for Education and Research*), ambos em colaboração com diversos outros pesquisadores incluindo o instituto ICIR (The ICSI Center for Internet Research) (ICIR, 2003).

Desenvolvido sob o paradigma de programação orientada a objetos, proporciona uma melhor reutilização de código que, agregada à natureza *open source* do projeto, estimula seu crescimento através da contribuição de terceiros. Novas classes ou módulos podem ser adicionados pela extensão da hierarquia de classes atual (NS2, 2003b).

O simulador é escrito em C++ e utiliza OTcl (WETHERALL; LINDBLAD, 1995) como interface de comandos e de configuração. As razões básicas para o emprego destas duas linguagens são que:

- linguagens compiladas (C++) são adequadas para eficiente manipulação de *bytes*, cabeçalhos de pacotes e para implementação de algoritmos que rodam sobre grande conjuntos de dados. Para estas tarefas, o tempo de execução é importante;

- linguagens do tipo *script* (OTcl) são mais adequadas para aquelas tarefas que são executadas apenas uma vez durante a simulação, mas precisam sofrer ajustes freqüentes (parâmetros de entrada e topologia, por exemplo). Para estas tarefas, um baixo tempo de reconfiguração é essencial.

A utilização do simulador propriamente dita é feita através de *scripts* OTcl. A construção de determinado experimento envolve a criação e a execução de um arquivo com extensão ".tcl", como o exemplo apresentado na figura 4.2.

Um script de simulação é responsável basicamente por:

- definir uma topologia de rede;

- estabelecer um padrão de tráfego para a rede modelada;

- coletar resultados;

Os resultados das simulações são geralmente gravados em arquivos para processamento posterior. Uma das opções disponíveis é a gravação de um arquivo com formato específico para o visualizador *Nam* (*Network Animator*) (NAM, 2003) que permite observar de forma gráfica o andamento da simulação.

No NS apenas um evento pode estar em execução em um determinado instante (*single-thread*). Caso dois ou mais eventos estejam agendados para o mesmo momento, eles são

disparados segundo a política FIFO - *First In First Out*. Não há suporte à execução parcial, nem à preempção.

Os elementos fundamentais de modelagem são os nós (*nodes*) e elos de ligação (*links*). Estas duas figuras possibilitam a representação de topologias de redes arbitrárias. Os nós são definidos através de alguns atributos básicos (Figura 4.1): um endereço, lista de vizinhos, lista de agentes (responsáveis pela implementação dos protocolos), um identificador do tipo do nó e por um módulo de roteamento.

Fonte: Manual NS – pág. 39

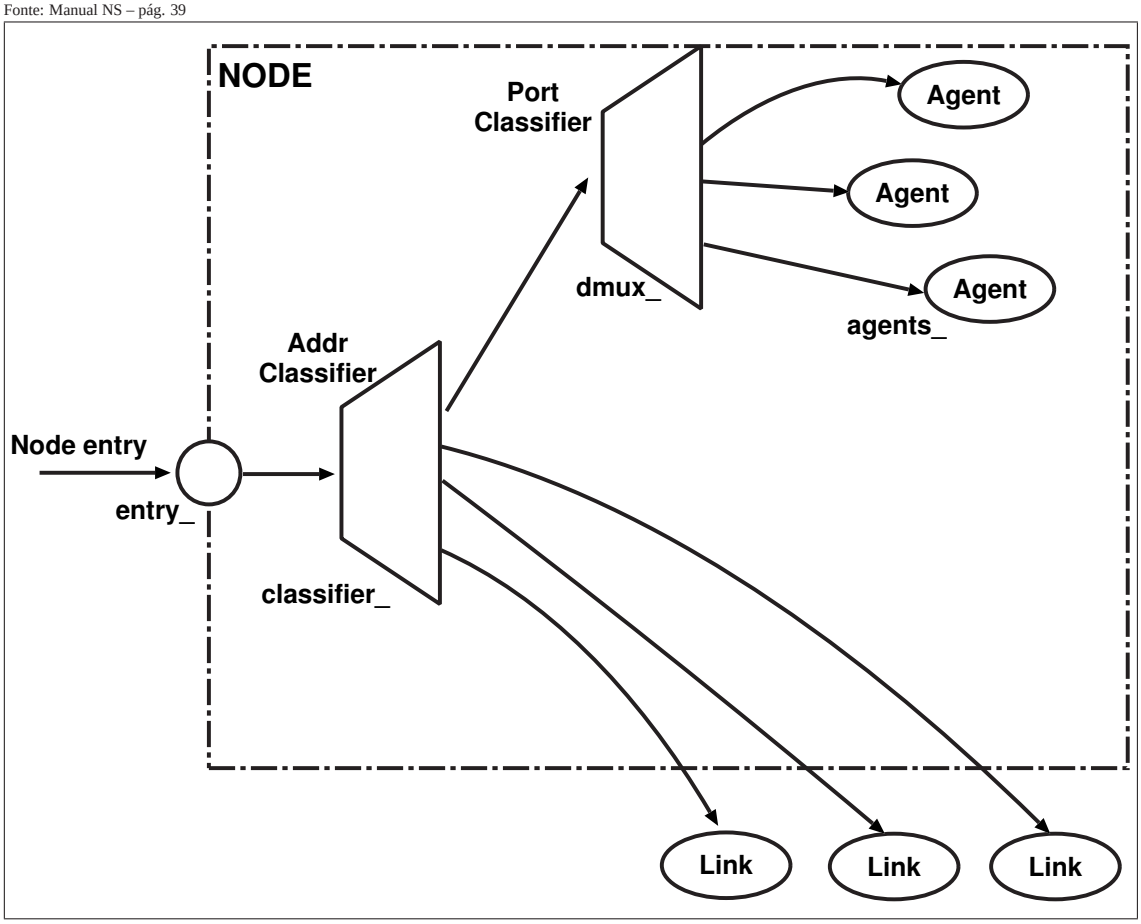


Figura 4.1: Estrutura básica de um nó no simulador

Os elos interligam os nós, modelam suas interfaces de saída e as características das linhas de transmissão. O enfileiramento de pacotes no NS está integrado aos elos que podem possuir uma ou mais filas.

Diversas políticas de filas e de escalonamento podem ser configuradas como *Droptail*, *Random Early Detection* (RED), entre outras. A perda de pacotes se dá pelo estouro da capacidade das filas ou através da configuração de modelos de falhas, associados aos canais durante as simulações. Há ainda previsto o suporte para a emulação, ou seja, o simulador pode ser integrado com uma rede real para injetar ou coletar tráfego.

As figuras 4.2 e 4.3 ilustram um script simples de entrada para o simulador e o respectivo cenário modelado.

```
#Cria um objeto simulador
set ns [new Simulator]

#Cria quatro nós
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Estabelece as ligacoes entre os nós
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

#Ajusta em 10 o tamanho da fila no elo (n2-n3)
$ns queue-limit $n2 $n3 10

#Cria uma conexão TCP
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink

#Estabelece FTP sobre a conexão TCP
set ftp [new Application/FTP]
$ftp attach-agent $tcp
```

```
# continuação ...

#Cria uma conexão UDP
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null

#Tráfego CBR sobre UDP
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set packet_size_ 1000
$cbr set rate_ 1mb

#Escalona eventos para os agentes CBR e FTP
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"

#Escalona final de simulação
$ns at 5.0 "exit 0"

#Inicia simulação
$ns run
```

Figura 4.2: Exemplo de script Otcl - NS2

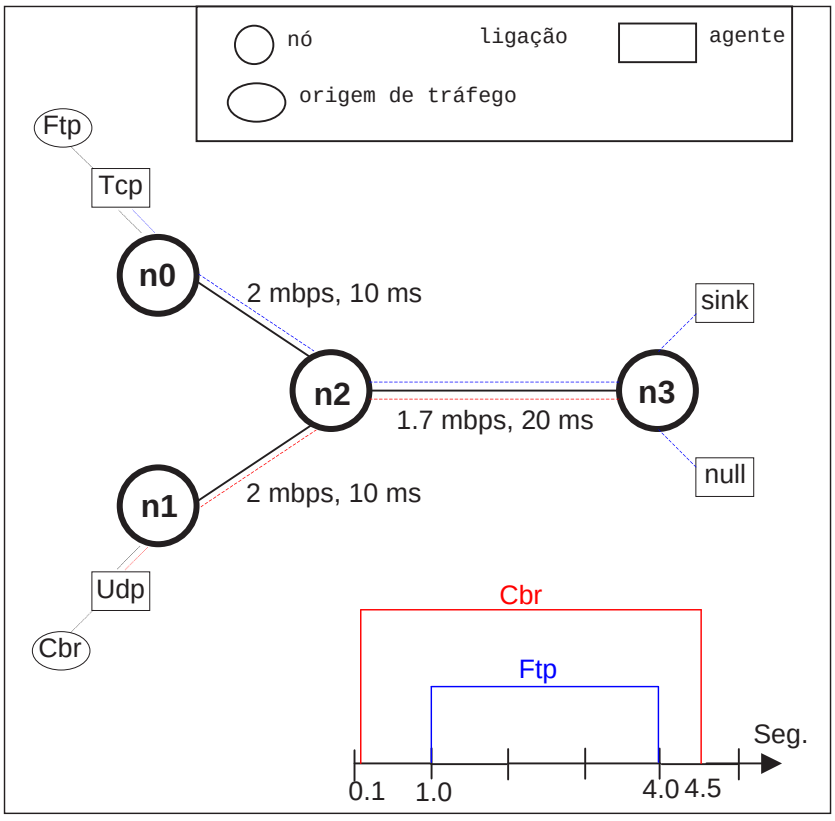


Figura 4.3: Cenário modelado pelo script da figura 4.2

4.3 Suporte no Simulador aos Mecanismos Previstos nas Normas IEEE

Não existe no simulador NS suporte para modelagem de redes locais virtuais (VLANs). Como não há previsão para a sinalização de prioridades, em uma rede local simulada todos os quadros recebem o mesmo tratamento.

Esta limitação não impede o estudo dos dispositivos introduzidos nas normas IEEE. É possível simular o comportamento desta classificação de tráfego no nível de enlace através das funcionalidades de serviços diferenciados (*Diffserv*) previstos no nível de rede e implementadas em um módulo do simulador (PIEDA et al., 2000). A adoção deste procedimento é justificada pela flexibilidade, facilidade de uso e vocação científica do simulador, mas requer o ataque a dois pontos principais:

- modelagem do ambiente local de interesse através de nós isolados e de conexões independentes, uma vez que a implementação de rede local disponível no simulador não satisfaz os requisitos necessários.

- certificação de que a implementação do módulo de serviços diferenciados disponível no simulador é plenamente compatível com o modelo de diferenciação introduzido pelos padrões IEEE 802.1D/Q.

O objeto de interesse da análise são as redes locais tipo *ethernet* totalmente baseadas em *switches* - *Switched Ethernet*. Estas redes são formadas por uma série de conexões ponto-a-ponto entre estações e comutadores. A figura de um barramento compartilhado não existe, pois cada estação possui canais independentes tanto para transmissão quanto para recepção de mensagens.

Não há problemas para construção de uma topologia desta natureza a partir dos elementos básicos disponíveis no simulador: nós e *links*. *Switches* de inúmeras portas podem ser facilmente modelados através de nós com várias ligações de entrada e saída.

Já a correta percepção da compatibilidade entre os modelos de qualidade de serviço previstos nos níveis de enlace e de rede requer uma discussão mais detalhada, com a apresentação das principais características do módulo disponível no NS, conforme seções seguintes.

4.4 Módulo de Serviços Diferenciados Disponível no Simulador

A arquitetura de Serviços Diferenciados (*DiffServ*) (BLAKE et al., 1998) é um mecanismo IP de qualidade de serviço baseado na marcação de pacotes e na conseqüente diferenciação entre categorias de tráfego, que podem agregar diversos fluxos independentes. As classes passam a receber tratamento distinto da rede através de níveis de prioridade relativa, definidos pelos usuários em função de requisitos específicos das aplicações.

O comportamento da rede é configurado através da implementação de regras que definem qual a hierarquia de prioridades a ser obedecida pelas diversas entidades no desempenho de funções de escalonamento e/ou descarte de pacotes. O tratamento diferenciado se evidencia, por exemplo, em momentos de congestionamento quando um número maior de pacotes com baixa prioridade é descartado em relação a pacotes de maior prioridade.

O NS possui um módulo *DiffServ* desenvolvido com base nesta arquitetura e implementado através da inserção de uma nova classe de filas (*dsRed – Differentiated Services RED*). Seu funcionamento está fundamentado na sinalização da categoria de cada pacote

através de um código (*codepoint*), inserido em seu cabeçalho IP. Com base nesta diferenciação de pacotes, podem ser estabelecidas regras de comportamento que definam o tratamento dispensado a mensagens de classes distintas.

A política de filas utilizada como padrão é a RED, com uma fila distinta para cada classe de pacotes configurada na rede. Cabe ao usuário, através do ajuste de diversos parâmetros durante a montagem dos *scripts* de simulação, definir as classes existentes e a hierarquia de prioridades a ser obedecida. Pode-se escolher qual algoritmo adotado para o escalonamento de pacotes entre as várias filas, além de alterar o método de gerenciamento interno de cada uma delas.

A idéia é dispensar melhor tratamento a determinadas classes de pacotes através de uma marcação que resulte no redirecionamento destes para filas com prioridade de escalonamento superior às demais.

O módulo *DiffServ* do NS é construído sobre três componentes básicos:

- Política - definição sobre qual nível de serviço determinada categoria de tráfego deve receber por parte da rede;
- Roteador de Borda - responsável pelo policiamento e medição de tráfego, além da marcação dos pacotes com o *codepoint* correspondente a política especificada; e
- Roteador de Núcleo - encarregado de inspecionar o código carregado em cada pacote e encaminhá-lo de acordo.

No simulador, o policiamento e a conformação do tráfego de acordo com taxas pré-estabelecidas para chegada de mensagens, como para a condição normal ou de pico, é parte integrante da definição das políticas de tráfego. Entre os algoritmos disponíveis estão “Token Bucket”, “TSW2CM” e “TSW3CM” (FANG; SEDDIGH; NANDY, 2000). A partir da aplicação destes algoritmos é possível rebaixar a categoria ou descartar quadros que estejam fora dos padrões definidos para determinado fluxo.

4.4.1 Exemplo Comentado

O *script* de simulação apresentado a seguir exemplifica o uso e expõe a forma de configuração do módulo de serviços diferenciados do NS. Uma análise mais detalhada está disponível em (ANDREOZZI, 2001).

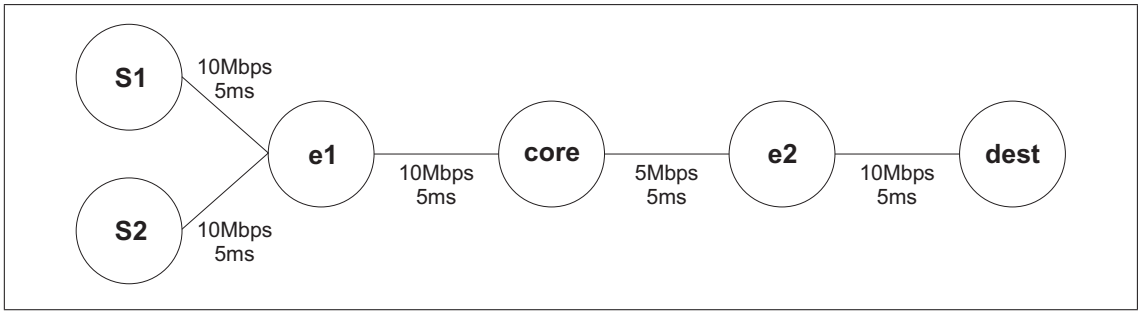


Figura 4.4: Topologia modelada

```
set ns [new Simulator]
set cir0 1000000
set cbs0 3000
set rate0 2000000
```

```
set cir1 1000000
set cbs1 10000
set rate1 3000000
set testTime 85.0
set packetSize 1000
```

Trecho responsável por declarar uma instância do simulador e definir alguns valores utilizados no decorrer do *script*.

```
set s1 [$ns node]
set s2 [$ns node]
set e1 [$ns node]
set core [$ns node]
set e2 [$ns node]
set dest [$ns node]

$ns duplex-link $s1 $e1 10Mb 5ms DropTail
$ns duplex-link $s2 $e1 10Mb 5ms DropTail
$ns simplex-link $e1 $core 10Mb 5ms dsRED/edge
$ns simplex-link $core $e1 10Mb 5ms dsRED/core
$ns simplex-link $core $e2 5Mb 5ms dsRED/core
$ns simplex-link $e2 $core 5Mb 5ms dsRED/edge
$ns duplex-link $e2 $dest 10Mb 5ms DropTail
```

Criação da topologia (Figura 4.4). Note-se a utilização da classe dsRED na configuração das filas dos canais de ligação entre entidades de borda, responsáveis pela marcação de tráfego, e de núcleo, responsável pelo repasse diferenciado.

```
set qE1C [[$ns link $e1 $core] queue]
set qE2C [[$ns link $e2 $core] queue]
set qCE1 [[$ns link $core $e1] queue]
set qCE2 [[$ns link $core $e2] queue]
```

Atribuição de um nome (qE1C, qE2C, . . .) para as filas dos canais de tráfego diferenciado com o objetivo de facilitar referências futuras.

```
# Parametros DSRed de Edge1 para Core:

$qE1C set numQueues_ 2          ;#Numero de filas/classes (2)
$qE1C set setSchedularMode PRI  ;#escalonamento de prioridade absoluta entre as filas
$qE1C setMREDMode DropTail      ;#As filas devem ser comportar individualmente como FIFO
$qE1C addPolicyEntry [$s1 id] [$dest id] TokenBucket 10 $cir0 $cbs0
$qE1C addPolicyEntry [$s2 id] [$dest id] TokenBucket 11 $cir1 $cbs1
$qE1C addPolicerEntry TokenBucket 10 11
$qE1C addPolicerEntry TokenBucket 11 11
$qE1C addPHBEntry 10 0 0
$qE1C addPHBEntry 11 1 0
$qE1C configQ 0 0 20            ;#fila 0 com tamanho máximo de 20 pacotes
$qE1C configQ 1 0 10            ;#fila 1 com tamanho máximo de 10 pacotes
```

Bloco de configuração dos parâmetros para a marcação efetuada em *Edge1*. Pacotes originados em *s1* recebem o marcador “10”, enquanto pacotes com origem em *s2* são identificados com o valor “11”. O algoritmo “*TokenBucket*” é utilizado para policiamento e marcação.

Uma entrada “*addPolicerEntry*” é exigida para cada par (algoritmo de policiamento, marcador) e define a seqüência de rebaixamento de classes dos pacotes. No exemplo em questão, pacotes de código “10” que excedam as taxas pré-estabelecidas são rebaixados de classe (“11”).

As linhas com o comando “addPHBEntry” indicam o mapeamento entre códigos de pacotes e filas. Pacotes com código 10 são colocados na fila de número 0 (prioritária) e

pacotes com código 11 na fila de número 1. Quanto menor o número da fila, maior sua prioridade no processo de escalonamento de pacotes para transmissão.

O comando “configQ” é responsável por ajustar os parâmetros das filas. Em filas “*DropTail*”, o único parâmetro ajustável é o tamanho máximo do enfileiramento, determinado pelo terceiro argumento do comando “ConfigQ”.

```
# Parametros DSRed de Edge2 para Core:

$qE2C set numQueues_ 2          ;#Numero de filas/classes (2)
$qE2C set setSchedularMode PRI  ;#escalonamento de prioridade absoluta entre as filas
$qE2C setMREDMode DropTail      ;#As filas devem ser comportar individualmente como FIFO
$qE2C addPolicyEntry [$dest id] [$s1 id] TokenBucket 10 $cir0 $cbs0
$qE2C addPolicyEntry [$dest id] [$s2 id] TokenBucket 11 $cir1 $cbs1
$qE2C addPolicerEntry TokenBucket 10 11
$qE2C addPolicerEntry TokenBucket 11 11
$qE2C addPHBEntry 10 0 0
$qE2C addPHBEntry 11 1 0
$qE2C configQ 0 0 20            ;#fila 0 com tamanho máximo de 20 pacotes
$qE2C configQ 1 0 10            ;#fila 1 com tamanho máximo de 10 pacotes
```

O trecho de configuração dos parâmetros de diferenciação de tráfego em *edge2* é muito semelhante ao bloco anterior. A única diferença está na inversão no sentido de marcação do tráfego.

```
# Core -> Edge1:
$qCE1 set numQueues_ 2
$qCE1 set setSchedularMode PRI  ;#escalonamento de prioridade absoluta entre as filas
$qCE1 setMREDMode DropTail      ;#As filas devem ser comportar individualmente como FIFO
$qCE1 addPHBEntry 10 0 0
$qCE1 addPHBEntry 11 0 1
$qCE1 configQ 0 0 20
$qCE1 configQ 1 0 10

# Core -> Edge2:
$qCE2 set numQueues_ 2
$qCE2 set setSchedularMode PRI  ;#escalonamento de prioridade absoluta entre as filas
$qCE2 setMREDMode DropTail      ;#As filas devem ser comportar individualmente como FIFO
$qCE2 addPHBEntry 10 0 0
$qCE2 addPHBEntry 11 0 1
$qCE2 configQ 0 0 20
$qCE2 configQ 1 0 10
```

Configuração de repasse de quadros na entidade central da rede conforme classificação definida em “edge1” e “edge2”. Deve-se configurar um entrada para cada código de marcação possível, “10” ou “11” no caso.

```
# Set up one CBR connection between each source and the
destination:
set cbr0 [new Agent/CBR]
$ns attach-agent $s1 $cbr0
$cbr0 set packetSize_ $packetSize
$cbr0 set interval_ [expr 1.0 / [expr $rate0 / 8000.0]]
set null0 [new Agent/Null]
$ns attach-agent $dest $null0
$ns connect $cbr0 $null0
set cbr1 [new Agent/CBR]
$ns attach-agent $s2 $cbr1
$cbr1 set packetSize_ $packetSize
$cbr1 set interval_ [expr 1.0 / [expr $rate1 / 8000.0]]
set null1 [new Agent/Null]
$ns attach-agent $dest $null1
$ns connect $cbr1 $null1

proc finish {} {
global ns
exit 0
}
```

```
$qE1C printPolicyTable
$qE1C printPolicerTable
$ns at 0.0 "$cbr0 start"
$ns at 0.0 "$cbr1 start"
$ns at 20.0 "$qCE2 printCoreStats"
$ns at 40.0 "$qCE2 printCoreStats"
$ns at 60.0 "$qCE2 printCoreStats"
$ns at 80.0 "$qCE2 printCoreStats"
$ns at $testTime "$cbr0 stop"
$ns at $testTime "$cbr1 stop"
$ns at [expr $testTime + 1.0] "finish"
$ns run
```

Trecho final encarregado de inserir tráfego na rede e comandar início e fim da simulação.

4.5 Compatibilidade Funcional entre os Modelos

Os dois sistemas de QoS apresentados, IEEE 802.1Q/D e *Diffserv* IP, são funcionalmente coincidentes. Apesar de definidos em níveis distintos da arquitetura de rede, derivam de um mesmo conceito teórico baseado na marcação de tráfego e na utilização de prioridades relativas. Neste modelo único, atribuem-se prioridades a conjuntos de pacotes; e as entidades ao longo do caminho aplicam um método de repasse correspondente ao valor sinalizado, geralmente em alguma porção do cabeçalho das mensagens.

Na verdade, e exatamente devido ao fato de serem implementados em camadas distintas, a única diferença fundamental entre os dois sistemas está relacionada com o tempo de processamento dos pacotes. Quanto mais elevado o nível da arquitetura de rede, mais cabeçalhos precisam ser interpretados e portanto, maior o tempo gasto com o encaminhamento e repasse de mensagens.

Em uma rede real, quatro componentes formam o atraso fim-a-fim observado por determinado pacote (Figura 3.3):

- o tempo de processamento: tempo necessário para processar um pacote em cada nó da rede;
- tempo de transmissão: tempo gasto com a inserção do pacote na linha de transmissão (tamanho do pacote / taxa do canal);
- tempo de propagação: tempo que determinado pacote leva para percorrer a linha;
- tempo de enfileiramento: tempo de espera que um pacote experimenta nas filas encontradas ao longo do caminho;

Conforme discutido no capítulo anterior, a fração correspondente ao tempo de processamento é geralmente desprezada durante as análises. Trata-se de um tempo basicamente fixo e de extrema dependência dos equipamentos e sistemas de *software* específicos empregados. Isto tanto é verdade que não existe suporte a sua modelagem no simulador.

Assim, devido à liberdade de configuração característica do simulador, é possível utilizar o módulo *Diffserv* disponível para simular o mesmo comportamento e organização das filas previstos nas normas IEEE. Os resultados obtidos com simulações, efetuadas no nível de rede sobre uma topologia baseada em ligações ponto-a-ponto entre os nós, são compatíveis com aqueles obtidos em uma rede local *ethernet* baseada em *switches*.

As únicas filas existentes são as filas de roteamento, que possuem exatamente o mesmo comportamento das filas formadas em comutadores compatíveis com as especificações 802.1D/Q.

Entre os ajustes necessários é preciso estabelecer que a política utilizada no gerenciamento individual das filas de cada classe de tráfego seja o descarte simples com algoritmo FIFO (*First In First Out*). No simulador este comportamento está disponível sob a identificação "*DropTail*".

Além disso, o escalonamento entre as diversas filas deve ser definido como de prioridade absoluta – escalonador "PRI" no NS. A exemplo do previsto na norma IEEE 802.1D, um pacote de determinada fila só será escolhido para transmissão caso não existam mensagens enfileiradas nas filas de maior prioridade.

4.6 Modelagem dos Cenários de Interesse

A representação no simulador de uma configuração qualquer de rede requer o ataque a dois aspectos fundamentais de modelagem: topologia (disposição física) e perfil de tráfego empregado (característico das aplicações).

4.6.1 Topologia

A representação no NS das topologias consideradas nos dois casos de estudo introduzidos no capítulo anterior não apresenta dificuldade. Entretanto, algumas decisões tomadas quando da modelagem dos cenários de interesse merecem destaque.

Em primeiro lugar há de se observar que o módulo de serviços diferenciados disponível no simulador prevê a existência dos chamados roteadores de borda. O papel desempenhado por estas entidades é a marcação do tráfego que ingressa na rede. Num ambiente local, este conceito não faz muito sentido. A tarefa de diferenciação de tráfego deve ser desempenhada pelos comutadores ou pelas próprias estações através da emissão de quadros que já carreguem a informação de prioridade em seus cabeçalhos.

Assim, optou-se por modelar o comportamento de cada estação da rede através de dois nós do simulador. Cabe ao primeiro nó gerar os quadros e ao segundo marcá-los, desempenhando a função do roteador de borda prevista no módulo *Diffserv* do NS. Como cada estação é composta por dois nós e responsável pela marcação exclusiva de seu próprio tráfego, não existe um ponto de concentração inicial sobre o qual os fluxos devam ser submetidos. As figuras 4.5 e 4.6 ilustram o modelo básico de cada estação da rede e os comandos utilizados no simulador para sua construção, respectivamente.

A ligação entre os dois nós internos que modelam uma estação tem os parâmetros de taxa do canal e de tempo de propagação ajustados para valores extremos, com objetivo de evitar sua influência no atraso total observado durante as simulações. Isto é necessário por tratar-se de um processamento interno às estações que, a exemplo dos demais tempos de processamento envolvidos, foi desconsiderado durante os experimentos.

Outro fator importante é que a adequação ao modelo QoS de nível de enlace exige que os parâmetros que regulam os mecanismos de policiamento e conformação sejam superdimensionados durante as simulações. Desta forma, a classificação de tráfego é mantida constante sem possibilidade de alteração na classe de determinado pacote ou de perda de mensagens em outra situação, senão por excesso nas filas dos comutadores.

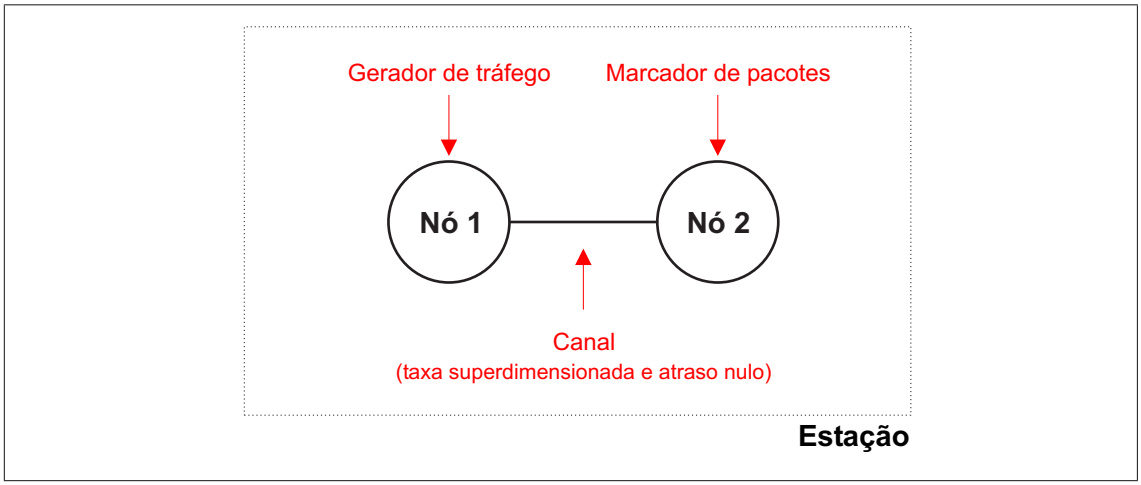


Figura 4.5: Modelo adotado para estações da rede de controle

```
$ns duplex-link $n(estacao0_no1) $n(estacao0_no2) 10000000000 0.000000001 DropTail
$ns duplex-link $n(estacao0_no2) $n(switch) 100Mb 0.0000005 dsRed/edge
set fila(estacao0_no1-switch) [[$ns link $n(estacao0_no1) $n(switch)] queue]
```

Figura 4.6: Código de modelagem de uma estação no simulador

4.6.2 Tráfego

Uma das premissas adotadas durante a formulação do modelo de cálculo apresentado no capítulo 3 está relacionada com a transmissão de quadros por parte das entidades pertencentes à rede de controle. O comportamento periódico assumido se reflete na transmissão ao controlador de quadros com tamanho fixo, segundo intervalos regulares de tempo.

Este comportamento está implementado no simulador através do objeto *CBR - Constant Bit Rate*, instanciado a partir da classe "*Application/Traffic/CBR*". A figura 4.2 ilustra a utilização deste gerador de tráfego.

Os geradores de padrões de tráfego disponíveis no NS (*Exponential*, *Paretto*, CBR, ...) modelam o comportamento do nível de aplicação. Seu uso efetivo está condicionado com a amarração a um agente de transporte – caso do agente *UDP* empregado sob o tráfego *CBR* no exemplo da figura 4.2.

Além das estações que integram a rede de controle, também os fluxos pertencentes a rede de dados foram modelados através de tráfego *CBR*. Esta não é a abordagem mais correta, pois não representa um perfil realista de tráfego em redes comuns. Entretanto, ela foi adotada porque não há interesse na análise temporal dos fluxos de dados.

A inserção destes fluxos não prioritários nas simulações atende dois objetivos principais:

- constituir um mecanismo de controle que permita aumentar ou diminuir a carga total da rede. Pois uma das finalidades da análise é observar o atraso total de quadros prioritários quando a rede se encontra em um patamar bem elevado de utilização;
- disputar com quadros de prioridade mais elevada canais compartilhados por classes diferentes. Deseja-se possibilitar nos experimentos efetuados a ocorrência do fenômeno de inversão de prioridade discutido na seção 3.3. Por esta razão, e também

em busca do pior cenário possível, os quadros inseridos na rede de dados possuem o maior tamanho permitido (1530 bytes).

Estes dois objetivos são plenamente atingidos através de taxas constantes de transmissão (CBR). Assim, todo tráfego presente nas simulações é resultante da comunicação entre pares de estações e implementado por taxas fixas de emissão de mensagens.

4.7 Alterações Necessárias no Simulador - NS2

4.7.1 Gerador de Tráfego CBR

Os parâmetros de configuração para um objeto CBR no simulador são:

PacketSize_: tamanho constante do pacotes gerados (expresso em *bytes*).

rate_: Taxa de geração em pacotes/segundo.

interval_: Intervalo entre os pacotes. É necessário definir apenas um dos parâmetros *rate* e *interval_*, pois um é derivado do outro.

random_: controla a inserção ou não de ruído aleatório nos tempos de partida previstos para os pacotes. Como padrão este parâmetro está desligado, fazendo com que pacotes sejam gerados exatamente segundo o intervalo de tempo configurado.

maxpkts_: Número máximo de pacotes a enviar. Valor padrão é de 2^{28}.

Durante as simulações, o processo de envio de quadros deve ser iniciado separadamente para cada estação da rede (*$cbr start*). Mesmo que estes eventos sejam escalonados para o mesmo instante, uma ordem de início será obrigatoriamente estabelecida entre as diversas estações. O origem deste fato reside nas características do escalonador de eventos do NS, que atende eventos simultâneos segundo uma política FIFO.

Com a transmissão em intervalos rigorosamente regulares, esta ordem empregada se manterá durante todo o período do experimento, fazendo com que os quadros da rede de controle atinjam o comutador repetidas vezes sob a mesma seqüência. O tamanho da fila encontrada por quadros transmitidos de uma determinada estação será o mesmo a cada período de atividade.

Portanto, para tornar os resultados mais realistas, não é interessante que o envio dos quadros por parte das estações seja estritamente periódico. Deseja-se uma pequena variação nos intervalos de partida, tornando aleatória a ordem de envio assumida entre as estações a cada processo de transmissão.

O parâmetro *random_* do gerador de tráfego CBR se propõe exatamente a habilitar a inserção de ruído temporal entre intervalos consecutivos de envio. O problema é que esta variável do tipo liga/desliga não fornece nenhum controle sobre a quantidade de variação empregada aos tempos de partida dos pacotes.

A implementação em C++ da classe CBR está disponível no arquivo fonte "*cbr_traffic.cc*" da distribuição do simulador. Sua inspeção revela que a variação temporal v aplicada sobre os intervalos de geração de quadros, quando o parâmetro *random_* está habilitado, se traduz adição ou subtração de um valor entre os limites 0 e $\frac{interval\_}{2}$ ($t \pm v, \quad 0 \leq v \leq \frac{interval\_}{2}$) – conforme trecho de código apresentado na figura 4.7.

```
double CBR_Traffic::next_interval(int& size)
{
        // Recompute interval in case rate_ or size_ has changes
        interval_ = (double)(size_ « 3)/(double)rate_;
        double t = interval_;
        if (random_)
                t += interval_ * Random::uniform(-0.5, 0.5);
        size = size_;
        if (++seqno_ < maxpkts_)
                return(t);
        else
                return(-1);
}
```

Figura 4.7: Recálculo do intervalo de geração de quadros (CBR) – arquivo tools/cbr_traffic.cc

O ruído inserido é sorteado segundo uma distribuição uniforme entre os limites $-\frac{interval\_}{2}$ e $+\frac{interval\_}{2}$. O emprego de uma função de sorteio uniforme garante que os valores entre os extremos estabelecidos sejam equiprováveis.

Uma rede de controle geralmente pressupõe a ação sincronizada de seus participantes. Nos casos em estudo, isto significa que as transmissões de quadros pelas diversas estações devem ocorrer simultaneamente a cada período de amostragem, ou de forma mais realista, muito próximas no tempo.

O emprego de uma variação temporal que pode assumir no limite extremo valores correspondentes a metade do intervalo de geração dos quadros ($random\_ = 1$), significa o abandono de sincronismo entre as estações. Quadros oriundos dos diversos participantes estarão dispersos ao longo do tempo de um período de transmissão. Por outro lado, a opção para geração rigorosamente periódica de quadros (*random_ = 0*) também não é adequada, pois supõe sincronismo absoluto entre entidades distintas.

Desta forma, uma pequena alteração se fez necessária no simulador. O parâmetro *random_* do gerador CBR foi transformado em uma variável de controle da variação empregada sobre os intervalos entre pacotes consecutivos, conforme trecho apresentado na figura 4.8. Com esta modificação, o parâmetro *random_* passa a controlar a quantidade de ruído inserida, ao invés de simplesmente sinalizar (liga/desliga) a necessidade de variação nos intervalos de partida dos pacotes. O comportamento disponível na implementação original é mantido, pois:

- caso *random_* assuma um valor nulo, nenhum ruído é introduzido e a geração é absolutamente periódica; e,
- caso *random_* assuma o valor 1 o ruído inserido será aleatoriamente sorteado entre os valores de mesma probabilidade do intervalo $[-\frac{interval\_}{2}, +\frac{interval\_}{2}]$

Entretanto, com a nova implementação, *random_* pode assumir valores intermediários entre 0 e 1. Quanto mais próximo de 1 maior será o ruído introduzido. Como a modificação foi inserida na porção do simulador escrita em C++ foi necessário recompilar o código para que a alteração tivesse efeito e pudesse ser utilizada a partir dos *scripts* de simulação.

```
$ diff tools/cbr_traffic.cc tools/cbr_traffic.cc.original
86,91c86,87
<         if (random_)
<         {
<           if (random_ > 1)
<                 random_ = 1.0;
<           t += interval_ * Random::uniform(-random_/2.0, random_/2.0);
<       }
--
>       if (random_)
>               t += interval_ * Random::uniform(-0.5, 0.5);
```

Figura 4.8: Alteração implementada no gerador de tráfego CBR – NS

4.7.2 Precisão do Relatório de Eventos

Em geral, a abordagem adotada para análise dos resultados de experimentos efetuados no NS é o pós-processamento sobre os arquivos de saída do simulador. Estes arquivos (*trace files*) registram os eventos ocorridos durante as simulações: transmissão, recepção, descarte e enfileiramento de pacotes. Durante a montagem dos *scripts* de simulação pode-se escolher entre registrar todos eventos da rede simulada ou selecionar os canais de transmissão e tipos de ocorrências que merecem registro.

```
<código> <tempo> <no1> <no2> <pacote>

<código> := [r|d|+|-] r=receive, d=drop, +=enque -=dequeue;
<tempo>  := tempo da simulação em segundos;
<no1>    := nó atual do pacote
<no2>    := próximo nó para o qual o pacote será repassado;
<pacote> := <tipo> <tam> < flags> <f_id> <no_org>.<p_org> <no_dst>.<p_dst> <seq> <p_id>

<tipo>   := [cbr|tcp|ack|telnet|...];
<tam>    := tamanho do pacote em bytes;
<flags>  := [C|P] C=congestion, P=priority;
<f_id>   := identificador do fluxo conforme definido para IPV6;
<no_org> := nó que originou o pacote (camada de transporte);
<p_org>  := porta de origem do pacote (agente);
<no_dst> := nó destino final do pacote (camada de transporte);
<p_dst>  := porta de destino do pacote (agente);
<seq>    := número de seqüência do pacote;
<p_id>   := identificador único para cada novo pacote;
```

Figura 4.9: Formato dos registros nos relatórios de eventos – NS

Os relatórios são organizados em linhas cujo formato é apresentado na figura 4.9. Na configuração padrão do simulador, o campo **<tempo>**, fundamental para processamento e correta interpretação dos eventos, possui precisão de 6 casas decimais. Entretanto, muitos dos tempos envolvidos nos experimentos montados para validação do modelo de cálculo deduzido estão abaixo deste limite. Este é o caso do tempo de propagação dos quadros nas linhas de transmissão (100 metros), conforme equação 3.12.

Desta forma, a precisão dos relatórios gerados foi ajustada em 8 casas decimais – biblioteca de funções "*trace/basetrace.h*"da distribuição original do simulador. A figura 4.10 ilustra o ponto alterado. Novamente a modificação implementada exigiu à recompilação do código afetado.

```
$ diff trace/basetrace.h trace/basetrace.original
62c62
< #define PRECISION 1.0E+8

> #define PRECISION 1.0E+6
```

Figura 4.10: Ajuste na precisão dos relatórios de simulação – NS

5 EXPERIMENTOS E RESULTADOS

Este capítulo apresenta os resultados extraídos por simulação dos cenários propostos como estudos de caso no capítulo 3. Um paralelo comparativo é traçado entre os resultados teóricos previstos e aqueles obtidos na rede simulada. A metodologia utilizada durante a experimentação é esclarecida em conjunto com algumas decisões necessárias de modelagem.

5.1 Metodologia

A extração das medidas de interesse está condicionada a execução de no mínimo 10 replicações de cada experimento e se baseia no pós-processamento dos relatórios de eventos gerados pelas simulações. Para cada caso/topologia modelada, diversos experimentos são construídos através da variação de parâmetros da rede de controle, como número de estações e carga da rede de dados, procurando identificar possíveis influências sobre o tráfego prioritário.

Entre as principais medidas extraídas destacam-se:

Atraso na comunicação - $\mathbf{A_T}$: diferença temporal entre o momento de partida do pacote de sua estação origem e o instante de chegada ao destino. O valor obtido engloba todos tempos intermediários de injeção, transmissão e enfileiramento (equação 5.1).

$$A_T(origem, destino, x) = t_{chegada}(mesg(origem, destino, x)) - t_{partida}(mesg(origem, destino, x)) \tag{5.1}$$

onde a tupla $(origem, destino, x)$ identifica a mensagem de seqüência x no fluxo de comunicação entre a estações $origem$ e $destino$.

***Jitter*:** o módulo da diferença entre valores de atraso observados por dois pacotes consecutivos de um mesmo fluxo, ou seja, de um mesmo par de estações.

$$Jitter = \mid A_T(origem, destino, x)) - A_T(origem, destino, x+1) \mid \tag{5.2}$$

É importante observar que a definição de *Jitter* impede a influência do ruído inserido na geração do tráfego CBR (seção 4.7.1). Isto ocorre porque o cálculo considera a variação entre os valores de atraso observados por dois pacotes consecutivos e não a diferença direta entre tempos de chegada de duas mensagens em seqüência. Desta forma, o tempo

de partida de cada quadro é descontado, eliminando sua influência e captando a variação causada exclusivamente pelas condições e comportamento da rede simulada.

Todas medidas foram extraídas em seus valores mínimos, médios e máximos, individualmente para cada fluxo de comunicação. Os resultados obtidos para a rede de controle foram ao final consolidados, agregando valores de todos micro-fluxos existentes (estação$(n) \rightarrow controlador, \quad 0 < n < N$). Nesta consolidação, os valores médios são obtidos para cada experimento através da média aritmética de todas replicações e de todos fluxos de mesma prioridade na rede. Já os limites mínimos e máximos das medidas apresentadas são reflexos de valores pontuais observados em algum ponto da simulação e selecionados por comparação direta com todos demais.

5.2 Caso 1: N Estações x 1 *Switch* - 2 Prioridades

A modelagem deste cenário tem por base o estudo de caso apresentado no capítulo 3, e por finalidade, a confrontação com os resultados teóricos previstos. Portanto, a topologia considerada e os parâmetros da rede de controle são aqueles definidos na figura 3.4 e na tabela 3.1, respectivamente.

Durante os experimentos, a rede de dados é formada por 10 estações organizadas em 5 pares transmissores/receptores. O tamanho de quadro adotado é estabelecido no máximo de 1530 *bytes* permitido para rede. A tabela 5.1 resume as principais características deste tráfego não prioritário incluído nas simulações.

Tabela 5.1: Parâmetros principais da rede de dados – Caso 1

Descrição	**Valor**
Número de entidades	10 estações
Taxa constante de geração de quadros	5000 quadros/s
Tamanho dos quadros	12240 bits

A figura 5.1 apresenta a definição dos principais parâmetros de cada rodada das simulações, disposta na porção inicial dos *scripts* utilizados (Anexo A).

```
set opt(rede,controle) 15          ;#nro_estacoes rede de controle
set opt(tam_pacote,controle) 72    ;# menor quadro possivel - IEEE 802.3
set opt(periodo,controle) 0.001    ;# 1000 quadros/segundo
set opt(codepoint,controle) 10     ;# código utilizado para marcação dos pacotes de controle

set opt(rede,dados) 10             ;#nro_estacoes rede de dados
set opt(tam_pacote,dados) 1530     ;# maior quadro possível - IEEE 802.3
set opt(periodo,dados) 0.0002      ;# 5000 quadros/segundo
set opt(codepoint,dados) 0         ;# código utilizado para marcação dos pacotes de dados
```

Figura 5.1: Definição dos parâmetros de simulação (Caso 1)

Na montagem dos *scripts* de simulação, o policiamento de tráfego é tornado sem efeito através do superdimensionamento de seus parâmetros. O trecho apresentado na figura 5.2

ilustra este procedimento. As linhas 1 e 2 do trecho de código exposto estabelecem o algoritmo TSW2CM para policiamento do tráfego de controle (marcador 10) e de dados (marcador 0) entre uma estação qualquer (est) e o *switch*. O superdimensionamento se reflete na taxa de pico estabelecida em $1Gbps$ - valor que jamais será atingido pela própria limitação dos canais de transmissão $100Mbps$. As linhas 3 e 4 são exigidas pelo simulador, mas não têm efeito prático por definirem um novo marcador (20) a ser colocado em pacotes que ultrapassem a taxa de pico definida; o que é impossível.

```
1· $fila(est-switch) addPolicyEntry -1 [$n(controlador) id] TSW2CM 10 1000000000
2· $fila(est-switch) addPolicyEntry -1 -1 TSW2CM 0 1000000000
3· $fila(est-switch) addPolicerEntry TSW2CM 10 20
4· $fila(est-switch) addPolicerEntry TSW2CM 0 20
```

Figura 5.2: Superdimensionamento nos parâmetros do algoritmo de policiamento de tráfego – *scripts* de simulação

A tabela 5.2 apresenta a comparação entre os valores teóricos apresentados na seção 3.3.1 e aqueles obtidos por simulação. Todos resultados apresentados refletem medidas sobre variáveis da rede de controle. Não foram efetuadas análises sobre os fluxos de dados, uma vez que sua modelagem não corresponde a um perfil real e adequado de tráfego (CBR). Conforme esclarecido anteriormente, o objetivo de sua inclusão é identificar os efeitos causados sobre as comunicações prioritárias.

Tabela 5.2: Resultados dos experimentos $\times$ valores teóricos – Caso 1 ($p0$)

Estações	**Resultados Teóricos**			**Resultados Simulação**		
	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
5	0,01252	0,02596	0,03940	0,01252	0,02038	0,03553
10	0,01252	0,03940	0,07300	0,01252	0,03186	0,06427
15	0,01252	0,05956	0,10660	0,01252	0,05614	0,09302
30	0,01252	0,10660	0,20740	0,01252	0,09041	0,17940
50	0,01252	0,17380	0,34180	0,01252	0,12605	0,29462
70	0,01252	0,24100	0,47620	0,01252	0,21910	0,40963
90	0,01252	0,30820	0,61060	0,01252	0,32846	0,52499
110	0,01252	0,37540	0,74500	0,01252	0,31088	0,64017
130	0,01252	0,44260	0,87940	0,01252	0,42729	0,75444
150	0,01252	0,50980	1,01380	0,01252	0,44653	0,87011

Obs: *valores de A_T em milissegundos (ms)*

Os valores obtidos por meio de simulação se aproximam bastante daqueles derivados da aplicação direta do modelo de análise temporal. A correspondência exata no caso de $A_{T_{min}}$ indica a correta modelagem da topologia de rede, pois na ausência de filas, os únicos tempos envolvidos são aqueles gastos com a injeção e propagação das mensagens. Entretanto, a medida que mais estações são colocadas na rede de controle o valor extraído por experimentação, sempre menor, se afasta do limite teórico tanto para $A_{T_{med}}$ quanto para $A_{T_{max}}$.

Este comportamento é explicado por um fato negligenciado até aqui. As simulações são efetuadas sobre conexões ponto-a-ponto entre nós e através de tráfego CBR. Não é utilizada a figura de rede local disponível no simulador e, portanto, o protocolo CSMA/CD não é respeitado durante a execução dos *scripts* de simulação. A conseqüência direta, que influi diretamente nos resultados obtidos, é o desrespeito ao tempo mínimo de intervalo exigido pelo protocolo para a transmissão de quadros consecutivos em determinado canal ($0,96\mu s \rightarrow 100Mbps$).

O que ocorre é que, durante as simulações, os quadros são enfileirados no comutador e, como possuem o mesmo destino no caso da rede de controle, são despachados um imediatamente ao outro no canal de ligação com o controlador.

A_T atinge seu valor máximo quando determinado pacote encontra a fila cheia. Neste caso, o tempo gasto apenas com o intervalo entre quadros consecutivos é igual ao número de mensagens a sua frente (estações $-$ 1), multiplicado pelo intervalo mínimo exigido entre transmissões consecutivas. De posse desta informação, é possível obter o tempo de correção a ser aplicado sobre o valor prático de $A_{T_{max}}$ calculado sobre uma rede de controle formada por N estações, conforme equação 5.3.

$$\begin{aligned} Tempo_{correcao}(A_{T_{max}}) &= (N-1) \times T_{EQ} \\ &= N \times 0,96\mu s \end{aligned} \tag{5.3}$$

Por raciocínio semelhante, o tempo de correção para $A_{T_{med}}$ é dado pela equação 5.4.

$$\begin{aligned} Tempo_{correcao}(A_{T_{med}}) &= (\frac{N-1}{2}) \times T_{EQ} \\ &= (\frac{N-1}{2}) \times 0,96\mu s \end{aligned} \tag{5.4}$$

A tabela 5.3 e a figura 5.3 apresentam os valores experimentais já corrigidos.

Tabela 5.3: Valores experimentais corrigidos – Caso 1 ($p0$)

Estações	**Resultados Teóricos**			**Resultados Simulação**		
	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
5	0,01252	0,02596	0,03940	0,01252	0,02230	0,03937
10	0,01252	0,03940	0,07300	0,01252	0,03618	0,07291
15	0,01252	0,05956	0,10660	0,01252	0,06286	0,10646
30	0,01252	0,10660	0,20740	0,01252	0,10433	0,20724
50	0,01252	0,17380	0,34180	0,01252	0,14957	0,34166
70	0,01252	0,24100	0,47620	0,01252	0,25222	0,47587
90	0,01252	0,30820	0,61060	0,01252	0,37118	0,61043
110	0,01252	0,37540	0,74500	0,01252	0,36320	0,74481
130	0,01252	0,44260	0,87940	0,01252	0,48921	0,87828
150	0,01252	0,50980	1,01380	0,01252	0,51805	1,01315

Obs: *valores de A_T em milissegundos (ms)*

Com esta pequena correção sobre os valores extraídos da rede simulada, os resultados são praticamente coincidentes com as previsões apresentadas no capítulo 3 (Figuras 5.4 e 5.5).

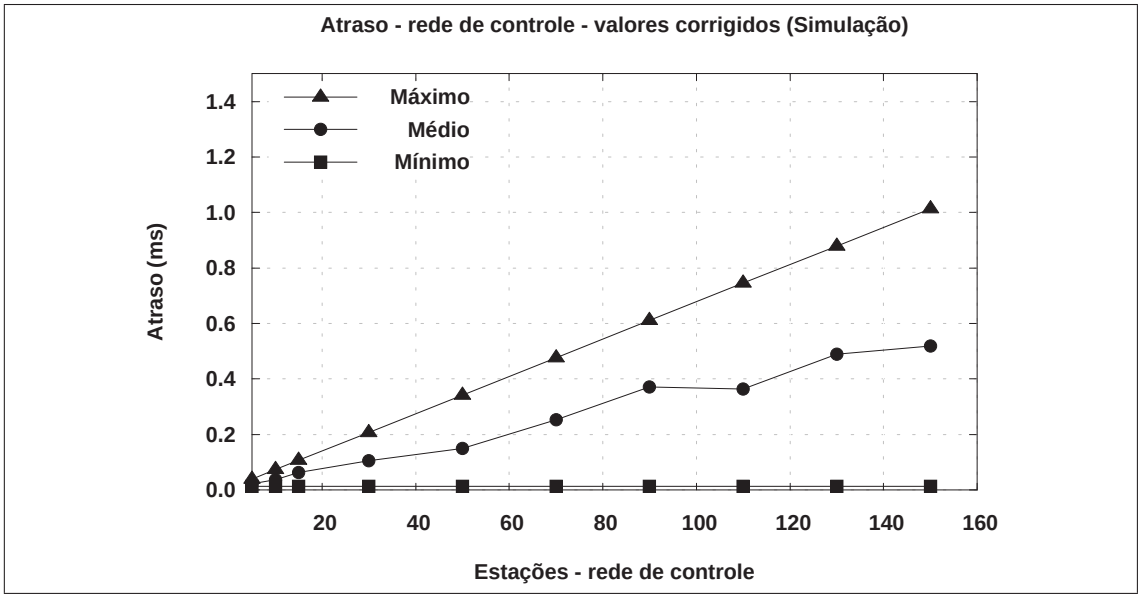


Figura 5.3: Valores obtidos por simulação – A_T (Caso 1)

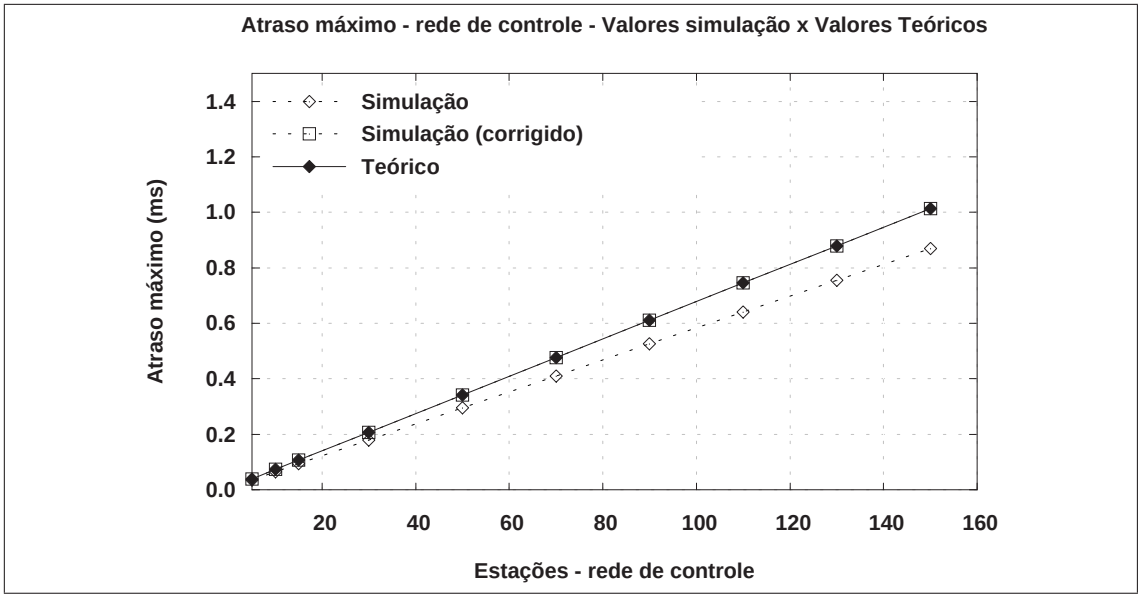


Figura 5.4: $A_{T_{max}}$ – comparação entre valor teórico e experimental (Caso 1)

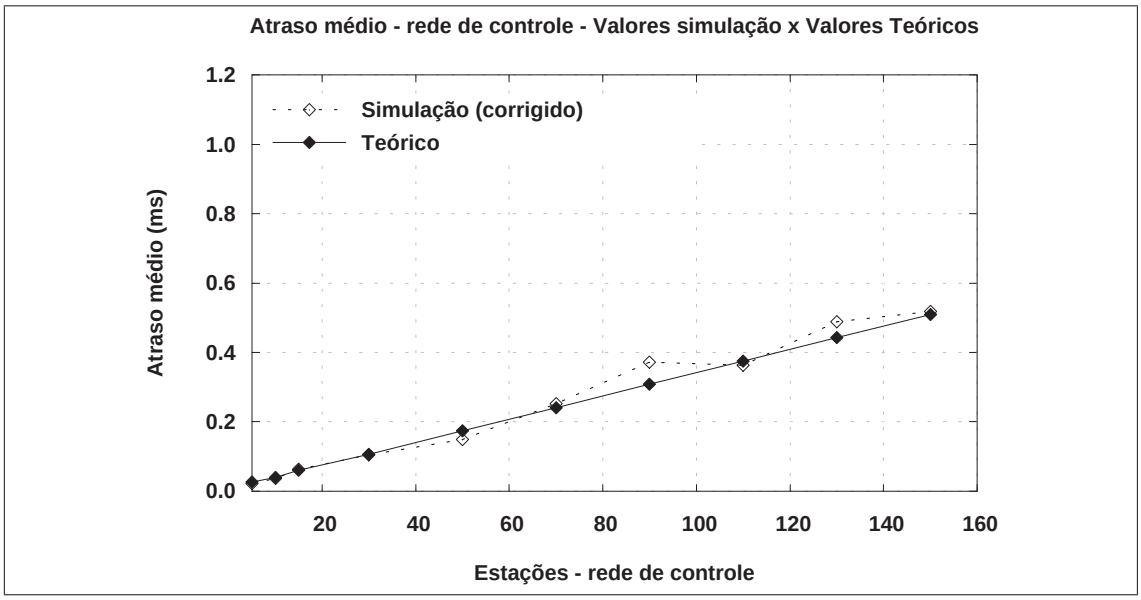


Figura 5.5: $A_{T_{med}}$ – comparação entre valor teórico e experimental (Caso 1)

Outra medida relevante observada durante as simulações é a variação sobre o valor de atraso de pacotes consecutivos (*jitter*) - figura 5.6. O valor médio calculado para a variação no atraso dos quadros se mantém em um patamar bem baixo quando comparado ao seu limite máximo observado. O mesmo comportamento é mantido enquanto mais estações são incorporadas à rede de controle.

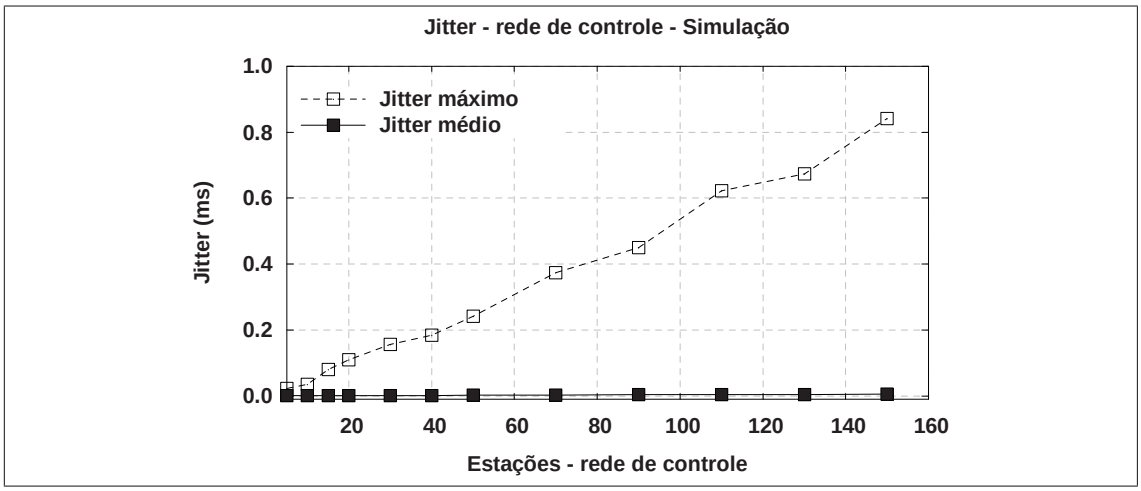


Figura 5.6: Jitter – fluxos da rede de controle (Caso 1)

A razão para isto está no fato de que valores pequenos de *jitter* são muito mais freqüentes nas comunicações entre as estações da rede. A distribuição de freqüências apresentada na tabela 5.4[1] e, em parte na figura 5.7, exemplifica esta situação em um fluxo específico de pacotes (estação → controlador), quando a rede de controle é composta por 15 estações.

Tabela 5.4: Distribuição de freqüências (*Jitter*)

Intervalo (μs)			Freqüência
00,00	⊢	04,57	9258
04,57	⊢	09,14	687
09,14	⊢	13,71	47
13,71	⊢	18,29	7
18,29	⊢	22,86	1
22,86	⊢	27,43	0
27,43	⊢	32,00	0
32,00	⊢	36,57	0
36,57	⊢	41,14	0
41,14	⊢	45,71	0
45,71	⊢	50,29	0
50,29	⊢	54,86	0
54,86	⊢	59,43	0
59,43	⊢⊣	64,00	1

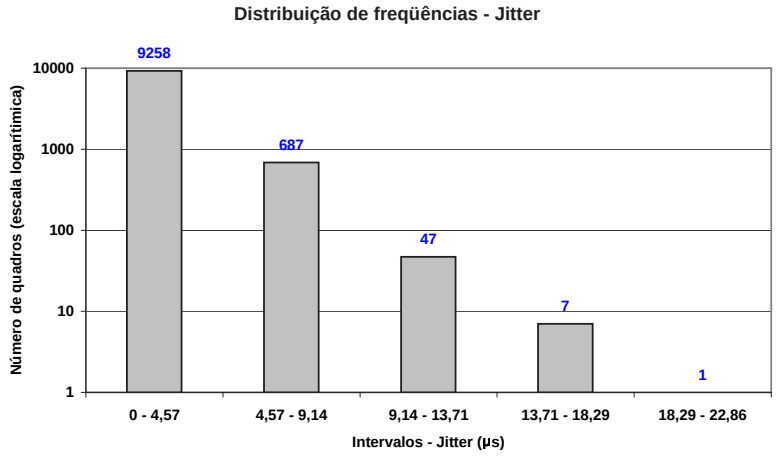


Figura 5.7: Jitter – histograma de freqüências (Caso 1)

[1]O valor sugerido pela regra de Sturges para o número i de intervalos de classes em uma distribuição de 10.000 quadros é 14.

$$\begin{aligned} i &\approxeq 1 + 3,3 \times log(n) \\ &\approxeq 1 + 3,3 \times log(10.000) \\ &\approxeq 14,2 \end{aligned}$$

Para demonstrar o isolamento e independência entre as duas classes de tráfego, os experimentos foram repetidos, excluindo a comunicação de dados e mantendo ativas apenas as estações da rede de controle. Os resultados obtidos foram praticamente os mesmos, conforme tabela 5.5 e figuras 5.8 e 5.9.

Tabela 5.5: Tráfego de controle $\times$ tráfego de controle $+$ dados – Caso 1 ($p0$)

Estações	**Tráfego controle + dados**			**Tráfego controle**		
	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{Jitter_{med}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{Jitter_{med}}$
5	0,03552	0,01252	0,00014	0,03552	0,01252	0,0002
10	0,06428	0,01252	0,00029	0,06425	0,01252	0,0004
15	0,09308	0,01252	0,00056	0,09309	0,01252	0,00058
20	0,12161	0,01252	0,00076	0,12188	0,01252	0,00066
30	0,17925	0,01252	0,00109	0,1793	0,01252	0,00087
40	0,23688	0,01252	0,00149	0,23674	0,01252	0,00134
50	0,29457	0,01252	0,00157	0,29451	0,01252	0,00191
70	0,40836	0,01252	0,00234	0,40308	0,01252	0,00221
90	0,52427	0,01252	0,00291	0,51854	0,01252	0,00345
110	0,63881	0,01252	0,00392	0,63983	0,01252	0,00392
130	0,7454	0,01252	0,00448	0,75538	0,01252	0,00425
150	0,86994	0,01252	0,00533	0,87053	0,01252	0,00509

Obs: *valores em milissegundos (ms)*

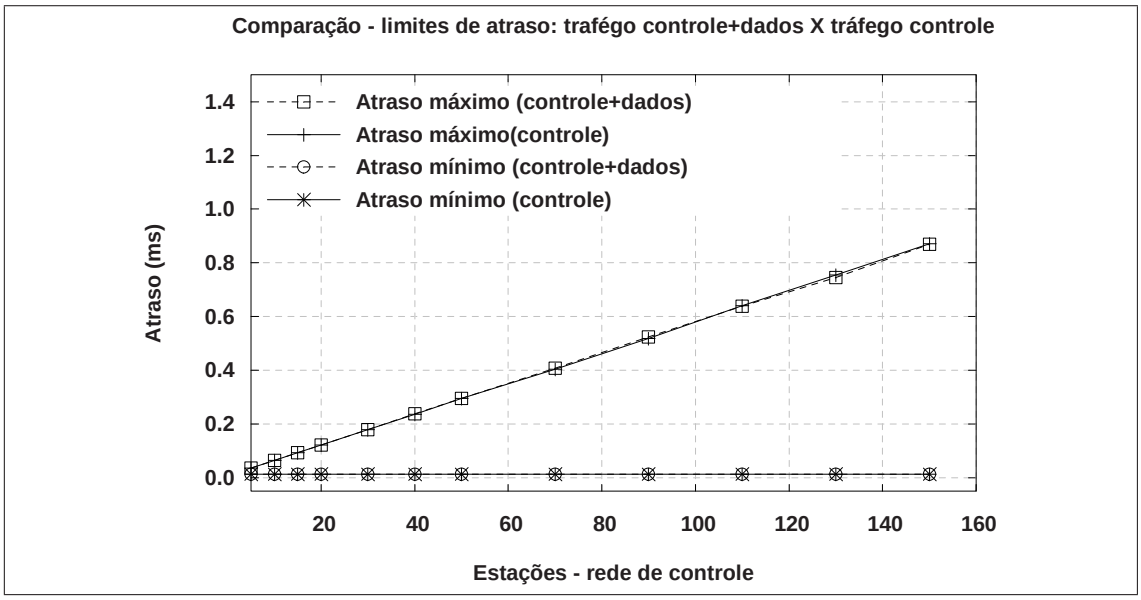


Figura 5.8: Influência do tráfego não prioritário nos limites de A_T (Caso 1)

Ressalte-se que, dada inexistência de canais compartilhados entre as duas classes de tráfego da rede, o isolamento observado é resultado exclusivo da segmentação proporcionada pelo comutador e não de tratamento diferenciado com base na prioridade dos pacotes. Esta afirmativa é respaldada pelo gráfico da figura 5.9 que não demonstra variações significativas no *jitter* em cargas distintas na rede: controle e controle+dados.

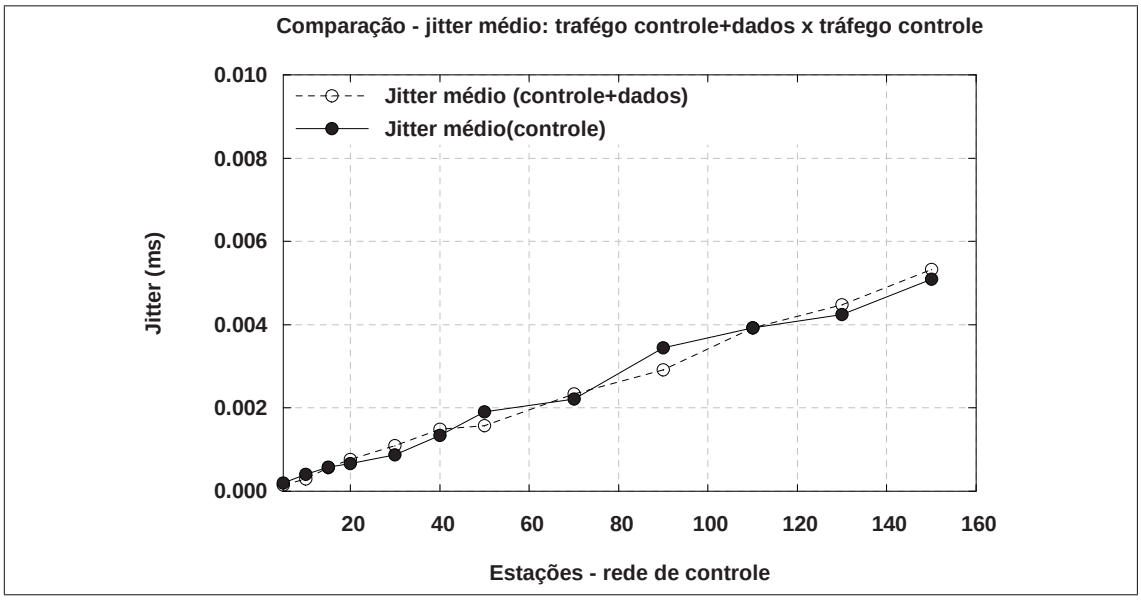


Figura 5.9: Influência do tráfego não prioritário no Jitter médio da rede de controle (Caso 1)

5.3 Caso 2: N Estações x 1 *Switch* - 3 Prioridades

O segundo caso em estudo remete à topologia ilustrada na figura 3.7 e aos parâmetros apresentados na tabela 3.3. A transmissão de mensagens é efetuada sob três prioridades diversas, estando a mais baixa associada ao tráfego de dados e as demais, a duas categorias distintas de pacotes da rede de controle.

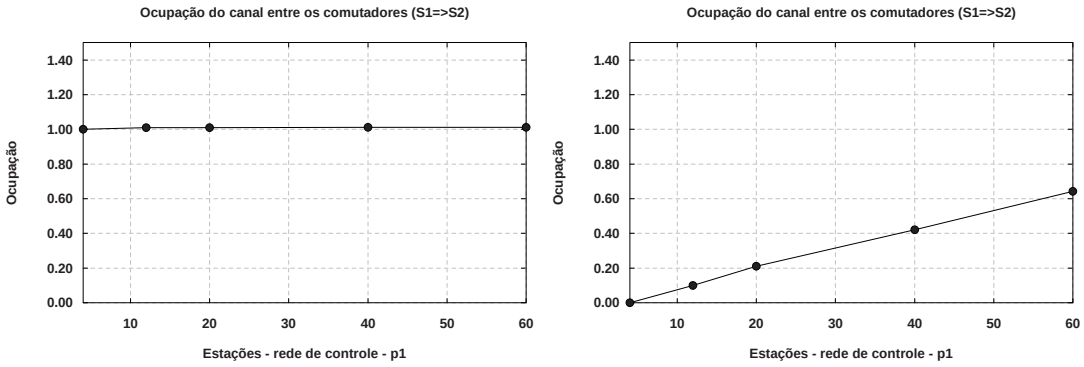
A diferença fundamental para o cenário anterior está na presença de um canal de concentração de todo tráfego gerado na rede, uma vez que receptores e transmissores são considerados em lados opostos em relação aos *switches*. Diante desta configuração, uma maior influência entre as distintas classes é esperada através do aumento no tempo de contenção residual (T_{Res} no modelo apresentado no capítulo 3), possivelmente observado no processo de disputa pelo canal de ligação entre os comutadores.

Tendo o objetivo de identificar corretamente estas possíveis interferências entre classes, o cenário é analisado sob duas composições distintas de carga. Inicialmente admite-se a coexistência das redes de controle e dados, ajustando o tráfego não prioritário para manter a utilização do canal de ligação entre os comutadores em torno de 1 (100%), em todas simulações efetuadas (Figura 5.10(a)). Em seguida, os experimentos são repetidos mantendo ativas apenas as estações da rede de controle (Figura 5.10(b)).

Os resultados obtidos são apresentados nas próximas seções, isoladamente para cada configuração de carga. A exemplo do caso anterior, durante a experimentação, os efeitos do processo de policiamento e conformação de tráfego foram anulados através do superdimensionamento dos parâmetros dos algoritmos envolvidos.

5.3.1 Tráfego: Rede de Controle + Dados

A tabela 5.6 apresenta as características principiais da rede de dados modelada, enquanto a figura 5.11 ilustra o ajuste dos parâmetros da rede nos *scripts* de simulação (Anexo B).



(a) controle+dados (b) controle

Figura 5.10: Taxa de utilização do canal S1 → S2 (Caso 2)

Tabela 5.6: Parâmetros principais da rede de dados – Caso 2

Descrição	**Valor**
Número de entidades	12 estações
Taxa constante de geração de quadros	5000 quadros/s
Tamanho dos quadros	12240 bits

```
set opt(rede,f0) 10          ;#nro_estacoes rede de controle - fluxo prioridade 0 - f0
set opt(tam_pacote,f0) 96    ;# tamanho do quadro - f0
set opt(periodo,f0) 0.001    ;# 1000 quadros/segundo - taxa de geracao - f0
set opt(codepoint,f0) 10

set opt(rede,f1) 20          ;#nro_estacoes rede de controle - fluxo prioridade 1 - f1
set opt(tam_pacote,f1) 72    ;# tamanho do quadro - f1
set opt(periodo,f1) 0.001    ;# 1000 quadros/segundo - taxa de geracao - f1
set opt(codepoint,f1) 11

set opt(rede,dados) 12
set opt(tam_pacote,dados) 1530
set opt(periodo,dados) 0.0002
set opt(codepoint,dados) 0
```

Figura 5.11: Definição dos parâmetros de simulação (Caso 2)

5.3.1.1 Fluxos de Prioridade 0

Novamente os valores obtidos por simulação são ligeiramente inferiores àqueles diretamente derivados da aplicação do modelo analítico de cálculo (Tabela 5.7). À medida que mais estações são inseridas na rede de controle, maior se torna a diferença observada. Conforme visto anteriormente, a razão para este fato está no desrespeito, durante as simulações, do tempo mínimo exigido pelo protocolo CSMA/CD entre duas transmissões consecutivas.

Tabela 5.7: Resultados dos experimentos $\times$ valores teóricos – Caso 2 ($p0$)

Estações	**Resultados Teóricos**			**Resultados Simulação**		
	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
2	0,02454	0,08574	0,15558	0,02454	0,08764	0,15290
6	0,02454	0,10302	0,19014	0,02454	0,10270	0,18322
10	0,02454	0,12030	0,22470	0,02454	0,11807	0,21333
20	0,02454	0,16350	0,31110	0,02454	0,15648	0,29009
30	0,02454	0,20670	0,39750	0,02454	0,19484	0,36689

Obs: *valores de A_T em milissegundos (ms)*

Tabela 5.8: Valores experimentais corrigidos – Caso 2 ($p0$)

Estações	**Resultados Teóricos**			**Resultados Simulação**		
	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
2	0,02454	0,08574	0,15558	0,02454	0,08812	0,15386
6	0,02454	0,10302	0,19014	0,02454	0,10510	0,18802
10	0,02454	0,12030	0,22470	0,02454	0,12239	0,22197
20	0,02454	0,16350	0,31110	0,02454	0,16560	0,30833
30	0,02454	0,20670	0,39750	0,02454	0,20876	0,39473

Obs: *valores de A_T em milissegundos (ms)*

A correção dos valores experimentais requer a aplicação das equações 5.3 e 5.4. Os resultados deste ajuste são apresentados na figura 5.12 e na tabela 5.8. Como o enfileiramento se dá apenas no primeiro comutador (S1, figura 3.7), a existência de dois canais compartilhados por mais de uma classe de tráfego (S1 $\rightarrow$ S2 e S2 $\rightarrow$ controlador) não implica que o tempo de correção seja aplicado duas vezes sobre os valores encontrados. O espaçamento adequado na saída de S1 seria mantido pelo repasse em S2.

Os gráficos das figuras 5.13 e 5.14 apresentam o confronto entre os valores de atraso teóricos e práticos, já com o emprego da correção necessária.

Em relação ao *jitter*, os resultados apresentados no gráfico da figura 5.15 não demonstram a mesma desproporcionalidade entre valores médios e máximos verificada no cenário anterior. Este comportamento tem como causa principal a disputa pelo canal de concentração inserido na rede que se reflete na variação imposta sobre o tempo de contenção residual T_{Res}. Esta parcela variável do atraso total passa a ser relevante à medida que a carga no canal é mantida em um valor elevado, colaborando para o aumento dos valores médios de *jitter*.

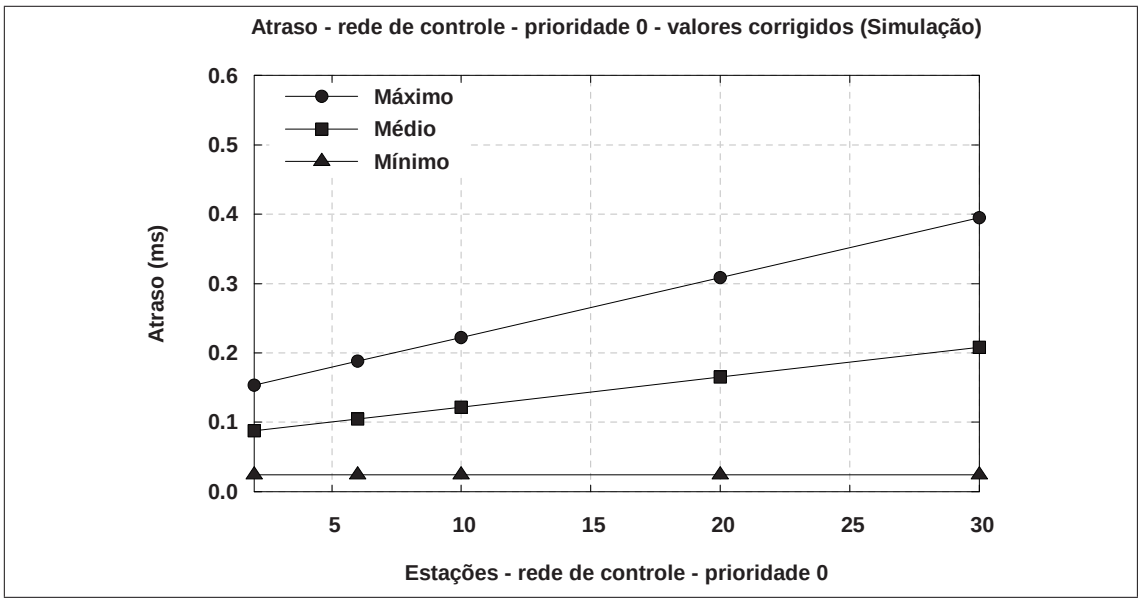


Figura 5.12: Valores obtidos por simulação – A_T (Caso 2, $p0$)

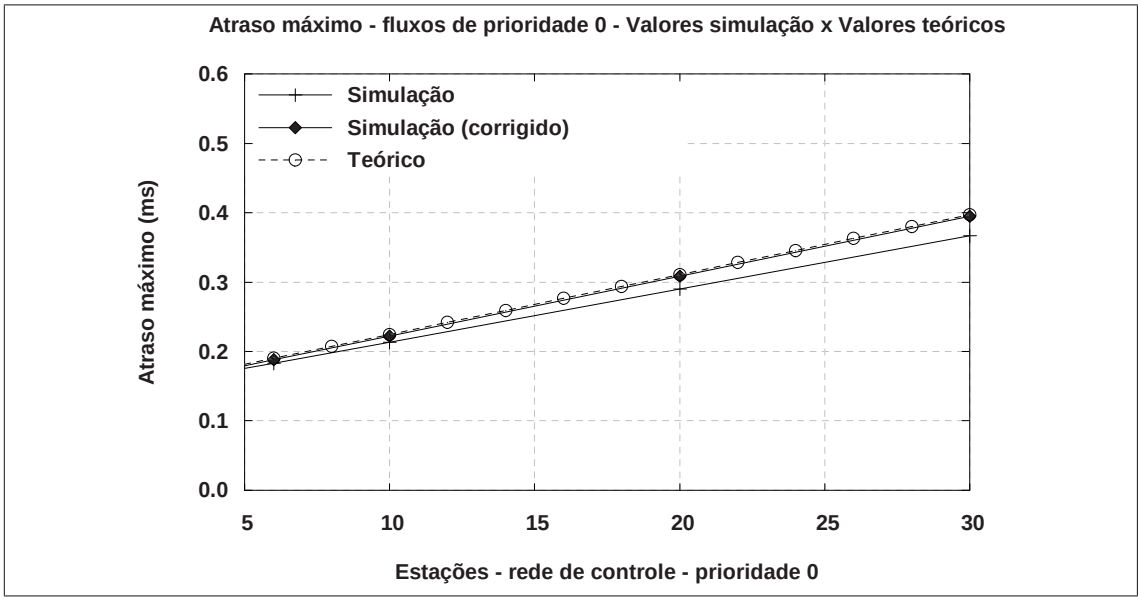


Figura 5.13: $A_{T_{max}}(p0)$ – comparação entre valor teórico e experimental (Caso 2)

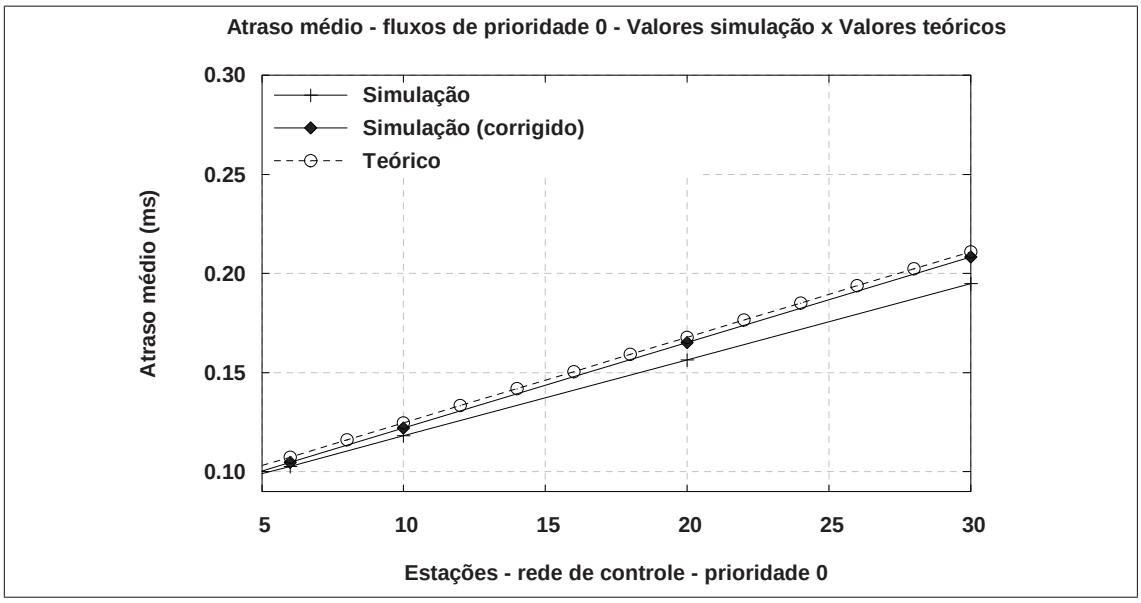


Figura 5.14: $A_{T_{med}}(p0)$ – comparação entre valor teórico e experimental (Caso 2)

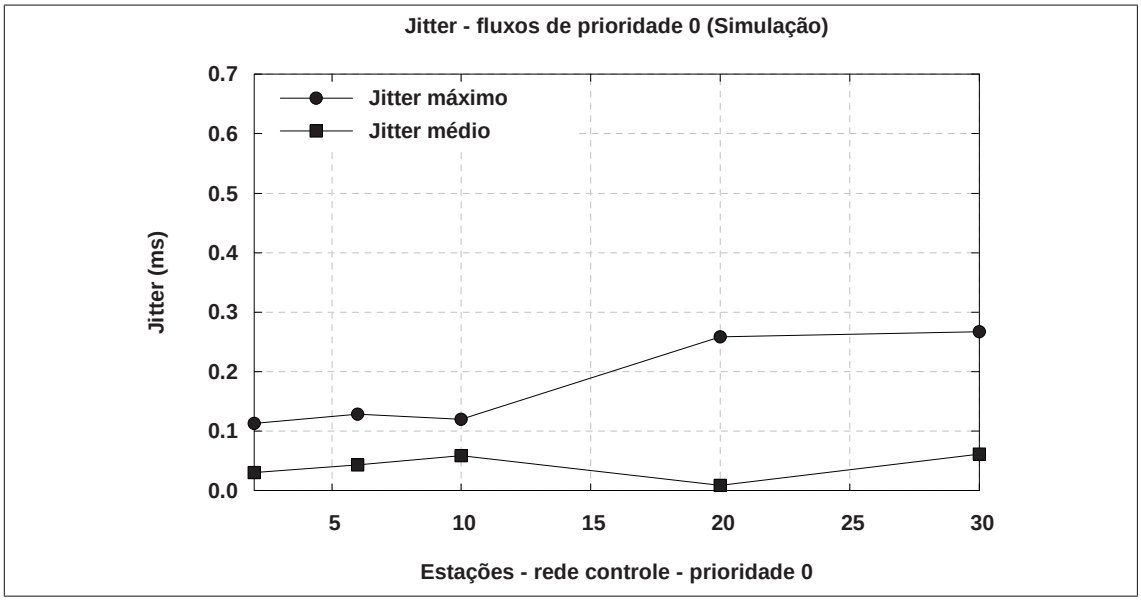


Figura 5.15: Jitter – fluxos de prioridade 0 (Caso 2)

5.3.1.2 Fluxos de Prioridade 1

Tabela 5.9: Resultados dos experimentos $\times$ valores teóricos – Caso 2 ($p1$)

Estações p(0)	*Estações p(1)*	**Resultados Teóricos**			**Resultados Simulação**		
		$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
2	4	0,01878	0,10398	0,17862	0,01878	0,10573	0,17561
6	12	0,01878	0,16542	0,26694	0,01878	0,15920	0,25230
10	20	0,01878	0,22686	0,35526	0,01878	0,21298	0,32853
20	40	0,01878	0,38046	0,57606	0,01878	0,34738	0,52050
30	60	0,01878	0,53406	0,79686	0,01878	0,48173	0,71248

Obs: *valores de A_T em milissegundos (ms)*

Mais uma vez a inspeção dos resultados extraídos (Tabela 5.9) revela a necessidade de correção dos valores. Como os resultados estão relacionados com quadros de prioridade 1, as equações 5.3 e 5.4, utilizadas até então para este fim, devem ser reescritas de modo a considerar o enfileiramento em ambas classes de tráfego e não apenas na de maior prioridade, conforme equações 5.5 e 5.6.

$$\begin{aligned} Tempo_{correcao}(A_{T_{max}}) &= (N_{p0} + N_{p1} - 1) \times T_{EQ} \\ &= (N_{p0} + N_{p1} - 1) \times 0,96\mu s \end{aligned} \tag{5.5}$$

onde N_{p0} e N_{p1} representam o número de estações transmitindo sob prioridades “0” e “1”, respectivamente.

$$\begin{aligned} Tempo_{correcao}(A_{T_{med}}) &= \left(N_{p0} + \frac{N_{p1} - 1}{2} \right) \times T_{EQ} \\ &= \left(N_{p0} + \frac{N_{p1} - 1}{2} \right) \times 0,96\mu s \end{aligned} \tag{5.6}$$

Tabela 5.10: Valores experimentais corrigidos – Caso 2 ($p1$)

Estações p(0)	*Estações p(1)*	**Resultados Teóricos**			**Resultados Simulação**		
		$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$	$\mathbf{A_{T_{min}}}$	$\mathbf{A_{T_{med}}}$	$\mathbf{A_{T_{max}}}$
2	4	0,01878	0,10398	0,17862	0,01878	0,10909	0,18041
6	12	0,01878	0,16542	0,26694	0,01878	0,17024	0,26862
10	20	0,01878	0,22686	0,35526	0,01878	0,23170	0,35637
20	40	0,01878	0,38046	0,57606	0,01878	0,38530	0,57714
30	60	0,01878	0,53406	0,79686	0,01878	0,53885	0,79792

Obs: *valores de A_T em milissegundos (ms)*

A tabela 5.10 e figura 5.16 apresentam os valores de A_T já corrigidos. Este procedimento de correção novamente revela a coerência entre os valores teóricos e experimentais (Figuras 5.17 e 5.18).

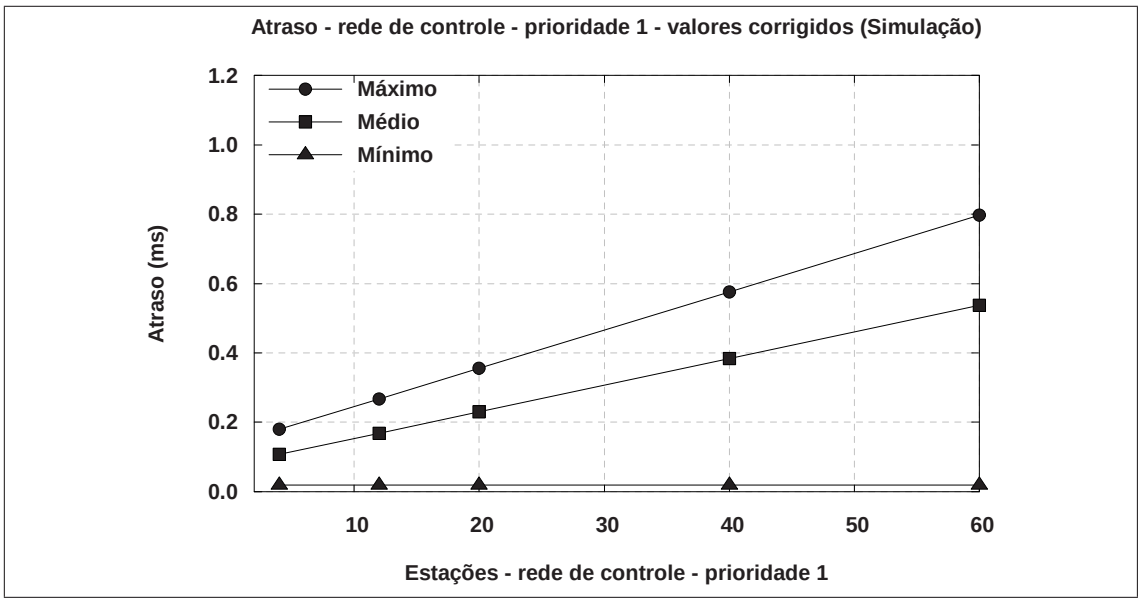


Figura 5.16: Valores obtidos por simulação – A_T (Caso 2,$p1$)

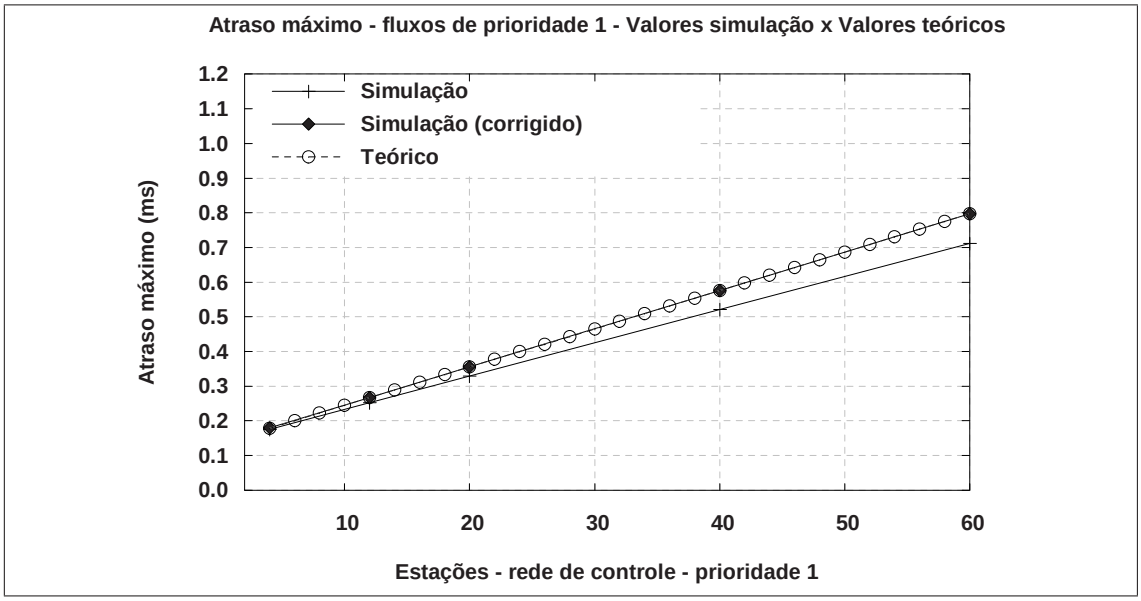


Figura 5.17: $A_{T_{max}}(p1)$ – comparação entre valor teórico e experimental (Caso 2)

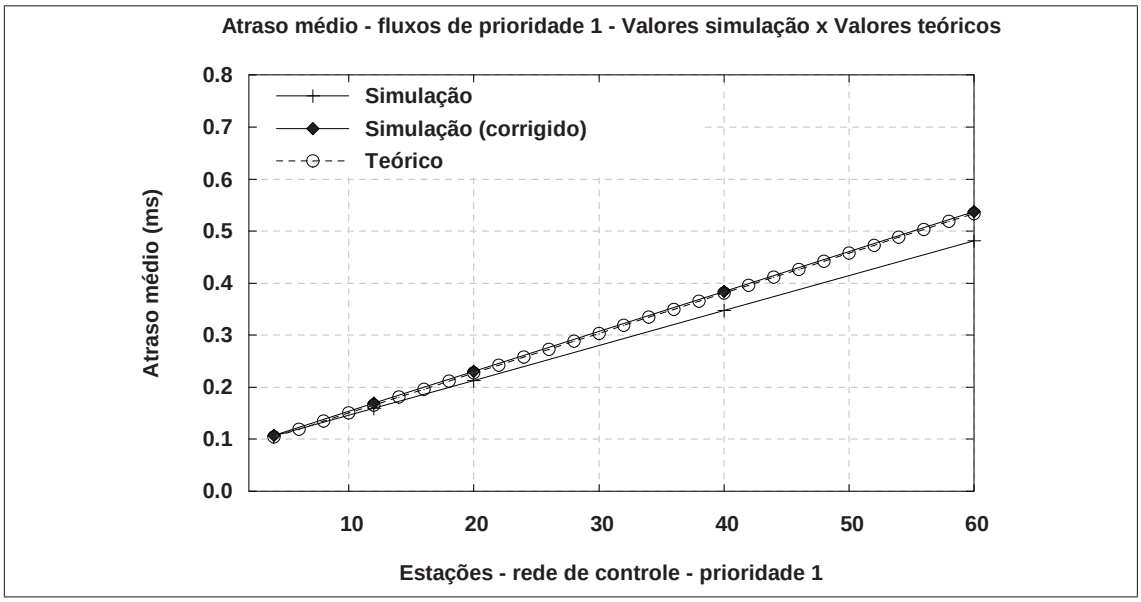


Figura 5.18: $A_{T_{med}}(p1)$ – comparação entre valor teórico e experimental (Caso 2)

O gráfico do *jitter* extraído para os fluxos de prioridade 1 (Figura 5.19) apresenta o mesmo comportamento exposto na figura 5.15 para os fluxos de maior prioridade. O aumento na variação de A_T é justificado pela influência de ambas as classes no tempo de enfileiramento experimentado por quadros de prioridade 1.

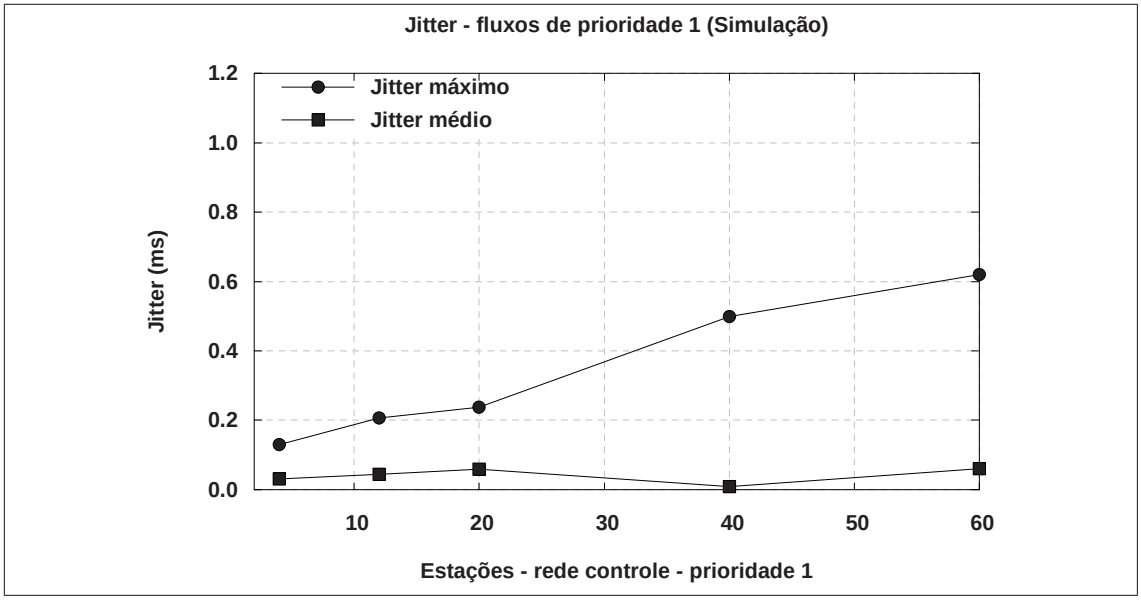


Figura 5.19: Jitter - fluxos de prioridade 1 - (Caso 2)

5.3.2 Tráfego: Apenas Rede de Controle

A série de gráficos apresentados nesta seção procura demonstrar o efeito do tráfego não prioritário (rede de dados) sobre a rede de controle. As comparações traçadas se baseiam na repetição dos mesmos experimentos e extração das mesmas medidas, mas com o diferencial de ativar apenas estações da rede de controle.

5.3.2.1 *Fluxos de Prioridade 0*

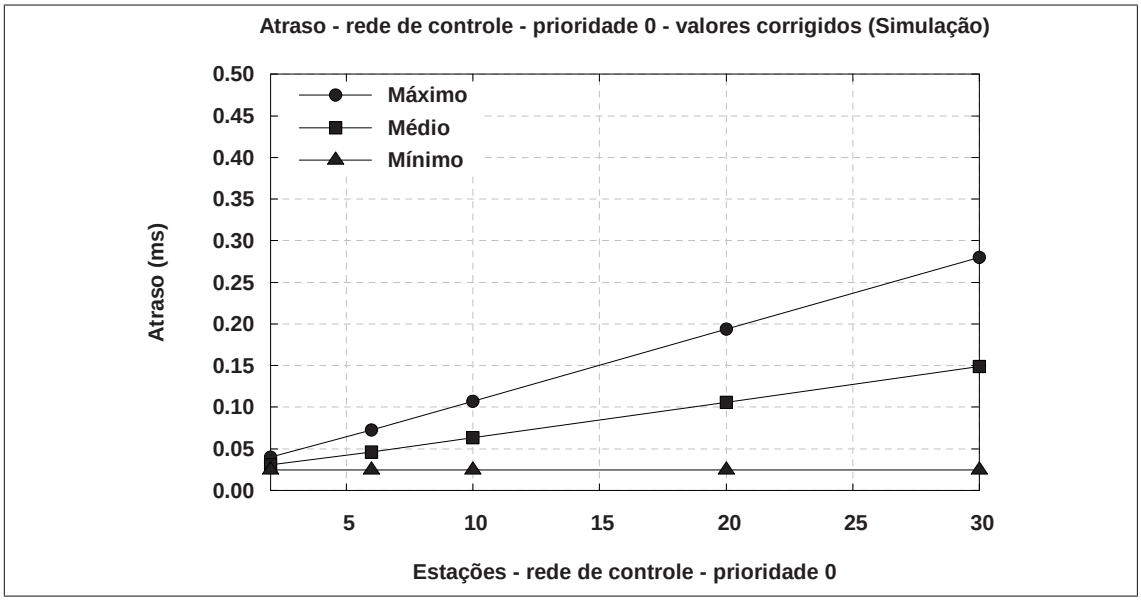


Figura 5.20: Valores obtidos por simulação com a rede de dados inativa (Caso 2, $p0$)

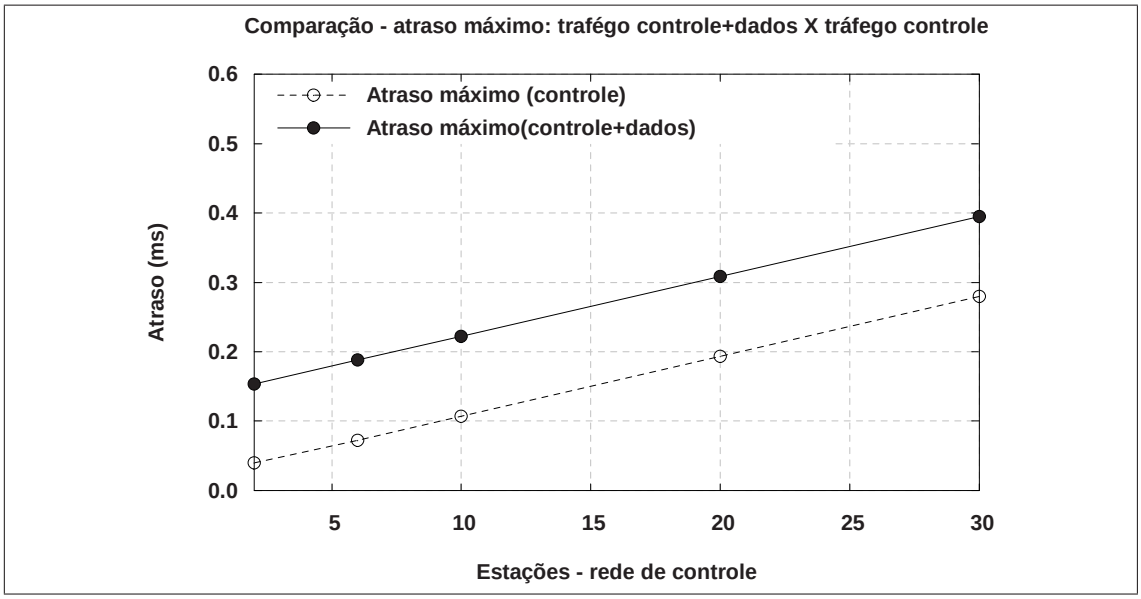


Figura 5.21: Influência do tráfego não prioritário sobre $A_{T_{max}}(p0)$

Os resultados das simulações são expostos na figura 5.20. Conforme esperado, os gráficos das figuras 5.21 e 5.22 demonstram que o tráfego de dados influi através da soma de um valor fixo sobre os atrasos observados na rede de controle. Trata-se do tempo gasto quando o canal já está ocupado com uma transmissão não prioritária, obrigando que quadros de maior prioridade aguardem sua liberação, dada a impossibilidade de interrupção do processo.

Note-se a coerência nos deslocamentos apresentados: aproximadamente $122\mu s$ para $A_{T_{max}}$ e $61\mu s$ para $A_{T_{med}}$. Estes são, respectivamente, os tempos gastos com a inserção no canal de um quadro completo e de metade de um quadro da rede de dados. Outro

fator importante é o aumento significativo na variação dos valores de atraso (Figura 5.23). Entretanto, em média e conforme esperado, a variação imposta está confinada sob o tempo gasto com metade de um quadro da rede de dados ($\approx 61\mu s$).

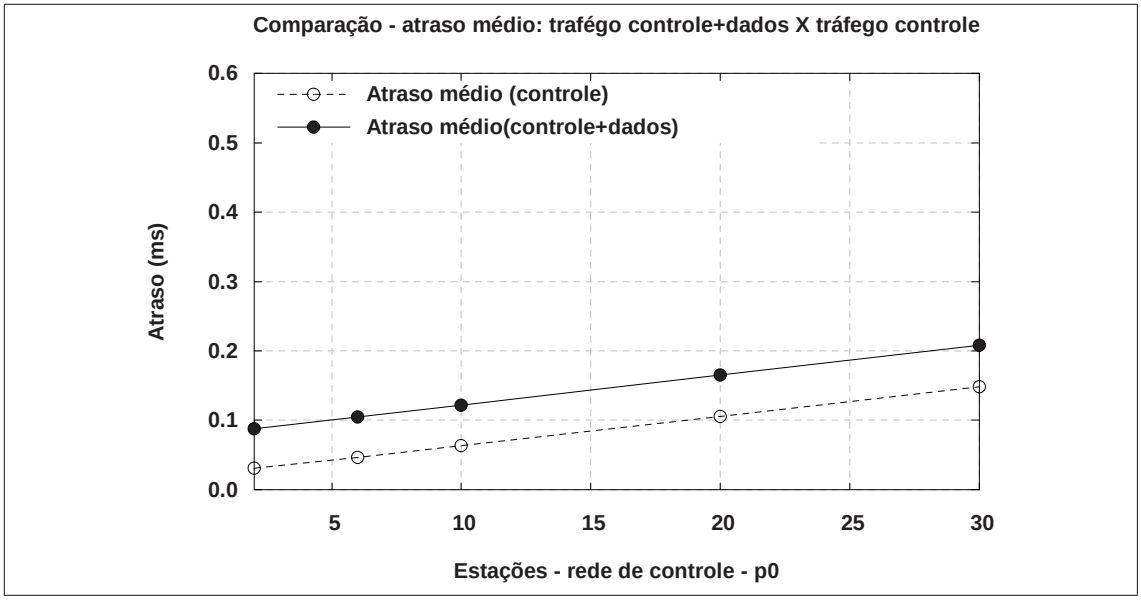


Figura 5.22: Influência do tráfego não prioritário sobre $A_{T_{med}}(p0)$

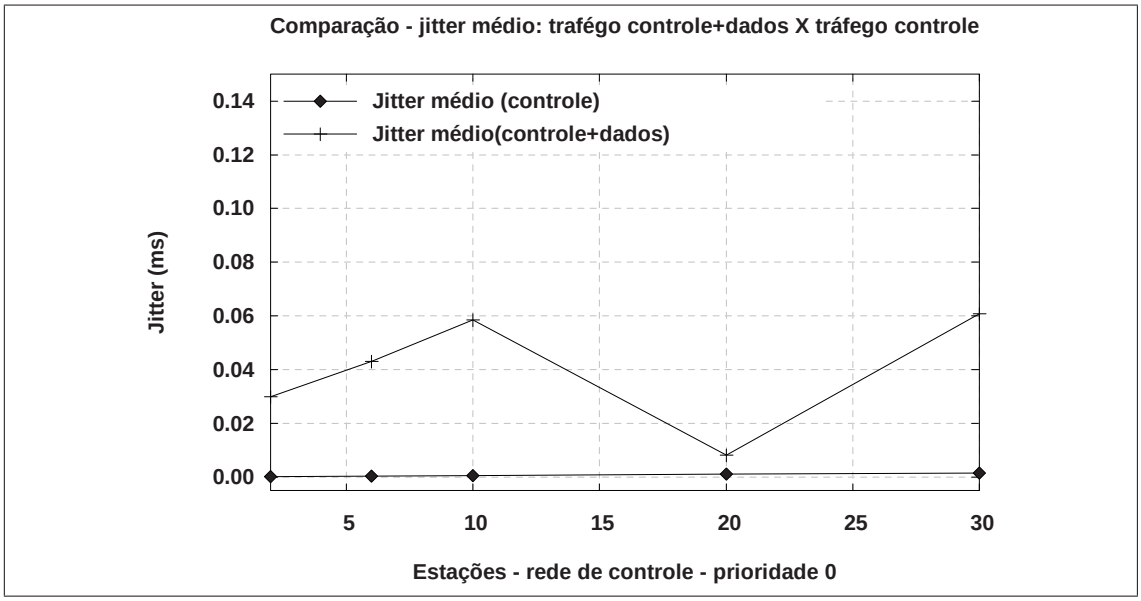


Figura 5.23: Influência do tráfego não prioritário no Jitter médio da rede de controle (Caso 2,$p0$)

5.3.2.2 Fluxos de Prioridade 1

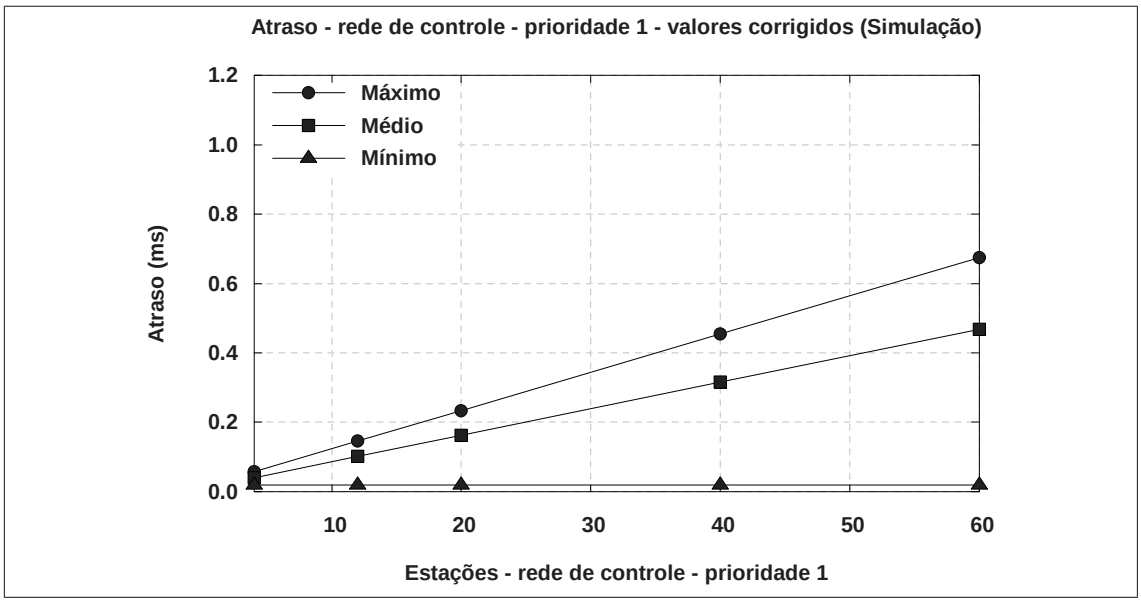


Figura 5.24: Valores obtidos por simulação com a rede de dados inativa (Caso 2,$p1$)

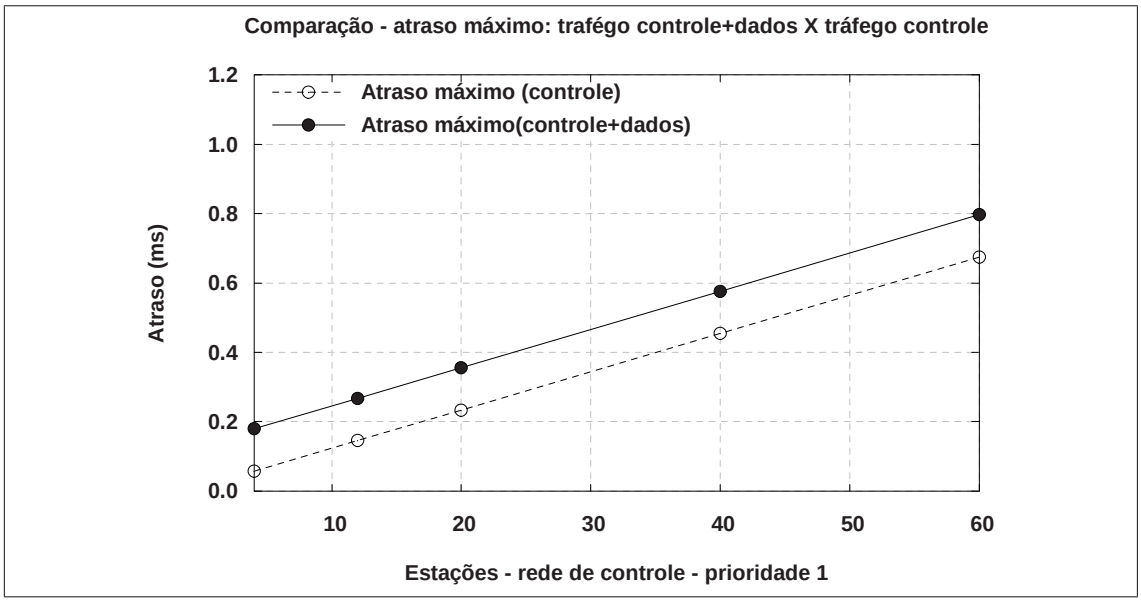


Figura 5.25: Influência do tráfego não prioritário sobre $A_{T_{max}}(p1)$

Os resultados exibidos na figura 5.24 e as comparações apresentadas nas figuras 5.25, 5.26 e 5.27 demonstram que a influência da rede de dados sobre os fluxos de quadros com prioridade 1 é a mesma desempenhada sobre os quadros de prioridade 0. Os efeitos causados por quadros não prioritários são traduzidos em uma influência negativa global sobre a rede de controle.

Apesar disto, as análises mostram que a variação imposta está confinada dentro de limites estabelecidos e previsíveis. O conhecimento do tamanho máximo e médio dos quadros da rede não prioritária é suficiente para definir a amplitude da interferência causada.

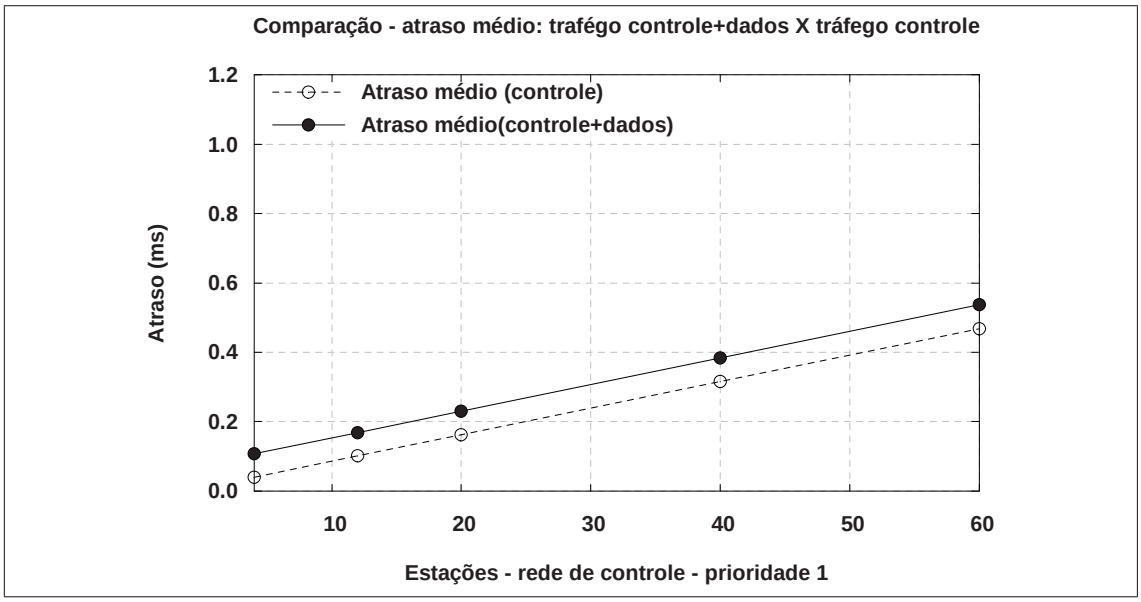


Figura 5.26: Influência do tráfego não prioritário sobre $A_{T_{med}}(p1)$

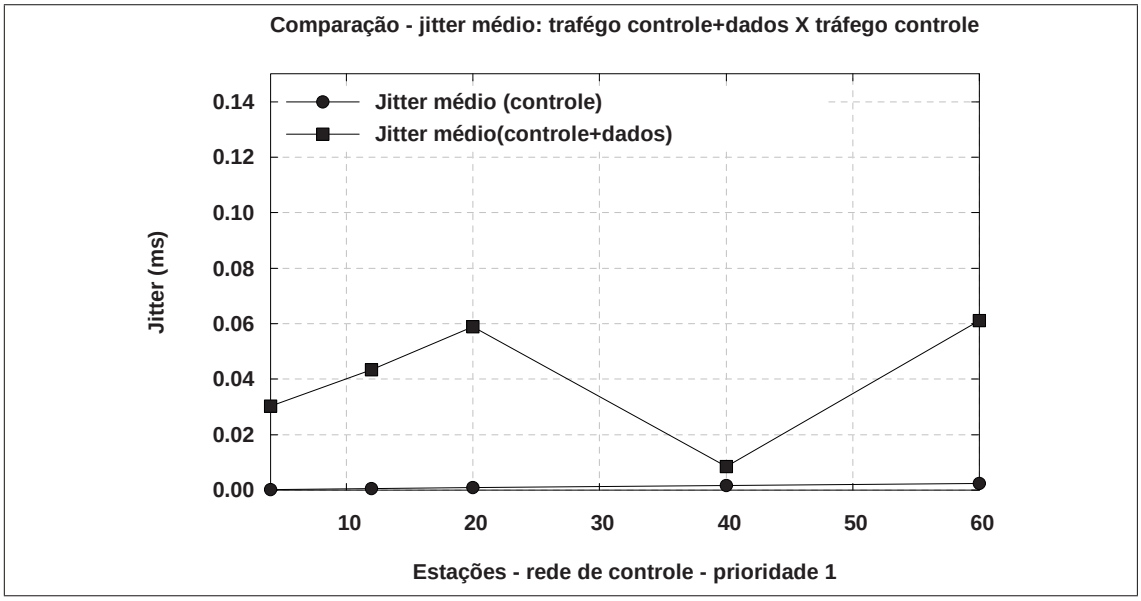


Figura 5.27: Influência do tráfego não prioritário no Jitter médio da rede de controle (Caso 2,$p1$)

6 CONCLUSÕES

O modelo analítico de cálculo proposto neste trabalho é capaz de prever os limites de atraso fim-a-fim observado por diferentes classes de tráfego nas comunicações em redes *ethernet* micro-segmentadas. O perfil temporal é traçado unicamente com base nos efeitos provocados pela estrutura de rede, mais especificamente, pelo nível de enlace da arquitetura.

A análise tem seu foco direcionado para ambientes de automação através de um cenário de estudo inspirado em sistemas de controle distribuídos, transmissões cíclicas e no modelo de comunicação mestre-escravo. Além do tráfego de controle, admite-se a convivência com uma categoria distinta de quadros que não possuem requisitos temporais – rede de dados.

Duas contribuições principais emergem do trabalho desenvolvido:

- o próprio modelo de cálculo analítico que considera a presença na rede de diferentes classes de tráfego;
- avaliação positiva a respeito do nível de determinismo temporal desta configuração de rede cada vez mais comum.

A dedução do modelo é baseada em uma análise determinística através da previsão do tráfego prioritário presente na rede a cada ciclo de transmissão. Esta abordagem é alternativa à técnica de aproximação do tráfego por distribuições conhecidas e análise do problema sob a ótica da teoria de filas, como em (SONG; KOUBAA; SIMONOT, 2002) e (IIDA et al., 2001).

Os resultados obtidos atestam a validade do modelo e, como conseqüência, a viabilidade de emprego desta tecnologia simples, barata e disponível em sistemas de controle. Embora o cenário considerado represente uma configuração bastante comum na área de automação, o uso generalizado do modelo requer o ataque complementar a alguns aspectos:

- estudo das influências de níveis superiores – tarefa que exige a observação de diversos níveis de abstração distintos como: protocolos de transporte, linguagens de programação, sistemas operacionais e etc. Neste sentido, o uso de soluções dedicadas a sistemas com características tempo real ganha força;
- latência adicionada pelos comutadores – a exemplo dos demais tempos de processamento, o tempo adicionado pelos *switches* foi desconsiderado.
- adequação a outros perfis de tráfego prioritário, como comunicações aperiódicas – exige a extensão do modelo. A manutenção da análise determinística está condicionada ao exato conhecimento das comunicações. Entre as abordagens alternativas

estão: a análise baseada em pior caso, a reserva de ciclos exclusivos para inserção de comunicações esporádicas, o estudo da distribuição de probabilidades (estatisticamente aceitável), entre outras.

Um ponto importante revelado é a possibilidade de inversão de prioridades quando o tráfego de diferentes classes é concentrado em canais de saída. A abordagem de pior caso, adotada para a rede de dados no estudo deste fenômeno, expõe uma influência negativa sobre a temporização dos quadros prioritários, mas demonstra que as oscilações inseridas são limitadas e previsíveis. A interferência verificada está atrelada aos tamanhos máximos (Figura 6.1) e médios (Figura 6.2) dos quadros de menor prioridade envolvidos na disputa, e acaba por definir a amplitude média da variação imposta sobre A_T (Figura 6.3).

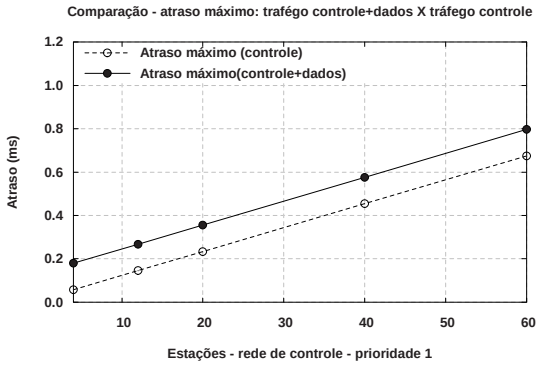


Figura 6.1: Interferência sobre $A_{T_{max}}(p1)$

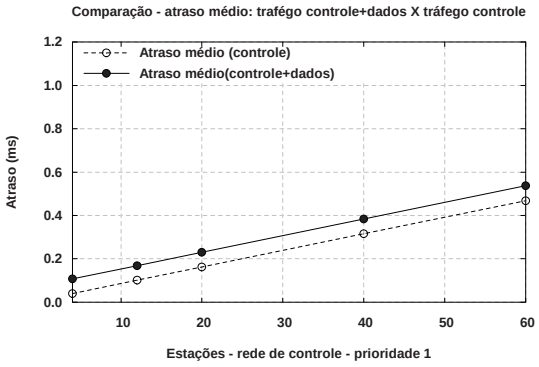


Figura 6.2: Interferência sobre $A_{T_{med}}(p1)$

Por outro lado, a ausência de pontos de concentração de tráfego permite verificar a independência entre classes com tratamento distinto, conforme demonstrado pela figura 6.4. A variação do atraso se mantém praticamente a mesma enquanto a carga é aumentada significativamente (controle $\times$ controle+dados).

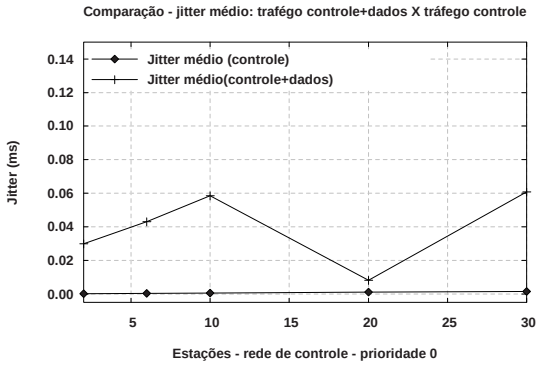


Figura 6.3: Interferência – Jitter (Caso 2,$p1$)

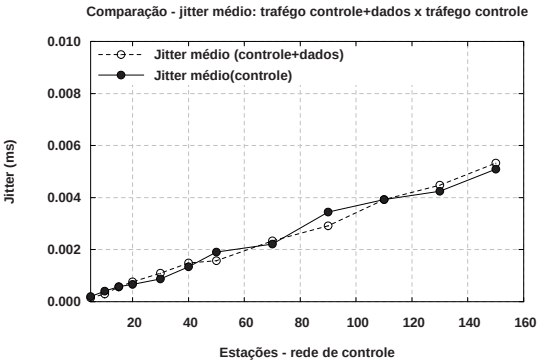


Figura 6.4: Interferência – Jitter (Caso 1)

Em resumo, os resultados permitem concluir que o uso de redes *ethernet* baseadas em *switches*, agregado à utilização dos mecanismos de diferenciação de tráfego disponíveis, representa uma tecnologia bastante promissora para sistemas de tempo real. Entretanto, é necessário que a arquitetura seja uniformizada em todas as camadas.

Embora definidos em 1998, e implementados na maioria dos equipamentos utilizados hoje em dia, os dispositivos de priorização especificados nas normas IEEE não foram completamente difundidos. Um passo importante seria sua disponibilidade para o nível

de aplicação através da alteração de protocolos ou interfaces intermediárias, como por exemplo, da *socket API*.

Como uma aplicação direta, a implementação do modelo proposto permitiria a obtenção automática do perfil temporal de configurações físicas distintas. A interface com o usuário poderia ser elaborada a partir do fornecimento de matrizes descritivas das topologias de interesse.

REFERÊNCIAS

3COM. **SuperStack Switches – Data Sheets**. Disponível em: <http://www.3com.com>. Acesso em: nov. 2003.

ALMES, G.; KALIDINDI, S.; ZEKAUSKAS, M. **A One-way Delay Metric for IPPM**: RFC 2679. [S.l.]: IETF - Internet Engineering Task Force, 1999.

ANDREOZZI, S. **Diffserv Simulations using the Network Simulator**: Requirements, Issues and Solutions. 2001. Dissertação (Mestrado em Ciência da Computação) — FACOLTÀ DI INGEGNERIA, UNIVERSITÀ DEGLI STUDI DI PISA, Pisa, Itália.

BAJAJ, S. et al. **Improving Simulation For Network Research**. [S.l.]: USC - University of Southern California, 1999. (USC Technical Report 99-702).

BELLO, L. L. et al. Performance Analysis of Ethernet Networks in the Process Control. In: IEEE INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELETRONICS, ISIE, 2000, Puebla, Mexico. **Proceedings...** [S.l.]: IEEE, 2000.

BELLO, L. L.; MIRABELLA, O. Design Issues for Ethernet in Automation. In: IEEE INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION, ETFA, 8., 2001, Antibes, France. **Proceedings...** [S.l.]: IEEE, 2001.

BLAKE, S. et al. **An Architecture for Differentiated Service**: RFC 2475. [S.l.]: IETF - Internet Engineering Task Force, 1998.

BOLLELLA, G. et al. **The Real-Time Specification for Java**. [S.l.]: Addison-Wesley, 2000. (The Java Series).

CARO, R. H. Ethernet Wins Over Industrial Automation. **IEEE Spectrum**, [S.l.], n.38, p.114–115, Jan. 2001.

CARREIRO, F. B.; FONSECA, J. A. G.; PEDREIRAS, P. Virtual Token-Passing Ethernet - VTPE. In: INTERNATIONAL CONFERENCE ON FIELDBUS SYSTEMS AND THEIR APPLICATIONS, FET, 2003. **Proceedings...** [S.l.: s.n.], 2003.

CHIUEH, T.; VENKATRAMANI, C. Supporting Real-time Traffic on Ethernet. In: IEEE REAL-TIME SYSTEMS SYMPOSIUM, 1994, San Juan, Puerto Rico. **Proceedings...** [S.l.: s.n.], 1994.

CHOW, M. Y.; TIPSUWAN, Y. Network-based Control Systems: a Tutorial. In: ANNUAL CONFERENCE OF THE IEEE INDUSTRIAL ELETRONICS SOCIETY, 27., 2001, Denver, United States. **Proceedings...** [S.l.]: IEEE, 2001. p.1593–1602.

COURT, R. Real-time Ethenet. **Computer Communications**, [S.l.], v.15, n.3, p.198–201, Apr. 1992.

DECOTIGNIE, J. D. A perspective on Ethernet as a Fieldbus. In: INTERNATIONAL CONFERENCE ON FIELDBUS SYSTEMS AND THEIR APPLICATIONS, FET, 4., 2001, Nancy, France. **Proceedings...** [S.l.: s.n.], 2001.

FANG, W.; SEDDIGH, N.; NANDY, B. **A Time Sliding Window Three Colour Marker (TSWTCM): RFC 2859**. [S.l.]: IETF - Internet Engineering Task Force, 2000.

HOANG, H.; JONSSON, M. Switched Real-Time Ethernet in Industrial Applications - Deadline Partitioning Scheme. In: ASIAN PACIFIC CONFERENCE ON COMMUNICATION, 9., 2003, Penang, Malaysia. **Proceedings...** [S.l.: s.n.], 2003.

HOANG, H. et al. Switched Real-Time Ethernet and Earliest Deadline First Scheduling - Protocols and Traffic Handling. In: INTERNATIONAL PARALLEL AND DISTRIBUTED PROCESSING SYMPOSIUM, IPDPS; 10., INTL. WORKSHOP ON PARALLEL AND DISTRIBUTED REAL-TIME SYSTEMS, 2002, Fort Lauderdale, Florida,USA. **Proceedings...** [S.l.: s.n.], 2002.

ICIR. **Home Page**. Disponível em: <http://www.icir.org>. Acesso em: mar. 2003.

IEC. **IEC 61158-4: Digital Data Communications for Measurement and Control – Fieldbus for Use in Industrial Control Systems – Part 4: Data Link Protocol Specification**. Geneva, Switzerland, 2003.

IEEE. **IEEE 802.1D - Part 3: Media Access Control (MAC) Bridges**. Piscataway, NJ - USA, 1998.

IEEE. **IEEE 802.3: CSMA/CD Access Method and Physical Layer Specifications**. Piscataway, NJ - USA, 2002.

IEEE. **IEEE 802.1Q - Virtual Bridged Local Area Networks**. Piscataway, NJ - USA, 2003.

IIDA, K.; TAKINE, T.; SUNAHARA, H.; OIE, Y. Delay analysis for CBR traffic under static-priority scheduling. **IEEE/ACM Trans. Netw.**, [S.l.], v.9, n.2, p.177–185, 2001.

ISO. **ISO 11898-2: Road Vehicles – Controller Area Network (CAN) – Part 2: High Speed Medium Access Unit**. [S.l.], 2003.

JASPERNEITE, J.; NEUMANN, P. Switched Ethernet for Factory Communication. In: IEEE INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION, ETFA, 8., 2001, Antibes, France. **Proceedings...** [S.l.]: IEEE, 2001.

JASPERNEITE, J. et al. Deterministic Real-Time Communication with Switched Ethernet. In: IEEE WORKSHOP ON FACTORY COMMUNICATION SYSTEMS, WFCS, 4., 2002, Vasteras, Sweden. **Proceedings...** [S.l]: IEEE, 2002. p.11–18.

KAPLAN, G. Ethernet's Winning Ways. **IEEE Spectrum**, [S.l.], n.38, p.113–114, Jan. 2001.

KHANNA, V. K.; SINGH, S. An Improved Piggyback Ethernet Protocol and its Analysis. **Computer Networks ISDN Systems**, [S.l.], v.26, n.11, p.1437–1446, Aug. 1994.

KWEON, S.-K.; SHIN, K. G.; WORKMAN, G. Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing. In: IEEE REAL TIME TECHNOLOGY AND APPLICATIONS SYMPOSIUM, 2000. **Proceedings...** [S.l.: s.n.], 2000. p.90–110.

LEE, K. C.; LEE, S. Performance Evaluation of Switched Ethernet for Real-time Industrial Communications. **Computer Standards & Interfaces**, [S.l.], v.24, n.5, p.411–423, 2002.

LIAN, F.; MOYNE, J.; TILBURY, D. Performance Evaluation of Control Networks: Ethernet, Controlnet and Devicenet. **IEEE Control Systems Magazine**, [S.l.], v.21, n.1, p.66–83, Feb. 2001.

MOLDOVANSKY, A. Utilization of Modern Switching Technology in Ethernet/IP Networks. In: INTERNATIONAL WORKSHOP ON REAL-TIME LANS IN THE INTERNET AGE, RTLIA, 1., 2002, Vienna, Austria. **Proceedings...** [S.l.]: Ed. Politema, 2002.

NAM. **Home Page – NAM**: Network Animator. Disponível em: <http://www.isi.edu/nsnam/nam/>. Acesso em: jan. 2003.

NI, L.; QIAO, W.; YANG, M. Switches and Switch Interconnects. In: INTERNATIONAL CONFERENCE ON MASSIVELY PARALLEL PROCESSING USING OPTICAL INTERCONNECTIONS – MPPOI97, 4., 1997, Montreal, Canada. **Proceedings...** [S.l]: IEEE Computer Society, 1997. p.122–129.

NS2. **Home Page – The Network Simulator – NS2**. Disponível em: <http://www.isi.edu/nsnam/ns/index.html>. Acesso em: jan. 2003.

NS2. **Home Page - NS2 - Class Hierarchy**. Disponível em: <http://www.isi.edu/nsnam/nsdoc-classes/hierarchy.html>. Acesso em: mar. 2003.

PARK, J. H.; YOON, Y. C. An Extended TCP/IP Protocol for Real-time Local Area Network. **Control Engineering Practice**, [S.l.], v.6, n.1, p.111–118, Jan. 1998.

PEDREIRAS, P.; ALMEIDA, L. Flexibility, Timeliness and Efficiency over Ethernet. In: INTERNATIONAL WORKSHOP ON REAL-TIME LANS IN THE INTERNET AGE, RTLIA, 1., 2002, Vienna, Austria. **Proceedings...** [S.l.]: Ed. Politema, 2002.

PIEDA, P. et al. **A Network Simulator Differentiated Services Implementation**. Open IP, Nortel Networks, 2000. Disponível em: <http://www7.nortel.com:8080/CTL/>. Acesso em: jan. 2003.

SAMAN. **Home Page – SAMAN Project**. Disponível em: <http://www.isi.edu/saman/index.html>. Acesso em: jan. 2003.

SHANNON, R. E. **System Simulation**: The Art and Science. [S.l.]: Prentice-Hall PTR, 1975.

SHIMOKAWA, Y.; SHIOBARA, Y. Real-Time Ethernet for Industrial Applications. In: INTERNATIONAL CONFERENCE ON INDUSTRIAL ELETRONICS, IECON, 1985, San Francisco, CA, United States. **Proceedings...** [S.l.]: IEEE, 1985.

SONG, Y. Time Constrained Communication over Switched Ethernet. In: INTERNATIONAL CONFERENCE ON FIELDBUS SYSTEMS AND THEIR APPLICATIONS, FET, 4., 2001, Nancy, France. **Proceedings...** [S.l.: s.n.], 2001. p.138–143.

SONG, Y.; KOUBAA, A.; SIMONOT, F. Switched Ethernet for Real-Time Industrial Communication: Modelling and Message Buffering Delay Evaluation. In: IEEE INTERNATIONAL WORKSHOP ON FACTORY COMMUNICATION SYSTEMS, WFCS, 4., 2002, Vasteras, Suécia. **Proceedings...** [S.l.]: IEEE, 2002.

STANKOVIC, J. A. Misconceptions About Real-Time Computing: A Serious Problem for Next-generation Systems. **Computer**, [S.l.], v.21, n.10, p.10–19, 1988.

STANKOVIC, J. A. et al. Strategic Directions in Real-time and Embedded Systems. **ACM Computing Surveys**, [S.l.], v.28, n.4, p.751–763, 1996.

STEFFEN, R.; ZELLER, M.; KNORR, R. Real-Time Communication over Shared Media Local Area Networks. In: INTERNATIONAL WORKSHOP ON REAL-TIME LANS IN THE INTERNET AGE, RTLIA, 2., 2003, Porto, Portugal. **Proceedings...** [S.l.]: Ed. Politema, 2003.

TANENBAUM, A. S. **Computer Networks**. 3rd ed. New Jersey: Prentice-Hall PTR, 1996.

THOMESSE, J. Fieldbus and Interoperability. **Control Engineering Practice**, [S.l.], v.7, n.1, p.81–94, 1999.

WALSH, G. C.; HONG, Y. Scheduling of Networked Control Systems. **IEEE Control Systems Magazine**, [S.l.], v.21, n.1, p.57–65, Feb. 2001.

WETHERALL, D.; LINDBLAD, C. J. Extending Tcl for Dynamic Object-Oriented Programming. In: USENIX ANNUAL TCL/TK WORKSHOP, 3., 1995, Toronto, Canada. **Proceedings...** [S.l.: s.n.], 1995. p.173–182.

WHEELIS, J. D. Process Control Communications: Token Bus, CSMA/CD or Token Ring? **ISA Trans**, [S.l.], v.32, n.2, p.193–198, July 1993.

ZIMMERMANN, H. OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection. **IEEECom**, [S.l.], v.COM-28, p.425–432, 1980.

ANEXO A SCRIPT DE SIMULAÇÃO - CASO 1

A seguir é apresentado o modelo básico dos *scripts* utilizados nas simulações do primeiro caso de estudo (Figura 3.4). Os identificadores mantidos entre “<” e “>” representam parâmetros alterados a cada rodada dos experimentos.

```
set opt(rede,controle) <est_controle>         ;#nro_estacoes rede de controle
set opt(tam_pacote,controle) 72               ;# menor quadro possivel - IEEE 802.3
set opt(periodo,controle) 0.001               ;# 1000 quadros/segundo
set opt(codepoint,controle) 10

set opt(rede,dados) 10
set opt(tam_pacote,dados) 1530
set opt(periodo,dados) 0.0002
set opt(codepoint,dados) 0

set opt(fim_geracao) 10.0

#############################################################################
# PROCEDIMENTOS #############################################################
#############################################################################

proc pcria_topologia {} {
    global nivel ns n fila qtrace l01 rf tf

    set n(switch) [$ns node]
    set n(controlador) [$ns node]

    puts $rf [format "c %d" [$n(controlador) id]]    ;# registro no arquivo de receptores

    $ns simplex-link $n(switch) $n(controlador) 100Mb 0.0000005 "dsRED/core"
    set fila(switch-controlador) [[$ns link $n(switch) $n(controlador)] queue]
    pconf_filaDiffS switch controlador

    $ns trace-queue $n(switch) $n(controlador) $tf

    pcria_rede_controle
    pcria_rede_dados
}

proc pcria_rede_controle {} {
    global ns n fila opt tf

    set no 0
    while {$no < $opt(rede,controle)} {
        set n(estacao,c,0,$no) [$ns node]
    set n(estacao,c,1,$no) [$ns node]
    $ns duplex-link $n(estacao,c,0,$no) $n(estacao,c,1,$no) 10000000000 0.000000001 DropTail

    $ns simplex-link $n(estacao,c,1,$no) $n(switch) 100Mb 0.0000005 dsRED/edge
    set fila(estacao,c,1,$no-switch) [[$ns link $n(estacao,c,1,$no) $n(switch)] queue]
    pconf_filaDiffS estacao,c,1,$no switch

    $ns trace-queue $n(estacao,c,1,$no) $n(switch) $tf

    $fila(estacao,c,1,$no-switch) addPolicyEntry -1 [$n(controlador) id] TSW2CM $opt(codepoint,controle) 1000000000
    $fila(estacao,c,1,$no-switch) addPolicyEntry -1 -1 TSW2CM 0 1000000000
    $fila(estacao,c,1,$no-switch) addPolicerEntry TSW2CM $opt(codepoint,controle) 0
    $fila(estacao,c,1,$no-switch) addPolicerEntry TSW2CM 0 0

    $ns simplex-link $n(switch) $n(estacao,c,1,$no) 100Mb 0.0000005 "dsRED/core"
    set fila(switch-estacao,c,1,$no) [[$ns link $n(switch) $n(estacao,c,1,$no)] queue]
    pconf_filaDiffS switch estacao,c,1,$no

    pconf_cbr estacao,c,0,$no controlador $opt(tam_pacote,controle) $opt(periodo,controle) 0.0001

    incr no
    }
}

proc pcria_rede_dados {} {
    global ns n fila opt tf rf

    set no1 0
    while {$no1 < [expr $opt(rede,dados) / 2]} {
```

```
        set no2 [expr $no1 + $opt(rede,dados)/2]

    set n(estacao,d,$no1) [$ns node]
    set n(estacao,d,$no2) [$ns node]

    $ns duplex-link $n(estacao,d,$no1) $n(switch) 100Mb 0.0000005 DropTail
    $ns trace-queue $n(estacao,d,$no1) $n(switch) $tf

    $ns duplex-link $n(switch) $n(estacao,d,$no2) 100Mb 0.0000005 DropTail
    $ns trace-queue $n(switch) $n(estacao,d,$no2) $tf

    pconf_cbr estacao,d,$no1 estacao,d,$no2 $opt(tam_pacote,dados) $opt(periodo,dados) 0.5

    puts $rf [format "d%d %d" $no1 [$n(estacao,d,$no2) id]]  ;# registra no arquivo receptores

    incr no1
    }
}

proc pconf_filaDiffS {n1 n2} {
    global n fila opt
    puts "Configurando fila $n1-$n2"
    $fila($n1-$n2) set numQueues 2
    $fila($n1-$n2) setNumPrec 1
    $fila($n1-$n2) setMREDMode DROP
    $fila($n1-$n2) setSchedularMode PRI
    $fila($n1-$n2) addPHBEntry $opt(codepoint,controle) 0 0
    $fila($n1-$n2) addPHBEntry $opt(codepoint,dados) 1 0
    $fila($n1-$n2) set limit_ [expr <est_controle>+100]
    $fila($n1-$n2) configQ 0 0 <est_controle> <est_controle> 0
    $fila($n1-$n2) configQ 1 0 100 100 0
}

proc pconf_cbr {origem destino tam_pacote intervalo rand} {
    global ns n sink udp cbr opt

    set ind [array size sink]

    set sink($ind) [new Agent/LossMonitor]
    $ns attach-agent $n($destino) $sink($ind)
    $sink($ind) clear

    set udp($ind) [new Agent/UDP]
    $udp($ind) set packetSize_ $tam_pacote
    $ns attach-agent $n($origem) $udp($ind)
    set cbr($ind) [new Application/Traffic/CBR]
    $cbr($ind) set packetSize_ $tam_pacote
    $cbr($ind) set interval_ $intervalo
    $cbr($ind) set random_ $rand
    $cbr($ind) attach-agent $udp($ind)
    $ns connect $udp($ind) $sink($ind)

    $ns at 0.0 "$cbr($ind) start"
    $ns at $opt(fim_geracao) "$cbr($ind) stop"
}

proc pfim {} {
    global ns tf
    $ns flush-trace
    close $tf
    exit 0
}

#############################################################################
# PRINCIPAL #################################################################
#############################################################################
set ns [new Simulator]

ns-random 0

set tf [open resultados/caso1.tr w]
set rf [open resultados/receptores.tr w]

pcria_topologia

$ns at 11.0 "$fila(switch-controlador) printStats"
$ns at 15.0 "pfim"

$ns run
```

ANEXO B SCRIPT DE SIMULAÇÃO - CASO 2

A mesma notação para os parâmetros de entrada (“<” e “>”) é mantida na apresentação do modelo de *scripts* utilizado para o segundo cenário de interesse (Figura 3.7).

```
set opt(rede,f0) <est_f0>     ;#nro_estacoes rede de controle - fluxo prioridade 0 - f0
set opt(tam_pacote,f0) 96     ;# tamanho do quadro - f0
set opt(periodo,f0) 0.001     ;# 1000 quadros/segundo - taxa de geracao - f0
set opt(codepoint,f0) 10

set opt(rede,f1) <est_f1>     ;#nro_estacoes rede de controle - fluxo prioridade 1 - f1
set opt(tam_pacote,f1) 72     ;# tamanho do quadro - f1
set opt(periodo,f1) 0.001     ;# 1000 quadros/segundo - taxa de geracao - f1
set opt(codepoint,f1) 11

set opt(rede,dados) 12
set opt(tam_pacote,dados) 1530
set opt(periodo,dados) 0.0002
set opt(codepoint,dados) 0
set opt(fim_geracao) 10.0

#############################################################################
# PROCEDIMENTOS #############################################################
#############################################################################
proc pcria_topologia {} {
    global ns n fila tf rf

    set n(switch1) [$ns node]
    set n(switch2) [$ns node]
    set n(controlador) [$ns node]

    puts $rf [format "c %d" [$n(controlador) id]]    ;# registro do no destino dos fluxos de controle

    $ns simplex-link $n(switch1) $n(switch2) 100Mb 0.0000005 "dsRED/core"
    set fila(switch1-switch2) [[$ns link $n(switch1) $n(switch2)] queue]
    $fila(switch1-switch2) set limit_ [expr <est_f0>*3+200]
    pconf_filaDiffS switch1 switch2

    $ns simplex-link $n(switch2) $n(controlador) 100Mb 0.0000005 "dsRED/core"
    set fila(switch2-controlador) [[$ns link $n(switch2) $n(controlador)] queue]
    pconf_filaDiffS switch2 controlador

    $ns trace-queue $n(switch2) $n(controlador) $tf

    pcria_rede_controle
    pcria_rede_dados
}

proc pcria_rede_controle {} {
    global ns n fila opt tf

    foreach fluxo {f0 f1} {
       set no 0
       while {$no < $opt(rede,$fluxo)} {
          set n($fluxo,0,$no) [$ns node]
          set n($fluxo,1,$no) [$ns node]
          $ns duplex-link $n($fluxo,0,$no) $n($fluxo,1,$no) 10000000000 0.000000001 DropTail

          $ns simplex-link $n($fluxo,1,$no) $n(switch1) 100Mb 0.0000005 dsRED/edge
          set fila($fluxo,1,$no-switch1) [[$ns link $n($fluxo,1,$no) $n(switch1)] queue]
          pconf_filaDiffS $fluxo,1,$no switch1

          $ns trace-queue $n($fluxo,1,$no) $n(switch1) $tf

          $fila($fluxo,1,$no-switch1) addPolicyEntry -1 [$n(controlador) id] TSW2CM $opt(codepoint,$fluxo) 1000000000
          $fila($fluxo,1,$no-switch1) addPolicyEntry -1 -1 TSW2CM 0 1000000000
          $fila($fluxo,1,$no-switch1) addPolicerEntry TSW2CM $opt(codepoint,$fluxo) 0
          $fila($fluxo,1,$no-switch1) addPolicerEntry TSW2CM 0 0

          $ns simplex-link $n(switch1) $n($fluxo,1,$no) 100Mb 0.0000005 "dsRED/core"
          set fila(switch1-$fluxo,1,$no) [[$ns link $n(switch1) $n($fluxo,1,$no)] queue]
          pconf_filaDiffS switch1 $fluxo,1,$no

          pconf_cbr $fluxo,0,$no controlador $opt(tam_pacote,$fluxo) $opt(periodo,$fluxo) 0.0001

          incr no
```

```
       }
    }
}

proc pcria_rede_dados {} {
    global ns n fila opt tf rf

    set no1 0
    while {$no1 < [expr $opt(rede,dados) / 2]} {
        set no2 [expr $no1 + $opt(rede,dados)/2]

        set n(dados,$no1) [$ns node]
        set n(dados,$no2) [$ns node]

        $ns duplex-link $n(dados,$no1) $n(switch1) 100Mb 0.0000005 DropTail
        $ns trace-queue $n(dados,$no1) $n(switch1) $tf

        $ns duplex-link $n(switch2) $n(dados,$no2) 100Mb 0.0000005 DropTail
        $ns trace-queue $n(switch2) $n(dados,$no2) $tf

        pconf_cbr dados,$no1 dados,$no2 $opt(tam_pacote,dados) $opt(periodo,dados) 0.0001

        incr no1
    }
}

proc pconf_filaDiffS {n1 n2} {
    global n fila

    $fila($n1-$n2) set numQueues 3
    $fila($n1-$n2) setNumPrec 1
    $fila($n1-$n2) setMREDMode DROP
    $fila($n1-$n2) setSchedularMode PRI
    $fila($n1-$n2) addPHBEntry 10 0 0
    $fila($n1-$n2) addPHBEntry 11 1 0
    $fila($n1-$n2) addPHBEntry 0 2 0
    $fila($n1-$n2) configQ 0 0 <est_f0> <est_f0> 0
    $fila($n1-$n2) configQ 1 0 <est_f1> <est_f0> 0
    $fila($n1-$n2) configQ 2 0 1000 1000 0
}

proc pconf_cbr {origem destino tam_pacote intervalo rand} {
    global ns n sink udp cbr opt

    set ind [array size sink]
    set sink($ind) [new Agent/LossMonitor]
    $ns attach-agent $n($destino) $sink($ind)
    $sink($ind) clear

    set udp($ind) [new Agent/UDP]
    $udp($ind) set packetSize_ $tam_pacote
    $ns attach-agent $n($origem) $udp($ind)
    set cbr($ind) [new Application/Traffic/CBR]
    $cbr($ind) set packetSize_ $tam_pacote
    $cbr($ind) set interval_ $intervalo
    $cbr($ind) set random_ $rand
    $cbr($ind) attach-agent $udp($ind)
    $ns connect $udp($ind) $sink($ind)
    $ns at 0.0 "$cbr($ind) start"
    $ns at $opt(fim_geracao) "$cbr($ind) stop"
}

proc calc_utilizacao_s1_s2 {} {
   global fila opt
   set pktsf0 [expr [$fila(switch1-switch2) getStat pkts $opt(codepoint,f0)] -
                    [$fila(switch1-switch2) getStat drops $opt(codepoint,f0)]]
   set pktsf1 [expr [$fila(switch1-switch2) getStat pkts $opt(codepoint,f1)] -
                    [$fila(switch1-switch2) getStat drops $opt(codepoint,f1)]]
   set pktsdados [expr [$fila(switch1-switch2) getStat pkts $opt(codepoint,dados)] -
                       [$fila(switch1-switch2) getStat drops $opt(codepoint,dados)]]
   set totalbits [expr $pktsf0*$opt(tam_pacote,f0)*8+$pktsf1*$opt(tam_pacote,f1)*8+$pktsdados*$opt(tam_pacote,dados)*8]
   set maxbits [expr $opt(fim_geracao) * 100000000]
   set utilizacao [expr $totalbits/$maxbits]
   puts [format "u: s1->s2 = <est_f0> %2.1f" $utilizacao]
}

proc pfim {} {
    global ns tf

    $ns flush-trace
    close $tf
    calc_utilizacao_s1_s2
    exit 0
}
#############################################################################
# PRINCIPAL #################################################################
#############################################################################
remove-all-packet-headers       ;# removes all except common
add-packet-header IP Message    ;# hdrs reqd for cbr traffic

set ns [new Simulator]
ns-random 0
set tf [open resultados/caso2.tr w]
set rf [open resultados/receptores.tr w]

pcria_topologia
$ns at 11.0 "$fila(switch1-switch2) printStats"
$ns at 15.0 "pfim"
$ns run
```