Este exemplar corresponde à redação final da Tese/Dissertação devidamente corrigida e defendida por: Sérgio Araújo Yunes
e aprovada pela Banca Examinadora.
Campinas, 22 de janeiro de 2002.
COORDENADOR DE PÓS-GRADUAÇÃO
CPG-IC

1002066 51

Policiamento de Tráfego IP em Redes ATM

Sérgio Araújo Yunes

Dissertação de Mestrado

UNICAMP
BIBLIOTECA CENTRAL
SEÇÃO CIRCULANTE

Instituto de Computação
Universidade Estadual de Campinas

Policiamento de Tráfego IP em Redes ATM

Sérgio Araújo Yunes

Outubro de 2001

Banca Examinadora:

- Prof. Dr. Nelson Luis Saldanha da Fonseca
  Instituto de Computação - UNICAMP (Orientador)
- Prof. Dr. Carlos Becker Westphall
  Departamento de Ciência da Computação - UFSC
- Prof. Dr. Edmundo Roberto Mauro Madeira
  Instituto de Computação - UNICAMP
- Prof. Dr. Célio Cardoso Guimarães
  Instituto de Computação - UNICAMP (Suplente)

UNIDADE BC
N.º CHAMADA:
UNICAMP
Y92p
V.
TOMBO BC/47569
PROC. 837/02
C [ ] D [X]
PREÇO R$ 11,00
DATA 06-02-02
N.º CPD

CM00163103-7

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Yunes, Sérgio Araújo

Y92p Policiamento de tráfego IP em redes ATM / Sérgio Araújo Yunes - - Campinas, [S.P. :s.n.], 2001.

Orientador : Nelson Luis Saldanha da Fonseca

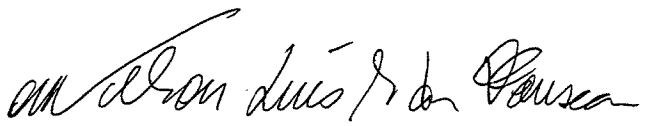
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Redes digitais de serviços integrados. 2. Redes de computação. 3. Fractais. I. Fonseca, Nelson Luis Saldanha da. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Policiamento de Tráfego IP em Redes ATM

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Sérgio Araújo Yunes e aprovada pela Banca Examinadora.

Campinas, 11 de outubro de 2001.



Prof. Dr. Nelson Luis Saldanha da Fonseca
Instituto de Computação - UNICAMP
(Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 11 de outubro de 2001, pela Banca Examinadora composta pelos Professores Doutores:

Prof. Dr. Carlos Becker Westphall
UFSC

Prof. Dr. Edmundo Roberto Mauro Madeira
UNICAMP

Prof. Dr. Nelson Luís Saldanha da Fonseca
IC - UNICAMP

© Sérgio Araújo Yunes, 2001.
Todos os direitos reservados.

Resumo

Desde as primeiras recomendações da ITU-T sobre o padrão ATM no final da década de 80, muitos trabalhos foram realizados com intuito de fornecer melhor suporte para tráfego IP sobre ATM. IP foi desenvolvido sobre as bases de um modelo sem conexão, enquanto ATM segue um modelo orientado a conexão, o que traz dificuldades na integração destas duas tecnologias.

Esta dissertação propõe um novo mecanismo de policiamento de tráfego, chamado de Balde Furado para Pacote (BFP), que tenta minimizar a perda de pacotes IP e de células ATM. Mostra-se que o mecanismo de descarte seletivo de células MFIFD (*Modified First-In-First-Drop*) quando usado com BFP gera os mesmos resultados das políticas de descarte de pacotes. Além disso, estuda-se a integração entre mecanismos de descarte de pacote e de célula.

Abstract

Since the first ITU-T recommendation on ATM in the late 80's, the support of IP traffic over ATM has been a focus of attention of the research community. IP follows a connectionless model, while ATM a connection-oriented model, which bring challenges to the integration of these two technologies.

This work proposes a policing algorithm called Packet Leaky Bucket (PLB) which attempts to minimise the loss of IP packets and ATM cells. It is shown that the push-out mechanism MFIFD (Modified First-In-First-Drop) used with BFP leads to the same results of packet discarding policies. Furthermore, the integration of packet and cell discard mechanisms is studied.

Agradecimentos

Agradeço...

Primeiramente, à Deus pelo dom da vida e por conduzir sempre os meus passos com amor supremo, sendo o motivo maior de toda felicidade e existência.

Aos meus amigos de república que fizeram deste tempo longe de casa uma acolhedora e divertida convivência.

Gostaria de agradecer a FAPESP pelo suporte financeiro ao meu trabalho porque, sem ele, não haveria condições de realizá-lo.

Agradeço aos meus pais, André Luiz e Maria Neuza, e as minhas irmãs pelo incentivo e carinho em todos os momentos. A vocês dedico este trabalho.

Ao meu orientador Prof. Dr. Nelson Luiz Saldanha da Fonseca, pela excelente orientação. Suas preciosas idéias, seu entusiasmo e sua dedicação foram fundamentais para a realização deste trabalho.

Ao Prof. Altigran, pela sua dedicação e serenidade que motivou a muitos.

Agradeço aos colegas, professores, funcionários, enfim, a toda a comunidade do Instituto de Computação da UNICAMP, pelo ambiente agradável, organizado e propício ao trabalho.

Aos colegas de mestrado pela convivência, apoio e amizade.

Aos demais professores do Instituto de Computação e aos funcionários.

Ao CENAPAD/SP, pelo apoio computacional.

Conteúdo

Resumo vii

Abstract viii

Agradecimentos ix

1 Introdução 1

2 Tráfego de Auto-Semelhante 3
- 2.1 Auto-Semelhança: Conceitos e Propriedades . . . 3
  - 2.1.1 Processos Auto-Semelhantes de Segunda-Ordem . . . 4
  - 2.1.2 Dependência de Longa-Duração . . . 4
  - 2.1.3 Propriedades . . . 5
- 2.2 Modelos de fonte *on/off* com caudas longas . . . 8
  - 2.2.1 Movimento Browniano Fractal . . . 9
- 2.3 Métodos para Estimativa do Parâmetro H . . . 10

3 Gerenciamento de Tráfego em Redes ATM 13
- 3.1 Categorias de Serviço . . . 14
- 3.2 Controle de Admissão de Conexões . . . 15
- 3.3 Mecanismos de Policiamento de Tráfego . . . 16
- 3.4 Descarte Seletivo de Células . . . 18
- 3.5 Políticas de Descarte de Pacotes para Redes ATM . . . 22

4 Policiamento para Tráfego IP sobre ATM 25
- 4.1 Princípios do Mecanismo Balde Furado para Pacote . . . 26
- 4.2 Validação do Mecanismo Balde Furado para Pacote . . . 29
- 4.3 O Modelo de Simulação e Tráfego . . . 30
- 4.4 Resultados Numéricos . . . 33
  - 4.4.1 Tráfego Sintético . . . 33

4.4.2 Tráfego Real 44
4.4.3 Tráfego Sintético X Tráfego Real 52

5 Descarte de Informações em Redes ATM **53**
5.1 Descarte de Células vs. Descarte de Pacotes 54
5.1.1 Tráfego Sintético 54
5.1.2 Tráfego Real 64
5.2 Integração entre Descarte de Células e Descarte de Pacotes 73
5.2.1 Integração entre Taildrop e MFIFD 73
5.2.2 Integração entre EPD e MFIFD 74
5.2.3 Tráfego Real 77
5.3 O Mecanismo Balde Furado para Pacote no Contexto da Classe de Serviço GFR 83
5.3.1 SF-GCRA e comutadores ATM com descarte de células 85

6 Conclusões **91**
6.1 Principais Contribuições 92
6.2 Trabalhos Futuros 93

Bibliografia **95**

Lista de Tabelas

4.1 Sumário dos *traces* de tráfego IP agregado. 31
4.2 Sumário dos parâmetros utilizados nas simulações com tráfego sintético, fluxo agregado. 34
4.3 Sumário dos parâmetros utilizados nas simulações com tráfego sintético, fluxo de interferência. 39
4.4 Sumário dos parâmetros utilizados nas simulações com tráfego real, fluxo agregado. 44
4.5 Sumário dos parâmetros utilizados nas simulações com tráfego real, fluxo de interferência. 48

Lista de Figuras

2.1 Função de autocorrelação de um processo exatamente auto-semelhante de segunda-ordem com parâmetro de Hurst $H = 1 - \beta/2$. 5
2.2 Exemplos de estruturas de autocorrelação a) processo com dependência de curta duração b) processo com dependência de longa duração. 6

3.1 Exemplos de políticas de organização de *buffer*. 20
3.2 Um exemplo de perda de célula devido a diferentes distribuições de tamanho de fila produzidas pelas políticas de descarte LIFD e FIFD. 22

4.1 Balde Furado para Pacote - BFP. 27
4.2 Modelo de simulação utilizado para validar o balde furado para pacote. . . 29
4.3 Modelo de simulação e tráfego. 30
4.4 *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego sintético . . 36
4.5 *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego sintético. . 38
4.6 *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético. 41
4.7 *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético. 43
4.8 *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real. 46
4.9 *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real. 47
4.10 *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 50
4.11 *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 51

5.1 *Goodput de célula vs.* Carga num ambiente com 1 nó e tráfego sintético. . . 56
5.2 *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético 59
5.3 *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego sintético. . 62
5.4 *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético. 64

5.5 *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real. 66
5.6 *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 68
5.7 *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real. 70
5.8 *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 72
5.9 *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real. 78
5.10 *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 79
5.11 *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real. 81
5.12 *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real. 82
5.13 *Goodput* de pacote e célula *vs.* Carga num ambiente composto por 1 nó, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254. 86
5.14 *Goodput* de pacote e célula *vs.* Carga num ambiente composto por 1 nó, *buffer* = 150 e tráfego real - *trace* 12SDC-964130254. 87
5.15 *Goodput* de pacote e célula *vs.* Taxa de geração de fichas num ambiente composto por 1 nó, carga 0.9, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254. 89
5.16 *Goodput* de pacote e célula *vs.* Tamanho do balde num ambiente composto por 1 nó, carga 0.9, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254. . 90

Capítulo 1

Introdução

É cada vez mais comum a situação na qual a pilha de protocolos TCP/IP é usada sobre redes ATM. Redes ATM possuem funções para prover garantias de Qualidade de Serviço (QoS - *Quality of Service*) e possuem uma grande variedade de mecanismos para controle de tráfego. Além disso, ATM utiliza uma pequena unidade de transferência, células de 53-*bytes*, fazendo com que pacotes IP sejam fragmentados na fonte.

O crescente uso de TCP/IP sobre ATM gerou uma vasta quantidade de pesquisa. Em essência, a questão é como melhor gerenciar o tamanho de segmentos TCP, tamanho da janela e políticas de controle de congestionamento de um lado, e qualidade de serviço e mecanismos de controle de tráfego ATM do outro, a fim de alcançar alto desempenho para tráfego TCP/IP, alocação justa entre as várias conexões TCP, e uso eficiente da rede ATM subjacente. Entretanto, existem diferenças marcantes na forma de operação destas duas tecnologias. IP foi desenvolvido sobre as bases de um modelo sem conexão (*connectionless*), enquanto ATM foi originalmente projetado para serviços orientados a conexão. Assim sendo, desenvolver técnicas para transportar tráfego de aplicações IP sobre redes ATM é de fundamental importância.

Em redes ATM, mecanismos de controle de tráfego foram definidos com o objetivo de garantir as requisições de QoS do usuário. Entre esses mecanismos menciona-se: mecanismo de controle de admissão, mecanismo de policiamento e mecanismo de descarte seletivo. Implementações convencionais de policiamento em redes ATM são baseadas no *Generic Cell Rate Algorithm* (GCRA), definido pelo ATM Forum [The99]. Este algoritmo atua diretamente sobre células da camada ATM sem considerar o pacote que estas células pertecem. Um mecanismo de descarte seletivo descarta células em uma situação de transbordo, de acordo com o nível de prioridade da célula. O nível de prioridade expressa a relevância da célula, por exemplo, em uma codificação de vídeo, células com prioridade baixa são menos significantes para a qualidade da imagem. Além do descarte seletivo em nível de células, várias políticas de descarte de pacotes foram definidas a fim de preservar

a integridade dos pacotes no nível da camada de transporte. Para algumas aplicações, tal como IP sobre ATM, se uma das células de um pacote é perdida o pacote inteiro deve ser retransmitido. Portanto, transmitir células de um pacote corrompido[1] ocupa banda passante e pode, conseqüentemente, aumentar o congestionamento na rede. Assim sendo, mecanismos de descarte seletivo de célula se preocupam com a utilização a nível de *link* de rede sem levar em consideração a integridade do pacote da camada de transporte, enquanto que as políticas de descarte de pacote levam em consideração o desempenho a nível da camada de transporte.

Nesta dissertação, propõem-se um mecanismo de policiamento de tráfego, chamado de Balde Furado para Pacote, cujo objetivo é minimizar a perda de pacotes e de células, bem como manter as especificações do contrato de tráfego. Investiga-se a eficácia deste mecanismo tanto a nível de pacote como a nível de célula, comparando-o com o mecanismo balde furado tradicional (GCRA). Em seguida, compara-se o desempenho entre o mecanismo de descarte de célula MFIFD (*Modified First-In-First-Drop*) e as políticas de descarte de pacote *Taildrop* e EPD (*Early Packet Discard*) e avalia-se se existem vantagens em se integrar políticas de descarte de pacote com mecanismos de descarte de célula. Para obter os resultados, analisa-se o compromisso entre o *goodput* de célula (a proporção de células boas transmitidas pelo número total de células que chegaram na fila) e o *goodput* de pacote (a proporção de pacotes bons transmitidos pelo número total de pacotes que chegaram na fila), utilizando tráfego sintético e tráfego *Internet* real. Estende-se a análise também para um ambiente de rede formado por vários nós.

Essa dissertação está organizada da seguinte maneira: no capítulo 2, são apresentados os principais conceitos de processos auto-semelhantes. No capítulo 3, são introduzidos os mecanismos de gerenciamento de tráfego em redes ATM. No capítulo 4, introduz-se o mecanismo de policiamento alternativo balde furado para pacote, e apresenta-se os resultados da avaliação deste em relação ao mecanismo balde furado convencional. No capítulo 5, compara-se o desempenho entre descarte seletivo de célula e descarte de pacotes. Além disso, propõem-se a integração entre mecanismos de descarte seletivo de célula e políticas de descarte de pacote. No capítulo 6, conclusões são derivadas, e propõe-se alguns trabalhos futuros.

[1] Pacote que teve pelo menos uma de suas células perdida durante a transmissão.

Capítulo 2

Tráfego de Auto-Semelhante

Após as conclusões do estudo realizado por Leland et al. [LTWW93], no qual se estabeleceu a base para se considerar o fenômeno de auto-semelhança no tráfego de redes, uma gama de outros trabalhos investigou este fenômeno. Estes trabalhos demonstraram que o tráfego de rede atual é auto-semelhante e exibe o fenômeno de dependência de longa-duração (tráfego em rajadas sobre uma larga faixa de escalas de tempo). Estas conclusões contrastam com os paradigmas até então utilizados para modelagem de tráfego como, por exemplo, o paradigma de que tráfego de dados é adequadamente descrito por certos modelos Markovianos. A descoberta e, mais importante, a formulação concisa e o reconhecimento de que tráfego de redes pode não exibir as propriedades apresentadas pelos paradigmas até então utilizados na modelagem de tráfego, tem significativamente influenciado o panorama das pesquisas em teletráfego, provocando uma reavaliação de algumas de suas premissas fundamentais.

Neste capítulo são apresentados os principais conceitos relacionadas com processos auto-semelhantes. A seção 2.1 conceitua auto-semelhança e apresenta algumas propriedades dos processos auto-semelhantes. O fenômeno de auto-semelhança no nível de fonte de dados é abordado na seção 2.2, juntamente com o movimento Browniano fractal (mBf). Ainda na seção 2.2, mostra-se um método para geração de dados auto-semelhantes baseado no mBf. Na seção 2.3, discute-se alguns métodos para se estimar o grau de auto-semelhança de um processo.

2.1 Auto-Semelhança: Conceitos e Propriedades

Auto-semelhança ou fractal foram conceitos inicialmente descritos por Mandelbrot [Man82]. Este conceito descreve o fenômeno onde certa propriedade de um objeto é preservada em diferentes escalas no espaço e/ou no tempo. Se um objeto é auto-semelhante ou fractal, então suas partes quando ampliadas assemelham-se a forma do todo.

Para o propósito de modelagem de tráfego, onde a variação aleatória é um componente essencial, faz-se uso de auto-semelhança estocástica. Auto-semelhança estocástica admite a infusão de não-determinismo, tal como acontece em medições de tráfego real (*traces*), não obstante, é uma propriedade que pode ser ilustrada visualmente.

Processos estocásticos auto-semelhantes são invariantes em sua estrutura de correlação em diferentes escalas de tempo. De fato, a função de autocorrelação é um importante aspecto na formulação matemática de processos auto-semelhantes, conforme veremos adiante. Assim, olhando-se vários gráficos de um processo estocástico auto-semelhante em diferentes escalas de tempo eles irão parecer muito "semelhantes", ainda que não seja com finos detalhes como acontece com fractais determinísticos.

2.1.1 Processos Auto-Semelhantes de Segunda-Ordem

Auto-semelhança de segunda-ordem representa um dos mais comuns *frameworks* para modelagem de tráfego de rede. Um processo $X(t)$ é chamado *exatamente auto-semelhante de segunda-ordem* com parâmetro de *Hurst* H $(1/2 < H < 1)$ se sua função de autocorrelação e:[Cox84] [TG97]:

$$r(k) = \frac{\sigma}{2}[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}], \quad k = 1, 2, 3, \ldots \tag{2.1}$$

e $X(t)$ é chamado *assintoticamente auto-semelhante de segunda-ordem* com parâmetro H, se para todo $k = 1, 2, 3, \ldots$

$$\lim_{m\to\infty} r^{(m)}(k) = \frac{\sigma}{2}[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}] \tag{2.2}$$

A equação 2.1 implica que $r(k) = r^{(m)}(k)$ para todo $m \geq 1$. Dessa forma, auto-semelhança de segunda-ordem captura a propriedade de que a estrutura de correlação é exatamente ou assintoticamente preservada durante agregação de tempo.

2.1.2 Dependência de Longa-Duração

Vamos retornar a definição de auto-semelhança de segunda ordem e sua autocovariância $r(k)$. Seja $g(k) = r(k)/\sigma^2$ sua *função de autocorrelação*. Então para $0 < H < 1$, $H \neq \frac{1}{2}$, temos que:

$$g(k) \sim H(2H-1)k^{2H-2}, \quad k \to \infty \tag{2.3}$$

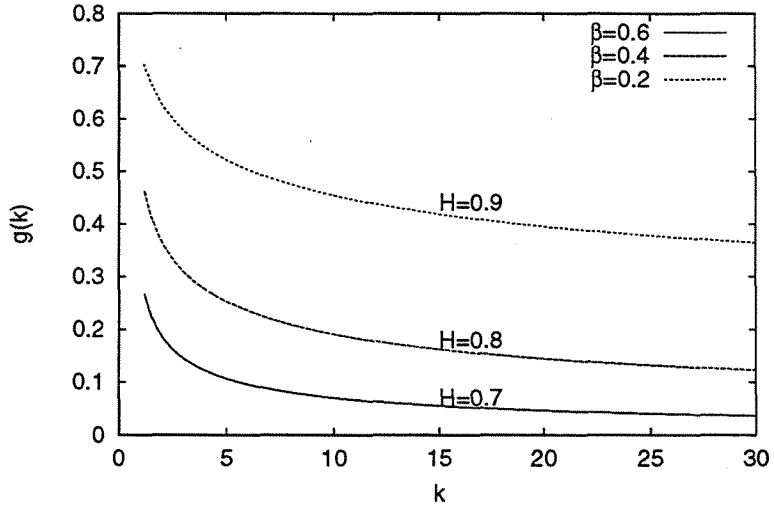


Figura 2.1: Função de autocorrelação de um processo exatamente auto-semelhante de segunda-ordem com parâmetro de Hurst $H = 1 - \beta/2$.

No caso particular de $\frac{1}{2} < H < 1$, $g(k)$ comporta-se assintoticamente como $ck^{-\beta}$ para $0 < \beta < 1$, onde $c > 0$ é uma constante, $\beta = 2 - 2H$ (ou $H = 1 - \beta/2$), e temos que:

$$\sum_{k=-\infty}^{\infty} g(k) = \infty \tag{2.4}$$

ou seja, quando a função de autocorrelação $g(k)$ decresce hiperbolicamente tal que a condição (2.4) é satisfeita, então temos um processo com dependência de longa-duração (LRD - *Long Range Dependence*). Por outro lado, caso a função de autocorrelação seja totalizável, então temos um processo com dependência de curta-duração (SRD - *Short Range Dependence*).

A Figura 2.1 mostra a função de autocorrelação $g(k)$ como função de k para três valores de β. Note que a função de autocorrelação decresce lentamente, ou seja, hiperbolicamente.

2.1.3 Propriedades

A auto-semelhança é expressa matematicamente em um número de modos equivalentes:

(i) a variância da média amostral decresce mais lentamente do que a variância do tamanho da amostra (*variância decrescendo lentamente*), ou seja, $var(X^{(m)}) \sim am^{-\beta}$, com $m \to \infty$, com $0 < \beta < 1$;

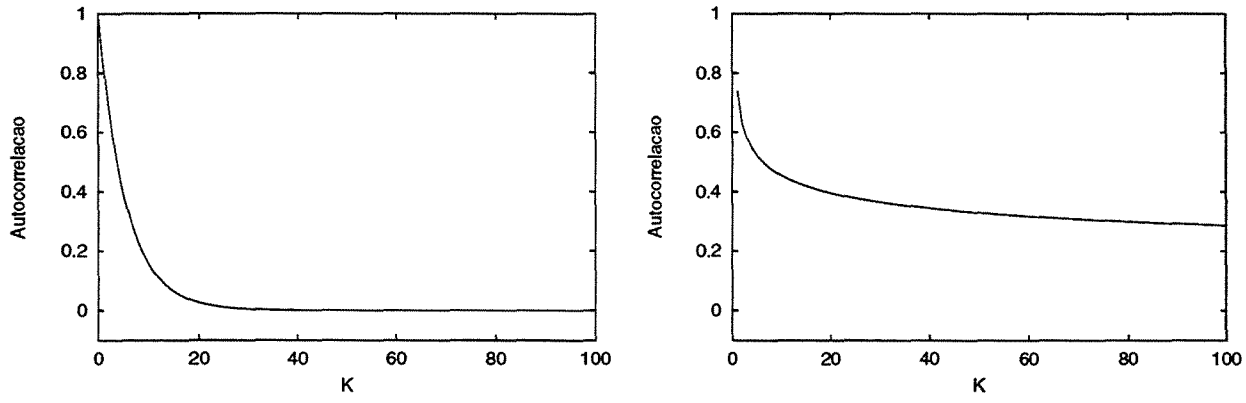


Figura 2.2: Exemplos de estruturas de autocorrelação a) processo com dependência de curta duração b) processo com dependência de longa duração.

(ii) as autocorrelações decrescem hiperbolicamente ao invés de exponencialmente, implicando em uma função de autocorrelação não totalizável, conforme a condição (2.4). Veja Figura 2.2b, e

(iii) a densidade espectral (Transformada de Fourier) $f(.)$ perto da origem comporta-se como $f(\lambda) \sim a\lambda^{-\gamma}$, para $\lambda \to 0$, $0 < \gamma < 1$ e $\gamma = 1 - \beta$.

Os processos auto-semelhantes contrapõem-se aos modelos de tráfego tradicionais devido a estes serem caracterizados por:

(i) a variância da média amostral decresce com a variância do tamanho da amostra, ou seja, $var(X^{(m)}) \sim am^{-1}$, com $m \to \infty$;

(ii) uma função de autocorrelação que decresce exponencialmente ($g(k) \sim \rho^k, 0 < \rho < 1$), implicando numa função de autocorrelação totalizável $\sum_k g(k) < \infty$, dependência de curta-duração. Veja a figura 2.2a, e

(iii) a densidade espectral limitada na origem.

A Figura 2.2 compara o comportamento da autocorrelação de um processo com dependência de curta-duração (figura 2.2a) com o comportamento de um processo com dependência de longa-duração (figura 2.2b). Nota-se que no caso de dependência de curta duração tem-se valores expressivos de autocorrelação apenas para pequenos valores de k ($k < 10$), enquanto que para dependência de longa duração continua-se a ter valores

expressivos de autocorrelação mesmo para valores maiores de k (veja também a figura 2.1).

A característica mais notável dos processos auto-semelhantes de segunda-ordem é que seus processos agregados $X^{(m)}$ possuem uma estrutura de correlação não degenerativa, com $m \to \infty$. A existência de uma estrutura de correlação não degenerativa para os processos $X^{(m)}$, com $m \to \infty$, está em completo contraste com os modelos típicos de tráfego de pacotes até então considerados na literatura, os quais têm processos agregados, $X^{(m)}$, que tendem para ruído puro de segunda-ordem, ou seja, para todo $k \geq 1$,

$$r^{(m)}(k) \to 0, \quad \text{com } m \to \infty, \text{ para } k \geq 1. \tag{2.5}$$

Uma das grandes vantagens dos processos auto-semelhantes é a parcimonidade do número de parâmetros para representar uma série temporal de centenas de milhares de observações. No caso de processos que não são auto-semelhantes, existe a necessidade de se ter um grande número de parâmetros para modelar longas séries. Em contraste, a componente de longa duração pode ser modelada com um único parâmetro quando se usa processos auto-semelhantes. ry

Dois processos estocásticos que produzem representações elegantes de fenômenos auto-semelhantes são: o ruído Gaussiano fractal (*fractional Gaussian noise*) e os processos fractal auto-regressivo de médias móveis integradas (*fractional autoregressive integrated moving-average processes* – F-ARIMA) . O ruído Gaussiano fractal $X = (X_k : k \geq 0)$ com parâmetro $H \in (0,1)$ foi introduzido em [MN68], e é um processo Gaussiano estacionário μ, variância σ^2, e função de autocorrelação $r(k) = 1/2(|k+1|^{2H} - |k|^{2H} + |k-1|^{2H}), k > 0$. É possível mostrar que o ruído Gaussiano fractal é exatamente auto-semelhante de segunda-ordem com parâmetro de auto-semelhança H entre $1/2 < H < 1$. A série de ruídos Gaussianos fractais $X_i, i > 1$ é o incremento do *movimento Browniano fractal* (mBf), isto é $X_i = B_H(i+1) - B_H(i), i \geq 1$ (mais detalhes sobre mBf na seção 2.2.1).

Os *processos fractais ARIMA(p,d,q)* são uma generalização da classe de modelos Box-Jenkins [BJ76], onde se permite que o parâmetro d assuma valores não inteiros. São processos assintoticamente auto-semelhantes de segunda-ordem com parâmetro auto-semelhante $d + 1/2$, onde $0 < d < 1/2$. O F-ARIMA foi usado para modelar tráfego de vídeo VBR (*Variable Bit Rate*) [BSTW95, HDLK95] e o desempenho de fila do F-ARIMA $(1, d, 0)$ foi analisado através de simulação para um servidor (*First Come First Serve* - FCFS)[AM95].

2.2 Modelos de fonte *on/off* com caudas longas

Diversos estudos de medidas de tráfego de uma variedade de tipos de redes demonstraram que o tráfego atual de rede tem natureza auto-semelhante. Apesar dessas descobertas faltava compreender qual a explicação física para que o tráfego agregado observado tivesse esse comportamento. Willinger et. al [WTE96, WTSW97] fornecem uma explicação simples e plausível em termos de tráfego de fontes individuais que formam o tráfego agregado. Desenvolvendo uma abordagem proposta originalmente por Mandelbrot [Man69], eles mostram que a superposição de muitas fontes *on/off* independentes e identicamente distribuídas (*i.i.d*), cada uma das quais exibindo um fenômeno chamado de efeito Noé (*Noah effect*), resulta em um mBf, que é um processo exatamente auto-semelhante de segunda-ordem (mais detalhes sobre mBf na seção 2.2.1).

Uma fonte *on/off* idealizada é uma fonte onde períodos de geração de pacotes (períodos *on*) podem ser sucedidos por outros períodos *on*. Da mesma forma períodos sem geração de pacotes (períodos *off*) podem ser sucedidos por outros períodos *off*. Em um modelo *on/off* de alternância estrita, um período *on* não pode ser seguido de outro período *on*, e um período *off* por outro período *off*. Os modelos de fontes *on/off* de alternância estrita são comumente considerados na literatura de comunicações. Em [WTE96] foram demonstradas as condições para que fontes *on/off* idealizadas gerassem um processo auto-semelhante, e em [WTSW97] os mesmos resultados foram obtidos usando fontes de alternância estrita. Apesar dos resultados em [WTE96] serem essencialmente os mesmos apresentados em [WTSW97], a prova da convergência com fontes *on/off* com alternância estrita é muito mais complexa. Uma prova rigorosa é fornecida em [WTS97].

O efeito Noé para um modelo de fonte *on/off* individual significa longos períodos *on* e longos períodos *off* com probabilidade não negligenciável. Em outras palavras, o efeito Noé implica que cada fonte *on/off* exibe características que encerram uma grande extensão de escalas de tempo. O efeito Noé é sinônimo de síndrome de variância infinita – a observação empírica de que muitos fenômenos naturais podem ser bem descritos usando distribuições com variância infinita (para referências veja [ST94, TL86, WTE96]). Distribuições com cauda longa, ou seja, distribuições que decrescem hiperbolicamente, apresentam síndrome de variância infinita. Uma variável aleatória Z tem uma distribuição com cauda longa se:

$$Pr(Z > x) \sim cx^{-\alpha}, \quad x \to \infty \tag{2.6}$$

onde $0 < \alpha < 2$ é chamado parâmetro de forma (*shape parameter*) e c é uma constante positiva. Assim, a cauda da distribuição, assintoticamente, decresce hiperbolicamente. Além da variância infinita, outra característica importante de distribuições com cauda longa é que elas também têm média infinita quando $0 < \alpha \leq 1$. Uma distribuição

que satisfaz as condições (2.6) é a distribuição de Pareto (veja [ST94]). Sua função de distribuição é dada por:

$$Pr[Z \leq x] = 1 - \left(\frac{b}{x}\right)^{\alpha}, \quad b \leq x \tag{2.7}$$

onde $0 < \alpha < 2$ é o parâmetro de forma e b é chamado de parâmetro de localização. A média é dada por $\alpha b/(\alpha - 1)$. A principal característica de uma variável aleatória seguindo uma distribuição de cauda longa é que ela exibe extrema variação. Na prática isso quer dizer que uma distribuição de cauda longa gera valores bem elevados com uma probabilidade não negligenciável de modo que uma amostragem de dessa distribuição resulta em um conjunto onde a maioria dos valores são "pequenos" e o restante são valores extremamente "grandes".

Uma relação simples entre o parâmetro α e o parâmetro H é dada por $H = (3-\alpha)/2$, onde H mede o grau de auto-semelhança do fluxo do tráfego agregado [WTSW97].

2.2.1 Movimento Browniano Fractal

O movimento Browniano ordinário, $B(t)$, descreve o movimento de uma partícula em um líquido sujeito a colisões e outras forças [Pap78]. É uma função aleatória real com incrementos Gaussianos independentes tal que:

$$E[B(t+s) - B(t)] = 0 \qquad Var[B(t+s) - B(t)] = \sigma^2|s| \tag{2.8}$$

Mandelbrot define um movimento Browniano fractal (mBf) como sendo o movimento médio de $dB(t)$ no qual incrementos que ultrapassam $B(t)$ são ponderados por $(t-s)^{H-1/2}$ [MN68].

Definição: Seja H tal que $0 < H < 1$. O mBf é definido pela integral de Weyl de $B(t)$.

$$B_H(t) = \frac{1}{\Gamma(H+1/2)} \int_{-\infty}^{o} ((t-s)^{H-1/2} - (-s)^{H-1/2}) dB(s) + \int_0^t (t-s)^{H-1/2} dB(s) \tag{2.9}$$

Essa equação tende para o movimento Browniano ordinário se $H = 1/2$. Sua propriedade auto-semelhante é baseada no fato de que $B_H(\rho s)$ é idêntica em distribuição a $\rho^h * B_H(s)$. Os incrementos do mBf, Y_j, formam uma seqüência estacionária chamada *ruído Gaussiano fractal (rGf)*:

$$Y_j = B_H(j+1) - B_H(j), \quad j = \ldots, -1, 0, 1, \ldots \tag{2.10}$$

Esses incrementos não são independentes a não ser que se tenha movimento Browniano puro, *i.e.* $H = 1/2$. Além disso, a lei de Hurst estabelece que $Var[B_H(t+s) - B_H(t)] = \sigma s^{2H}$, *i.e.*, um modelo de chegada mBf é também capaz de capturar a alta *variabilidade* inerente exibida pelo tráfego de rede real.

O mBf normalizado com parâmetro $H \in [\frac{1}{2}, 1)$ é um processo estocástico $B(t), t \in (-\infty, +\infty)$, caracterizado pelas seguintes propriedades [Nor95]:

1. $B(t)$ tem incrementos estacionários;
2. $B(0) = 0$, e $EB(t) = 0$ para todo t;
3. $EB(t)^2 = |t|^{2H}$ para todo t;
4. $B(t)$ tem caminhos contínuos; e
5. $B(t)$ é Gaussiano, *i.e.*, todas suas distribuições conjuntas são Gaussianas.

Esse processo foi descoberto por Kolmogorov [Kol40], mas pouca atenção lhe foi dispensada até o artigo pioneiro de Mandelbrot e Van Ness [MN68].

2.3 Métodos para Estimativa do Parâmetro H

O grau de auto-semelhança (parâmetro H) de uma série temporal pode ser estimado através das seguintes ferramentas gráficas: (1) análise do domínio de tempo baseado em estatística R/S, (2) análise de variância de processos agregados $X^{(m)}$, e (3) análise baseada em periodograma no domínio da freqüência. A seguir são descritos os métodos para estimativa do grau de auto-semelhança de um processo.

1. **estatística R/S** — Descrita em [MW69] consiste em plotar $log(R(n)/S(n))$ versus $log(n)$, onde $R(n)/S(n) = 1/S(n)[max(0, W_1, W_2, \ldots, W_n) - min(0, W_1, W_2, \ldots, W_n)]$, com $W_k = (X_1 + X_2 + \cdots + X_k) - k\overline{X}(n)(k \geq 1)$. Quando H é bem definido, a plotagem típica de extensão de re-escala ajustada inicia com uma zona transiente de natureza de dependência de curta duração na amostra, mas eventualmente decresce e oscila em torno de uma reta com certo declive. Análise gráfica R/S é usada para determinar se tais comportamentos assintóticos parecem sustentados pelos dados. No caso afirmativo, um valor estimado de H, $\hat{H}$, é dado pelo declive assintótico, podendo variar entre 1/2 e 1.

2. **análise de variância de processos agregados X^m** — Observa-se que para processos auto-semelhantes de segunda-ordem, a variância dos processos agregados $X^{(m)}, m \geq 1$, decresce linearmente (para grandes m) em uma escala log-log. A

análise de variância de processos agregados é obtida pela plotagem $log(var(X^{(m)}))$ contra $log(m)$ ("tempo") e pelo ajuste de uma linha quadrática mínima, ignorando-se os valores menores de m. Valores estimados do declive assintótico, $-\hat{\beta}$, entre -1 e 0 sugerem auto-semelhança, e uma estimativa de grau de auto-semelhança é dada por $\hat{H} = 1 - \hat{\beta}/2$.

3. **análise baseada em periodograma** — Consiste em plotar $log(g(w))$ versus w, onde $g(w) = \frac{1}{2\pi n}|\sum_{j=1}^{n} X_j e^{ijw}|^2$, $0 \leq w \leq \pi$ de $X = (X_1, X_2, \ldots, X_n)$, e aplicar um ajuste linear de mínimo-quadrado em uma seção do gráfico correspondente a w pequeno. O declive desse ajuste é igualado a $-(1-\hat{\beta})$, de onde $\hat{H}$ é computado com $1 - \hat{\beta}/2$.

A estatística R/S tem propriedades robustas, particularmente com respeito às distribuições com caudas longas. Contudo, ela tem a desvantagem de poder conduzir a estimativas imprecisas [Ber92]. A análise de variância de processos agregados é também um ponto de partida útil na análise de dados.

Análises de dados mais refinadas baseadas em métodos MLE (*Maximum Likehood Estimate*) são usadas para estimar H quando X_t é um processo Gaussiano ou aproximadamente Gaussiano. Esses métodos e suas aproximações são baseados na densidade espectral de X_t e propriedades dos métodos MLE são discutidas por vários autores [Ber92, Dah89, Yaj85, FT86]. Um método específico que tem sido usado amplamente é aproximação de *Whittle* MLE. Em [MTT97] discute-se as qualidades dos estimadores *Whittle* "padrão", *Whittle* "agregado" e *Whittle* "local".

O estimador *Whittle* padrão é um método paramétrico que assume como conhecida à densidade da série espectral, com exceção de poucos parâmetros que são estimados. Essa suposição de parâmetros permite estimativas muito precisas quando a série sendo examinada ajusta-se exatamente ao modelo assumido. Se, por outro lado, a série não é especificada de forma correta no modelo o estimador paramétrico pode gerar valores incorretos. Uma técnica adicional para obtenção de um valor estimado de H é fornecida pelo estimador *Whittle* agregado. Esse estimador pode ser usado se a série temporal é muito longa. A idéia é agregar os dados, o que cria uma nova série mais curta,

$$X_i^{(m)} := \frac{1}{m} \sum_{j=m(i-1)+1}^{mi} X_j.$$

A nova série aumenta o desvio padrão do estimador. Por outro lado, $m^{1-H} X_i^{(m)}$ converge fracamente para um ruído Gaussiano fractal. Se o nível de agregação m é bastante alto e a dependência de longa duração está presente, então a nova série estará perto de um ruído Gaussiano fractal. O estimador *Whittle* padrão é então usado com um modelo de rGf na série agregada.

Alternativamente, pode se utilizar um estimador semiparamétrico, *Whittle* local. Para estimar a dependência de longa duração, o estimador *Whittle* local geralmente assume que a autocorrelação para a série comporta-se como h^{2H-2}, onde h é o *lag*, e H é o parâmetro de dependência de longa duração. De forma equivalente ele pode assumir que a densidade espectral é proporcional a $|v|^{1-2H}$ para freqüências pequenas de v. Esse estimador foi recentemente desenvolvido por [Rob94]. Ele é obtido através de um método não gráfico pela minimização de uma função similar aquela do *Whittle* padrão.

Capítulo 3

Gerenciamento de Tráfego em Redes ATM

Em redes ATM recursos como banda passante e *buffers* são compartilhados entre os usuários. Recursos são alocados para um usuário somente quando ele tem algo a transmitir, permitindo a rede utilizar multiplexação estatística para maximizar recursos. A suposição de que multiplexação estatística pode ser usada para maximizar banda passante é que todos os usuários não transmitem a taxa de pico simultaneamente. Entretanto, demandas por tráfego são aleatórias e toda vez que a taxa de entrada é maior que a capacidade do canal, congestionamento acontece. Em uma situação de congestionamento, o tamanho da fila pode se tornar muito grande em um curto período de tempo, ocasionando transbordo (*buffer overflow*) e perda de células. Gerenciamento de tráfego é responsável por garantir a Qualidade de Serviço (QoS - *Quality of Service*) desejada pelos usuários. O problema é especialmente difícil durante períodos de elevada carga, principalmente, se a demanda de tráfego não pode ser prevista. As funções de gerenciamento de tráfego podem ser divididas em dois grupos [CFZ98]: (1) aquelas que atuam durante a fase de estabelecimento da conexão, e (2) aquelas que atuam durante a conexão. Como exemplos de funções do primeiro grupo cita-se: Controle de Admissão de Conexões (CAC) e Roteamento. Como exemplos de funções do segundo grupo cita-se: policiamento de tráfego e modelagem de tráfego (*shaping*).

Este capítulo está dividido da seguinte maneira: na seção 3.1, descreve-se as categorias de serviço em redes ATM. Na seção 3.2, apresenta-se o mecanismo de controle de admissão. A seção 3.3 define e descreve alguns dos mecanismos de policiamento de tráfego. Na seção 3.4, mostra-se os mecanismos de descarte seletivo de células. Na seção 3.5, mostra-se algumas das políticas de descarte de pacotes.

3.1 Categorias de Serviço

As categorias de serviço fornecidas pela camada ATM relacionam as características de tráfego e os requisitos de QoS. Em geral, funções tal como roteamento, controle de admissão de conexões (CAC - Connection Admission Control) e alocação de recursos são estruturadas de forma diferenciada para cada categoria de serviço.

A arquitetura serviços especificada pelo ATM Forum consiste das seguintes categorias de serviço:

- **CBR** (*Constant Bit Rate*) - é usado para conexões que necessitam de uma quantidade fixa de largura de banda durante todo o tempo de vida da conexão. Uma conexão CBR é caracterizada pela sua PCR (*Peak Cell Rate*). A categoria CBR suporta aplicativos que operam em tempo real, ou seja, que requerem atraso e variação de atraso rigidamente limitados, como por exemplo aplicativos de voz, vídeo e de emulação de circuitos.

- **VBR** (*Variable Bit Rate*) - está dividido em duas sub-categorias baseado nos requisitos de *delay* das aplicações: *real time* VBR (rt-VBR) e *non-real time* VBR (nrt-VBR). A categoria de serviço rt-VBR suporta aplicativos que operam em tempo real e que possuem tráfego em rajadas. Uma conexão rt-VBR é caracterizada em termos dos parâmetros PCR, SCR (*Sustainable Cell Rate*) e MBS (*Maximum Burst Size*). Aplicações típicas incluem tráfego de voz e vídeo. A categoria de serviço nrt-VBR suporta aplicativos que possuam características de tráfego em rajadas e que não operam em tempo real, ou seja, que não requerem atraso e variação de atraso rigidamente limitada. É caracterizada em termos dos parâmetros PCR, SCR e MBS. Nenhum limite de atraso é associado a esta categoria.

- **UBR** (*Unspecified Bit Rate*) - algumas vezes referido como serviço "melhor esforço" é direcionado para aplicações que não operam em tempo real e que emitem tráfego em rajadas. Estas incluem transferências de arquivos e e-mail. UBR não oferece garantias de qualidade de serviço e não requer qualquer conhecimento anterior sobre as características do tráfego;

- **ABR** (*Available Bit Rate*) - fornece uma taxa de célula mínima (MCR- *Minimum Cell Rate*) somada a qualquer largura de banda disponível. Uma conexão ABR é caracterizada por sua PCR e MCR, e são usadas por aplicações que podem adaptar seu tráfego de acordo com as informações de congestionamento fornecidas pela rede. Aplicações são similares aquelas para UBR;

- **GFR** (*Garanteed Frame Rate*) - recentemente definida pelo ATM Forum (março de 1999), suporta aplicações que operam sem retrições de tempo. É projetada

para aplicações que podem requerer garantia de taxa mínima e podem se beneficiar acessando largura de banda adicional disponível na rede.

3.2 Controle de Admissão de Conexões

Controle de Admissão de Conexões (CAC -*Control Admission Connection*) é definido como o conjunto de ações tomadas durante a fase de estabelecimento da chamada no sentido de determinar se uma conexão requisitada deve ser aceita ou deve ser rejeitada. De acordo com o algoritmo CAC, uma requisição de conexão é aceita somente quando os recursos disponíveis ao longo do caminho da conexão são suficientes para atender os requisitos de QoS desta conexão e de todas as outras conexões existentes.

Caso uma requisição de conexão seja aceita pelo CAC, é gerado um *contrato de tráfego*, um acordo que especifica as características de uma conexão entre o usuário e a rede ATM. Essas características são definidas por dois elementos básicos: o Descritor de Tráfego e os parâmetros de QoS. Enquanto o usuário enviar tráfego em conformidade com o Descritor de Tráfego da Conexão, a rede fornecerá o QoS negociado. O Descritor de Tráfego é um conjunto de parâmetros que descrevem a utilização de largura de banda esperada que a conexão necessita. Estes parâmetros são:

- **PCR** (*Peak Cell Rate*) - representa a taxa de emissão de pico da fonte. PCR não pode ser maior que a velocidade do *link* da fonte. O inverso da PCR é igual a taxa de chegada mínima teórica de células no comutador ATM;
- **SCR** (*Sustainable Cell Rate*) e *MBS* (*Maximum Burst Size*) - SCR representa o tempo médio teórico entre chegadas de células com relação à velocidade do *link*. MBS especifica o número máximo de células que podem ser transmitidas a taxa de pico (PCR) e ainda estar de acordo com a SCR negociada;
- **MCR** (*Minimun Cell Rate*) - Representa a largura de banda mínima requerida. A largura de banda disponível pela rede pode variar, mas não pode tornar-se menor que MCR. MCR pode ser igual a zero.

Os parâmetros de QoS são usados para medir a qualidade de serviço da conexão e quantificar o desempenho fim-a-fim da rede na camada ATM. Estes parâmetros são:

- **CLR** (*Cell Loss Rate*) - é definida como a razão entre o número de células perdidas e o número total de células transmitidas. CLR refere-se somente a perda de células de alta prioridade;
- **CTD** (*Cell Transfer Delay*) - é o tempo decorrido entre a transmissão de uma célula e sua recepção pelo destinatário final;

- **CDV** (*Cell Delay Variation Time*) - algumas vezes chamado de *jitter*, mede a flutuação dos atrasos de transferência de células.

3.3 Mecanismos de Policiamento de Tráfego

Um mecanismo de policiamento é responsável por avaliar se as fontes geradoras de tráfego estão respeitando os parâmetros descritos no contrato de tráfego negociado durante a fase de estabelecimento da conexão com o intuito de proteger e manter a QoS de conexões já existentes. A violação do contrato de tráfego pode ser causada por vários fatores, desde estimativas imprecisas de um determinado parâmetro de tráfego ou mesmo pelo comportamento mal intencionado de um usuário. Fontes que violam o contrato de tráfego estão sujeitas a penalização pelo mecanismo de policiamento. A penalização é feita através do descarte imediato das células mal comportadas ou através da identificação dessas células mal comportadas para descarte futuro, caso encontrem uma situação de congestionamento. A seguir são apresentados alguns dos algoritmos de policiamento descritos na literatura.

O mecanismo de policiamento **Balde Furado** (**BF** - *Leaky Bucket*) [Tur86] é o mais popular. De fato, o **GCRA** (*Gereneric Cell Rate Algorithm)*, mecanismo de policiamento adotado pelo ATM Forum, é equivalente ao mecanismo balde furado [The99]. No mecanismo de policiamento BF, uma célula necessita de uma permissão para entrar na rede. Permissões são geradas a uma taxa constante, e existe um número máximo de permissões (tamanho do balde) aceitas em um reservatório de permissões (balde). Cada permissão corresponde a uma célula. Assim, quando uma célula chega e existem permissões, a célula é aceita e uma permissão é removida do reservatório de permissões. Caso não existam permissões, a célula é descartada ou marcada para descarte futuro. O mecanismo BF também pode ser implementado com um contador, uma taxa de vazamento e um limite. O contador é decrementado (se o valor for maior que zero) à taxa de vazamento e é incrementado a cada chegada de célula até o valor máximo definido pelo limite. As células que ao chegarem encontrarem o contador no seu valor máximo são descartadas ou marcadas. Em uma outra variante, chamada de **balde furado com memória** (*buffered leaky bucket*), células são armazenadas em um *buffer* caso não existam permissões disponíveis. Células que ao chegar encontrarem o *buffer* cheio são descartadas. Da mesma forma que no BF, cada célula que deixa o *buffer* e entra na rede consome uma permissão.

Existe também uma outra classe de mecanismos de policiamento baseados em janelas [Rat91]. O mecanismo de **Janela Saltitante** (**JS** - *Jumping Window*) limita o número máximo de células que uma fonte pode transmitir dentro de um intervalo de tempo T, que representa o tamanho da janela, a um valor N. O mecanismo possui um contador que é incrementado sempre que chega uma célula da fonte monitorada, e as células só passam a

ser descartadas ou marcadas quando o valor do contador ultrapassar N. Um novo intervalo começa imediatamente após o final do precedente e o contador sempre é zerado ao final do intervalo. Portanto o intervalo de tempo durante o qual uma célula contribui para o valor do contador varia de zero ao tamanho da janela. Para o mecanismo de JS ter uma taxa equivalente a do mecanismo BF, a razão entre o limite superior do número de células durante um intervalo e a duração do intervalo deve ser igual à taxa de vazamento do BF. No mecanismo **Janela Saltitante com Gatilho** (**JSG** - *Triggered Jumping Window*) o início da janela é disparado (*triggered*) pela chegada de uma célula, de modo que janelas consecutivas não são necessariamente consecutivas no tempo como acontece no mecanismo JS. Uma desvantagem do mecanismo JS/JSG é o fato de este perder o histórico do tráfego gerado pela fonte que está sendo monitorada. No mecanismo **Janela Deslizante** (**JD** - *Moving Window*) um histórico de tráfego é mantido por um período de tempo igual ao tamanho da janela o que permite ao mecanismo "lembrar" o comportamento recente do tráfego. Entretanto, o custo para implementação do mecanismo JD é alto devido à necessidade de se registrar o número de células que chegaram e o tempo de chegada de até N células. No mecanismo de **Média Móvel Ponderada Exponencialmente** (**MMPE** - *Exponentially Weighted Moving Average*) o número de células aceitas durante uma janela é determinado por uma soma ponderada do número de células aceitas nas janelas anteriores. [Rat91] mostrou que BF e MMPE são mecanismos mais promissores do que JD e JSG com respeito ao tempo de reação, tráfego no pior caso e sensibilidade à variação de carga.

Nem o mecanismo de BF nem os mecanismos baseados em janelas são eficientes no policiamento de fontes VBR [BCT91, YSS95]. O reduzido número de parâmetros disponíveis para ajustar o processo de policiamento frente às diversas naturezas de tráfego faz com que esses mecanismos sejam poucos flexíveis. Mesmo fontes descritas somente por sua taxa média e de pico são difíceis de monitorar pelo uso desses mecanismos. Considere por exemplo fontes em rajada. Nesse caso um valor baixo de limite no mecanismo de BF pode fazer com que células bem comportadas sejam marcadas e, conseqüentemente, a probabilidade de violação seja alta. Por outro lado, um valor alto de limite faz com que o tempo necessário para detectar a violação seja alto, agravando congestionamento. Além do mais, a rajada do fluxo de saída de um sistema de fila aumenta com o limite. Outra opção é escolher uma taxa de vazamento alta, mas assim não seria possível policiar a taxa média da fonte. A solução usual é escolher uma taxa de vazamento próxima da taxa de pico da fonte, entretanto, isto pode desperdiçar banda passante uma vez que fontes em rajada normalmente não transmitem a taxa máxima todo o tempo. Em mecanismos de policiamento baseados em janela o tempo de reação à violação é fortemente dependente da duração da janela para uma certa taxa.

Para superar a limitação dos mecanismos de policiamento tradicionais, foram criados

mecanismos de policiamento multi-nível. Em cada estado de um mecanismo multi-nível existe um mecanismo simples [SFMS96]. Por exemplo, em cada estado de um mecanismo balde furado de multi-nível (*Multi Level Leaky Bucket*) existe uma taxa de vazamento, um contador e um limite [MS95]. Quando o limite do estado i é atingido, o mecanismo "pula" para o próximo estado que tem uma taxa de vazamento maior. Quando o contador voltar a zero, o mecanismo retorna ao estado anterior. Além dos parâmetros já conhecidos, em cada estado existe um tempo máximo de residência dentro de um intervalo de duração fixa (ciclo). Se o tempo de residência em um estado excede o seu valor máximo permitido, células são consideradas mal comportadas. Uma vez que um mecanismo multi-nível reage mais rápido as mudanças de estado das fontes do que mecanismos simples, mecanismos multi-nível são mais eficientes na monitoração de fontes VBR [SFMS96]. Outros trabalhos sobre mecanismos de policiamento podem ser encontrados em [MPL95, Vec94, HY96]

3.4 Descarte Seletivo de Células

Devido a natureza da taxa variável do tráfego multimídia, perdas de células podem ocorrer com freqüência em redes que utilizam multiplexação estatística, tal como redes ATM. Uma maneira de evitar estas perdas é alocar largura de banda de acordo com taxa de pico da fonte, o que torna multiplexação estatística ineficaz. Uma vez que diferentes aplicações têm diferentes requisitos de perda, atender a todos esses requisitos não é uma tarefa simples. Os requisitos de QoS relacionados a perda de uma aplicação são usualmente traduzidos em dois parâmetros de desempenho: a taxa de perda de células e o tamanho do intervalo de perdas (número de células perdidas consecutivamente). Descarte Seletivo é um mecanismo de controle de congestionamento que visa habilitar a rede a tratar os diversos requisitos de perda das aplicações. Nesse caso, células são descartadas de acordo com o seu nível de prioridade, indicada pelo bit CLP (*Cell Loss Priority*) situado no cabeçalho da célula ATM [The99]. O nível de prioridade de uma célula pode ser determinado tanto estatisticamente como dinamicamente. Quando determinado estatisticamente o nível de prioridade indica a importância de uma célula. Prioridades determinadas dinamicamente estão relacionadas a discrepâncias entre o modelo de tráfego negociado e o modelo atual de uma conexão. Nessa situação células de baixa prioridade correspondem a células que violaram o contrato de tráfego (células mal comportadas).

Descarte seletivo está relacionado somente com o gerenciamento de espaço em *buffer* e não com a ordem de transmissão. Um mecanismo de descarte seletivo é completamente especificado pela política de organização do *buffer* e pela política de descarte. Uma política de organização do *buffer* define qual parte do *buffer* pode ser ocupado por qual célula, enquanto uma política de descarte escolhe uma célula para ser descartada entre as células da classe com prioridade mais baixa. As políticas de organização do *buffer* mais

comuns são [KK80]:

- **Compartilhamento Completo** - todos os espaços de *buffer* são acessíveis para todas as classes. Células são descartadas se e somente se o *buffer* está cheio. Esta política minimiza a perda total de células;
- **Compartilhamento Completo com Descarte** - funciona da mesma maneira que a política Compartilhamento Completo, mas se uma célula encontra o *buffer* cheio e existe uma célula da classe com prioridade baixa na fila então esta célula é descartada;
- **Particionamento Completo** - cada classe tem a sua própria fila individual, ou seja, não existe compartilhamento de *buffer* entre as classes. Cada fila pode ser otimizada para manipular o fluxo de tráfego esperado. Apesar de ser uma política atrativa por não ter *overhead* de controle, perdas ocorrem mesmo se existe espaço disponível no *buffer*;
- **Compartilhamento Parcial** - cada classe tem uma posição limite até a qual as células podem ser aceitas no *buffer*. Se uma célula chega e existem mais células na fila do que a posição limite da sua classe, então a célula é perdida;
- **Compartilhamento Completo com Tamanho de Fila Máximo** - existe um limite máximo para o número de células enfileiradas de cada classe;
- **Compartilhamento Completo com Tamanho de Fila Mínimo** - cada classe tem um espaço mínimo inviolável de *buffer*. O resto do espaço em *buffer* é compartilhado da mesma forma que a política Compartilhamento Completo;
- **Compartilhamento com Garantia de Tamanho Mínimo de Fila** - difere do compartilhamento completo por garantir um espaço mínimo de *buffer* para cada classe na situação de transbordo. Nessa situação, células das classes que excedem esse espaço mínimo são descartadas de acordo com o seu nível de prioridade. Essa política tenta maximizar a ocupação do *buffer* e ao mesmo tempo garante uma certa taxa de perda de células para todas as classes [LZ92].

A Figura 3.1 ilustra os cenários de perda para diferentes políticas de organização de *buffer*.

Políticas de organização do *buffer* que perdem células somente quando ocorre transbordo são chamadas de políticas de conservação da taxa de perda. São exemplos de políticas de conservação da taxa de perda: Compartilhamento Completo (com e sem descarte) e Compartilhamento Completo com Garantia de Tamanho Mínimo de Fila. Políticas que

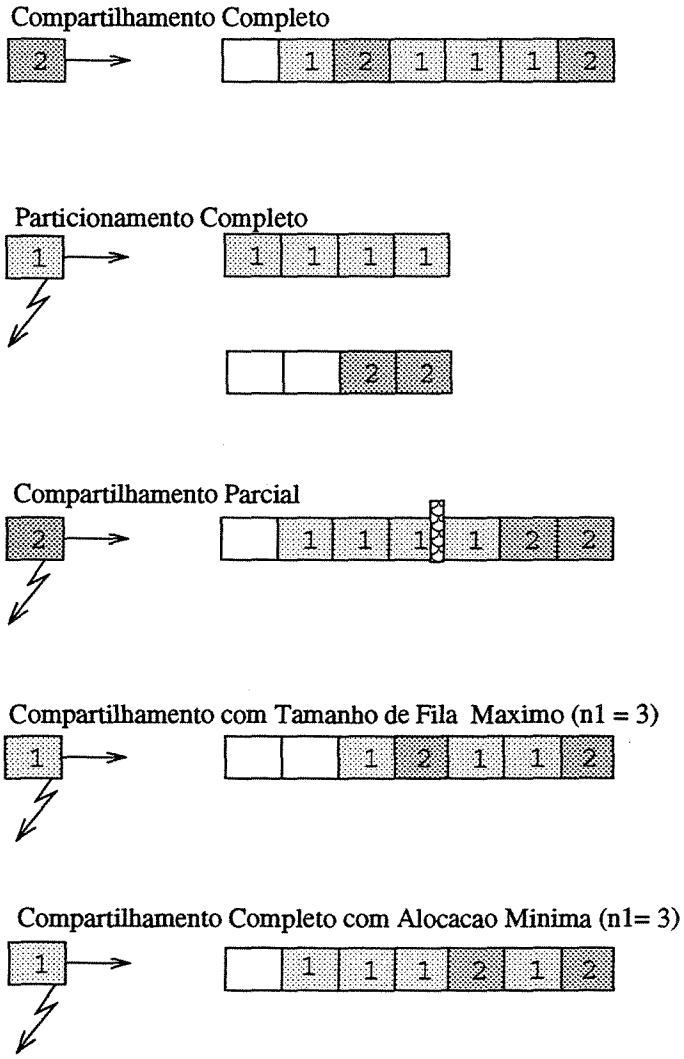


Figura 3.1: Exemplos de políticas de organização de *buffer*.

conservam a taxa de perda são de interesse especial pois minimizam a perda total de células e portanto maximizam a vazão. Outros trabalhos sobre organização de *buffer* podem ser encontrados em [THP94, CGK95, CH96].

Uma política de descarte de célula seleciona uma célula para ser descartada entre as células da classe com prioridade baixa. As políticas mais comuns são LIFD (*Last-In-Fisrt-Drop*), FIFD (*Fisrt-In-Fisrt-Drop*), RAND (*Random*) e MFIFD (*Modified First-In-First-Drop*). Descartar células da classe com prioridade baixa em diferentes posições define diferentes distribuições de fila e, conseqüentemente, pode ter impacto diferenciado no parâmetro de QoS. Como exemplo, suponha que existem duas células da classe com prioridade baixa na fila, uma no final e outra no início (Figura 3.2). Se uma célula da classe com prioridade alta encontra o *buffer* cheio, conforme a política LIFD, a célula da classe com prioridade baixa no final da fila é descartada. De acordo com a política FIFD a célula do início da fila é descartada. Após várias chegadas de células da classe com prioridade alta, o *buffer* estará cheio novamente. Na fila com LIFD, a célula no início da fila antes das chegadas de células da classe com prioridade alta (que não foi descartada) já foi transmitida e, conseqüentemente, a célula da classe com prioridade alta é perdida. Na fila com FIFD, contudo, a célula da classe com prioridade baixa no final da fila antes das chegadas de células da classe com prioridade alta não foi descartada e pode ser agora descartada para liberar espaço para a nova célula da classe com prioridade alta. A política MFIFD sempre descarta uma célula da classe com prioridade mais baixa para liberar espaço de *buffer* para uma célula que chegou, seja qual for o seu nível de prioridade. Assim, a política MFIFD concentra as células da classe com prioridade baixa no final da fila e, portanto, maximiza a taxa de perda das células da classe com prioridade baixa, ao mesmo tempo em que minimiza a taxa de perda da classe com prioridade alta [THP94].

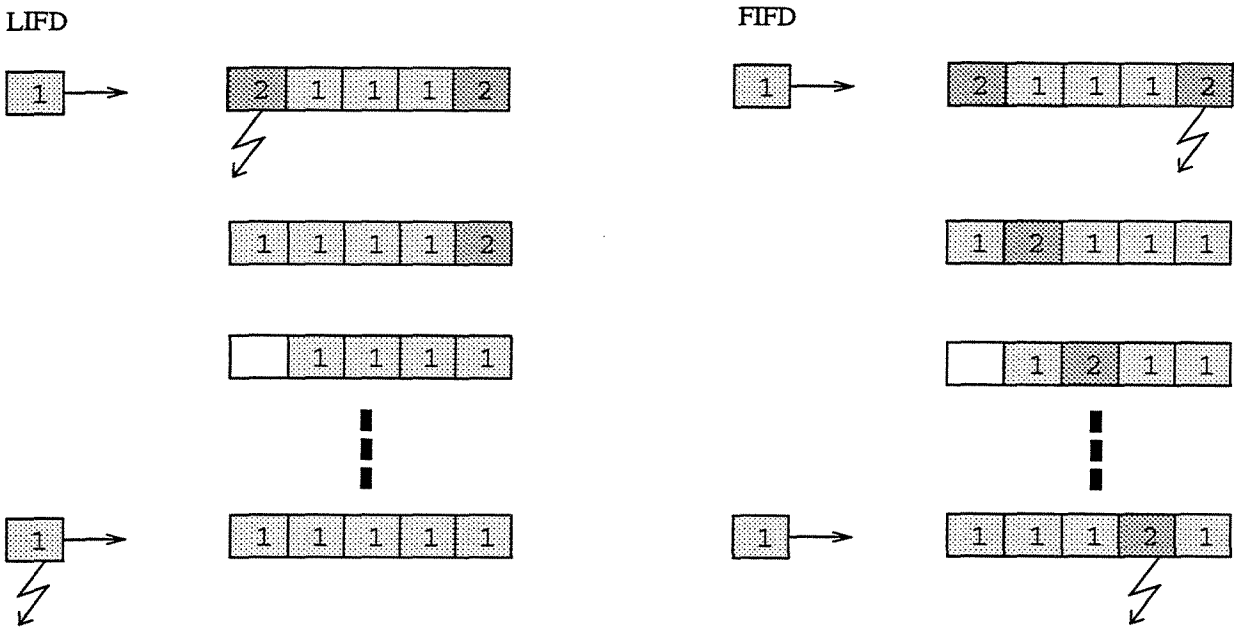


Figura 3.2: Um exemplo de perda de célula devido a diferentes distribuições de tamanho de fila produzidas pelas políticas de descarte LIFD e FIFD.

3.5 Políticas de Descarte de Pacotes para Redes ATM

Em redes ATM a informação do usuário é transmitida por meio de células de tamanho fixo de 53-*bytes* (48-*bytes* de carga útil e 5-*bytes* de cabeçalho). Uma vez que a informação a ser transmitida é usualmente maior do que a caga útil de uma célula, pacotes são segmentados em várias células ATM. A Camada de Adaptação ATM (AAL - *ATM Adaptation Layer*) executa a segmentação de pacotes de tamanho variável em células no lado do transmissor e a remontagem dessas células em pacotes no lado do receptor.

O padrão ATM especifica controle de erro para o cabeçalho da célula, mas não para a sua carga útil. Controle de erro da carga útil é manipulado de forma fim-a-fim e no caso de aplicações sensíveis a perda se uma célula do pacote é perdida, o pacote completo deve ser retransmitido. Trata-se do problema da fragmentação e remontagem de pacotes que pode afetar severamente o desempenho da aplicação e a utilização da rede. Então, transmitir células de um pacote corrompido desperdiça largura de banda e contribuí para formação de congestionamento na rede. O problema então consiste em transmitir pacotes completos através de uma rede que manipula e envia células de 53-*bytes*.

Políticas de descarte de pacotes são usadas para reduzir congestionamento e transmissão de tráfego que inevitavelmente seria retransmitido. Para aplicações de dados uti-

lizando serviço do tipo "melhor esforço", políticas de descarte de pacotes são mecanismos que implementados em cada nó intermediário e que decidem, reativamente ou proativamente, descartar pacotes para reduzir congestionamento e liberar espaço em *buffer*. Essas políticas podem ser aplicadas no caso do transporte de pacotes TCP/IP sobre ATM, assim como para o transporte de pacotes de outros protocolos. Dentre as políticas de descarte de pacotes encontradas na literatura estão:

- ***Taildrop*** - nesta política quando uma célula é perdida devido a transbordo do *buffer*, todas as células subseqüentes do mesmo pacote são descartadas, ou seja, não são admitidas no *buffer* [FR95]. Apesar do descarte das células pertencentes ao pacote corrompido, as primeiras células do pacote ainda são transmitidas;

- ***Early Packet Discard*** **(EPD)** - utiliza uma posição limite (*threshold*) no *buffer* que indica o princípio de congestionamento. Quando o nível de ocupação do *buffer* atinge o threshold, nenhum pacote novo é admitido no *buffer*, ou seja, pacotes completos são descartados [FR95]. Entretanto, células pertencentes a pacotes aceitos antes do *threshold* ser atingido são admitidas, caso exista espaço disponível no *buffer*. EPD é usualmente usada junto com *Taildrop*, melhorando ainda mais o desempenho;

- ***Selective Discard*** **(SD)** - é uma versão aperfeiçoada de EPD com o objetivo de melhorar a justiça entre as conexões. Quando o *threshold* é alcançado, SD descarta os pacotes novos de conexões que estão utilizando mais espaço em *buffer* no momento [GJK+97];

- ***Early Selective Packet Discard*** **(ESPD)** - é um aperfeiçoamento de EPD que trata o problema da sincronização global do TCP. Para alcançar esse objetivo ESPD seleciona fontes específicas para descartar seus pacotes, liberando espaço em *buffer* para as sessões restantes. ESPD utiliza dois *thresholds*. Depois de alcançar o *threshold* maior, toda conexão que envia um pacote novo é marcada e todos pacotes seguintes são descartados até a marca ser removida. A marca é removida quando o nível de ocupação do *buffer* alcançar o *threshold* menor. Assim, numa situação de congestionamento, ESPD restringe o descarte de pacotes sobre um certo número de conexões [CP98];

- ***Age Priority Packet Discarding*** **(APPD)** - proposta para diminuir o desperdício de largura de banda e alcançar justiça entre pacotes de tamanhos diferentes. Nessa política, toda vez que não existe espaço em *buffer* suficiente para acomodar células de diferentes pacotes competindo por recursos, APPD prioriza os pacotes que possuem mais células já admitidas no *buffer* (pacote com maior idade) [CT98].

Priorizar pacotes com maior idade diminui o desperdício de largura de banda e reduz o número de células corrompidas;

- ***Longest-Packet-In*** **(LPI)** - maximiza o número de células dos pacotes transmitidos com sucesso e, dado que células têm tamanho fixo, isto equivale a maximizar a largura de banda para transmitir pacotes. LPI sempre admite um pacote que chega no *buffer* caso exista espaço disponível. Caso contrário, ela descarta um ou mais pacotes enfileirados de modo que o tamanho total desses pacotes descartados seja menor que o pacote que começa a chegar [FF98].

Outras políticas de descarte de pacotes podem ser encontradas em [Tur96, LNO96, LB99].

Capítulo 4

Policiamento para Tráfego IP sobre ATM

Originalmente o mecanismo de policiamento balde furado foi concebido para redes ATM. Dessa forma, o mecanismo atua diretamente sobre as células que saem da origem, limitando o tráfego gerado sem considerar o pacote a qual essas células pertencem. Como conseqüência, células em diferentes pacotes podem ser descartadas (ou marcadas para descarte futuro). Uma vez que a camada ATM não implementa retransmissão de células, se uma das células que compõem um pacote é perdida durante a transmissão, o pacote completo terá que ser retransmitido quando este erro for detectado pelas camadas superiores. Assim sendo, a perda de uma única célula implica na perda e retransmissão de todo o pacote (problema da fragmentação e remontagem de pacotes, seção 3.5).

Baseado nestas observações sugerimos um mecanismo de policiamento (BFP - *Balde Furado para Pacote*) que é uma variante do mecanismo balde furado convencional, mas que porém, policia o fluxo de células considerando os pacotes aos quais elas pertencem, usando como parâmetro a prioridade da célula. Esta prioridade pode ser *alta* ou *baixa*. Células com prioridade baixa são marcadas para serem descartadas em situações de congestionamento da rede. Compara-se esse mecanismo alternativo com o o mecanismo balde furado convencional, investigando-se o compromisso entre o *goodput* de célula (proporção de células boas transmitidas pelo número total de células que chegaram na fila) e o *goodput* de pacote (proporção de pacotes bons transmitidos pelo número total de pacotes que chegaram na fila).

Este capítulo está organizado da seguinte maneira: na seção 4.1, introduz-se o mecanismo balde furado para pacote. Na seção 4.2, valida-se o mecanismo balde furado para pacote com respeito ao policiamento de tráfego. Na seção 4.3, descreve-se o modelo de simulação adotado para estudar o impacto gerado pelo descarte seletivo de células no tráfego policiado pelos mecanismos balde furado para pacote e balde furado. Na seção

4.4, os principais resultados são discutidos.

4.1 Princípios do Mecanismo Balde Furado para Pacote

Como dito anteriormente, a motivação para criação de um mecanismo de policiamento baseado em pacotes é o fato de que a perda de uma célula implica na perda do pacote inteiro o qual a célula pertence, uma vez que o destino não será capaz de remontá-lo. O princípio do mecanismo balde furado para pacote é fazer com que o todas as células que compõem um pacote gerado pela fonte transmissora sejam marcadas com a mesma prioridade (alta ou baixa). Para marcar todas as células de um pacote com o mesmo nível de prioridade, em algumas situações, células consideradas como excessivas ao contrato de tráfego são marcadas com prioridade alta e células geradas em períodos de conformidade com o contrato são marcadas com prioridade baixa. Dessa forma, é possível marcar todas as células de um pacote com a mesma prioridade, e ao mesmo tempo penalizar a fonte transmissora com a mesma intensidade que o mecanismo balde furado convencional. De fato, as definições de *User Parameter Control* da ITU-T (I.371 [ITU95]) e do ATM Forum (UNI 3.1 [The94]) explicitamente permitem tal comportamento. Elas requerem somente que a quantidade de células julgadas como mal comportadas estejam dentro de uma certa tolerância da quantidade de células que seriam julgadas como mal comportadas pelo "mecanismo de policiamento perfeito".

No mecanismo balde furado para pacote (Algoritmo 1), existem dois baldes chamados de *balde positivo* e *balde negativo*. O balde positivo refere-se ao modelo convencional do mecanismo balde furado para policiamento de células. A existência de fichas no balde negativo representa a deficiência de fichas no balde positivo, ou seja, indica que foram transmitidas mais células do que fora previamente definido pelo contrato de tráfego e que estas células foram transmitidas com prioridade alta. O balde positivo (B^+) pode comportar uma quantidade máxima ($TamanhoBalde$) de fichas que são geradas à taxa constante (Tx). Não há limite para o número de fichas no balde negativo (B^-), podendo inclusive conter valores menores que zero. Todo pacote novo que chega recebe uma prioridade que pode ser:

- Alta: se o número de fichas em B^- for menor ou igual a zero no instante da chegada da primeira célula do pacote;
- Baixa: se o número de fichas em B^- for maior que zero no instante da chegada da primeira célula do pacote.

Toda célula que chega recebe a mesma prioridade definida para o pacote que ela pertence, ou seja:

- Alta: neste caso, subtrai-se uma ficha de B^+. Caso não existam mais fichas em B^+, adiciona-se uma ficha em B^- (Figura 4.1a);
- Baixa: neste caso, subtrai-se uma ficha de B^+ e subtrai-se uma ficha de B^-. Caso não existam mais fichas em B^+, ambos os baldes permanecem inalterados (Figura 4.1b).

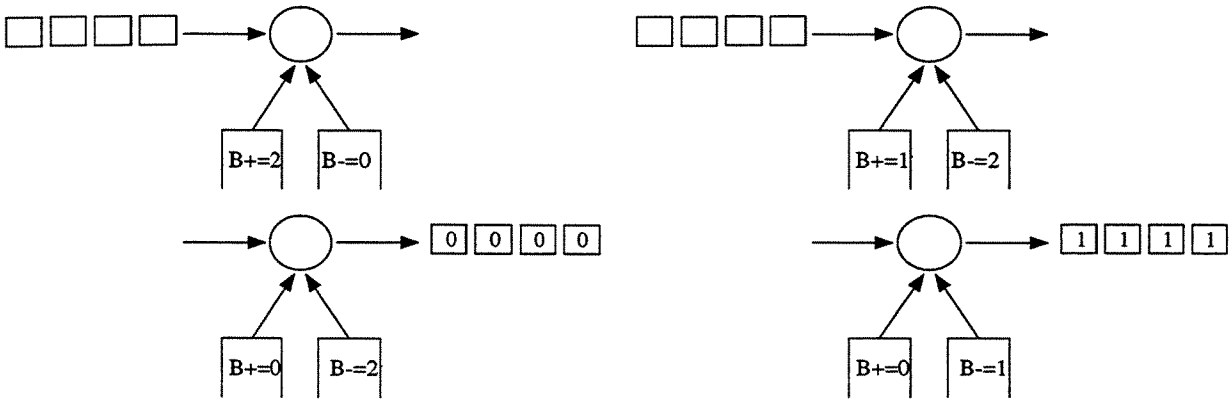


Figura 4.1: Balde Furado para Pacote - BFP.

Algoritmo 1 MECANISMO BALDE FURADO PARA PACOTE - BFP

Variáveis:

$TamanhoBalde$: Quantidade máxima de fichas permitidas no balde
B^+: Contador de fichas do balde positivo
B^-: Contador de fichas do balde negativo
Tx: Taxa de geração de fichas por unidade de tempo
$PrioPacote$: Prioridade do Pacote
$PrioCelula$: Prioridade da Célula

/* Acumula fichas até o limite máximo do tamanho do balde */
Incremente B^+ à taxa Tx;
$B^+ := min\{TamanhoBalde, B^+\}$

/* Durante a chegada de células */
se "célula é a primeira do pacote" então
    se $(B^- <= 0)$ então
        $PrioPacote :=$ "Alta"
    senão
        $PrioPacote :=$ "Baixa"
    fimse
fimse
$PrioCelula := PrioPacote$
se ($PrioCelula =$ "Alta")
    se $(B^+ > 0)$
        $B^+ := B^+ - 1$
    senão
        $B^- := B^- + 1$
    fimse
senão
    se $(B^+ > 0)$
        $B^+ := B^+ - 1$
        $B^- := B^- - 1$
    fimse
fimse

4.2 Validação do Mecanismo Balde Furado para Pacote

Para se verificar a equivalência entre os mecanismos BP e BFP com respeito a policiamento de tráfego, criou-se o modelo de simulação ilustrado na Figura 4.2. O modelo mostra uma fonte gerando tráfego constituído por células ATM. O tráfego gerado é então policiado tanto pelo mecanismo BF como pelo mecanismo BFP. Ao final da simulação a *Probabilidade de Violação* (PV - *Violation Probability*) é computada para ambos mecanismos. PV é a probabilidade de uma célula ser marcada pelo mecanismo de policiamento, no caso o BF e BFP.

$$PV = \frac{Células\ marcadas}{Células\ transmitidas} \tag{4.1}$$

Resultados das simulações demonstraram que o mecanismo BFP proposto apesar de agregar células com prioridade baixa num mesmo pacote, gera a mesma probabilidade de violação do mecanismo BF convencional. Em outras palavras, o mecanismo BFP marca com prioridade baixa a mesma quantidade de células que são marcadas quando a fonte geradora de tráfego é policiada pelo mecanismo BF convencional.

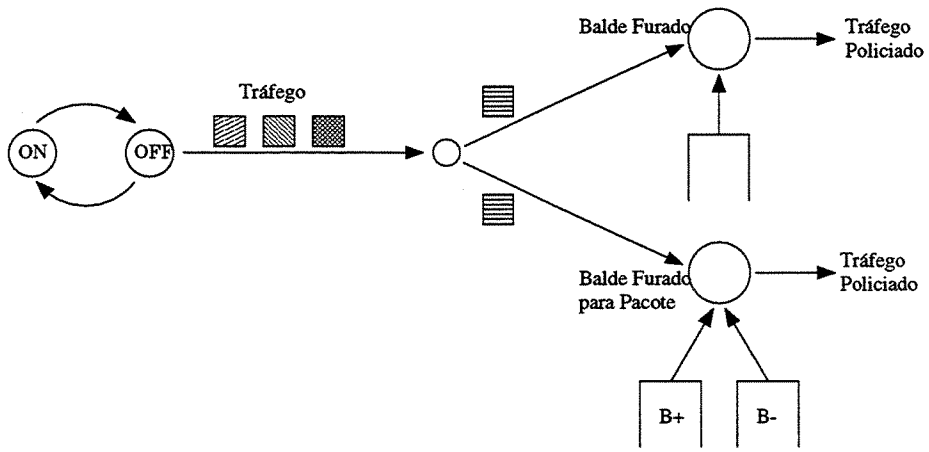


Figura 4.2: Modelo de simulação utilizado para validar o balde furado para pacote.

4.3 O Modelo de Simulação e Tráfego

Para se avaliar a eficácia de BFP, submete-se o tráfego policiado pelo mecanismo a nós de rede com descarte de célula. No modelo de simulação (Figura 4.3), todos os nós da rede, modelados como uma fila, tem *buffers* com capacidade para o mesmo número de células, utilizam a mesma política de organização de *buffer* e o mesmo mecanismo de descarte seletivo de célula. Somente o *fluxo agregado* atravessa toda a rede, ou seja, passa por todos os nós. Todos os outros fluxos de tráfego, chamados de *fluxos de interferência*, passam por um único nó cada (veja Figura 4.3). Não há fluxo de interferência no primeiro nó. Com este modelo é possível estudar o desempenho fim-a-fim apresentado pelo tráfego agregado, com diferentes mecanismos de policiamento e diferentes estratégias de descarte de célula e de pacote.

O conjunto de filas é alimentado por um processo agregado de várias fontes *on-off* homogêneas. No modelo de tráfego *on-off*, uma fonte alterna entre dois estados diferentes conhecidos como período *on* e período *off*. Durante o período *on*, a fonte gera células pertencentes a um pacote. Estas células são espaçadas de forma uniforme por um número fixo de fatias de tempo (*time slots*). Períodos *off* por sua vez correspondem a intervalos de "silêncio", ou seja, intervalos entre a geração de pacotes.

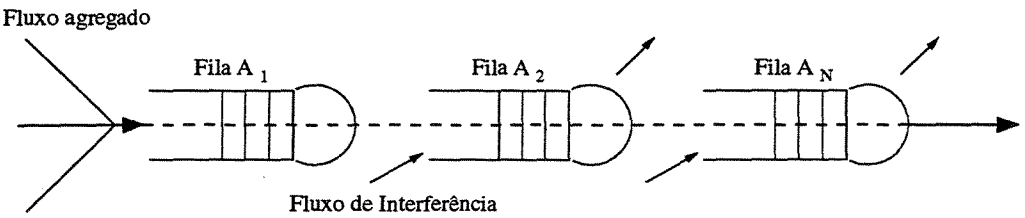


Figura 4.3: Modelo de simulação e tráfego.

Nos experimentos de simulação realizados, considerou-se dois tipos de tráfego: *sintético* e *real*. No caso do tráfego sintético, gera-se processos com dependência de longa-duração utilizando a distribuição de Pareto tanto para a duração dos períodos *on* como para duração dos períodos *off*. O resultado foi um tráfego agregado auto-semelhante com dependência de longa duração [Wil97]. A carga de um fluxo de tráfego é expressa por:

$$\rho = N \times \epsilon$$

onde N é o número de fontes e

$$\epsilon = \frac{T_{on}}{(T_{on} + T_{off})}$$

é a taxa de atividade, isto é, a probabilidade estacionária de uma fonte estar em estado *on*. T_{on} e T_{off} são a duração média dos períodos *on* e *off*, respectivamente. Varia-se a carga do sistema pela mudança de N ou de ϵ.

Tráfego real foi gerado a partir de *traces* de tráfego das redes de *Colorado State University*, *San Diego Super Computer Center*, *Michigan universities*, *Columbia University*, *Florida universities* e *Indiana University*, EUA. Todos os *traces* são constituídos pelo tráfego IP agregado observado em *backbones* ATM que ligam estas redes ao resto da *Internet*. Estes *traces* estão disponíveis no *site* do *National Laboratory for Applied Network Research* (NLANR, http://www.nlanr.net).

A tabela 4.1 resume as principais características dos *traces* utilizados.

REDE-TRACE	LINK	DURAÇÃO	FRAME	Nº PACOTES IP
COS-9615166581	OC3c	98 seg.	ATM-AAL5	836.000,00
12SDC-964130254	OC12c	94 seg.	ATM-AAL5	327.000,00
MRT-964378549	OC3c	81 seg.	ATM-AAL5	434.000,00
BWY-974951706	OC3c	81 seg.	ATM-AAL5	327.000,00
FLA-990265951	OC3c	82 seg.	ATM-AAL5	301.000,00
IND-970275951	OC12c	85 seg.	ATM-AAL5	270.000,00

COS-9615166581 = *Colorado State University*
12SDC-964130254 = *San Diego Super Computer Center*
MRT-964378549 = *Michigan universities*
BWY-974951706 = *Columbia University*
FLA-990265951 = *Florida universities*
IND-970275951 = *Indiana University*

Tabela 4.1: Sumário dos *traces* de tráfego IP agregado.

Apesar de se tratar de um tráfego IP sobre ATM com classe de serviço AAL5, os *traces* coletados só contém informações sobre o cabeçalho do pacote IP e o *time stamp* do pacote. Assim fez-se necessário o seguinte pre-processamento antes da utilização dos *traces* nos simuladores:

- Segmentação de pacotes IP em células - seja tp_{bytes} o tamanho do pacote IP em *bytes*, então o número de células necessárias para se transportar este pacote é calculado por:

$$tp_{celulas} = \lceil (tp_{bytes} + 8)/48 \rceil$$

onde 8 representa o *trailer* inserido pelo protocolo AAL5 para cada pacote IP [Hei93].

- Conversão do *time stamp* de segundos para *time slots* - seja:

  tm = taxa média do *trace* em células/segundo

  ρ = carga do sistema

  fc = fator de conversão calculado por:

$$fc = \frac{\rho}{tm}$$

  $ts_{segundo}$ = *time stamp* expresso em segundos

  $ts_{timeslot}$ = *time stamp* expresso em *time slots*

  então:

$$ts_{timeslots} = \lfloor \frac{ts_{segundos}}{fc} \rfloor$$

- Divisão do *trace* entre as fontes - A fim gerar tráfego para o número de fontes utilizado na simulação, dividi-se o *trace* da seguinte maneira: considerando uma simulação para y fontes, a leitura do *trace* é feita em blocos contendo y registros (cada registro corresponde à chegada de um pacote IP) e dentro de cada bloco lido, um registro é alocado para cada uma das y fontes geradoras de tráfego de forma seqüencial, ou seja, o primeiro registro lido é alocado para fonte 1, o segundo registro lido é alocado para fonte 2 e assim sucessivamente até o final do bloco.

4.4 Resultados Numéricos

A fim de avaliar a eficácia do mecanismo balde furado para pacote (BFP), compara-se o mesmo com o mecanismo balde furado convencional (BF). Analisa-se o compromisso entre maximizar o número de células transmitidas em pacotes com sucesso e maximizar o número de pacotes transmitidos com sucesso. Estudos anteriores adotaram diferentes objetivos de desempenho para comparar políticas de descarte. Turner [Tur96] e Romanov e Floyd [FR95] consideraram a vazão efetiva (*effective throughput*) que é a proporção de células boas em pacotes não corrompidos no fluxo de saída pelo total de células no fluxo de saída. Lapid et. al [LRS97] definiram o *goodput* como a proporção de células boas no fluxo de saída pelo número total de células que chegaram na fila. No presente trabalho, defini-se o *goodput* de célula como em Lapid et. al. O *goodput* de célula mostra quanto de tráfego não é desperdiçado. Ele difere da vazão efetiva no sentido de que a vazão efetiva não leva em consideração células descartadas. A vazão efetiva considera apenas o fluxo de saída. Além disso, apresenta-se o *goodput* de pacote que é a proporção de pacotes bons no fluxo de saída pelo número total de pacotes que chegaram na fila. O *goodput* de pacote fornece a percentagem de pacotes de usuário transmitidos com sucesso. Em resumo, o *goodput* de célula fornece a eficácia de uma política em nível de *link* de rede, enquanto o *goodput* de pacote indica a eficácia de uma política em nível da camada de transporte. Nos experimentos de simulação, as filas utilizam CCD (Compartilhamento Completo com Descarte) como a política de organização do *buffer*. Com CCD, células são descartadas somente em situações de transbordo, em outras palavras, essa política de organização do *buffer* minimiza a perda total de células e, conseqüentemente, maximiza a vazão. Com intuito de minimizar a taxa de perda dos pacotes com prioridade alta utiliza-se o mecanismo de descarte seletivo de célula MFIFD. Nos experimentos de simulação, investiga-se o impacto da carga da rede e parâmetro de Hurst no *goodput* de célula e no *goodput* de pacote utilizando tanto tráfego sintético é tráfego real em redes formadas por um único nó e por vários nós.

4.4.1 Tráfego Sintético

Ambiente composto por um único nó

Investiga-se primeiramente a eficácia dos mecanismos BFP e BF num ambiente composto por um único nó. A tabela 4.2 sumariza os parâmetros utilizados para o fluxo agregado nos experimentos de simulação. O tamanho médio de 10 células para o pacote corresponde basicamente ao número de células necessárias para se transportar um pacote IP de 512-*bytes*.

Na Figura 4.4, mostra-se o *goodput* de célula como função da carga da rede, para

FLUXO AGREGADO	NÓ	MECANISMO DE POLICIAMENTO
N = 16 H = 0.70, 0.80, 0.85, 0.90 TMP = 10	TBu = 150 POB = CCD MD = MFIFD	para ρ_{Agr}=1.0: Tx=0.062 e TBa=10 para ρ_{Agr}=0.9: Tx=0.056 e TBa=10 para ρ_{Agr}=0.8: Tx=0.050 e TBa=10 para ρ_{Agr}=0.7: Tx=0.044 e TBa=10

N-Número de fontes
H-Parâmetro de Hurst
TMP-Tamanho médio do pacote em células
TBu-Tamanho do *buffer* em células
POB-Política de organização do *buffer*
MD-Mecanismo de descarte
ρ_{Agr}-Carga do fluxo agregado
Tx-Taxa de geração de fichas por *time slot*
TBa-Tamanho do balde em células

Tabela 4.2: Sumário dos parâmetros utilizados nas simulações com tráfego sintético, fluxo agregado.

diferentes valores do parâmetro de Hurst. As legendas "BF MFIFD" e "BFP MFIFD" significam que o tráfego é policiado pelo mecanismo BF e BFP, respectivamente, e que MFIFD é o mecanismo de descarte de células utilizado pelo nó. Com o aumento do parâmetro de Hurst, períodos de transbordo tornam-se mais longos e, conseqüentemente, o *goodput* de célula diminui. O mecanismo BFP fornece *goodput* de célula superior ao mecanismo BF considerando parâmetro de Hurst até 0.8. Com o aumento do parâmetro de Hurst a diferença no *goodput* de célula entre os dois mecanismos diminui. De fato, para H=0.85 praticamente não há diferença no *goodput* fornecido pelos mecanismos e, para H=0.9 o mecanismo BF fornece *goodput* de célula superior. Nota-se que dependência de longa-duração tem impacto considerável no *goodput* de célula produzido por ambos mecanismos. Por exemplo, para carga 0.7 e parâmetro H=0.7 o *goodput* de célula produzido por BFP é 0.77, e para H=0.9 é 0.57. No caso de BF, o *goodput* de célula passa de 0.892 para 0.574.

No caso do *goodput* de pacote (Figura 4.5), o mecanismo BFP mostrou-se mais sensível ao parâmetro de Hurst e tem desempenho inferior independente da carga da rede. Por exemplo, para carga 1.0 e parâmetro H=0.7, BFP fornece *goodput* de pacote 0.91 e BF 0.945. Mantendo a carga em 1.0 e aumentado o parâmetro H para 0.9, BFP fornece *goodput* de pacote 0.769 e BF 0.93, de modo que as reduções foram de 0.141 e 0.015, para BFP e BF respectivamente.

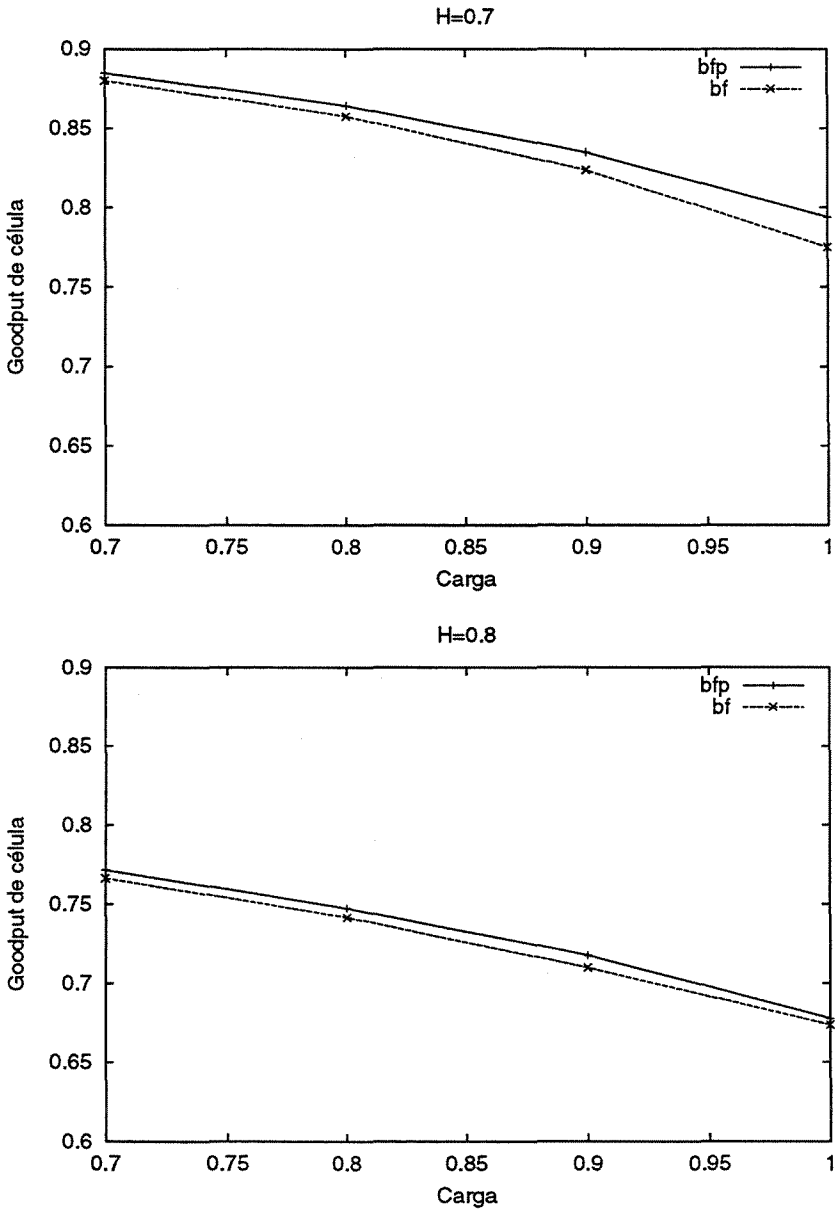

H=0.7
bfp
bf
Goodput de célula
Carga
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.7
0.75
0.8
0.85
0.9
0.95
1
H=0.8
bfp
bf
Goodput de célula
Carga

Figura 4.4a

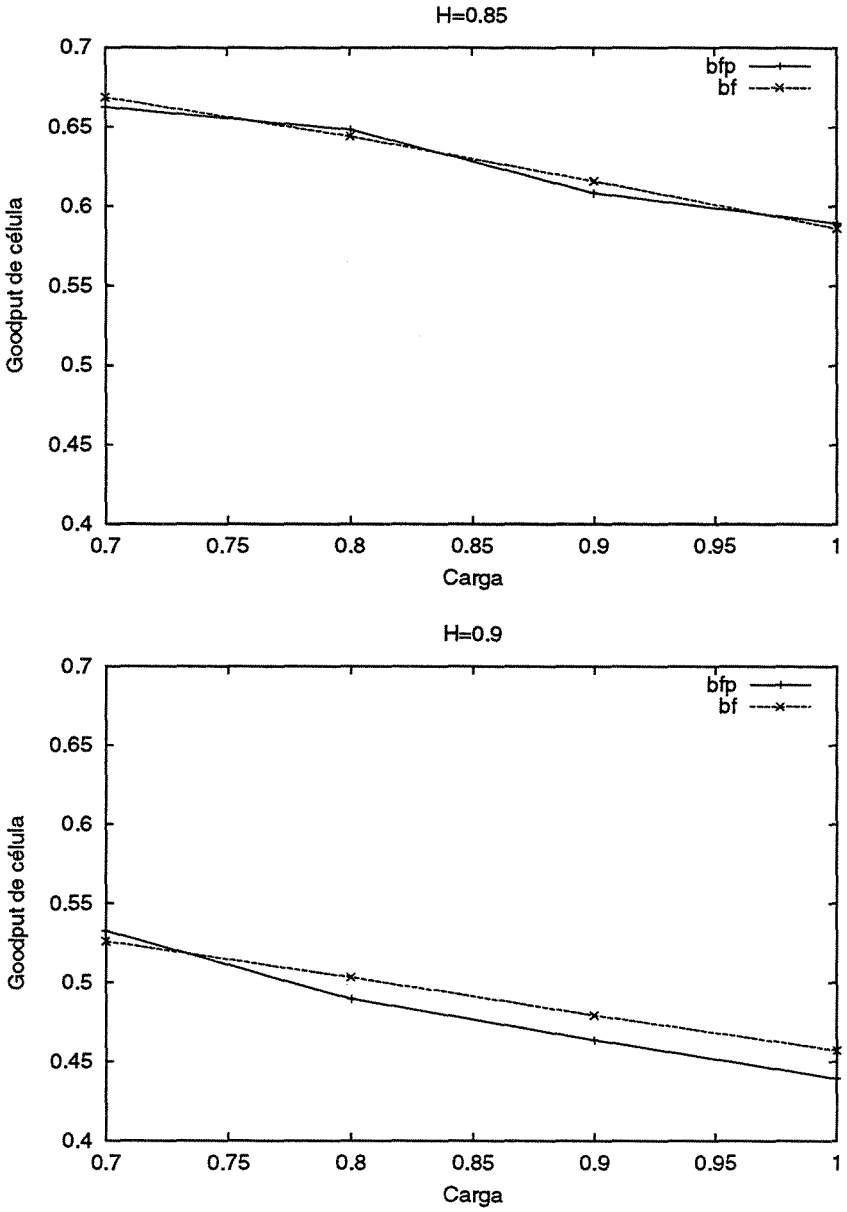


Figura 4.4b

Figura 4.4: *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego sintético

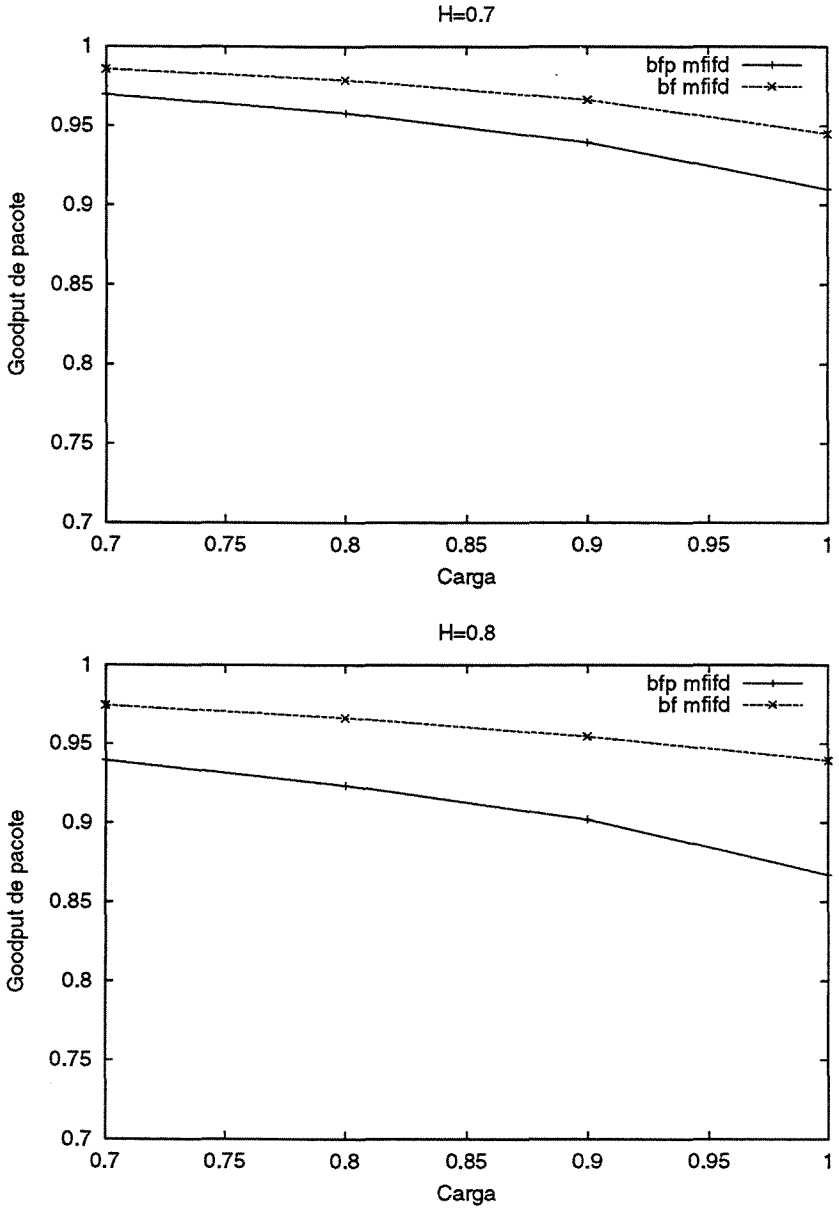

H=0.7
bfp mfifd
bf mfifd
Goodput de pacote
Carga
1
0.95
0.9
0.85
0.8
0.75
0.7
0.7
0.75
0.8
0.85
0.9
0.95
1
H=0.8
bfp mfifd
bf mfifd
Goodput de pacote
Carga

Figura 4.5a

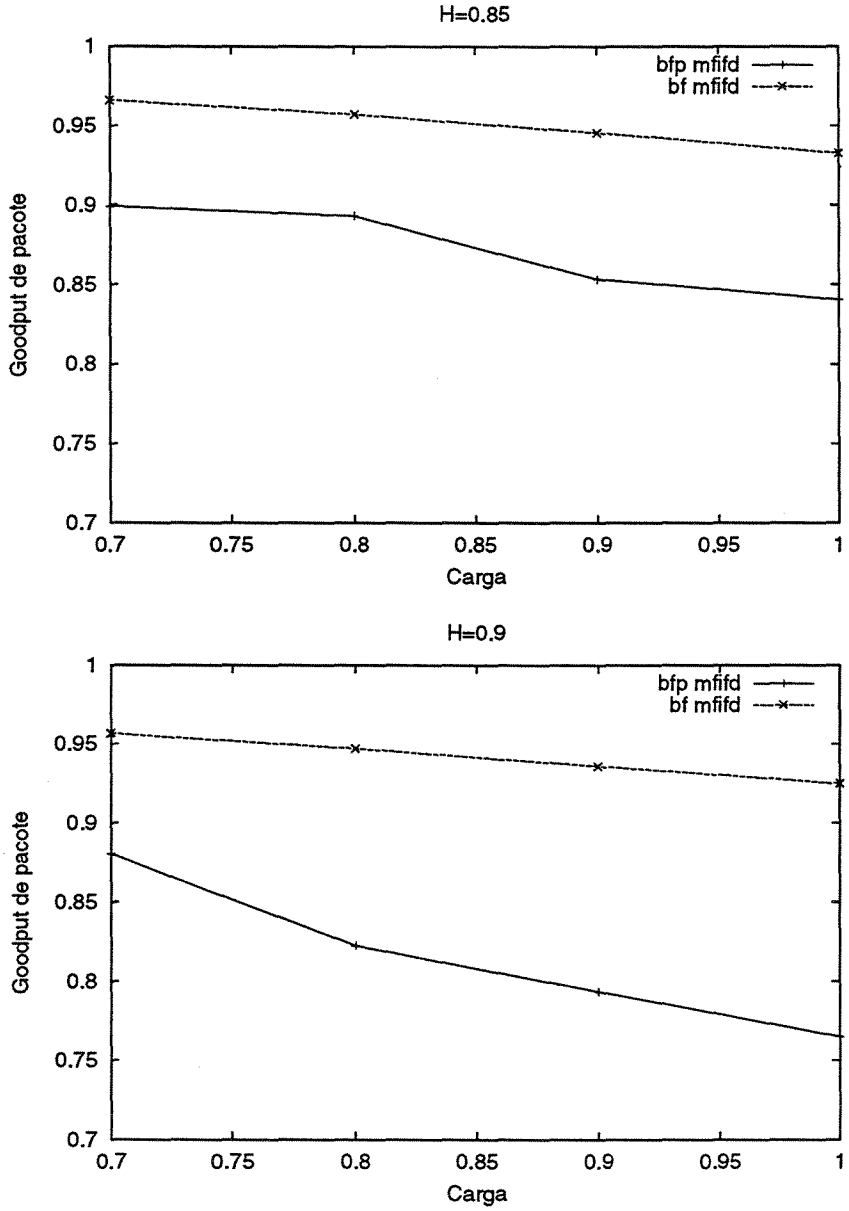


Figura 4.5b

Figura 4.5: *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego sintético.

Ambiente composto por vários nós

Nesta etapa investiga-se se o comportamento apresentado pelos mecanismos BFP e BF num ambiente de rede formado por um único nó é mantido num ambiente de rede formado por vários nós. Os parâmetros utilizados para o fluxo agregado são os mesmos

utilizados no ambiente composto por um único nó, descritos na seção 4.4.1, tabela 4.2. A tabela 4.3 sumariza os parâmetros de simulação utilizados para o fluxo de interferência. Note que os parâmetros são os mesmos em todos nós da rede, salvo o primeiro nó onde não há fluxo de interferência.

FLUXO DE INTERFERÊNCIA	NÓ	MECANISMO DE POLICIAMENTO
N = 1 para cada nó H = 0.70, 0.80, 0.85, 0.90 TMP = 10	TBu = 150 POB = CCD MD = MFIFD	ρ_{Int}=0.1, Tx=0.1 e TBa=10 independente de ρ_{Agr}

N-Número de fontes
H-Parâmetro de Hurst
TMP-Tamanho médio do pacote em células

TBu-Tamanho do *buffer* em células
POB-Política de organização do *buffer*
MD-Mecanismo de descarte

ρ_{Agr}-Carga do fluxo agregado
ρ_{Int}-Carga do fluxo de interferência
Tx-Taxa de geração de fichas por *time slot*
TBa-Tamanho do balde em células

Tabela 4.3: Sumário dos parâmetros utilizados nas simulações com tráfego sintético, fluxo de interferência.

Na Figura 4.6, mostra-se o *goodput* de célula como função da carga da rede, para diferentes valores do parâmetro de Hurst. Tanto o mecanismo BF quanto o mecanismo BFP fornecem *goodput* de célula inferiores aos que foram apresentados no ambiente de rede composto por um único nó (seção 4.4.1). Isto era esperado uma vez que agora num ambiente de rede com 5 nós aumenta-se a possibilidade de se perder células (e o pacote correspondente). Apesar do desempenho inferior em relação ao ambiente composto por único nó, o comportamento apresentado por ambos mecanismos se mantém, ou seja, considerando parâmetros e Hurst até 0.8, o mecanismo BFP fornece *goodput* de célula superior. Com o aumento do parâmetro de Hurst (H>0.8) a diferença entre o *goodput* dos mecanismos diminui e para H=0.9, BF fornece *goodput* de célula superior. Nota-se que no ambiente com vários nós a diferença entre o *goodput* de célula dos mecanismos torna-se maior. Por exemplo, para carga 1.0 e parâmetro H=0.7, a diferença entre o *goodput* de célula fornecido por BFP e BF é de 0.02, para o ambiente com um único nó, ao passo que no ambiente com 5 nós a diferença está acima de 0.05.

Tal como acontece com o *goodput* de célula, o comportamento apresentado pelos mecanismos BF e BFP também se manteve no ambiente composto por vários nós, ou seja, com o mecanismo BF fornecendo *goodput* de pacote superior independente da carga ou parâmetro de Hurst (Figura 4.7). Além disso, no ambiente com vários nós a diferença entre o *goodput* de pacote dos mecanismos torna-se maior. Por exemplo, para carga 1.0 e parâmetro H=0.8, a diferença entre o *goodput* de pacote fornecido por BF e BFP é de

0.07, para o ambiente com um único nó, ao passo que no ambiente com 5 nós a diferença é de 0.11.

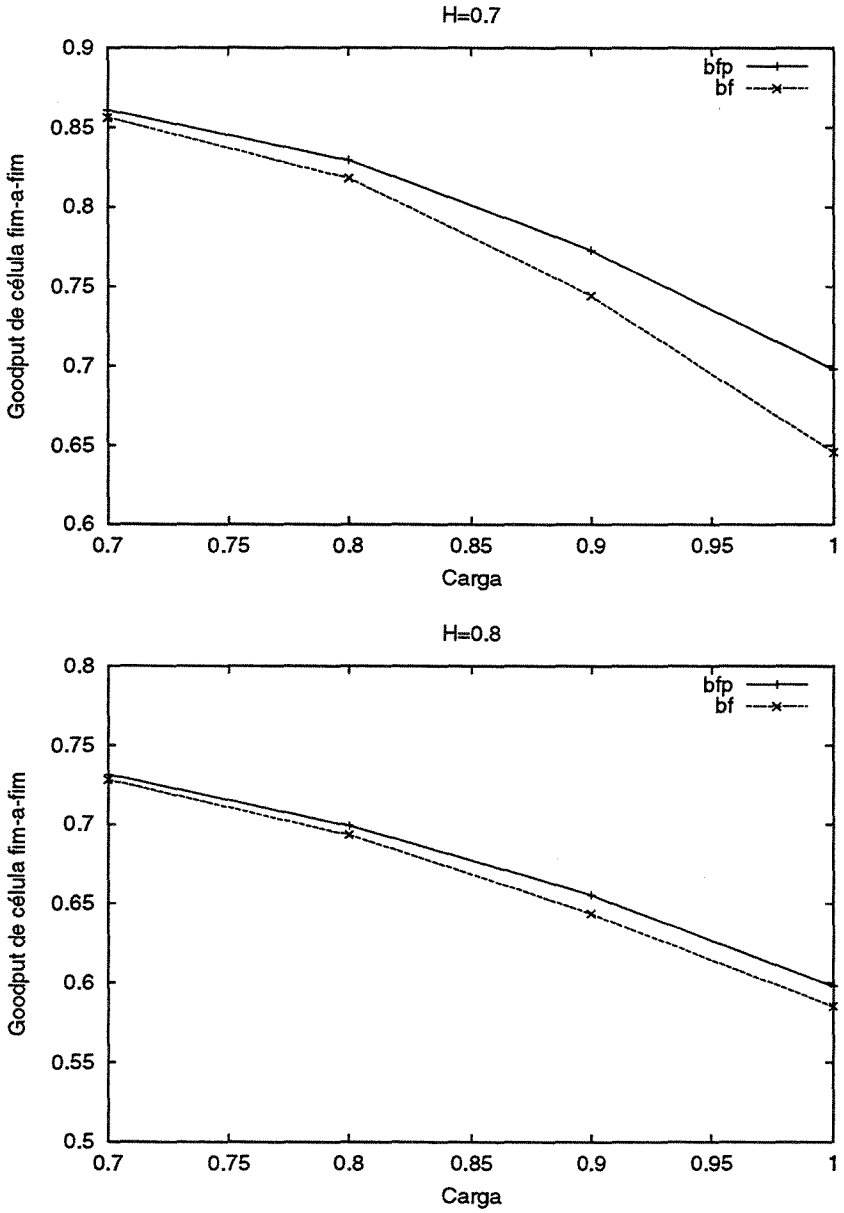


Figura 4.6a

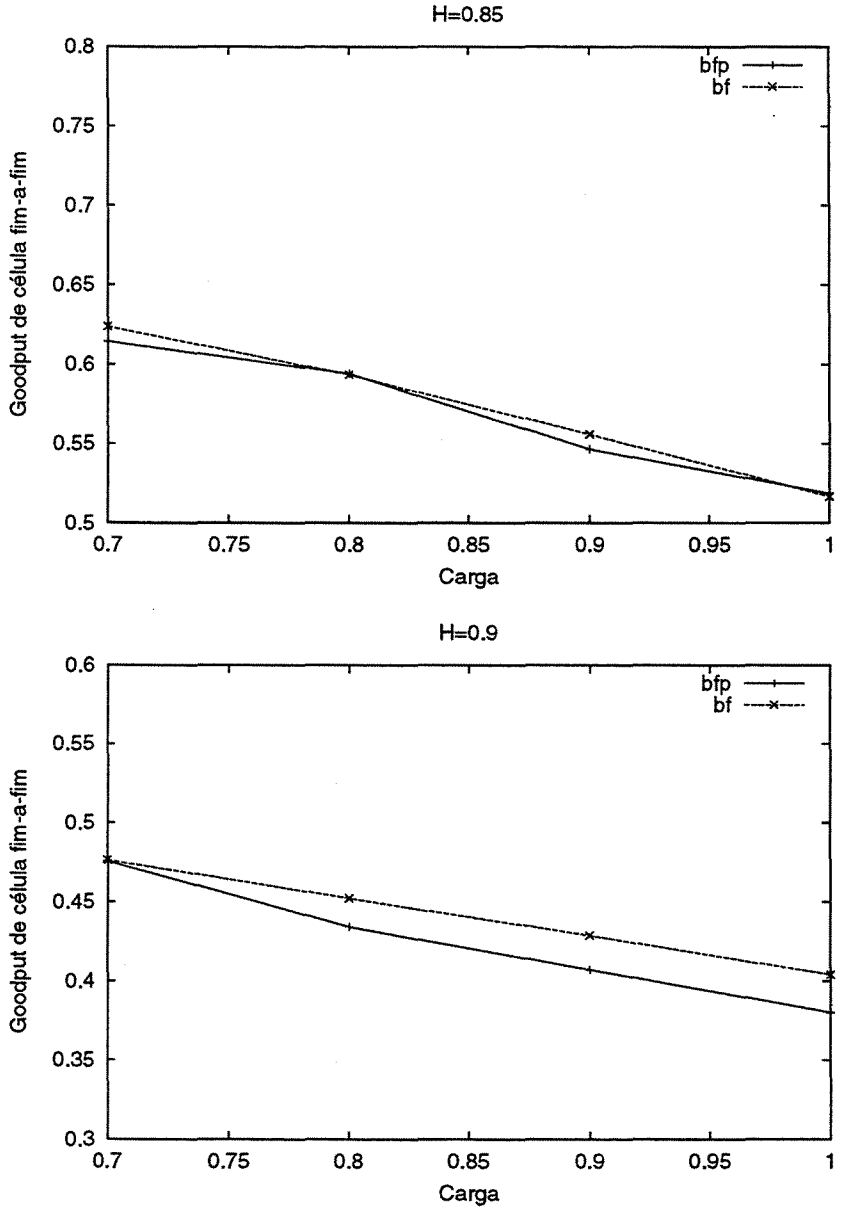


Figura 4.6b

Figura 4.6: *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético.

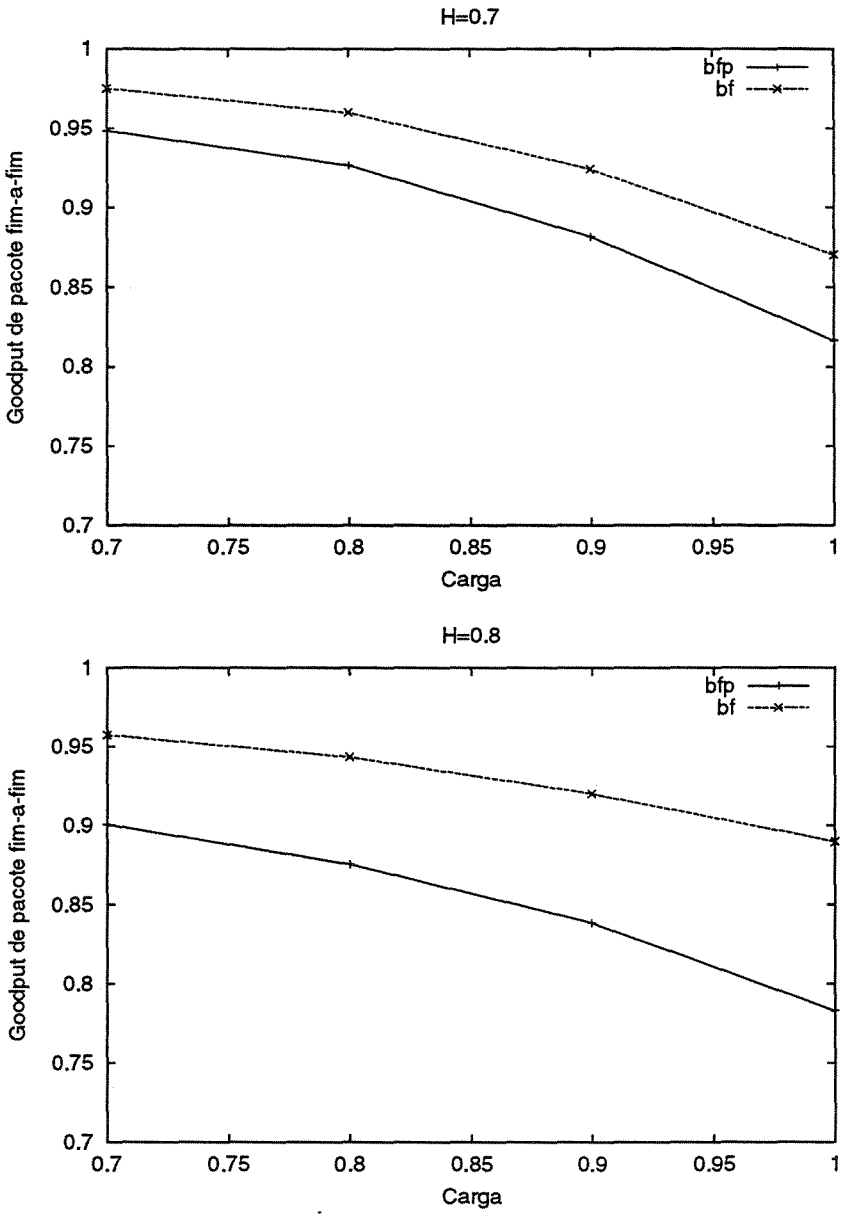

H=0.7
bfp
bf
Goodput de pacote fim-a-fim
Carga
1
0.95
0.9
0.85
0.8
0.75
0.7
0.7
0.75
0.8
0.85
0.9
0.95
1
H=0.8
bfp
bf
Goodput de pacote fim-a-fim
Carga

Figura 4.7a

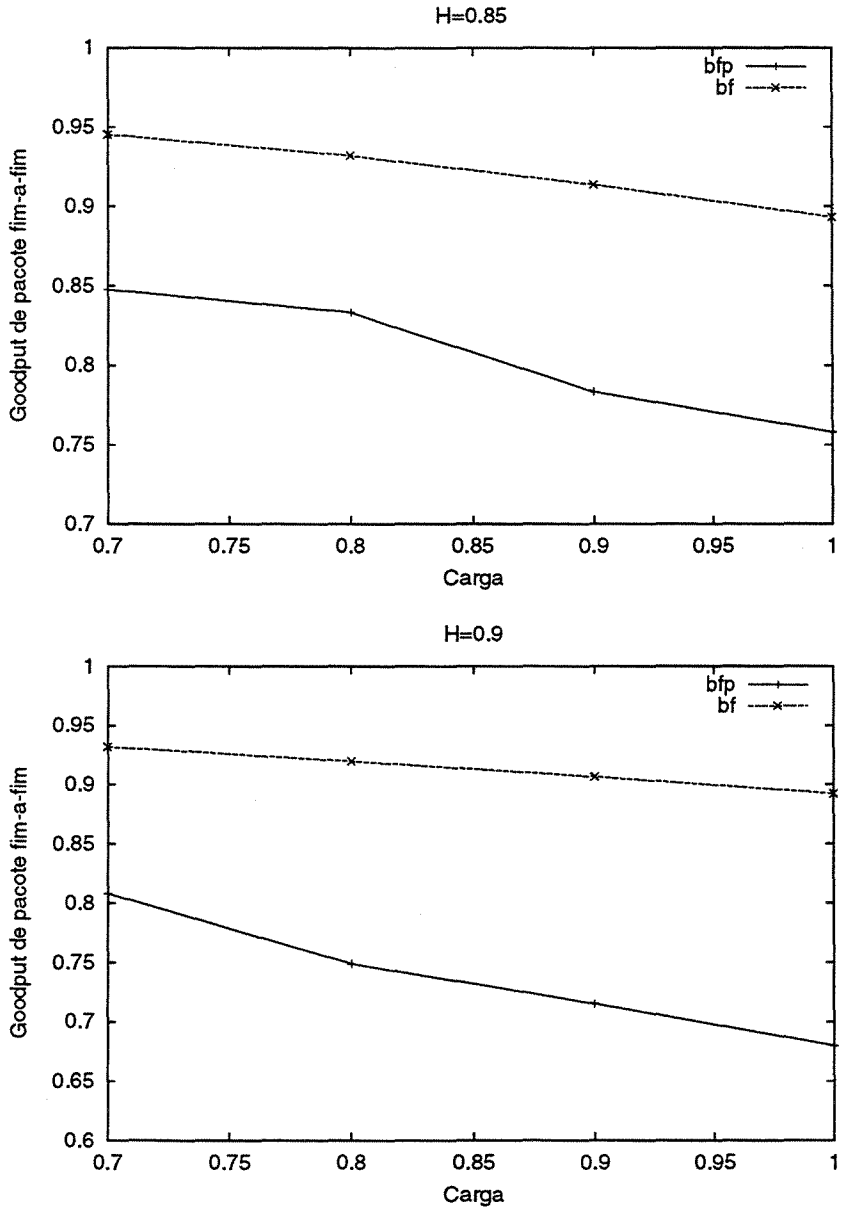


Figura 4.7b

Figura 4.7: *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético.

4.4.2 Tráfego Real

Nesta seção, utiliza-se nos experimentos de simulação tráfego real, gerado a partir dos *traces* descritos na seção 4.3, tabela 4.1. Mostra-se resultados para os *traces* 12SDC-964130254 e COS-9615166581, que representam o comportamento padrão apresentado por todos os *traces*.

Ambiente composto por um único nó

A tabela 4.4 sumariza os parâmetros utilizados para o fluxo agregado nos experimentos de simulação. Para se estimar o parâmetro de Hurst dos *traces* utilizou-se a análise gráfica *variance time*.

Trace 12SDC-964130254		
FLUXO AGREGADO	NÓ	MECANISMO DE POLICIAMENTO
N = 16 H = 0.93 TMP = 16	TBu = 250 POB = CCD MD = MFIFD	para ρ_{Agr}=1.0: Tx=0.062 e TBa=16 para ρ_{Agr}=0.9: Tx=0.056 e TBa=16 para ρ_{Agr}=0.8: Tx=0.050 e TBa=16 para ρ_{Agr}=0.7: Tx=0.044 e TBa=16
Trace COS-9615166581		
FLUXO AGREGADO	NÓ	MECANISMO DE POLICIAMENTO
N = 16 H = 0.91 TMP = 11	TBu = 150 POB = CCD MD = MFIFD	para ρ_{Agr}=1.0: Tx=0.062 e TBa=11 para ρ_{Agr}=0.9: Tx=0.056 e TBa=11 para ρ_{Agr}=0.8: Tx=0.050 e TBa=11 para ρ_{Agr}=0.7: Tx=0.044 e TBa=11

N-Número de fontes
H-Parâmetro de Hurst
TMP-Tamanho médio do pacote em células
TBu-Tamanho do *buffer* em células
POB-Política de organização do *buffer*
MD-Mecanismo de descarte
ρ_{Agr}-Carga do fluxo agregado
Tx-Taxa de geração de fichas por *time slot*
TBa-Tamanho do balde em células

Tabela 4.4: Sumário dos parâmetros utilizados nas simulações com tráfego real, fluxo agregado.

Na Figura 4.8, mostra-se o *goodput* de célula em função da carga da rede. O mecanismo BFP gerou *goodput* de célula superior ao do mecanismo BF, com uma diferença máxima de 0.175 no caso do *trace* 12SDC-964130254 e 0.122 no caso do *trace* COS-961516658. Com o aumento da carga, a redução do *goodput* de célula fornecido pelo mecanismo BF

foi mais acentuada que o foi para mecanismo BFP. Por exemplo, no caso do *trace* COS-961516658-1 o *goodput* de célula de BF que é 0.94 para carga 0.7, cai para 0.688 quando a carga passa para 1.0, portanto com redução de 0.252. No caso do mecanismo BFP a mesma variação na carga diminui o *goodput* de célula de 0.97 para 0.81, redução de 0.16.

Assim como o *goodput* de célula, o *goodput* de pacote fornecido pelo mecanismo BFP também é superior ao que é fornecido pelo mecanismo BF, como mostra a Figura 4.9. A diferença de *goodput* de pacote entre os mecanismos chega a 0.045 no caso do *trace* 12SDC-964130254 e 0.03 no caso do *trace* COS-961516658-1. Apesar do desempenho superior, a diferença entre o *goodput* de pacote fornecido por BFP e BF manteve-se muito próximo com o aumento da carga da rede. Por exemplo, para o *trace* 12SDC-964130254 a diferença entre o *goodput* de pacote dos mecanismos foi de 0.035 para carga 0.7, e 0.039 para carga 1.0.

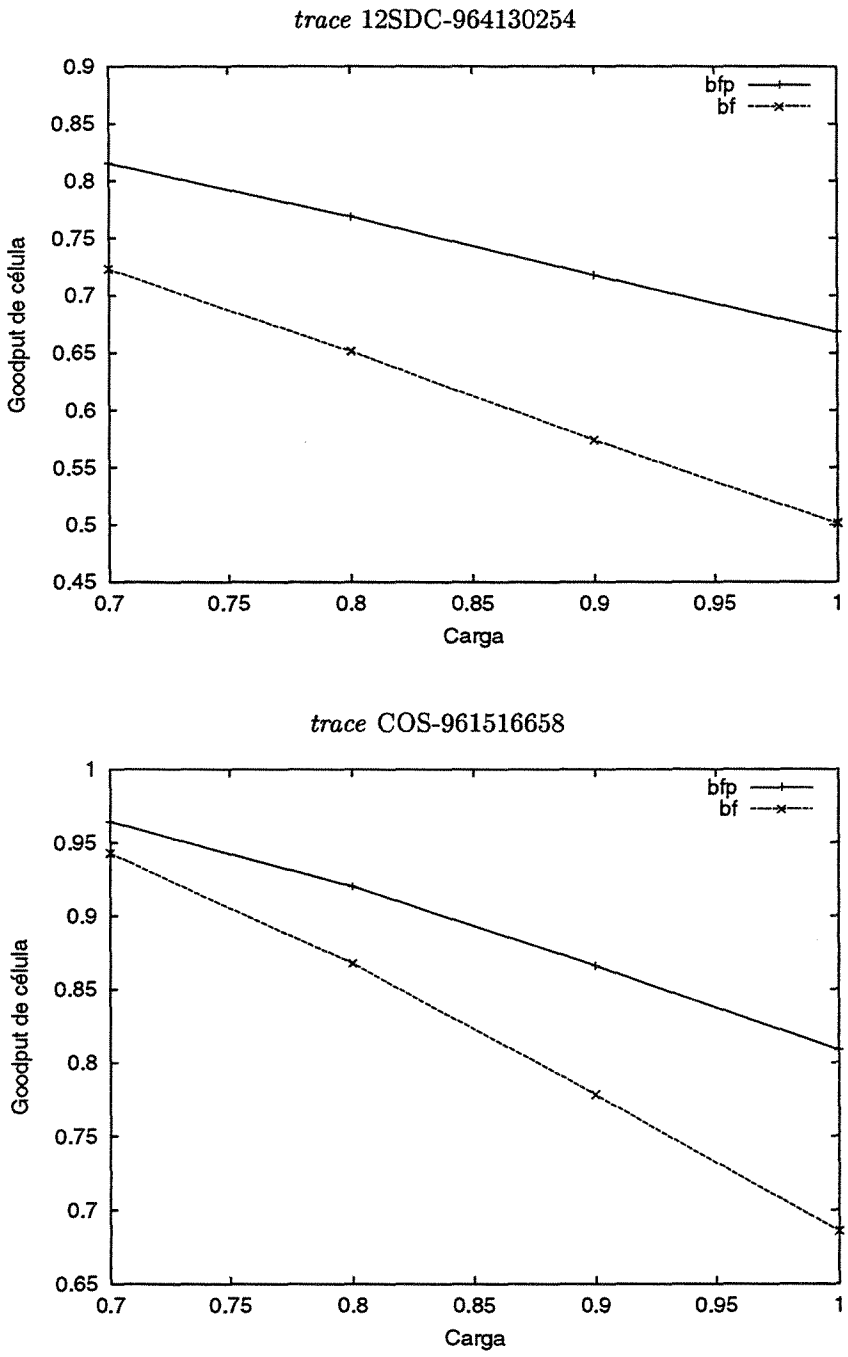


Figura 4.8: *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real.

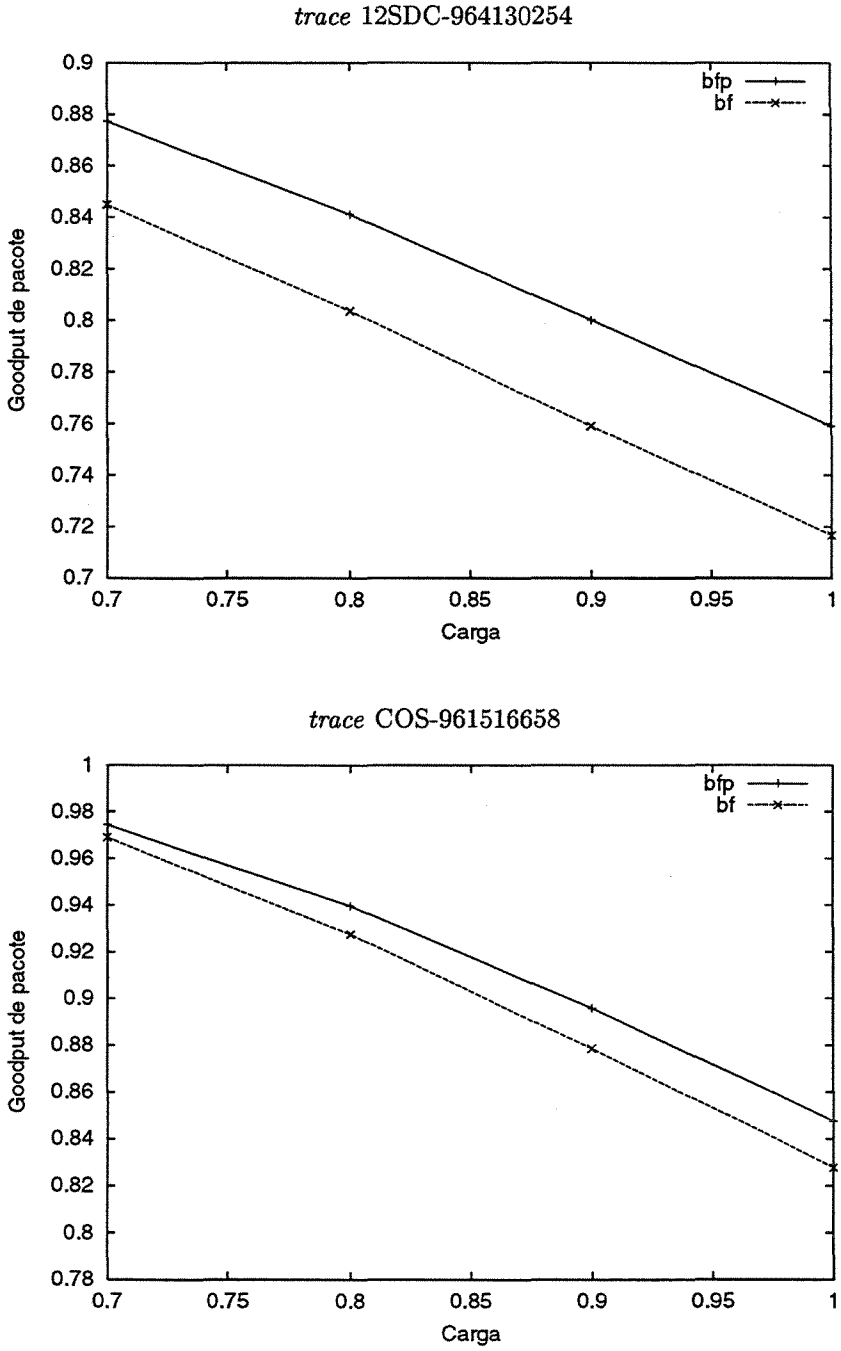


Figura 4.9: *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real.

Ambiente composto por vários nós

Nesta etapa investiga-se se o comportamento apresentado pelos mecanismos BFP e BF num ambiente composto por um único é mantido num ambiente de rede composto por vários nós. Os parâmetros utilizados para o fluxo agregado são os mesmos utilizados no ambiente composto por um único nó, descritos na seção 4.4.2, tabela 4.4. A tabela 4.5 sumariza os parâmetros de simulação utilizados para o fluxo de interferência.

Trace 12SDC-964130254		
FLUXO DE INTERFERÊNCIA	NÓ	MECANISMO DE POLICIAMENTO
N = 2 para cada nó H = 0.93 TMP = 16	TBu = 250 POB = CCD MD = MFIFD	para ρ_{Agr}=1.0: ρ_{Int}=0.12, Tx=0.062 e TBa=16 para ρ_{Agr}=0.9: ρ_{Int}=0.11, Tx=0.056 e TBa=16 para ρ_{Agr}=0.8: ρ_{Int}=0.10, Tx=0.050 e TBa=16 para ρ_{Agr}=0.7: ρ_{Int}=0.08, Tx=0.044 e TBa=16
Trace COS-9615166581		
FLUXO DE INTERFERÊNCIA	NÓ	MECANISMO DE POLICIAMENTO
N = 16 H = 0.91 TMP = 11	TBu = 150 POB = CCD MD = MFIFD	para ρ_{Agr}=1.0: ρ_{Int}=0.12, Tx=0.062 e TBa=11 para ρ_{Agr}=0.9: ρ_{Int}=0.11, Tx=0.056 e TBa=11 para ρ_{Agr}=0.8: ρ_{Int}=0.10, Tx=0.050 e TBa=11 para ρ_{Agr}=0.7: ρ_{Int}=0.08, Tx=0.044 e TBa=11

N-Número de fontes
H-Parâmetro de Hurst
TMP-Tamanho médio do pacote em células
TBu-Tamanho do *buffer* em células
POB-Política de organização do *buffer*
MD-Mecanismo de descarte
ρ_{Agr}-Carga do fluxo agregado
ρ_{Int}-Carga do fluxo de interferência
Tx-Taxa de geração de fichas por *time slot*
TBa-Tamanho do balde em células

Tabela 4.5: Sumário dos parâmetros utilizados nas simulações com tráfego real, fluxo de interferência.

Na Figura 4.10, mostra-se o *goodput* de célula em função da carga da rede. O comportamento apresentado por ambos mecanismos se mantém, ou seja, BFP fornece *goodput* de célula superior independente da carga da rede. Além disso, no ambiente com vários nós a diferença entre o *goodput* de célula fornecido pelos mecanismos torna-se maior. Por exemplo, no caso do *trace* 12SDC-964130254 e carga 1.0, a diferença entre o *goodput* de célula fornecido por BFP e BF é de 0.175, para o ambiente com um único nó, ao passo que no ambiente com 5 nós a diferença sobe para 0.22.

Em relação ao *goodput* de pacote, o comportamento apresentado pelos mecanismos no ambiente com 5 nós também se mantém, com o desempenho superior do mecanismo

BFP, independente da carga da rede (Figura 4.11). Também, no ambiente com vários nós a diferença entre o *goodput* de pacote fornecido pelos mecanismos é ligeiramente menor. Por exemplo, no caso do *trace* 12SDC-964130254 e carga 1.0, diferença entre o *goodput* de célula fornecido por BFP e BF foi de 0.039, para o ambiente com um único nó, ao passo que no ambiente com 5 nós a diferença cai para 0.029.

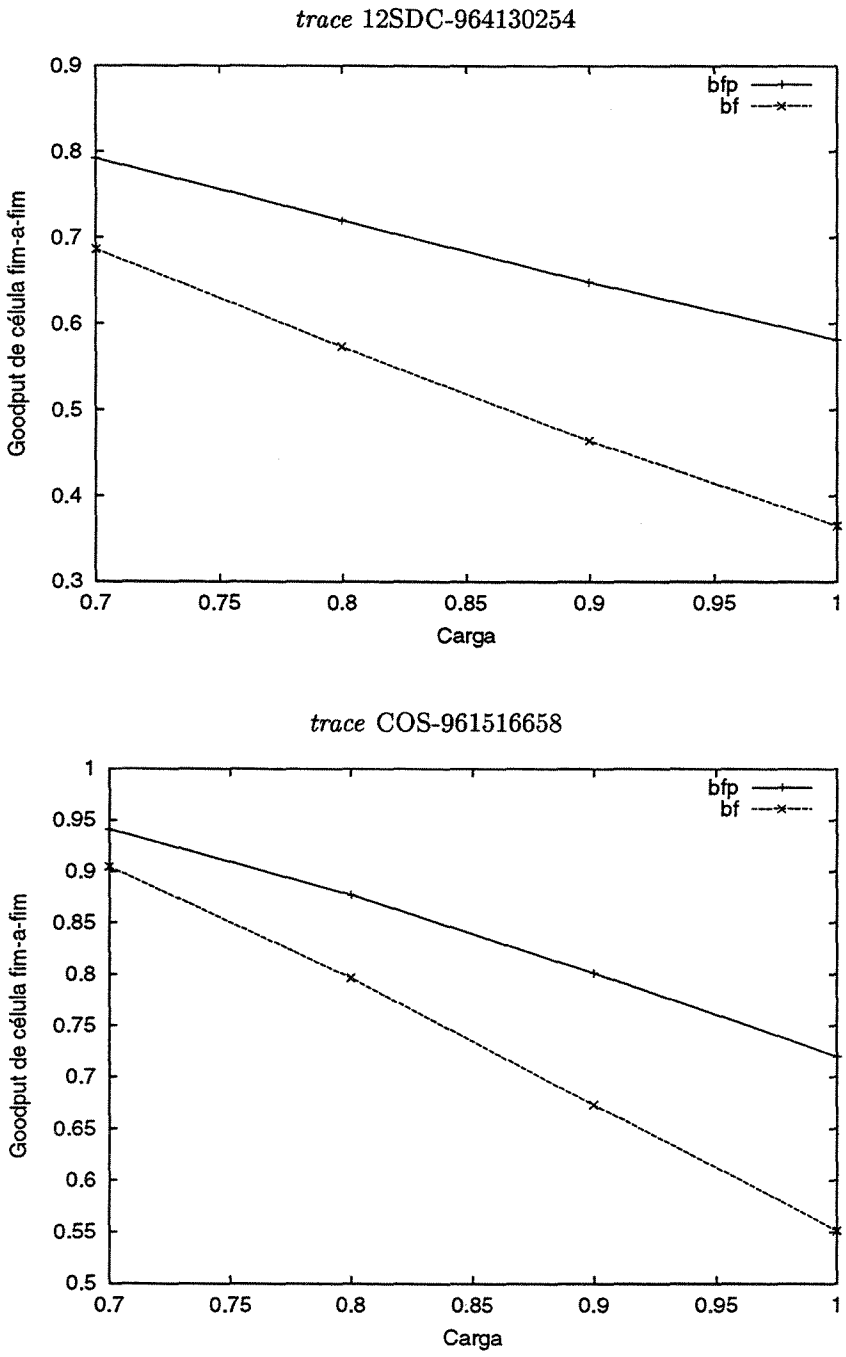


Figura 4.10: *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

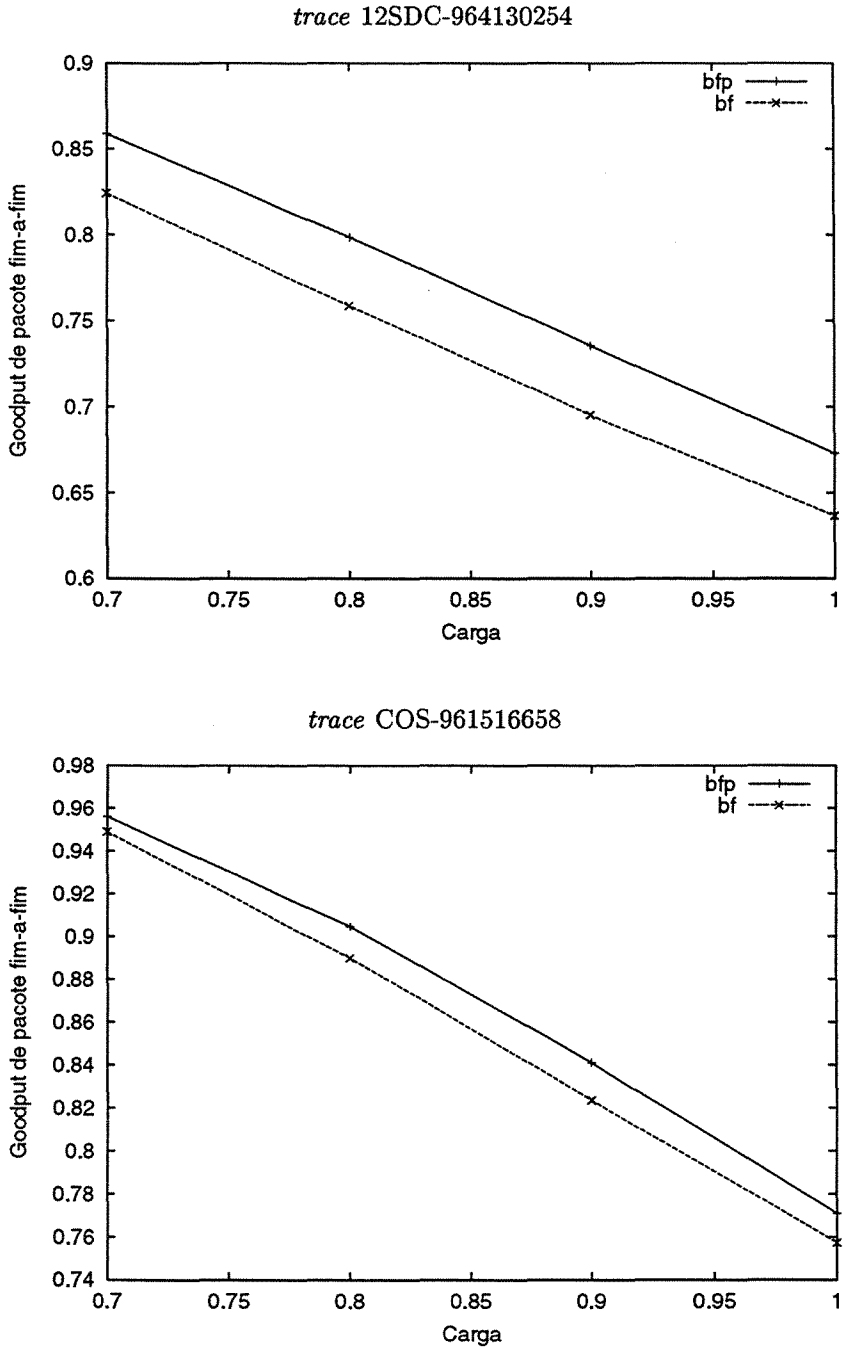


Figura 4.11: *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

4.4.3 Tráfego Sintético X Tráfego Real

Uma explicação plausível para as diferenças de desempenho apresentadas pelos mecanismos BF e BFP quando submetidos a tráfego sintético e tráfego real é a seguinte. No caso do tráfego sintético, o tamanho do pacote em células (período *on*) é gerado através da distribuição de Pareto com média 10. Sabe-se que a distribuição de Pareto é uma distribuição do tipo cauda-longa (*heavy tailed*), ou seja, uma distribuição que decai hiperbolicamente. Na prática isto significa que a probabilidade de que observações (período *on*) extremamente grandes ocorram não é negligenciável. Assim, uma amostra de uma distribuição de cauda-longa resulta num conjunto com a maioria dos valores "pequenos" e alguns valores extremamente grandes (valores observados nos experimentos de simulação para o tamanho do período *on* variaram de 3 a 2.729.075). De acordo com o mecanismo BFP (seção 4.1), um pacote que teve sua primeira célula marcada com prioridade alta terá todas as suas células marcadas com prioridade alta e, caso a quantidade de fichas do balde positivo não seja suficiente para atender todas as células do pacote, fichas são geradas no balde negativo, indicando a quantidade de células com prioridade alta enviadas além do permitido. Dessa forma um pacote extremamente grande que começa a ser marcado com prioridade alta irá gerar um número positivo também extremamente grande de fichas no balde negativo e, conseqüentemente, uma grande quantidade dos pacotes subseqüentes serão marcados com prioridade baixa até que o número de fichas do balde negativo seja menor ou igual a zero. O resultado é uma cadeia de transmissão formada por um pacote extremamente grande com prioridade alta seguido por vários pacotes menores com prioridade baixa. Com o mecanismo BF a situação é diferente. O pacote extremamente grande que começa a chegar terá apenas algumas de suas células marcadas com prioridade alta e o restante das células marcadas com prioridade baixa, de modo que as fichas geradas após a transmissão deste pacote e a chegada do próximo pacote (período *off*) serão usadas normalmente para marcar os pacotes subseqüentes com prioridade alta. O resultado é uma cadeia de transmissão formada por um pacote extremamente grande com a grande maioria das suas células marcadas com prioridade de baixa seguido por vários pacotes menores marcados com prioridade alta.

No caso do tráfego real a situação descrita não ocorre, pois tamanho do maior pacote identificado nos *traces* possui 32 células ATM, número de células necessárias para se transmitir um pacote *Ethernet* cujo MTU é de 1500-*bytes*.

Capítulo 5

Descarte de Informações em Redes ATM

Fragmentação em redes ATM é quase inevitável devido ao tamanho relativamente pequeno e fixo das células ATM (53-*bytes*). O problema da fragmentação em redes ATM levou ao desenvolvimento de várias políticas de descarte de pacotes com o objetivo de se transportar tráfego IP sobre ATM de forma eficiente. Neste capítulo avalia-se a real necessidade de se implementar políticas de descarte de pacotes em redes ATM diante dessa nova situação onde o tráfego IP sobre ATM é policiado e, conseqüentemente, células de dados do usuário podem ser seletivamente descartadas por mecanismos de descarte de célula. Além disso, avalia-se se existem vantagens em se integrar mecanismos de descarte de célula e políticas de descarte de pacotes. Para tal, introduz-se duas políticas: EPD+MFIFD e *Taildrop*+MFIFD que correspondem, respectivamente, à integração entre as políticas EPD e *Taildrop* com o mecanismo de descarte de células MFIFD. Por fim, avalia-se o algoritmo BFP no contexto da categoria de serviço GFR definido pelo ATM Forum.

Este capítulo está organizado da seguinte maneira: Na seção 5.1, compara-se mecanismos de descarte de células com políticas de descarte de pacotes. Na seção 5.2, introduz-se as políticas EPD+MFIFD e *Taildrop*+MFIFD utilizadas na integração entre descarte de células e descarte de pacotes. Tanto na seção 5.1 como na seção 5.2 resultados numéricos são apresentados para um ambiente formado por um único nó e para um ambiente com vários nós . A seção 5.3 avalia o algoritmo do BFP dentro do contexto da classe de serviço GFR.

5.1 Descarte de Células vs. Descarte de Pacotes

Nessa seção compara-se o mecanismo de descarte de célula MFIFD com as políticas de descarte de pacote *Taildrop* e EPD. A motivação para esta comparação é que embora comutadores possam implementar políticas de descarte de pacote, isso não é um requisito obrigatório, ao passo que mecanismos de policiamento e mecanismos de descarte de célula são padrões já estabelecidos em redes ATM. Além disso, a categoria de serviço UBR com descarte de pacote não prove garantias de QoS, o que pode ser alcançado com policiamento e descarte de células. Analisa-se o compromisso entre maximizar o número de células transmitidas em pacotes com sucesso e maximizar o número de pacotes transmitidos com sucesso.

Nos experimentos de simulação, utiliza-se o modelo de simulação descrito na seção 4.3. Nas simulações com as políticas de descarte de pacotes (*Taildrop* e EPD) o tráfego gerado pelas fontes não é policiado. A política EPD descarta pacotes completos toda vez que o nível de ocupação do *buffer* alcança um determinado limite chamado de *threshold.* O desempenho de EPD está diretamente relacionado ao valor do *threshold* escolhido. No entanto, a escolha do *threshold* depende de uma variedade de parâmetros do sistema, o que dificulta o ajuste deste para um valor ótimo [LB99]. Neste trabalho, o *threshold* da política EPD é configurado para 70% e 80% do tamanho do *buffer*, sem preocupação com a escolha de um valor ótimo. Investiga-se o impacto da carga da rede e parâmetro de Hurst no *goodput* de célula e no *goodput* de pacote utilizando tanto tráfego sintético como tráfego real e ambiente de rede formado por um único nó ou por vários nós. Os parâmetros utilizados nas simulações são os mesmos utilizados no capítulo 4, tabelas 4.2 e 4.3 para tráfego sintético e tabelas 4.4 e 4.5 para tráfego real.

5.1.1 Tráfego Sintético

Na Figura 5.1, mostra-se o *goodput* de célula como função da carga da rede num ambiente com um único nó, e para diferentes valores do parâmetro de Hurst. As legendas "epd70" e "epd80" significam que a política de descarte de pacote EPD é aplicada com um valor de *threshold* igual a 70% e 80% do *buffer*, respectivamente. Independente da carga ou do parâmetro de Hurst, EPD e *Taildrop* fornecem *goodput* de célula superior aos mecanismos de descarte de célula, BF E BFP. A política *EPD* fornece o maior *goodput* de pacote para ambos valores do *threshold.* De fato, a variação no valor do *threshold* de 70% para 80% do tamanho do *buffer* gerou resultados muito próximos para o *goodput* de célula desta política. Além disso, EPD é capaz de manter o *goodput* de célula acima de 0.68, mesmo sob um fluxo com H=0.9, o que não acontece comos mecanismos BF e BFP, onde o *goodput* de célula fica abaixo de 0.46. Nota-se que mecanismos de descarte de célula são mais sensíveis ao parâmetro de Hurst do que as políticas de descarte de

pacote. Por exemplo, para carga de 0.7, o aumento do parâmetro H de 0.7 para 0.9 causa as seguintes reduções no *goodput* de célula: 0.23 para EPD, 0.29 para *Taildrop*, 0.350 para BFP e 0.359 para BF.

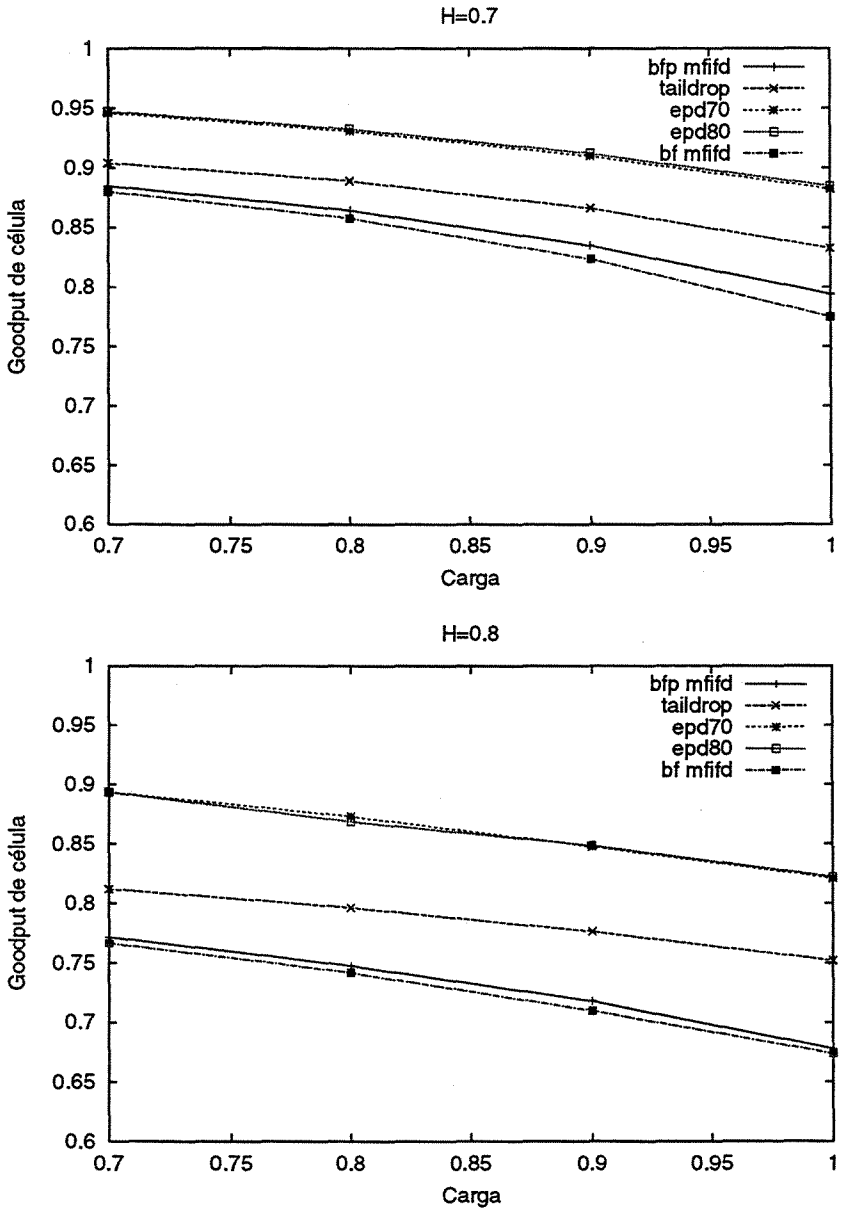


Figura 5.1a

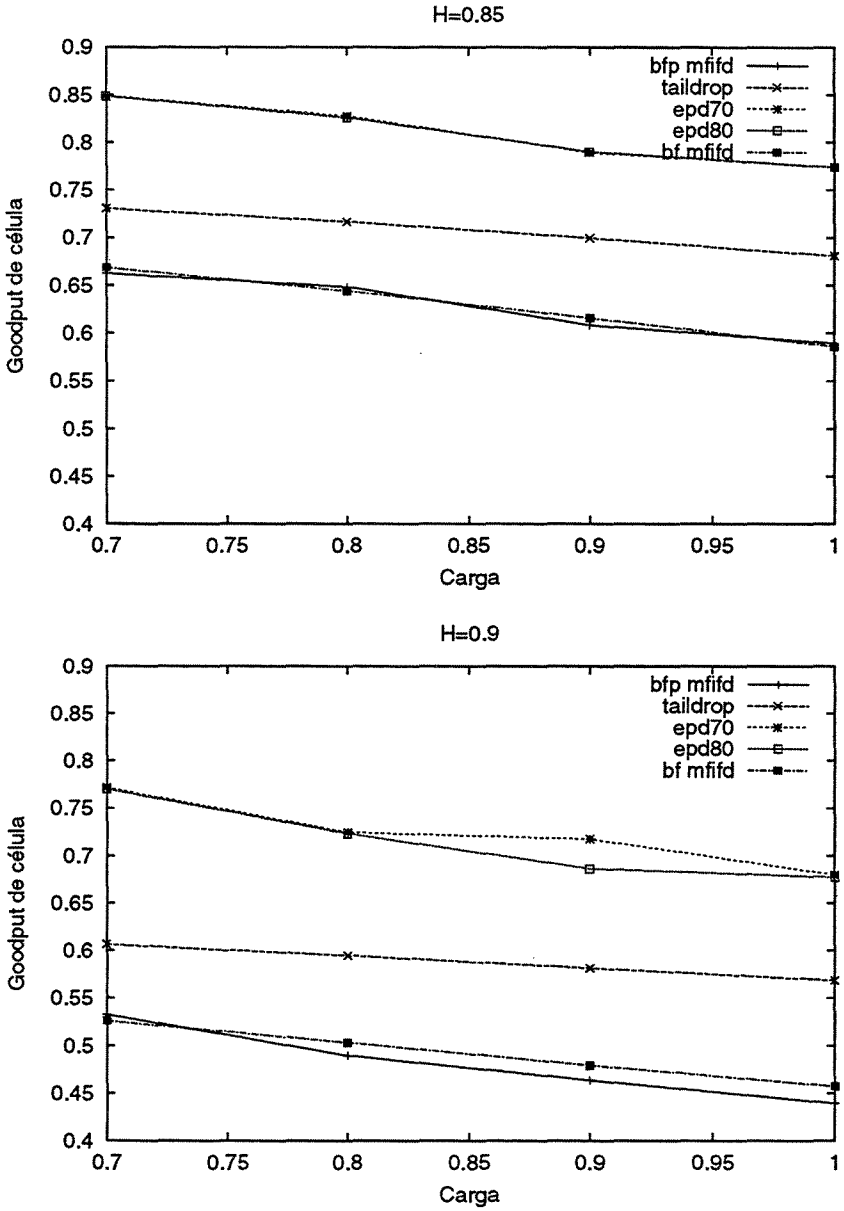


Figura 5.1b

Figura 5.1: *Goodput de célula vs.* Carga num ambiente com 1 nó e tráfego sintético.

Na Figura 5.2, mostra-se o *goodput* de célula como função da carga da rede num ambiente com cinco nós, e para diferentes valores do parâmetro de Hurst. Nota-se que com o aumento do número de nós da rede, as políticas de descarte de pacote ainda fornecem *goodput* de célula superior aos mecanismos de descarte de célula, independente da carga ou do parâmetro de Hurst, entretanto, o *goodput* de célula fornecido por *Taildrop*

e, principalmente, por EPD, tem redução mais acentuada do que o fornecido por BF e BFP. Por exemplo, para carga 1.0 e parâmetro H=0.85, o aumento do número de nós de 1 para 5 causa a seguintes reduções no *goodput* de célula: 0.185 para EPD, 0.09 para *Taildrop*, 0.07 para BFP e 0.7 para BF. De fato, a redução no *goodput* de célula de EPD é tão acentuada com o aumento do número de nós da rede, que num ambiente com 5 nós, carga 1.0 e parâmetro H>0.8, EPD e *Taildrop* fornecem *goodput* de célula quase indistinguíveis.

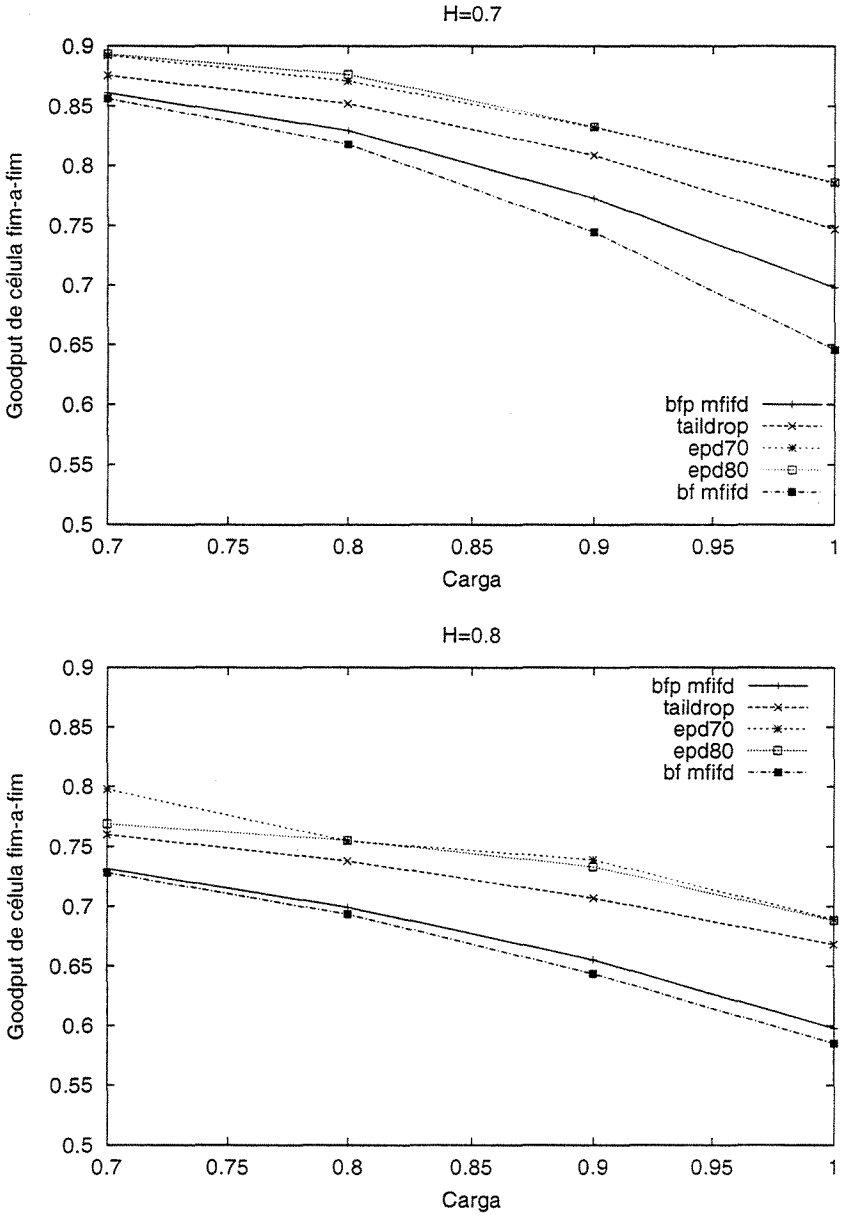

H=0.7
Goodput de célula fim-a-fim
Carga
bfp mfifd
taildrop
epd70
epd80
bf mfifd
H=0.8
Goodput de célula fim-a-fim
Carga
bfp mfifd
taildrop
epd70
epd80
bf mfifd

Figura 5.2a

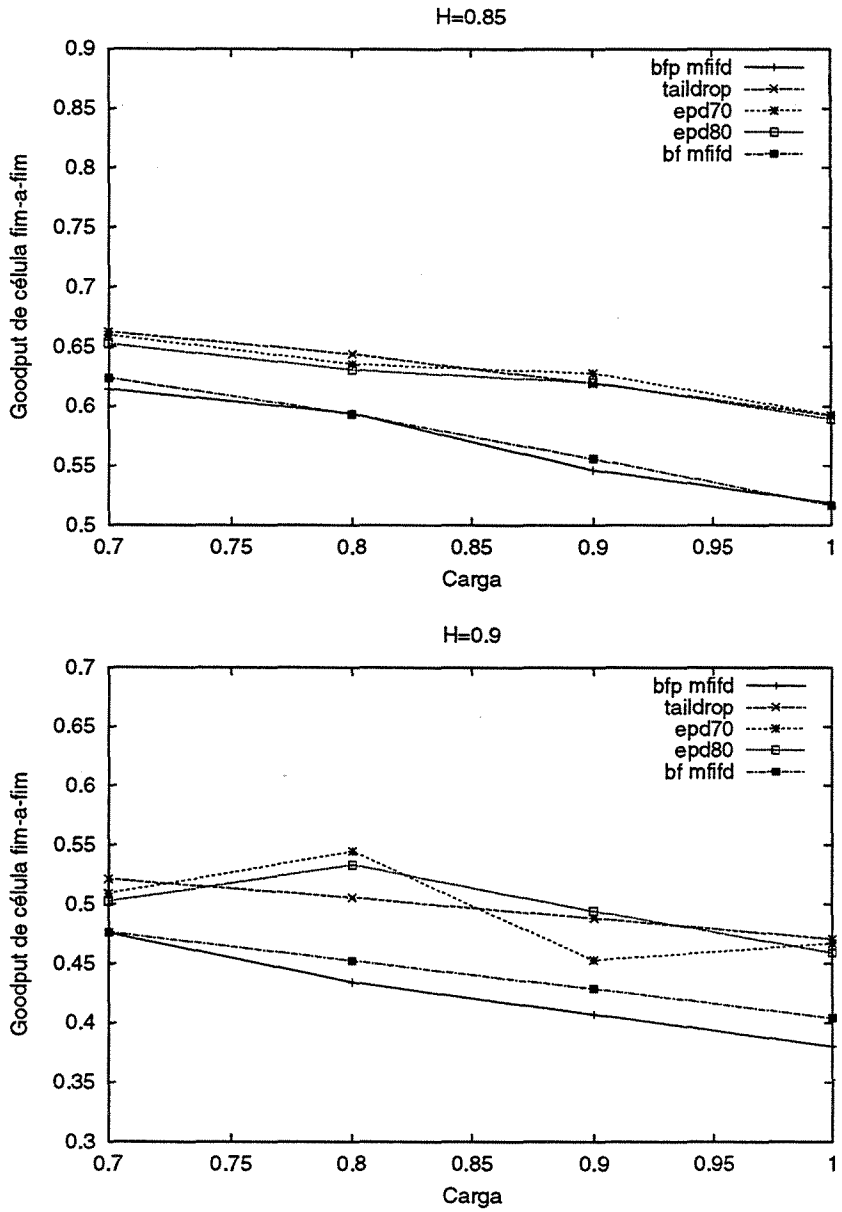


Figura 5.2b

Figura 5.2: *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético

Por outro lado, considerando o *goodput* de pacote, a política EPD tem desempenho inferior a política *Taildrop* e mecanismos BF e BFP, seja num ambiente de rede formado por um único ou vários nós, como mostram as Figuras 5.3 e 5.4 (salvo para H=0.9 e ambiente com um único nó, onde EPD80 obteve *goodput* de pacote superior BFP para carga maior

que 0.8, veja Figura 5.3a). Num ambiente com um único nó (Figura 5.3), *Taildrop* e em seguida BF, fornecem *goodput* de pacote superior a BFP e EPD, independente da carga ou do parâmetro de Hurst. *Taildrop* e BF mantêm o *goodput* de pacote acima de 0.92, mesmo sob um fluxo com H=0.9, o que não acontece com BFP e EPD, onde o *goodput* de pacote fica abaixo de 0.79. Nota-se que BF e principalmente *Taildrop* são menos sensíveis ao parâmetro de Hurst do que BFP e EPD. Por exemplo, para carga de 0.7, o aumento do parâmetro H de 0.7 para 0.9 causa as seguintes reduções no *goodput* de pacote: 0.004 para *Taildrop*, 0.025 para BF, 0.09 para BFP e 0.15 para EPD80 e EPD70.

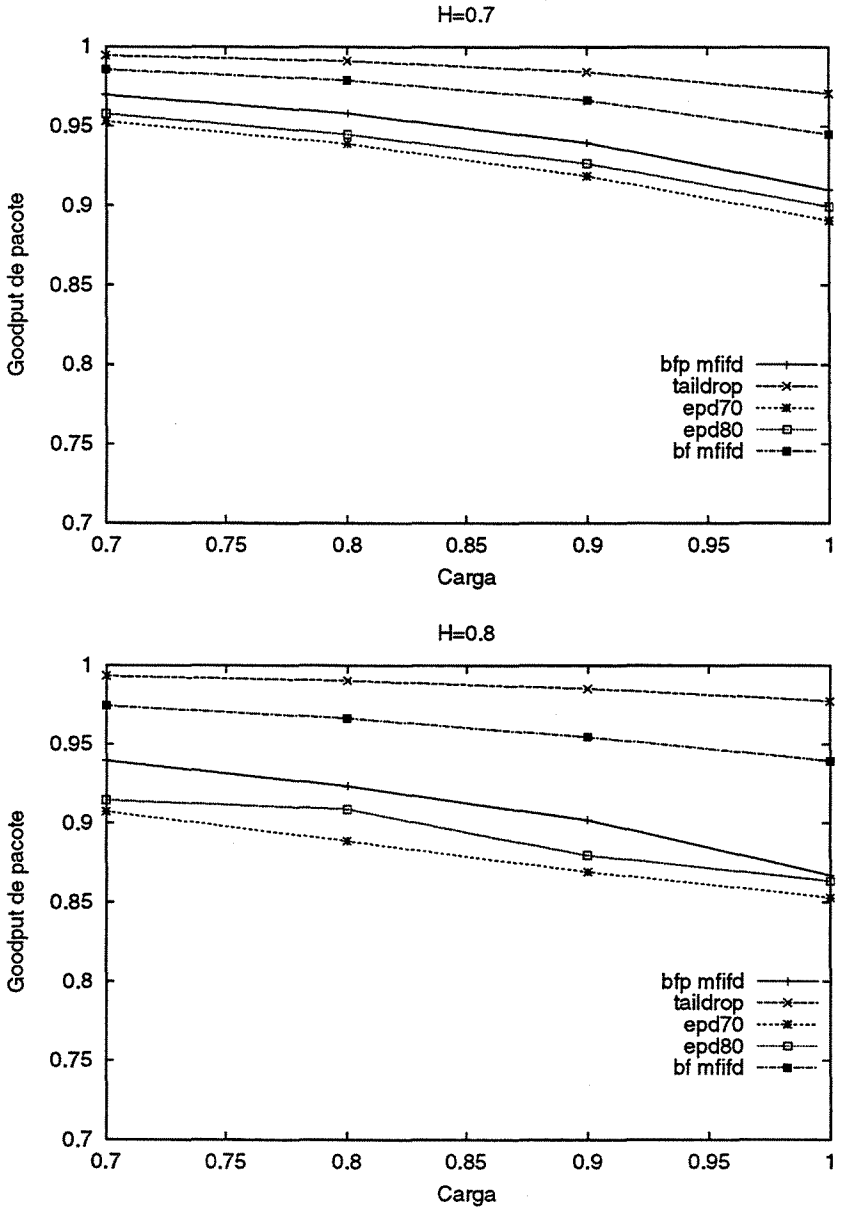

H=0.7
Goodput de pacote
Carga
bfp mfifd
taildrop
epd70
epd80
bf mfifd
1
0.95
0.9
0.85
0.8
0.75
0.7
0.7
0.75
0.8
0.85
0.9
0.95
1
H=0.8
Goodput de pacote
Carga
bfp mfifd
taildrop
epd70
epd80
bf mfifd

Figura 5.3a

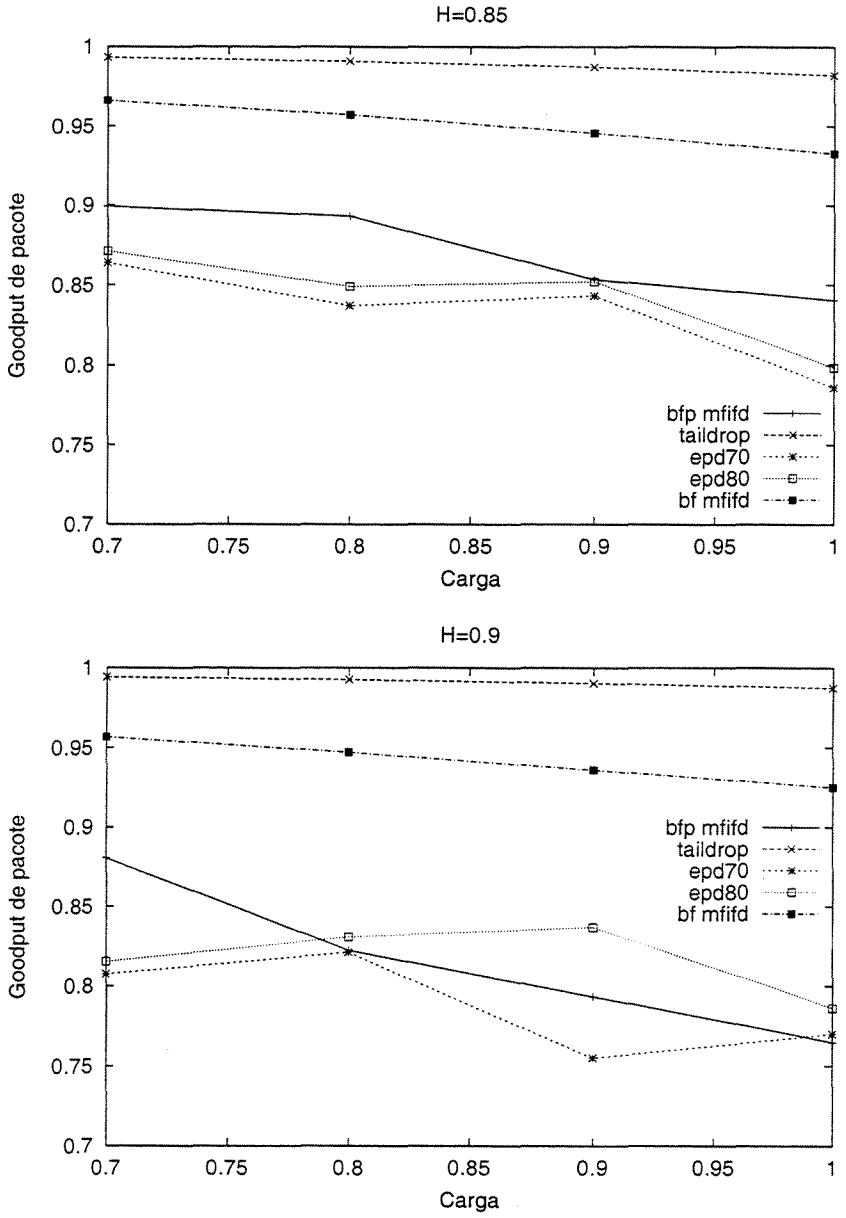


Figura 5.3b

Figura 5.3: *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego sintético.

Com o aumento do número de nós da rede, a política *Taildrop* e o mecanismo BF ainda fornecem *goodput* de pacote superior a BFP EPD, independente da carga ou do parâmetro de Hurst (Figura 5.4). Contudo, o aumento do número de nós causa maior impacto no *goodput* de pacote fornecido por *Taildrop* do que o fornecido por BF, de modo ao contrário do que acontece no ambiente com um único nó, no ambiente com cinco nós

o mecanismo BF fornece *goodput* de célula superior a *Taildrop*. Por exemplo, para carga 1.0 e parâmetro H=0.9, o aumento do número de nós de 1 para 5 causa as seguintes reduções no *goodput* de pacote: 0.035 para BF e 0.11 para *Taildrop*, ou seja, a queda de desempenho de *Taildrop* foi 3 vezes maior do que a queda de desempenho de BF.

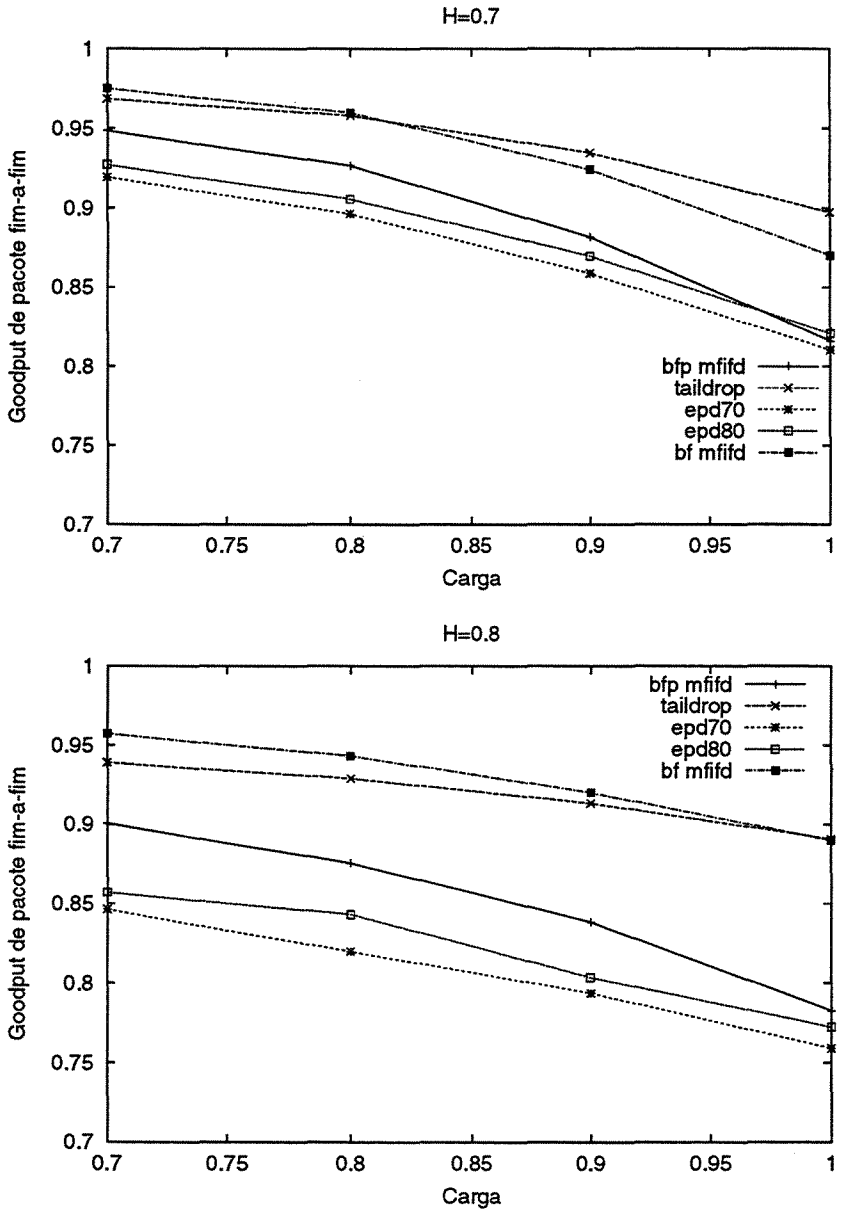


Figura 5.4a

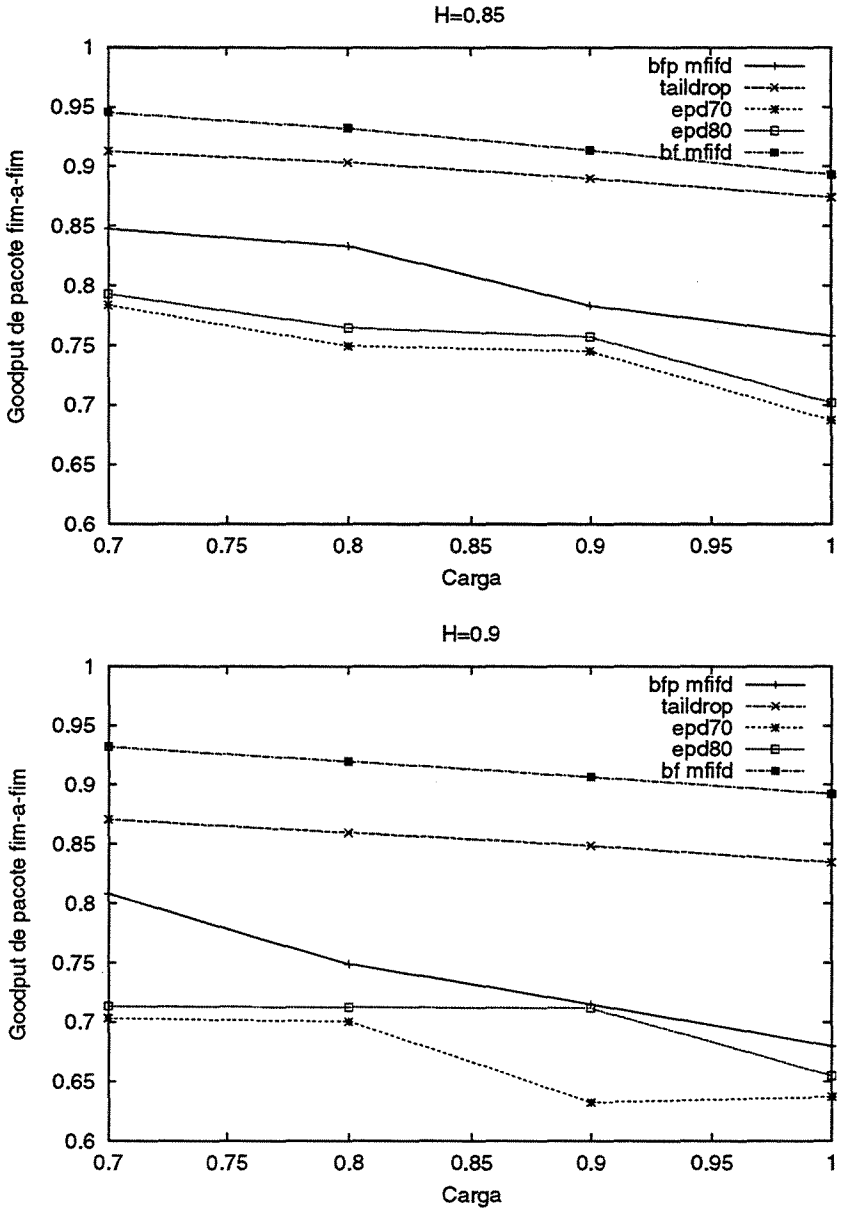


Figura 5.4b

Figura 5.4: *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego sintético.

5.1.2 Tráfego Real

Na Figura 5.5, mostra-se o *goodput* de célula em função da carga da rede num ambiente com um único nó. Nos *traces* utilizados, EPD fornece *goodput* de célula superior, seguido

pelos mecanismos BFP, *Taildrop* e BF. Nota-se que em alguns *traces*, *Taildrop* fornece *goodput* de célula superior a BFP, entretanto, a diferença foi sempre inferior a 0.01. O *trace* COS-961516658 é um exemplo. Neste caso, diferença máxima entre o *goodput* de célula fornecido por *Taildrop* e BFP foi de 0.005. Para carga 0.7 a diferença entre o *goodput* de célula fornecidos por EPD em relação a BFP e *Taildrop* é muito pequena, porém, EPD se mostrou mais estável ao aumento da carga que os demais mecanismos. Por exemplo, no caso do *trace* COS-961516658, o aumento da carga da rede de 0.7 para 1.0 causa as seguintes reduções no *goodput* de célula: 0.123 para EPD70, 0.123 para EPD80, 0.155 para BFP e 0.159 para *Taildrop*. Também ficou bem evidenciado, o desempenho muito inferior de BF em relação EPD, BFP e *Taildrop*, principalmente para carga maior que 0.8. Por exemplo, para carga 1.0 e *trace* COS-961516658, o *goodput* de célula fornecido foi de 0.850 para EPD80, 0.845 para EPD70, 0.809 para *Taildrop*, 0.809 para BFP e 0.686 para BF.

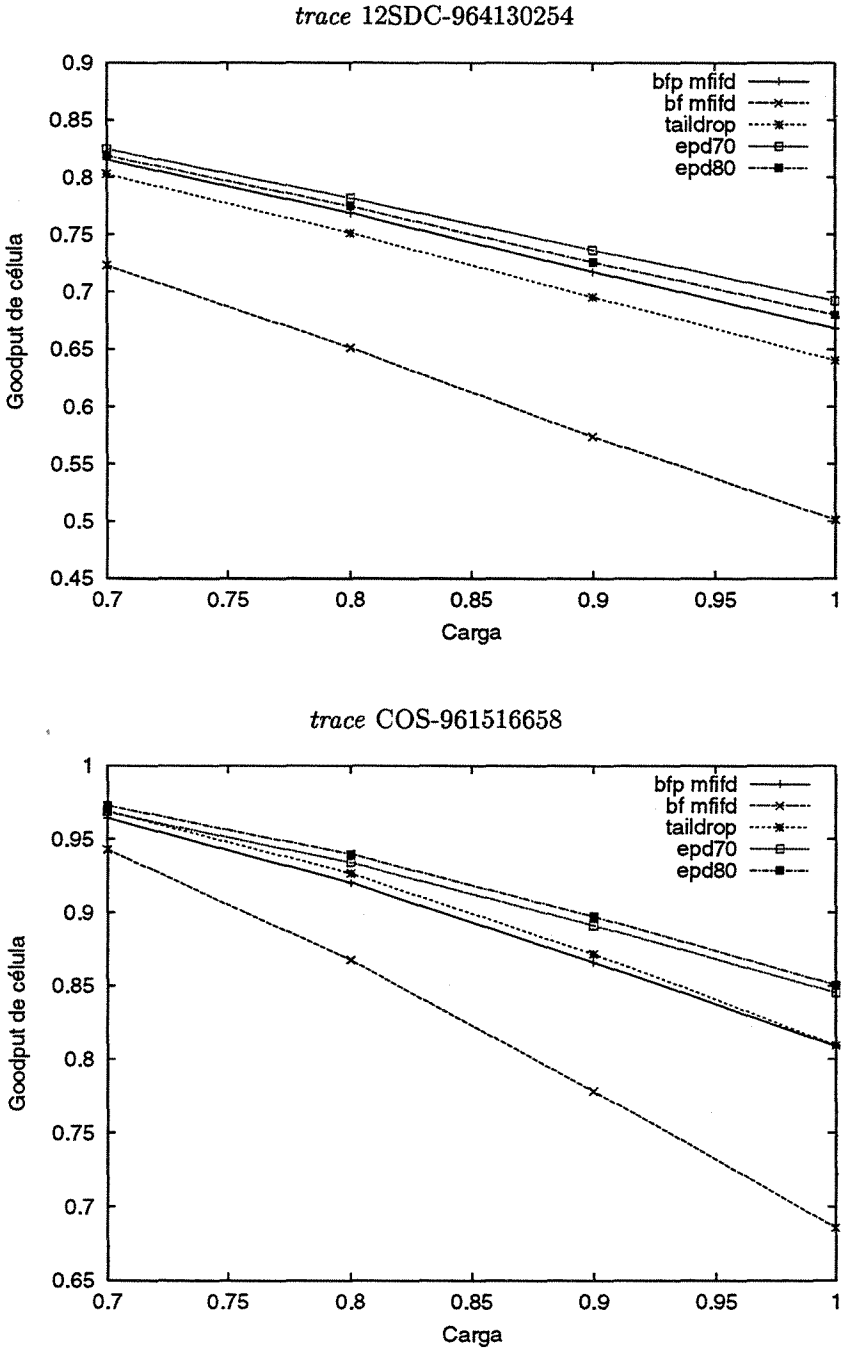


Figura 5.5: *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real.

Com o aumento do número de nós da rede, a ordem de desempenho se mantém, ou seja, EPD fornece *goodput* de célula superior, seguido por BFP, *Taildrop* e BF. Entretanto, EPD se mostrou mais estável ao aumento do número de nós da rede do que os demais

mecanismos. Por exemplo, para carga 1.0 e *trace* 12SDC-964130254, o aumento do número de nós da rede de 1 para 5 causou as seguintes reduções no *goodput* de célula: 0.059 para EPD70, 0.059 para EPD80, 0.086 para BFP, 0.089 para *Taildrop* e 0.136 para BF. Nota-se que mesmo com o aumento do número de nós, BFP se mostrou tão estável quanto *Taildrop*. O mecanismo BF se mostrou o menos estável ao aumento do número de nós da rede.

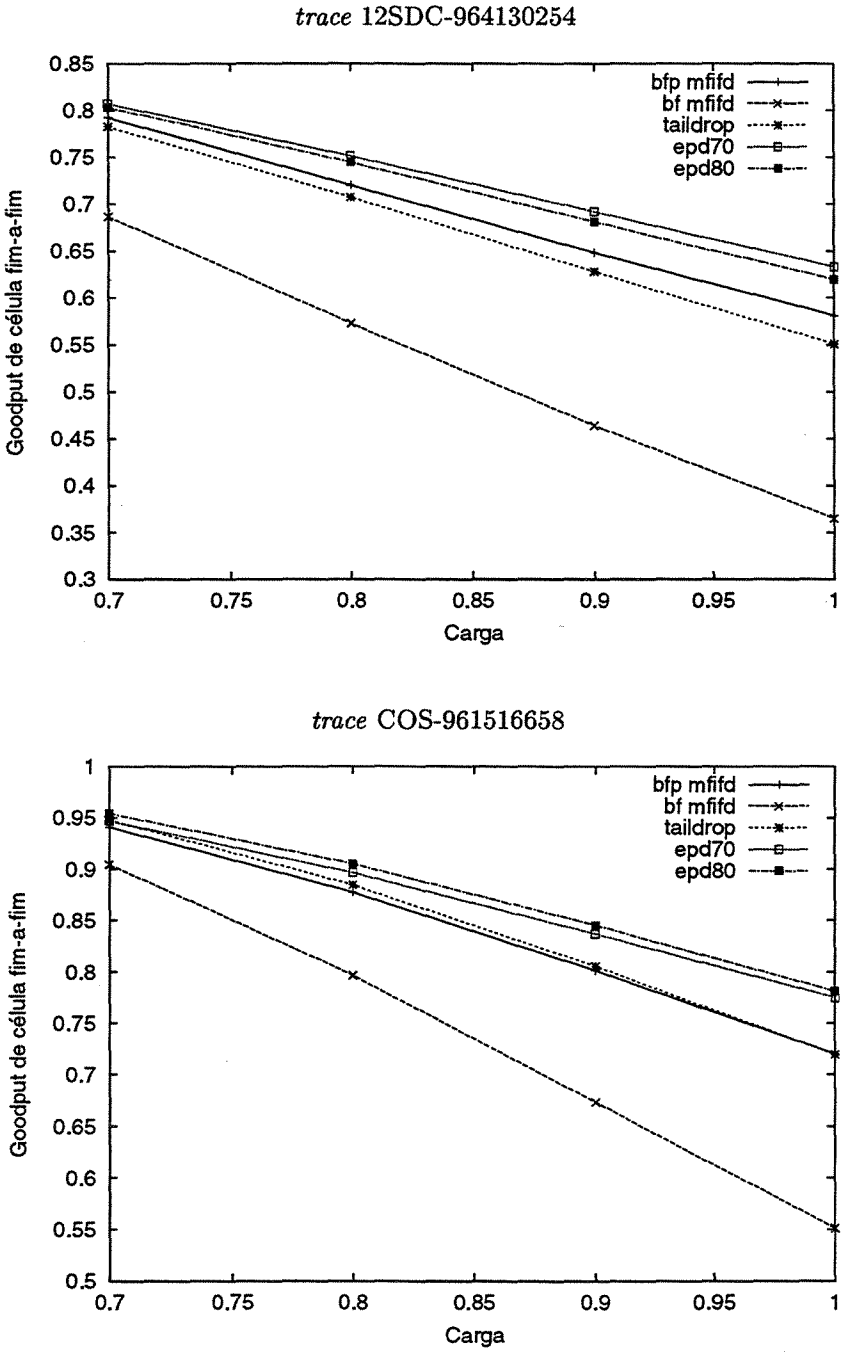


Figura 5.6: *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

Na Figura 5.7, mostra-se o *goodput* de pacote em função da carga da rede num ambiente com um único nó. Nos *traces* utilizados, *Taildrop* fornece *goodput* de pacote superior, seguido pelos mecanismos BFP, EPD e BF. No caso do *trace* COS-961516658, o *goodput*

de pacote oferecido por BF e EPD70 está muito próximo, (a diferença entre o *goodput* de pacote oferecido por EPD70 e BF foi de 0.001 para carga da rede 0.7 e 0.015 para carga da rede 1.0). Entretanto, nos demais *traces* nota-se que a diferença entre o *goodput* de pacote oferecido por EPD70 em relação BF e, conseqüentemente, por *Taildrop*, BFP e EPD80 em relação a BF, é acentuada. No caso do *trace* 12SDC-964130254 e carga 0.7, a diferença entre o *goodput* de pacote oferecido por EP70 e BF foi de 0.020, entre EPD80 e BF 0.036, entre BFP e BF 0.032 e entre *Taildrop* e BF a diferença foi de 0.036. Com o aumento da carga a diferença torna-se ainda mais acentuada. Por exemplo, para carga 1.0 a diferença foi de entre o *goodput* de pacote oferecido por EPD70 e BF foi de 0.036, entre EPD80 e BF 0.041, entre BFP e BF 0.042 e entre *Taildrop* e BF a diferença foi de 0.060.

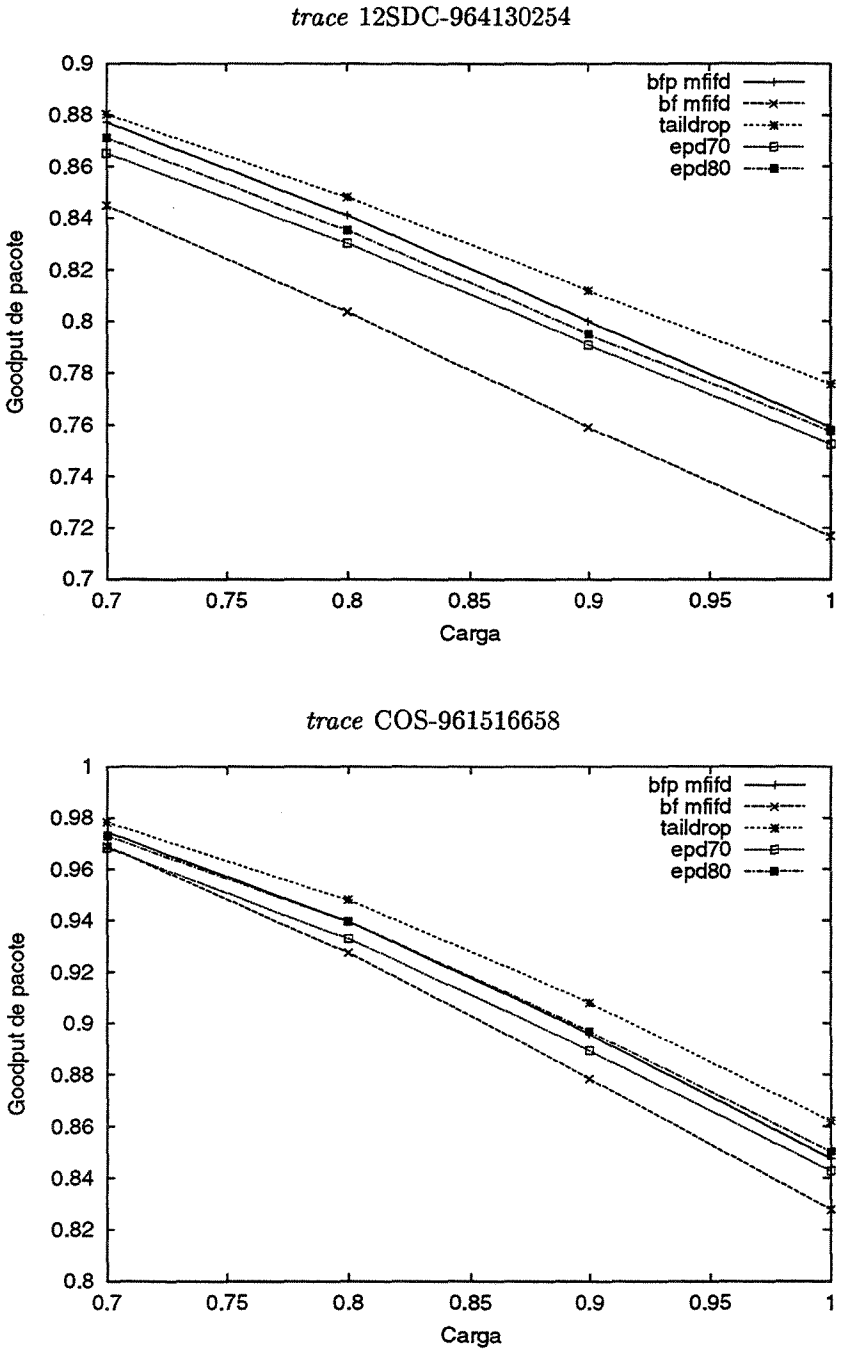


Figura 5.7: *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real.

Na Figura 5.8, mostra-se o *goodput* de pacote como função da carga da rede num ambiente com cinco nós. Tal como acontece no ambiente formado por um único nó, no ambiente com cinco nós as políticas *Taildrop* e BF forneceram, respectivamente, o maior

e o menor *goodput* de pacote. Por outro lado, BFP forneceu *goodput* de pacote inferior a EPD (veja por exemplo na Figura 5.8, o *trace* 12SDC-964130254 e cargas acima de 0.8). De fato, EPD se mostrou mais estável ao aumento do número de nós da rede do que BFP. Por exemplo, para carga 1.0 e *trace* 12SDC-964130254, o aumento do número de nós da rede de 1 para 5 causou as seguintes reduções no *goodput* de célula: 0.059 para EPD70, 0.060 para EPD80, 0.087 para BFP.

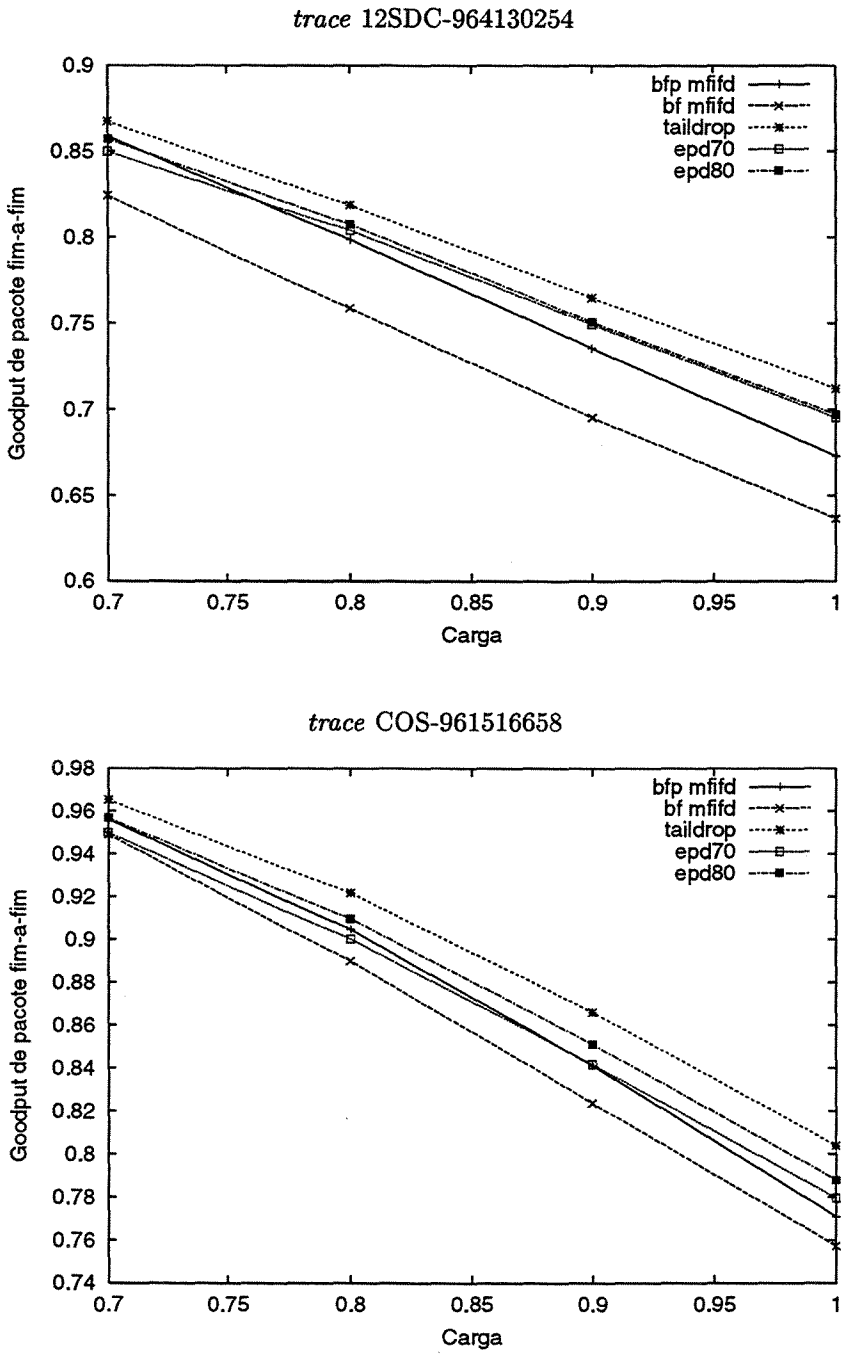


Figura 5.8: *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

Nota-se que BFP gerou desempenho mais balanceado entre a camada de transporte (*goodput* de pacote) e a camada de rede (*goodput* de célula) do que as políticas *Taildrop*

e EPD. Em outras palavras, quando a métrica de desempenho é o *goodput* de célula, a política EPD tem o melhor desempenho, seguida por BFP e por último *Taildrop*. Mudando a métrica de desempenho para o *goodput* de pacote, *Taildrop* passa a ter o melhor desempenho, BFP continua com o segundo melhor desempenho e EPD passa a ter o pior desempenho.

Por outro lado, resultados apresentados pelo mecanismo BF foram insatisfatórios, com desempenho em nível da camada de transporte e de rede inferiores a todas as outras políticas.

5.2 Integração entre Descarte de Células e Descarte de Pacotes

Esta seção visa estudar as vantagens em se integrar políticas de descarte de pacote com mecanismos de descarte de célula. Introduz-se duas novas políticas resultantes da integração entre EPD e *Taildrop* com o mecanismo de descarte de células MFIFD. Nas próximas seções encontra-se a descrição das políticas integradas propostas.

5.2.1 Integração entre Taildrop e MFIFD

O algoritmo 5.2.1 apresenta a integração proposta entre *Taildrop* e MFIFD. Na política *Taildrop*, uma célula que chega e encontra o *buffer* cheio é perdida por transbordo, e todas as células restantes do pacote o qual a célula pertence não são admitidas no *buffer*. No caso da integração entre *Taildrop* e MFIFD a idéia consistem em primeiro tentar liberar espaço em *buffer* por meio de MFIFD para que a célula que chega e encontra o *buffer* cheio possa ser admitida, ao invés de imediatamente descarta-la por transbordo.

Ao tentar liberar espaço em *buffer* por meio de MFIFD, duas situações podem ocorrer. Na primeira, existe uma célula marcada com prioridade baixa no *buffer* que é então descartada, e assim libera espaço para que a célula que chega seja admitida. Além disso, aplica-se *Taildrop* no pacote da célula descartada por MFIFD. Na segunda situação, não existem células marcadas com prioridade baixa no *buffer*, conseqüentemente nenhuma célula é descartada por MFIFD e a célula que chega é perdida por transbordo. Aplica-se *Taildrop* no pacote a qual a célula perdida por transbordo do *buffer* pertence. Através do algoritmo 5.2.1, pode-se notar que o mecanismo de descarte de células MFIFD tem precedência sobre a política de descarte de pacote *Taildrop*. Em outras palavras, quando uma célula chega e encontra o *buffer* cheio, o tentará primeiro retirar do *buffer* por meio de MFIFD uma célula marcada com prioridade baixa a assim liberar espaço para a célula que chega. Chamamos essa política integrada de *Taildrop*+MFIFD.

Algoritmo 2 POLÍTICA TAILDROP+MFIFD

Variáveis:

C_{chegou}: Célula que chegou para ser admitida no *buffer*
$C_{descartada}$: Célula descartada do *buffer* por MFIFD
ListaDescarte: Lista com os pacotes marcados para descarte por *Taildrop*

```
/* Durante a chegada de células */
se ( Pacote de C_chegou ∈ ListaDescarte )
   descarta C_chegou /* descarte por Taildrop */
   se "C_chegou era a última do Pacote"
      remove o Pacote de C_chegou da ListaDescarte
   fimse
senão
   /* Tenta admitir célula que chegou no buffer */
   se "buffer não está cheio"
      admite C_chegou no buffer
   senão
      /* Aplica descarte de célula */
      C_descartada := MFIFD
      se ( C_descartada = " " ) então
         /* Não há células com prioridade baixa no buffer */
         descarta C_chegou /* transbordo do buffer */
         inclui o Pacote de C_chegou na ListaDescarte
      senão
         /* Uma célula com prioridade baixa foi removida do buffer por MFIFD */
         admite C_chegou no buffer
        inclui o Pacote de C_descartada na ListaDescarte
      fimse
   fimse
fimse
```

5.2.2 Integração entre EPD e MFIFD

O algoritmo 5.2.2 apresenta a integração proposta. A política EPD descarta pacotes completos que chegam quando o nível de ocupação do *buffer* ultrapassa um valor máximo conhecido como *threshold*. No caso da integração entre EPD e MFIFD a idéia consiste em descartar, por meio do mecanismo MFIFD, células marcadas com prioridade baixa a fim

de deixar o nível de ocupação do *buffer* abaixo do valor de *threshold*, e assim permitindo a admissão de novos pacotes. Caso a quantidade de células marcadas com prioridade baixa não seja suficiente para deixar o nível de ocupação do *buffer* abaixo do *threshold*, nenhuma célula é descartada e EPD é aplicado normalmente, ou seja, os pacotes que chegam são descartados por completo. Chamamos essa política integrada de EPD+MFIFD.

Algoritmo 3 POLÍTICA EPD+MFIFD

Variáveis:

C_{chegou}: Célula que chegou para ser admitida no *buffer*
$C_{descartada}$: Célula descartada do *buffer* por MFIFD
ListaDescarte: Lista com os pacotes marcados para descarte por *Taildrop* e EPD
Threshold: Indicador de congestionamento da política EPD
CelulasPrioridadeBaixa: Quantidade atual de células com prioridade baixa armazenadas no *buffer*
OcupacaoBuffer: Quantidade atual de células armazenadas no *buffer*
i: Variável de controle

```
/* Durante a chegada de células */
se ( Pacote de C_chegou ∈ ListaDescarte )
   descarta C_chegou /* descarte por Taildrop ou EPD */
   se "C_chegou era a última do Pacote"
      remove o Pacote de C_chegou de ListaDescarte
   fimse
senão
   /* Tenta admitir célula que chegou no buffer */
   se "buffer não está cheio"
      se ( OcupacaoBuffer > Threshold E "C_chegou é a primeira do Pacote" )
         se ( CelulasPrioridadeBaixa >= (OcupacaoBuffer − Threshold ) )
            para (i := 1; i < (OcupacaoBuffer − Threshold); i++)
               C_descartada := MFIFD
               inclui o Pacote de C_descartada na ListaDescarte
            fimpara
            /* Agora OcupacaoBuffer <= Threshold */
            admite C_chegou no buffer
         senão
            descarta C_chegou /* descarte por EPD */
            inclui o Pacote de C_chegou na ListaDescarte
         fimse
      senão
         admite C_chegou no buffer
      fimse
   senão /* buffer está cheio */
      descarta C_chegou /* transbordo do buffer */
      inclui o Pacote de C_chegou na ListaDescarte
   fimse
fimse
```

5.2.3 Tráfego Real

A fim de avaliar a eficácia das políticas integradas *Taildrop*+MFIFD e EPD+MFIFD, compara-se essas com as políticas *Taildrop*, EPD e BFP. Analisa-se o compromisso entre maximizar o número de células transmitidas em pacotes com sucesso e maximizar o número de pacotes transmitidos com sucesso. Os parâmetros utilizados nas simulações são os mesmos utilizados no capítulo 4, tabelas 4.4 e 4.5.

Nas Figuras 5.9 e 5.10, mostra-se o *goodput* de célula como função da carga da rede num ambiente composto por um único nó e cinco nós, respectivamente. A legenda "BFP EPD80+MFIFD" significa que o tráfego é policiado pelo mecanismo BFP e o *threshold* da política integrada EPD+MFIFD é 80% do tamanho do *buffer*. Nota-se melhora no desempenho de BFP quando este é integrado com a política de descarte de pacote *Taildrop*. A explicação é que na política integrada BFP *Taildrop*+MFIFD, *Taildrop* atua de forma complementar a MFIFD, descartando a cauda de pacotes corrompidos por transbordo do *buffer* ou pelo mecanismo descarte de células MFIFD. Quando MFIFD atua de forma isolada, células inúteis pertencentes a cauda de pacotes corrompidos são admitidas no *buffer* contribuindo para o congestionamento. Por exemplo, no caso da Figura 5.9, carga 1.0 e *trace* 12SDC-964130254, o *goodput* de célula foi de 0.640 para *Taildrop*, 0.668 para BFP e 0.682 para BFP *Taildrop*+MFIFD. Com relação a integração entre EPD e MFIFD, os resultados não mostraram ganhos. BFP EPD+MFIFD apresentou *goodput* de de célula inferior ao apresentado quando EPD e BFP foram utilizados de forma isolada, mostrando não ser uma boa estratégia descartar um conjunto de células do início do *buffer* em função de apenas um pacote que chega. Por exemplo, para carga 1.0, *trace* COS-961516658 e ambiente de rede com um único nó, o *goodput* de célula foi de 0.850 para EPD80, 0.809 para BFP e 0.803 para BFP EPD80+MFIFD.

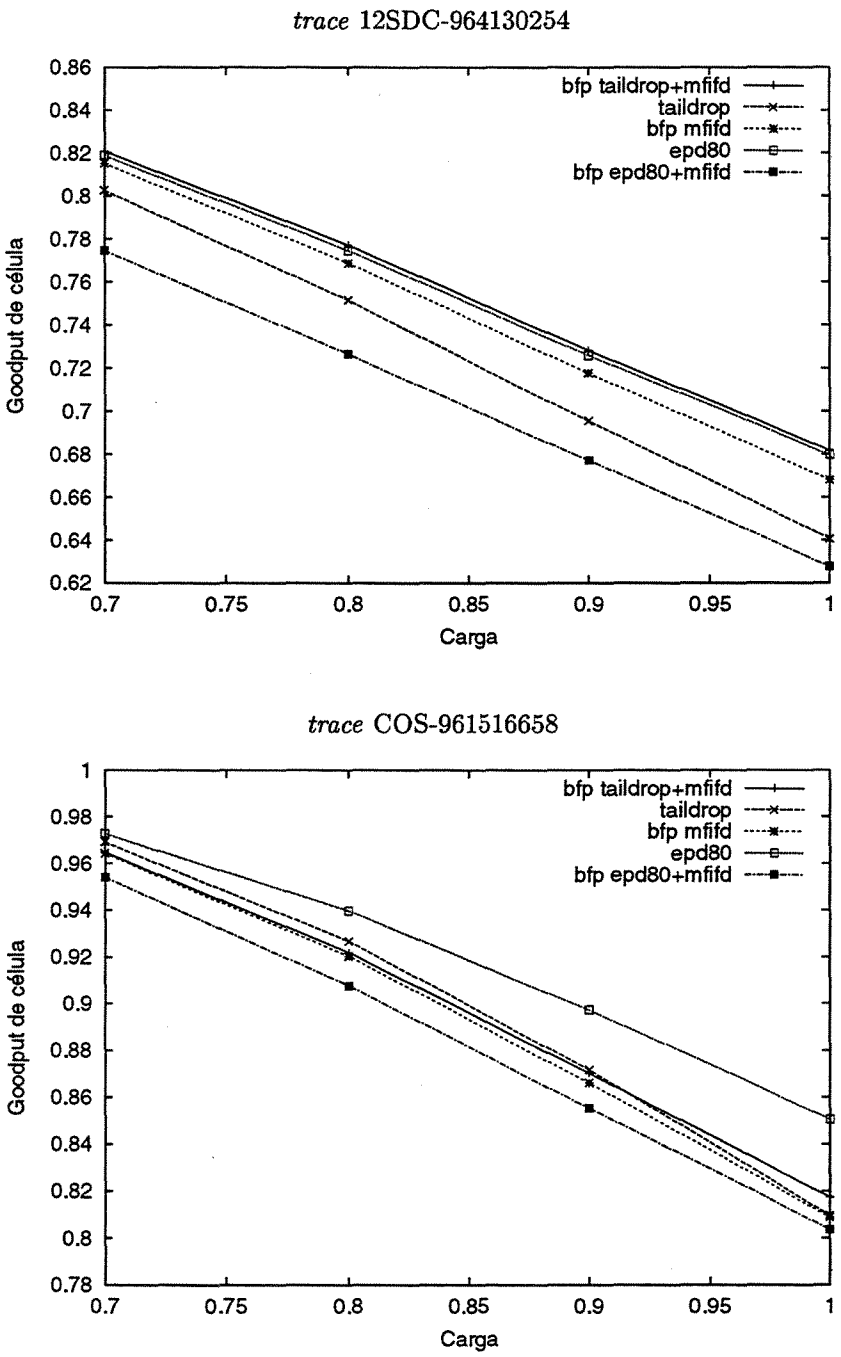


Figura 5.9: *Goodput* de célula *vs.* Carga num ambiente com 1 nó e tráfego real.

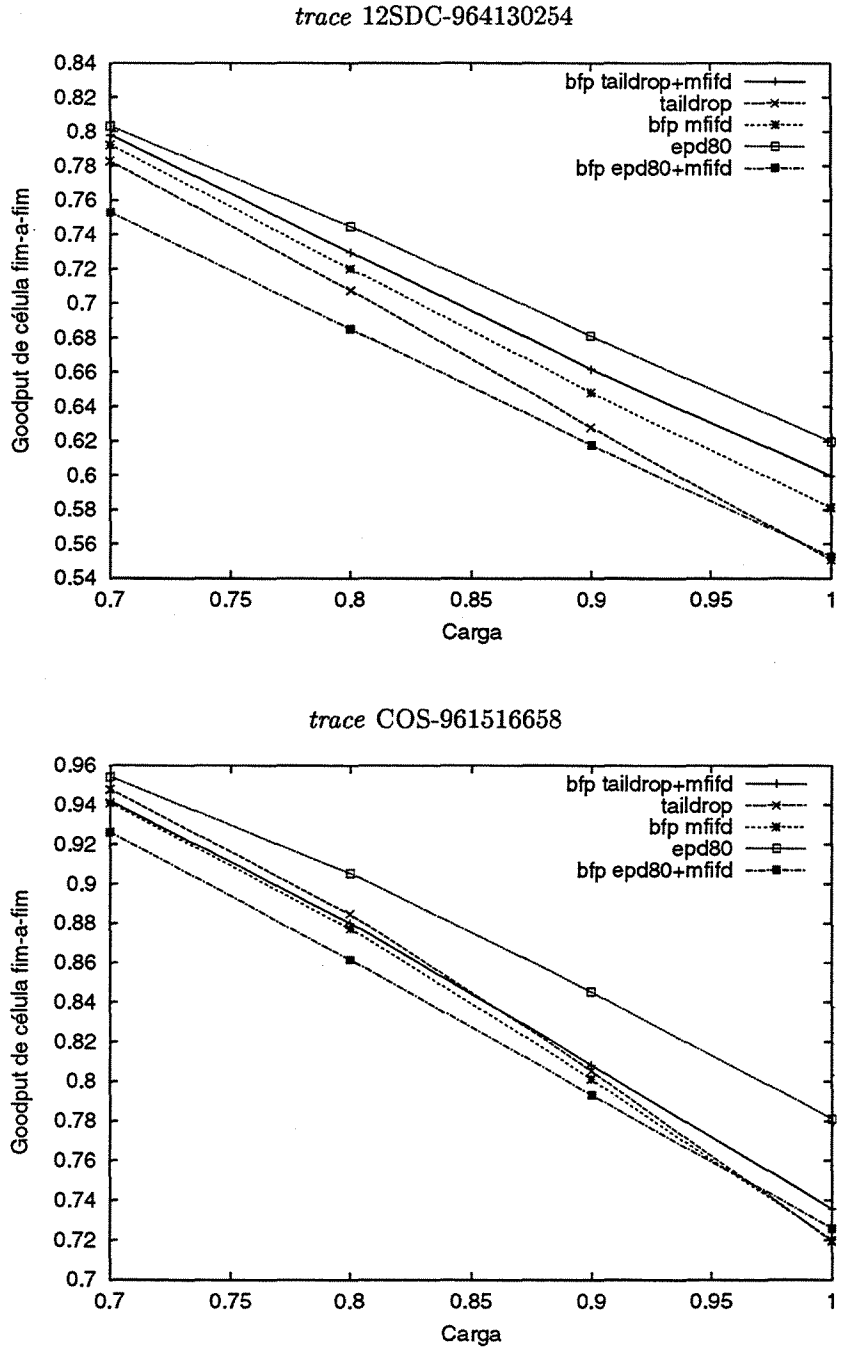


Figura 5.10: *Goodput* de célula fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

Nas Figuras 5.11 e 5.12, mostra-se o *goodput* de pacote como função da carga da rede num ambiente composto por um único nó e cinco nós, respectivamente. Da mesma forma

que acontece com o *goodput* de célula, também há melhora no desempenho de BFP quando quando este é integrado com a política de descarte de pacote *Taildrop*, ou seja, *Taildrop* atua de forma complementar a MFIFD, descartando a cauda de pacotes corrompidos por transbordo do *buffer* ou pelo mecanismo descarte de células MFIFD. Por exemplo, no caso da Figura 5.11, carga 1.0 e *trace* 12SDC-964130254, o *goodput* de pacote foi de 0.776 para *Taildrop*, 0.759 para BFP e 0.770 para BFP *Taildrop*+MFIFD. Com relação a integração entre EPD e MFIFD, os resultados não mostraram ganhos para o *goodput* de pacote, de modo que BFP EPD+MFIFD apresentou desempenho inferior a EPD e BFP atuando de forma isolada. Por exemplo, para carga 1.0, *trace* COS-961516658 e ambiente de rede com um único nó, o *goodput* de pacote foi de 0.850 para EPD80, 0.847 para BFP e 0.842 para BFP EPD80+MFIFD.

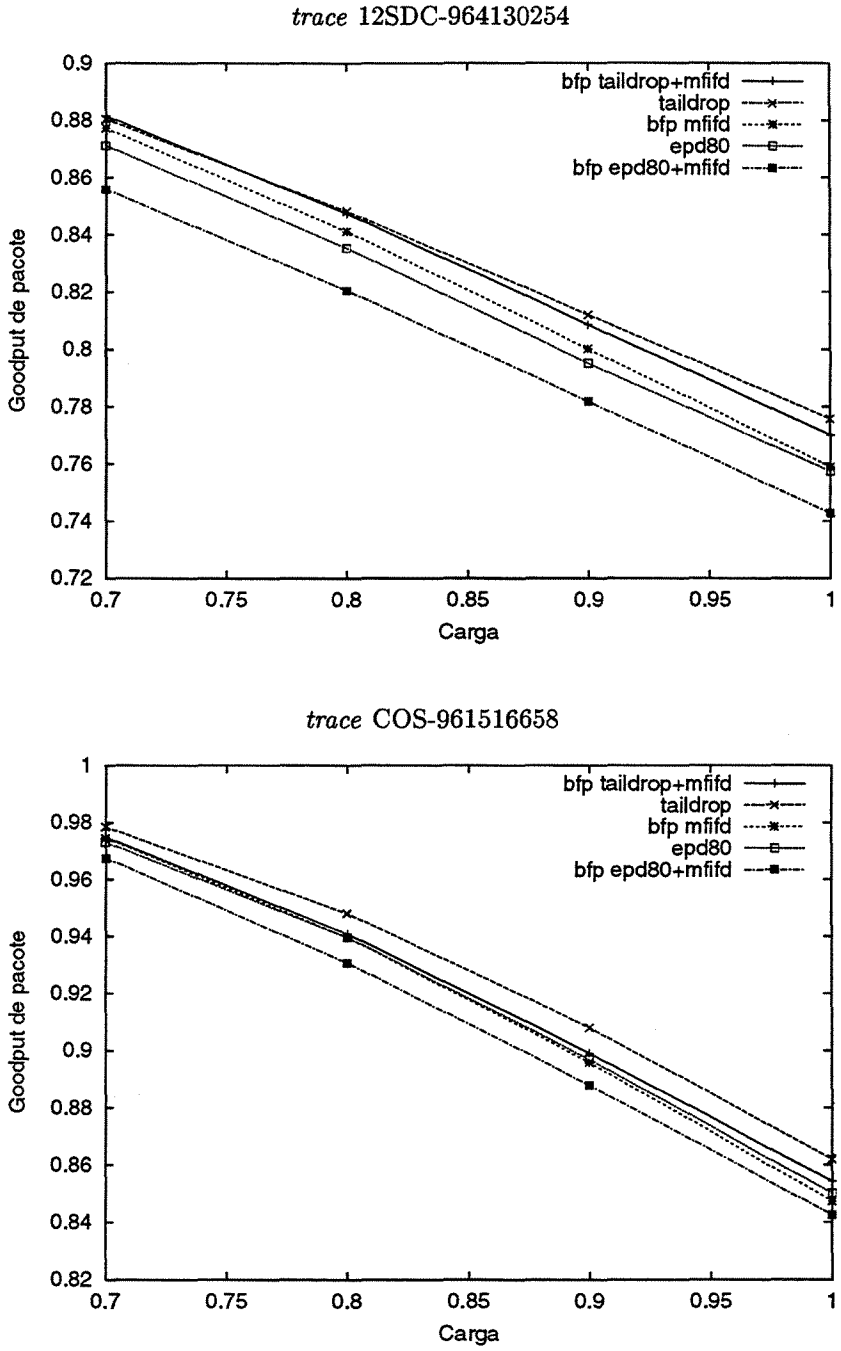


Figura 5.11: *Goodput* de pacote *vs.* Carga num ambiente com 1 nó e tráfego real.

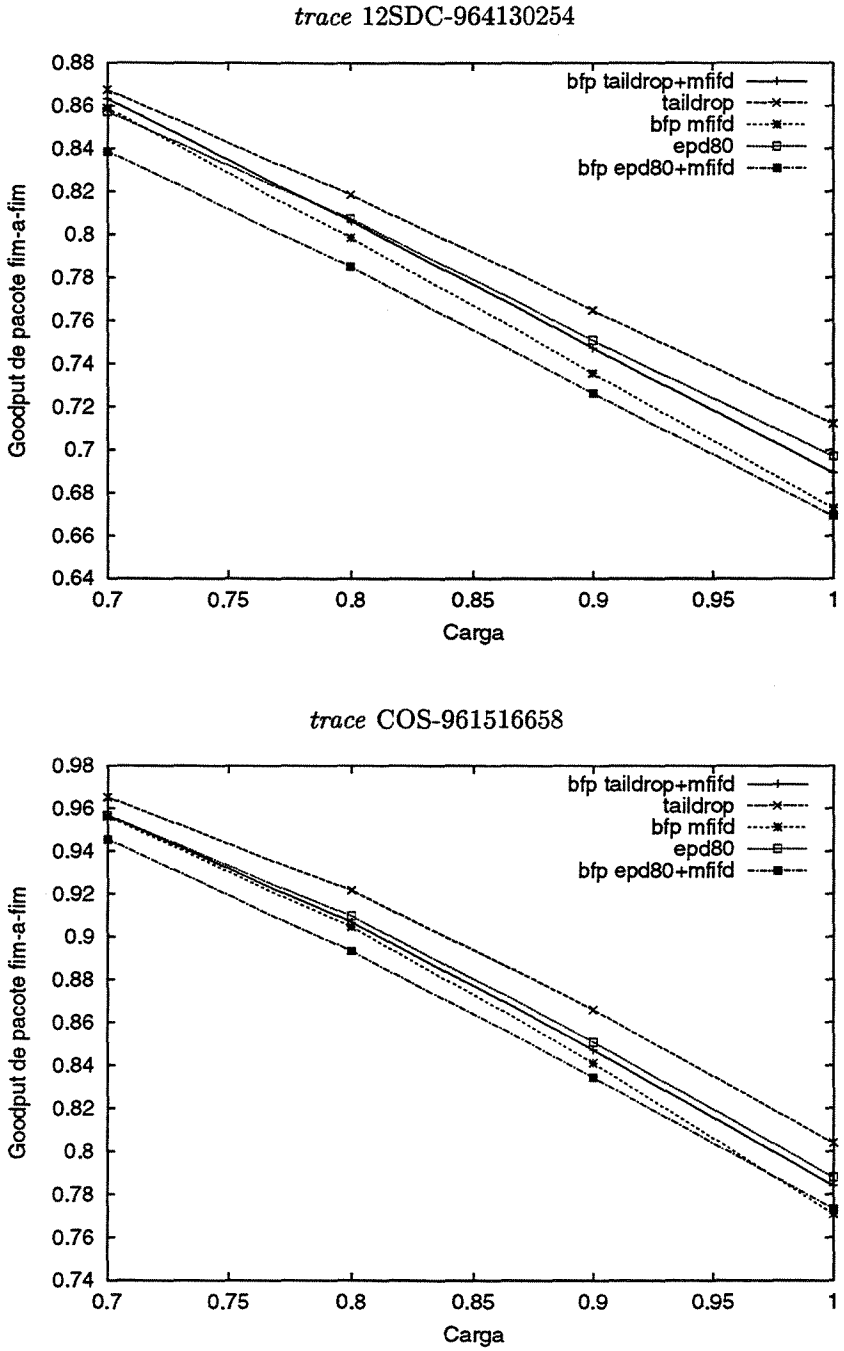


Figura 5.12: *Goodput* de pacote fim-a-fim *vs.* Carga num ambiente com 5 nós e tráfego real.

5.3 O Mecanismo Balde Furado para Pacote no Contexto da Classe de Serviço GFR

Espera-se que GFR (*Guaranteed Frame Rate*) torne-se uma importante categoria de serviço para suportar de forma eficiente tráfego TCP/IP em redes ATM. A garantia de serviço é baseado em pacotes AAL5 e, em situações de congestionamento, a rede tenta descartar pacotes AAL5 completos. O contrato de tráfego GFR é composto de quatro parâmetros principais: PCR (*Peak Cell Rate*), MCR (*Minimun Cell Rate*), MBS (*Maximum Burst Size*) e MFS (*Maximum Frame Size*). PCR é a taxa máxima que é permitida ao usuário transmitir. MFS é o tamanho máximo do pacote AAL5 que pode ser enviado pelo usuário. Especificando um MCR maior que zero, o usuário requisita a rede a garantia de uma largura de banda mínima. Expresso em células por segundo, MCR corresponde a largura de banda média que é reservada para o VC (*Virtual Channel*) dentro da rede durante o tempo da conexão. MCR é similar ao parâmetro SCR da categoria de serviço VBR, embora MCR forneça a largura de banda mínima para pacotes AAL5 inteiros, enquanto SCR fornece a largura de banda mínima para células individuais. MBS controla o limite superior a rajada máxima de tráfego permitida a qual MCR se aplica [The99].

A categoria de serviço GFR define uma particular utilização do bit CLP das células ATM. Dado que a unidade lógica de informação é pacote AAL5, GFR impõe que todas as células do pacote tenham o bit CLP marcado com mesmo valor. Pacotes AAL5 com bit CLP=1 são considerados com prioridade baixa e devem ser transmitidos pela rede com a filosofia do "melhor esforço". A garantida de largura de banda mínima é aplicada apenas aos pacotes marcados com bit CLP=0.

A principal diferença entre GCRA e F-GCRA é que F-GCRA declara pacotes completos com bit CLP=0 como *elegíveis* ou *não elegíveis* para a largura de banda mínima definida por MCR. Os pacotes AAL5 elegíveis são aqueles que devem ser entregues ao destino de modo a preencher a largura de banda mínima garantida pela rede ao transmissor. Pacotes com bit CLP=1 não são considerados como elegíveis para a largura de banda mínima. Enquanto F-GCRA é usado para especificar quais pacotes são elegíveis para largura de banda mínima, deve ser notado que GFR explicitamente permite ao transmissor enviar pacotes em excesso. F-GCRA é um algoritmo ideal que pode ser difícil de implementar na prática. Assim o ATM Forum definiu o simple-F-GCRA, uma versão simplificada que é equivalente a F-GCRA(T, L) para conexões onde os pacotes contem células marcadas com o mesmo bit CLP e passam pelo teste GCRA(PCR, τ_{pcr}). Veja o Algoritmo 5.3.

Algoritmo 4 SIMPLE-F-GCRA(T, L) - SF-GCRA

Parte 1: Chegada da primeira célula do quadro no tempo t_a:

X´ := X − (t_a − LPT)

se (X´ > L) ou (a célula tem CLP=1) então
    elegível := false /* célula não elegível */
senão
    elegível := true /* célula elegível */
fimse

se (elegível) então
    X´:= X − (t_a − LPT)
    X := max(0, X´) + T
    LPT := t_a
fimse

Parte 2: Chegada das células subseqüentes do quadro no tempo t_a:

se (elegível) então
    X´:= X − (t_a − LPT) /* célula elegível */
    X := max(0, X´) + T
    LPT := t_a
fimse

Com o intuito de facilitar a comparação com o algoritmo BFP (seção 4.1), o algoritmo 5.3 apresenta a versão do SF-GCRA implementado com balde e fichas. Note que o balde pode conter quantidade negativa de fichas ($B < 0$), indicando que células em excesso ao contrato de tráfego foram marcadas com prioridade alta. Esta violação é compensada com as geração de fichas no período em que não há chegada de células. No caso do algoritmo BFP, violação é compensada marcando células que estão em conformidade com o contrato com prioridade baixa. De fato, o algoritmo SF-GCRA gerou probabilidade de violação inferior a gerada pelo mecanismo BFP nos experimentos de simulação realizados.

Algoritmo 5 VERSÃO DO SF-GCRA IMPLEMENTADO COM FICHAS

Variáveis:

$TamanhoBalde$: Quantidade máxima de fichas permitidas no balde
B: Contador de fichas do balde
Tx: Taxa de geração de fichas por unidade de tempo
$PrioPacote$: Prioridade do Pacote
$PrioCelula$: Prioridade da Célula

/* Acumula fichas até o limite máximo do tamanho do balde */
Incremente B à taxa Tx;
$B := min\{TamanhoBalde, B\}$

/* Durante a chegada de células */
se "célula é a primeira do pacote" então
    se $(B > 0)$ então
        $PrioPacote :=$ "Alta"
    senão
        $PrioPacote :=$ "Baixa"
    fimse
fimse
$PrioCelula := PrioPacote$
se ($PrioCelula =$ "Alta")
    $B := B - 1$
fimse

5.3.1 SF-GCRA e comutadores ATM com descarte de células

Nesta seção compara-se o mecanismo SF-GCRA com os mecanismos BF e BFP. Analisa-se o compromisso entre maximizar o número de células transmitidas em pacotes com sucesso e maximizar o número de pacotes transmitidos com sucesso. Nos experimentos de simulação, considera-se o modelo descrito na seção 4.3. Mostra-se os resultados para o tráfego real e ambiente de rede composto por um único nó.

Na Figura 5.13, mostra-se o *goodput* de pacote e o *goodput* de célula como função da carga da rede utilizando tráfego real gerado a partir do *trace* 12SDC-964130254. Nota-se que BFP fornece *goodput* de pacote e de célula superior a SF-GCRA, que por sua vez é superior a BF. Além disso, a taxa de redução tanto para o *goodput* de célula quanto para o *goodput* de pacote foi ligeiramente mais acentuada para SF-GCRA do que para BFP. Por exemplo, com o aumento da carga de 0.7 para 1.0, a redução no *goodput* de pacote

foi de 99 para SF-GCRA e 990 para BFP. No caso do *goodput* de célula, a redução foi de 99 para SF-GCRA e 99 BFP. Com o aumento do tamanho do *buffer*, o desempenho oferecido por BFP e SF-GCRA tende a se aproximar, até um ponto em que não há mais diferença entre os mecanismos. Isso pode ser visto na Figura 5.14. Neste caso o tamanho do *buffer* utilizado é 50do que o tamanho de *buffer* utilizado na Figura 5.13.

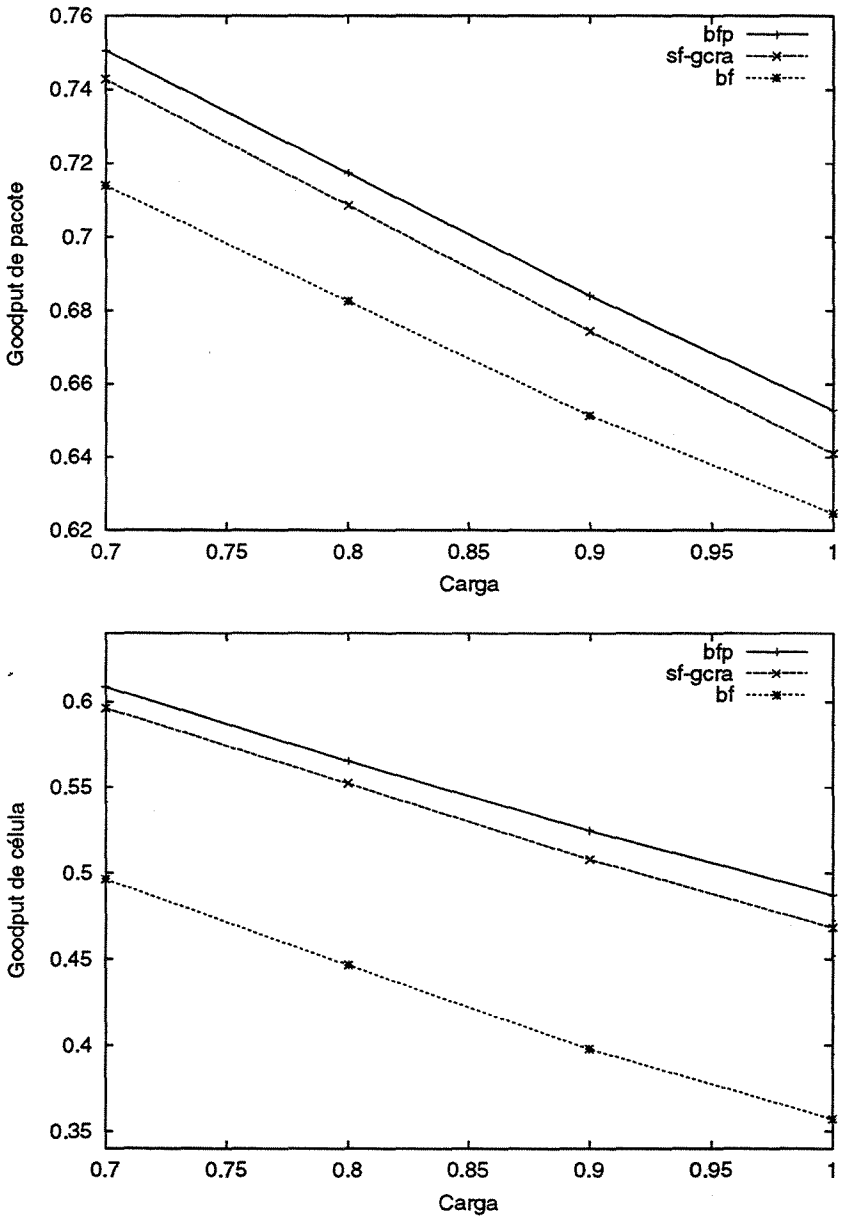


Figura 5.13: *Goodput* de pacote e célula *vs.* Carga num ambiente composto por 1 nó, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254.

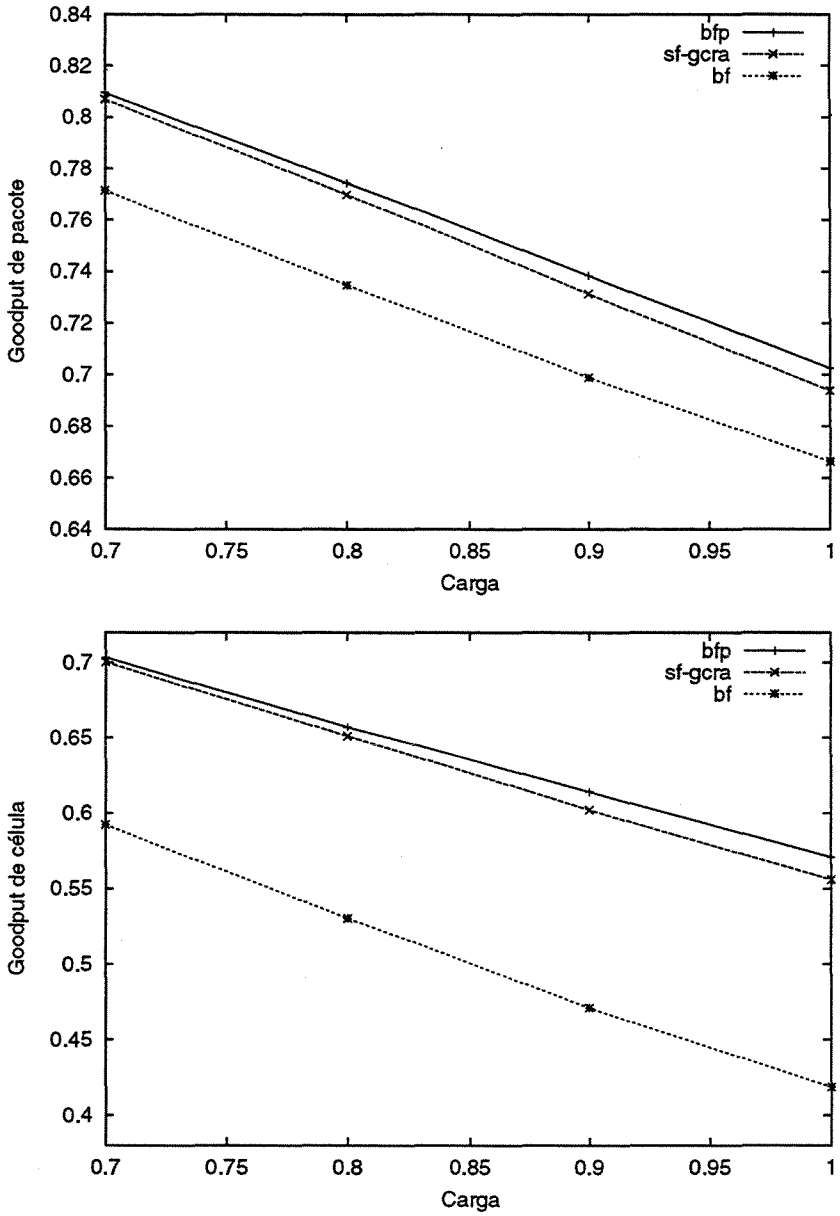


Figura 5.14: *Goodput* de pacote e célula *vs.* Carga num ambiente composto por 1 nó, *buffer* = 150 e tráfego real - *trace* 12SDC-964130254.

Na Figura 5.15, mostra-se o *goodput* de pacote e de célula como função da taxa de geração de fichas para uma carga fixa igual 0.9. Nota-se que mesmo com taxa de geração de fichas abaixo da taxa média da fonte (menor que 0.056) ou acima (maior que 0.056), o mecanismo BFP se mostrou mais estável em relação a SF-GCRA e BF. À medida que a taxa de geração de fichas aumenta, o desempenho dos mecanismos BFP, SF-GCRA

e BF sofre melhora acentuada até um ponto máximo. No caso do mecanismo BFP, o ponto máximo ocorreu quando a taxa de geração de fichas foi de 0.036, enquanto para SF-GCRA e BF o ponto máximo ocorreu para taxa de geração de fichas 0.028 e 0.020, respectivamente. Após o ponto máximo, BFP foi o mecanismo que teve a menor queda de desempenho com o aumento da taxa de geração de fichas. Por exemplo, a redução no *goodput* de pacote com o aumento da taxa de geração de fichas 0.036 para 0.084 foi de 0.006 para BFP, 0.021 para SF-GCRA e 0.014 para BF.

Na Figura 5.16, mostra-se o *goodput* de pacote e de célula como função do tamanho do balde para uma carga fixa igual 0.9. Tam como acontece com a taxa de geração de fichas, o mecanismos BFP também mostrou ser mais estável a variação do parâmetro tamanho do balde. Por exemplo, para tamanho do balde 8, o *goodput* de pacote oferecido foi de 0.684 para BFP e 0.677 para SF-GCRA. Aumentando-se o tamanho do balde para 16, o *goodput* de pacote de BFP continua em 0.684 e SF-GCRA passa para 0.674, ou seja, BFP permanece com desempenho superior.

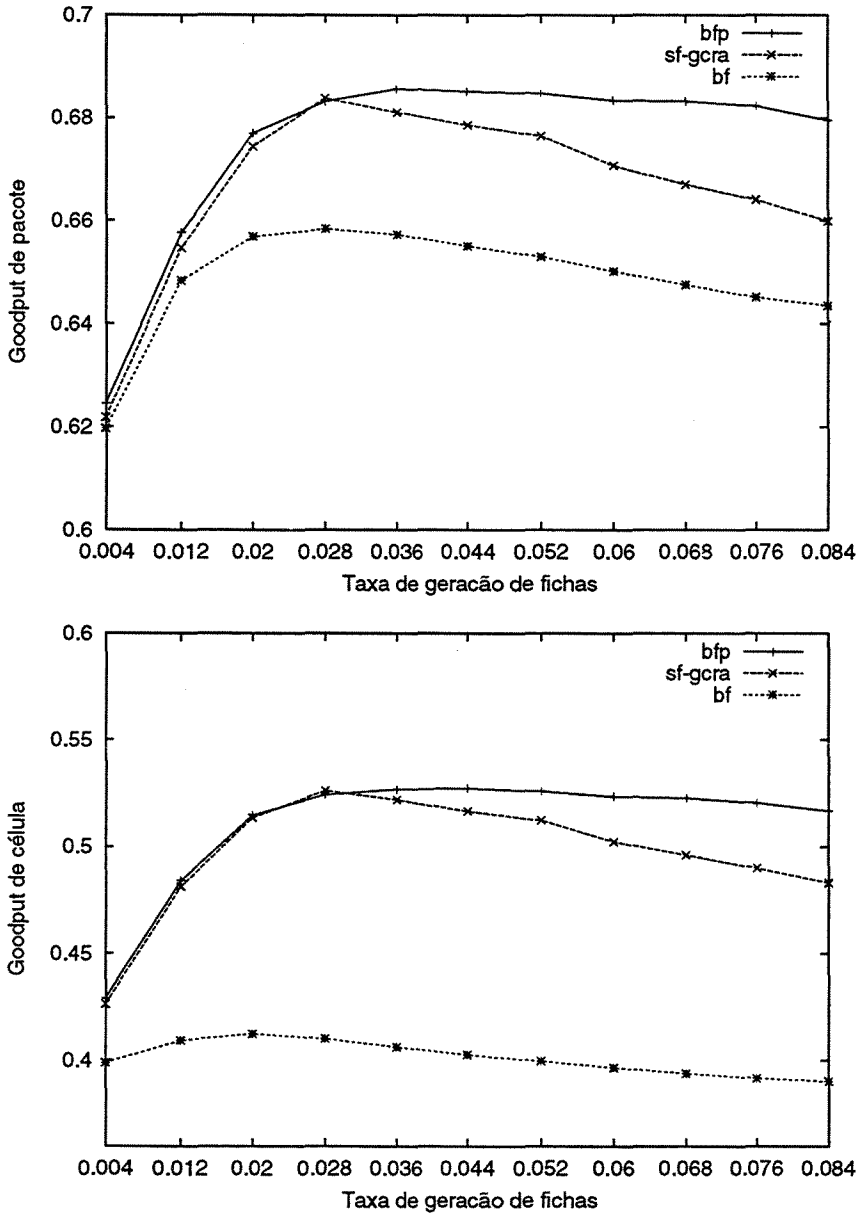


Figura 5.15: *Goodput* de pacote e célula *vs.* Taxa de geração de fichas num ambiente composto por 1 nó, carga 0.9, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254.

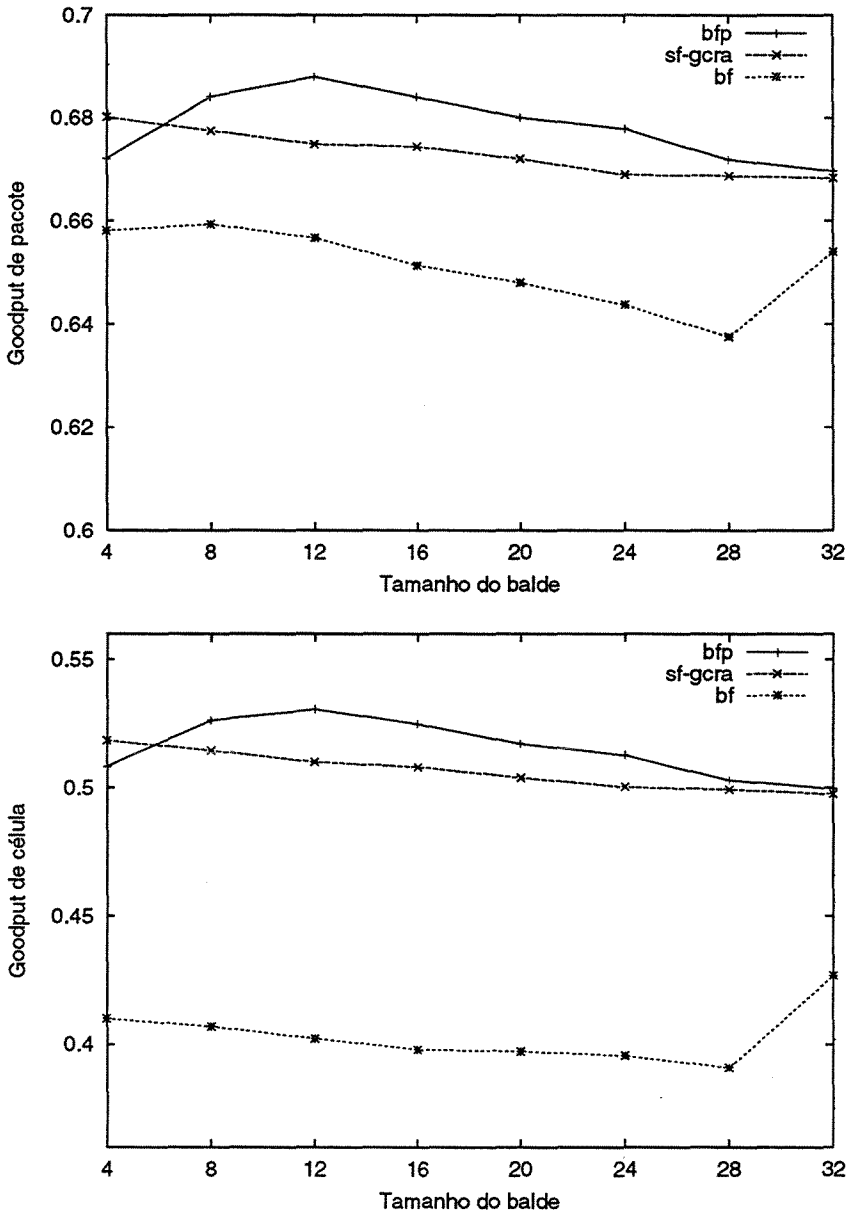


Figura 5.16: *Goodput* de pacote e célula *vs.* Tamanho do balde num ambiente composto por 1 nó, carga 0.9, *buffer* = 100 e tráfego real - *trace* 12SDC-964130254.

Capítulo 6

Conclusões

A tecnologia do Modo de Transferência Assíncrono (ATM - *Asynchronous Transfer Mode*) foi proposta em meados de 1980 como uma evolução das redes públicas para suportar serviços de banda larga. Desde a primeira recomendação da ITU-T sobre ATM no final da década de 80, muitos trabalhos foram realizados com intuito de fornecer melhor suporte para tráfego TCP/IP sobre redes ATM. Redes ATM são capazes de prover diferentes tipos de serviços, incluindo serviços do tipo "melhor esforço" comparáveis aos serviços fornecidos atualmente pela *Internet*. Além disso, uma motivação comum para utilização de tecnologias de rede de alta velocidade, tal como ATM, continua a ser a provisão de garantias de Qualidade de Serviço (QoS) para as aplicações que elas interconectam. Entretanto, IP foi independentemente desenvolvido sobre as bases de um modelo sem conexão, enquanto ATM foi originalmente projetada para serviços orientados a conexão, o que traz dificuldades para a integração destas duas tecnologias.

Esta dissertação propôs um mecanismo de policiamento de tráfego, Balde Furado para Pacote (BFP), que policia o fluxo de células levando em consideração o limite dos pacotes (AAL5 *frames*). Mostrou-se através de simulações que o mecanismo Balde Furado para Pacote, apesar de agregar células mal comportadas em um mesmo pacote, gera probabilidade de violação igual a do mecanismo Balde Furado (GCRA). Investigou-se a eficácia do mecanismo Balde Furado para Pacote e do mecanismo Balde Furado sob a influência de processo com dependência de longa-duração, utilizando tráfego sintético e real. Considerou-se no estudo a disciplina de descarte seletivo de célula MFIFD (*Modified First-In-First-Drop*) e a política de organização do *buffer* CCD (Compartilhamento Completo com Descarte). Analisou-se através de simulação o impacto das características do processo de entrada (tais como carga do sistema e parâmetro de Hurst) e número de nós da rede no *goodput* de célula e de pacote.

No estudo entre descarte seletivo de célula e de pacote, investigou-se a eficácia da disciplina MFIFD e mecanismos de policiamento BFP e BF em relação as políticas *Tail-*

drop e EPD. Também, introduziu-se duas políticas, *Taildrop*+MFIFD e EPD+MFIFD, que integram descarte seletivo de célula com descarte seletivo de pacote, e comparou-se essas políticas com as políticas *Taildrop* e EPD e com o mecanismo de descarte de célula MFIFD. Analisou-se o compromisso entre o *goodput* de célula e o *goodput* de pacote. Por fim, comparou-se o mecanismo BFP e BF com SF-GCRA, o algoritmo de policiamento definido pelo ATM Forum para a classe de serviço GFR.

6.1 Principais Contribuições

Pode-se resumir as principais contribuições da presente dissertação nos seguintes itens:

- Os resultados de simulação com *traces* mostraram que o tráfego policiado pelo mecanismo Balde Furado para Pacote produz maior *goodput* de pacote e de célula que o mecanismo Balde Furado convencional quando este tráfego passa por um nó comutador que utiliza CCD como política de organização do *buffer* e MFIFD como mecanismo de descarte seletivo de células. Além disso, o melhor desempenho do mecanismo Balde Furado para Pacote se mantém quando o tráfego policiado atravessa um ambiente de rede composto por vários nós comutadores;

- Comparando descarte de células diretamente com descarte de pacotes verificou-se que a política *Taildrop* fornece o maior *goodput* de pacote enquanto EPD fornece o maior *goodput* de célula. O mecanismo MFIFD com tráfego policiado por BFP fornece o segundo maior *goodput* de célula e o segundo maior *goodput* de pacote, portanto um desempenho mais balanceado entre a camada de transporte (pacotes IP) e *link* de rede (células ATM). No ambiente de rede formado por vários nós comutadores o desempenho dos mecanismos de descarte de células tem queda de desempenho mais acentuada com o aumento da carga da rede do que as políticas de descarte de pacotes (*Taildrop* e EPD), independente do mecanismo de policiamento utilizado (BF ou BFP);

- Foram propostos dois algoritmos para integração entre mecanismos de descarte de célula e políticas de descarte de pacote. Na integração entre EPD e MFIFD, EPD+MFIFD, verificou-se que não houve vantagens pois o *goodput* de célula e de pacote desta política foi inferior a todas outras políticas (EPD, *Taildrop* e o mecanismo MFIFD). Entretanto, a integração entre *Taildrop* e MFIFD, *Taildrop*+MFIFD, se mostrou atrativa pois, junto com a política *Taildrop*, gerou o maior *goodput* de pacote e junto com a política EPD gerou o maior *goodput* de célula.

- Comparando o mecanismo BFP e BF com o mecanismo de policiamento SF-GCRA da classe de serviço GFR, verificou-se que o tráfego policiado pelo mecanismo BFP

fornece maior *goodput* de célula e de pacote que os mecanismos BF e SF-GCRA. E ainda, que o mecanismo SF-GCRA desempenho superior a BF, principalmente em relação ao *goodput* de célula;

6.2 Trabalhos Futuros

A seguir descreve-se algumas sugestões para extensão deste trabalho:

- Um dos problemas com as políticas de descarte de pacotes *Taildrop* e EPD diz respeito à justiça dessas políticas com relação as conexões TCP. EPD e *Taildrop* não discriminam entre os pacotes pertencentes a múltiplas conexões TCP. Isto pode levar a situação onde uma conexão TCP finaliza a transmissão de um pacote e é capaz de aumentar o tamanho da sua janela de congestionamento, o que pode causar a célula de uma outra conexão TCP ser descartada devido a transbordo do *buffer*. A segunda conexão TCP, após *timeout* do pacote perdido, é então forçada a reduzir a sua janela de congestionamento. Dessa forma, a primeira conexão é capaz de obter injustamente uma porção maior de largura de banda. No caso do tráfego policiado pelo BFP e comutador com descarte de célula, espera-se que a maior parte das células descartadas sejam de baixa prioridade e portanto pertencentes a conexões que transmitiram além do que foi concedido no contrato de tráfego. Neste caso, e diferente do que acontece com as políticas EPD e *Taildrop*, são as conexões que consumiram largura de banda acima do permitido que serão forçadas a reduzir a sua janela de congestionamento. Pesquisas futuras devem investigar o efeito do descarte de células com respeito a justiça entre as conexões;

- As políticas EPD e *Taildrop* também sofrem do problema da sincronização global relacionado as conexões TCP. Todas vez que um pacote é descartado, TCP, como parte do seu mecanismo de controle de fluxo e congestionamento, assume que a rede está congestionada e inicia a sua fase de controle de congestionamento usando o mecanismo *slow start*. No princípio do congestionamento, se a política de descarte simultaneamente descarta um pacote de toda conexão, todas fontes TCP irão reduzir o tamanho de suas janelas ao mesmo tempo e a rede será subutilizada. No caso de BFP com MFIFD, espera-se que a penalização (descarte de células) das conexões que estão consumindo mais recursos que outras possa aliviar o problema da sincronização global uma vez que estas conexões serão forçadas a diminuir suas janelas de congestionamento primeiro. Portanto nem todas conexões TCP serão forçadas a diminuir suas janelas de congestionamento ao mesmo tempo. Pesquisas futuras devem investigar o efeito do descarte de células com respeito ao problema da sincronização global do TCP;

- Essa dissertação traz contribuições específicas para o entendimento dos mecanismos de policiamento BFP, BF e SF-GCRA, utilizando simulações com tráfego real. Pesquisas futuras devem investigar a derivação de resultados analíticos destes mecanismos;

- Estudar as características de outras políticas de descarte de pacotes (SD, ESPD, DfF, etc.) e propor novos algoritmos para integração destas com mecanismos de descarte de células (MFIFD, FIFD, LIFD, RAND) também é uma tarefa interessante a ser abordada.

Bibliografia

[AM95] A. Adas e A. Mukherjee. On resource management and QoS guarantees for long range dependent traffic. In *Proc. of IEEE INFOCOM'95*, pp. 779–787, 1995.

[BCT91] M. Butto, E. Cavallero, e A. Tonietti. Effectiveness of leaky bucket policing mechanism in ATM networks. *IEEE Journal on Selected Areas in Communications*, 9:335–342, abril de 1991.

[Ber92] J. Beran. Statistical methods for data with long-range dependence. *Statistical Science*, 7:404–427, 1992.

[BJ76] G. E. P. Box e G. M. Jenkins. Time Series Analysis: Forecasting and Control, 1976.

[BSTW95] J. Beran, R. Sherman, M. S. Taqqu, e W. Willinger. Long-Range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transaction on Communications*, 43(2/3/4):1566–1579, 1995.

[CFZ98] R. Choen, N. L. S. Fonseca, e M. Zukerman. *Multimedia communications networks: technologies and services*, capítulo Traffic management and networks dimensioning. Artech house, 1998.

[CGK95] I. L. G. Cidon, R. Guerin, e A. Khamisy. Optimal buffer sharing. *IEEE Journal on Selected Areas in Communications*, 7(13):1229–1240, setembro de 1995.

[CH96] A. K. Choudhury e E. L. Hahne. Dynamic queue lenght thresholds in a shared memory ATM switch. In *Proc. of INFOCOM´96*, pp. 679–687, 1996.

[Cox84] D. R. Cox. *Long-range dependence: A review.* Ames, IA: Iowa State Univ. Press., 1984.

[CP98] K. Cheon e S. Panwar. On the performance of ATM-UBR with early selective packet discard. In *Proc. of IEEE ICC*, pp. 221–227, 1998.

[CT98] H. Chiou e Z. Tsai. Performance of ATM switches with age priority packet discarding under on-off source model. In *Proc. of IEEE INFOCOM*, pp. 931–938, 1998.

[Dah89] R. Dahlhaus. Efficient parameter estimation for self-similar processes. *Annals of Statistics*, 17:1749–1766, 1989.

[FF98] N.L.S. Fonseca e M. J. Ferreira. On the effectiveness of the longest-packet-in packet discard policy. In *Proc. of GLOBECOM '98*, pp. 1747–1753, 1998.

[FR95] S. Floyd e A. Romanov. Dynamics of TCP traffic over ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(4):633–641, 1995.

[FT86] R. Fox e M. S. Taqqu. Large sample properties of parameter estimates for strongly stationary gaussian time series. *Ann. Statist.*, 14:517–532, 1986.

[GJK+97] R. Goyal, R. Jain, S. Kalyanaraman, S. Fahmy, e S. Kim. UBR+: Improving performance of TCP over ATM-UBR service. In *Proc. of IEEE ICC*, pp. 1042–1048, 1997.

[HDLK95] C. Huang, M. Devetsikiotis, I. Lambadaris, e A. Kaye. Modeling and simulation of self-similar variable bit rate compressed video: a unified approach. In *Proc.of ACM SIGCOMM'95*, pp. 114–125, 1995.

[Hei93] Heinanem, J. *Multiprotocol encapsulation over ATM Adaptation layer 5*. Technical report, RFC 1483, julho de 1993.

[HY96] M. G. Hluchyj e N. Yin. A second-order leaky bucket algorithm to garantee QoS in ATM networks. In *Proc.of IEEE GLOBECOM'96*, pp. 1090–1094, 1996.

[ITU95] ITU-T Study Group 13. Recommendation i.371 *traffic control and congestion control in b-isdn*. Technical report, ITU-T, 1995.

[KK80] F. Kamoun e L. Kleinrock. Analysis of shared finite storage in a computer network node enviroment under general traffic conditions. *IEEE Transaction on Communications*, 28:992–1003, 1980.

[Kol40] A. N. Kolmogorov. Wienersche spiralen and einige andere interessante kurven im hilbertschen raum. *C.R.(Doklady Acad. Sci. USSR (N.S.)*, 26:115–118, 1940.

[LB99] M. A. Labrador e S. Banarjee. Enhancing application throughput by selective packet dropping. In *Proc. of IEEE ICC*, pp. 1217–1222, 1999.

[LNO96] T. V. Lakshman, A. Neidhardt, e T. Ott. The drop from front strategy in TCP and in TCP over ATM. In *Proc. of IEEE INFOCOM*, pp. 1242–1250, 1996.

[LRS97] Y. Lapid, R. Rom, e M. Sidi. Analysis of packet discarding policies in high speed networks. In *Proc. of IEEE INFOCOM*, pp. 1193–1200, 1997.

[LTWW93] W. Leland, M. Taqqu, W. Willinger, e D. V. Wilson. On the self-similar nature of Ethernet traffic. In *Proc. of ACM SIGCOMM'93*, pp. 183–193, 1993.

[LZ92] P. Landsberg e C. Zukowski. A novel buffer sharing method: Complete sharing subject to guaranteed queue minimums. In *Proc. of The First International Conference of Computer Communications and Networks*, pp. 43–48, 1992.

[Man69] B. Mandelbrot. Long-run linearity, locally gaussian processes, h-espectra and infinite variances. *International Economic Review*, 10:82–113, 1969.

[Man82] B. B. Mandelbrot. The fractal geometry of nature. W. H. Freeman, New York, 1982.

[MN68] B. Mandelbrot e J. W. V. Ness. Fractional brownian motions, fractional noises and aplications. *SIAM Review*, 10:422–437, outubro de 1968.

[MPL95] L. G. Mason, A. Pelletier, e J. Lapointe. Towards optimal policing in ATM networks. In *Proc.of IEEE INFOCOM'95*, pp. 1238–1247, 1995.

[MS95] G. S. Mayor e J. A. Silvester. The multi-level leaky bucket mechanism. In *Proc.of* 4^{th} *IEEE International Conference on Computer Communications and Networks*, 1995.

[MTT97] A. Murad, M. S. Taqqu, e V. Teverovsky. Robustness of Whittle-type Estimators for Time Series with Long-Range Dependence. *to appear in Stochastic Models*, 1997.

[MW69] B. Mandelbrot e J. R. Wallis. Computer experiments with fractional gaussian noises. *Walter Resources Research*, 5:228–267, 1969.

[Nor95] I. Norros. On the Use of Fractional Brownian Motion in the Theory of Connectionless Networks. *IEEE Journal On Selected Areas in Communications*, 13(6):953–962, agosto de 1995.

[Pap78] A. Papoulis. *Probability, Random Variables and Stochastic Processes.* McGraw-Hill, 3ª edição, 1978.

[Rat91] E. P. Rathger. Modeling and perfomance comparison of policing mechanisms for ATM networks. *IEEE Journal on Selected Areas in Communications*, pp. 325–333, abril de 1991.

[Rob94] P. M. Robinson. Semiparametric anaylis of long-memory time series. *Ann. Statistic*, (22):515–539, 1994.

[SFMS96] J. A. Silvester, N.L.S. Fonseca, G. S. Mayor, e S. Sobral. The effectiveness of multi-level policing mechanism in ATM traffic control. In *Proc.of IEEE International Telecommunications Symposium*, pp. 98–102, 1996.

[ST94] G. Samarodnitsky e M. S. Taqqu. *Stable Non-Gaussian Processes: Stochastic Models with Infinite Variance.* Chapman and Hall, 1994.

[TG97] B. Tsybakov e N. D. Georgana. On Self-Similar Traffic in ATM Queues: Definitions, Overflow Probability Bound, and Cell Delay Distribution. *IEEE/ACM Transactions on Networking*, 5(3):397–409, junho de 1997.

[The94] The ATM Forum Technical Committee, User Network Interface (UNI) Working Group. *ATM User Network Interface Specification, UNI 3.1.* Technical report, ATM Forum, 1994.

[The99] The ATM Forum Technical Committee, Traffic Management Working Group. *Traffic Management Specification, Version 4.1.* Technical report, ATM Forum, março de 1999.

[THP94] L. Tassiulas, Y. C. Hung, e S. S. Panwar. Optimal buffer control during congestion in a ATM network node. *IEEE/ACM Transactions on Networking*, 4(2):374–386, agosto de 1994.

[TL86] M. S. Taqqu e J. B. Levy. *Using renewal processes to generate long-range dependece and high variability*, volume 11. in Dependence in Probability and Statistics, E. Eberlein and M. S. Taqqu, eds. Boston, MA:Birkhauser, 1986.

[Tur86] J. S. Turner. New directions in communications. *IEEE Magazine*, 24:8–15, 1986.

UNICAMP
BIBLIOTECA CENTRAL
SEÇÃO CIRCULANTE

[Tur96] J. S. Turner. Maintaining high throughput during overload in ATM switches. In *Proc. of IEEE INFOCOM*, pp. 287–295, 1996.

[Vec94] G. Veciana. Leaky buckets and self-tuning rate control. In *Proc.of IEEE GLOBECOM'94*, pp. 1207–1211, 1994.

[Wil97] Willinger, W. and Taqqu, M. S. and Sherman, R. and Wilson, D. Self-similarity through high variability: statistical analysis of ethernet lan traffic at source level. *IEEE/ACM Trans. Networking*, 5:71–86, 1997.

[WTE96] W. Willinger, M. S. Taqqu, e A. Erramilli. *A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks.* Stochastic Networks: Theory and Applictions, Clarendon Press, Oxford, U.K, 1996.

[WTS97] W. Willinger, M. S. Taqqu, e R. Sherman. Proof of a fundamental result in self-simlar traffic modeling. *Computer Commun. Rev.*, 27:5–23, 1997.

[WTSW97] W. Willinger, M. S. Taqqu, R. Sherman, e D. V. Wilson. Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Transactions on Networking*, 5(1):71–86, fevereiro de 1997.

[Yaj85] Y. Yajima. On estimation of long-memory time series models. *Australian Journal of Statistics*, 27:303–320, 1985.

[YSS95] N. Yamanaka, Y. Sato, e K.I. Sato. Permormance limitation of leaky bucket algorithm for ATM networks. *IEEE Transaction on Communications in Communications*, 43:2298–2230, 1995.