UNIVERSIDADE TECNOLÓGIA FEDERAL DO PARANÁ
CAMPUS CURITIBA GERÊNCIA DE PESQUISA E PÓS-GRADUAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E INFORMÁTICA INDUSTRIAL — CPGEI

LINCOLN HERBERT TEIXEIRA

ANÁLISE DE DESEMPENHO DO PROTOCOLO TCP TS-PRIO UTILIZANDO MODELOS MARKOVIANOS

DISSERTAÇÃO DE MESTRADO

CURITIBA
MAIO DE 2009

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial

---

DISSERTAÇÃO

apresentada à UTFPR

para obtenção do grau de

MESTRE EM CIÊNCIAS

por

LINCOLN HERBERT TEIXEIRA

ANÁLISE DE DESEMPENHO DO PROTOCOLO TCP TS-PRIO UTILIZANDO MODELOS MARKOVIANOS

Banca Examinadora:

Presidente e Orientador:

Prof. Dr. EMILIO C. GOMES WILLE — UTFPR

Co-orientador:

Prof. Dr. WALTER GODOY JÚNIOR — UTFPR

Examinadores:

Prof. Dra. KEIKO VERÔNICA ONO FONSECA — UTFPR

Prof. Dr. LUIZ CARLOS PESSOA ALBINI — UFPR

Curitiba, 11 de maio de 2009.

LINCOLN HERBERT TEIXEIRA

ANÁLISE DE DESEMPENHO DO PROTOCOLO TCP TS-PRIO UTILIZANDO MODELOS MARKOVIANOS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial da Universidade Tecnológica Federal do Paraná como requisito parcial para a obtenção de grau de "Mestre em Ciências" - Área de Concentração: Telemática.

Orientador: Prof. Dr. EMILIO C. GOMES WILLE

Curitiba

11.05.2009

Agradecimentos

Primeiramente, agradeço a Deus por me proteger e me iluminar nas horas difíceis. Peço a Ele que me permita usar os conhecimentos adquiridos para ajudar outras pessoas com meu trabalho.

Agradeço a todas as pessoas que, direta ou indiretamente, tornaram este trabalho possível de ser realizado, em especial ao meu orientador Professor Doutor Emilio Carlos Gomes Wille, que me despertou curiosidade e interesse em uma área antes desconhecida.

Aos meus pais, Luis e Syonara, e à minha irmã, Thalita, agradeço todo o amor, carinho, compreensão e apoio na fase mais difícil da minha vida, onde precisei abandonar o conforto da minha família, para correr atrás do meu sonho.

Às agências financiadoras CNPq e CAPES, pelo auxílio material.

Ao NATEC, e todas as pessoas envolvidas, principalmente meu co-orientador Professor Doutor Walter Godoy Júnior, por ter sido o laboratório que me acolheu e deu a oportunidade de crescer nos estudos e trabalho.

Em especial, agradeço ao Kleber Kendy Horikawa Nabas, que foi o primeiro a confiar no meu potencial e me deu a oportunidade de aprender a arte de educar. O tempo transformou colegas de trabalho em uma excepcional amizade.

E por último , mas não o menos importante, meus amigos, que por força maior da minha dedicação a este trabalho, tive que abdicar; mas tenho a certeza de que sempre entenderam enquanto estive ausente.

“A verdadeira felicidade custa pouco; sendo cara, é porque sua qualidade não presta.”

François René De Chateaubriand

Resumo

Este trabalho apresenta a implementação de um modelo do TCP-Reno, bem como uma técnica de modelamento de rede, detalhadamente apresentada, com o objetivo de mensurar a rajada e a vazão das conexões TCP compartilhadas em um roteador congestionado. Isso se faz necessário, para permitir estudar e prever o comportamento de uma rede. A validação do modelo implementado foi realizada por meio de comparações com resultados de simulações baseadas no software *Network Simulator* (NS-2).

Posteriormente a apresentação do modelo, e como objetivo principal deste trabalho, foi feita uma análise detalhada de uma modificação no protocolo que implementa QoS (*Quality of Service*) na camada de transporte. Essa modificação no protocolo, denominada de TCP TS-Prio, implementa uma diferenciação no tamanho da janela deslizante, através de atribuição de prioridade para determinados fluxos. Essa análise revela as qualidades e limitações da utilização desse protocolo em cenários diferentes do original do TCP TS-Prio, onde o autor focava principalmente no comportamento dinâmico do protocolo.

Como resultado do trabalho chega-se a conclusão de que o TCP TS-Prio é capaz de oferecer diferenciação de serviço, porém isto depende diretamente do estado de congestionamento da rede em questão.

Palavras-chave: Modelo TCP, Reno, TS-Prio, QoS, FPA, prioridade, congestionamento.

Abstract

This work presents the implementation of a model for the TCP-Reno protocol, as well as a detailed technique for modeling the network, with the purpose of measuring the burstiness and the throughput of TCP traffic in a congested shared router. This is necessary to study and to predict the behavior of a data network. The model was validated by comparing its results to simulations carried out with specialized simulation software, namely the Network Simulator (NS-2).

After presenting the model and the main goals of this work, a detailed analysis of a modified protocol that implements QoS (Quality of Service) in the transport layer was done. This protocol called TCP TS-Prio, implements a differentiation in the size of the sliding window, thus giving priorities to certain flows. This analysis is needed to reveal qualities and limitations of this protocol in several network environments (scenarios), because in the original work on TCP TS-Prio the author focused mainly on the dynamic behavior of the Protocol.

We then conclude that TCP TS-Prio is able to offer service differentiation, but this depends directly on the network congestion degree.

Key Words: TCP model, Reno, TS-Prio, QoS, FPA, priority, congestion.

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ACK - *Reconhecimento Positivo (Acknowledgment)*

AF - *Expedição Assegurada (Assured Forwarding)*

CTMC - *Cadeia de Markov em Tempo Contínuo (Continuous Time Markov Chain)*

DiffServ - *Serviços Diferenciados (Differentiated Services)*

DSCP - *DiffServ Code Point*

FIFO - *Primeiro a Entrar, Primeiro a Sair (First In First Out)*

FPA - *Aproximação do Ponto Fixo (Fixed Point Approximations)*

FTP - *Protocolo de Transferência de Arquivos (File Transfer Protocol)*

HTTP - *Protocolo de Transferência de Hipertexto (Hypertext Transfer Protocol)*

IETF - *Internet Engineering Task Force*

IntServ - *Serviços Integrados (Integrated Services)*

IP - *Protocolo de Internet (Internet Protocol)*

ISP - *Provedor de Acesso a Internet (Internet Service Provider)*

NS - *Simulador de Rede (Network Simulator)*

RSPV - *Resource Reservation Protocol (Protocolo de Reserva de Recursos)*

RTO - *Tempo de Retransmissão (Retransmission timeout)*

RTT - *Round-trip time*

QoS - *Qualidade de Serviço (Quality of Service)*

SBE - *Melhor Esforço Estático (Static Best Effort)*

TBE - *Melhor Esforço Estático (Transient Best Effort)*

TCP - *Protocolo de Controle de Transmissão (Transmission Control Protocol)*

ToS - *Tipo de Serviço (Type of Service)*

TS-Prio - *Prioridade Transitória e Estática (Transient & Static Priority)*

UDP - *User Datagram Protocol*

WAN - *Rede de Longa Distância (Wide Area Network)*

WEB - *Rede de Alcance Mundial (World Wide Web)*

Sumário

1 INTRODUÇÃO

1.1 Contexto

O TCP (*Transmission Control Protocol*) é o protocolo de transporte mais utilizado na Internet. A maioria das aplicações como HTTP, FTP e redes ponto a ponto utilizam o TCP para prover conexões fim-a-fim confiáveis. O TCP funciona ajustando a janela de congestionamento que é o que controla a quantidade de dados que serão inseridos na rede. Na maioria das implementações TCP esse ajuste é controlado pela perda de pacotes que indica congestionamento na rede. O tamanho da janela de congestionamento e a vazão de dados da rede (*throughput*) formam um mecanismo de realimentação que tenta estabilizar a rede e prover justiça, para que todos os usuários da rede sejam atendidos.

Em uma rede de elevada razão largura de banda/atraso, as conexões TCP transmitem os pacotes de dados em rajadas e, quando uma rajada de pacotes do tamanho da janela de congestionamento é transmitida, mais nenhum pacote pode ser transmitido até que os pacotes enviados tenham sido confirmados. Este comportamento contribui para a chegada em rajada nos roteadores, porém, isso pode causar estouros desnecessários de buffer e aumentar o atraso do enlace, causando problemas de desempenho na rede. Entretanto, é importante conseguir medir a quantidade de pacotes inseridos na rede para comparar os diferentes protocolos baseados em rajadas.

1.2 Caracterização do Problema e Objetivos deste Trabalho

Mecanismos de provisão de QoS (Quality of Service) se fazem necessários para satisfazer as necessidades do usuário da rede, que cada vez mais utilizam serviços como voz e vídeo na rede de dados. Para adicionar QoS na rede existem duas arquiteturas definidas pelo IETF (*Internet Engineering Task Force*): IntServ (*Integrated Service*) e DiffServ (*Differentiated Services*) [Kilkki, 1999].

O IntServ mantém a qualidade fim-a-fim para fluxos individuais com o auxílio do protocolo de reserva de recursos RSPV (*Resource Reservation Protocol*), o qual exige muita sinalização e não possui uma escalabilidade para grandes redes.

Na arquitetura DiffServ, os pacotes que entram na rede são agregados em diferentes classes. Cada classe tem uma marca a ela associada, utilizando os bits DSCP (*DiffServ Code Point*). Esta classificação não exige muita sinalização, tornando o DiffServ extremamente escalável e ainda assegura aos fluxos agregados uma QoS quantitativa fim-a-fim. Esta arquitetura se fundamenta no princípio de se concentrar na borda da rede a complexidade das operações envolvidas no tratamento do tráfego, de modo que o núcleo da rede permaneça o mais simples possível.

Em [Tostes, 2004] foi proposto o modelo TCP TS-Prio (*Transmission Control Protocol - Transient & Static Priority*) que implementa um método simples de diferenciação de serviços baseado na configuração de controle de congestionamento do TCP do servidor com uma marcação de prioridade e que mantém compatibilidade com as atuais implementações TCP. Este protocolo introduz DiffServ na camada de transporte, por meio de alterações nos parâmetros dos algoritmos utilizados na implementação TCP-Reno. O objetivo dessa modificação é oferecer diferenciação no atendimento de solicitações de serviços de transporte, ainda no nó gerador dos pacotes TCP durante a geração dos fluxos de aplicações. Este protocolo busca uma redução dinâmica no envio de pacotes do fluxo não prioritário, denominado TBE (*Transient Best Effort*), para propiciar maior vazão do fluxo prioritário, denominado SBE (*Static Best Effort*).

No trabalho original [Tostes, 2004] os autores analisaram principalmente o comportamento dinâmico do protocolo proposto. O objetivo desta dissertação é aquele de obter e analisar os principais indicadores de desempenho associados ao TCP TS-Prio.

Para alcançar este objetivo o modelo analítico do protocolo TCP desenvolvido em [Dimopoulos et al., 2005] foi implementado em linguagem *C/C++* e convenientemente modificado. Tal modelo é capaz de estimar o tamanho da rajada e a vazão de um sistema com conexões TCP. O modelo de rede utilizará as medições da rajada para calcular o atraso de fila e a probabilidade de perda de pacotes. Essa implementação se faz necessária para servir de base para analisar diferentes arquiteturas de redes com mecanismos de QoS.

Esse trabalho considera um sistema formado por múltiplas fontes TCP heterogêneas conectadas a um roteador de rede congestionado e o objetivo é obter o desempenho das conexões TCP. Os indicadores de interesse são os valores de RTT (*Round Trip Time*), probabilidade de perda de pacote na rede e a vazão da transferência de arquivos. Foi empregado o Método do Ponto Fixo (*Fixed Point Approximations* - FPA), similar a proposta de [Avrachenkov et al., 2002] e [Bu e Towsley, 2001], para a solução do sistema de equações envolvido.

As principais contribuições desse trabalho são:

- Utilizar técnicas de modelamento já existentes para mensurar o tamanho das rajadas das conexões TCP.
- Reproduzir o modelo apresentado em [Dimopoulos et al., 2005] que é utilizado para analisar o desempenho de protocolos baseados em janela que transmitem informações através de um roteador congestionado.
- Fornecer uma visão da influência do tráfego em rajadas no desempenho do sistema.
- Comparar os resultados obtidos com aqueles presentes em [Dimopoulos et al., 2005] onde utilizou-se o simulador NS-2 (*Network Simulator*) [NS-2].
- Adaptar o modelo para analisar o protocolo proposto por [Tostes, 2004] denominado de TCP TS-Prio que visa oferecer diferenciação de serviços.

1.3 Estrutura do Trabalho

Além dessa sessão introdutória, esta dissertação está organizada da seguinte forma: O capítulo 2 apresenta uma descrição detalhada do conceito do TCP, diferenciando dois tipos de implementação existentes. Esta descrição é fundamental para o entendimento do modelo implementado. O capítulo 3 descreve o modelo do sistema, e apresenta o Método do Ponto Fixo que é a técnica utilizada na solução do modelo. Este capítulo também descreve o modelo em Cadeia de Markov a Tempo Contínuo (CMTC) do protocolo TCP-Reno, bem como, o modelo de rede que é baseado em uma fila $M_X/M/1/K$. No capítulo 4 é feita a apresentação da proposta da dissertação, a apresentação do modelo TCP TS-Prio, desenvolvido por [Tostes, 2004], e a descrição da adaptação do modelo analítico necessária para analisar o modelo TCP TS-Prio. O capítulo 5 apresenta os cenários de rede utilizados, a validação do modelo e, também, a análise dos resultados com suas respectivas representações gráficas. O capítulo 6 mostra as conclusões, e indica o desenvolvimento de possíveis trabalhos futuros.

2 PROTOCOLO TCP

Na Internet os transmissores e os receptores, que são os usuários finais, necessitam de uma maneira inteligente para escolher as taxas de envio de dados. Os usuários finais abordam o problema utilizando o Protocolo de Controle de Transmissão (*TCP*), cuja função é ajustar a taxa de envio de dados em resposta as condições da rede, e retransmitir informações que possivelmente foram perdidas. Como a principal inteligência da Internet está nos usuários finais, a tarefa da arquitetura intermediária da Internet é relativamente simples, pacotes que chegam no nó de rede são roteados e seguem em direção ao destino.

Não existe nenhuma entidade central a qual uma nova conexão poderia contactar para encontrar a capacidade disponível da Internet, muito menos uma conexão pode ter uma garantia de que usaria uma parcela da capacidade total do enlace. Portanto, a cada nova conexão o próprio transmissor deve estimar a capacidade do enlace. Estimar a capacidade do enlace disponível é uma tarefa do TCP, e existem diferentes variantes do protocolo TCP que utilizam diferentes métodos para efetuar essa tarefa. A primeira versão do algoritmo de controle de congestionamento do TCP foi implementado no final da década de 1980, devido a colapsos de congestionamento [Jacobson, 1988]. Antes do TCP, uma conexão iniciava e injetava pacotes a uma alta taxa, sem se preocupar com o congestionamento da rede. A estratégia utilizada pelo TCP é enviar pacotes através da rede sem nenhuma forma de prevenção, somente depois, reagir com o que acontecer com os pacotes enviados. Se pacotes são corretamente recebidos, a taxa de envio aumenta; se pacotes são perdidos, a taxa de envio diminui [Stevens, 1994].

Toda fonte TCP estima a capacidade disponível da rede para que possa decidir quantos pacotes podem ser transmitidos. Os pacotes são enviados do transmissor ao receptor, e o receptor confirma a recepção desses pacotes, retornando um *ACK* (pequeno pacote de confirmação) para o transmissor. A fonte TCP mantém um contador, o tamanho da *janela de congestionamento*, que determina o número de pacotes que a fonte está apta a transmitir antes de parar a transmissão para esperar os pacotes de confirmação. O comportamento do TCP na WAN (*Wide Area Network*) tem um ciclo de processo: primeiramente a fonte TCP envia a quantidade de pacotes do atual tamanho da janela de congestionamento em direção ao receptor, esses pacotes trafegam

através de enlaces com diferentes atrasos de propagação e esperam em filas para finalmente alcançar o receptor. Em seguida, o receptor retorna um pacote de confirmação ACK, que pode trafegar através caminhos diferentes, e enfrentam seu próprios atrasos de propagações e de filas, para retornar ao transmissor. Assim, depois de um *RTT* (*round trip time*) (estimativa do tempo total de transmissão, ida e volta, de um único pacote), os ACKs que retornam irão confirmar que os pacotes foram entregues com sucesso. A taxa de envio então aumentará e o processo se repete. Em contrapartida, a falta do ACK depois de um RTT indica que a rede está com problemas. Perda de pacote ou ACKs são tidos como um indicador de rede congestionada e o controle de fluxo da fonte TCP responde ao congestionamento diminuindo a taxa de envio de pacotes. Desta forma, o sistema se auto-regula e novos pacotes somente serão enviados na rede se existirem respostas de confirmação ACKs. A forma particular com que as fontes reagem a eventos de perda dá origem a diferentes tipos de implementação do protocolo TCP. As versões mais utilizados são conhecidos com os nomes TCP-Tahoe [Jacobson, 1988], TCP-Reno [Allman et al., 1999], TCP-NewReno [Floyd, 1999], TCP-SACK [Floyd, 1997] e TCP-Vegas [Brakmo, 1994].

Um dos principais objetivos desse trabalho é realizar o modelamento do TCP, por isso apresentam-se detalhes sobre o TCP-Tahoe, para entender o funcionamento do controle da janela de congestionamento, a fase de *slow-start* e fase *timeout*. Como ele é a mais antiga versão do protocolo, seu algoritmo de recuperação de perdas é muito simples e ineficaz, portanto, apresenta-se também o TCP-Reno, que melhorou o algoritmo de recuperação de perda, e é capaz de manter uma alta taxa de transmissão de dados na presença de perdas ocasionais de pacotes. Para uma introdução mais detalhada sobre o assunto abordado é recomendada a leitura de [Kurose e Ross, 2006] e [Peterson e Davie, 2000].

2.1 TCP-Tahoe

O TCP-Tahoe é a mais antiga versão do protocolo TCP [Jacobson, 1988]. Durante a transferência de arquivos, o TCP-Tahoe move-se entre as chamadas fases *slow-start*, *congestion avoidance* e *timeout loss recovery*. A evolução da taxa de transferência, controlada pelo tamanho da janela de congestionamento, é mostrada na Figura 2.1.

Uma transferência de arquivo inicia-se na fase *slow-start* com o tamanho da janela de congestionamento *W*, que é inicializada em um pacote. Esse pacote, com número de sequência 1, é enviado pelo enlace em direção ao destinatário. Ao receber esse pacote de dados, o receptor acusa o recebimento, retornando um pequeno pacote denominado ACK. O próximo pacote

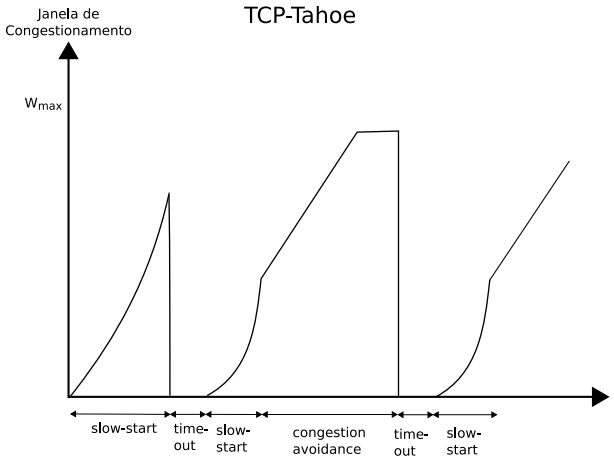

Figura 2.1: TCP-Tahoe: evolução da janela de congestionamento

esperado pelo receptor possui o número de sequência 2. Com o retorno do ACK também é incrementado o tamanho da janela de congestionamento do transmissor, de um para dois pacotes. Então, dois novos pacotes são enviados pelo enlace, e se nenhuma perda de pacote ocorre, dois ACKs serão retornados pelo receptor, e cada um incrementa a janela de congestionamento de um pacote, resultando em um tamanho de janela de congestionamento de quatro pacotes depois desses dois ciclos. Durante a fase de *slow start* o tamanho da janela de congestionamento é dobrado uma vez a cada RTT. Esse aumento de tamanho da janela continua até que o tamanho máximo da janela de congestionamento seja alcançado, ou até ocorrer uma perda de pacote. Se o tamanho máximo da janela é alcançado, ciclos livres de perdas consecutivos não aumentarão o tamanho da janela de congestionamento.

Se ocorrer períodos de congestionamento na rede e houver descarte de pacotes de um transmissor TCP-Tahoe, o transmissor descobrirá isso implicitamente pela falta de ACKs dos pacotes perdidos. Quando um pacote é enviado, um temporizador de *timeout* é iniciado com um valor estimado pela fonte dependente da média e a variância do RTT. Se cessa o retorno dos ACKs, devido a natureza *self-clocking* do TCP a fonte não estará apta a enviar nenhum pacote novo pelo enlace. Então, a transmissão de pacotes fica temporariamente paralisada.

Se o *timeout* expira antes que um ACK desse pacote tenha retornado, o TCP interpreta isso como uma perda. Depois do período de *timeout*, o TCP-Tahoe irá reiniciar o tamanho da janela de congestionamento para um pacote e ajustar o *slow-start threshold* para metade do valor da janela de congestionamento no instante que a perda de pacote foi descoberta. Então, a fase de *slow-start* será reinicializada onde o tamanho de janela de congestionamento é dobrada a cada RTT. Para essa segunda fase de *slow-start* a duplicação do tamanho da janela somente ocorrerá até a próxima de perda de pacote ou, até que a janela de congestionamento tenha alcançado o *slow-start threshold*. Uma vez que o valor de threshold foi alcançado, a fase agressiva do *slow-*

start termina e o incremento da janela de congestionamento continua de forma mais cautelosa, entrando no modo *congestion avoidance* como mostrado na Figura 2.1. Durante o *congestion avoidance*, todo retorno de ACKs incrementa o tamanho da janela de congestionamento com uma pequena proporção do tamanho de janela corrente. Desde que W pacotes sejam enviados a cada RTT, durante períodos livres de perdas, resultará em W ACKs retornados, então ocorrerá um incremento no tamanho da janela de congestionamento de um pacote por RTT. O incremento linear do tamanho da janela de congestionamento, durante a fase de *congestion avoidance*, continua até que o tamanho máximo da janela de congestionamento seja alcançado, ou até que outra perda de pacote seja identificada. Após esta perda de pacote se sucedem as fases de *timeout*, *slow-start* e *congestion avoidance* [Stevens, 1994].

O protocolo TCP-Tahoe move-se consecutivamente entre as fases *slow-start*, *congestion avoidance* e *timeout* até que todos os pacotes correspondentes desse fluxo tenham alcançado o receptor e tenham sido confirmados pelo transmissor. Além disso, note que o receptor TCP pode implementar *delayed acknowledgements*. Ao invés de enviar um ACK por cada pacote recebido, o receptor somente confirma (ACK) no segundo pacote. Desde que o ACK incrementa a janela de congestionamento do transmissor, a utilização do atraso dos ACKs impactará na vazão do TCP. Entretanto, é importante notar que o atraso dos ACKs não são permitidos quando ocorrem chegadas fora de sequência. Se um pacote que chega no receptor tem número de sequência diferente do esperado, isso imediatamente gera um retorno de ACK.

A resposta do *timeout*/*slow-start* aos eventos de congestionamento obtida pelo TCP-Tahoe é restritiva e conservativa. Tal ação drástica de decremento do tamanho da janela de congestionamento para um pacote em cada evento de congestionamento pode ter um impacto negativo significante na vazão (*throughput*) do protocolo.

2.2 TCP-Reno

O TCP-Reno retém a dinâmica do TCP-Tahoe em termos de operação das fases de *congestion avoidance* e de *slow-start*, bem como o significado da janela de congestionamento, o *slow start threshold* e o tamanho máximo da janela de congestionamento [Allman et al., 1999]. A reação para eventos de perda de pacotes entretanto foi modificada a fim de manter uma alta taxa de envio, em um meio de rede congestionado. A implementação menos refinada do *timeout* do TCP-Tahoe leva à longos períodos de inatividade, enquanto se espera o tempo de *timeout* expirar. Durante esse período de espera, o envio de pacotes é descontinuado resultando em uma baixa vazão. No TCP-Reno, a duração da fase de recuperação de perda foi melhorada com a

introdução do algoritmo *fast retransmit*, como observado na Figura 2.2.

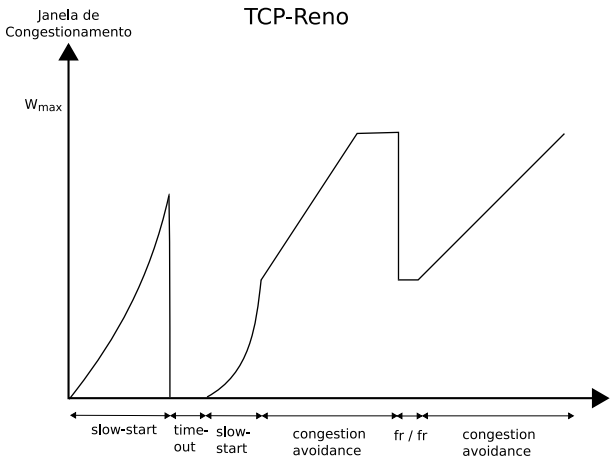

Figura 2.2: TCP-Reno evolução da janela de congestionamento [Stevens, 1994]

O *fast retransmit* é um mecanismo que as vezes resulta em uma rápida retransmissão de pacotes perdidos, onde antes somente era possível se o *timeout* expirasse. A idéia do algoritmo de *fast retransmit* é intuitiva e fácil de entender. A cada chegada de um pacote, o receptor responde retornando um pacote de confirmação. Assume-se que o pacote de número $n-1$ foi corretamente recebido, mas o pacote n foi perdido antes de alcançar o receptor. Assume-se também que os pacotes de número $n+1$, $n+2$ e $n+3$ são corretamente recebidos. Pela implementação do procedimento ACK, o receptor confirmará a recepção dos pacotes de número $n+1$, $n+2$ e $n+3$ retornando um ACK para o transmissor declarando que o próximo pacote esperado é com a sequência numérica n, o pacote perdido. Uma vez que o ACK do pacote $n-1$ afirma que o receptor esta esperando o pacote número n, em adição ao primeiro ACK, o transmissor irá receber três das chamadas confirmações duplicadas (*dup ACK*) para o pacote n.

Quando o transmissor recebe o primeiro *dup ACK*, ele conclui que o receptor recebeu pacotes em ordem errada e que isto pode significar que o pacote n foi perdido. Isso, entretanto, pode também significar que o pacote n esteja atrasado, porquê ele pode ter ido por um caminho diferente da rede. O transmissor então age de uma forma conservadora e espera até que seja recebido três *dup ACKs* antes de concluir que o pacote n tenha sido perdido e precise ser retransmitido. Esse mecanismo é chamado de *detecção de tripla duplicação*.

Se a janela de congestionamento é grande o suficiente e a taxa de perda de pacotes é pequena, então no mínimo três *dup ACKs* são retornados ao transmissor e a fase de recuperação de perda pode ser inicializada logo depois de um RTT, em vez de esperar pela expiração do *timeout*. O retorno dos ACKs provê ao transmissor a informação de que a rede não está congestionada, uma vez que uma larga proporção de pacotes enviados foram recebidos e confirmados com sucesso. Ajustando o tamanho da janela de congestionamento para um pacote, como feito pelo

TCP-Tahoe, e ativando o *slow-start* pode ser muito conservador. Isso leva à uma implementação do *fast retransmit* em combinação com o algoritmo *fast recovery*. O *fast recovery* substitui o *slow-start* depois que um evento de perda de pacote é descoberto pelos três *dup ACKs*. O efeito do *fast retransmit*/*fast recovery* (*fr/fr* na figura 2.2) é que, se uma perda de pacote é descoberta via triplo ACK duplicados, o primeiro pacote perdido será rapidamente reenviado e o tamanho da janela de congestionamento diminuirá à metade. Se o tamanho da janela resultante permitir isso, segue um aumento linear diretamente. Esse resultado é mais agressivo e mais eficaz para utilização da capacidade da rede, resultando em alta vazão para o transmissor TCP-Reno quando poucos pacotes são perdidos em cada evento de congestionamento.

O mecanismo de *fast retransmit*/*fast recovery* implementado no TCP-Reno é capaz de aumentar significativamente a vazão das fontes TCP que ocasionalmente sofrem perdas de pacotes. O objetivo buscado do *fast retransmit*/*fast recovery* é de reduzir pela metade a janela de congestionamento uma única vez a cada evento de congestionamento. Isto acontece se um único pacote é perdido, então o *fast retransmit*/*fast recovery* re-envia o primeiro pacote perdido e avança com o *congestion avoidance*. Se múltiplos pacotes são perdidos de uma janela simples, o algoritmo de *fast retransmit*/*fast recovery* do TCP-Reno pode entretanto induzir múltiplas invocações consecutivas, cada invocação reduz pela metade o tamanho da janela de congestionamento. Um ACK para um pacote previamente enviado e perdido pode ser chamado de ACK *parcial*. Na implementação do *fast retransmit*/*fast recovery* no TCP-Reno, a chegada de ACKs parciais iniciará um novo *fast retransmit*/*fast recovery* seguido pela redução pela metade da janela. Essas consecutivas reduções irão reduzir a janela de congestionamento tanto que o TCP-Reno por fim não estará apto a enviar qualquer novo pacote devido a restrição do tamanho de janela sobre o número de pacotes sem confirmação permitidos. Então, múltiplas perdas de pacotes podem finalmente fazer com que o transmissor tenha que esperar o *timeout* expirar mesmo quando os pacotes reenviados estejam sendo corretamente recebidos e confirmados.

2.3 TCP com Diferenciação de Serviços

2.3.1 DiffServ

Com a crescente necessidade de diferenciação de serviços nas redes, foi criado um Grupo de Trabalho de Serviços Diferenciados (*DiffServ*) do IETF. A idéia que prevalecia no grupo era de que para se prover a verdadeira QoS a uma rede, deveriam haver fortes garantias da mesma, sem as quais não haveria nenhuma QoS [Kilkki, 1999]. O grupo de Diferenciação de Serviços passou a dar importância a dois tópicos: a necessidade de níveis de precedência de descarte e

classes para os pacotes, de acordo com o atraso tolerável. Melhores características no que diz respeito aos atrasos dos pacotes serão requeridas futuramente nas redes IP, pelas aplicações de tempo real, e o DiffServ pode prover estas características.

Tais características de QoS variam de acordo com a necessidade dos fluxos de pacotes de cada usuário. É feito então um acordo de nível de serviço (*Service Level Agreement - SLA*), onde o usuário e o ISP (*Internet Service Provider*) fazem um acordo a respeito da QoS a ser provida aos fluxos. Com base no SLA, a Diferenciação de Serviços permite que o tráfego IP seja classificado em um número finito de classes de serviços que recebem diferentes tratamentos nos roteadores. Por exemplo, tráfegos pertencentes a uma classe de serviço de alta prioridade recebem uma forma de tratamento preferencial sobre tráfegos classificados em uma classe de serviços de baixa prioridade.

A proposta de Serviços Diferenciados não está apta a fornecer garantias explícitas a cada nó; porém, em elementos congestionados de uma rede, tráfegos com prioridade mais alta têm maiores probabilidades de serem entregues. Na ocorrência de um congestionamento, caso haja necessidade de descarte de pacotes, os tráfegos que requerem baixa prioridade para descarte são os últimos a serem descartados, ou seja, são os primeiros a serem enviados [Andrikopoulos e Pavlou, 1999].

Alguns conceitos e terminologias devem ser enfatizadas para que seja caracterizado os Serviços Diferenciados, entre eles o PHB (*Per Hop Behavior*) que significa uma combinação de itens no que diz respeito ao comportamento dos pacotes em cada salto: encaminhamento, classificação e descarte, ou seja, um determinado PHB reserva uma determinada percentagem da capacidade de um enlace da rede para aquele agregado de fluxos [Kilkki, 1999].

O PHB deve estar disponível em todos os roteadores, sendo ele, a única parte do DiffServ que é implementado nos roteadores internos. Os nós de borda incluem os mecanismos PHB e ainda sofisticados mecanismos condicionadores de tráfego, requeridos para fornecer o serviço desejado.

O encaminhamento do tipo AF (*Assured Forwarding*) oferece diferentes níveis de encaminhamento para pacotes IP, com o objetivo de garantir a entrega dos dados no tempo previsto [Heinanen et al., 1999]. Para que isso seja possível, classes IP são alocadas em cada um dos nós, reservando uma determinada quantidade de recursos de envio, por exemplo espaço de *buffers* e largura de banda. Os pacotes IP que desejarem usar os serviços providos pelos PHBs AF são marcados pelo roteador de borda.

2.3.2 TCP TS-Prio

O trabalho de [Tostes, 2004] propõe um método de diferenciação de serviços baseado na configuração da janela deslizante do TCP do *host* servidor e marcação simples de prioridade. Em outras palavras, busca-se otimizar a rede diferenciando os fluxos de dados ainda no servidor, independente da ocorrência de congestionamento. Esse método impõe alterações mínimas no código TCP, mantendo a compatibilidade com as implementações TCP dos nós clientes e considera ainda a existência de tráfego que apresenta fluxos de longa duração (*long-lived*) [Brownlee e Claffy, 2002]. Portanto, assume-se para a aplicação deste método [Tostes e Fonseca, 2005] que:

- existe uma separação do tráfego UDP e das aplicações TCP em classes *DiffServ* distintas;
- há uma subdivisão da classe AF (*Assured Forwarding*) em pelo menos duas prioridades *k* de serviços;
- o servidor TCP reconheça essas prioridades;
- existe uma identificação de Clientes com aplicações e perfis diferenciados através de contratos de nível de serviço (SLA) estabelecidos;
- a política de gerência de filas seja do tipo RED (*Random Early Detection*) [Braden et al., 1998] ou similar para evitar descarte de pacotes de um mesmo fluxo ao se detectar um congestionamento.

A abordagem proposta por [Tostes, 2004] trata de um servidor de FTP para fluxos de longa duração, agregados dentro de uma determinada classe AF. Os fluxos são agregados em dois tipos de serviço dentro da classe AF para atendimento das solicitações FTP. O primeiro tipo é denominado Melhor Esforço Transitório (*Transient Best Effort - TBE*) e agrega fluxos não prioritários. O segundo tipo é denominado Melhor Esforço Estático (*Static Best Effort - SBE*) e agrega os fluxos prioritários. Sendo assim os fluxos TBE possuem prioridade dinâmica, dependendo da existência de fluxos SBE. Os fluxos TBE são gerados da forma tradicional do TCP Reno, quando não existir fluxos SBE.

O TCP TS-Prio provê a diferenciação de serviços fim-a-fim por meio de redução dinâmica no envio de pacotes dos fluxos TBE para propiciar maior vazão aos fluxos SBE. O atendimento deve ser diferenciado para cada cliente conforme sua prioridade. Esta diferenciação se dá pela redução linear da *allowed window* (*allowed_wnd*), que é a janela de envio de dados cujo o valor é utilizado para transmissão dos próximos pacotes, dos fluxos com prioridade transitória TBE

durante os intervalos em que concorram com fluxos de prioridade estática SBE. Para realizar a diferenciação o resultado da *allowed_wnd*, após calculado conforme o TCP Reno original é alterado para:

$$allowed\_wnd = allowed\_wnd * \tau \tag{2.1}$$

Onde τ=1 para fluxos SBE em qualquer situação e para fluxos TBE quando não concorrentes com SBE, e 0<τ<1 para fluxos TBE concorrentes com SBE. A figura 2.3(a) mostra um fluxo TBE com *allowed_wnd*=10 enviando 10 pacotes sem concorrência de fluxos SBE. A figura 2.3(b) mostra que ao iniciar uma conexão SBE com *allowed_wnd*=10, o fluxo TBE passa a condição de não prioritário com τ=0,5, causando redução do tamanho da sua janela *allowed_wnd* para 5 pacotes. Essa sistemática permite que fluxos SBE aumentem sua taxa de envio mesmo sem alterar os valores de seus parâmetros de janela deslizante. Ao diminuir o tamanho da *allowed_wnd* para fluxos TBE obtém-se melhora na vazão dos fluxos SBE, fato confirmado com as simulações [Tostes, 2004].

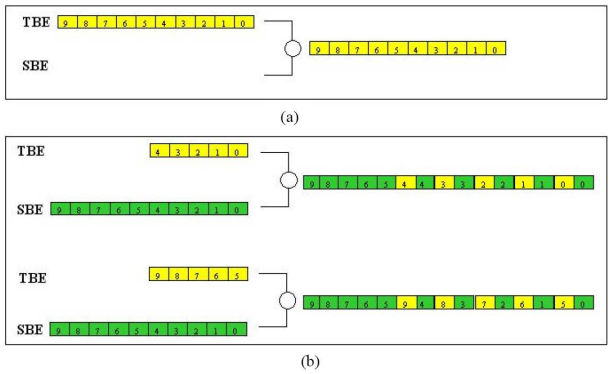

Figura 2.3: Comportamento das Janelas de Fluxos TBE e SBE

Quando fluxos não prioritários TBE ocorrem sem concorrência de fluxos prioritários SBE, a limitação de sua janela *allowed_wnd* é desnecessária. A figura 2.4 mostra o início e o término de transmissão de fluxos em instantes diferentes: TBE em ($t0$, tn) e SBE em ($t1$, $tn-1$). Na ausência do fluxo SBE, ($t0$, $t1$) e ($tn-1$, tn), pode-se alterar dinamicamente o parâmetro τ da *allowed_wnd* para melhorar o desempenho do fluxo TBE.

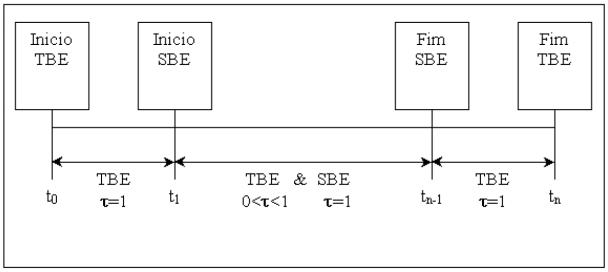

Figura 2.4: TS-Prio - Prioridade Transitória e Prioridade Estática

3 METODOLOGIA DE ANÁLISE

3.1 Introdução

Neste capítulo descreve-se o sistema proposto em [Dimopoulos et al., 2005] para análise do desempenho de um sistema com transmissões via protocolo TCP. Este sistema utiliza múltiplas fontes TCP heterogêneas cada uma com diferentes RTTs conectadas a uma rede com um roteador congestionado, como mostra a figura 3.1. O sistema utiliza a versão TCP-Reno, e é similar aos sistemas presentes em [Jacobson, 1988] e [Bu e Towsley, 2001] que modelam uma rede e fontes TCP separadamente.

O modelo de fonte TCP calcula a taxa λ_R com que a rajada de pacotes parte da fonte dado um atraso de fila T_Q e uma probabilidade de perda de pacote P. Diferentes tipos de fontes TCP podem ser modeladas e então agregadas, para encontrar a carga total λ_α oferecida à rede (figura 3.2). Em adição pode ser obtida a distribuição do tamanho das rajadas B geradas por todas as fontes. A rede é modelada como um simples roteador congestionado. Dada a taxa oferecida e a distribuição de tamanho da rajada, pode ser calculado o atraso de pacotes e a probabilidade de perda de pacote utilizando teoria de filas [Kleinrock, 1976]. Cada modelo é resolvido numericamente e o ponto onde os modelos interceptam indica o ponto de equilíbrio do sistema. Este ponto de equilíbrio é encontrado utilizando o método do Ponto Fixo [Martinez e Santos, 1995].

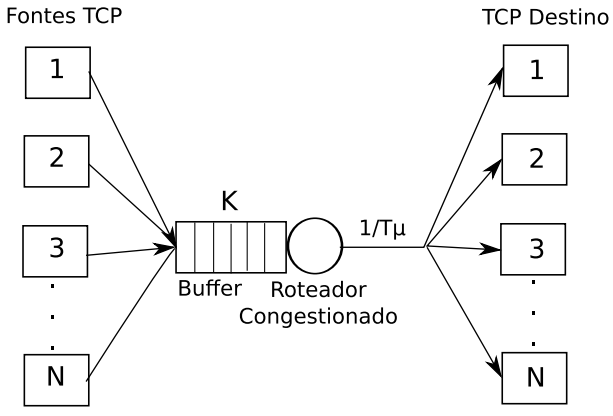

Figura 3.1: Topologia do Sistema

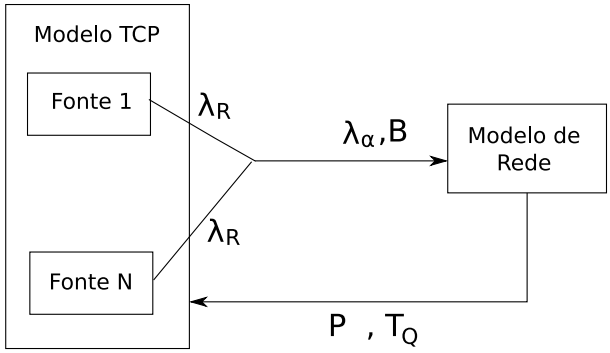

Figura 3.2: Configuração do Sistema

3.1.1 Método do Ponto Fixo

Seja T_{rtt}^{i} o valor médio do RTT da i-ésima conexão TCP. A carga oferecida à rede (vazão) é definida como:

$$\lambda_\alpha = \sum_{i=1}^{N} f_i(P, T_{rtt}^{i}) \tag{3.1}$$

onde $fi(P, T_{rtt}^{i}) = \lambda_R$ é a vazão da i-ésima fonte (obtida a partir do modelo TCP).

Sejam $p(C, \lambda_\alpha)$ e $q(C, \lambda_\alpha)$ a probabilidade de perda e atraso médio de pacote (obtidos pelo modelo de rede), respectivamente, e C é a capacidade do enlace, assim:

$$P = p(C, \lambda_\alpha) \tag{3.2}$$

$$T_Q = q(C, \lambda_\alpha) \tag{3.3}$$

O valor do RTT é dado por:

$$T_{rtt}^{i} = 2T_\tau^{i} + q(C, \lambda_\alpha) \tag{3.4}$$

onde T_τ^{i} é o atraso de propagação da i-ésima conexão.

Substituindo as equações (3.2) e (3.4) em (3.1), obtêm-se a seguinte equação não linear para λ_α:

$$\lambda_\alpha = \sum_{i=1}^{N} f_i(p(C, \lambda_\alpha), 2T_\tau^{i} + q(C, \lambda_\alpha)) \tag{3.5}$$

Cuja resolução consiste em atribuir um valor inicial para λ_α e iterar até a convergência; esta

estratégia é conhecida como método do Ponto Fixo.

3.1.2 Modelo de Fontes TCP

Neste item descreve-se o modelo de Fonte TCP proposto em [Dimopoulos et al., 2005]. De acordo com [Dimopoulos et al., 2005] o objetivo do modelo TCP é estimar o tamanho da rajada agregada e a vazão de todas as fontes TCP. Foi usada a distribuição de janela como uma estimativa para a distribuição em rajadas e assumido que a fonte TCP com um tamanho da janela W cria uma rajada de W pacotes, por exemplo, se o tamanho da janela é dez, então é assumido que uma rajada de dez pacotes sai da fonte com uma duração que é muito menor que o RTT. Esta definição de rajada somente é válida quando o RTT e a largura de banda são elevados. Quando o RTT é elevado, períodos de inatividade ocorrem entre as transmissões em rajadas. O funcionamento do TCP é modelado utilizando um processo de Markov $\{U_k\}$, onde k é o período onde ocorrem as mudanças de estados, que podem ser Ativo (N), Inativo (I), Timeout (T) e Fast Retransmit (F), como mostrado na figura 3.3 (exemplo com tamanho de janela oito).

Uma descrição das taxas de transição presentes no diagrama é mostrada na tabela 3.1. Os estados ativos representam a existência de dados TCP disponíveis para transmitir e são representados pelo vetor (W,W_t,N), onde W é o tamanho de janela atual, W_t é a janela de *threshold* e W_M é o tamanho máximo da janela de recepção. O estado ativo modela a forma como a janela se modifica em *slow start* e *congestion avoidance*. Quando ocorre um evento de perda, cada estado ativo pode transitar para um dos dois estados: *timeout* (W,T) ou *fast retransmit* (W,F). Uma transição para o estado inativo modela o tempo onde uma conexão não tem dados para enviar. O espaço de estados S consiste no conjunto de estados definido pela equação 3.6, onde $\lceil x \rceil$ significa o menor inteiro maior ou igual a x.

$$\begin{aligned}
(W,W_t,N) &\in \{(1,1,N)...(W_M,\lceil W_M/2\rceil,N)\} \\
&\quad W \in \{1,2,...W_M\} \\
&\quad W_t \in \{1,2,...\lceil W_M/2\rceil\} \\
(W,T) &\in \{(1,T),(2,T),...(W_M,T)\} \\
(W,F) &\in \{(4,F),(5,F),...(W_M,F)\} \\
I &\in \{I\}
\end{aligned} \tag{3.6}$$

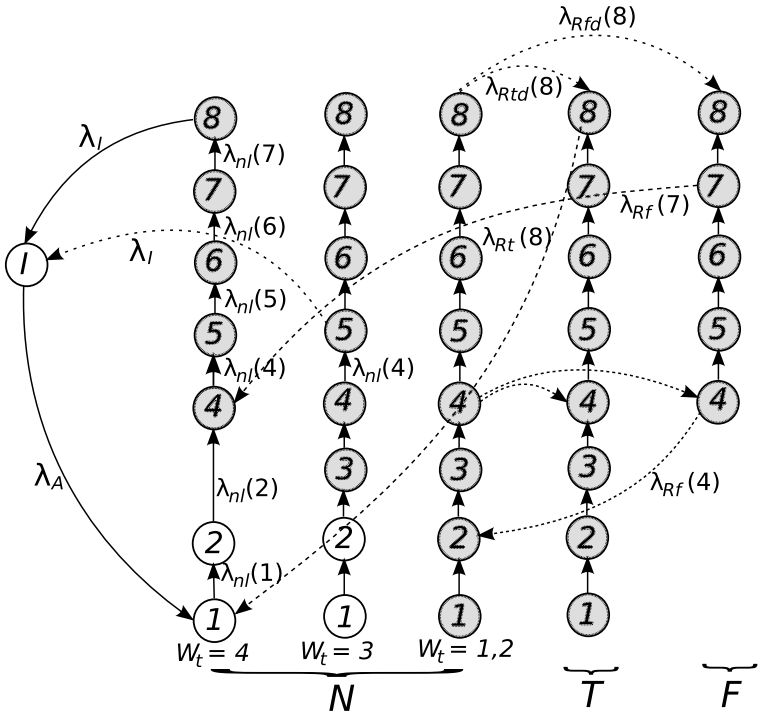

Figura 3.3: Diagrama de Transição de Estados [Dimopoulos et al., 2005]

Tabela 3.1: Descrição das Taxas de Transição

Taxas	Descrição
λ_I	Taxa de término da conexão
λ_A	Taxa de início de conexão
λ_{nl}	Taxa de transição normal
λ_{Rf}	Taxa de saída do $fastretransmit$
λ_{Rt}	Taxa de saída do $timeout$
λ_{Rfd}	Taxa de entrada do $fastretransmit$
λ_{Rtd}	Taxa de entrada do $timeout$

Para cada estado ativo existem três diferentes taxas de transição, *timeout*, *fast retransmit* e normal (nenhum pacote descartado), cada um tem uma probabilidade de ocorrer P_{Rt}, P_{Rf} e P_{nl}, respectivamente. Estas probabilidades são usadas para encontrar as taxas de transição esperadas para estados ativos e de perda. A duração do tempo gasto nos estados de *fast retransmit* e *timeout* determina a taxa de saída λ_{Rf} e λ_{Rt} desses estados respectivamente. A duração do tempo gasto no estado inativo determina a taxa inativa λ_I. O tempo gasto pela fonte no estado ativo e transmitindo é definido pela taxa ativa λ_A. A transição para o estado ($W = 1, W_t = \lceil W_M/2 \rceil, N$), que é o estado inicial das conexões TCP, representa uma nova conexão. Existem também, transições de todos os estados para o estado inativo, caracterizando fim da conexão.

A tabela 3.2 apresenta uma simples descrição das transições entre cada estado. Um estado ativo/transmitindo é denotado por U_k=(W,W_t,N). Todas as transições do estado U_k para U_{k+1}

Tabela 3.2: Descrição das Transições entre Estados

Transição			Descrição
(W,W_t,N)	–>	(2W,W_t,N)	*Slow start* quando o próximo tamanho de janela é menor que W_t
(W,W_t,N)	–>	(W_t,W_t,N)	*Slow start* quando o próximo tamanho de janela é maior que W_t
(W,W_t,N)	–>	(W+1,W_t,N)	*Congestion Avoidance* enquanto W é menor que W_M
(W,W_t,N)	–>	(W,T)	Taxa de detecção de perda por *Timeout*
(W,W_t,N)	–>	(W,F)	Taxa de detecção de perda por *FastRetransmit*
(W,W_t,N)	–>	I	Taxa de não envio de dados
(W,T)	–>	(1,$\lceil$ W/2 $\rceil$,N)	Taxa *Timeout*
(W,T)	–>	I	Nenhum dado para enviar
(W,F)	–>	($\lceil$ W/2 $\rceil$, $\lceil$ W/2 $\rceil$,N)	Taxa de *FastRetransmit*
(W,F)	–>	I	Nenhum dado para enviar
I	–>	(1,$\lceil$ W_M/2 $\rceil$,N)	Taxa de Novas Conexões

são dados pela equação 3.7 com suas respectivas taxas de transição. Por exemplo se $W_M = 8$ e o estado atual é *(4,2,N)* então a transição para o estado *(5,2,N)* ocorre se nenhum pacote for descartado. Quando uma perda ocorre então acontece uma transição para o estado *(4,T)* se for um *timeout*, ou *(4,F)* se for um *fast retransmit*. Se não existem mais dados para enviar, então ocorre uma transição para o estado inativo *I*.

$$U_{k+1} = \begin{cases} (2W, W_t, N) & \lambda_{nl} & W < W_t \\ & & 2W < W_t \\ (W_t, W_t, N) & \lambda_{nl} & W < W_t \\ & & 2W \geq W_t \\ (W+1, W_t, N) & \lambda_{nl} & W_t \leq W < W_M \\ (W, T) & \lambda_{Rtd} & \\ (W, F) & \lambda_{Rfd} & \\ I & \lambda_I & \end{cases} \tag{3.7}$$

Um *timeout* reduz o tamanho de janela para um e a janela de *threshold* para $\lceil W/2 \rceil$ que é a metade do tamanho da janela quando a perda ocorre. Do estado *timeout* U_k = *(W,T)*, contudo, pode ocorrer uma transição para um estado ativo (1,$\lceil W/2 \rceil$,N) ou para o estado inativo, caso não existam mais dados para enviar (equação 3.8).

$$U_{k+1} = \begin{cases} (1, \lceil W/2 \rceil, N) & \lambda_{Rt} \\ I & \lambda_I \end{cases} \tag{3.8}$$

As transições do estado *fast retransmit* $U_k = (W,F)$ são as mesmas do estado *timeout*, exceto que o tamanho da janela reduz pela metade quando ocorre uma perda, em vez de ir para janela de tamanho um como no *timeout*(equação 3.9).

$$U_{k+1} = \begin{cases} (\lceil W/2 \rceil, \lceil W/2 \rceil, N) & \lambda_{Rf} \quad W > 3 \\ I & \lambda_I \end{cases} \tag{3.9}$$

A transição do estado Inativo $U_k = I$ para o estado Ativo inicial (1,⌈W/2⌉,N) modela a chegada de uma nova conexão(equação 3.10).

$$U_{k+1} = (1, \lceil W_M/2 \rceil, N) \quad \lambda_A \tag{3.10}$$

Note que as taxas λ_{nl}, λ_{Rt} e λ_{Rf} estão todas em função do tamanho de janela *W*. Para encontrar a distribuição estacionária π_S do processo de Markov $\{U_k\}$ deve-se primeiro determinar as probabilidades de transições e taxas requeridas. Para encontrar a vazão é necessário determinar o número de pacotes transmitidos em cada estado e a duração de cada transição.

A seguir apresenta-se de forma sucinta o calculo das várias taxas de transição requeridas pelo modelo. A explicação detalhada destes cálculos é encontrada em [Dimopoulos et al., 2005].

Transições Ativas e Inativas

O tempo de conexão ativo $T_A = \frac{1}{\lambda_A}$ é a duração do tempo em que cada fonte TCP possui dados para transmitir. O tempo de inatividade da conexão $T_I = \frac{1}{\lambda_I}$ é a duração do tempo em que nenhum pacote pode ser transmitido pelas fontes.

Transições Normais

A probabilidade de que o tamanho da janela aumente é a probabilidade de que nenhum pacote seja descartado na rajada P_{nl} (equação 3.11), onde *P* é a probabilidade de uma simples perda de pacote e *W* é o tamanho da janela representando o número de pacotes transmitidos.

$$P_{nl}(W) = (1-P)^W \tag{3.11}$$

A taxa esperada de transição normal é derivada da duração da transição normal (um RTT, T_{rtt}) e da probabilidade da transição P_{nl} e é dada pela equação 3.12. Um RTT (equação 3.13) é composto pelo atraso de propagação do enlace T_τ, o atraso médio da fila T_Q e o tempo de serviço do enlace congestionado T_μ como mostra a figura 3.4. Foi assumido que o tempo para o processamento de um ACK é insignificante comparado com o atraso de propagação do enlace desde que os pacotes ACKs sejam pequenos.

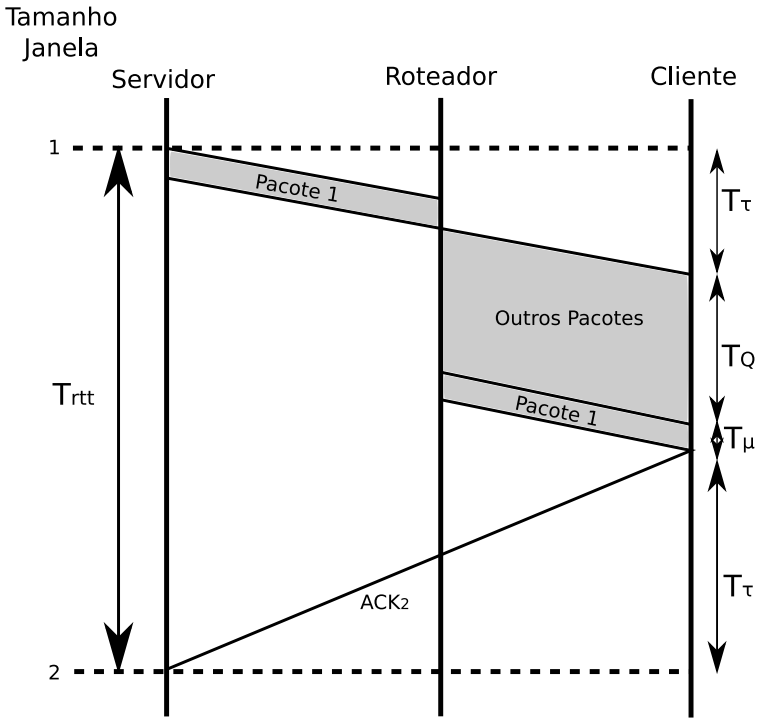

Figura 3.4: Componentes do RTT

$$\lambda_{nl}(W) \simeq \frac{P_{nl}(W)}{T_{rtt}} \tag{3.12}$$

$$T_{rtt} = 2T_\tau + T_\mu + T_Q \tag{3.13}$$

A taxa de transição λ_{nl} é a mesma para ambos *slow start* e *congestion avoidance*.

Transições com Perdas

Uma perda em uma rajada de tamanho *W* cria uma transição para o estado com perda (*W,T*) ou (*W,F*). Seja $P_N(X=x/W)$ a probabilidade que *x* de *W* pacotes sejam descartados (assumindo que as perdas de pacote são independentes, conforme equação 3.14). A probabilidade de um *timeout* é a probabilidade de menos de três pacotes serem transmitidos com sucesso em uma rajada (equação 3.15). A probabilidade do *fast retransmit* P_{Rf} é encontrada utilizando o teorema da probabilidade total como mostra a equação 3.16.

$$P_N(X = x|W) = \binom{W}{x} P^x (1-P)^{W-x} \tag{3.14}$$

$$P_{Rt}(W) = (1 - P_{nl}(W)) \sum_{x=W_M-2}^{W_M} P_N(x|W) \tag{3.15}$$

$$P_{Rf}(W) = 1 - P_{Rt}(W) - P_{nl}(W) \tag{3.16}$$

Com essas probabilidades pode-se aproximar as taxas de transição esperadas para o *fast retransmit* e *timeout*, conforme equações 3.17 e 3.18 respectivamente.

$$\lambda_{Rfd}(W) \simeq \frac{P_{Rf}(W)}{T_{rtt}} \tag{3.17}$$

$$\lambda_{Rtd}(W) \simeq \frac{P_{Rt}(W)}{T_{rtt}} \tag{3.18}$$

Fast Retransmit

Assume-se que leva aproximadamente um RTT para detectar uma perda por *fast retransmit* e um RTT para recuperar cada um dos pacotes perdidos. A taxa de saída do estado de *fast*

retransmit é dada pela equação 3.20 onde *X(W)* é o número esperado de pacotes descartados (equação 3.19).

$$X(W) = \sum_{x=1}^{W} xP_N(x|W) \tag{3.19}$$

$$\lambda_{Rf} = \frac{1}{X(W)T_{rtt}} \tag{3.20}$$

Se *X(W)* pacotes são descartados então (*W* - *X*) ACKs são recebidos, cada qual transmite um pacote. Cada pacote perdido é também retransmitido, fazendo o número de pacotes transmitidos igual a equação 3.21.

$$G_{Rf}(W) = W \tag{3.21}$$

Timeout

Se não existirem ACKs duplicados suficientes para disparar um *fast retransmit* então um *timeout* irá ocorrer quando o tempo de retransmissão expirar. T_O é a duração do *timeout* (RTO) de retransmissão. O RTO é definido como T_{RTT} + *4RTTVAR*, onde RTTVAR é a variância do RTT. A variância foi aproximada com o atraso médio da fila T_Q. A taxa de transição *timeout* é dada pela equação 3.22.

$$\lambda_{Rt} = \frac{1}{T_O} = \frac{1}{T_{rtt} + 4T_Q} \tag{3.22}$$

A probabilidade de Z perdas na janela de tamanho W é dada pela equação 3.23. O número de pacotes transmitidos durante um *timeout* é dado pela equação 3.25.

$$P(Z = z|W) = \frac{1}{1 - P_{nl}(W)} \sum_{x=1}^{W-(z-1)} P(Z|x, W) \tag{3.23}$$

onde:

$$P(Z = z|X = x, W) = \frac{\binom{W-z}{x-1}}{\binom{W}{x}} \tag{3.24}$$

$$G_{Rt}(W) = \left\lceil \sum_{z=1}^{3} P(z|W) \right\rceil - 1 \tag{3.25}$$

Dadas todas as taxas de transição (λ's) e o número de pacotes transmitidos (G's) para cada transição pode-se encontrar a distribuição estacionária de U_k e então, a vazão, a distribuição do tamanho da rajada, e a taxa de chegada de rajadas.

Vazão

A vazão total esperada da fonte TCP, em pacotes/segundo, é encontrada ao multiplicar a vazão gerada por cada estado (quantidade de dados dividida pelo tempo de transmissão de dados) pela probabilidade de estar no estado π_{U_K} (equação 3.26).

$$\lambda_R \simeq \sum_{W=1}^{W_M} \sum_{W_t=1}^{\lceil W_M/2 \rceil} \pi_{(W,W_t,N)} \frac{W}{T_{rtt}} + \sum_{W=1}^{W_M} \pi_{(W,F)} G_{Rf} \lambda_{Rf} + \sum_{W=1}^{W_M} \pi_{(W,T)} G_{Rt} \lambda_{Rt} \tag{3.26}$$

O primeiro termo é gerado por todos os estados ativos, o segundo e terceiro termos são gerados pelos estados *fast retransmit* e *timeout* respectivamente.

Distribuição do Tamanho da Rajada

Pode-se aproximar a distribuição de tamanho da rajada B utilizando a equação 3.27 que soma todos os estados de U_k com o mesmo tamanho de janela.

$$B(W) = \sum_{W_t=1}^{\lceil W_M/2 \rceil} \pi_{(W,W_t,N)} \qquad W \in \{1, 2, ...W_M\} \tag{3.27}$$

Nota-se que $B(W)$, por ser uma distribuição, deve ser normalizada.

Taxa de Chegada da Rajada

A taxa de chegada da rajada na fila do roteador, em rajadas/segundo, é estimada dividindo-se a vazão pela média do tamanho da rajada como segue:

$$\lambda_{Rb} = \frac{\lambda_R}{\overline{W}} \tag{3.28}$$

$$\overline{W} = \sum_{W=1}^{W_M} WB(W) \tag{3.29}$$

A taxa de chegada da rajada é requerida pelo modelo de rede descrito a seguir.

3.1.3 Modelo de Rede

A rede é modelada assumindo que existe um roteador de borda com enlace congestionado e outros roteadores presentes na rede não contribuem significativamente para o atraso de fila e perdas de pacotes. Este roteador é modelado matematicamente por uma fila com *buffer* limitado.

Considerando que o modelo de fila clássico $M/M/1/K$ [Kleinrock, 1976] não é capaz de modelar a correlação entre os pacotes que chegam ao roteador em diferentes momentos, adota-se o modelo de fila $M_X/M/1/K$ proposto em [Garetto e Towsley, 2003] e [Garetto et al., 2004]. Esse modelo corresponde a uma fila Markoviana com *chegada em grupos*.

A distribuição do tamanho da rajada (B) obtido na seção anterior será utilizada agora para aproximar a distribuição do tamanho dos grupos de pacotes (X) que entram na fila. A probabilidade de perda de pacotes P, e o atraso médio dos pacotes T_Q, são calculados pela solução da Cadeia de Markov em Tempo Contínuo (CTMC) que modela a fila. O modelo de fila $M_X/M/1/K$ foi extensivamente analisado e tem se mostrado bastante preciso na avaliação do comportamento de filas alimentadas com tráfego TCP [Garetto e Towsley, 2003].

A taxa de chegada em rajada na fila (λ_α) corresponde à taxa de chegada agregada de todas as fontes TCP. Consideram-se N diferentes fontes TCP, cada uma com uma taxa de chegada $\lambda_{Rb}(n)$, e com diferentes atrasos de propagação do enlace.

$$\lambda_\alpha = \sum_{n=1}^{N} \lambda_{Rb}(n) \tag{3.30}$$

A taxa de chegada esperada da rajada de tamanho W é a probabilidade da rajada $B(W)$ multiplicada pela taxa de chegada.

$$\lambda_b(W) = B(W)\lambda_\alpha \tag{3.31}$$

A figura 3.5 mostra o diagrama de transição de estados da fila $M_X/M/1/K$, onde μ_S é a taxa de serviço que depende da capacidade do enlace congestionado. Nota-se que o modelo proposto e utilizado em [Dimopoulos et al., 2005] difere deste modelo no que se refere as taxas de serviço.

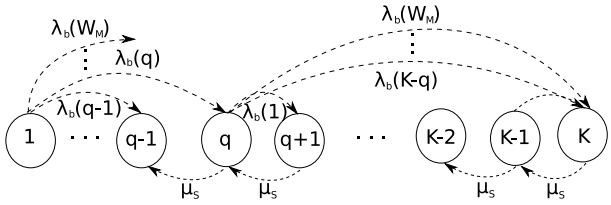

Figura 3.5: Diagrama de Transição de Estados da fila

No modelo utilizado em [Dimopoulos et al., 2005] a taxa de serviço para o estado $K-1$ é modificada por um fator de correção de modo a melhor representar o comportamento do sistema quanto à chegada das rajadas de pacotes.

4 PROPOSTA E RESULTADOS

A proposta dessa dissertação é analisar o comportamento do protocolo TCP TS-Prio, proposto em [Tostes, 2004], que busca a diferenciação de serviços. Para que fosse possível chegar a este resultado, foi necessário modificar o modelo analítico TCP proposto em [Dimopoulos et al., 2005], bem como os algoritmos necessários do Método do Ponto Fixo, obtendo assim uma ferramenta computacional para análise e projeto.

Para o modelamento do TCP TS-Prio o procedimento adotado foi o seguinte: os fluxos SBE apresentam tamanho máximo de janela fixo, e os fluxos TBE reduzem suas janelas máximas pela metade quando existe a presença dos fluxos SBE (isto é alcançado simplesmente reduzindo o diagrama de transição de estados, que descreve o protocolo, pela metade).

Para medir a diferenciação de serviço, existente entre os fluxos prioritários e não-prioritários, foi utilizado o índice de justiça proposto por [Jain, 1991].

Este capítulo está dividido em duas partes. A primeira parte trata da validação do modelo analítico implementado (que é baseado no modelo proposto em [Dimopoulos et al., 2005]). Na segunda parte estuda-se o comportamento do protocolo TCP TS-Prio. Para isso o modelo analítico implementado é adaptado de modo a simular fontes do tipo SBE e TBE. O modelo é então utilizado para a análise de desempenho do protocolo TCP TS-Prio, considerando alguns casos de interesse.

4.1 Validação

Um modelo Markoviano para análise do protocolo TCP-Reno, hamado nesta dissertação de modelo original, foi proposto em [Dimopoulos et al., 2005]. Este modelo foi validado pela comparação com resultados numéricos gerados por simulações realizadas no NS-2. A topologia de rede considerada é mostrada na Figura 5.1. O enlace entre os roteadores R1 e R2 é o único elo congestionado da topologia, tem uma capacidade de 40 Mbps, o buffer pode acomodar até 50 pacotes e o tipo de fila utilizado no roteador é *Drop-tail*. Cada fonte TCP-Reno é simulada

tendo como entrada uma fonte de dados do tipo ON-OFF exponencial, com um tamanho de janela inicial setado em um para estar de acordo com o modelo analítico. Os demais parâmetros utilizados no sistema podem ser observados na Tabela 5.1.

O modelo analítico implementado nesta dissertação corresponde aquele proposto em [Dimopoulos et al., 2005], exceto pelo uso da fila tipo $M_X/M/1/K$ (proposta e analisada em [Garetto e Towsley, 2003] e [Garetto et al., 2004]). A seguir compara-se os resultados obtidos utilizando o modelo analítico implementado com os resultados presentes em [Dimopoulos et al., 2005], visando a validação do modelo implementado.

Tabela 4.1: Valores Utilizados na Análise

Descrição	Variável	Valor
Tamanho do Pacote	S	1500 bytes
Tamanho da Janela	W_M	16 pacotes
Capacidade do Link	T_σ	0.3ms (40Mbps)
Retardo de Propagação	T_τ	200ms
Tempo Ativo	T_A	4s
Tempo Inativo	T_I	1.6s
Tamanho do Buffer	K	50 pacotes

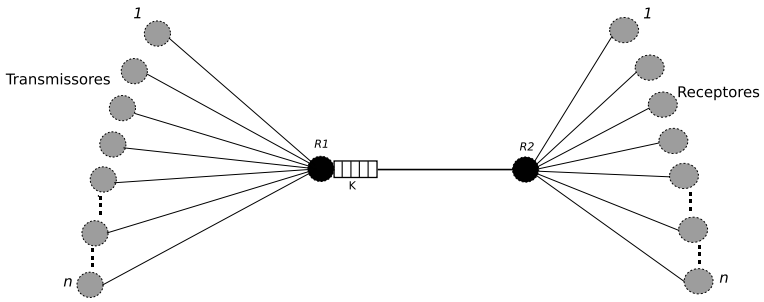

Figura 4.1: Topologia de Rede

A figura 5.2 mostra o valor da vazão pela variação do número de fontes. Quando existe apenas um pequeno número de fontes, a vazão não está limitada pelo controle de congestionamento do TCP. Conforme esse número aumenta, o buffer começa a ficar congestionado, então, perdas de pacotes começam a ocorrer. Pela técnica de controle de congestionamento do TCP, o tamanho da janela de transmissão começa a diminuir e, naturalmente, a vazão vai estabilizando. Este é o ponto onde as perdas de pacotes mais sofrem efeitos pelo tráfego em rajadas.

Também pode-se observar, que o *buffer* está sendo totalmente utilizado quando a vazão chega próximo a um, onde os modelos fornecem resultados mais próximos da simulação. O valor da vazão em todas as figuras está normalizado em função da capacidade do enlace congestionado. Os valores maiores do que um indicam que a carga oferecida ao roteador é maior que a capacidade do enlace e pacotes devem ser descartados.

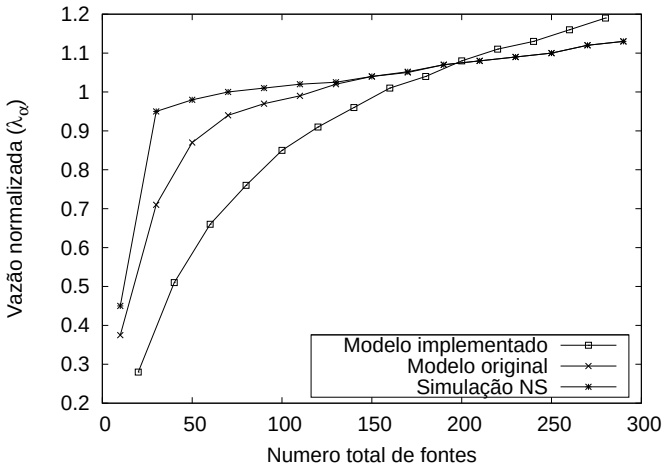

Figura 4.2: Vazão

A redução do tamanho da janela e consequentemente da rajada é mostrada claramente na figura 5.3, onde o tamanho médio da janela diminui quando o número de fontes aumenta.

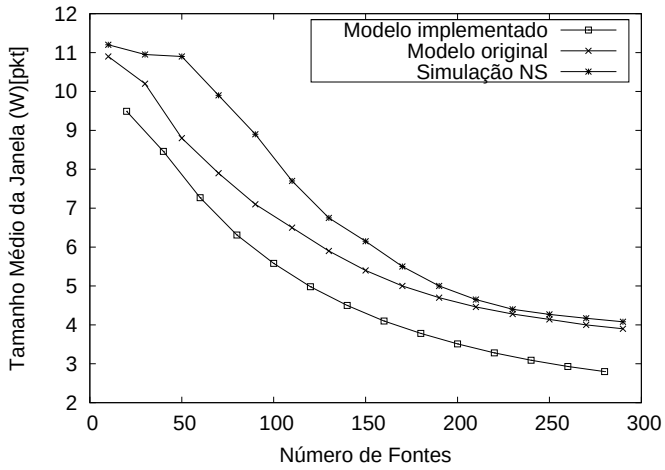

Figura 4.3: Tamanho médio da janela

A figura 5.4 mostra a probabilidade de perda de pacotes aumentando, como esperado, conforme aumenta o número de fontes.

Observam-se diferenças entre os resultados obtidos utilizando o modelo implementado e aqueles obtidos com o modelo original. Acredita-se que tais diferenças são oriundas do modelo de fila $M_X/M/1/K$ utilizado no modelo implementado neste trabalho. Entretanto, o comportamento geral dos indicadores pode ser previsto com o uso do modelo implementado, o que valida o seu uso.

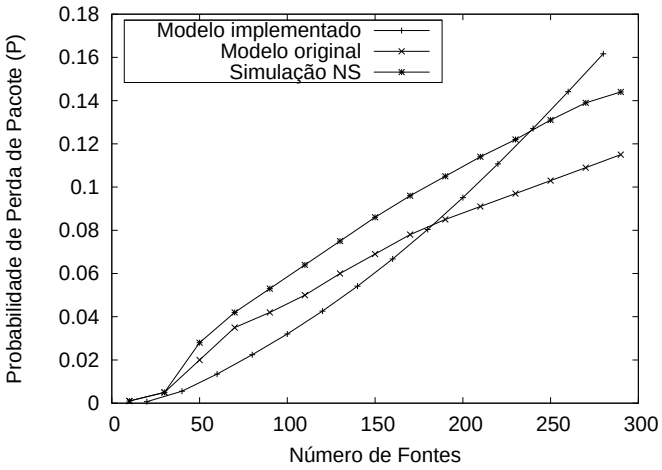

Figura 4.4: Probabilidade de Perda de Pacotes

4.2 Análise do TCP TS-Prio

Assim validado o sistema implementado, é possível fazer a análise do protocolo TCP TS-Prio que é a proposta desta dissertação. A seguir serão demonstrados e analisados os resultados de desempenho obtidos para este novo protocolo.

4.2.1 Cenário 1

Na figura 5.5 pode-se observar a alteração realizada no modelo de fontes, onde no modelo original tinha-se apenas um grupo de fontes transmissoras, porém agora, tem-se uma separação das fontes transmissoras em dois grupos, um grupo com prioridade e outro sem prioridade, isso foi necessário para que fosse possível simular o comportamentos dos fluxos TCP TS-Prio.

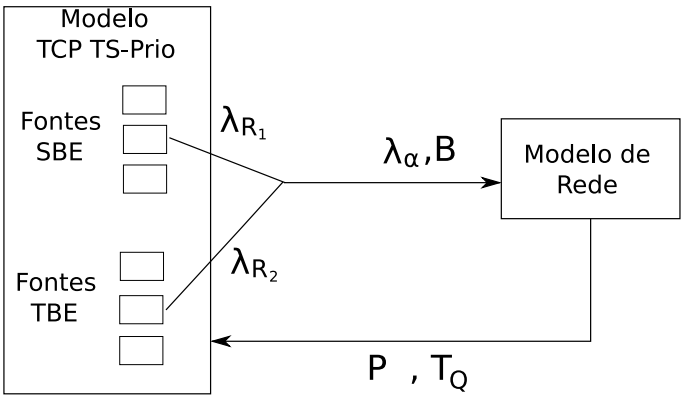

Figura 4.5: Configuração da Simulação TCP TS-Prio

Com este cenário é possível obter o desempenho para uma topologia semelhante aquela da figura 5.1, porém considerando o TCP TS-Prio. Este cenário é alimentado por dois grupos de

fontes TCP, o primeiro grupo é denominado SBE (*Static Best Effort*) que como já dito anteriormente, agrega fluxos com prioridades. O segundo grupo, é denominado TBE (*Transient Best Effort*) que agrega fluxos não prioritários. Esse cenário pode corresponder ao caso onde uma ISP provê conectividade entre 2 sítios (matriz e filial), pertencentes a uma empresa qualquer.

Para analisar o comportamento dos dois grupos foram montadas várias configurações do cenário. O primeiro cenário a ser analisado possui um número variável de fontes, sendo o mesmo número para os dois grupos, e para os tempos T_τ = 0,05 segundo e T_τ = 0,1 segundo. Os fluxos SBE apresentam tamanho máximo de janela igual a 16 segmentos. Os fluxos TBE reduzem suas janelas máximas a 8 segmentos quando existe a presença dos fluxos SBE no mesmo enlace.

Nas figuras 5.6 e 5.7 pode-se observar o comportamento da probabilidade de perdas de pacotes e o tempo de fila, para o fluxo agregado, conforme o número de fontes aumenta. O número de fontes é a soma entre os dois grupos de fontes TCP existentes nesse cenário.

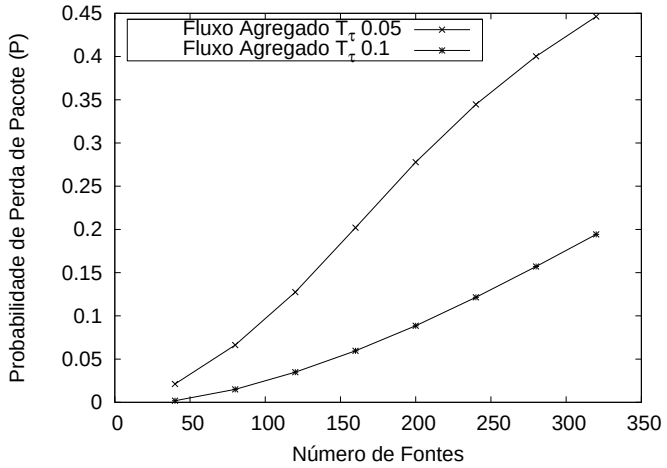

Figura 4.6: Probabilidade de perda de pacote do fluxo agregado

Também é possível observar nas figuras 5.8 e 5.9 o comportamento dos fluxos separadamente, tanto para a vazão quanto para o valor do tamanho médio da janela TCP.

Pode-se verificar que conforme o número de fontes TCP aumenta, a vazão também aumenta, porém como o fluxo SBE tem prioridade, ele vai utilizar mais a rede. Também é possível visualizar que com essa prioridade, o valor médio do tamanho da janela do fluxo prioritário é maior do que o não prioritário.

A influência do retardo de propagação pode também ser observada nas figuras. O aumento do atraso de propagação faz com que o TCP ajuste para cima suas estimativas do RTT. Neste caso, o RTO (*Retransmission timeout*) tem seu valor elevado de modo que o protocolo per-

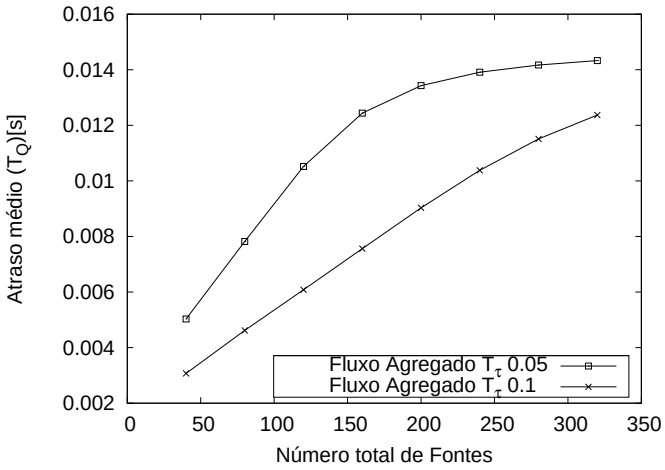

Figura 4.7: Atraso médio do fluxo agregado

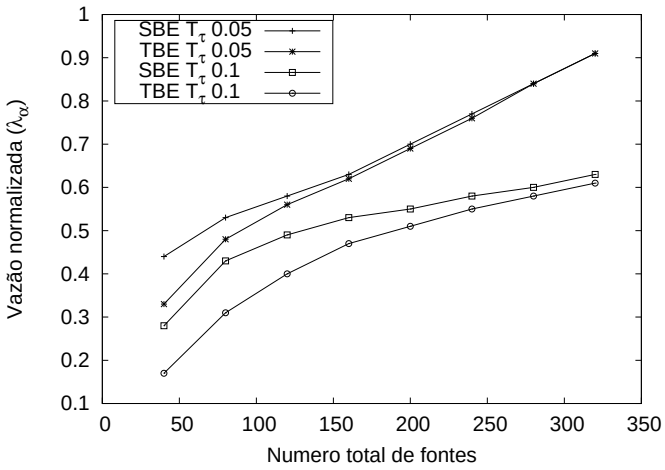

Figura 4.8: Vazão dos fluxos SBE e TBE

manece por mais tempo nos estados normais, elevando assim a janela média. Por outro lado, o TCP envia rajadas de dados e espera mais tempo pelos ACKs, o que reduz a vazão e por consequência o valor da probabilidade de perda de pacotes e do atraso médio.

A segunda configuração de cenário a ser analisado, possui um número fixo de fontes TCP para o fluxo SBE, fixado esse valor em quarenta fontes. O número de fontes para os fluxos TBE é variável e crescente, e considera-se T_τ = 0,1 segundo.

Nas figuras 5.10 e 5.11 pode-se analisar o comportamento da probabilidade de perdas de pacotes e o tempo de fila, conforme o número de fontes aumenta. O número de fontes corresponde à soma entre os dois grupos de fontes TCP existentes nesse cenário.

Já nas figuras 5.12 e 5.13, pode-se observar o comportamento dos fluxos separadamente, tanto para a vazão quanto para o valor do tamanho médio da janela TCP utilizada por cada fluxo.

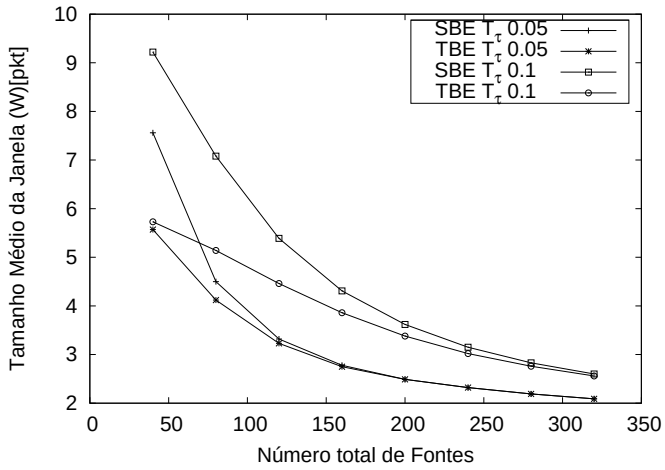

Figura 4.9: Tamanho médio da janela dos fluxos SBE e TBE

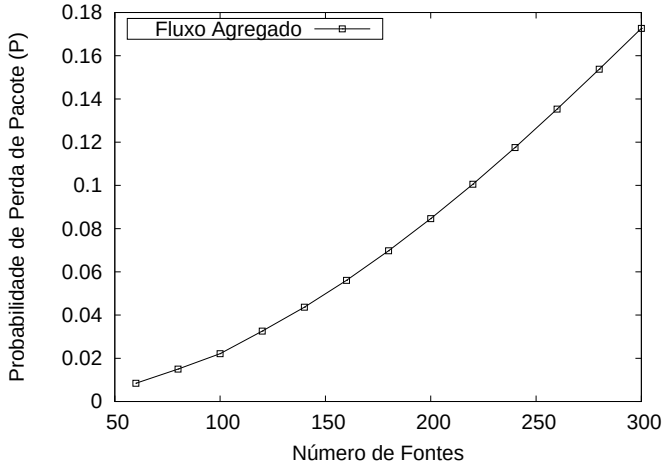

Figura 4.10: Probabilidade de perda de pacote do fluxo agregado

Nessa configuração é possível perceber que quando o número de fontes SBE e TBE são iguais (totalizando 80 fontes), o SBE utiliza mais a rede (vazão) e o tamanho de janela médio é maior. Porém conforme o número de fontes TBE aumenta, a vazão do fluxo SBE diminui, pois ele possui um número menor de fontes. O mesmo ocorre com o tamanho médio da janela, porém conforme a rede fica congestionada, o tamanho médio de janela tende a entrar em equilíbrio, tanto dos fluxos SBE e TBE.

A terceira configuração do cenário à ser analisada, possui um número fixo de fontes TCP para o fluxo TBE, fixado esse valor em quarenta fontes. O número de fontes para os fluxos SBE é variável e crescente, e considera-se T_τ = 0,1 segundo.

O comportamento da probabilidade de perdas de pacotes e o tempo de fila, nessa configuração, é muito semelhante a configuração anterior.

Nas figuras 5.14 e 5.15, pode-se analisar o comportamento dos fluxos separadamente, tanto

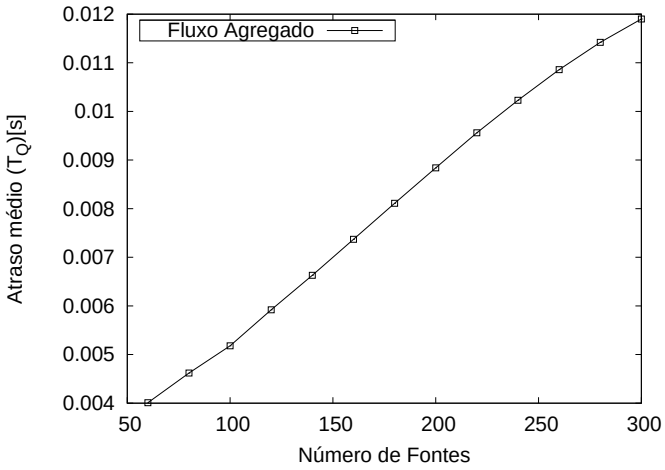

Figura 4.11: Atraso médio do fluxo agregado

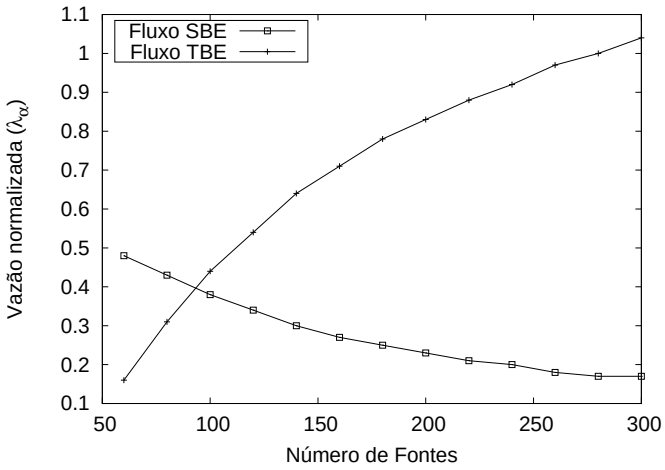

Figura 4.12: Vazão dos fluxos SBE e TBE

para a vazão quanto para o valor do tamanho médio da janela TCP utilizada por cada fluxo.

Já nesta configuração, como visto anteriormente, conforme aumenta o número das fontes SBE, aumenta a vazão na rede e consequentemente o valor da vazão dos fluxos TBE diminui. Porém, também é possível perceber que o tamanho de janela tente a entrar em equilíbrio. Onde, independente do fluxo ter ou não prioridade, conforme aumenta a carga na rede, o tamanho da janela de transmissão dos dados tende a se igualar.

Utilizando a topologia anterior (figura 5.1), foi criada uma nova simulação, onde o número de fontes de cada fluxo (SBE e TBE) aumenta de forma independente, conforme a figura 5.16.

As figuras 4.14 até 4.18 mostram os resultados obtidos com essa nova simulação.

Inicialmente somente os fluxos TBE estão presentes e o número de fontes aumenta com o passar do tempo até o valor de 50 fontes. Como esperado, o tamanho médio da janela diminui e a

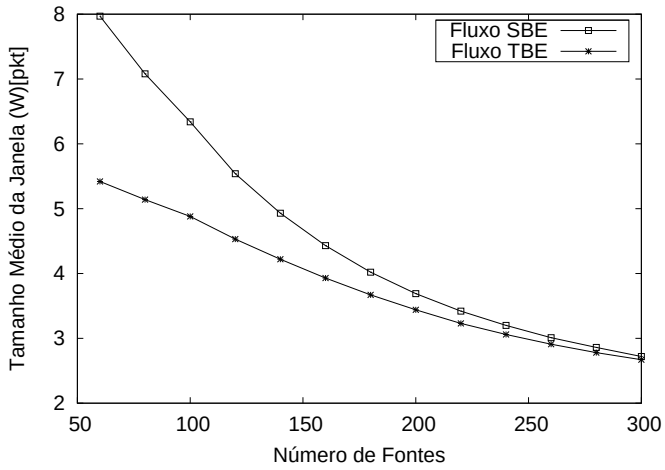

Figura 4.13: Tamanho médio da janela dos fluxos SBE e TBE

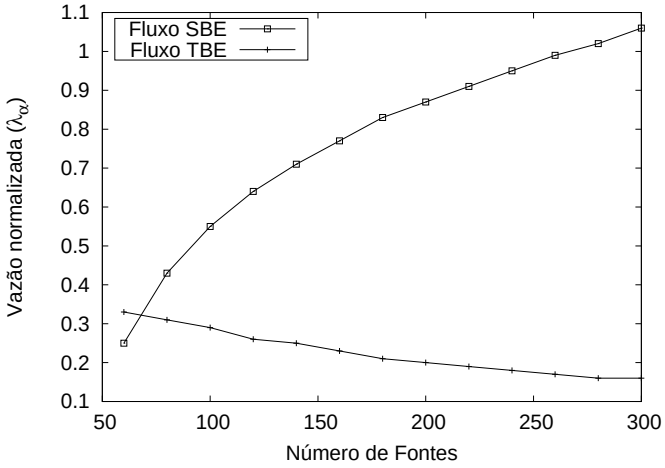

Figura 4.14: Vazão dos fluxos SBE e TBE

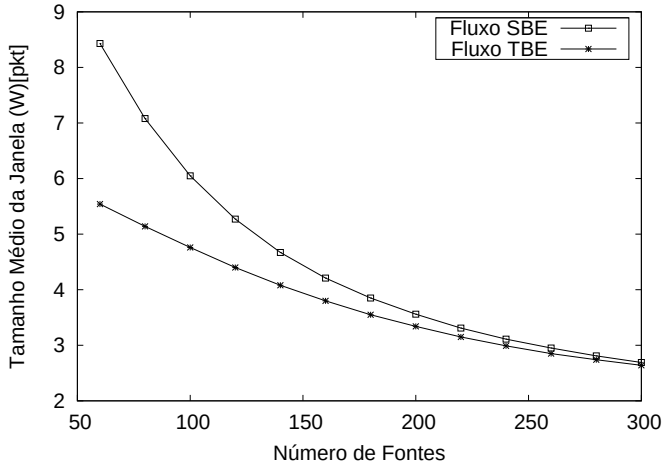

Figura 4.15: Tamanho médio da janela dos fluxos SBE e TBE

vazão aumenta com o aumento do número de fontes. No instante de tempo igual a oito, os fluxos SBE passam a estar presentes e, então, as fontes TBE reduzem suas janelas máximas à metade

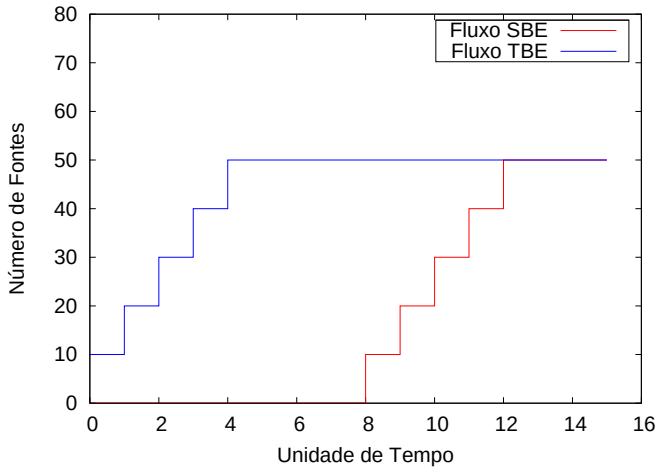

Figura 4.16: Configuração da simulação dos fluxos SBE e TBE

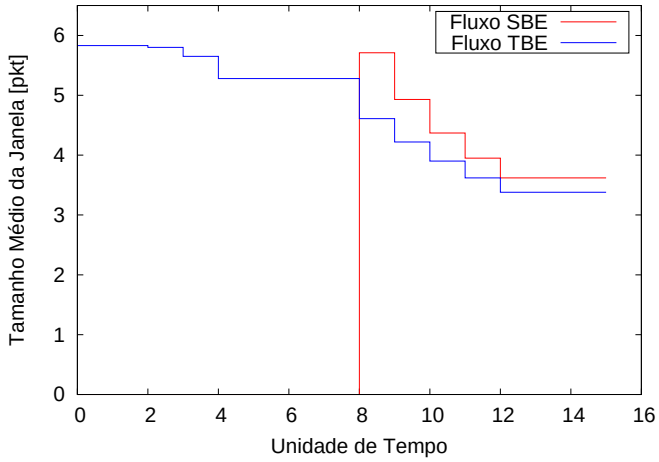

Figura 4.17: Tamanho médio da janela dos fluxos SBE e TBE

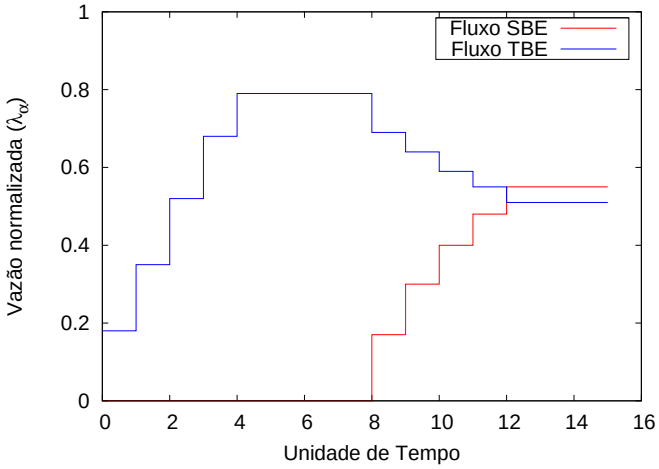

Figura 4.18: Vazão dos fluxos SBE e TBE

(dando prioridade aos fluxos SBE). Com a presença dos dois tipos de fluxos, o comportamento do sistema é semelhante aos casos já analisados.

4.2.2 Cenário 2

Uma nova topologia de rede foi desenvolvida, com o objetivo de observar o comportamento das fontes SBE e TBE considerando valores diferenciados para T_τ como mostra a Figura 5.19. O enlace entre os roteadores R1 e R2 é agora uma parte da topologia, continuando com uma capacidade de 40 Mbps. Porém agora foram adicionados mais dois roteadores, sendo o R3 o elo entre as conexões das fontes SBE e o roteador R4 o elo entre as fontes TBE. O buffer de *R*1 (único roteador congestionado) pode acomodar até 50 pacotes e o tipo de fila utilizado no roteador é *Drop-tail*. Esse cenário foi alimentado com 100 fontes, sendo 50 fontes para fluxos TBE e 50 fontes para os fluxos SBE. O valor do atraso de propagação T_τ entre os roteadores *R*2 e *R*3 é de 0,1 segundo e entre os roteadores *R*2 e *R*4 o T_τ é variável.

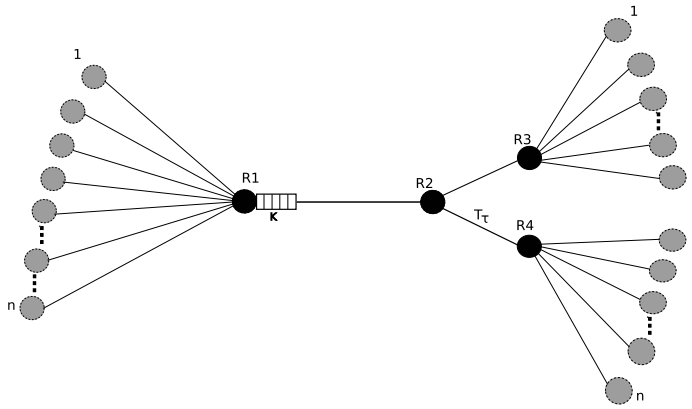

Figura 4.19: Topologia para o cenário 2

Esse cenário pode corresponder ao caso onde um ISP provê conectividade entre 3 sítios (matriz e duas filiais com necessidades diferenciadas quanto à prioridade de serviço) pertencentes a uma empresa qualquer.

Observando os gráficos, das figuras 5.20 à 5.23, pode-se concluir que com o aumento da distância física entre os roteadores *R*2 e *R*4 (aumento do T_τ), os fluxos TBE tem sua vazão reduzida (espera maior por ACKs) e como consequência suas necessidades por banda. Desta forma os fluxos SBE podem aumentar sua vazão e valor médio da janela. A vazão agregada diminui e como consequência diminuem o atraso médio e probabilidade de perda de pacotes.

4.2.3 Índice de Justiça

Para que fosse possível medir a distribuição dos recursos de rede e comparar a diferença na utilização do enlace pelos diversos fluxos, foi utilizado o índice de justiça definido por [Jain, 1991] para os cálculos. Para justificar seu índice de justiça [Jain, 1991] explica que:

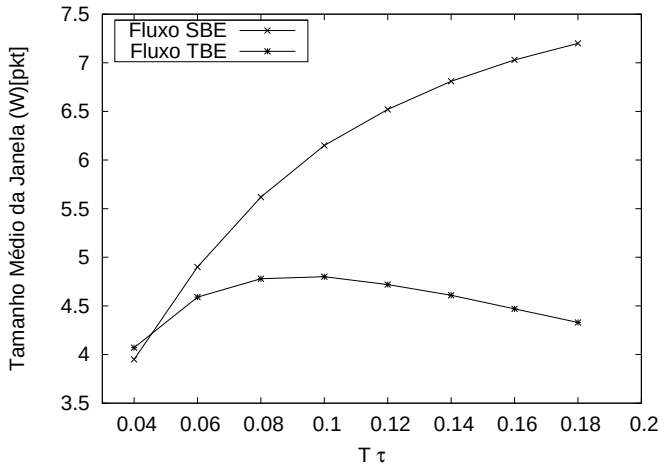

Figura 4.20: Tamanho médio da janela dos fluxos SBE e TBE

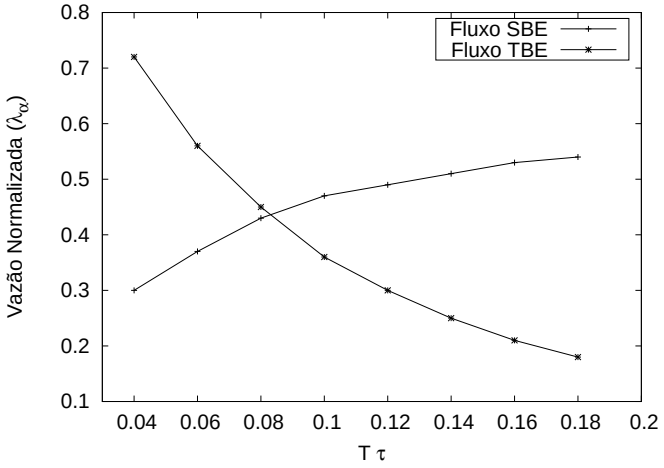

Figura 4.21: Vazão dos fluxos SBE e TBE

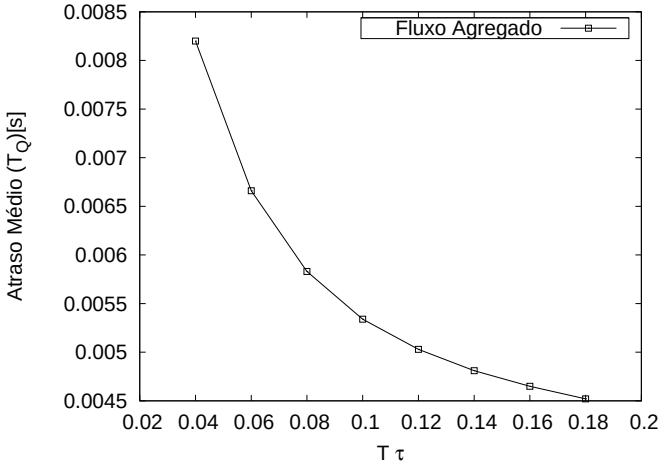

Figura 4.22: Atraso médio do fluxo agregado

“A rede é um sistema multiusuário. É necessário que todos os usuários sejam tratados de forma justa. Portanto foi adicionada a justiça como uma métrica de desempenho. Ela é definida como

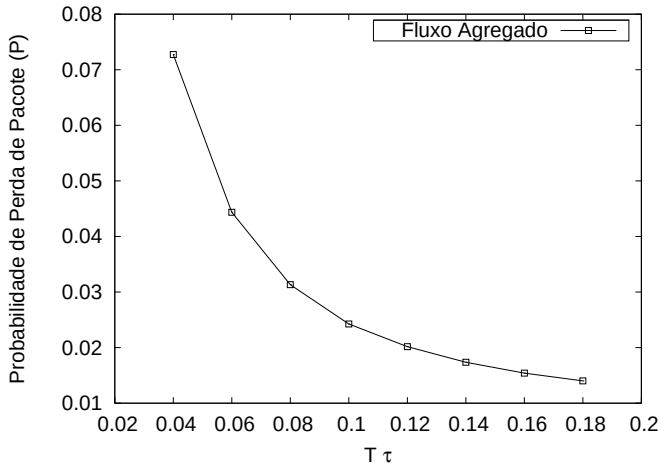

Figura 4.23: Probabilidade de perda de pacote do fluxo agregado

uma função da variação de vazão sobre número de fontes. Para um dado conjunto de vazão $(th_1, th_2, ..., th_n)$, a função apresentada na expressão 5.1 pode ser usada como um índice de justiça para o conjunto.

$$f(th_1, th_2, ..., th_n) = \frac{(\sum_{i=1}^{n} th_i)^2}{n \sum_{i=1}^{n} th_i^2} \tag{4.1}$$

Para todos os valores não negativos de th_i, o índice de justiça sempre se situa entre 0 e 1. Se todos os usuários, recebem igual vazão, o índice de justiça é 1. Se apenas *k* de *n* fontes recebem igual vazão e o restante $n - k$ das fontes recebem vazão zero, o índice de justiça é (k/n). Para outras distribuições a métrica também dá intuitivos de justiça." Analisando o gráfico da vazão do segundo cenário segundo o índice de justiça, tem-se a figura 5.24.

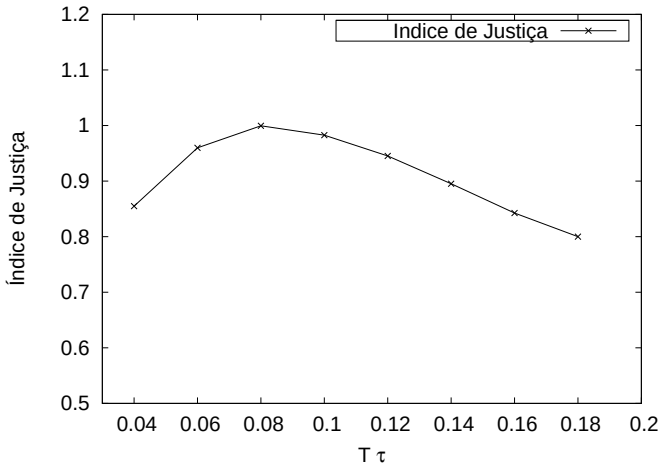

Figura 4.24: Índice de Justiça

Em função da diferenciação de serviços entre os dois caminhos da topologia, o uso igua-

litário de banda se dá quando os fluxos SBE percorrem um caminho de $T_\tau = 0,08$ segundos, conforme esperado.

5 CONCLUSÃO

Neste capítulo se encontra uma revisão dos capítulos apresentados nessa dissertação, uma breve discussão sobre as dificuldades encontradas do decorrer da implementação da ferramenta computacional, uma reapresentação dos resultados, revisando os objetivos propostos e, por fim, as conclusões obtidas após a análise dos resultados, frisando as contribuições oferecidas e indicando possíveis trabalhos futuros.

5.1 Revisão dos Objetivos e Dificuldades Encontradas

Com a necessidade de utilizar diferenciação de serviço nas redes TCP, foi apresentado o TS-Prio, que é uma modificação da versão Reno do TCP. Ele implementa uma diferenciação dos serviços simplesmente utilizando uma técnica de diferenciação no controle de congestionamento. Como o trabalho original do TCP TS-Prio focava principalmente no comportamento dinâmico da proposta, surgiu a necessidade de ser desenvolvida alguma técnica para analisar o desempenho do protocolo, revelando suas qualidades e limitações.

Para que fosse possível desenvolver a ferramenta computacional que permitiu fazer essa análise mais aprofundada do protocolo, foi necessário fazer um estudo detalhado do TCP, bem como suas diferentes implementações. Posteriormente foi implementada uma técnica já desenvolvida de modelamento do TCP, porém com diferenças quanto ao modelo de rede, pois já se tinha em mãos uma outra implementação para o modelo de rede (fila $M_X/M/1/K$). Apesar da não reprodução total do modelo [Dimopoulos et al., 2005], foi possível validar a implementação desenvolvida comparando os resultados obtidos. Com a adaptação do modelo validado, de modo a caracterizar o funcionamento do TCP TS-Prio, foi possível realizar o objetivo proposto que corresponde a análise de desempenho do protocolo.

Apesar de se possuir conhecimento da linguagem de programação utilizada, foi desgastante a implementação desse sistema para análise de desempenho. Ele foi se tornando complexo, pois, além de considerar diferentes modelos com muitas fórmulas, o software implementado

envolve cálculos com matrizes de tamanho considerável (para a janela de tamanho 16 do TCP, obteve-se uma cadeia de Markov tamanho 142x142).

5.2 Discussão dos Resultados e Contribuições

Na seção de resultados, encontra-se a análise obtida para diferentes cenários, utilizando a ferramenta computacional desenvolvida. Fica claro o funcionamento da diferenciação dos serviços proposto pelo TCP TS-Prio, porém apenas com um número limitado de fontes transmissoras(conexões). A partir de um certo número de fontes transmissoras, o desempenho das fontes prioritárias e não prioritárias passa a ser equivalente, ou seja, a capacidade de diferenciação dos fluxos é reduzida drasticamente. Assim o TCP TS-Prio é capaz de oferecer garantias de qualidade de serviço, porém isto depende diretamente do estado de congestionamento da rede em questão. Para que este objetivo possa ser alcançado faz-se necessário um sistema de controle de admissão que, de posse de dados sobre o estado da rede, pode informar à aplicação as situações onde a diferenciação de serviços é garantida.

O trabalho desenvolvido nessa dissertação, contribuiu, não apenas para afirmar o funcionamento do TS-Prio, mais também para revelar suas limitações e características, em diferentes cenários e com diferentes configurações.

5.3 Trabalhos Futuros

Utilizando como base este atual modelo é possível produzir outros estudos. É sugerido aqui estender a implementação da topologia, para redes mais complexas, simulando outros tipos de problemas que as ISPs enfrentam nos dias de hoje.

Como outra proposta de trabalho futuro, pode-se implementar o modelo de fila utilizado por [Dimopoulos et al., 2005], bem como utilizar outros modelos analíticos do TCP presentes na literatura para representar o TCP TS-Prio. Obtendo-se assim uma visão comparativa com o modelo desenvolvido nessa dissertação.

Referências Bibliográficas

[Allman et al., 1999] M. Allman, V. Paxson, e W. Stevens, RFC 2581, "TCP congestion control", 1999.

[Andrikopoulos e Pavlou, 1999] I. Andrikopoulos e G. Pavlou, "Supporting Differentiated Services in MPLS Networks", IWQoS, Junho de 1999.

[Avrachenkov et al., 2002] K. Avrachenkov, U. Ayesta, E. Altman, P. Nain, e C. Barakat, "The Effect of Router Buffer Size on the TCP Performance", Proceedings of LONIIS workshop on Telecommunication Networks and Teletraffic Theory, St.Petersburg, Russia, pp.116-121, Janeiro de 2002.

[Braden et al., 1998] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", IETF RFC 2309, Abril de 1998.

[Brakmo, 1994] L. S. Brakmo, S. W. O'Malley, e L. L. Peterson, "TCP vegas: New techniques for congestion detection and avoidance", SIGCOMM, pp 24-35, 1994.

[Brownlee e Claffy, 2002] N. Brownlee,KC. Claffy, "Understanding Internet Traffic Streams: Dragonflies and Tortoises", IEEE Comunications Magazine, Outubro de 2002.

[Bu e Towsley, 2001] T. Bu e D. Towsley, "Fixed point approximation for TCP behavior in an AQM network", Proceedings of SIGMETRICS 2001.

[Dimopoulos et al., 2005] P. Dimopoulos, P. Zeephongsekul, Z. Tari, "Modeling the burstiness of the TCP protocol", Proceedings of the 12th Annual Meeting of the IEEE / ACM International Symposium in Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Outubro de 2004.

[Floyd, 1997] S. Floyd, J. Mahdavi, M. Mathis, e M. Podolsky, RFC 2883, "An extension to the selective acknowledgement(SACK) option fot TCP", 1997.

[Floyd, 1999] S. Floyd, RFC 2582, "The NewReno Modification to TCP's Fast Recovery Algorithm", 1999.

[Garetto e Towsley, 2003] M. Garetto, D. Towsley, "Modeling, Simulation and Measurements of Queuing Delay under Long-tail Internet Traffic", ACM SIGMETRICS 2003, San Diego, CA, Junho de 2003.

[Garetto et al., 2004] M. Garetto, L. Cingo, M. Meo e M. Ajmone Marsan, "Closed queuing network models of interacting long-lived TCP flows", IEEE/ACM Transactions on Networking(TON), Abril de 2004.

[Heinanen et al., 1999] J. Heinanen, F. Baker, W. Weiss e J. Wroclawski, “RFC 2597, Assured Forwarding PHB Group”, Telia Finland, Cisco System, Lucent Technologies and MIT LCS, Junho de 1999. Disponível em: www.ietf.org/rfc/rfc2597.txt

[Jacobson, 1988] V. Jacobson, “Congestion avoidance and control”, ACM Computer Communication Review, Stanford, CA, Agosto de 1988.

[Jain, 1991] R. Jain “The Art of Computer Systems Performance Analysis - Techniques for Experimental Design, Measurement, Simulation and Modeling” New York: Wiley, 1991.

[Kleinrock, 1976] L. Kleinrock, “Queueing Systems, Volume II: Computer Applications”, Wiley Interscience, New York, 1976.

[Kilkki, 1999] K. Kilkki, “Differentiated Services for the Internet”, Macmillian Technology Series, USA 1999.

[Kurose e Ross, 2006] J. Kurose e K. Ross, “Redes de computadores e a internet : uma abordagem top-down”, Pearson Addison Wesley, 2006.

[Martinez e Santos, 1995] J. M. Martinez, S. A. Santos, “Métodos Computacionais de Otimização”, 20 Colóquio Brasileiro de Matemática, Rio de Janeiro. Anais: Instituto de Matemática Pura e Aplicada, 1995. 256 p.

[NS-2] Network Simulator, Ver.2; (NS-2), disponível em “http://www.isi.edu/nsnam/ns/”.

[Peterson e Davie, 2000] L. Peterson e B. Davie, “Computer Networks. A systems approach”, Academic Press, 2000.

[Stevens, 1994] W. Richard Stevens, “TCP/IP Illustrated: the protocols”, Addison-Wesley professional computing series, 1994.

[Tostes, 2004] M. V. Tostes, “TCP TS-Prio: Uma Abordagem de Diferenciação de Serviços Fim-a-Fim Utilizando Redução da Janela Deslizante”, Dissertação de mestrado, CPGEI/UTFPR, Curitiba, Setembro de 2004.

[Tostes e Fonseca, 2005] M. V. Tostes, K. V. O. Fonseca, “TCP TS-Prio: Uma Abordagem de Diferenciação de Serviços Fim-a-Fim para classes AF em Redes Diffserv Utilizando Redução da Janela Deslizante”, SBRC2005, Fortaleza, Maio de 2005.

RESUMO:

Este trabalho apresenta a implementação de um modelo do TCP-Reno, bem como uma técnica de modelamento de rede, detalhadamente apresentada, com o objetivo de mensurar a rajada e a vazão das conexões TCP compartilhadas em um roteador congestionado. Isso se faz necessário, para permitir estudar e prever o comportamento de uma rede. A validação do modelo implementado foi realizada por meio de comparações com resultados de simulações baseadas no software *Network Simulator* (NS-2).

Posteriormente a apresentação do modelo, e como objetivo principal deste trabalho, foi feita uma análise detalhada de uma modificação no protocolo que implementa QoS (*Quality of Service*) na camada de transporte. Essa modificação no protocolo, denominada de TCP TS-Prio, implementa uma diferenciação no tamanho da janela deslizante, através de atribuição de prioridade para determinados fluxos. Essa análise revela as qualidades e limitações da utilização desse protocolo em cenários diferentes do original do TCP TS-Prio, onde o autor focava principalmente no comportamento dinâmico do protocolo.

Como resultado do trabalho chega-se a conclusão de que o TCP TS-Prio é capaz de oferecer diferenciação de serviço, porém isto depende diretamente do estado de congestionamento da rede em questão.

PALAVRAS-CHAVES

Modelo TCP, Reno, TS-Prio, QoS, FPA, prioridade, congestionamento.

ÁREAS/SUB-ÁREAS DO CONHECIMENTO

1.00.00.00 - 3 Ciências Exatas e da Terra

1.03.04.04 - 5 Teleinformática

2009

Nº: 503