Janela Móvel Particionada: Proposta de um Novo Mecanismo de Policiamento de Tráfego

Marcio Faerman[1] *Célio Vinicius N. de Albuquerque*[1]

Otto Carlos M. B. Duarte

Grupo de Teleinformática e Automação - GTA
COPPE/EE - Prog. de Eng. Elétrica/Dept. de Eletrônica
Universidade Federal do Rio de Janeiro
CP 68504 - CEP 21945-970 Rio de Janeiro, RJ
E-mail: faerman,celio,otto@coe.ufrj.br

Resumo

Este artigo descreve um mecanismo de policiamento de tráfego baseado em janela. A idéia chave do mecanismo consiste na agregação do processamento dos eventos de chegada. Suas principais características são a simplicidade, a baixa complexidade de implementação, a economia de memória e a utilização de um único temporizador de baixa precisão. Sua flexibilidade de parametrização permite obter uma faixa de desempenho que varia dos resultados obtidos para o mecanismo de janela saltitante até os obtidos pelo mecanismo de janela deslizante. Resultados de simulação e de implementação considerando tráfegos geométricos e do tipo rajada/silêncio são fornecidos para comparação dos mecanismos. Os mecanismos foram implementados em ambiente UNIX e DOS utilizando uma arquitetura de implementação que privilegia o desempenho. Os resultados obtidos comprovam a eficácia do mecanismo proposto.

Abstract

This paper describes a window based policing mechanism. The mechanism key idea consists in aggregating the processing of arrival events. Its main characteristics are simplicity, low implementation complexity, memory economy and use of a single low precision timer. Its performance behavior ranges from jumping window to sliding window results. Simulation and implementation results considering geometric and bursty traffics are obtained for mechanism comparison. The mechanisms were implemented in UNIX and DOS environment using an implementation architecture that privileges performance. The results confirm proposed mechanism efficiency.

[1] Alunos de Mestrado do Programa de Engenharia Elétrica da COPPE/UFRJ.

1 Introdução

A Rede Digital de Serviços Integrados em Banda Larga (RDSI-BL) tem como objetivo ser uma rede de propósito geral que suporte serviços de tempo real, conversacionais, de distribuição, orientados a conexão, não orientados a conexão e que atenda uma ampla faixa de taxa de transmissão, com tráfegos de taxa constante ou variável. Esta rede tem como desafio não só atender às aplicações atuais como também aplicações futuras que ainda não foram projetadas. Conseqüentemente, a flexibilidade deve ser privilegiada para atender este imenso desafio.

A RDSI-BL deve atender de forma eficiente ao maior número possível de usuários e garantir a veiculação do tráfego contratado. Parâmetros de Qualidade de Serviço (*Quality of Service* - QoS), como taxa de perda de células, atraso, variação do atraso, etc., devem ser definidos, monitorados e garantidos pela rede.

Nos últimos anos muito se caminhou, porém vários problemas restam em aberto a procura de uma solução satisfatória. O controle de tráfego da RDSI-BL é sem dúvida um dos problemas mais complexos a ser resolvido.

A fim de garantir um bom desempenho da RDSI-BL é necessário um conjunto de mecanismos de controle de tráfego. Vários controles são previstos [1] tais como: controle de admissão de conexão, controle de prioridade, controle de congestionamento e controle de parâmetros de utilização/rede.

O controle de admissão de conexão diz respeito a aceitação ou não de uma nova conexão pela rede. A decisão de aceitação é baseada na existência de recursos suficientes para satisfazer a qualidade de serviço requerida sem haver alteração da qualidade de serviço das conexões já existentes. Já o controle de prioridade refere-se às diferentes ações tomadas pela rede no processamento do fluxo de uma conexão que contém células de diferentes prioridades indicadas no cabeçalho através do *bit* prioridade de perda de célula. Enquanto que o controle de congestionamento lida com o estado dos elementos da rede de forma que uma sobrecarga de tráfego não altere a qualidade de serviço "contratada" pelas conexões já estabelecidas.

Este trabalho concentra-se no controle de parâmetros de utilização/rede que refere às ações efetuadas pela rede, durante a fase de transferência de informação, a fim de monitorar e controlar o tráfego do usuário. O principal objetivo deste controle é a proteção dos recursos da rede contra tráfegos "mal comportados" intencionais ou não. Estes tráfegos são aqueles que violam os parâmetros negociados durante a fase de estabelecimento de conexão podendo inclusive alterar a qualidade de serviço de outras conexões. Os mecanismos usados por este controle têm a função de detectar as possíveis violações dos parâmetros contratados pela conexão. Os mecanismos que executam esta função também são conhecidos como mecanismos de policiamento.

Vários mecanismos de policiamento têm sido propostos. Entre os mais citados na literatura tem-se o balde furado (*Leaky Bucket*) [2,3] e suas extensões [4], o contador de pico [5] e os mecanismos baseados em janela. Entre estes últimos destacam-se a janela saltitante, janela deslizante e média móvel exponencialmente ponderada [2,5,6].

Dentre os mecanismos de policiamento, o que apresenta concomitantemente vantagens referentes à memorização do tráfego da fonte, ao controle da taxa média, além da limitação do comprimento da rajada, é a janela deslizante. Porém este mecanismo tem sido alvo de críticas referentes à sua complexidade de implementação [2,5]. Já Dittmann *et al* [6] apresenta uma proposta de implementação para a janela deslizante, onde defende um nível de complexidade não tão crítico.

Este artigo propõe e analisa o mecanismo de janela móvel particionada, um mecanismo de policiamento baseado em janela que requer uma complexidade de implementação inferior ao mecanismo de janela deslizante e que é tão restritivo quanto este em relação à duração das rajadas. Este mecanismo também apresenta a vantagem de ser extremamente flexível. De acordo com a demanda do contrato entre rede e usuário, a configuração da janela móvel particionada mapeia seu algoritmo desde a simplicidade da janela saltitante até o amplo conjunto de vantagens oferecido pela janela deslizante.

Na seção 2 os mecanismos de janela saltitante e deslizante são descritos assim como seus tráfegos de pior caso. A seção 3 propõe o mecanismo janela móvel particionada e identifica o seu tráfego de pior caso. A seção 4 descreve os modelos de simulação e os procedimentos da implementação dos mecanismos. A seção 5 apresenta os resultados numéricos obtidos na simulação e na implementação e realiza uma análise comparativa dos mecanismos. Finalmente as conclusões são apresentadas na seção 6.

2 Mecanismos de Policiamento Baseados em Janela

Os mecanismos baseados em janela consistem em limitar, em um determinado valor *C* (crédito máximo), o número de células injetadas na rede durante um intervalo de tempo *T*. Este intervalo de tempo é denominado **janela**. Os mecanismos baseados em janela diferem entre si segundo a maneira pela qual as janelas evoluem ao longo do tempo e pela forma como o valor dos créditos é considerado.

A parcela de tráfego que em uma determinada janela excede o valor do crédito máximo é chamada tráfego violador e as células que a compõem são simplesmente descartadas ou "marcadas". Nesta situação, as células marcadas são transferidas caso não sejam causadoras de congestionamento para a rede, com possível sobretaxa para o usuário. Este trabalho considera o tráfego submetido a rede como sendo o tráfego não violador.

Entre os mecanismos baseados em janela mais citados na literatura têm-se: Janela Saltitante, Janela Deslizante ou Retangular, Janela Saltitante Sincronizada, Janela Triangular, Janela com Média Móvel Exponencialmente Ponderada. Apenas os dois primeiros mecanismos serão abordados neste trabalho para efeito de comparação com o mecanismo proposto.

Janela Saltitante

O mecanismo Janela Saltitante - JS (*Jumping Window* **-** JW) considera janelas com períodos de tempo fixos e iguais a *T* justapostas ao longo do tempo. Assim, uma janela tem

início imediatamente após o término da janela anterior. Durante uma janela somente *C* células poderão ser submetidas a rede. Caso *C* células já tenham sido recebidas pela rede durante uma janela, as células seguintes serão consideradas violadoras e, conseqüentemente, descartadas ou marcadas.

Este mecanismo é bastante simples de ser implementado através de um temporizador e de um contador. As janelas justapostas deste mecanismo são obtidas pelo temporizador na sua configuração auto-disparável, ou seja, que torna a disparar no momento que estoura. O contador deve ser inicializado com o valor *C* do crédito máximo e decrementado de uma unidade a cada célula que é injetada na rede (figura 1).

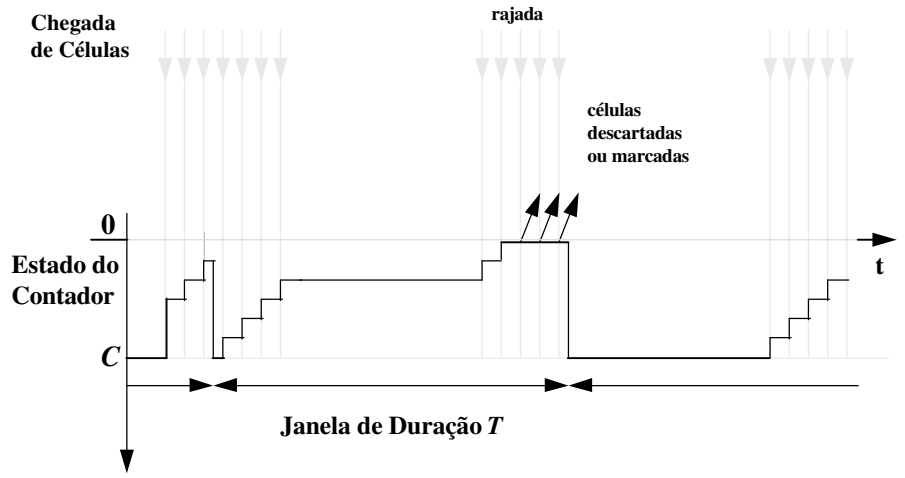


Figura 1: O mecanismo de janela saltitante.

Um padrão de tráfego bastante utilizado na análise dos mecanismos de policiamento é o denominado tráfego de pior caso. Define-se como tráfego de pior caso, o tráfego que passa pelo mecanismo de policiamento, explorando totalmente a taxa policiada sem sofrer qualquer tipo de penalização e que possui características que dificultem sua veiculação pela rede. A condição mais severa para a rede é durante as rajadas pois dependendo do estado da rede as células poderão ser armazenadas (provocando atrasos) e até mesmo perdidas por transbordo de memória. Assim, pode-se concluir que o tráfego de pior caso é do tipo rajada/silêncio onde o comprimento da rajada é o maior possível.

Se a fonte estiver sincronizada com a janela do mecanismo janela saltitante - JS, ela pode gerar duas rajadas concatenadas, de *C* células, ao fim de um intervalo e no início do intervalo seguinte. Um período de silêncio de $2T$ - $(2C \times t_{cel})$ será necessário, onde t_{cel} denota o intervalo entre chegadas de células, durante a rajada. Portanto, o ciclo de ocupação para o JS tem duração de $2T$ (figura 2).

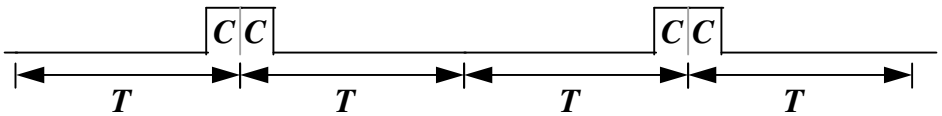


Figura 2: Tráfego de pior caso para o mecanismo janela saltitante.

Janela Deslizante

O mecanismo de Janela Deslizante - JD (*Sliding Window* - SW) também limita em *C* o número máximo de células submetidas a rede durante uma janela com duração *T*. A diferença consiste na forma como são consideradas as janelas. O início de uma janela está associado ao instante de chegada de cada célula e seu término exatamente *T* unidades de tempo depois. Assim, várias janelas superpostas vão sendo abertas e fechadas ao longo do tempo e todas restringindo o tráfego em *C* células no máximo, em qualquer período de tempo *T*. A figura 3 ilustra o comportamento deste mecanismo. A superposição das janelas produz um efeito de "memorização" do tráfego.

Uma outra forma de vizualizar este mecanismo é ao invés de se considerar várias janelas, considerar uma única janela que desliza ao longo do tempo memorizando os instantes de chegada das células. Esta memorização é realizada através de um contador que é decrementado (memoriza) no instante da chegada de uma célula e incrementado (apaga da memória) *T* unidades de tempo após.

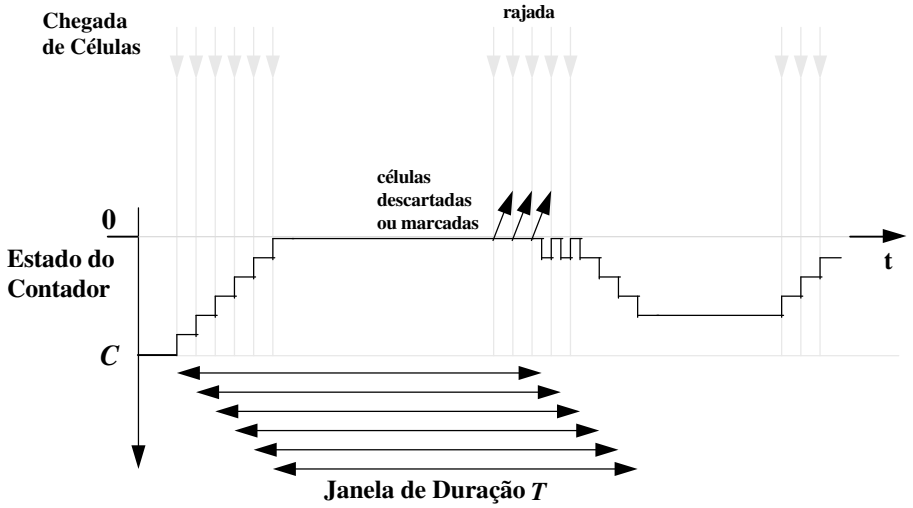


Figura 3 - O mecanismo de janela deslizante.

Pela comportamento deste mecanismo, o número de células submetido a rede durante qualquer intervalo de tempo de duração *T* está limitado ao crédito máximo *C*. Portanto, o mecanismo janela deslizante apresenta a vantagem de limitar a taxa policiada do tráfego durante **qualquer** intervalo de duração *T* ao valor *C*/*T*. Neste mecanismo o tráfego de pior caso é dado por uma rajada de comprimento *C*, seguida por um período de silêncio cuja duração é $T - (C \times t_{cel})$. como ilustrado na figura 4.

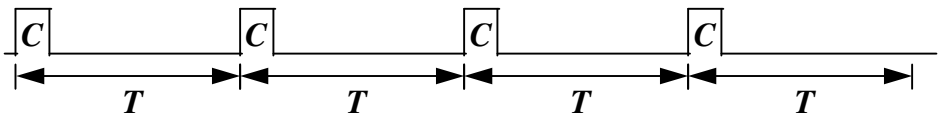


Figura 4: Tráfego de pior caso para o mecanismo janela deslizante.

O JD também apresenta a vantagem de memorizar o comportamento do tráfego durante um determinado tempo. No entanto, sua complexidade de implementação é

relativamente alta, tendo em vista a necessidade do armazenamento dos instantes de chegadas das células, disparando um temporizador para cada chegada de célula, além de armazenar o número de chegadas, de até *C* células. Isto pode ser extremamente custoso para janelas com intervalo de longa duração. É possível simplificar o esquema de armazenamento dos instantes de chegadas a partir da associação de um *bit* para cada posição de *T* onde a célula pode ter chegado. Porém para grandes valores de *C*, este esquema também apresentará um custo elevado.

3 O Mecanismo Janela Móvel Particionada

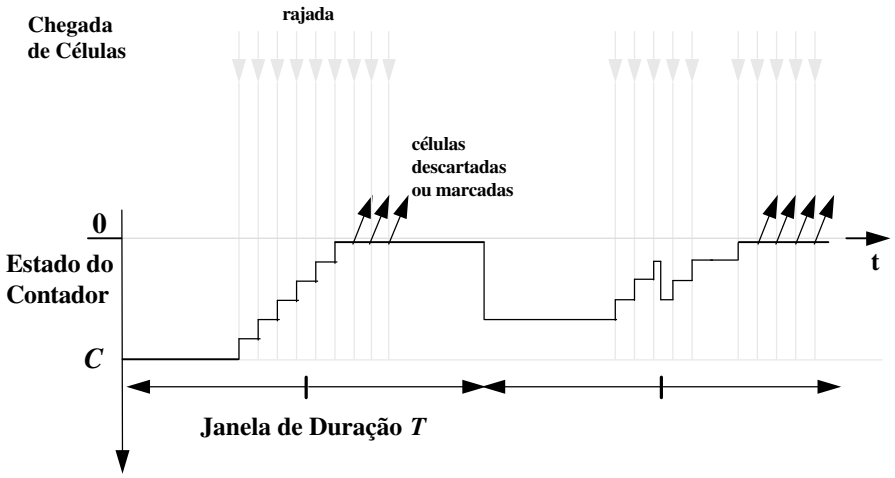
O mecansimo Janela Móvel Particionada - JMP, proposto nesta seção, busca um compromisso entre a simplicidade da janela saltitante e a eficiência da janela deslizante em relação a memorização do tráfego, policiamento acurado da taxa média e restrição a comprimento de rajadas.

Um dos principais fatores que leva o mecanismo de janela deslizante a demandar uma grande complexidade de implementação é o armazenamento do instante da chegada de uma célula. Por outro lado, as janelas justapostas do mecanismo de janela saltitante podem ser simplesmente implementadas por um temporizador auto-disparável.

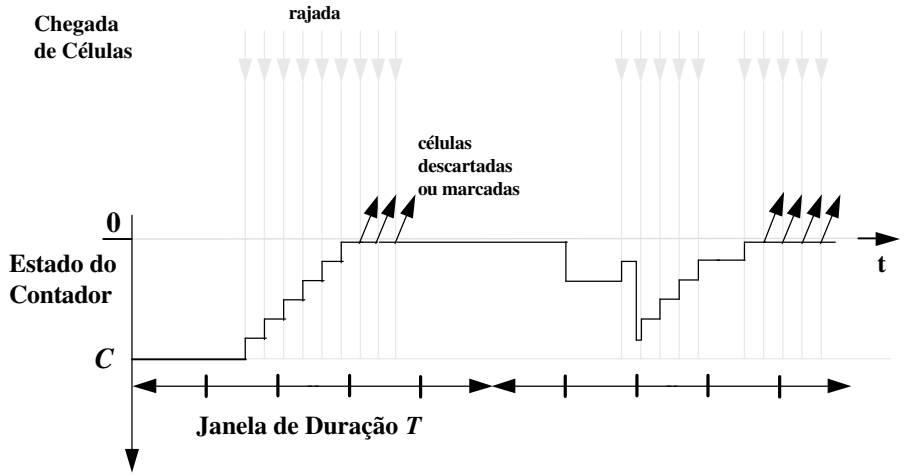
Para se buscar a eficiência em relação a restrição do comprimento máximo da rajada é necessário "memorizar" informações ou em outras palavras superpor janelas.

Neste mecanismo a janela de duração *T* é subdividida em *P* períodos de tempo iguais, denominados partições. Analogamente ao JS e JD, a cada chegada de célula um contador, estando abaixo do limite de créditos *C*, é decrementado. *T* unidades de tempo após o início de uma partição p_i, o contador é incrementado do número de células que chegaram durante esta partição p_i. Assim, o mecanismo janela móvel particionada é baseado na idéia de "janelas sobrepostas que saltam, de partição em partição" ao invés de deslizarem.

Observa-se que para o número de partições *P* igual a um, este mecanismo é equivalente à janela saltitante. Quando *P* tende a infinito, a duração de cada partição tende a zero e este mecanismo aproxima-se do mecanismo janela deslizante.



(a)



(b)

Figura 5: O mecanismo janela móvel particionada para a) 2 partições e b) 5 partições.

O tráfego de pior caso é gerado por rajadas de comprimento *C*. As rajadas são sincronizadas de forma alternada com as partições. O final de uma rajada sincroniza com o final de uma partição, enquanto que a rajada posterior começa no mesmo instante do início da partição que ocupa. Este padrão se repete periodicamente. Quando a duração da rajada é menor que a duração da partição, o período de silêncio mínimo entre as rajadas tem duração igual a P - 1 partições. Como cada partição dura T/P unidades de tempo, a duração do período mínimo de silêncio, nesta situação , pode ser expressa como:

$$\frac{(P-1)\times T}{P}.$$

Assim como na janela deslizante este mecanismo limita o comprimento da rajada a C células enquanto na janela saltitante o comprimento máximo da rajada é de $2C$ células. No entanto, o mecanismo não consegue mais manter a característica da janela deslizante onde em qualquer período de tempo igual a T apenas C células serão submetidas a rede. Como na janela saltitante pode-se chegar até a $2C$ células em período de tempo T mas elas estarão espaçadas de um período de tempo no mínimo igual a uma partição que no pior caso será de $T/2$ para duas partições. Pode-se constatar que desde a sua configuração mais simples, a partir de duas partições, o mecanismo janela móvel particionada já é tão restritivo quanto o mecanismo janela deslizante em termos de comprimento da rajada, limitando-a a C células.

Para um aumento no número de partições o intervalo mínimo entre as rajadas também aumenta. Já para P tendendo a infinito, a duração das partições tende a zero e cada célula de uma rajada chega em uma partição diferente. Nesta situação, o padrão do tráfego de pior caso da janela móvel particionada tende ao padrão do tráfego de pior caso da janela deslizante.

O gerenciador de qualidade de serviço é responsável por parametrizar o número de partições deste mecanismo, de acordo com os requisitos de QoS estabelecidos.

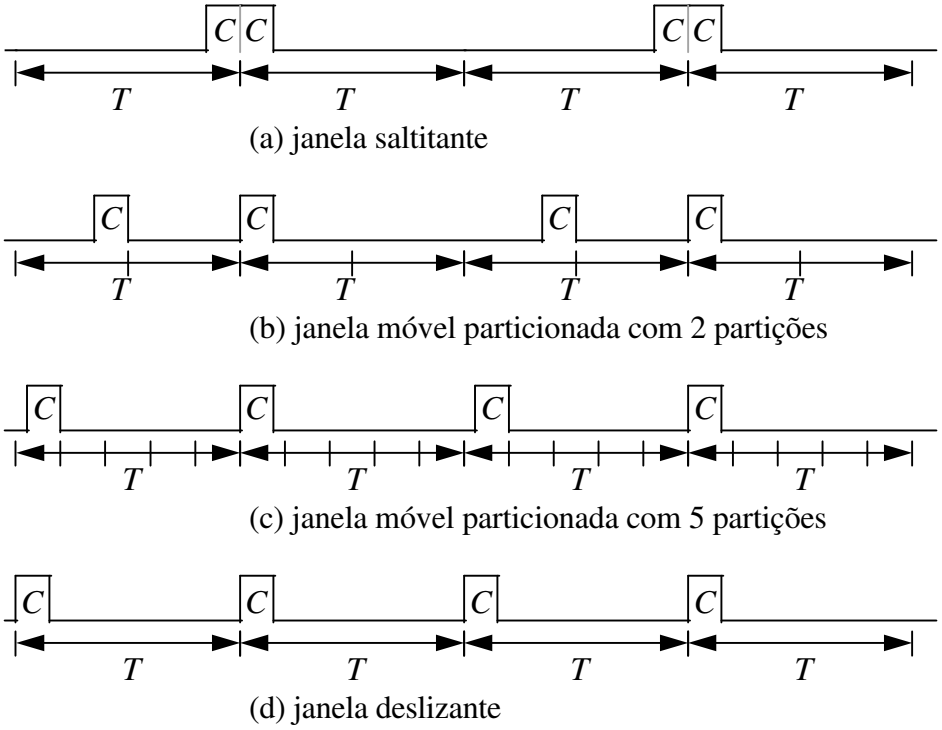


Figura 6: Comparação do tráfego de pior caso para mecanismo janela móvel particionada

4 Simulação e Implementação

Esta seção apresenta descrições estruturadas da simulação e da implementação dos mecanismos de policiamento (MP) considerados. Para a simulação dos MPs fez-se uso do software *Queueing Networks Analysis Package* - QNAP o qual é baseado em redes de filas utilizando um intervalo de confiança de 0.95. Desta forma cada MP é modelado por elementos de redes de filas conforme descrito a seguir.

A janela saltitante foi modelada através de uma fila com tempo de serviço constante e igual a *T* para gerar as janelas justapostas. A fila pedágio verifica se cada célula gerada pela fonte é ou não violadora, de acordo com o valor do contador do mecanismo (figura 7).

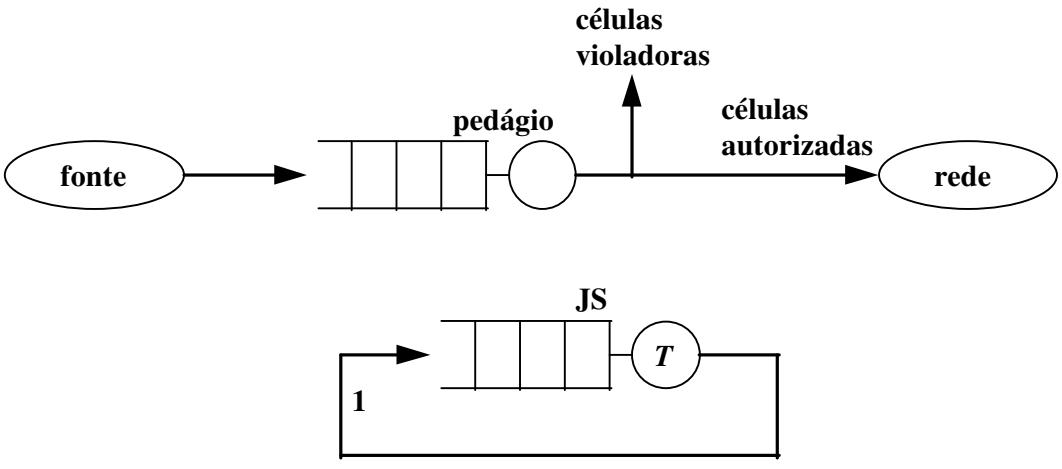


Figura 7: Modelo de simulação da janela saltitante.

A janela deslizante foi modelada (figura 8) como um sistema de descarte com múltiplos servidores, onde os tempos de serviço são determinísticos com duração *T*. O número de servidores é igual ao limite máximo *C* do contador.

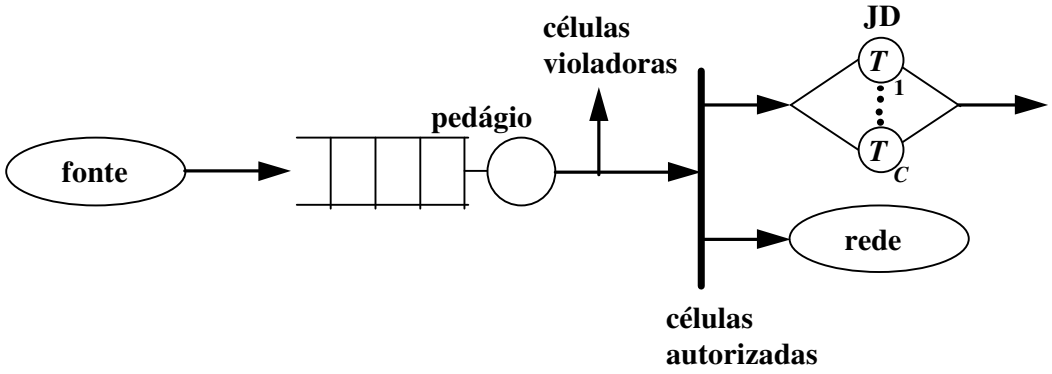


Figura 8: Modelo de simulação da janela deslizante.

Para simular a janela móvel particionada (figura 9) utilizou-se uma fila com um servidor com tempo de serviço determinístico igual a *T*/*P*. Foi utilizado um "vetor FIFO" para controle do mecanismo. Ao final de cada intervalo de tempo *T*/*P* o contador do mecanismo é incrementado do conteúdo da última posição do vetor FIFO. Neste instante, o vetor é deslocado de uma posição, e na primeira posição é armazenado o número de células que passaram pelo mecanismo durante a última partição. A fila pedágio verifica a condição de violação das células de acordo com o valor do contador, como nos mecanismos anteriores.

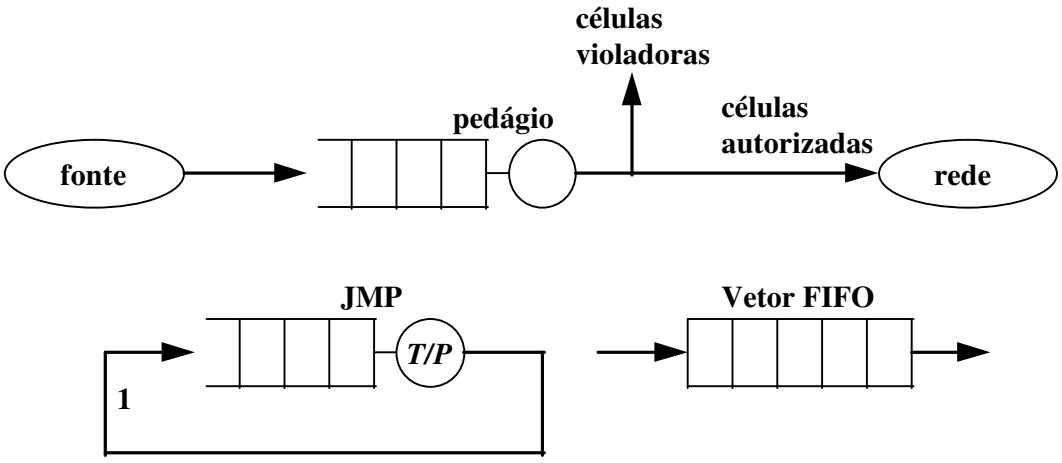


Figura 9: Modelo de simulação da janela móvel particionada.

A implementação dos mecanismos de policiamento fez uso de um ambiente de desenvolvimento de sistemas de comunicação de alto desempenho com garantias de qualidade de serviço proposto em [7-10]. Este ambiente define módulos otimizados de gerenciamento de memória, gerenciamento de temporizações, escalonamento de tarefas, além de estruturas de dados para as interfaces entre os módulos, a aplicação, os protocolos do sistema de comunicação e o meio de comunicação. Os mecanismos de policiamento foram implementados em linguagem C nos ambientes UNIX e DOS.

O mecanismo janela saltitante é implementado de acordo com o seguinte procedimento. Após o estabelecimento da conexão e a determinação dos parâmetros do mecanismo, o JS cria um temporizador auto-disparável responsável pela atualização do credito a cada *T* unidades de tempo. Paralelamente, a cada célula entregue ao JS, este testa a variável crédito e se o resultado for positivo, o mecanismo decrementa os créditos e entrega a célula para ser transmitida, multiplexada com as demais conexões; caso contrário, a célula é considerada violadora e neste caso, optou-se por descartar a célula.

O mecanismo de janela deslizante pode ser implementado por meio de diversos temporizadores e por um contador de créditos. A cada célula submetida a rede, se o contador de créditos for positivo o mecanismo decrementa o contador e dispara um temporizador com duração *T* unidades de tempo. No estouro de cada temporizador (final da janela) a variável crédito é incrementada. Este esquema será referenciados neste trabalho como janela deslizante contínua. Esta implementação apresenta o inconveniente de requerer um grande e variável número de temporizadores implicando em uma maior utilização de memória e de CPU, além de dificultar uma reserva eficiente de recursos para a conexão.

Dittmann *et al* [6] propõem uma outra forma de implementação consistindo na pré-alocação de um registrador de deslocamento no qual cada *bit* seria responsável por registrar a chegada de uma célula na unidade de tempo correspondente. Este esquema será referenciado como janela deslizante discretizada. Para que se permita a chegada de apenas uma célula por unidade de tempo é necessário que a unidade de tempo tenha a duração igual ao tamanho da célula dividido pela taxa máxima de transmissão da fonte, por exemplo 2,7 μs para 155 Mbps.

O mecanismo janela móvel particionada apresentado neste trabalho foi implementado por meio de um vetor FIFO de tamanho fixo alocado no estabelecimento da conexão de acordo com os parâmetros do tráfego e do mecanismo. Diferentemente dos outros mecanismos baseados em janela, além dos parâmetros *T* e *C*, este mecanismo define o parâmetro *P* que indica o número de partições em cada janela de comprimento *T*. Este parâmetro é extremamente importante na definição da utilização da memória e da CPU. Quanto maior o número de partições *P*, maior será a memória necessária para alocação do vetor FIFO, maior será a utilização da CPU e mais próximo este mecanismo estará do JD. Quanto menor o número de partições, menor será a memória necessária, menor será a utilização da CPU e mais próximo estará do mecanismo JS. Ao se completar a fase de estabelecimento de conexão, aloca-se o vetor FIFO com tamanho *P* x nb, onde nb é o número de bits necessário para representar o número máximo de células que podem ser transmitidas em uma partição. Analiticamente este valor é:

$$M = P \times nb;$$

onde:

$$nb = \log_2 \left\lceil T \Big/ \left(P \times t_{cel} \right) \right\rceil.$$

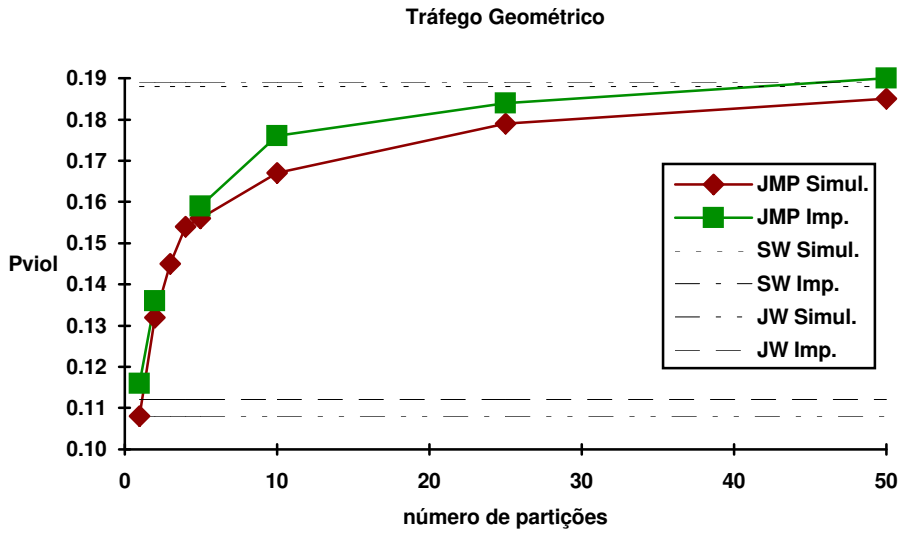
O temporizador é responsável pela atualização do crédito a cada *T/P* unidades de tempo. Esta atualização corresponde ao incrementado do crédito do número de células que chegaram durante a partição p_i iniciada *T* unidades de tempo antes desta atualização. Este valor está armazenado na última posição do vetor FIFO.

5 Comparação dos Mecanismos

Esta seção apresenta resultados quantitativos e qualitativos, obtidos na simulação e implementação, comparando os mecanismos abordados. A comparação enfoca o número de temporizadores e a sua precisão, a máxima duração da rajada, a memória necessária e a probabilidade de violação.

Para a computação da probabilidade de violação (p_{viol}) dos mecanismos, tanto na simulação quanto na implementação, utilizou-se dois padrões de tráfego: um tráfego geométrico onde a média do intervalo entre as chegadas de células é igual a cinco unidades de tempo e um tráfego rajada/silêncio. Neste último, o tamanho da rajada obedeceu a uma distribuição geométrica com média 10, onde a taxa de transmissão da rajada é igual a 1 célula/unidade de tempo e o período de silêncio seguiu uma distribuição geométrica com média 40 unidades de tempo. Portanto, este tráfego apresenta nível de rajada igual a 5 e, como o tráfego anterior, taxa média igual a 1/5 células/unidade de tempo. Para todos os mecanismos foram arbitrados os parâmetros $T = 50$ e $C = 10$, representando uma taxa média policiada proporcional a 1/5 células/unidade de tempo.

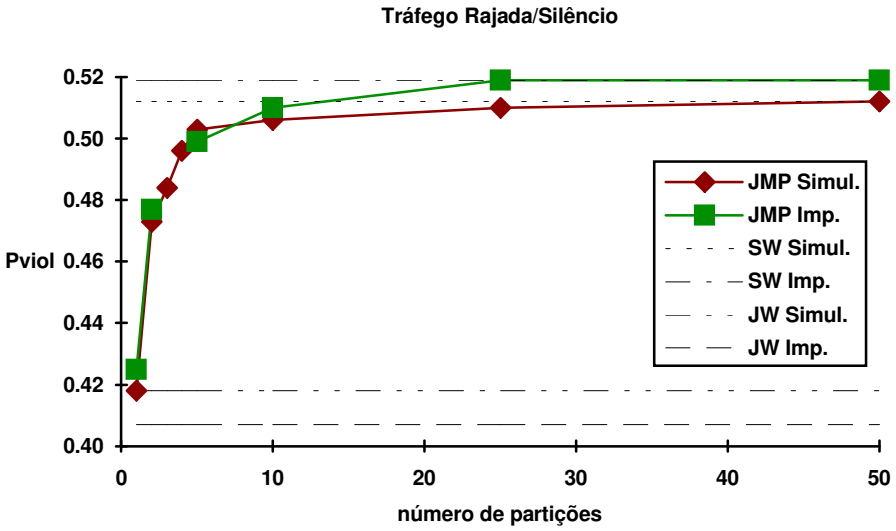
Gráfico 1: Probabilidades de violação da JMP em função do número de partições para o tráfego geométrico.



O gráfico 1 mostra as probabilidades de violação do mecanismo janela móvel particionada em função do número de partições computadas, quando o tráfego utilizado foi o geométrico, e o gráfico 2, o rajada/silêncio. Os limites inferiores e superiores dos gráficos são as p_{viol} obtidas para os mecanismos janela saltitante e deslizante, respectivamente. Em ambos os gráficos é possível constatar a semelhança entre os resultados obtidos pela simulação e pela implementação, sendo os resultados da simulação mais precisos que os da implementação, pois foram simuladas cem vezes mais amostras de tráfego que a quantidade utilizada na implementação. Pode-se observar que para o número de partições P igual a um, a probabilidade de violação do mecanismo janela móvel particionada é a mesma do mecanismo janela saltitante, e a medida que o número de partições aumenta, a probabilidade de violação aproxima-se rapidamente da probabilidade de violação do mecanismo janela deslizante. No caso do tráfego rajada/silêncio a p_{viol} aproxima-se mais rapidamente, ou seja, a partir de um número menor de partições, de seu valor final (p_{viol} da janela deslizante) que no caso do

tráfego geométrico. No entanto as probabilidades de violação no caso do tráfego rajada/silêncio para todos os mecanismos é bastante superior às p_{viol} obtidas com o tráfego geométrico.

Gráfico 2: Probabilidades de violação da JMP em função do número de partições para o tráfego rajada/silêncio.



A tabela 1 contém informações comparativas entre os mecanismos. A notação $\{x\}$ representa o número de *bits* necessário para representar o valor x. Com relação ao número de temporizadores, observa-se que o mecanismo janela deslizante contínua necessita do maior valor. Deve-se ressaltar que quanto maior o número de temporizadores, maior será a utilização da CPU. Observa-se da tabela a característica do mecanismo janela saltitante permitir uma duração máxima da rajada igual a $2C$.

Tabela 1: Comparação entre os mecanismos.

Mecanismo	JS	JD contínua	JD discretizada	**JMP**
Número de Temporizadores	1	C	1	**1**
Máxima Duração da Rajada	$2C$	C	C	$\boldsymbol{C}$
Precisão dos Temporizadores	baixa (taxa policiada}	alta (ilimitada)	alta (taxa max. fonte)	**ajustável em toda faixa**
Memória para Temporizadores	{Tmp.}	$C \times$ {Tmp.}	{Tmp.}	**{Tmp.}**
Memória para Contadores e Parâmetros	$\{C\} + \{C_{max}\}$	$\{C\} + \{C_{max}\}$	$\{C\} + \{C_{max}\}$	$\boldsymbol{\{C\} + \{C_{max}\} + \{P\}}$
Memória de Controle	———	———	$\left\lceil \frac{T}{t_{cel}} \right\rceil$	$\boldsymbol{P \times \left\{ T / \left(P \times t_{cel} \right) \right\}}$

Um fator importante a ser considerado é a precisão requerida pelos temporizadores. O mecanismo janela deslizante contínua necessita de temporizadores de altíssima precisão, pois eles devem ser disparados na chegada de cada célula, ou seja, em qualquer instante de tempo. A janela deslizante discretizada já é um pouco mais factível, porém há ainda a necessidade de uma precisão bastante alta, por exemplo 2.7μs para a taxa de transmissão de 155 Mbps. O mecanismo janela saltitante é o que necessita de temporizadores de menor precisão, ou seja, da ordem do tamanho da janela *T*. O mecanismo JMP permite a utilização de temporizadores com toda a faixa de precisão, desde a utilizada na JS até a utilizada na JD, conforme o número de partições utilizadas.

Um outro fator bastante importante na análise da complexidade de implementação é a quantidade de memória necessária para cada mecanismo. As duas variações da janela deslizante são comparáveis em relação a quantidade de memória necessária. A janela saltitante é a que necessita da menor quantidade de memória. Já a janela móvel particionada necessita de uma quantidade de memória variável de acordo com o número de partições, porém sempre inferior a quantidade necessária para a janela deslizante. Por exemplo, o caso da janela deslizante discretizada com $T = 60 \times t_{cel}$ e $C = 30$. Neste caso ela necessita de 60 *bits* para memorizar o tráfego de uma janela. Se utilizarmos a janela móvel particionada com 20 partições, em cada partição poderão chegar no máximo 3 células, representadas por 2 *bits*. Portanto o número total de *bits* para representar o vetor FIFO será de um total de 40 bits. Para um JMP com 10 partições, 30 bits são necessários, e com 5 partições, 20 *bits*. Observa-se no gráfico 2 que para 5 partições, o comportamento da janela móvel particionada já está bastante próximo da janela deslizante.

6 Conclusão

Este trabalho apresentou o mecanismo janela móvel particionada. A idéia chave do mecanismo consiste na divisão da janela em intervalos de tempo iguais. Esta discretização do intervalo de aferição do tráfego permite agregar o processamento dos eventos de chegada durante cada partição. A funcionalidade e a complexidade deste mecanismo variam desde a janela saltitante (mecanismo proposto com $P = 1$) até a janela deslizante, de acordo com o número de partições utilizadas.

Dentre as qualidades importantes do mecanismo é possível citar a memorização do tráfego, o policiamento acurado da taxa média e a restrição do tamanho máximo da rajada a *C* células, além da baixa complexidade de implementação e flexibilidade na precisão do temporizador utilizado. O tráfego de pior caso deste mecanismo pode chegar até a $2C$ células em período de tempo *T* mas elas estarão espaçadas de um período de tempo no mínimo igual a uma partição, que no pior caso será de $T/2$ para duas partições.

Resultados de simulação e de implementação obtidos para padrões de tráfego geométrico e rajada/silêncio comprovam que mesmo com um número pequeno de partições ($P = 5$) obtém-se uma funcionalidade bastante próxima da janela deslizante, com uma baixa complexidade de implementação e uma economia considerável de memória e processamento.

Para o exemplo citado na seção 5, através deste mecanismo, com um único temporizador de precisão *T*/5, limita-se o comprimento da rajada em *C* células, obtém-se uma economia de 66% de memória em relação a janela deslizante discretizada e consegue-se um resultado bastante próximo deste mecanismo. Em termos de probabilidade de violação os resultados para tráfego rajada/silêncio apresentam uma diferença relativa de apenas 4 %.

7 Agradecimentos

Os autores agradecem a colaboração dos professores Bruno Baynat e Kim Thai da Universidade Paris VI que durante a missão CAPES/COFECUB muito ajudaram no trabalho de simulação.

8 Referências

[1] ITU-T Recomendação I.311, B-ISDN General Networks Aspects, Genebra, 1991.

[2] E. Rathgeb, "Modeling and Performance Comparision of Policing Mechanisms for ATM Networks", IEEE JSAC, pp.325-334, vol. 9, no. 3, abril de 1991.

[3] M. Butto, E. Cavallero, A. Tonietti, "Effectiveness of the "Leaky Bucket" Policing Mechanism in ATM Networks", IEEE JSAC, pp. 335-342, vol. 9, no. 3, abril de 1991.

[4] A. Berger,"Performance Analysis of a Rate Control Throttle Where Tokens and Jobs Queue", IEEE INFOCOM, pp. 30-38, 1990.

[5] J. Junior, J. Monteiro, "On The Efficiency of Policing Mechanisms for ATM Networks", IEEE ITS, pp.470-474, agosto de 1994.

[6] L. Dittman, S. Jacobsen, K. Moth, "Flow Enforcement Algorithms for ATM Networks", IEEE JSAC, pp. 343-349, vol 9. no. 3, abril de 1991.

[7] C. V. Albuquerque, M. L. Lanza, O. C. Duarte, "Sistemas de Comunicação Abertos em Redes Locais: Uma Implementação e Medidas de Desempenho em Ambientes de Computadores Pessoais", XIII Congresso Sociedade Brasileira de Computação (SEMISH), pp. 327-329, Florianópolis, SC, Brasil, setembro de 1993.

[8] C. V. Albuquerque, M. D. Nunes, O. C. Duarte, "Performance Measurements in a Manufacturing Communication System", IEEE International Conference on Circuits and Systems ISCAS' 94, pp.3.69-3.72, Londres, Inglaterra, maio de 1994.

[9] L. F. Baginsky e O. C. Duarte, "Um Modelo de Implementação de Alto Desempenho para Protocolos de Comunicações Estruturados em Camadas", Revista da Sociedade Brasileira de Computação", vol. 6, pp. 46-56, abril de 1991.

[10] L. F. Baginsky e O. C. Duarte, "Um Modelo de Implementação de Alto Desempenho para Sistemas Abertos", IX Congresso da Sociedade Brasileira de Telecomunicações, São Paulo, pp. 19.3.1-19.3.5, setembro de 1991.