Capítulo IV

Implementação de Mecanismos de Policiamento

IV.1 Introdução

Neste capítulo são descritas propostas de implementação [AFD95a, AFD95b] dos mecanismos de policiamento considerados neste trabalho. Estas implementações foram realizadas utilizando os módulos básicos da Arquitetura GTA/MM propostos em [Alb95]. São também apresentados resultados experimentais obtidos a partir de simulação e implementação dos mecanismos baseados em janela. A partir de tais resultados, aprofunda-se a análise do mecanismo janela móvel particionada proposto no capítulo III.

A Arquitetura GTA/MM consiste de três módulos básicos utilizados no desenvolvimento de todo sistema de comunicação e de alguns módulos específicos, opcionais, que definem as características do sistema de comunicação a ser desenvolvido. Os módulos básicos são o Módulo de Gerenciamento de Memória (GM), o Módulo de Escalonamento de Tarefas (ET) e do Módulo de Gerenciamento de Temporizações (GT). Os módulos específicos são o Módulo de Gerenciamento do Sistema, o Protocolo de Comunicação, o Módulo de Sincronização e o Módulo de Garantia de Qualidade de Serviço que engloba o Módulo de Controle de Tráfego e Policiamento.

Os módulos específicos visam oferecer um conjunto de facilidades para a implementação do sistema. O Módulo de Gerenciamento do Sistema configura os demais módulos de acordo com as características do sistema desejado. Os Protocolos de Comunicação visam oferecer o conjunto de funcionalidades existentes nos principais

protocolos de alta velocidade [NDu94, Nun95, VDP95]. As funcionalidades são configuráveis ou por primitivas de serviços ou pelo módulo de gerenciamento do sistema. O Módulo de Sincronização visa coordenar a ordenação temporal de informações multimídia (voz, vídeo, gráficos, dados, etc.). Através de Redes de Petri temporizadas [WQG94], este módulo implementa os tipos de sincronização inter e intramídia. O Módulo de Garantia de Qualidade de Serviço possui implementados os mecanismos *Virtual Clock* [Zha91], *Stop-and-Go* [Gol90], *Jitter-Earliest-Due-Date* [VZF91] e *Hierarchical Round Robin* [ZKe91]. Este módulo engloba ainda as implementações dos seguintes mecanismos de policiamento de tráfego: balde furado, balde furado com suavização do tráfego de saída, janela saltitante, janela saltitante sincronizada, janela deslizante contínua e janela móvel particionada [AFD95a].

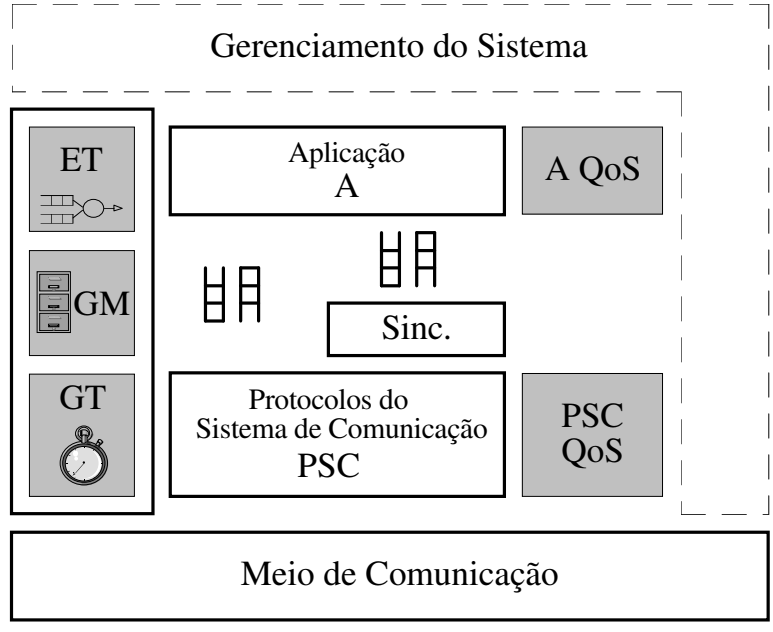


Figura IV.1: A arquitetura de implementação GTA/MM.

Na seção IV.2 serão apresentados os módulos básicos da arquitetura GTA/MM. A implementação dos mecanismos de policiamento é descrita na seção IV.3. Finalmente, a seção IV.4 apresenta dados comparativos dos mecanismo de policiamento apresentados neste capítulo.

IV.2 Módulos Básicos

O Módulo de Gerenciamento de Memória, de maneira a evitar excesso de fragmentação, divide a memória em regiões de transmissão e recepção. Este modelo permite um uso mais ordenado da memória tornando o gerenciamento mais rápido e eficiente. Isto é possível porque os fluxos de transmissão e recepção são independentes.

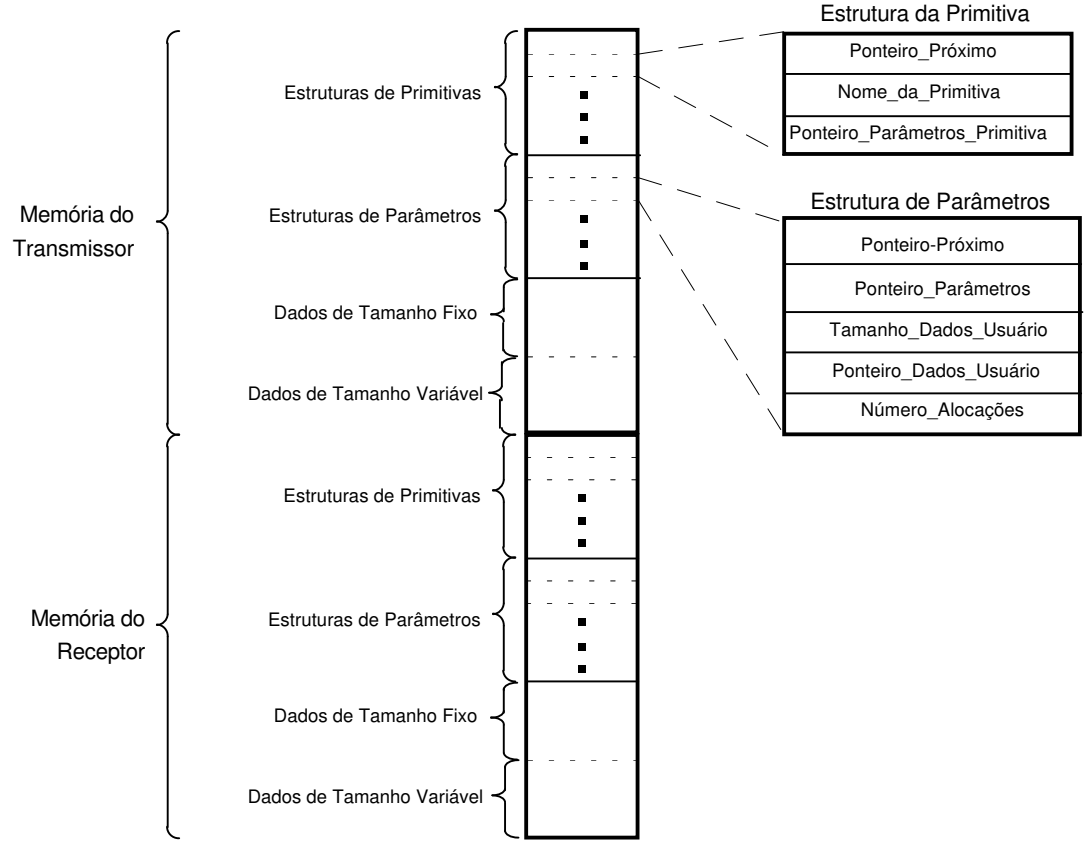
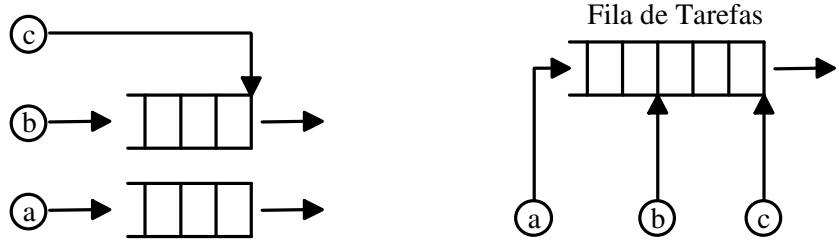


Figura IV.2: Gerenciamento de Memória

Estas regiões são ainda subdivididas em quatro partes. Duas estão reservadas para estruturas de dados de tamanho fixo das interfaces entre camadas adjacentes e entre os módulos básicos. Uma região é alocada para estruturas de tamanho variável usadas na passagem de parâmetros entre camadas de um mesmo sistema. A quarta região de memória

alocada de acordo com cada fluxo estabelecido e contém estruturas de dados de tamanho fixo, denominados blocos, que são usados pelos dados que serão efetivamente transmitidos.

Diferentemente dos gerenciadores de memória de propósito geral, este gerenciador armazena as informações de controle de alocação separado dos dados, permitindo a implementação eficiente e sem cópias de funções de segmentação, remontagem, concatenação, separação, liberação de parte dos dados, entre outras.



Modelo Implementação

a - ponto de entrada das tarefas de menor prioridade.
b - ponto de entrada das tarefas de maior prioridade.
c - ponto de entrada das tarefas de prioridade máxima
e de saída de todas as tarefas.

Figura IV.3: Módulo de escalonamento de tarefas.

O Módulo de Escalonamento de Tarefas armazena e determina a seqüência das tarefas a serem executadas. Além disso, possui funções para permitir níveis de prioridade para os fluxos em tempo real. O comportamento da fila de tarefas é modelado como o de duas filas onde um elemento da fila de menor prioridade só é servido quando a fila de maior prioridade estiver vazia. De forma a priorizar o atendimento de tarefas de altíssima prioridade, como por exemplo o estouro de temporizadores associados à funções de tempo real, estas tarefas são inseridas na primeira posição da fila de tarefas (ponto **c** da Figura IV.3). Este ponto é também o ponto de saída de todas as tarefas. Um segundo nível de prioridade é dado às tarefas dos fluxos em tempo real inseridas no ponto **b** da Figura IV.3. As demais tarefas de menor prioridades são inseridas no final da fila de tarefas (ponto **a** da Figura IV.3). Implementando o

modelo de duas filas com prioridades através de uma única fila com três pontos de entrada e um único ponto de saída evita-se testes a cada retirada da fila para verificar se a fila de maior prioridade está vazia ou não.

O Módulo de Gerenciamento de Temporizações é bastante eficiente. Ele é baseado na proposta de Varghese & Lauck [VLa87], no qual uma analogia a um relógio real é simulada (Figura IV.4) através de um contador em módulo. Esta implementação privilegia o desempenho, evitando testes desnecessários a cada interrupção, permitindo portanto a criação de um grande número de temporizadores dinamicamente.

São fornecidas funções para disparar temporizadores simples e auto-disparáveis, reiniciar, interromper e obviamente sinalizar o estouro de um temporizador. Nesta arquitetura, o estouro de um temporizador é tratado como uma primitiva de serviço, ou seja, a indicação de estouro (evento) é colocada na fila apropriada da interface, e a tarefa correspondente é colocada na fila de prioridade máxima do escalonador de tarefas.

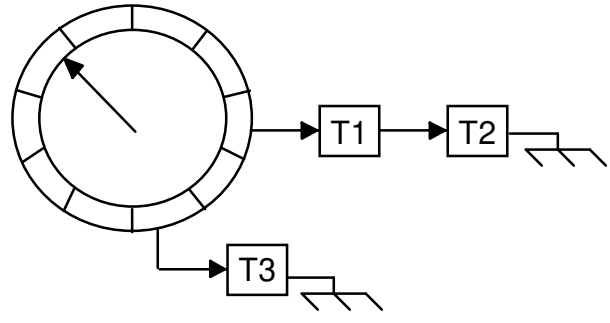


Figura IV.4: Temporização proposta por Varguese & Lauck

Os protocolos de comunicação utilizados atualmente na Arquitetura GTA/MM são os protocolos denominados SC0 e SC2. Estes protocolos correspondem, respectivamente, aos protocolos Transporte classe 0 e classe 2, propostos no Modelo de Referência OSI, que foram alterados para atender as características de protocolos de alta velocidade (*light weighed*

protocols). As funções de estabelecimento de conexão implícita e de controle de fluxo por taxa de transmissão foram inseridas e por outro lado as funções de concatenação/separação (raramente usadas mas obrigatórias) foram retiradas.

IV.3 Implementação dos Mecanismos de Policiamento

IV.3.1 Balde Furado

O mecanismo balde furado (*leaky-bucket*) é implementado por meio de um contador de créditos e de um temporizador auto-disparável. Ao se completar o estabelecimento da conexão e se determinar os parâmetros do mecanismo, dispara-se o temporizador auto-disparável responsável por incrementar os créditos a cada ($1/\lambda_{méd}$) unidades de tempo, onde $\lambda_{méd}$ é a taxa média policiada. O contador de créditos pode ser incrementado até o limite máximo $C_{máx}$. A cada célula submetida ao mecanismo, se o crédito for positivo, este é decrementado e a célula é transmitida. Este procedimento está ilustrado no fluxograma da Figura IV.5.

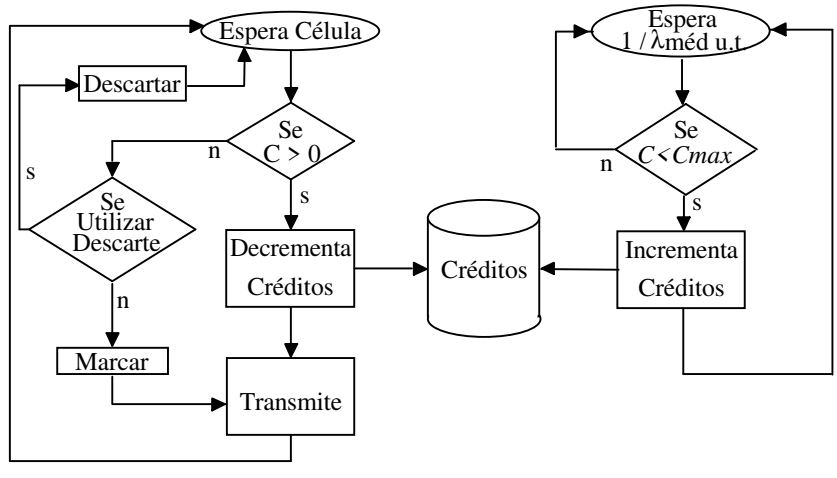


Figura IV.5: Diagrama em blocos da implementação do balde furado.

Uma outra implementação deste mecanismo, proposta por Dittmann *et al* [DJM91], é baseada em marcações de tempo de forma que a atualização do contador é feita através de

uma subtração a cada chegada de célula. O valor a ser subtraído dos créditos é calculado de acordo com o tempo decorrido desde a chegada da última célula. Este esquema possui as desvantagens de requerer marcas de tempo demasiadamente precisas e de executar operações de subtração não-unitárias. O pseudocódigo deste esquema é o seguinte:

A cada chegada de célula faça:

1. Δt := tempo_ultima_chegada - tempo_atual;
2. tempo_ultima_chegada := tempo_atual;
3. se $C < \Delta t \times \lambda_{méd}$
   então $C := 0$;
   senao $C := C - \Delta t \times \lambda_{méd}$;
4. se $C > C_{máx} - 1$
   então descartar célula;
   senao $C := C + 1$;

IV.3.2 Balde Furado com Suavização do Tráfego de Saída

Foi implementado ainda o mecanismo do balde furado com limitação da taxa máxima de transmissão, ou com suavização do tráfego de saída. Esta variação do balde furado é implementada pela adição ao mecanismo original de um temporizador auto-disparável de controle da taxa máxima de saída e de uma fila FIFO. Neste esquema (Figura IV.6), a cada célula entregue ao mecanismo, se o crédito for positivo, este é decrementado e a célula é enviada para a fila FIFO de saída. O temporizador de suavização do tráfego de saída é responsável por retirar as células da fila FIFO e enviá-las para serem transmitidas a cada $1/\lambda_{máx}$ unidades de tempo, onde $\lambda_{máx}$ é a taxa máxima de transmissão contratada.

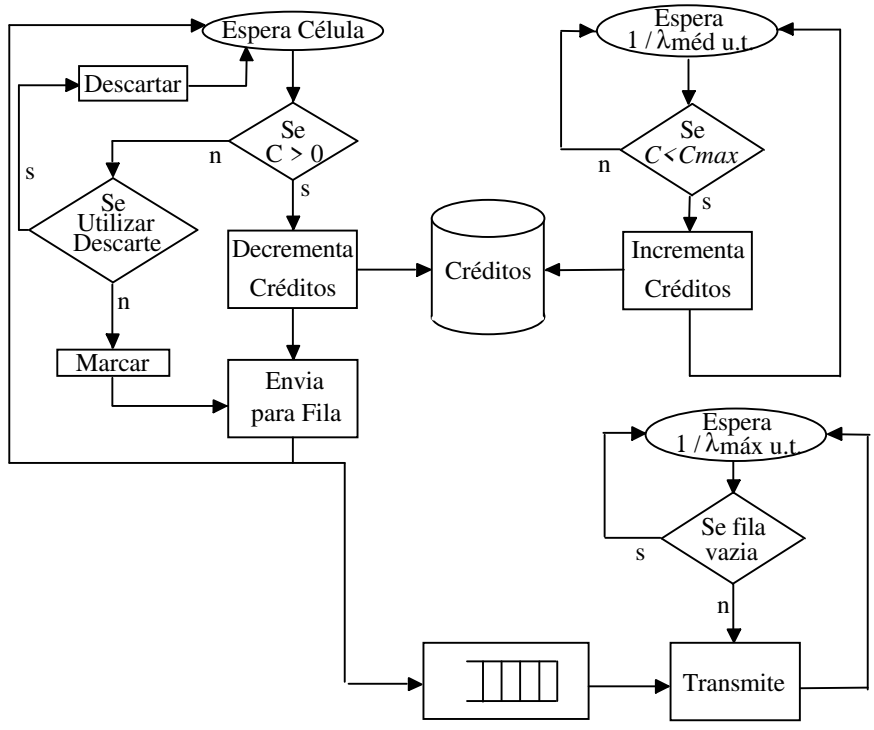


Figura IV.6: Diagrama em blocos da implementação do balde furado com suavização do tráfego de saída.

IV.3.3 Janela Saltitante

O mecanismo de janela saltitante (*jumping window*) pode ser implementado por meio de um temporizador e um contador, de acordo com o seguinte procedimento. Ao concluir a fase de estabelecimento da conexão e de determinação dos parâmetros do mecanismo, o mecanismo janela saltitante cria um temporizador auto-disparável responsável pela atualização do credito a cada *T* unidades de tempo até o termino da conexão. Paralelamente, a cada célula entregue ao janela saltitante, este mecanismo 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 nesta implementação, optou-se por descartar a célula.

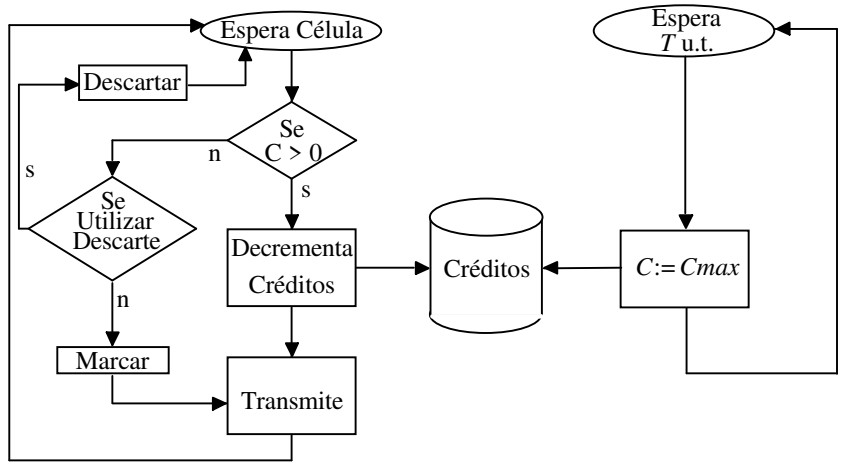


Figura IV.7: Diagrama em blocos da implementação da janela saltitante.

IV.3.4 Janela Saltitante Sincronizada

O mecanismo janela saltitante sincronizada (*triggered jumping window*) foi implementado de forma similar a janela saltitante, no entanto o temporizador de atualização de créditos é criado no instante da chegada da primeira célula de cada janela de comprimento *T*.

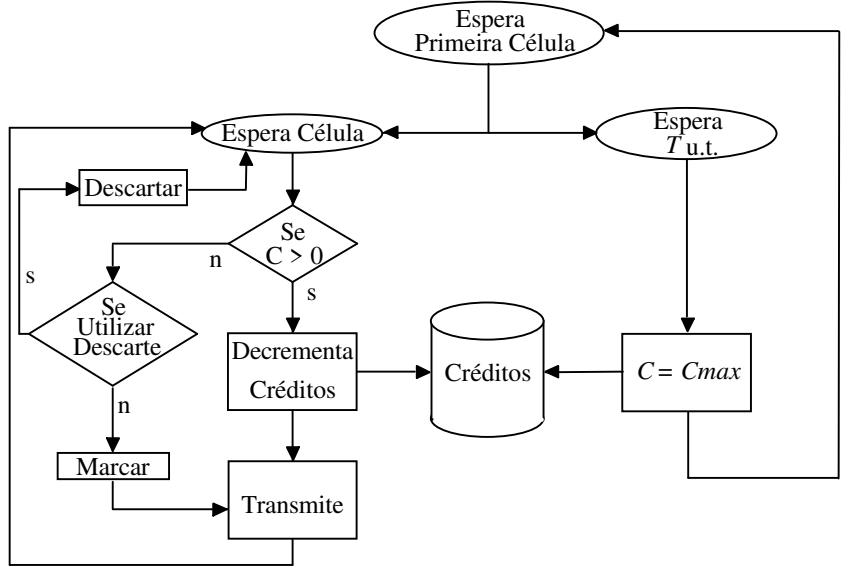


Figura IV.8: Diagrama em blocos da implementação da janela saltitante sincronizada.

Isto é implementado de forma bastante simples através de uma pequena alteração na máquina de estados do mecanismo, onde após o estabelecimento da conexão e após cada "estouro" do

temporizador, a janela saltitante sincronizada fica no estado "Espera Primeira Célula". Ao receber uma célula neste estado, dispara o temporizador, decrementa o crédito e passa para o estado de transmissão normal.

IV.3.5 Janela Deslizante Contínua

O mecanismo de janela deslizante contínua - JDC (*sliding window*) pode ser implementado por meio de diversos temporizadores e por um contador. A cada célula entregue ao mecanismo JDC, se o crédito for positivo, o mecanismo decrementa o crédito e dispara um temporizador com duração igual a *T* unidades de tempo. No "estouro" de cada temporizador a variável correspondente ao crédito é incrementada. A utilização de um grande e variável número de temporizadores implica em uma maior utilização da memória e da CPU, além de dificultar uma reserva de recursos eficiente para a conexão.

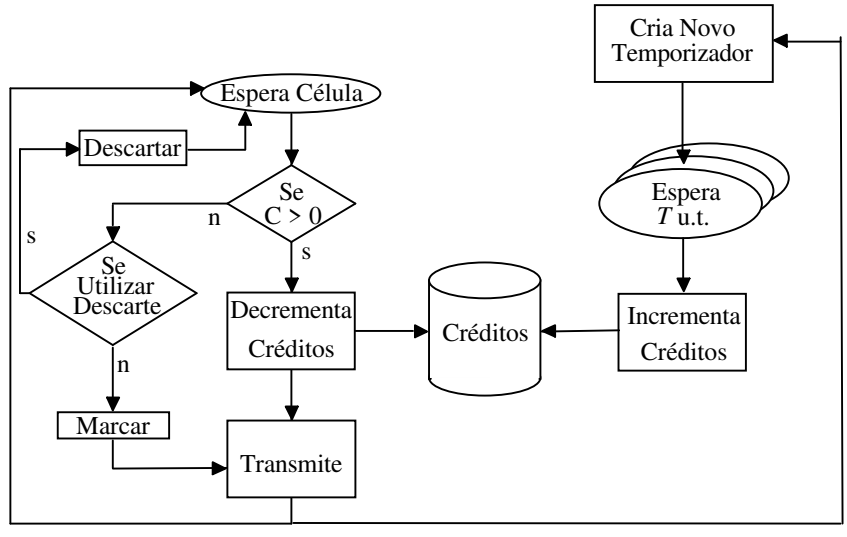


Figura IV.9: Diagrama em blocos da implementação da janela deslizante contínua.

IV.3.6 Janela Deslizante Discretizada

O mecanismo janela deslizante discretizada - JDD (*discretized sliding window*) é implementado de forma semelhante ao mecanismo JDC, porém com o uso de um único temporizador e de um vetor de deslocamento de *bits*, no qual cada *bit* deste vetor registra a chegada de uma célula na unidade de tempo correspondente, onde em cada unidade de tempo (por exemplo, t_{cel} = 2,7 μs para taxa de transmissão de 155 Mbps) permite a chegada de apenas uma célula. Neste esquema, além do vetor de *bits*, existe a necessidade de apenas um temporizador auto-disparável e um contador. Este esquema permite uma melhor utilização da memória e da CPU. O *bit* final do vetor de *bits* indica se chegou uma célula *T* unidades de tempo anteriores a atual. Este bit é somado ao número de créditos e deslocado a cada unidade de tempo t_{cel}. Cada célula não violadora, entregue ao mecanismo, seta o *bit* inicial do vetor e decrementa a variável crédito. Após *T* unidades de tempo este *bit* terá sido deslocado para a posição do *bit* final do vetor e será somado ao número de créditos.

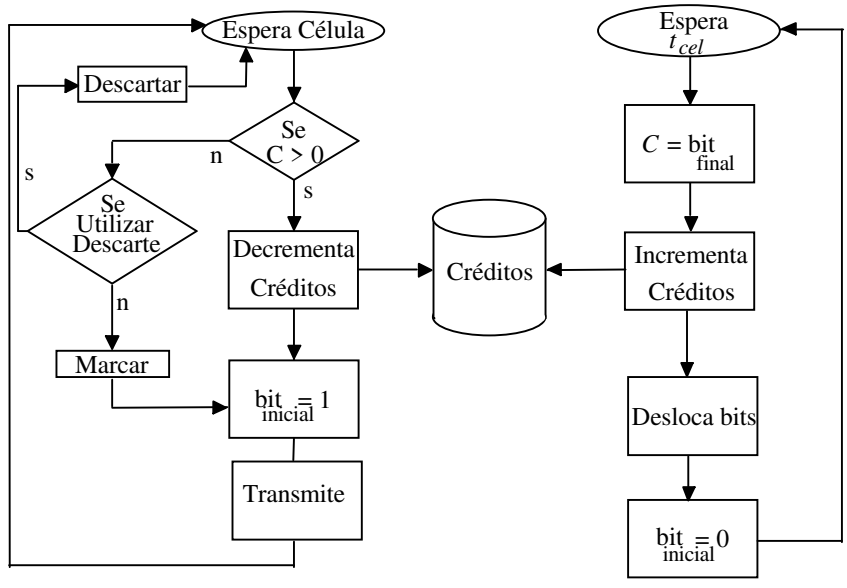


Figura IV.10: Diagrama em blocos da implementação da janela deslizante discretizada.

IV.3.7 Janela Móvel Particionada

O mecanismo janela móvel particionada - JMP foi implementado por meio de um temporizador auto-disparável e 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 mecanismo janela deslizante discretizada. Quanto menor o número de partições, menor será a memória necessária, menor será a utilização da CPU e mais próximo estará do mecanismo janela saltitante.

O temporizador é responsável pela atualização do crédito a cada *T/P* unidades de tempo. Esta atualização corresponde ao incremento 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.

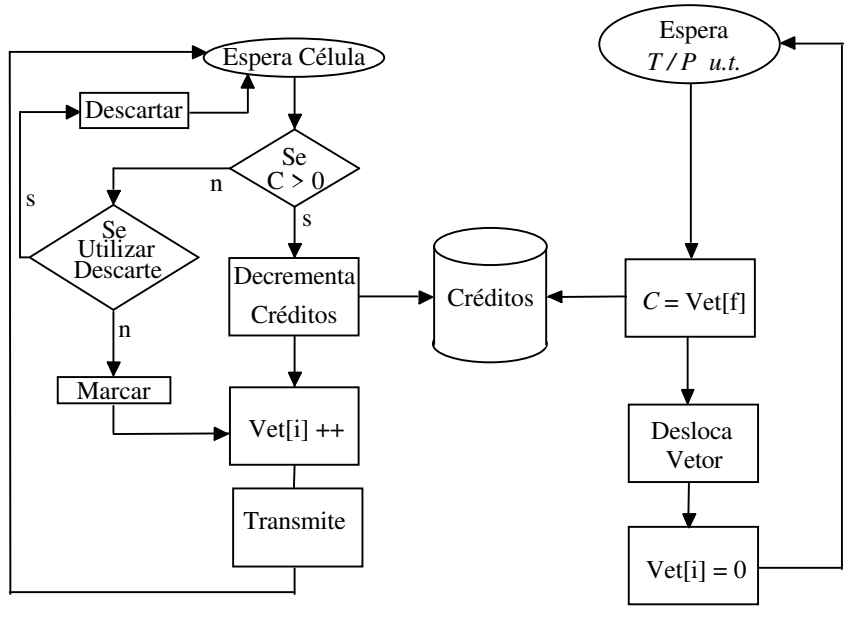


Figura IV.11: Diagrama em blocos da implementação da janela móvel particionada.

IV.4 Análise Comparativa

Esta seção visa apresentar dados comparativos obtidos a partir de uma implementação específica, enfocando a complexidade de implementação dos mecanismos. São analisados a quantidade de memória necessária, o número de temporizadores, a precisão exigida para os temporizadores, o tempo médio de processamento total de cada célula (incluindo todo o processamento do protocolo do sistema de comunicação: alocação de estruturas de primitivas, preenchimento de cabeçalhos, retiradas e inserções em filas, etc.) e o tempo médio de processamento específico do mecanismo (onde são computados apenas os procedimentos referentes ao mecanismo: criação de temporizadores, testes lógicos, atualização de variáveis, etc.) para cada célula. Para o caso específico dos mecanismos baseados em janela, foi analisada a probabilidade de violação, verificando-se o comportamento do mecanismo janela móvel particionada, com relação aos mecanismos janela saltitante e janela deslizante.

A Tabela IV.1 contém informações comparativas entre os mecanismos. A notação $\{x\}$ representa o número de *bits* necessário para representar o valor x. O tempo de processamento de cada célula, total e específico de cada mecanismo, foi medido em uma estação de trabalho SUN SPARC 2. Os tempos de todos os mecanismos foram obtidos através da média de 1.000 medições, todas feitas nas mesmas condições de operação da máquina. A estação estava dedicada para a execução dos processos dos mecanismos. A computação dos tempos foi obtida através de um relógio virtual que computa separadamente o tempo de processamento de cada processo.

Quanto aos resultados obtidos, observa-se na Tabela IV.1 que os mecanismos balde furado, janela saltitante e janela saltitante sincronizada são os mais eficientes, com tempos de processamento, geral e específico, da ordem de 160 μs e 140μs. O mecanismo janela

deslizante contínua é um dos mais custosos em relação ao tempo de processamento, 192 μs e 167 μs, devido ao disparo dos temporizadores a cada chegada de célula. O mecanismo balde furado com suavização do tráfego de saída teve um custo de processamento alto devido à manutenção da fila de saída. Já o mecanismo janela móvel particionada mostrou possuir um tempo médio de processamento de cada célula relativamente independente do número de partições e de valor intermediário ao janela saltitante e ao janela deslizante discretizada, da ordem de 176 μs e 150 μs.

Tabela IV.1: Comparação entre os mecanismos de policiamento.

Mecanismo	BF	BF suav.	JS	JSS	JDC	JDD	JMP
Tempo de Proc. Total	165 μs	194 μs	160 μs	165 μs	192 μs	176 μs	176 μs
Tempo de Proc. do Mecanismo	143 μs	170 μs	141 μs	142 μs	167 μs	150 μs	150 μs
Número de Temporizadores	1	2	1	1	C	1	1
Precisão dos Temporizadores	variável	variável	baixa	baixa	alta	alta	ajustável
Memória para Temporizadores	{Tmp.}	{Tmp.}	{Tmp.}	{Tmp.}	$C \times$ {Tmp.}	{Tmp.}	{Tmp.}
Memória para Contadores e Parâmetros	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$	$\{C\}$ + $\{C_{\max}\}$ + $\{P\}$

Um fator importante a ser considerado é o número de temporizadores necessários para a implementação do mecanismo. Observa-se que o mecanismo janela deslizante contínua necessita do maior valor: C temporizadores Deve-se ressaltar que quanto maior o número de temporizadores, maior será a utilização da CPU. O número de temporizadores do mecanismo balde furado com suavização do tráfego de saída não pode ser comparado com os demais, visto que este equivale a dois mecanismos de policiamento em cascata, necessitando portanto do dobro de temporizadores.

Quanto à precisão requerida pelos temporizadores, os temporizadores do mecanismo janela deslizante contínua exigem uma precisão altíssima, podendo ser disparados na chegada de cada célula, e portanto, em qualquer instante de tempo. A janela deslizante discretizada já é menos crítica, porém há ainda a necessidade de uma precisão bastante alta, por exemplo 2.7 μs para a taxa de transmissão de 155 Mb/s. O balde furado necessita de temporizadores da ordem do inverso da taxa média policiada e da taxa máxima de transmissão. Para taxas de por exemplo 100 células / segundo, o temporizador deve ser de 10 ms. Já a janela saltitante utiliza um temporizador da ordem de T unidades de tempo, e portanto sua precisão, em média, não será crítica. Uma das grandes contribuições do mecanismo janela móvel particionada é permitir a utilização de temporizadores com toda a faixa de precisão, desde a utilizada na janela saltitante até a utilizada na janela deslizante contínua, conforme o número de partições utilizado.

A quantidade de memória necessária para cada mecanismo pode ser dividida na memória utilizada pelos temporizadores, na memória utilizada pelos contadores e parâmetros dos mecanismos e na memória de controle auxiliar. Esta última é necessária apenas para os mecanismos janela deslizante e janela móvel particionada, que necessitam, respectivamente, de

$$\left\lceil \frac{T}{t_{cel}} \right\rceil \text{ e } P \times \left\{ T \Big/ \left(P \times t_{cel} \right) \right\} \ bits \tag{IV.1}$$

para memorização do tráfego, onde a notação $\{x\}$ na equação (IV.1) representa a quantidade de *bits* necessários à representação do valor x. As demais regiões de memória estão ilustradas na Tabela IV.1. Pode-se reparar que a janela deslizante contínua necessita de $C \times \{\text{Tmp.}\}$ *bits* para os temporizadores, enquanto a janela móvel particionada necessita de {P} *bits* para armazenar o número de partições. 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, onde a quantidade de memória de controle auxiliar é sempre inferior à quantidade necessária para a janela deslizante. Por

exemplo, o caso da janela deslizante discretizada com $T = 60 \times t_{cel}$, onde t_{cel} é o tempo necessário para a transmissão de uma célula, e $C = 30$. Neste caso ela necessita de 60 *bits* para memorizar o tráfego de uma janela. Se a janela móvel particionada com 20 partições for utilizada, 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*.

Teoricamente, o comportamento do mecanismo janela móvel particionada é, em termos de complexidade de implementação e de probabilidade de violação (p_{viol}), intermediário aos dos mecanismos janela saltitante e deslizante. A probabilidade de violação (p_{viol}) é dada pela probabilidade de, em uma chegada, o mecanismo detectar uma célula como violadora dos parâmetros contratados.

Além da computação da p_{viol} obtida na implementação, resultados de simulação foram analisados e comparados com a implementação. Os modelos de simulação estão descritos no capítulo III. Para a simulação dos mecanismos fez-se uso de um programa baseado em redes de filas, utilizando-se um intervalo de confiança de 0.95.

Em termos de computação da probabilidade de violação 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.

A Figura IV.12 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 a Figura IV.13, o rajada/silêncio. Os limites inferiores e superiores dos gráficos são as p_{viol} obtidas para os mecanismos janela saltitante e deslizante, respectivamente.

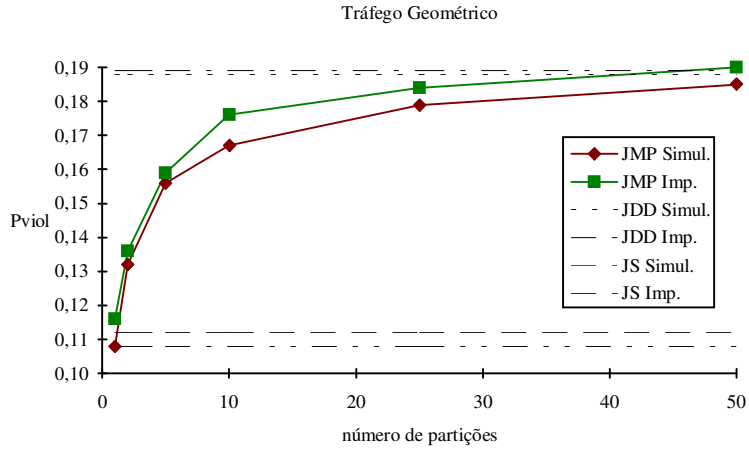


Figura IV.12: Probabilidades de violação do mecanismo JMP em função do número de partições para o tráfego geométrico.

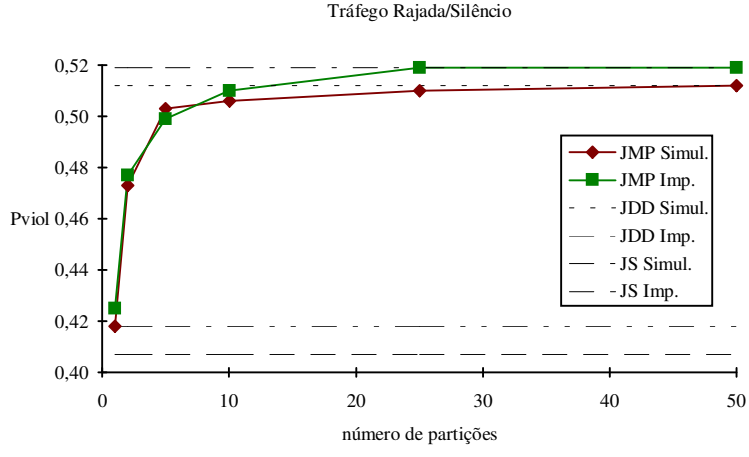


Figura IV.13: Probabilidades de violação do mecanismo JMP em função do número de partições para o tráfego rajada/silêncio.

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.

Os 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.