Uma Proposta de Algoritmo Assíncrono para Difusão Confiável Atômica

Líliam Terezinha Ribeiro Dueñas[1]
Liliam@cti.ansp.br

Joni da Silva Fraga
Fraga@atlas.lcmi.ufsc.br

Laboratório de Controle e Microinformática - LCMI
DEEL/CTC/UFSC
Cx Postal 476 Florianópolis - SC

Abstract

This paper presents a new reliable broadcast protocol which satisfies the proprieties of agreement, total ordering and asynchrounous termination, even in the presence of M-degree omission faults. It can recover the messages even in the presence of R faulty sites, minimizing, in many cases, the state transfer costs in application-replicated processing models. Each commitment is done in groups of L messages, under the supervision of a coordinator site, and with the participation of the majority of the group. The number of protocol messages is fixed, so that for a high value of L , the control cost approaches zero.

Resumo

Este artigo apresenta uma proposta de protocolo de difusão confiável que atende as propriedades de acordo, ordenação total e terminação assincrona mesmo na presença de faltas de omissão com grau M. Este protocolo permite a recuperação das mensagens mesmo na presença de R estações faltosas, minimizando, em muitos casos, os custos de transferência de estados em modelos de processamentos replicados na aplicação. Cada engajamento é feito em grupos de L mensagens (administrado por uma estação coordenadora) com a participação da maioria das estações do grupo. O número de mensagens-protocolo é fixo de forma que para um L alto o custo de controle tende a zero.

1 - Introdução

Nos últimos tempos, tem ganho interesse crescente a noção de grupo ou processamento replicado. As necessidades de consistência de estados entre replicas de processos fez com que este interesse se extendesse a modelos de comunicação de grupo.

---

[1] Atualmente Pesquisadora Associada no Centro Tecnológico para Informática - CTI

As necessidades de confiabilidade em comunicação de grupo determinou nos últimos dez anos o aparecimento de um grande número de algoritmos e protocolos de difusão confiável. Estes protocolos se caracterizam por atender uma faixa larga de propriedades envolvendo aspectos de ordem, acordo, sincronismo e mesmo de confiabilidade.

Neste artigo é apresentada a proposta de um algoritmo de difusão atômica, assíncrono, com características de robustez e tolerando faltas de omissão sucessivas dos participantes da comunicação. Este trabalho teve origem na necessidade de implementar um serviço de difusão confiável no sistema ADES [Fraga 89], criando desta forma um suporte para o desenvolvimento de modelos de tolerância a faltas fundamentados em processos replicados. As condições de aplicações em tempo real fez com que este algoritmo se adequasse as necessidades de baixa sobrecarga (overhead) e sobretudo que fosse um algoritmo limitado no tempo.

Nos itens subseqüentes a literatura é examinada e é apresentado o algoritmo proposto pelos autores. Em seguida são verificadas as propriedades do mesmo e discutida a sua comparação com exemplos relevantes na literatura.

2 - Trabalhos relacionados

A condição de protocolo de difusão confiável está fundamentada em três propriedades básicas: acordo, ordenação e terminação [Cristian 85]. Conforme o maior ou menor rigor destas propriedades define-se diferentes classes de protocolos de difusão.

A propriedade de acordo garante, mesmo na presença de faltas que ou todos os participantes corretos recebem uma mensagem difundida, ou nenhum destes a aceita. O engajamento (aceitação) da mensagem dependerá, portanto, das propriedades envolvidas no protocolo utilizado. O espectro de faltas é amplo, envolvendo faltas de "crash", omissão, temporização e faltas arbitrárias e, quanto mais restritivas as hipóteses de faltas, menos complexos e de menor custo é a implantação do acordo. Assim protocolos como [Birman 87], [Chang 84] e [Luan 90] garantem acordo na presença de faltas de crash e omissão. Protocolos que atendem hipóteses mais gerais devem se utilizar de acordos normalmente mais complexos: "Byzantine agreement" [Lamport 82].

Os protocolos de difusão podem ser divididos em classes, segundo o tipo de ordenação usado [Shivastava 91] : "**Protocolos sem ordenação**", "**Protocolo de difusão FIFO**", "**Protocolos com ordenação causal**" e "**Protocolos com ordenação total**". Estes últimos, também chamados de "**protocolo de difusão atômica**" são os utilizados quando as réplicas de processos são ativas e o "determinismo de estado" tem que ser mantido [Schneider 90].

Em [Verissimo 89] são definidos alguns parâmetros que servem para caracterizar a terminação de um protocolo:

- **Tempo de Execução (T_e)**: é o intervalo de tempo entre o pedido de enviar uma mensagem m_i e a confirmação da última indicação de recebimento da mesma pelos participantes do grupo.

- **Tempo de Inconsistência** (T_i): é o maior intervalo de tempo entre indicações de recebimento da mensagem pelos participantes destinatários.

Um protocolo de difusão é então dito "Rígido" se para cada execução existir um τ, tal que $|Ti| \leq \tau$, com $\tau > 0$. Por outro lado, este protocolo é dito "Estável" se existir um σ, tal que, para cada duas execuções quaisquer $|T_{e1} - T_{e2}| \leq \sigma$, com $\sigma > 0$.

Com base nestas definições um protocolo é tanto mais síncrono quanto menor forem τ e σ quando comparados a T_e. Os protocolos que se aproximam da rigidez e estabilidade completa, isto é, τ e σ desprezíveis em relação a T_e, são ditos com **Sincronismo Forte**. É o caso, por exemplo de [Cristian 85]. No sentido oposto, apresentam-se os protocolos "assíncronos". Nestes protocolos as noções de rigidez e estabilidade são relaxados, sendo requerido apenas que terminem, isto é, é necessário que se tenha uma condição de T_e limitado: $Pr \{ T_e > T_e(max) \} \simeq 0$.

Esta condição, garante o tempo de execução limitado, mas não o suficiente para garantir a adequação destes protocolos para aplicações em tempo real. Valores muito grandes de T_e (latência de mensagem) implicarão no baixo desempenho do protocolo. Exemplos de protocolos assíncronos estão em [Chang 84], [Birman 87] e [Luan 90].

Atualmente existe um vasto número de proposições de protocolos de difusão na literatura. A seguir são apresentados alguns destes protocolos considerados como os mais representativos.

2.1 - Principais protocolos

O protocolo de difusão de [Chang 84] apresenta características de protocolo assíncrono com ordenação total, suportando propriedades de acordo, de ordenação e de terminação, em presença de faltas por omissão. O "commit" de mensagem é centralizado sobre o conceito de token circulante. A estação possuidora do token é denominada "**token site**" e é a responsável, enquanto possuidora do token, pelo reconhecimento das mensagens difundidas, pela retransmissão de mensagens quando requisitado por algum receptor e também, pela transferência do token para o próximo token site. As estações operacionais do sistema são relacionadas em uma lista denominada "**token list**" que determina o anel virtual para a transferência do token. Uma mensagem é engajada neste protocolo após R+1 transferências de token ("R-resilient").

O ABCAST de [Birman 87] possui o engajamento distribuído onde cada estação é responsável em dar a ordem total e o acordo sobre as suas mensagens difundidas. O timestamp associado com cada mensagem é negociado com as estações do grupo incluindo duas fases de trocas de mensagens.

O protocolo apresentado em [Cristian 85] na verdade representa uma família de protocolos de difusão atômica que resistem a faltas de severidade progressiva. Estes protocolos se baseiam em difusão sobre uma rede ponto a ponto. As propriedades de acordo, ordenação e de terminação são suportadas através de uma execução fortemente síncrona, implementada a partir da sincronização de relógios físicos (locais).

O protocolo de difusão discutido em [Luan 90] é um protocolo assincrono com ordenação total que suporta faltas por omissão. As mensagens são recebidas por todas estações numa única ordem determinada por decisão de consenso de maioria em cada rodada de engajamento.

O protocolo de [Melliar-Smith 90] é um protocolo assincrono, da classe dos protocolos com difusão totalmente ordenada (difusão atômica). O "TRANS PROTOCOL" realiza técnica de reconhecimento por carona: cada difusão de mensagem de aplicação transporta reconhecimentos de mensagens difundidas anteriormente. A ordenação depende destes reconhecimentos.

2.2 - Modelo Geral de Protocolos de difusão confiável

Um sistema distribuído típico, consiste de um conjunto de estações interconectadas por uma rede de comunicação. Em cada estação pode existir um ou mais processos se executando, no entanto, por uma questão de simplicidade, será assumida a existência de apenas um processo por estação.

O Protocolo de Difusão é um serviço que opera numa camada denominada "Camada de Difusão Confiável" e que se localiza entre a camada de aplicação e o suporte de comunicação remota do sistema. O suporte de comunicação remota contém todo o hardware e software necessários para habilitar uma mensagem a ser enviada de um processo para outro. A transmissão de uma mensagem pelo suporte de comunicação (transmissão física na rede) pode se dar através de serviços de comunicação ponto a ponto ou de comunicação multiponto (difusão).

O envio de uma mensagem inclui, em geral, outras informações que devem ser utilizadas pela camada de difusão nas outras estações. Dependendo do protocolo de difusão que está sendo executado, pode haver mais rodadas de comunicação entre as estações a nível da camada de difusão no sentido de garantir todas as propriedades básicas do protocolo. O engajamento das mensagens se dará com a verificação das propriedades e a conseqüente passagem destas do serviço de difusão confiável para a camada de aplicação.

Um protocolo de difusão típico, se dá em três fases distintas: (i) "difusão" da mensagem de aplicação (ii) "recepção" da mensagem (iii) "verificação" das propriedades de acordo e ordenação e "aceitação" da mensagem (engajamento). Uma execução de um protocolo de difusão confiável é esboçado em CSP na figura 2.1. As diferenças entre os algoritmos existentes estão nas distintas técnicas utilizadas nestas três fases.

3 - Algoritmo proposto

Este protocolo, a exemplo de [Chang 84] e [Luan 90], caracteriza-se pelo processo de engajamento centralizado sobre uma estação: *a coordenadora*. Neste sentido, as estações participantes se apresentam ordenadas em um anel virtual segundo seus

endereços lógicos (i= 1,2, ... , N). O papel de coordenadora é transferido, através de *passagem de token*, entre as estações participantes do grupo, no fim de cada engajamento. As comunicações, segundo o algoritmo se dão através de difusões entre os participantes do grupo G, formado por N estações { E_1, E_2, ... , E_N } no sistema S. Uma *execução de protocolo* é caracterizada pelas ações que ocorrem entre duas passagens sucessivas do token (duas trocas sucessivas de coordenadoras).

```
X :: Protocolo Difusão Confiável
APj :: Aplicação
CD(i) :: Camada de Difusão Confiável na estação i
msg :: Mensagem da Aplicação
mdif :: Mensagem difundida pela Camada de Difusão Confiável

X:: * [     [  APj?msg  mdif := msg + mensagens de controle;
               (i:1..N) CD(i);  CD(i)!mdif                                  ]
        [] [  (i:1..N) CD(i);  CD(i)?mdif  <  Ordena mdif no buffer de recepção >  ]
        [] [  < Existe mdif no buffer de recepção da CD(j) >;
              * [  < Protocolo de acordo>;  <Reordena mdif no buffer de recepção>    ]
              * [  < Aceita mdif >;  < Envia mdif aceita para a APj >        ]   ]   ]
```

Figura 2.1 - Algoritmo Básico de Difusão Confiável

O protocolo tem como premissa a tolerância a *faltas de omissão* [Veríssimo 91]. O *grau de omissão* é M, isto é, uma estação permanece entre as estações operacionais até que atinja M omissões sucessivas, a partir do que a estação é dita em "crash" e é retirada do grupo de estações operacionais. O grau de robustez ("resiliency") assumido é R, isto é, até R estações podem falhar que informações sobre "commits" anteriores se manterão.

Uma estação E_i, pertencente ao grupo G especificado, pode receber ou enviar, sobre o suporte de comunicação, as mensagens de aplicação independentes da execução de protocolo em andamento. A única limitação imposta é que o número de mensagens não engajadas presentes na camada de difusão confiável referentes as emissões de uma de estação E_i, não ultrapassem a K mensagens. A limitação de K se deve a capacidade máxima de "bufferização", por estação emissora, de mensagens (ainda) não engajadas presentes na camada de difusão confiável. Em cada estação, a capacidade de bufferização é representada pelas N filas F_i (uma fila para cada estação E_i), depositárias das mensagens difundidas na rede e ainda não engajadas (figura 3.1).

Toda mensagem difundida através do suporte de comunicação apresenta o número da estação emissora e o número da mensagem (ordem local da estação emissora), como informações de controle acrescentadas pela camada de difusão confiável na estação emissora. Diante destas informações, toda mensagem recebida do suporte de

comunicação pela camada de difusão confiável, é colocada numa das N filas de mensagens não engajadas (F_i), correspondente ao emissor E_i.

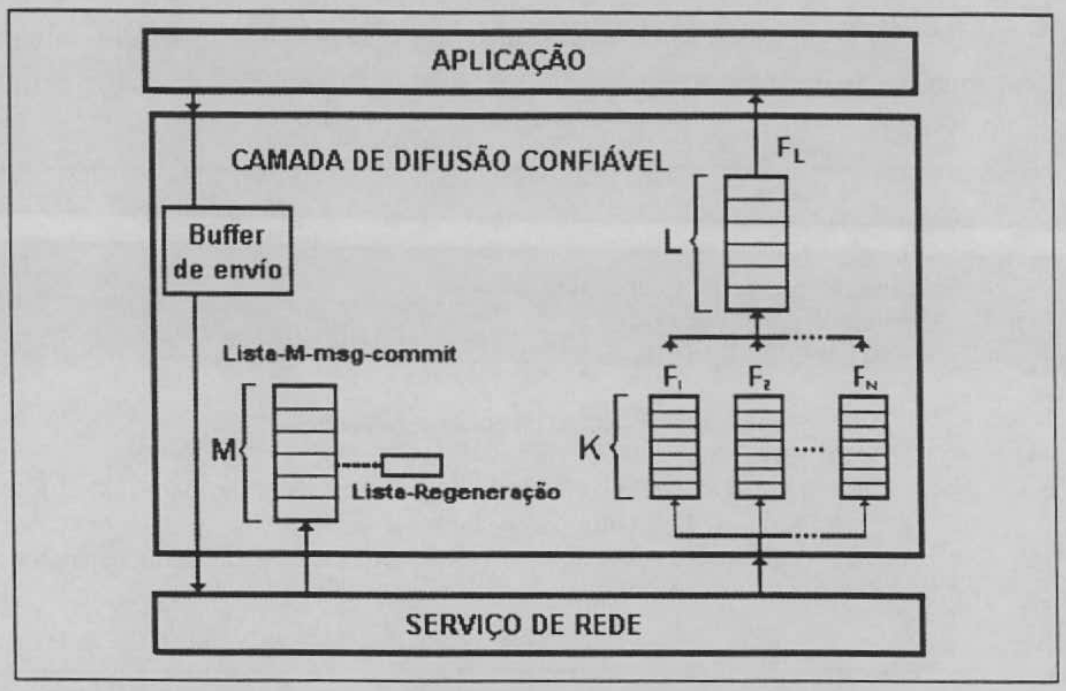

Figura 3.1 - Estratificação do protocolo proposto

A capacidade de engajamento por execução de protocolo é L mensagens. O mecanismo de engajamento é ativado quando na camada de difusão se completam L mensagens difundidas, presentes entre as filas F_i de mensagens não engajadas ou ainda por decurso de prazo. A estação coordenadora envia então uma mensagem-protocolo, "pedido-de-commit", às demais estações dando inicio ao *período de engajamento*. Desta forma, L (ou menos) mensagens presentes nas "Filas de mensagens não engajadas" (F_i) serão propostas para o engajamento, juntamente com a indicação de uma "nova coordenadora". Assim, se as demais estações concordarem com a lista de mensagens a engajar e estiverem aptas para o "commit", estas enviam um reconhecimento positivo ao pedido-de-commit. No caso de desacordo com a lista de mensagens ou com a transferência do token, o reconhecimento deve ser negativo.

Se a maioria das estações enviaram mensagem de reconhecimento, com nenhum reconhecimento negativo ao pedido-de-commit, a estação coordenadora difunde a mensagem-protocolo "commit" sinalizando o engajamento das mensagens que estavam em negociação. Esta mensagem de commit deve também transferir o token a uma nova coordenadora. A recepção desta mensagem-protocolo "commit" por uma estação implica no engajamento das mensagens especificadas no "pedido-de-commit", ou seja, estas são colocadas na fila F_L (fila de mensagens engajadas, figura 3.1), à disposição da aplicação. A mensagem commit é armazenada na Lista-M-msg-commit (figura 3.1) para permitir a recuperação de estações que se omitiram a esta execução de protocolo. A figura 3.2

sintetiza os passos para o engajamento de mensagens. Os aspectos de ordenação e do engajamento em si serão apresentados em mais detalhas no item a seguir.

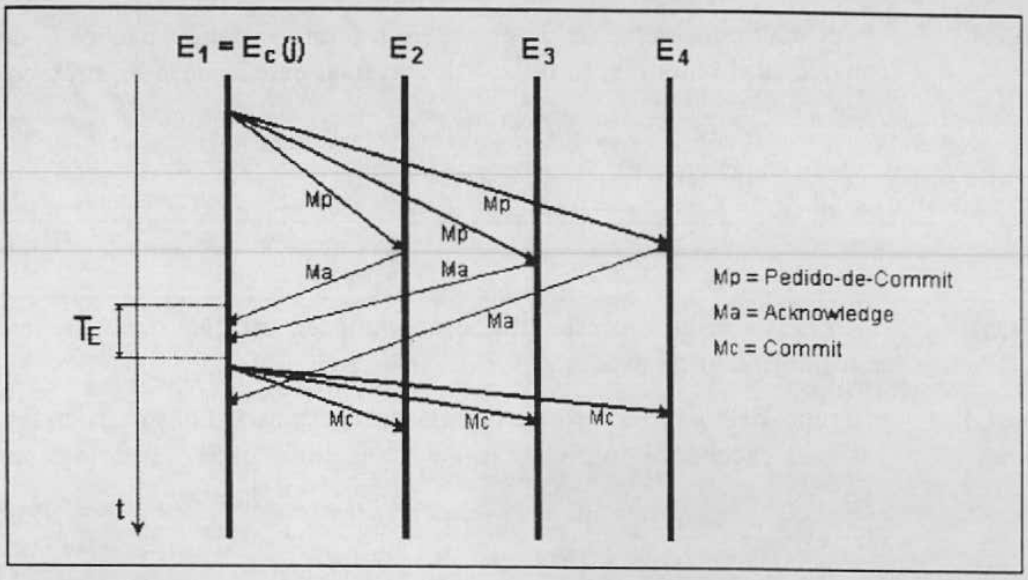

Figura 3.2 - Período de Engajamento

3.1 - Especificação do protocolo

Neste item são apresentados em detalhes as fases e as técnicas adotadas no algoritmo proposto. Inicialmente são introduzidas algumas definições que servem de veículo às descrições posteriores.

Definições

Considerando a execução de protocolo j, tem-se:

$E_c(j)$: é a estação coordenadora na execução de protocolo j. Controla o processo de engajamento de mensagens e a reposição das mesmas na execução j.

$E_c(j+1)$: é a estação "Nova Coordenadora". Esta estação irá receber o token no fim da execução de protocolo j. A nova coordenadora é a estação possuidora da última mensagem a ser engajada em j.

G : O grupo G é um conjunto composto de N estações $\{ E_1, E_2, .. , E_N\}$ do sistema S.

$G_p(j)$: é o Grupo-Participantes na execução de protocolo j. Ou seja, estações consideradas operacionais no inicio da execução de protocolo j.

$G_L(j)$: é o Grupo-L. Formado pelas estações E_i origem das mensagens que estão em processo de engajamento na execução de protocolo j. $G_L(j)$ é composto pelas estações-L.

$G_A(j)$: é o Grupo-Ausentes na execução j, isto é, o conjunto de estações E_i pertencentes a $G_p(j)$ que se "omitiram" durante a execução de protocolo j.

$G_r(j)$: Grupo-Recuperando a execução j é o conjunto de estações E_i pertencentes a $G_A(j)$ que estão em processo de recuperação da execução de protocolo j.

$G_f(j)$: Grupo-Falho na execução de protocolo j, formado pelas estações E_i de $G_A(j)$ consideradas em situação de "crash" no final da execução de protocolo j:

$$G_f(j) = \bigcap_{m=j-M+1}^{m=j} \left(G_A(m) - G_r(m)\right)$$

$G_I(j)$: o Grupo-Inserido corresponde ao conjunto de estações inseridas em $G_p(j)$ antes do início da execução j.

$G_F(j)$: o Grupo-Falho em G é formado pelas estações de G consideradas falhas em todas as execuções do protocolo até j e que não foram reinseridas em $G_p(j)$:

$$G_F(j) = \bigcup_{m=1}^{m=j} \left(G_f(m-1) - G_I(m)\right)$$

$G_{EG}(j)$: é o Grupo-Engajante formado pelo conjunto de estações E_i de $G_p(j)$ que reconheceram ao pedido-de-commit da execução de protocolo j e que supostamente, engajaram as mensagens propostas desta execução.

3.1.1 - Difusão de uma mensagem sobre o suporte de comunicação

Toda mensagem vinda da aplicação, na camada de difusão confiável, é acrescida das informações identificando a mensagem e sua estação emissora. A camada de difusão confiável, na estação E_i difunde, então, a mensagem ($m_{i,k}$) com suas respectivas informações de controle, em $G_p(j)$, usando os serviços de rede.

A estação E_i pode emitir mensagens de sua aplicação em $G_p(j)$ em qualquer instante da execução de protocolo j. A única limitação imposta é que o número de mensagens não engajadas presentes na camada de difusão confiável, referentes a emissões de E_i, não ultrapassem o valor K.

Na recepção de uma mensagem $m_{i,k}$ difundida sobre a rede, as informações de controle são verificadas, de modo que a mensagem possa ser colocada na fila F_i correspondente a sua estação emissora E_i, na ordem do seu número de seqüência k (ordem de emissão da sua estação de origem). Esta mensagem então, passará a fazer parte das mensagens não engajadas.

As mensagens duplicadas são descartadas; o controle de identificação está baseado no número de sequência k da estação emissora. Uma mensagem duplicada é detectada quando o número da sequência recebida é menor que o número esperado para esta emissora. A perda de mensagem tem sua detecção quando uma mensagem apresenta o número de sequência maior do que o número esperado para o emissor em questão. A

perda, pelo receptor, da sequência de numeração nas mensagens recebidas referente a um emissor E_i, faz com que a estação receptora envie a E_i um pedido de retransmissão da(s) mensagem(s) cujo número de identificação está em falta em sua fila F_i.

3.1.2 - Periodo de Engajamento

O protocolo, conforme descrito anteriormente, apresenta uma capacidade de engajamento de L mensagens por execução do protocolo. As mensagens da aplicação se acumulam na camada de difusão confiável, nas N filas F_i, até atingir o valor L de mensagens ou ainda expirar o "Timeout de ativação" para o periodo de engajamento (T_A). Em uma destas situações, a estação coordenadora de $G_p(j)$, a $E_c(j)$, deve iniciar o processo de engajamento destas mensagens, enviando um "pedido-de-commit", o qual da inicio ao periodo de engajamento j.

Mensagens a engajar, ordenação e a nova coordenadora

As mensagens em negociação e que deverão ser engajadas no final da execução j, são as presentes nas filas F_i dos emissores E_i pertencentes a $G_L(j)$. A definição dos elementos de $G_L(j)$ depende das mensagens presentes nas filas F_i na camada de difusão. Neste sentido, é definido $B_i(j)$ como a "bufferização j" presente em F_i no inicio do engajamento j, ou seja, o conjunto de mensagens provenientes de E_i, armazenadas em F_i, segundo a ordem local definida pelos números de sequência das mensagens:

$$\mathbf{B_i(j) = \{ m_{i,k}, m_{i,k+1}, \ldots, m_{i,k+s} \} \wedge}$$

$$\mathbf{m_{i,k} \rightarrow m_{i,k+1} \rightarrow \ldots \rightarrow m_{i,k+s}} \qquad \text{(ordem local)}$$

onde a relação, notada por "$\rightarrow$", indica precedência com base no número de sequência das mensagens (k).

Considerando a estação E_i a coordenadora da execução j (E_i=$E_c(j)$) e λ a distância da nova coordenadora ($E_c(j+1)$) em relação a E_i na ordenação do anel, tem-se então:

$$\sum_{m=i}^{m=\lambda} card(B_m(j)) \leq L \qquad \textbf{(3.1)}$$

onde card($B_i(j)$) é a função que indica a cardinalidade do conjunto $B_i(j)$ e $1 \leq \lambda \leq N$.

A bufferização $B_L(j)$ define o conjunto das mensagens engajadas no final da execução de protocolo j e presentes em F_L. O valor L é a cardinalidade máxima que $B_L(j)$ pode alcançar (na situação sem esgotamento do timeout T_A). As mensagens são ordenadas em $B_L(j)$ segundo uma ordem total tomando como base as ordens locais em cada $B_i(j)$ e a ordem estabelecida entre as bufferizações no engajamento j:

$$\mathbf{B_i(j) \rightarrow\!\!\rightarrow B_{i+1}(j) \rightarrow\!\!\rightarrow \ldots \rightarrow\!\!\rightarrow B_{i+\lambda}(j)}$$

onde a relação notada por "$\rightarrow\rightarrow$", indica a precedência com base no número de sequência (i) das estações no anel virtual. Assim, duas mensagens engajadas em j, $m_{i,k}$ e $m_{s,r}$, apresentarão a relação de precedência: $\mathbf{m_{i,k} \Rightarrow m_{s,r}}$ indicando que $m_{i,k}$ precede $m_{s,r}$ segundo a ordem total estrita, notada por "$\Rightarrow$", se:

$B_i(j) \rightarrow\rightarrow B_s(j)$ onde $m_{i,k} \in B_i(j)$, $m_{s,r} \in B_s(j)$, para $i < s$

OU

$m_{i,k} \rightarrow m_{s,r}$ onde $m_{i,k}$, $m_{s,r} \in B_i(j)$, para $i = s \wedge k < r$

Algumas considerações podem ser tiradas destas relações:

1) A nova coordenadora será a estação que tiver a última mensagem em $B_L(j)$, segundo a ordenação "$\Rightarrow$" e mantendo $card(B_L(j)) \leq L$.
2) Toda mensagem que chegue em F_i após o inicio do periodo de engajamento j, pertence a $B_i(j+1)$.
3) Toda mensagem pertencente a fila da nova coordenadora ($E_c(j+1)$) que não entrar na bufferização $B_L(j)$, no final da execução de protocolo j, não será considerada pertencente a $B_{i+\lambda}(j)$, mas sim $B_{i+\lambda}(j+1)$

Processamento do pedido-de-commit(j)

A coordenadora $E_c(j)$ inicia o periodo de engajamento j quando L mensagens estão presentes nas filas F_i ou por esgotamento do decurso de prazo (timeout T_A) para a ativação do periodo de engajamento j. A estação $E_c(j)$ deve então enviar um "**pedido-de-commit(j)**". Esta mensagem-protocolo contem informações tais como: o número da versão atual do token e uma lista identificando as últimas mensagens a engajar em cada fila F_i das estações E_i pertencentes a $G_L(j)$ e ainda, a informação sobre a próxima coordenadora. Toda estação ao receber o pedido-de-commit(j) deve verificar as seguintes condições:

C1 - Lista de mensagens a engajar, produzida pela coordenadora está correta: isto é, não excluirem mensagens de F_i que num engajamento anterior, onde E_i pertencia ao grupo de estações-L, não foram consideradas.

Sejam $\mathbf{P_{EGi}(j)}$ o ponteiro da última mensagem a ser engajada em F_i, durante a execução de protocolo j para $E_i \in G_L(j)$ e, $\mathbf{L_{me}(j)}$: a lista de ponteiros $P_{EGi}(j)$ indicando na Execução de protocolo j, quais as últimas mensagens de cada F_i das estações $E_i \in G_L(j)$, que terão suas mensagens engajadas. Isto é:

$$L_{me}(j) = \{ P_{EG_i}(j) \mid E_i \in G_L(j) \wedge (i = 1, 2, \ldots, N) \}$$

O mecanismo de circulação de token determina a circulação do papel de coordenadora nos engajamentos de mensagens mas também, corresponde a circulação

do status de estação em $G_L(j)$. Considerando a execução de protocolo m, onde m < j (m antecede j), tal que:

$$E_i \in \{G_L(m) \cap G_L(j)\} \quad \wedge$$

$$E_i \notin \{G_L(m+1) \cup G_L(m+2) \cup \ldots \cup G_L(j-1)\}$$

neste caso, o ponteiro $P_{EG_i}(j)$ é dito "**Apropriado**" se:

$$P_{EG_i}(j) > P_{EG_i}(m) \quad \vee$$

$$P_{EG_i}(j) = SM$$

onde SM corresponde a indicação "sem mensagem".

Se a lista $L_{mc}(j)$ apresentar $P_{EG_i}(j)$ "apropriados" para todas as filas F_i de estações $E_i \in G_L(j)$, será considerada correta.

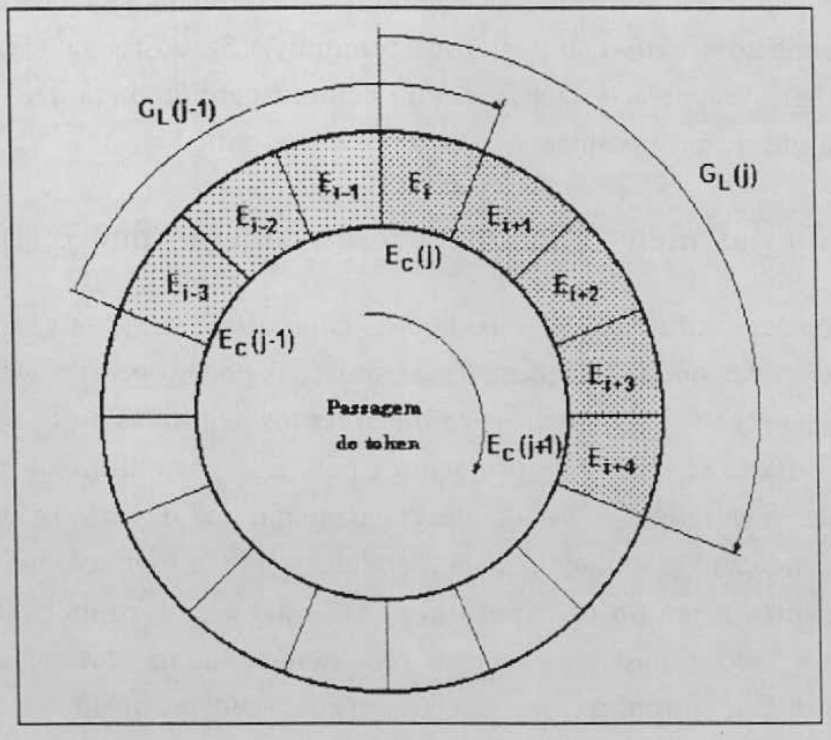

Figura 3.3 - Passagem de Token

Na situação considerada, para m = j-1 e $E_i \in \{G_L(m) \cap G_L(j)\}$ tem-se E_i participando, em duas execuções de protocolos consecutivas, das estações-L o que implica que E_i obrigatoriamente deve ser a $E_c(j)$. Em casos de m ≠ j-1, duas participações consecutivas de E_i como estações-L ($E_i \subset G_L(j) \wedge E_i \subset G_L(m)$) corresponde a uma circulação completa do token. A partir das considerações acima, pode-se afirmar que as mensagens difundidas serão engajadas no máximo em uma circulação completa do token. A figura 3.3 sintetiza o mecanismo de passagem do token.

C2 - Toda estação E_s que receber o pedido-de-commit(j) deve estar apta a participar do Periodo de engajamento j. Uma estação E_s estará apta

a participar do periodo de engajamento j se $E_s \in G_p(j) \wedge E_s \notin G_A(j-1)$. Se E_s esteve "ausente" na execução j-1 ($E_s \in G_A(j-1)$), não poderá se manifestar ao pedido-de-commit(j) antes de recuperar os commits anteriores que tenha perdido.

Não basta $E_s \in G_p(j) \wedge E_s \notin G_A(j-1)$ para estar apta para a execução j. É necessário que apresente a mesma imagem das mensagens nas filas F_i (as mesmas bufferizações $B_i(j)$) que a $E_c(j)$ está propondo para o engajamento). Se uma estação E_s apresentar, para uma das filas F_i, um ponteiro $P_{Fi}(j)$ tal que: $\mathbf{P_{Fi}(j) < P_{EGi}(j)}$, então, a proposição de mensagens para engajamento através do ponteiro $P_{EGi}(j)$ pela $E_c(j)$, não está completamente presente nas filas F_i da estação E_s. As mensagens não presentes deverão ser recuperadas através de um pedido-de-retransmissão à estação emissora das mensagens perdidas.

Verificadas e satisfeitas as duas condições C1 e C2, a estação E_s envia a $E_c(j)$ um reconhecimento positivo ao pedido-de-commit(j). Se C1 não for satisfeita E_s deve enviar um reconhecimento negativo ao pedido-de-commit(j). Se C1 for satisfeita mas não C2, E_s deve primeiro recuperar a mensagem ou commit perdido, para depois, se ainda for possível, reconhecer positivamente ao pedido-de-commit(j).

Tratamento das mensagens de reconhecimento por $E_c(j)$

Uma vez difundido o pedido-de-commit(j), $E_c(j)$ fica aguardando o reconhecimento das outras estações. Duas situações podem ocorrer então:

- **$E_c(j)$ recebe** um ou mais **reconhecimentos negativos**: neste caso é necessário abortar a execução de protocolo j pois $E_c(j)$, tentou impor uma lista $L_{mc}(j)$, que não reflete o estado de engajamento das F_i das estações $E_i \subset G_L(j)$ (condição C1 violada: existem ponteiros não apropriados).
- **Expira o tempo de espera** (T_E) em $E_c(j)$ e os **reconhecimentos** recebidos são todos "**positivos**": neste caso tem-se ainda duas possibilidades. Na primeira a minoria das estações reconheceu positivamente ao "pedido-de-commit". Nesta situação, diz-se que não foi formado um "commit quorum" e portanto, $E_c(j)$ deve abortar o periodo de engajamento j (situação de fracasso), reiniciando-a após um timeout. Na outra situação (situação de sucesso), a maioria das estações reconheceu positivamente ao pedido-de-commit(j). Houve então a formação de um "commit quorum". A estação $E_c(j)$ pode enviar a mensagem commit(j), que determina o engajamento das mensagens, apontadas por $L_{mc}(j)$, a todas as estações. A mensagem commit(j) transfere também o token para a nova coordenadora ($E_c(j+1)$).

Mensagem de "commit"

A mensagem de commit difundida por $E_c(j)$, após o recebimento do reconhecimento da maioria das estações participantes deve carregar informações

importantes como identificadores de $E_c(j)$ e $E_c(j+1)$, número do commit e $L_{me}(j)$. Além destas informações, deve carregar também outras determinando atualizações nos grupos:

- **Grupo-R**: deve estar presente os identificadores das R estações que, além da coordenadora ($E_c(j)$), armazenarão as mensagens engajadas neste commit. As R estações escolhidas são as R primeiras estações a reconhecer o pedido-de-commit(j). Estas (R+1) estações serão as responsáveis pela restauração do commit(j) para estações que venham a solicitá-lo.
- **Grupo-Ausentes**: $G_A(j)$, isto é, contém nome das estações que se omitiram ao pedido-de-commit(j).
- **Grupo-falho**: $G_f(j)$, isto é, contém nome das estações que se omitiram das M últimas execuções de protocolo.
- **Grupo-Recuperando**: $G_r(j)$ contém nome das estações que enviaram pedido-de-recuperação-de-commit(j). Encontrando-se em processo de recuperação do commit(j) e/ou de anteriores.

Afim de garantir que, seja qual for a estação nova coordenadora ($E_c(j+1)$), esta estará habilitada a participar na recuperação de outras estações, todas estações ao receberem a mensagem "commit(j)" colocam-na em uma Fila-M-msg-commit, (figura 3.1) ordenada pelo número do commit e com capacidade M. A mensagem de commit(j) é fundamental para uma estação em recuperação poder localizar as estações que mantêm ainda cópias das mensagens engajadas na execução de protocolo j (o Grupo-R do engajamento).

As estações fazem também o engajamento das mensagens especificadas por $E_c(j)$ na ordem do campo $L_{me}(j)$. Caso a estação pertença ao grupo-R ,esta deverá ainda armazenar as L mensagens a engajar na "**Lista-Regeneração**". Estas mensagens serão mantidas armazenadas por M execuções de protocolo, quando então serão descartadas. As estações, ao receberem a mensagem "commit(j)", devem ainda retirar de $G_p(j)$ as estações pertencentes ao Grupo-falho $G_f(j)$.

Passagem do Token

A nova coordenadora ($E_c(j+1)$) será, a princípio a estação possuidora da última bufferização $B_1(j)$ que completará $B_L(j)$. Se no entanto, esta estação (a pretendida $E_c(j+1)$) não tiver respondido ao pedido-de-commit(j), a estação E_i ($E_i \in G_L(j)$) imediatamente anterior será tomada, então, como a nova coordenadora, desde que, tenha respondido ao pedido-de-commit(j). Neste caso, como o protocolo não admite que seja transferido o token a uma estação ausente, a pretendida $E_c(j+1)$ é retirada de $G_L(j)$ e suas mensagens de $B_L(j)$. As únicas mensagens, de estações ausentes, que serão desconsideradas pela mensagem "commit(j)" no engajamento j (na bufferização $B_L(j)$) serão das estações que, com a nova proposição de $E_c(j+1)$, ficarem fora do $G_L(j)$, delimitado a partir desta nova coordenadora . O protocolo também não admite que $E_c(j)$

transfira o token para si mesma. Caso as condições favoreçam tal decisão, a execução do protocolo deve ser abortada e reiniciada após um timeout. A condição para a não ruptura do anel lógico é que:

$$E_c(j),\ E_c(j+1) \in \{G_L(j) \cup G_{EG}(j)\} \quad \wedge \quad E_c(j) \neq E_c(j+1) \qquad (3.2)$$

3.1.3 - Tratamento de exceções e robustez do algoritmo

Como é permitido a uma estação se ausentar de até M execuções de protocolo sem ser considerada faltosa (grau de omissão M) é necessário criar um suporte para recuperação desta estação. Assim, as mensagens engajadas em cada execução de protocolo m, serão mantidas armazenadas por R estações além da $E_c(m)$ durante M execuções de protocolo. Com isto, mesmo que R estações venham a faltar, uma estação que tenha se ausentado de menos de M execuções de protocolo pode ainda vir a recuperar as mensagens e continuar operando sem a necessidade de reinserção ao sistema (sistema R-resistente). Afim de determinar se uma estação está em estado de "crash" foram criadas listas-de-observação, contendo os M-1 Grupos-Ausentes, $\{G_A(j-M+1), G_A(j-M+2), \ldots, G_A(j-1)\}$, um para cada um dos (M-1) últimos commits executados.

Uma estação E_i se encontrando nos (M-1) Grupos-Ausentes (G_A) da Lista-de-observância e estando fora dos últimos (M-1) Grupos-Recuperando (G_r) será considerada em "crash" e inserida em $G_f(j)$. Será retirada de $G_p(j+1)$, fazendo parte de $G_F(j+1)$:

$$G_f(j) = \left\{ E_i \mid E_i \in \bigcap_{m=j-M+1}^{m=j} G_A(m) \quad \wedge \quad E_i \notin \bigcup_{m=j-M+1}^{m=j} G_r(m) \right\}$$

$$G_F(j+1) = \left\{ E_i \mid E_i \in \bigcup_{m=1}^{m=j} G_f(m) \quad \wedge \quad E_i \notin \bigcup_{m=1}^{m=j} G_i(m+1) \right\}$$

No momento em que $G_A(m) = \varnothing$ tem-se a situação em que todas as estações ou recuperam o commit(m) ou foram retiradas do sistema, não havendo necessidade de continuar a armazenar as mensagens engajadas nas (M-1) execuções de protocolo anteriores a m. Portanto as R estações, que mantêm cópias destas mensagens na camada de difusão deverão descartá-las na situação de $G_A(m)$ sem elementos. Outra situação de descarte de mensagens pelo grupo-R de um commit é quando a distância deste em relação a execução de protocolo atual for maior que M.

3.1.4 - Reconfiguração do token

Após a estação E_i participar do grupo de estações-L de um engajamento (m) ($E_i \in G_L(m)$), é acionado um timeout, devendo a estação então participar novamente do grupo de estações-L antes deste tempo expirar. O timeout, denominado de T_{reconf}, deverá cobrir o anel "máximo", ou seja, na pior situação a estação E_i voltará a ser estação-L em NK/L execuções de protocolo. Assim:

$$T_{reconf} > \left(\frac{NK}{L} \times (\textit{tempo máximo de uma execução do protocolo}) \right)$$

A figura 3.4 ilustra as relações de tempo envolvidas na execução de um protocolo e o item 3.2.2 explica a faixa de valores deste tempo.

Se, após T_{reconf}, E_i não for novamente estação-L, E_i considera que o token foi perdido ($E_c(j)$ em crash), invocando então sua regeneração às demais estações pertencentes a $G_p(j)$. Para investigar a existência do token na rede e regenerar de forma única e correta o token, é usado o algoritmo de regeneração proposto por [Nishio 91].

3.2 - Desempenho e análise de parâmetros

Dos quatro parâmetros apresentados anteriormente (K, L, M, R), M e R são dependentes apenas dos recursos de memorização das estações envolvidas, de forma que, não determinam nenhuma sobrecarga (overhead) adicional ao algoritmo. O parâmetro R indica o número de estações que manterão, na camada de difusão confiável, as mensagens engajadas em uma execução de protocolo. Este parâmetro determina a robustez do algoritmo no sentido de se poder restaurar os engajamentos perdidos, em estações que tiveram ausentes, mesmo na presença de R "crash" de estações em G. O parâmetro M determina o número de execuções consecutivas de protocolo que uma estação pode se ausentar sem ser tida como faltosa, isto é, M determina o grau de omissão do sistema.

Os parâmetros K e L estão relacionados com as filas F_i e F_L respectivamente e, conseqüentemente com as bufferizações $B_i(j)$ e $B_L(j)$. O desempenho do protocolo depende destes valores e suas relações. Pelas condições de circulação do token e baseado na relação (3.2) em que a coordenadora $E_c(j)$ não pode continuar com o token na execução de protocolo (j+1), tem-se que:

$$(K + 1) \leq L \leq NK \qquad (3.3)$$

o limite inferior garante que não serão engajadas somente mensagens de $E_c(j)$. O limite superior garante que L é inferior ou igual a capacidade de armazenamento do anel (NK), e logo, não haverá a circulação completa do anel em uma passagem do token.

3.2.1 - Sobrecarga de mensagens de controle

Sob condições normais de execução de protocolo, com formação de "commit quorum", em uma rede de topologia física qualquer, envolvendo N estações participantes e ligações ponto-a-ponto (o pior caso) o número máximo de mensagens-protocolo inclui:

- N mensagens "pedido-de-commit"
- N mensagens de reconhecimento positivo do pedido-de-commit
- N mensagens "commit"

o número total de mensagens trocadas nestas condições será igual a 3N mensagens-protocolo, isto é, 3N mensagens-protocolo trafegam pela rede por execução de protocolo.O overhead então, é dado pelo número de mensagens-protocolo que trafegam pela rede, por mensagem engajada. Ou seja,

$$overhead = \frac{3N}{L}$$

Logo, quanto maior for a capacidade de engajamento por execução, menor o custo de overhead provocado pelo protocolo. Com base em (3.3) que define os limites de variação de L, tem-se então:

$$\frac{3}{K} \leq overhead \leq \frac{3N}{K+1}$$

Assim, o overhead é inversamente proporcional ao valor de K.

3.2.2 - Latência

Para caracterizar a latência máxima de uma mensagem, ou seja, o tempo máximo que esta mensagem pode permanecer na camada de difusão confiável, antes de seu engajamento, são introduzidas as definições de alguns tempos relacionados com a execução do protocolo:

t_{A_j} : tempo onde inicia, em uma execução de protocolo j, a ativação do processo de engajamento das mensagens propostas para esta execução.

t_{EGj}: tempo de duração do periodo de engajamento j.

t_{EPj}: tempo de duração da execução de protocolo j.

A figura 3.4 mostra estes tempos, de onde obtem-se a relação: $t_{EPj} = t_{A_j} + t_{EG_j}$

Neste protocolo, são assumidas as seguintes condições limites dos tempos :

T_A : é o "timeout" para a ativação de um periodo de engajamento.

$$0 \leq t_{Aj} \leq T_A \tag{3.4}$$

T_E : é o "timeout" limitando a espera de respostas ao pedido-de-commit pela coordenadora

$$T_E < t_{EGj} \tag{3.5}$$

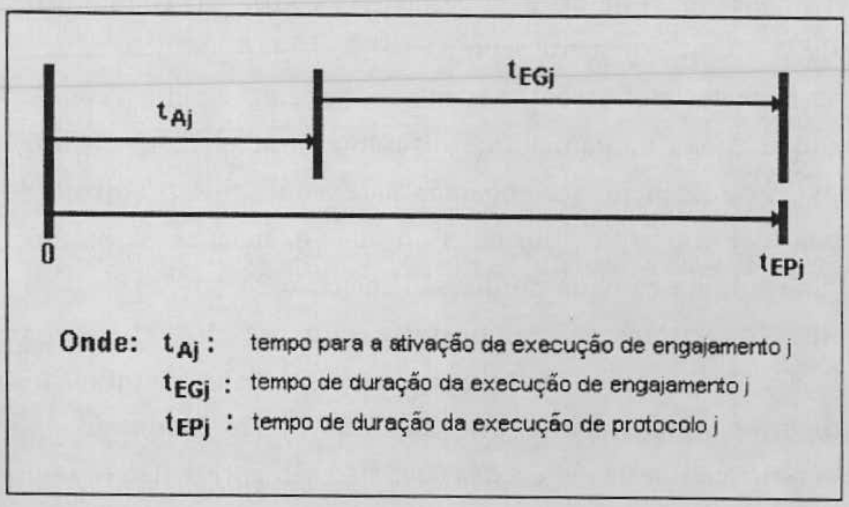

Figura 3.4 - Tempo de uma execução de protocolo

Para que se tenha condições de caracterizar valores para estes limites, é necessário que se faça algumas considerações sobre o suporte de comunicação. A rede de topologia física é qualquer com ligações ponto-a-ponto (pior caso). É assumido que o grafo de comunicações de rede, restrito aos componentes corretos (estações) é sempre conectado. Sejam D o máximo diâmetro do subgrafo correto do sistema de comunicação e d o atraso em um enlace entre duas estações, teremos então:

$$\delta = D \cdot d \tag{3.6}$$

onde δ é o valor de tempo máximo gasto na transferência de mensagens no suporte de comunicação. Numa análise de pior caso podemos então assumir que:

$$t_{EGj} \leq 3N\delta \tag{3.7}$$

Considerando a situação sem exceção, com as N estações respondendo ao pedido-de-commit(j) no periodo limitado por T_E , pode-se dimensionar T_E em:

$$T_E = 2N\delta \tag{3.8}$$

onde 2N corresponde às mensagens de pedido-de-commit(j) e as respectivas respostas, emitidas dentro do limite T_E. Com base nas relações (3.5), (3.7) e (3.8) e na condição

das N estações participantes, assumiu-se então: $t_{EGj} = T_E + N\delta$ onde $N\delta$ representa a emissão de N mensagens de commit. Como o segundo membro é uma constante, introduz-se então, T_{EG} como o tempo limite de engajamento:

$$t_{EGj} = T_{EG} = T_E + N\delta = 3N\delta \qquad (3.9)$$

Em relação ao dimensionamento de T_A (timeout para a ativação do engajamento), é levado em conta as necessidades de tempo para a ocupação da capacidade de engajamento (L mensagens) na camada de difusão confiável. Diante disto, é assumido: $T_A = \sigma L\delta$ onde σ é uma constante, com $\sigma > 1$

Com base nestas definições podemos estabelecer certas condições para uma mensagem $m_{i,k}$ que chega na camada de difusão confiável. Esta mensagem terá duas situações possíveis de latência: ser engajada na execução de protocolo de sua chegada na camada (cenário 1) ou ainda, em duas ou mais execuções de protocolo sucessivas, apartir de sua chegada na camada de difusão (cenário 2).

Assumindo os valores de L impostos pela expressão (3.3) e considerando $\sigma = 3N/(K+1)$ e $N \geq (K+1)/3$, obtem-se então os valores de latência indicados na tabela 3.1, para os dois cenários citados acima. Em [Dueñas 92] são encontrados todos os passos e provas para a obtenção destas equações (que não são apresentadas aqui por motivo de limitação de espaço).

	L = NK	L = K + 1
Cenário 1	$Latmax = N\left(\frac{K}{K+1}\right)T_{EG} + T_{EG}$	$2\,T_{EG}$
Cenário 2	$Latmax = N\left(\frac{K}{K+1}\right)T_{EG} + 2T_{EG}$	$Latmax = \left(\left[(N-1)\left(\frac{K}{K+1}\right)\right]\right)T_{EG} + 2T_{EG}$

Tabela 3.1 - Latência Máxima

Das equações abaixo (tabela 3.1) fica evidenciado que mesmo em situações onde o suporte de comunicação é sobrecarregado, a opção de L pequeno (L = K+1) não representa uma situação de latência pior que as de L=NK. Embora, em termos de overhead, o custo de mensagens de controle por mensagem engajada seja maior para L = K+1. Em termos de subutilização ou da simples ocupação da capacidade de engajamento (L), o desempenho do algoritmo para L = K+1 é muito superior se comparado com valores de L próximos do máximo (cenário 1). A figura 3.5 apresenta uma melhor comparação das distintas opções de escolha dos parâmetros de projeto Lat1 e Lat2 nesta figura se referem aos cenários 1 e 2 respectivamente.

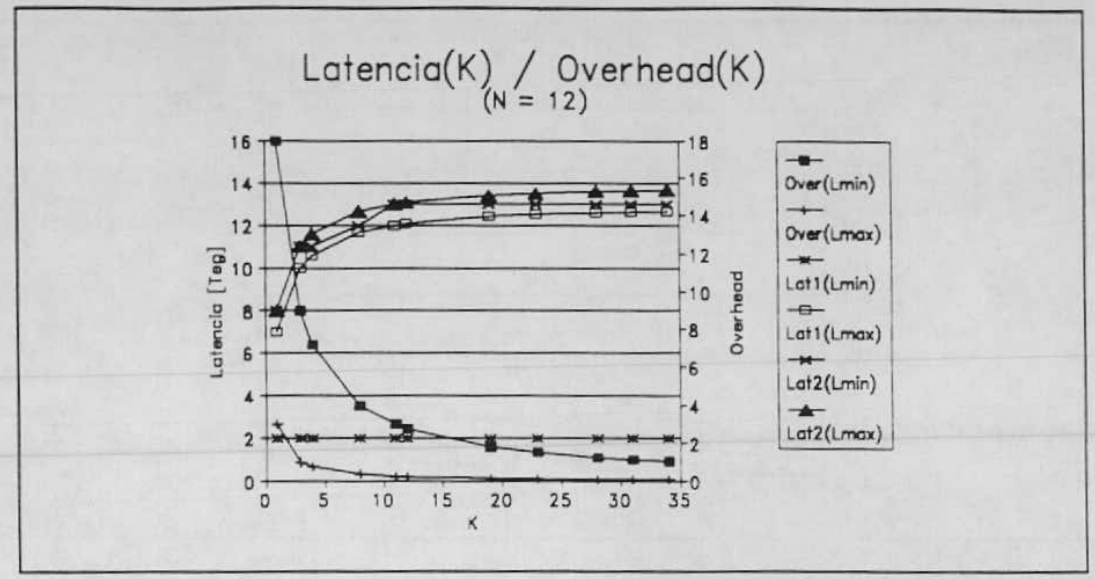

Figura 3.5 - Latência [T_{EG}]/Overhead x K

4 - Comparação

Critérios para a escolha de um protocolo estão normalmente ligados a aspectos de custo. Neste sentido, o algoritmo apresentado é analisado juntamente a alguns dos mais relevantes protocolos presentes na literatura. O estudo comparativo realizado em [Dueñas 92] tem como base a latência e o overhead dos protocolos. Neste estudo são consideradas para os protocolos condições idênticas de carga (número de mensagens a engajar é L).

Um aspecto não muito preciso na literatura é o referente a limites de tempos ("timeouts"). O dimensionamento destes tempos influenciam no desempenho destes algoritmos. Por exemplo, no protocolo de [Birman 87], o tempo limite para a recepção dos N timestamps é mais critico que o similar no protocolo proposto, para a espera dos reconhecimentos aos "pedidos-de-commit", visto que a decisão no primeiro se dá com a presença das respostas de todas as estações operacionais, sem omissões. No estudo comparativo realizado em [Dueñas 92] os tempos, a exemplo da carga, também foram assumidos com valores idênticos para os diferentes protocolos.

A latência e o "overhead" dos protocolos são apresentados na tabela 4.1. Nas figuras 4.1 e 4.2 são assumidos L = N e R = 3, onde N é o número de estações participantes do grupo, R é o limite de faltas possíveis de tolerância e δ é o tempo de transmissão, no pior caso entre dois processadores.

Neles pode-se observar que o protocolo proposto possui um excelente overhead (mensagens-protocolo por mensagens difundidas pela aplicação), mesmo quando comparado com os demais, e a sua latência máxima é da mesma ordem dos protocolos mais significativos apresentados na literatura, superando em desempenho, alguns deles.

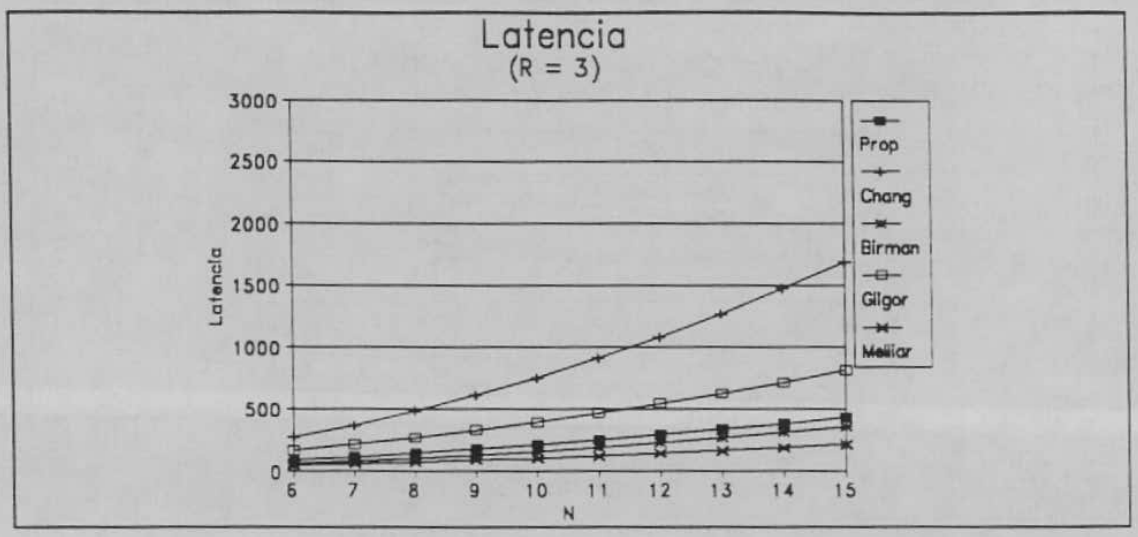

Figura 4.1- Latência [δ] x Número de estações para R=3

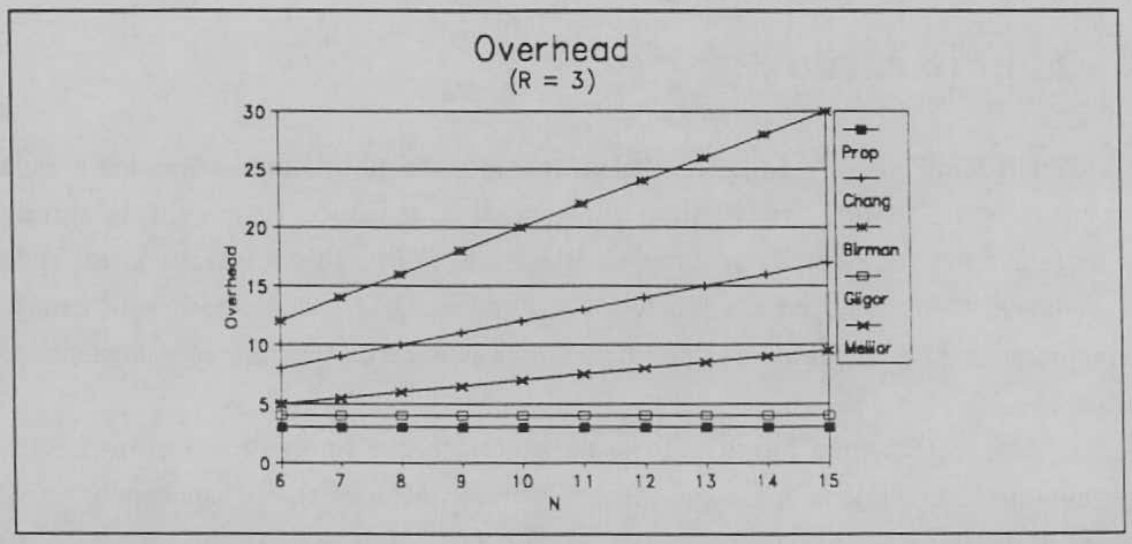

Figura 4.2 - Overhead x Número de estações, para R=3

5 - Comentários sobre o protocolo proposto

Este trabalho apresenta um algoritmo que, como [Chang 84], [Melliar-Smith 90] e outros se enquadra na classe de protocolos de difusão atômica assincrona. No entanto, seu tempo máximo por execução de protocolo (T_A+T_{EG}) é limitado e conhecido, bem como o tempo máximo que, uma mensagem espera, a partir de sua difusão, para ser engajada. Este tempo de latência da mensagem, para a pior situação e considerando $T_A = (L/L_{min}\ T_{EG})$, tende a $(N+2)T_{EG}$, em um grupo com N participantes onde $N \leq K \leq (3N-2)$ e T_{EG} é o tempo de uma execução de engajamento. Como T_{EG} varia linearmente com o número de mensagens-protocolo trocadas na execução de engajamento, a latência máxima de uma mensagem na pior das hipóteses é $O(N^2)$.

Um dos principais aspectos colocados, quando da elaboração desta proposta de protocolo, era tentar apresentar custos de mensagem-protocolo por mensagem engajada baixos. Este aspecto, para sistemas em tempo real, é importante já que, uma vez impedindo sobrecargas de mensagens de controle, o meio de comunicação ficaria mais disponível para mensagens de aplicação (inclusive para comunicação ponto a ponto), tão importantes para os atendimentos de tempos de respostas e deadlines. Neste sentido, protocolos como o de [Luan 90] e [Melliar-Smith 90] serviram de modelo. Assim, o engajamento se faz com L mensagens por vez e os custos de mensagens de controle de cada engajamento são distribuídas entre estas L mensagens. Em vista disto, quanto maior o L, menor é o overhead apresentado pelo algoritmo.

L=NK K=1	Protocolo proposto	Chang	Cristian	Birman	Gligor	Melliar-Smith
Overhead [1]	3	[N, N+2]	0	2N	4	$\left[0,\left(\frac{N+R+1}{2}\right)\right]$
Latência[2]	$\frac{N+4}{2}3N\delta$	$2{,}5RN^2\delta$	$2(N-1)\delta$	$(1{,}5N+1)N\delta$	$(3N^2+8N+4R)\delta$	$\left(\frac{N+R+1}{4}\right)3N\delta$

[1]Overhead é o número de mensagens de protocolo necessárias para a difusão de uma mensagem
[2]N é o número de estações participantes e R a resistência do sistema

Tabela 4.1 - Latência máxima dos protocolos comparados

O protocolo proposto é limitado, embora assíncrono. Seu desempenho foi apresentado no item anterior, na forma de latência das mensagens, para diferentes situações. O uso deste protocolo em uma rede local tipo Ethernet, conforme os valores calculados para a latência em [Dueñas92] (50 [ms] a 392 [ms]), demonstram a aplicabilidade deste protocolo em aplicações de tempo real, onde os tempos de resposta e deadlines envolvidos não sejam inferiores ao limite máximo da latência.

A escolha da forma de ordenação no engajamento das L mensagens, foi no sentido da flexibilização, evitando a utilização de mecanismos globais criados a partir de recursos distribuídos para a obtenção de uma ordenação total estrita. Evitando com isto os custos no sentido da manutenção da coerência nas ordenações obtidas nestes meios, bastantes suscetíveis aos atrasos variáveis e baixa confiabilidade dos suportes de comunicação. Por isto a ordenação proposta, baseada em ordenações locais e na disposição das estações emissoras no anel virtual, é mais barata em termos de custos de mensagens e material necessário quando comparado com outras técnicas utilizadas (sincronização de relógios, técnicas de consenso, etc ...)

O protocolo foi projetado no sentido de tolerar a presença de faltas de omissão e crash. O acordo neste protocolo é obtido com a maioria dos participantes corretos. O

tratamento de faltas neste protocolo (recuperação de commits) não implica em alterações da evolução normal do algoritmo para estações corretas. As estações em processo de recuperação do estado de commit, executam os engajamentos perdidos em paralelo com a execução atual do protocolo para as demais. Este ponto é bastante importante se comparado com protocolos como o de [Birman 87] e de [Chang 84] onde aspectos de detecção, tratamento e recuperação de exceções implicam em acréscimos significativos no tempo de execução do protocolo. O protocolo proposto a exemplo de [Chang 84] necessita de reconfiguração do anel quando a estação possuidora do token apresenta uma falta de "crash". Porém os custos envolvidos na reconfiguração são bem menos expressivos no proposto.

Como ainda existem dificuldades de aplicar a técnica de verificação em especificações do tamanho encontrado na maioria das aplicações práticas, o protocolo proposto foi descrito na FDT Estelle* [ISO 9074], [Courtiat 87], e simulado por análise interativa da especificação através da ferramenta ESTIM desenvolvida no LAAS-CNRS [Saqui 90].

No modelo do suporte de comunicação foi criado um modelo Estelle chamado de perturbação que gerava em vários cenários as seguintes faltas:

- Perda de mensagens da aplicação
- Perda de mensagens de "pedido-de-commit"
- Perda de mensagens de "commit"
- Perda de mensagens de "pedido-de-commit" e "commit"

A simulação embora não sendo um modo exaustivo de verificação, permitiu que, explorando caminhos particulares do grafo de alcançabilidade, se observasse a conformidade do protocolo proposto com o serviço desejado pelos usuários de um broadcast atômico. A ferramenta ESTIM permite estabelecer estatisticas durante as sessões de simulação, permitindo se conhecer todas as transições disparadas, as mais requisitadas e as que nunca ocorreram. Com base nestas estatisticas, ocorridas durante as simulações, pode-se garantir que as especificações satisfizeram as propriedades de ordenação, acordo e terminação do protocolo.

6 - Conclusão

Foi apresentado neste artigo uma proposta de um protocolo de difusão confiável. O protocolo proposto possui uma terminação assincrona, mas com tempo máximo de terminação limitado e conhecido. Uma análise comparativa com os protocolos mais representativos mostrou resultados favoráveis ao protocolo proposto. Enquanto a maioria dos protocolos analisados exibiam um crescimento linear do overhead com o número de mensagens engajadas, o proposto apresentou um overhead distribuído entre o número de mensagens difundidas. A latência revelou-se da mesma ordem dos mais significativos.

Este trabalho foi desenvolvido no LCMI/UFSC e se encaixa na linha de desenvolvimento de suportes para a implementação de tolerância a faltas através de modelos de alta replicação de processos.

7 - Bibliografia

[Birman 87] Birman K.P. e Joseph T.A., "Reliable Comunication in the Presence of Failures"; *ACM Transaction on Computer Systems*, vol5, No 1, Feb 1987, 47-76

[Chang 84] Chang Jo-Mei, Maxemchuk N.F.,"Reliable Broadcast Protocols"; *ACM Transaction on Computer Systems*, vol 2, No 3, August 1984, 251 - 273

[Courtiat 87] Courtiat J.P., Dembinski P., Groz R., Jard C.,"Estelle: An ISO Language for Distributed Algorithms an Protocols", *TSI*, mai, 1987

[Cristian 85] Cristian F, Aghili H, Strong, Dolev, "Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement"; *FTCS15*, Ann Arbor, USA, june 1985

[Dueñas 92] Dueñas L.T.R.;"Uma Proposta de Algoritmo Assincrono para Difusão Confiável Atômica", tese de mestrado, *LCMI-UFSC*,dezembro 1992

[Fraga89] Fraga J.S, Farines J.M, Abreu W, Silva E.S, Nacamura J.L, Coelho C.O; "ADES: Ambiente de Desenvolvimento e Execução de Software Distribuído"; *Seminário Franco Brasileiro em Sistemas Informáticos Distribuídos*, Florianópolis, Brasil, setembro 1989

[ISO 9074] "Estelle, a Formal Description Technique Based on an Extended State Transition Model"; *ISO - DIS 9074*, june 1987

[Lamport 82] Lamport, Shostak, Pease, "The Byzantine General Problems";*ACM Transaction on Programming Languages and Systems*,vol4,No3,July 1982,382-401

[Luan 90] Luan SW, Gligor VD,"A Fault-Tolerant Protocol for Atomic Broadcast"; IEEE Transaction on Parallel and Distributed Systems,vol1,No3,July 1990,271-285

[Melliar-Smith] Melliar-Smith PM,Moser LE,Agrawala V,"Broadcast protocols for Distributed Systems"; *IEEE Transaction on Parallel and Distributed Systems*, vol1, No"1, January 1990, 17-25

[Nishio 91] Nishio S, Li KF, Manning EG;"A Resilient Mutual Exclusion Algorithm for Computer Network", *IEEE Transaction on Parallel and Distributed Systems*,vol1,No3,July 1990, 344-355

[Saqui 90] Saqui-Sannes P,"The ESTIM User Manual for release 4.0 on Sun3 & Sun4 machines", november 1990

[Schneider 90] Schneider S, *ACM Computing Surveys*, vol22, No4, december 1990

[Shivastava 91] Ezhilchelvan P.D., Shivastava S.K., "A Distributed Systems Architecture Suporting High Availability and Reliability"; *Proc 2th International Working Conference on Dependable Computing for Critical Applications*, Tucson, Arizona, USA, February 1991, 36 - 48.

[Verissimo 89] Verissimo p, "Comunicação em Grupo Fiável em Sistemas Distribuídos sobre Rede Local", Tese de doutorado, *INESC*, Portugal, Dezembro 1989