Capítulo 4: Estado Global

Ajay Kshemkalyani e Mukesh Singhel

Distributed Computing: Principles, Algorithms, and Systems

Cambridge University Press

Introdução

Gravar o estado global de um sistema distribuído é um importante paradigma.

A falta de memória compartilhada, relógio global, tempo de entrega das mensagens imprevisível faz deste um problema nada trivial.

Neste capítulo, primeiro, o estado global é definido e depois são discutidas técnicas de gravação do estado global.

Diferentes tipos de algoritmos para gravação do estado global são apresentados para diferentes redes.

Modelo do Sistema

O sistema consiste em um conjunto de **n** processos **p1, p2, ...,pn** que são conectados por canais.

Não há memória compartilhada nem relógio físico global e os processos só se comunicam através de trocas de mensagens.

$\mathbf{C_{ij}}$ denota o canal do processo pi para o processo **pj** e seu estado é denotado por $\mathbf{Sc_{ij}}$

A ação tomada por um processo é modelada em três formas de evento: evento interno, evento de envio e evento de recebimento.

Para a mensagens $\mathbf{m_{ij}}$ que é enviada pelo processo pi para o processo **pj**, definiremos **send($\mathbf{m_{ij}}$)** e **rec($\mathbf{m_{ij}}$)** os eventos de envio e recebimento respectivamente.

Modelo de Sistema

A qualquer instante, o estado do processo pi, denotado por $\mathbf{Ls_i}$, é o resultado da seqüência de todos os eventos executados por pi até este instantes.

Para um evento e e um estado de processo $\mathbf{Ls_i}$, $\mathbf{e \in Ls_i}$ se e pertencer ao conjunto de eventos que levou pi até o estado $\mathbf{Ls_i}$.

Para um evento e e um estado de processo $\mathbf{Ls_i}$, $\mathbf{e \notin Ls_i}$ se se e não pertencer ao conjunto de eventos que levou pi até o estado $\mathbf{Ls_i}$.

Para o canal $\mathbf{C_{ij}}$, o seguinte conjunto de mensagens pode ser definido baseado no estado local dos processos pi e pj

Transito: $\mathbf{transit(Ls_i, Ls_j) = \{m_{ij} \mid send(m_{ij}) \in Ls_i \wedge rec(m_{ij}) \notin Ls_i\}}$

Modelo de Comunicação

Relembrando os três modelos de comunicação:FIFO, não FIFO e OC.

No modelo FIFO. Cada canal atua como um fila Primeiro a entrar primeiro a sair, logo a ordenação de mensagens é mantida.

No modelo não-FIFO, os canais atuam como um conjunto no qual o remetente coloca mensagens e o destinatário as recebe de forma aleatória.

Um sistema que suporta Ordenação causal de mensagens satisfaz a propriedades abaixo:

Para cada duas mensagens m_{ij} e m_{kl}, se $send(m_{ij}) \rightarrow send(m_{kl})$ então $rec(m_{ij}) \rightarrow rec(m_{kl})$

Estado Global Consistente

O estado global de um sistema distribuído é um conjunto de estados locais dos processos e o estado de seus canais.

Estado global **GS** será definido como

$$\mathbf{GS = \{\cup_i LS_i, \cup_{ij} SC_{ij}\}}$$

Um estado global GS é consistente se satisfizer as duas condições a seguir

C1: $\mathbf{send(m_{ij}) \in LS_i \Rightarrow mij \in SC_{ij} \oplus rec(m_{ij}) \in LS_j}$

C2: $\mathbf{send(m_{ij}) \notin LS_i \Rightarrow \ mij \notin SC_{ij} \wedge rec(m_{ij}) \notin LS_j}$

Interpretação em termos de Cortes

Um corte no diagrama de espaço tempo é uma linha que une pontos arbitrários nas linhas de progresso de dois processos criando os espaços **PASSADO** e **FUTURO**.

Um estado global consistente corresponde a um corte no qual cada mensagens recebida no **PASSADO** foi enviada no **PASSADO**.

Por exemplo, considerando o diagrama de espaço-tempo ilustrado na figura 4.1

O corte **C1** é inconsistente porque a mensagem **m1** segue do **FUTURO** para o **PASSADO**.

O corte **C2** é consistente e a mensagem **m4** tem de ser capturada no estado do canal $\mathbf{C_{12}}$

Modelo de Execução Distribuída

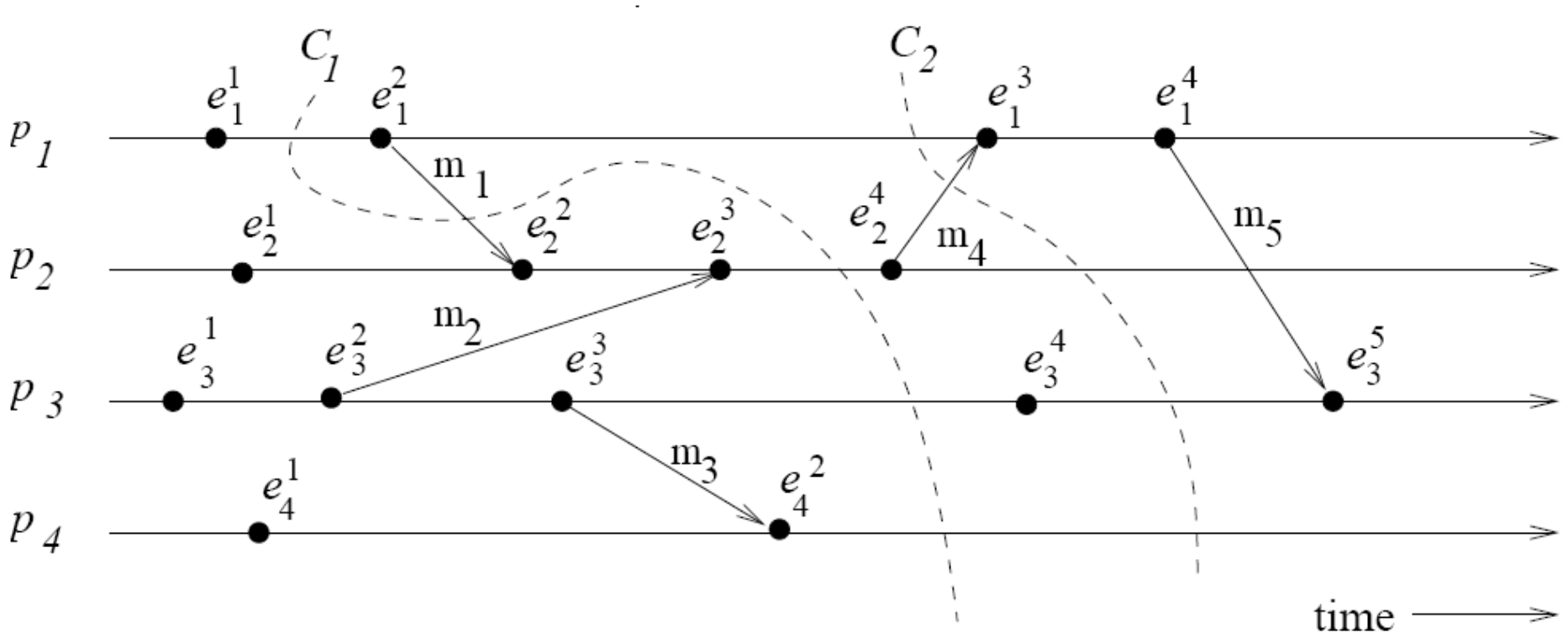

Figura 4.1 Interpretação em termos de Cortes

Pontos de atenção

Na gravação do estado global temos dois pontos de destaque

1: Como distinguir entre as mensagens que devem e as que não devem ser gravadas no estado global.

- Qualquer mensagem que é enviada por um processo antes da gravação do estado global tem de ser gravada.
- Qualquer mensagem que é enviada por um processo depois da gravação do estado global não pode ser gravada.

2:Como determinar o instante que o processo inicia a gravação do estado global

- O processo **pj** deve começar a gravar seu estado global antes de processar a mensagem $\mathbf{m_{ij}}$ que foi mandada pelo processo **pi** após o inicio da gravação de seu estado global

Gravação de estado global para canais FIFO

Algoritmo Chandy-Lamport

O algoritmo Chandy-Lamport utiliza uma mensagem de controle chamada ***Marker***, cuja finalidade em sistemas FIFO é separar mensagens por canais.

Depois de gravar um estado global, um processo envia um ***Marker*** por todos os seus canais de saída antes de enviar mais mensagens de computação.

O ***Marker*** separa as mensagens que tem que ser incluídas no estado global e quais não devem ser incluídas

Um processo começa a gravar seu estado global assim que receber um ***marker*** por qualquer um de seus canais de entrada.

Algoritmo Chandy-Lamport

O algoritmo pode ser inicializado por qualquer processo que execute uma “regra de envio de ***marker***”, onde o processo grava seu estado local no estado global e envia um ***marker*** para cada canal de saída.

Um processo que ainda não tenha iniciado a gravação do estado global, ao receber um ***marker*** por um canal inicia a “regra de recebimento do ***marker***”, onde o processo marca o canal de entrada do ***marker*** como vazio. Grava seu estado local no estado global e envia um ***marker*** por cada um de seus canais de saída,

O algoritmo termina quando um processo recebe um ***marker*** por cada um de seus canais de entrada.

Todo o estado global é disseminado e todos os processos podem determinar o estado global.

Marker Sending Rule for process i

1. Process i records its state.
2. For each outgoing channel C on which a marker has not been sent, i sends a marker along C before i sends further messages along C.

Marker Receiving Rule for process j

On receiving a marker along channel C:

if j has not recorded its state **then**

Record the state of C as the empty set

Follow the "Marker Sending Rule"

else

Record the state of C as the set of messages received along C after j's state was recorded and before j received the marker along C

Corretude e complexidade

Corretude

Devido as propriedades dos canais FIFO, nenhuma mensagem enviada após um ***marker*** naquele canal será gravada como pertencendo ao estado global. Desta forma a condição **C2** é mantida.

Quando o processo **pj** recebe a mensagem $\mathbf{m_{ij}}$ que precede o ***marker*** no canal $\mathbf{C_{ij}}$ uma das duas coisas acontecem: Ou o processo **pj** não começou a gravação de seu estado global logo a mensagem $\mathbf{m_{ij}}$ não é gravada. De outra forma $\mathbf{m_{ij}}$ é colocada no estado do canal $\mathbf{C_{ij}}$, logo a condição **C1** é mantida.

Complexidade

Um única instancia do algoritmo rodando requer complexidade de mensagens **O(e)** complexidade de tempo de **O(d)**, onde **e** é o número de arestas do grafo da aplicação e **d** o diâmetro deste grafo.

Propriedades do Estado Global

- O estado global gravado pode não corresponder a nenhum estado global que realmente ocorre durante a computação.
- Isso ocorre porque um processo pode modificar seu estado de forma assíncrona antes de o ***marker*** ser enviado por outro processo e este gravar seu estado.
  - Mas o sistema pode ter passado por este estado em uma execução anterior
  - O estado global é um estado válido em uma execução anterior e se uma propriedade estável se apresenta no sistema antes do estado global ser gravado, será apresentado no estado global.
  - Logo, gravação do estado global é útil em detectar propriedades estáveis.

Estado Global em sistemas não FIFO

- Em um sistema não **FIFO**, um ***marker*** não pode ser utilizado para delimitar mensagens que devem ou não ser colocadas no estado do canal.
- Em sistemas não **FIFO**, deve haver algum tipo de inibição de envio de mensagens fora de seqüência ou mecanismo de ***piggyback*** para detectar mensagens fora de seqüência

Algoritmo de Mattern

O algoritmo de Mattern é baseado em relógios vetores e assume que é iniciado apenas por um mestre.

O mestre atualiza seu relógio vetor e seleciona um tempo futuro ***s*** no qual gostaria de iniciar a gravação do estado global. Então ele envia via ***broadcast*** o vetor s e para todas as atividades até receber a confirmação do termino ***broadcast*** (***feedback***).

Quando um processo recebe o valor s, este retorna o ***feedback*** do ***broadcast*** para o mestre.

Depois de receber o ***feedback*** de todos os processos o mestre eleva seu relógio vetor até s e envia via ***broadcast*** uma mensagem de controle a todos os processos.

Cada processo que recebe a mensagem de controle força seu relógio vetor a ser maior que s, se este já não o for,

Algoritmo de Mattern

Cada Processo grava seu estado local e o manda para o mestre no momento (logo após) seu relógio aumenta de um valor menor que s para um valor maior que s

O estado de $\mathbf{C_{ij}}$ é composto de todas as mensagens enviadas para $\mathbf{C_{ij}}$ cujo o ***timestamp*** é menor que s e foi recebido por **pj** depois da gravação do estado global $\mathbf{LS_j}$.

Algoritmo de Mattern

- Um algoritmo de terminação para canais não FIFO é necessário para detectar que nenhuma mensagem de computação está em transito
- Um dos esquemas a seguir pode ser utilizado para detectar terminação

Primeiro método:

- Cada processo i mantém um contador que indica a diferença entre o número de mensagens que foram enviadas e recebidas antes de iniciar a gravação do estado global.
- O processo envia para o mestre a lista de mensagens de computação faltantes junto com a gravação do estado global. Todas as menagens faltantes são posteriormente enviadas ao mestre ao serem recebidas.
- O mestre termina ao receber todas as mensagens faltantes.

Algoritmo de Mattern

Segundo método

- Cada mensagem de gravação de estado global carregam via ***piggyback*** o valor de mensagens de computação enviadas no canal entes de iniciar a gravação do estado global.
- Cada processo mantém um contador de mensagens de computação recebidas em cada canal.
- Um processo sabe que a gravação de estado global terminou quando o número de mensagens de computação recebidas é igual ou maior que o número recebido via ***piggyback***.