Ordenação

Sistemas Distribuídos e Tolerância a Falhas

Baseado no trabalho de:
Benjamim Marques M1440
Daniel Félix M1441
João Duarte a14951

UBI 2008

Índice

- Introdução
- FIFO
- Ordenação Causal
- Ordenação Total
- Algoritmos

Introdução

- Ordenação
  - Objectivo – Determinar à posteriori a ordem pela qual um conjunto de eventos ocorreu.

➔ Permite indicar quais os eventos que ocorreram primeiro e assumir, ou excluir, relações de causa efeito entre eles.

➔ Exemplo de aplicação: se fizermos o registo da ordem pela qual um conjunto de eventos ocorre, poderemos repetir uma computação não determinística

Introdução (cont.)

- Ordenação
  - Objectivo – Garantir que um conjunto de eventos ocorra segundo uma ordem pré-determinada.

➔ Pode ser conseguido por protocolos de entrega ordenada de mensagens.

Noção de Ordem

- A noção mais intuitiva é a ordem física, em que os eventos ocorrem numa linha de tempo real.
- Esta ordem pode ser capturada se for atribuído a todos os eventos um "timestamp" com o valor de um relógio global.

Noção de Precedência

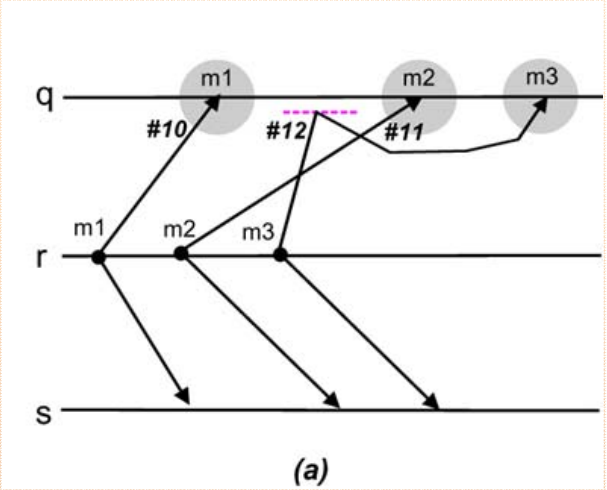
- Se *a* e *b* são eventos no mesmo processo e *a* ocorre antes de b, dizemos que $a \rightarrow b$ *(a precede b).*
- Se *a* é o envio da mensagem *m* e *b* é a recepção da mensagem *m*, então $a \rightarrow b$.
- Se $a \rightarrow b$ e $b \rightarrow c$, então $a \rightarrow c$.

Ordem FIFO (First-In-First-Out)

- Quaisquer duas mensagens enviadas pelo mesmo processo são entregues a qualquer processo pela ordem de envio.
- A ordem FIFO é assegurada pela atribuição de um número de sequência local.
- O receptor entrega as mensagens à aplicação pela ordem dos seus números de sequência.

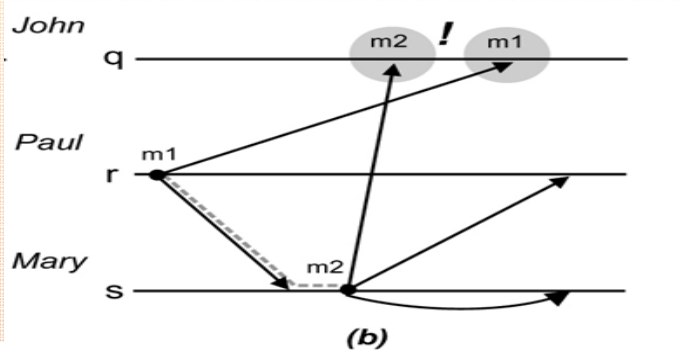
FIFO (continuação)

- Para isso o receptor deve ter um *buffer* temporário para as mensagens recebidas fora de ordem e caso exista alguma em falta pedir a retransmissão da mesma.



FIFO (continuação)

- A ordenação FIFO pode ser insuficiente.
- John está à espera de receber as mensagens por ordem de execução das tarefas.
- m1, m2 e m3 representam tarefas que deverão ser executadas em sequência
- !!!



Ordenação causal

- Consiste na garantia de que as mensagens enviadas por processos diferentes são entregues pela ordem correcta no receptor.
  - *Causal delivery*
    Se
    $envio_p(m) \rightarrow envio_q(n)$
    Então
    $entrega_r(m) \rightarrow entrega_r(n)$.
- A informação sobre a sequência lógica dos eventos é incluída nas mensagens.

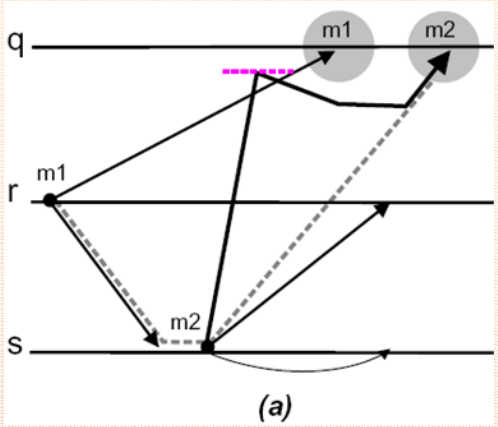
Ordenação causal (continuação)

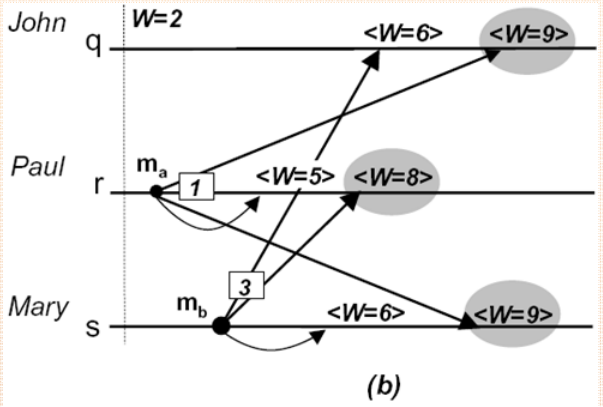
- Limitações:
  - Não garante que as mensagens dos processos concorrentes sejam ordenadas.

Ordem causal

Problema Causal

w <- max (w, v) +3



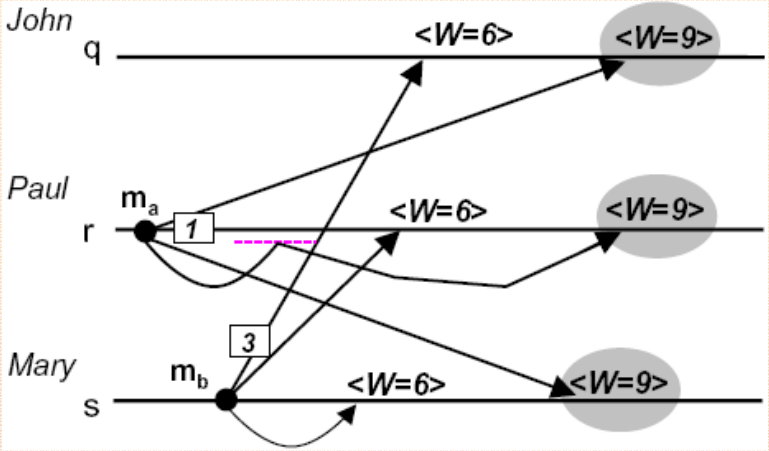


Ordenação Total

- Garante que quaisquer duas mensagens entregues a qualquer par de receptores são entregues na mesma ordem para ambos.
- Consiste na garantia que a entrega das mensagens se faz em todos os processos pela mesma ordem.
  - Se um processo entregar m1 antes de entregar m2, então qualquer outro processo que entregar m2 irá entregar antes m1.

Ordenação Total (continuação)

- Não implica nem a ordenação FIFO nem a Causal, a ordem de entrega pode ser arbitrária, desde que seja a mesma em todos os processos.



Algoritmos de ordenação Causal

- Objectivo:
  - Assegurar que as mensagens são entregues à aplicação de forma a respeitar a ordem causal, se m1 e m2 devem ser entregues ao mesmo processo e m1 precede m2 então m2 é entregue depois de m1.

Algoritmos de ordenação Causal (continuação)

- Uma das maneiras mais intuitivas é fazer com que cada mensagem carregue o sua própria lista "passado".
- Para isto é necessário que cada processo mantenha uma lista de mensagens enviadas.

Algoritmos de ordenação Causal (continuação)

- Protocolo:
  - Quando uma mensagem é enviada, leva a lista "passado" do seu processo emissor no campo de controlo.
  - Depois de enviar a mensagem, o emissor adiciona essa mensagem á sua lista.
  - Quando uma mensagem é recebida é verificado o seu campo de controlo. As mensagens que se encontram nessa lista que ainda não foram entregues podem ser imediatamente entregues mesmo que não tenham sido recebidas ainda.

Algoritmos de ordenação Causal (continuação)

- Depois de essas mensagens terem sido entregues à aplicação a mensagem recebida é adicionada a lista "passado" de mensagens recebidas do receptor.

- Isto garante que as mensagens enviadas serão todas entregues mesmo que as anteriores se percam, pois a última carrega todas as outras como garantia.

- Um ponto negativo deste protocolo é o tamanho exagerado do campo de controlo, pois este pode crescer indefinidamente podendo conter muitas mensagens. O que torna este protocolo impraticável.

Algoritmos de ordenação Causal (continuação)

- Deve ser completado com um mecanismo para eliminar informação obsoleta.
- Por exemplo uma mensagem que já foi entregue a todos os receptores pode ser descartada.

Algoritmos de ordenação Causal (continuação)

- Uma forma de resolver este problema é reduzir o tamanho do campo de controlo guardando na lista "passado" apenas os identificadores das mensagens, em vez das mensagens completas.
- Nesta ideia assume-se que um 3º componente é responsável pela garantia de entrega.
- Ou seja, se uma mensagem for perdida, é de alguma forma retransmitida até ser recebida por todos os receptores pretendidos.

Algoritmos de ordenação Causal (continuação)

- Nesta forma a diferença é que quando uma mensagem é entregue e o campo de controlo verificado, se alguma mensagem ainda não tiver sido entregue a última é posta em espera até todas as outras chegarem e serem entregues.
- Só depois a última é entregue e o seu identificador adicionado a lista “passado” do receptor.

Algoritmos de ordenação Total

- O objecto da ordenação total é assegurar que todas as mensagens são entregues a todos os receptores pela mesma ordem.

Algoritmos de ordenação Total

- Sequenciador
  - Consiste em seleccionar um processo especial e atribuir-lhe a tarefa de ordenar todas as mensagens.
  - Todos os emissores enviam as suas mensagens para o sequenciador.
  - Que por sua vez atribui um número de sequência único a todas mensagens e posteriormente irá retransmiti-las a todos os receptores pretendidos.

=> Problema: falha do sequenciador

- ...