

Mestrado em Engenharia Informática – Algoritmos e Sistemas Distribuídos

Primeiro teste – 6 de Novembro de 2012 – Duração: 2 horas

Responda às questões no espaço reservado para o efeito após cada alínea, ou na última página caso necessite mais espaço. Indique claramente qualquer pressuposto que tenha de fazer para resolver a uma questão.

Número	42006	Nome	Bruno Filipe Gonçalves Candeias
--------	-------	------	---------------------------------

1. [9 valores] Considere a seguinte especificação para o problema da difusão com ordem total (ou *total order broadcast*), em que um conjunto de processos podem ter inputs sob a forma “send(m)” e outputs sob a forma “deliver(m)”. Numa execução, cada processo pode ter um número arbitrário de inputs e outputs.

[OT1] Se um processo correcto enviar uma mensagem m (ou seja, tiver um input da forma send(m)), então todos os processos correctos fazem a certo ponto (“eventually”) output de deliver(m).

[OT2] Se um processo fizer output de deliver(m), então todos os processos correctos fazem a certo ponto (“eventually”) output de deliver(m).

[OT3] Uma mensagem m não pode estar repetida em mais do que um output do mesmo processo, e se um processo fizer output de deliver(m), então m foi anteriormente enviada.

[OT4] Se dois processos p e q fizerem ambos o output de duas mensagens: deliver(m) e deliver(m’), então o output de deliver(m) e deliver(m’) é feito pela mesma ordem no processo p e no processo q.

- (a) Para cada condição da especificação, indique se esta é uma condição de safety ou de liveness. (Responda apenas safety ou liveness junto a cada condição.)

OT1: Liveness OT2: Liveness OT3: Safety OT4: Safety

- (b) Considere o seguinte trace de uma execução sem falhas de dois processos p1 e p2. (A notação usada para descrever o trace indica, para cada elemento da sequência de inputs/outputs, o processo que faz o input ou o output, seguido de “:”, seguido do input ou output correspondente.)

p1:send(m1) , p2:send(m2) , p1:deliver(m3) .

Para cada condição da alínea anterior, indique se a condição é ou não verificada pelo trace (responda sim ou não e justifique sucintamente). No caso das condições de liveness que não são verificadas, escreva também uma extensão do trace onde esta passe a ser verificada.

OT1:	Não. (...), p1:deliver(m1), p2:deliver(m1), p1:deliver(m2), p2:deliver(m2)
OT2:	Não. (...), p2:deliver(m3)
OT3:	Não. É feito deliver(m3) sem nunca ser enviado.
OT4:	Sim.

- (c) O problema da **difusão com ordem total tem solução** num sistema assíncrono com canais fiáveis, supondo um total $n=2f+1$ processos onde $f \geq 1$ processos podem ter falhas por crash? Se sim, escreva o pseudo-código de um algoritmo que resolve o problema. Se não, demonstre que não existe solução para o problema.

```

P: all processes      lastAck      buffer

upon send_i(m)
  forall p in P
    send(p, <tag, m>)
  tag++
;

upon deliver_i(tag, m)
  if(tag > lastAck[p])
    forall j in P
      send(j, <tag, m>)
  if( tag == lastAck + 1)
    deliver(m);
    lastAck = tag
    deliver_in_order(lastAck)
  else
    buffer = buffer + {<tag, m>}
;

```

2. [8 valores] Considere o problema de implementar um variável de leitura/escrita atômica. Suponha que valor inicial da variável é zero.

- (a) Indique se os seguintes traces de execuções obedecem à atomicidade. Se sim, indique na figura possíveis pontos de serialização. Se não, justifique sucintamente porquê.

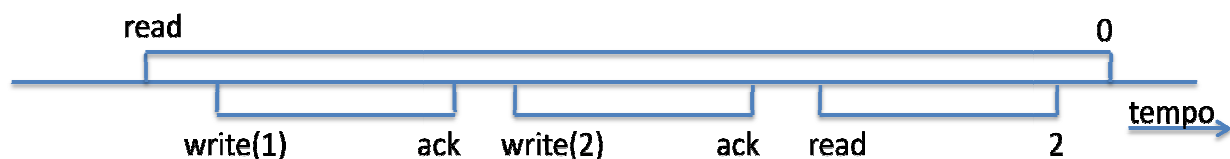
(a.1)



Atômico? Não

Justificação: A partir do primeiro momento em que o primeiro read lê 1, é porque o valor inicial já foi incrementado, ou seja, a escrita foi serializada antes do primeiro read. Logo, todos os reads subsequentes deverão retornar 1.

(a.2)



Atômico? Sim

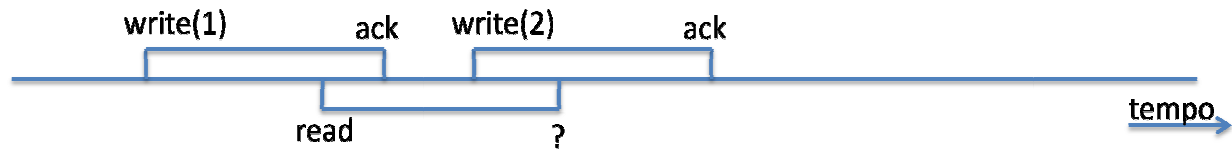
Número:

Nome:

Justificação:

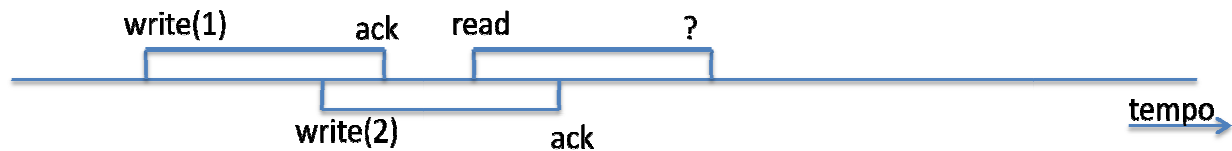
(b) Para os outputs indicados com '?', indique todos os outputs que seriam permitidos de forma a obedecer à condição de atomicidade.

(b.1)



Possíveis valores: 0, 1, 2

(b.2)



Possíveis valores: 1, 2

(c) Considere a adaptação do algoritmo ABD para o modelo de falhas Bizantinas leccionada na aula. Demonstre, através de um diagrama temporal da execução de um contra-exemplo, que a atomicidade não é garantida se reduzirmos o número de réplicas e a dimensão dos quóruns para $2f+1$ e $f+1$, respectivamente. (Note que deve manter os outros mecanismos usados no ABD para falhas Bizantinas, nomeadamente as assinaturas dos clientes sobre o par $\langle \text{tag}, \text{valor} \rangle$ escrito.)

R1 _____

R2 _____

R3 _____

C1 _____

C2 _____

3. [3 valores] Complete o seguinte código do yfs. Esta função incompleta lê, do ficheiro com o identificador *inum*, os bytes no intervalo [*offset*,*offset+nbytesread*] (caso existam), pondo o resultado em *buffer*. O valor *nbytesread* pode ter de ser ajustado para corresponder ao número de bytes que pode ser lido. (Não é necessário implementar a sincronização de acessos concorrentes.)

```
yfs_client::readfile(inum inumber, unsigned long long offset,
                    unsigned int nbytesread, std::string &buffer)
{
    std::string extent;

    if (  ) != extent_protocol::OK)
        return IOERR;

    char *strtmp = new char[nbytesread];
    memcpy(strtmp, extent.c_str() + offset, nbytesread);
    buffer = std::string(strtmp, nbytesread);
    delete strtmp;
    return OK;
}
```

[Espaço adicional para resolução do teste. Caso use este espaço, indique-o claramente no local inicial para a resolução.]