

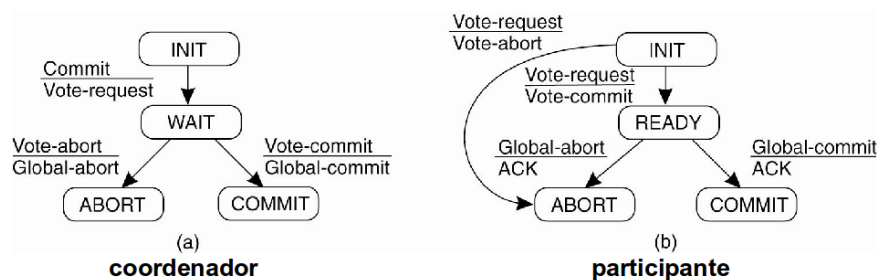
# Sistemas Distribuídos - COS470 2019/1

## Quinta Lista de Exercícios

**Dica:** Para ajudar no processo de aprendizado responda às perguntas integralmente, mostrando o desenvolvimento das respostas.

**Questão 1:** O protocolo *Two Phase Commit* (2PC) evita *deadlocks* em sistemas transacionais distribuídos? Explique sua resposta dando um exemplo, se for o caso. Em caso negativo, como podemos lidar com *deadlocks* nestes sistemas?

**Questão 2:** Considere o diagrama de transição de estados do protocolo *Two Phase Commit* (2PC):



1. Explique o que acontece quando um processo participante falha no estado INIT. Como o protocolo recupera desta falha?
2. Explique o que acontece quando um processo participante falha no estado READY. Como o protocolo recupera desta falha?
3. Explique o que acontece quando o coordenador falha no estado WAIT. Como o protocolo recupera desta falha?

**Questão 3:** Explique os conflitos *read-write* e *write-write* que surgem quando temos sistemas distribuídos com dados replicados.

**Questão 4:** Considere as seguintes execuções de instruções em diferentes processos, cada qual com sua memória local (assuma que inicialmente os valores das variáveis são zero). Determine quais casos (execuções) respeitam o modelo de consistência sequencial, indicando uma possível ordenação para as instruções.

- |  |   |
|--|---|
| 1. P1: W(x,1);<br>P2: R(x,0); R(x,1)                 | 5. P1: W(x,1);<br>P2: W(x,2);<br>P3: R(x,2); R(x,1);<br>P4: R(x,1); R(x,2);                                 |
| 2. P1: W(x,1);<br>P2: R(x,1); R(x,0);                | 6. P1: W(x,1); R(x,1); R(y,0);<br>P2: W(y,1); R(y,1); R(x,1);<br>P3: R(x,1); R(y,0);<br>P4: R(y,0); R(x,0); |
| 3. P1: W(x,1);<br>P2: W(x,2);<br>P3: R(x,1); R(x,2); | 7. P1: W(x,1); R(x,1); R(y,0);<br>P2: W(y,1); R(y,1); R(x,1);<br>P3: R(y,1); R(x,0);                        |
| 4. P1: W(x,1);<br>P2: W(x,2);<br>P3: R(x,2); R(x,1); |   |

**Questão 5:** Considerando novamente as execuções de instruções acima em diferentes processos, determine quais casos (execuções) respeitam o modelo de consistência causal, indicando uma possível ordenação para as instruções.

**Questão 6:** Em se tratando de sistemas tolerante a falhas, qual é a diferença entre disponibilidade (*availability*) e confiabilidade (*reliability*)? Dê um exemplo que ilustre as diferenças.

**Questão 7:** Considere um componente com  $MTTF = 2.5$  ano e  $MTTR = 32$  horas. Considere o uso de componentes redundantes para projetar um sistema cujo componente tem disponibilidade de 99.99%. Assuma que falhas nestes componentes são independentes.

- Determine o número de componentes redundantes necessários.
- Determine a probabilidade do sistema com redundância ter ao menos um componente em falha.

**Questão 8:** Considere a organização de componentes redundantes TMR (*Triple Modular Redundancy*). Explique o que ocorre nos seguintes casos:

1. Exatamente um componente e um votador falha (*crash failure*) em cada linha.
2. Dois votadores falham (*crash failure*) na mesma coluna.
3. Dois votadores falham (*value failure*) em colunas diferentes mas com o mesmo valor de saída.

**Questão 9:** Explique por que falhas do tipo *crash failure* são mais fáceis de lidar do que *value failure*. Dê um exemplo concreto.