PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e de Informática

Saulo de Moura Zandona Freitas[1]

Lista #8

Computação Distribuída

[1]Aluno de Graduação em Ciência da Computação– saulomzf@gmail.com

obs É necessário colocar os timestamps dos processos!! (*carimbo* = *C* = *clock* = *timestamp*)]Ordene todos os eventos do sistema distribuído abaixo usando relógios lógicos:
Processo A: inst A1; send C; recv B; inst A2; send D; recv D; inst A3;
Processo B: send A; inst B1; recv C; inst B2; recv D;
Processo C: inst C1; inst C2; recv A; inst C3; send B;
Processo D: inst D1; recv A; send A; inst D2; send B

obs É necessário colocar os timestamps dos processos!! (*carimbo* = *C* = *clock* = *timestamp*)

1 Regras Básicas para Relógios Lógicos

As seguintes regras são aplicadas para atribuir timestamps aos eventos:

1. Cada processo possui um relógio lógico que começa em 1.
2. O relógio é incrementado sempre que ocorre um evento local.
3. Quando um processo envia uma mensagem, ele anexa seu timestamp atual.
4. Quando um processo recebe uma mensagem, ele atualiza seu relógio para ser o maior valor entre seu relógio atual e o timestamp da mensagem recebida, incrementando em seguida.

2 Processos e Eventos

Vamos aplicar as regras mencionadas a cada processo e evento do sistema distribuído.

2.1 Inicialização

Cada processo começa com o valor 1 em seu relógio lógico:

- Processo A: $C(A) = 1$
- Processo B: $C(B) = 1$
- Processo C: $C(C) = 1$
- Processo D: $C(D) = 1$

2.2 Eventos e Timestamps

2.2.1 Processo A

1. **inst A1** $\rightarrow C(A) = 1$
2. **send C** $\rightarrow C(A) = 2$
3. **recv B** $\rightarrow C(A) = 3$
4. **inst A2** $\rightarrow C(A) = 4$
5. **send D** $\rightarrow C(A) = 5$
6. **recv D** $\rightarrow C(A) = 8$
7. **inst A3** $\rightarrow C(A) = 9$

2.2.2 Processo B

1. **send A** $\rightarrow C(B) = 1$
2. **inst B1** $\rightarrow C(B) = 2$
3. **recv C** $\rightarrow C(B) = 6$
4. **inst B2** $\rightarrow C(B) = 7$
5. **recv D** $\rightarrow C(B) = 10$

2.2.3 Processo C

1. **inst C1** $\rightarrow C(C) = 1$
2. **inst C2** $\rightarrow C(C) = 2$
3. **recv A** $\rightarrow C(C) = 3$
4. **inst C3** $\rightarrow C(C) = 4$
5. **send B** $\rightarrow C(C) = 5$

2.2.4 Processo D

1. **inst D1** $\rightarrow C(D) = 1$
2. **recv A** $\rightarrow C(D) = 6$
3. **send A** $\rightarrow C(D) = 7$
4. **inst D2** $\rightarrow C(D) = 8$

5. **send B** $\rightarrow C(D) = 9$

3 Consolidação dos Timestamps

Com base nos cálculos realizados, os timestamps finais para os eventos de cada processo são:

- **Processo A:** (1, 2, 3, 4, 5, 8, 9)
- **Processo B:** (1, 2, 6, 7, 10)
- **Processo C:** (1, 2, 3, 4, 5)
- **Processo D:** (1, 6, 7, 8, 9)

4 Conclusão

Os eventos foram ordenados utilizando relógios lógicos e a relação de causalidade, assegurando a consistência na distribuição dos eventos no sistema distribuído. Os timestamps são consistentes com a relação "happened-before", garantindo que os eventos sejam corretamente ordenados dentro e entre os processos