Laboratório Tempo e estado global

Nome: Matheus Henrique

Curso: RC

Matricula: 475398

Parte 1 – Relógios Físicos

Na primeira parte, cada processo usou somente o relógio físico disponibilizado pelo sistema operacional, utilizando o método (System.currentTimeMillis()). O timestamp físico correspondente registrou cada evento de envio e recebimento de mensagens.

```
PS C:\Users\Matheus\Desktop\LaboratorioRelogios\parte1_fisico> java .\Main.java
[P2] RECEBIMENTO de P1 | Tempo físico recebido: 1768236598319
[P1] ENVIO | Tempo físico: 1768236598319
[P0] ENVIO | Tempo físico: 1768236598743
[P1] RECEBIMENTO de P0 | Tempo físico recebido: 1768236598743
[P0] RECEBIMENTO de P2 | Tempo físico recebido: 1768236599308
[P2] ENVIO | Tempo físico: 1768236599308
```

Durante a execução, foram observadas inconsistências na ordem dos eventos. Em alguns casos, o log apresentou eventos de recebimento ocorrendo antes dos respectivos envios, Esse comportamento viola a relação causal, pois um evento de recebimento não pode ocorrer antes do envio da mensagem correspondente.

Parte 2 – Relógio Lógico de Lamport

Na segunda parte, foi implementado o algoritmo de **relógio lógico de Lamport**. Cada processo manteve um contador lógico inicializado em zero, atualizado conforme as seguintes regras:

- Incremento a cada evento local;
- Incremento antes do envio de mensagens;
- Atualização no recebimento:
  L(p) = max(L(recebido), L(p)) + 1

```
PS C:\Users\Matheus\Desktop\LaboratorioRelogios\parte2_lamport> java .\Main.java
[P0] EVENTO LOCAL | Lamport: 1
[P1] EVENTO LOCAL | Lamport: 1
[P0] ENVIO | Lamport: 2
[P1] RECEBIMENTO de P0 | Novo Lamport: 3
[P2] EVENTO LOCAL | Lamport: 1
[P2] ENVIO | Lamport: 2
[P0] RECEBIMENTO de P2 | Novo Lamport: 3
[P1] ENVIO | Lamport: 4
[P2] RECEBIMENTO de P1 | Novo Lamport: 5
```

Durante a execução, notou-se que os valores do relógio lógico mantiveram constantemente a relação causal entre os eventos. Sempre que um evento de envio ocorreu antes de um evento de recebimento, o valor do relógio lógico do recebimento foi superior ao do envio.

Parte 3 – Relógios Vetoriais

Na Parte 3, foram implementados relógios vetoriais com o objetivo de identificar tanto a relação de causalidade quanto a concorrência entre eventos. Cada processo manteve um vetor lógico que representa o conhecimento sobre os eventos dos demais processos.

Cada processo tem um vetor V[i] de tamanho N (número de processos)

- Evento local: V[i][i]++
- Envio de mensagem: envia cópia de V[i]
- Recepção: V[i][j] = max(V[i][j], Vmsg[j]) para todo j, depois V[i][i]++

```
PS C:\Users\Matheus\Desktop\LaboratorioRelogios\parte3_vetorial> java .\Main.java
[P0] EVENTO LOCAL | Vetor: [1, 0, 0]
[P1] EVENTO LOCAL | Vetor: [0, 1, 0]
[P1] ENVIO | Vetor: [0, 2, 0]
[P2] RECEBIMENTO de P1 | Vetor: [0, 2, 1]
[P0] ENVIO | Vetor: [2, 0, 0]
[P1] RECEBIMENTO de P0 | Vetor: [2, 3, 0]
[P2] EVENTO LOCAL | Vetor: [0, 2, 2]
[P2] ENVIO | Vetor: [0, 2, 3]
[P0] RECEBIMENTO de P2 | Vetor: [3, 2, 3]
```

Por meio da comparação entre vetores, foi possível determinar se um evento ocorreu antes de outro ou se ambos são concorrentes, algo que não é possível apenas com o relógio de Lamport.