Avaliação de Checagem — MPI em Laboratório

Aplicações de clusterização em servidores

Tema: Comunicação entre processos com MPI (Scatter, Gather, rank, size)
Aluno: *[Seu nome aqui]*
Data: [Data aqui]

Objetivo

Executar, analisar e modificar um programa MPI que distribui dados entre processos, realiza cálculos locais e coleta os resultados para ordenação.

Arquivo base

O código que você deve utilizar está nesse diretorio:

```
media_mpi.c
```

Compile com:

```
mpicc media_mpi.c -o media_mpi
```

Execute com:

```
mpirun -np 4 ./media_mpi
```

Parte 1 — Execução básica

1.1. Saída esperada

Execute o programa com 4 processos. Copie aqui a saída do terminal:

```
[COLE AQUI SUA SAÍDA]
```

Parte 2 — Análise de funcionamento

2.1. O que faz `MPI_Scatter` neste código?

Resposta:

2.2. Qual o papel de `MPI_Gather`?

Resposta:

2.3. Por que a ordenação das médias acontece apenas no processo 0?

Resposta:

Parte 3 — Modificação

3.1. Modifique o código para que cada processo envie também seu maior valor local, além da média.

Use `MPI_Gather` para coletar ambos os dados no processo 0.

- Faça um **commit com sua modificação** e anexe abaixo o arquivo completo.

3.2. Copie aqui a saída do seu programa modificado:

```
[SAÍDA DO PROGRAMA MODIFICADO]
```

Análise com utilitários Linux

4.1. Use o comando `time` para medir o tempo de execução do programa com 2, 4 e 6 processos.

Anote abaixo:

Processos	Tempo (real)
2	
4	
6	

4.2. Use `htop` ou `top` para observar o uso de CPU. O uso foi balanceado entre os processos?

Resposta:

4.3. Use `strace`, `taskset` ou `MPI_Wtime` para investigar comportamento adicional do programa. Comente algo que tenha achado interessante:

Resposta:

Observações

- Faça commits frequentes com mensagens claras.
- Crie um `commit` final com a tag `atividade-finalizada`.
- Envie o link do seu repositório *forkado* com a atividade completa para luis.professor@uniatenas.edu.br

---

Boa prática!