Atividade prática 4 - Desenvolvimento de aplicação paralela

Esdras Fernandes da Cruz Leonardo Felipe Pacuola

1 Introdução

O número π é uma constante fundamental em matemática e física. Uma maneira simples de aproximá-lo numericamente é o *método de Monte Carlo*, que faz uso de geração de pontos aleatórios para estimar áreas e proporções. Neste relatório, exploramos duas vertentes desse método: a versão serial e a versão paralela usando a biblioteca `mpi4py` (MPI para Python). Avaliamos o desempenho em quatro cenários: execução serial e paralela com 2, 4 e 6 processos ("ranks").

2 Metodologia

2.1 Algoritmo Monte Carlo

- Gera $N = 10^7$ pontos aleatórios uniformes (x, y) no quadrante unitário $[0, 1] \times [0, 1]$.
- Conta quantos satisfazem $x^2 + y^2 \leq 1$. Esses pontos são considerados "dentro" do quarto de círculo.
- A proporção dentro$N$ aproxima $\pi/4$. Logo:

$$\pi \approx 4 \times \frac{\text{dentro}}{N}.$$

2.2 Paralelização com MPI

- Cada processo (rank) executa N/size lançamentos independentes.
- Sincroniza todos com `comm.Barrier()` para medir o tempo coletivo.
- Agrega, no rank 0, as contagens locais via `comm.reduce(..., op=MPI.SUM)`.
- No rank 0, calcula-se π e registra-se o tempo total de execução.

2.3 Coleta de dados de tempo

Executaram-se 10 repetições para cada cenário:

- Serial (sem MPI)
- Paralelo com 2 ranks
- Paralelo com 4 ranks
- Paralelo com 6 ranks

Registraram-se os tempos de execução em segundos e calcularam-se média e mediana.

3 Resultados

Tabela 1: Tempos de execução e speedups obtidos.

Cenário	Média (s)	Mediana (s)	Speedup teórico ×	Speedup obtido ×
Serial	4,526	4,530	1,00 ×	1,00 ×
2 ranks	2,289	2,280	2,00 ×	1,98 ×
4 ranks	1,132	1,130	4,00 ×	4,00 ×
6 ranks	1,206	1,200	6,00 ×	3,75 ×

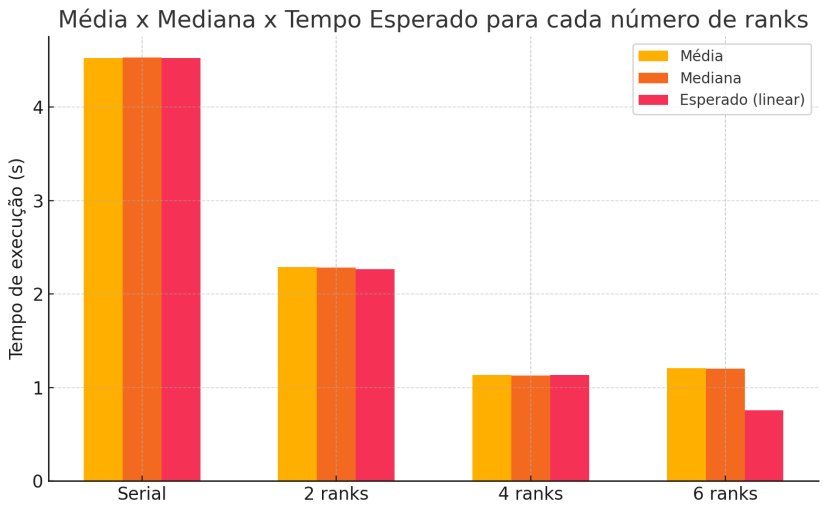

Figura 1: Comparação da média, mediana e tempo esperado para 1, 2, 4 e 6 ranks.

4 Discussão

- **Escalabilidade até 4 ranks**: O ganho de desempenho de serial até 2 e 4 ranks é quase linear:
  - 2 ranks: tempo médio $\approx 2.29\,\mathrm{s}$ ($2\times$ mais rápido).
  - 4 ranks: tempo médio $\approx 1.13\,\mathrm{s}$ ($4\times$ mais rápido).
- **Sobrecarga com 6 ranks**: O tempo médio sobe para $\approx 1.21\,\mathrm{s}$, gerando speedup de $3{,}75\times$ em vez dos $6\times$ teóricos. Principais causas:
  - Overhead de sincronização (`Barrier`).
  - Custo de comunicação no `reduce`.
  - Variações de carga entre ranks.

5 Conclusão

O método de Monte Carlo para estimar π apresenta excelente paralelismo até 4 processos, atingindo quase speedup ideal. Acima disso, o overhead de MPI passa a dominar e reduz o retorno do paralelismo.