**Trabalho de avaliação na disciplina de Processamento Paralelo e Distribuído**

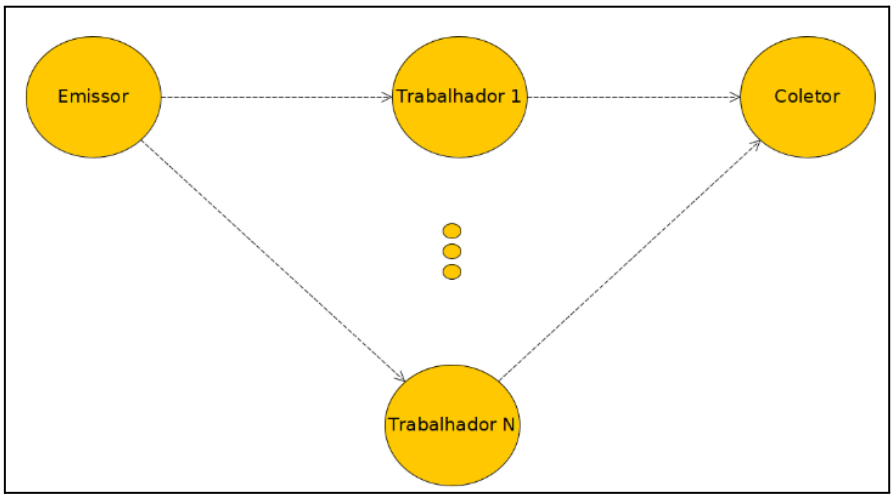
Aluno: Vítor Mateus Backes Barth
Email: vitorbackesbarth@gmail.com
Repositório do código: https://github.com/vitorbarth/parallel-programming-mpi

**Implementação do padrão FARM usando OpenMPI**

Com o intuito de calcular a quantidade de números primos presentes no intervalo de um dado número e apresentar resultados de performance obtidos com o paralelismo da operação, este projeto apresenta uma implementação do padrão FARM utilizando o OpenMPI em C++.

O padrão FARM (Farming out/Functional Algorithm with Reduction Mapping) é um padrão de programação paralela frequentemente usado em computação distribuída para realizar tarefas intensivas em computação de forma eficiente. Este padrão envolve a divisão de uma tarefa em várias partes menores, que são então distribuídas para serem processadas em paralelo por diferentes processadores ou nós de computação.

Neste trabalho o padrão é organizado com 1 Emissor, 1 ou mais Trabalhadores e 1 Coletor.




O Emissor recebe a solicitação de cálculo e o repassa aos Trabalhadores que por sua vez realizam cada um uma fatia da tarefa, a executar e envia o resultado ao Coletor, que por sua vez, unifica e apresenta o resultado final.

Exemplo: Ao calcular a quantidade de números primos até 1.000, tendo 2 Trabalhadores, o primeiro se encarregará do intervalo 1 - 500 e segundo do intervalo 501 - 1.000.

**Ambiente de Execução**

O ambiente de execução foi montado com 4 VMs Ubuntu Server 23.10, virtualizadas com Oracle VM VirtualBox, cada VM contou com 2GB de memória e 2CPUs (com restrição de execução de 50%).

Possibilitado rodar a aplicação com até 8 nodos (1 Emissor, 6 Trabalhadores e 1 Coletor).

O processador utilizado foi um Intel Core i5-1135g7 @ 2.40GHz 1.38 GHz.

**Resultados**

Foram realizados testes com três níveis de carga, calculando a quantidade de números primos até 500.000, 100.000 e 50.000, iniciando com o modo sequencial (apenas 1 Trabalhador) e paralisando com até 6 Trabalhadores. Cada uma dessas configurações foi executada 3 vezes.

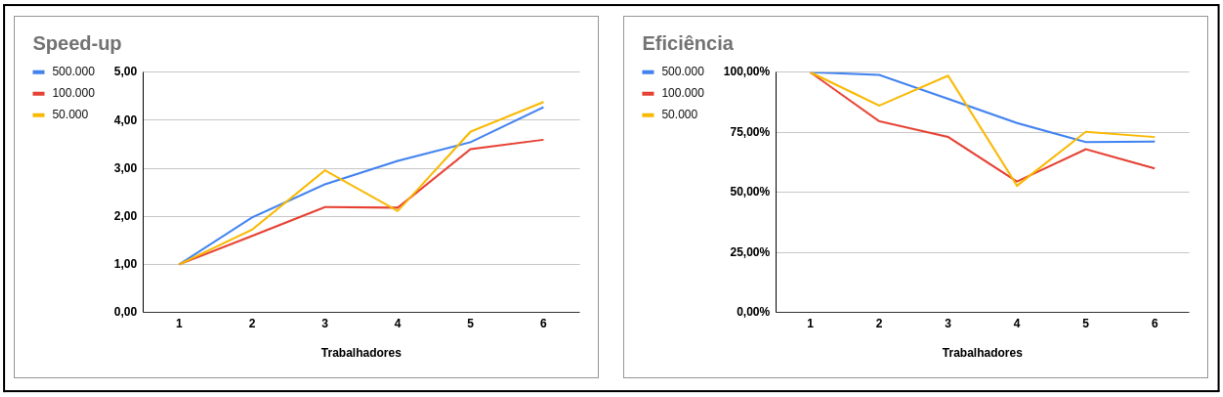
Os tempos apresentados a seguir estão em milissegundos.

| 500000 | | | | | | |
|---|---|---|---|---|---|---|
| Trabalhadores | 1 | 2 | 3 | 4 | 5 | 6 |
| Tempos | 32420 | 16121 | 12284 | 10088 | 9224 | 7560 |
| | 32759 | 17114 | 12270 | 10384 | 9467 | 7675 |
| | 32840 | 16568 | 13570 | 10667 | 9245 | 8034 |
| Tempo Médio | 32759 | 16568 | 12284 | 10384 | 9245 | 7675 |

| 100000 | | | | | | |
|---|---|---|---|---|---|---|
| Trabalhadores | 1 | 2 | 3 | 4 | 5 | 6 |
| Tempos | 1249 | 738 | 541 | 499 | 349 | 343 |
| | 1122 | 752 | 529 | 544 | 372 | 329 |
| | 1186 | 745 | 574 | 554 | 328 | 330 |
| Tempo Médio | 1186 | 745 | 541 | 544 | 349 | 330 |

| 50000 | | | | | | |
|---|---|---|---|---|---|---|
| Trabalhadores | 1 | 2 | 3 | 4 | 5 | 6 |
| Tempos | 305 | 194 | 117 | 142 | 101 | 79 |
| | 346 | 201 | 170 | 164 | 91 | 75 |
| | 410 | 243 | 106 | 168 | 92 | 91 |
| Tempo Médio | 346 | 201 | 117 | 164 | 92 | 79 |

A média desses tempos foi utilizada para calcular o Speedup obtido com a paralisação, o qual se dá pela divisão do tempo da execução sequencial pelo tempo da execução paralela. Possibilitando assim a visualização da real eficiência entregue pela paralelização deste processo.




| 500000 | | | |
|---|---|---|---|
| Trabalhadores | Tempo Médio | Eficiência | Speedup |
| 1 | 32759 | 100,00% | 1,00 |
| 2 | 16568 | 98,86% | 1,98 |
| 3 | 12284 | 88,89% | 2,67 |
| 4 | 10384 | 78,87% | 3,15 |
| 5 | 9245 | 70,87% | 3,54 |
| 6 | 7675 | 71,14% | 4,27 |

| 100000 | | | |
|---|---|---|---|
| Trabalhadores | Tempo Médio | Eficiência | Speedup |
| 1 | 1186 | 100,00% | 1,00 |
| 2 | 745 | 79,60% | 1,59 |
| 3 | 541 | 73,07% | 2,19 |
| 4 | 544 | 54,50% | 2,18 |
| 5 | 349 | 67,97% | 3,40 |
| 6 | 330 | 59,90% | 3,59 |

| 50000 | | | |
|---|---|---|---|
| Trabalhadores | Tempo Médio | Eficiência | Speedup |
| 1 | 346 | 100,00% | 1,00 |
| 2 | 201 | 86,07% | 1,72 |
| 3 | 117 | 98,58% | 2,96 |
| 4 | 164 | 52,74% | 2,11 |
| 5 | 92 | 75,22% | 3,76 |
| 6 | 79 | 73,00% | 4,38 |

Dado a natureza ambiente de testes utilizado, com a incapacidade de isolar o processamento desta atividade, tendo concorrência com o SO da máquina física, não é possível afirmar que os resultados obtidos possam ser replicados com coerência.

Em ambos os testes realizados, constatou-se um bom desempenho, paralisando o processo com 2 e 3 Trabalhadores.

Embora apresente redução nos tempos de execução, a paralisação com 4 à 6 Trabalhadores apresentou reduções consideráveis de eficiência.