Universidade Estadual de Campinas – UNICAMP
Faculdade de Tecnologia – FT

PROJETO

MERGE AND SORT

SISTEMAS OPERACIONAIS

GRUPO BETERRABA

Prof. André Leon Sampaio Gradvohl

Antonio Carlos De Lima	194302
Paloma Eduarda Salvador de Mello	260610

SUMÁRIO

1. REPOSITÓRIO NO GITHUB

Segue abaixo link do repositório no GitHub que contém todos os arquivos relacionados ao projeto, como:

- Relatório em PDF com link para o download do vídeo
- O programa “Merge and Sort”
- Testes que serão realizados

Link GitHub: https://github.com/tonylimao/Beterraba

Link do Vídeo:

https://drive.google.com/file/d/16YX2KMC25UVCksVaoCsVQAUHOVJYlGGZ/view?usp=sharing

2. DESCRIÇÃO DO PROBLEMA

O projeto realizado pela equipe Beterraba, devido a soma dos últimos números do RA dos integrantes, e somando o resto da divisão por 2 com 1, o projeto selecionado foi o de índice 1. Segue abaixo a soma:

RA-1: 194302; RA-2: 260610.

2 + 0 = 2

2 / 2 = 1 resto 0

0 + 1 = 1

O projeto visa a criação de um programa que utilize múltiplas threads (2,4,8,16) para ordenar de forma crescente uma série aleatória de números inteiros lidos de N arquivos fornecidos pelo usuário e grava-los em um único arquivo de saída. Paralelamente, deve-se realizar uma análise do desempenho desse programa com 2,4,8 e 16 threads, utilizando um marcador de tempo como parâmetro de comparação.

3. SOLUÇÃO APRESENTADA

Para o desenvolvimento do programa, o grupo criou 5 funções void para solucionar o problema apresentado e depois uni-as na função principal. Para melhores resultados foram criadas variáveis globais e uma struct que serviu como base para passar os parâmetros para as threads.

O programa foi ordenado conforme a necessidade, sendo primeiro o recebimento e leitura dos valores dos arquivos, escrevendo-os em um arquivo de saída (sem ordenação), em seguida gravando os valores em um vetor e ordenando-os para então salvá-los novamente em um arquivo de saída, desta vez ordenado. No final é apresentado o tempo total de execução das threads.

4. INSTRUÇÕES PARA A COMPILAÇÃO

Ao realizar o download da Branch Master no repositório GitHub Beterraba, basta compilar o código via terminal da seguinte maneira:

gcc mergesort.c -o mergesort -lpthread

Após a compilação do programa, basta executá-lo da seguinte forma:

./mergesort numeroDeThreads aquivos.In arquivo.Out

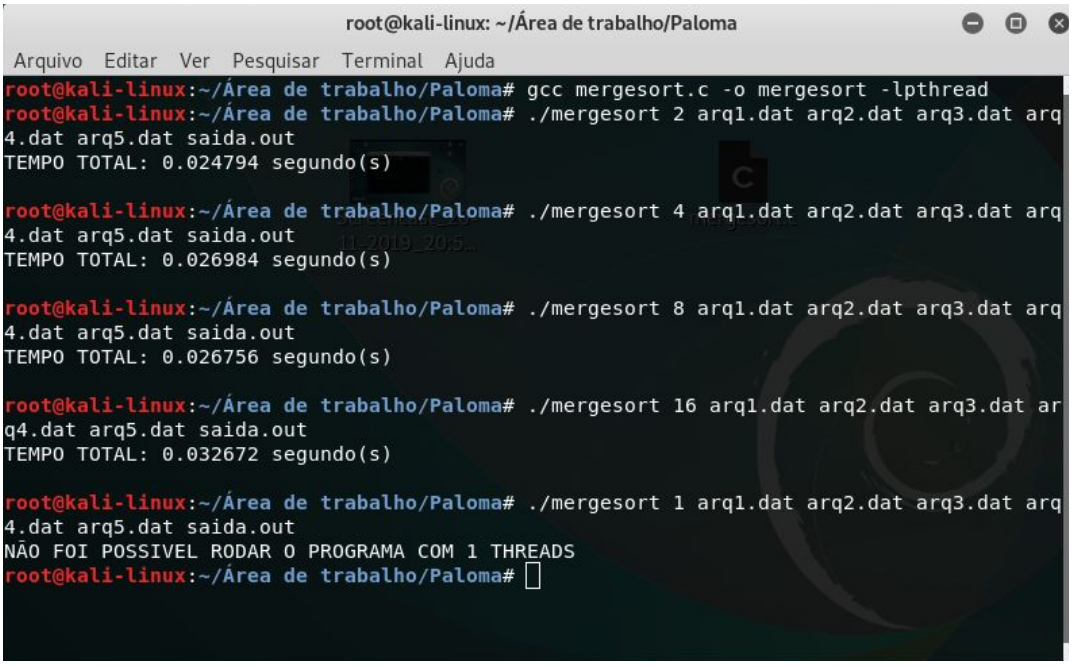
Exemplos:

./mergesort 2 arq1.in arq2.in arq3.in arq4 arq5 saida.out

./mergesort 4 arq1.in arq2.in arq3.in arq4.in arq5.in saida.out

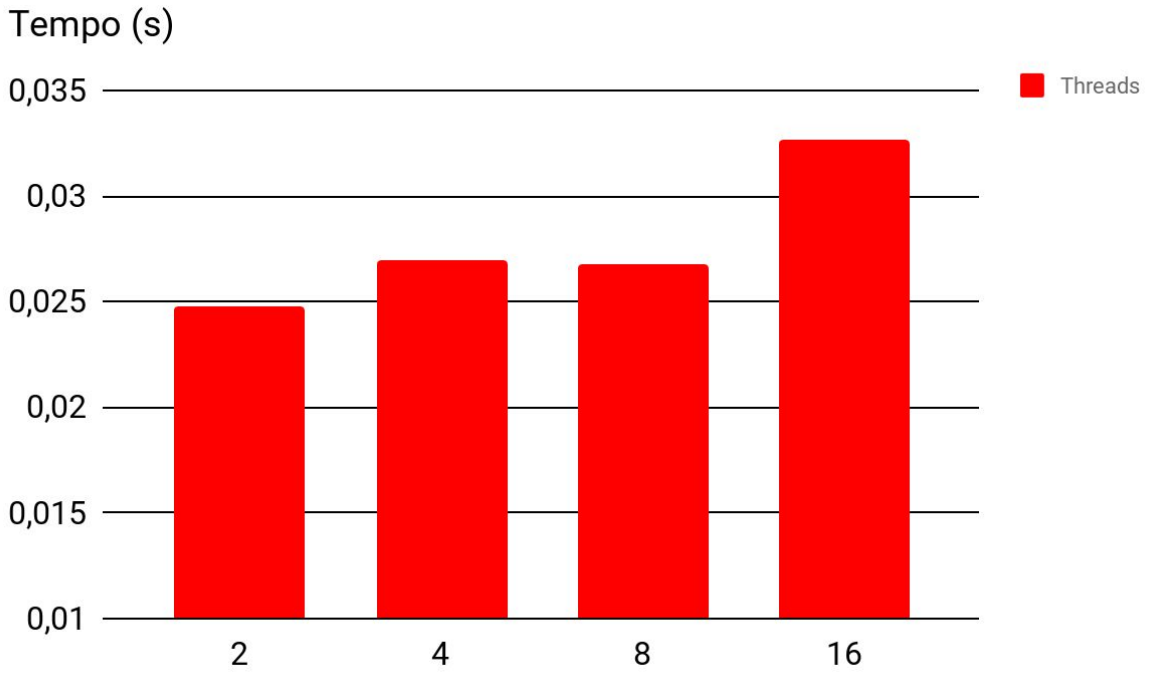
./mergesort 8 arq1.in arq2.in arq3.in arq4.in arq5.in saida.out

./mergesort 16 arq1.in arq2.in arq3.in arq4 arq5.in saida.out



5. GRÁFICO

Teste feito com as entradas *arq1.dat, arq2.dat, arq3.dat, arq4.dat E arq5.dat* com 2, 4, 8 e 16 threads:



Para os testes, foi utilizado um notebook Intel(R) Core(TM) i5-2450M CPU @ 2.50 GHz 4,00 GB RAM.

6. CONCLUSÃO

Com base nos testes feito, na execução do programa, ao se aumentar os números de threads observa-se uma diminuição na eficiência de processamento e consequentemente um aumento no tempo de execução das threads. Isso ocorre, pois um processador i5 possui apenas 4 threads, forçando o programa a esperar para que as demais sejam atendidas, perdendo assim sua eficiência na execução.