UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA

NATHAN D'AMICO CARDOSO – RA: 173537

VICTOR HUGO SILVA – RA: 253078

EXECUÇÃO DE LINHAS DE COMANDO DIVIDIDAS EM *THREADS*

LIMEIRA

2024

1. INTRODUÇÃO

O projeto prático de Sistemas Operacionais, disciplina TT304, ministrada pelo Professor Doutor André Leon S. Gradvohl propõe aos alunos o desenvolvimento de um trabalho onde haja a consolidação dos conhecimentos sobre programação *multithread.* O projeto consiste na criação de um programa que utilize múltiplas threads para ler valores inteiros de N arquivos de entrada e armazená-los em ordem crescente em um único arquivo de saída. Os dados dos arquivos de entrada não estão ordenados e podem estar repetidos.

O programa deverá ser escrito para o sistema operacional Linux e obrigatoriamente utilizar a Linguagem C (pura)1 e a biblioteca POSIX Threads. A definição do problema a ser resolvido, bem como as demais informações sobre o projeto estão nas seções a seguir.

Todas as informações dessa introdução são encontradas no arquivo de especificação do projeto prático da disciplina TT304 – Sistemas Operacionais

2. INSTRUÇÕES DE COMPILAÇÃO DO PROGRAMA

Antes da compilação do programa em seu sistema operacional, confira se já possui instalado as bibliotecas utilizadas na compilação e execução do programa, sendo elas: "*stdio.h"; "stdlib.h"; "pthread.h"; "string.h"; "time.h"; "locale.h".*

A compilação do programa e sua execução são feitas diretamente no terminal, onde com a utilização do *GNU Compiler Collection* (GCC) o programa é compilado. A linha de entrada do terminal para compilação do programa no nosso caso é:

gcc ./mergesort.c -o mergesort

Esse comando define o GCC como compilador para o programa, compila o código do arquivo "mergesort.c", e o parâmetro "-o" define o arquivo de saída da compilação e por fim o arquivo "mergesort" é definido como o executável do resultado da compilação.

Caso não haja erros de compilação o terminal não retornará nenhuma mensagem, e logo em seguida deve-se inserir a linha de execução do programa no terminal, definida no arquivo de especificação do projeto por:

.\mergesort N arq1.dat arq2.dat arq3.dat arq4.dat arq5.dat -o saida.dat

Essa entrada define "mergesort" como programa a ser executado, "N" é referente ao número de *threads* que serão utilizados na execução, os arquivos ".dat" são arquivos de entrada de texto a serem organizados pelos *threads*, "-o" define o arquivo de saída da compilação e "saida.dat" é o arquivo de saída da execução onde serão armazenados os números ordenados em ordem crescente pelos *threads.*

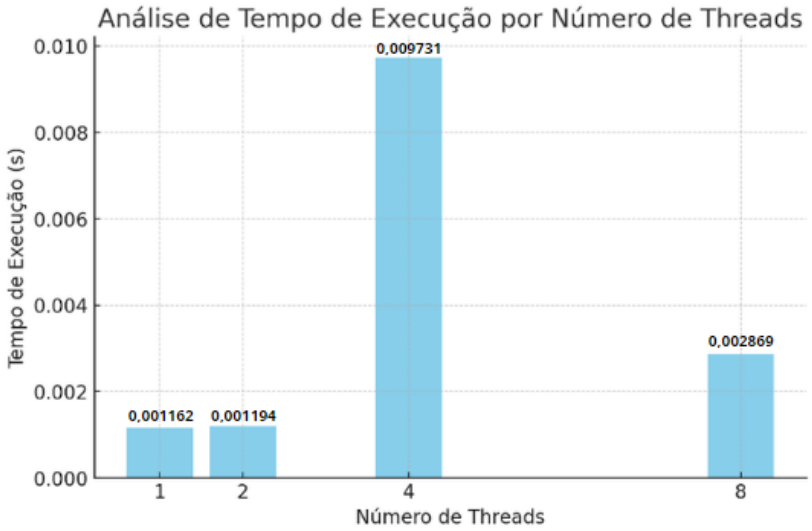
3. GRÁFICOS COMPARATIVOS DOS TEMPOS DE EXECUÇÃO

Os experimentos de execução do projeto exigem que haja a execução-teste utilizando cinco arquivos de texto com mil números inteiros aleatórios cada, e que estes arquivos sejam organizados por um, dois, quatro e oito *threads* respectivamente em execuções distintas. O intuito é comparar o tempo de execução do programa em situações que envolvam diferentes números de *threads.*

Imagem 1 - Execução do programa em utilização de “N” Threads

```
vhsilva@ubuntu:~/Documents/Threads-master$ ./mergesort 1 arq1.dat arq2.dat arq3.dat arq4.dat arq5.dat -o saida.dat
Tempo de execução do Thread 1: 0.001162 segundos.
Tempo total de execução das threads: 0.001162 segundos
vhsilva@ubuntu:~/Documents/Threads-master$ ./mergesort 2 arq1.dat arq2.dat arq3.dat arq4.dat arq5.dat -
o saida.dat
Tempo de execução do Thread 1: 0.000570 segundos.
Tempo de execução do Thread 2: 0.000624 segundos.
Tempo total de execução das threads: 0.001194 segundos
vhsilva@ubuntu:~/Documents/Threads-master$ ./mergesort 4 arq1.dat arq2.dat arq3.dat arq4.dat arq5.dat -
o saida.dat
Tempo de execução do Thread 1: 0.001980 segundos.
Tempo de execução do Thread 2: 0.002596 segundos.
Tempo de execução do Thread 3: 0.001830 segundos.
Tempo de execução do Thread 4: 0.003325 segundos.
Tempo total de execução das threads: 0.009731 segundos
vhsilva@ubuntu:~/Documents/Threads-master$ ./mergesort 8 arq1.dat arq2.dat arq3.dat arq4.dat arq5.dat -
o saida.dat
Tempo de execução do Thread 1: 0.000175 segundos.
Tempo de execução do Thread 2: 0.000188 segundos.
Tempo de execução do Thread 3: 0.000328 segundos.
Tempo de execução do Thread 4: 0.000109 segundos.
Tempo de execução do Thread 5: 0.000125 segundos.
Tempo de execução do Thread 6: 0.000455 segundos.
Tempo de execução do Thread 7: 0.000121 segundos.
Tempo de execução do Thread 8: 0.001368 segundos.
Tempo total de execução das threads: 0.002869 segundos
vhsilva@ubuntu:~/Documents/Threads-master$
```

Imagem 2 - Gráfico de análise do tempo de execução por “N” Threads



4. CONCLUSÃO

Observa-se a partir da análise dos experimentos que o tempo de execução total do programa varia de acordo com número de *threads* utilizado em sua execução, mostrando que a utilização de um sistema *multithread* possui vantagens e desvantagens, onde existem situações em que apresenta a vantagem do compartilhamento de recursos para otimização da execução das tarefas exigidas pelo programa, mas que em determinadas circunstâncias seu tempo de processamento pode ser comprometido dependendo do hardware que estiver hospedando as execuções.

Site onde o vídeo explicativo está publicado:

https://www.youtube.com/watch?v=n1r_mR0yCSM

Site do repositório do GitHub onde o código fonte está localizado:

https://github.com/NathanDAmico/Threads

https://github.com/victor84333/Trabalho-de-Threads