Trabalho Prático
Mergesort - Ordenação com Múltiplas Threads

Disciplina: TT304 - Sistemas Operacionais
Prof. Dr. André Leon S. Gradvohl

Alunos:

Hitallo Alves Teles Azevedo - 196454
João Pedro Leite Calsavara - 197837

Grupo:

Two threads

Sumário

Descrição do Problema

O objetivo do projeto foi desenvolver um programa em linguagem C para realizar a leitura, ordenação e gravação no arquivo de saída, a partir de arquivos de entrada utilizando múltiplas threads. Cada arquivo de entrada possui uma sequência de inteiros não ordenados, um em cada linha. O programa deveria ler simultaneamente os arquivos, realizar a ordenação dos inteiros e, por fim, gravar o resultado em um único arquivo de saída. A proposta visava avaliar o impacto do aumento no número de threads no desempenho do processamento.

Para validar o programa, foram utilizados cinco arquivos, cada um com 1000 inteiros. Os testes foram realizados em quatro configurações de threads: 1, 2, 4 e 8 threads, e cada caso de teste foi executado cinco vezes para obter a média dos tempos de execução.

Instruções para compilar o programa

O projeto utiliza um Makefile para facilitar o processo de compilação. Para compilar o programa, utilize o comando:

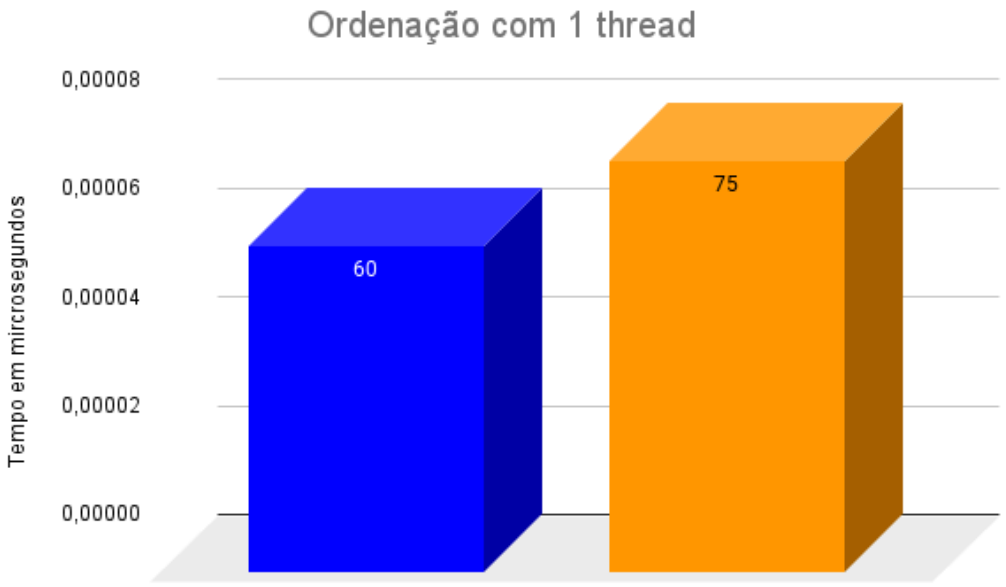
```
Unset
make
```

Certifique-se de estar no diretório raiz do programa.

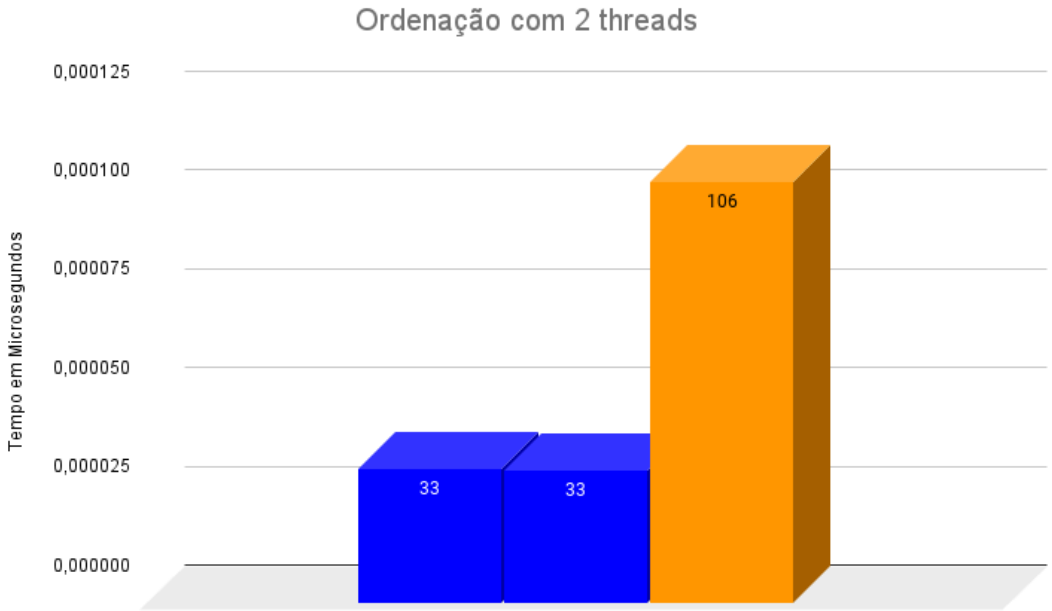
Nota: Para o pleno funcionamento do programa, é necessário que os arquivos de entrada estejam localizados no diretório do projeto -> /inputs.

Gráficos com os tempos de execução dos experimentos

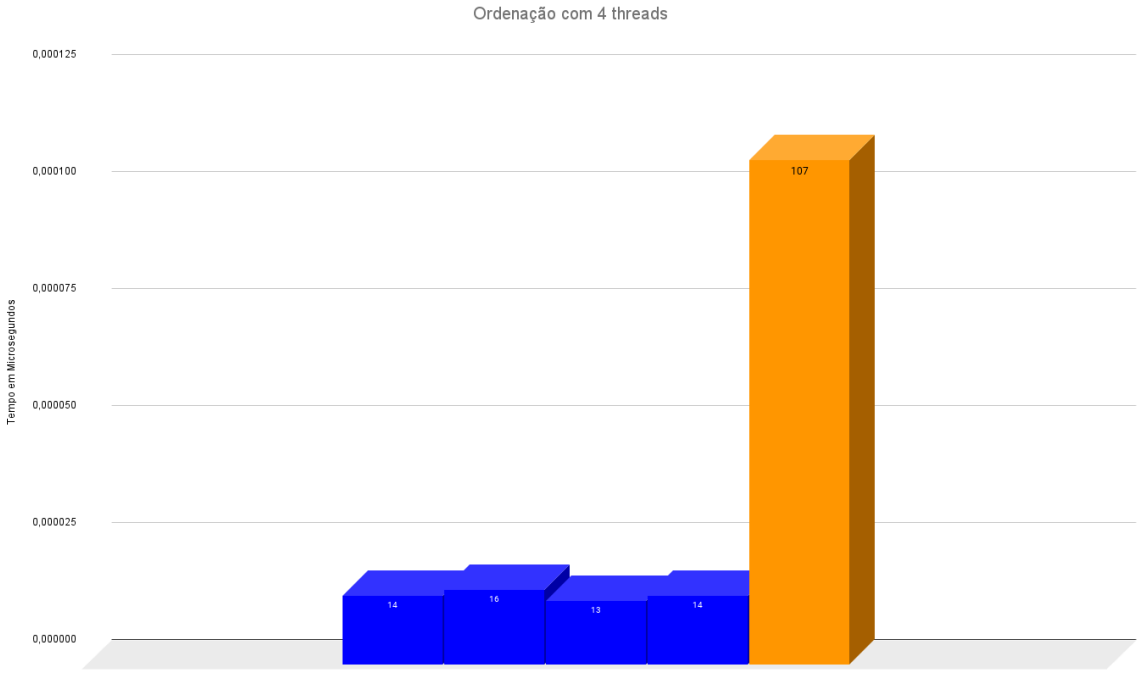
Tempo de execução total
Tempo de execução por thread



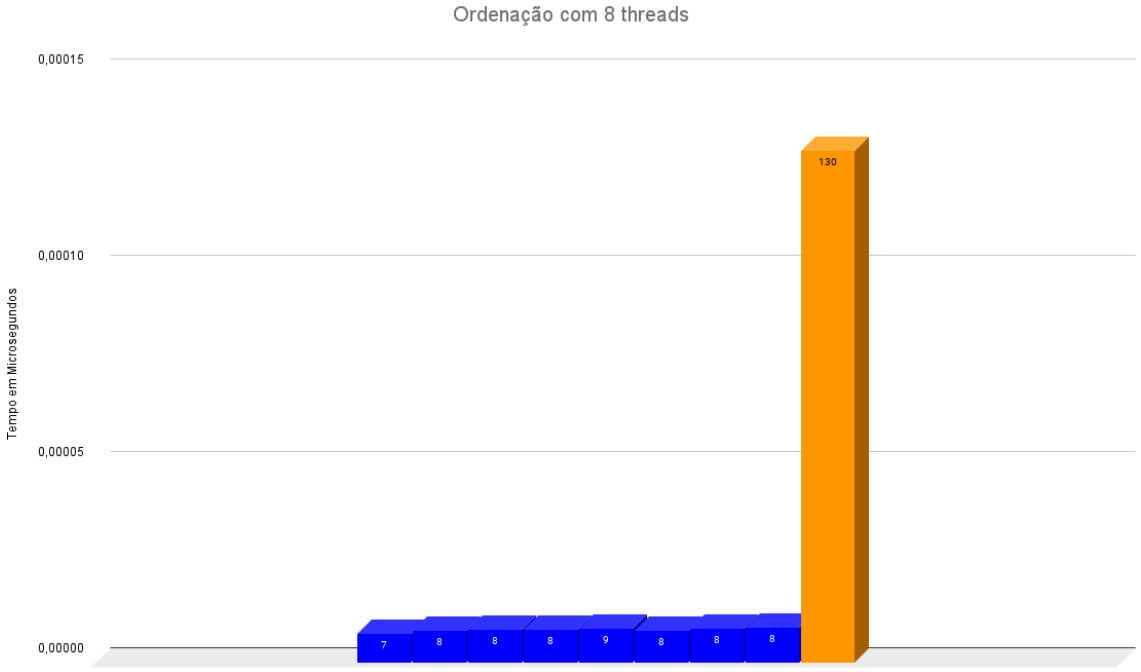
Tempos de execução em microsegundos: 60, 75.



Tempos de execução em microsegundos: 33, 33, 106.



Tempos de execução em microsegundos: 14, 16, 13, 14, 107.



Tempos de execução em microsegundos: 7, 8, 8, 8, 9, 8, 8, 8, 130.

Resultados e Conclusões

Os testes resultaram nos seguintes tempos médios para cada configuração de threads:

- **1 Thread:** 60 microssegundos para processamento na única thread, com um tempo total de 75 microssegundos.
- **2 Threads:** 33 microssegundos para cada thread, resultando em um tempo total de 106 microssegundos.
- **4 Threads:** 14, 16, 13 e 14 microssegundos para cada thread, com tempo total de 107 microssegundos.
- **8 Threads:** seis threads com 8 microssegundos, uma com 9 microssegundos e uma com 7 microssegundos, totalizando 130 microssegundos.

Esses resultados indicam que, embora o uso de múltiplas threads tenha permitido a execução paralela de tarefas, o tempo total não diminuiu proporcionalmente com o aumento no número de threads. A configuração com 1 thread apresentou o menor tempo total, enquanto o uso de 8 threads resultou em um tempo total mais elevado.

Esses dados mostram que a sobrecarga de gerenciamento de múltiplas threads aumenta com o número de threads, o que, em alguns casos, pode anular os ganhos de paralelismo. Isso sugere que, para tarefas de I/O e processamento leve, um número menor de threads pode oferecer um desempenho mais eficiente.

Informações Gerais

Link do vídeo: https://youtu.be/-b-9yzjy3Eg

Repositório do Github: https://github.com/hitalloazevedo/mergesort