Especificação do Projeto Prático da disciplina TT304 – Sistemas Operacionais

Prof. André Leon S. Gradvohl, Dr.
gradvohl@ft.unicamp.br

Introdução

O projeto prático da disciplina TT304 – Sistemas Operacionais é um dos seus componentes de avaliação. O principal objetivo desse projeto é a consolidação dos conhecimentos sobre programação com múltiplos *threads*. Para tanto, 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.

Definição do problema a ser resolvido

Considere que existem N arquivos de entrada com diferentes quantidades de valores inteiros não ordenados e que podem estar repetidos ou não. O programa deverá ler os valores inteiros desses vários arquivos e, de forma ordenada, armazená-los em um único arquivo de saída.

O programa deve funcionar para T *threads*, onde T pode ser 2, 4 ou 8 *threads*. A quantidade de *threads* que o programa utilizará nunca deverá ultrapassar o valor de T. O programa deve usar a máxima quantidade de *threads* possível.

A implementação da solução poderá utilizar o algoritmo de ordenação que achar mais conveniente. No entanto, caso os autores utilizem algoritmos que já existem, devem citar a fonte desses algoritmos.

Entradas e saídas do programa

O programa deverá receber todos os dados necessários para seu processamento no início da sua execução. Em outras palavras, nenhuma solicitação de dados deve ser feita ao usuário durante a execução do programa e todos os dados devem ser informados na linha de comando para instanciar o programa.

Detalhes sobre a passagem de parâmetros pela linha de comando para programas na linguagem C podem ser encontrados no *site* https://github.com/gradvohl/ArgsLinhaComando.

Entradas: O número de *threads* que o programa deve utilizar, os nomes dos arquivos de entrada e o nome do arquivo de saída. Para os testes, considere pelo menos 5 arquivos com 1000 valores cada.

Construa o programa de modo que os valores de entrada sejam lidos da linha de comando. Por exemplo:

```
./mergesort 4 arq1.dat arq2.dat arq3.dat –o saida.dat
```

Onde:

- `./mergesort` é o nome do programa;
- `4` é o número de threads;
- `arq1.dat arq2.dat arq3.dat` são os arquivos com dados de entrada; e

[1] A linguagem C pura não deve ser confundida com a linguagem C++. Portanto, caso qualquer comando ou função da linguagem C++ seja usado, isso acarretará nota zero neste componente de avaliação.

- `-o saida.dat` indica o arquivo que contém os dados de saída.

Importante: para facilitar a correção, garanta que o nome do seu programa seja `mergesort`. Considere que todos os arquivos de entrada contêm números inteiros (em caracteres), sendo um inteiro em cada linha.

Saídas: Além do arquivo ordenado, o programa deverá informar na tela os tempos de execução de cada thread de processamento e o tempo total de execução de todos os threads. Por exemplo

```
Tempo de execução do Thread 0: 0.092 segundos.
Tempo de execução do Thread 1: 0.067 segundos.
Tempo de execução do Thread 2: 0.073 segundos.
Tempo de execução do Thread 3: 0.058 segundos.
Tempo total de execução: 5.922 segundos.
```

Importante: o tempo de execução de cada *thread* é contado a partir do início daquele *thread* até o final dele (dentro do *thread*). O tempo total de execução de todos os *threads* de processamento considera o tempo de criação do conjunto dos threads de processamento até a finalização de todo o conjunto.

Relatório do projeto

Além da implementação do programa, o projeto precisará produzir um relatório dos experimentos com os tempos de processamento do programa para as seguintes situações:

- Número de *threads* de processamento (T): T = 1, T = 2, T = 4 e T = 8 *threads*.
- Arquivos com 1000 inteiros.

São, portanto, quatro situações a serem analisadas, gerando resultados para quatro experimentos. Importante lembrar que, o programa deve trabalhar com quaisquer tamanhos de arquivos (não limitado a 1000 inteiros).

Para cada situação analisada, deve ser calculado exclusivamente o tempo de total de processamento, isto é, o tempo gasto apenas nas operações com a ordenação dos valores. O tempo com as operações de leitura e gravação devem ser desconsiderados. Esses tempos de processamento devem ser comparados entre as diversas situações para obter as conclusões dos experimentos.

Para a medição do tempo de processamento, sugere-se a utilização da primitiva `clock_gettime`, conforme o exemplo a seguir.

```
#include <stdio.h>
#include <time.h>
void fun(){
    printf("A fun() iniciou. \n");
    printf("Pressione enter para parar a função. \n");
    while(1)
     if (getchar()) break;
    printf("A função parou \n");
}
int main(){
   struct timespec inicio, fim;
   double tempoTotal;
   clock_gettime(CLOCK_MONOTONIC, &inicio);
   fun();
   clock_gettime(CLOCK_MONOTONIC, &fim);
   tempoTotal = (fim.tv_sec - inicio.tv_sec); // segundos
   tempoTotal += (fim.tv_nsec - inicio.tv_nsec) / 10e9; // nanosegundos
   printf("A função fun() gastou %f segundos. \n", tempoTotal);
   return 0;
}
```

Requisitos importantes do programa desenvolvido

Há alguns requisitos fundamentais que devem ser respeitados no programa desenvolvido. Esses requisitos estão a seguir.

O primeiro requisito é que todos os vetores precisam ser alocados dinamicamente (com o comando `malloc` ou equivalente). Havendo qualquer tipo de alocação estática de tipos compostos (vetores) no programa, isso implicará em nota zero no respectivo laboratório.

O terceiro requisito é que o programa deve compilar sem erros. Para tanto, as instruções para a compilação devem estar muito bem definidas. Recomenda-se que seja criado um `makefile` ou um *script* para a compilação (consulte um breve tutorial disponível no Moodle da disciplina). Se o programa não compilar, ou aparecerem erros de compilação, isso implicará em nota zero no respectivo laboratório.

O quarto requisito é que os dados também devem estar armazenados em arquivos texto, conforme estabelecido na seção “Entradas e Saídas do Programa”. Se os arquivos de saída do programa não respeitarem esse requisito, isso implicará no desconto de 5 pontos da nota do projeto.

Por fim, o quinto requisito é utilizar exatamente a linha de comando indicada no exemplo informado na seção anterior (inclusive com o nome do programa). Se o programa não funcionar com a linha de comando indicada, isso implicará no desconto de 5 pontos da nota do projeto.

Os produtos do projeto e a entrega

Os produtos que devem ser entregues como resultado do projeto são os seguintes:

i. O código fonte do programa, na linguagem C, completo, documentado e pronto para ser compilado em sistemas Linux.
ii. Um relatório contendo a descrição do problema; as instruções para compilar o programa; os gráficos com os tempos de execução dos experimentos; e as conclusões a respeito dos resultados obtidos.
iii. Um vídeo mostrando o código fonte do programa, a compilação do programa, trechos dos arquivos de entrada e as execuções do programa para os experimentos.

Todo o código fonte documentado e o relatório devem estar disponíveis em um repositório Git. No Ambiente Virtual de Aprendizagem da Disciplina (Moodle), deve ser publicado apenas o relatório no formato *Portable Document Format* (PDF). Neste relatório deve constar o endereço do repositório Git e do *site* onde o vídeo está publicado.

Para o item (i), a sugestão é a utilização de um `makefile` que facilite a compilação em sistemas Linux. Consulte um breve tutorial disponível no Moodle da disciplina.

Para o item (ii), o relatório deve estar no formato PDF. Outros formatos não serão aceitos e, se forem enviados, terão nota zero automaticamente.

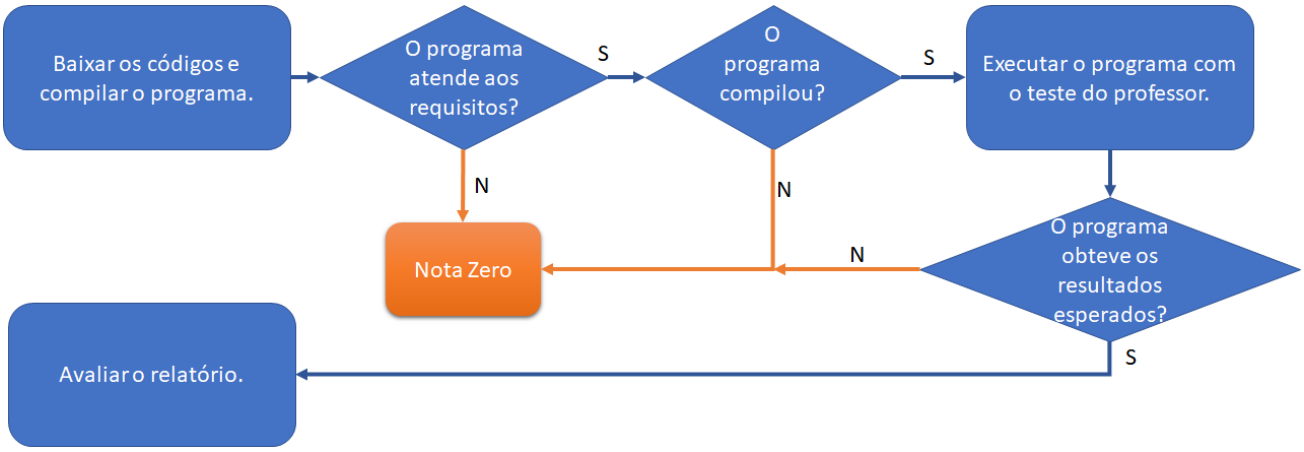
Para o item (iii), sugere-se o uso dos programas para capturar a tela do computador e criar vídeos. O vídeo deve estar publicado em um *site* a que o professor tenha acesso, independentemente de estar acessando a partir da rede da Unicamp ou não. A ausência do vídeo ou a impossibilidade de acesso acarretarão nota zero neste projeto. O vídeo poderá ser removido após a publicação das notas.

A falta de qualquer um dos itens acarretará nota zero nesse componente de avaliação.

Na data indicada no plano de ensino da disciplina, até as 23h55, o relatório deverá ser publicado no Moodle e todos os componentes do grupo devem concordar com o envio. Envios sinalizados como "rascunho" ou não confirmados por todos os membros do grupo serão considerados não enviados e terão nota zero.

Avaliação dos produtos do projeto

A avaliação do programa desenvolvido seguirá o seguinte fluxograma:



Os produtos do projeto, serão avaliados conforme os seguintes critérios.

- A qualidade dos códigos produzidos, inclusive o grau de modularidade dos códigos, e a inovação no que se refere ao uso dos núcleos de processamento disponíveis.
- A qualidade da documentação do código.
- A qualidade do texto do relatório, incluindo a correta utilização da norma gramatical da língua portuguesa.
- A estética e a qualidade do vídeo apresentado, bem como seu conteúdo e se cumpre com o que foi solicitado no início dessa seção.

A Tabela 2 a seguir informa alguns descontos na pontuação, caso determinadas situações ocorram.

Tabela 1 Situações em que podem ocorrer descontos na pontuação.

Situação	Desconto
Compilação com advertências (*warnings*)	-2 pontos
Código pouco ou mal documentado	-2 pontos
Código confuso ou pouco modularizado	-2 pontos
Vídeo incompleto, difícil de entender ou com baixa qualidade (de áudio ou de imagem)	-2 pontos
Relatório confuso ou com muitos erros de ortografia e gramática	-3 pontos
Código com alocação dinâmica em duas ou mais etapas	-5 pontos
Arquivos de saída fora do padrão	-5 pontos
Resultado diferente do esperado	-10 pontos
Programa desenvolvido em outra linguagem de programação que não seja C pura.	-10 pontos

A avaliação do projeto possui componentes objetivos (resultados esperados) e subjetivos (código confuso, documentação insuficiente, vídeo difícil de entender). Portanto, somente serão realizadas as revisões do projeto em relação à nota para aqueles projetos com nota menores do que 7,0 (sete). Notas iguais ou superiores à 7,0 (sete) não são passíveis de revisão.

Os autores terão direito a um *feedback* (esclarecimentos sobre a avaliação) sobre seu projeto se, e somente se, o projeto for entregue até uma semana antes da data de entrega prevista no Plano de Desenvolvimento da Disciplina. A razão para tanto é evitar a sobrecarga de solicitações durante as avaliações finais da disciplina. Essa solicitação deve ser feita por e-mail, em até 24 horas após a entrega do projeto. Solicitações até esse prazo serão indeferidas.

A data de entrega está estabelecida no Plano de Desenvolvimento da Disciplina.