Universidade do Vale do Itajaí
Disciplina: Sistemas Operacionais
Professor: Felipe Viel
Avaliação M1 – IPC, Threads e Paralelismo

Instruções

1. Esta avaliação deve ser feita individualmente e em até trio.

2. Data de entrega: 08/04/2024 até as 19:00. Trabalhos em atraso implicarão em 1,5 ponto de desconto por dia e, após 7 dias, não mais será aceita a entrega.

3. Esta avaliação tem por objetivo consolidar o aprendizado sobre conceitos de IPC, threads, concorrência e paralelismo.

4. A implementação deverá ser desenvolvida utilizando a linguagem de programação de sua preferência (C, C++, Python, Java, C#, Javascript/Node.js, Rust, etc). Porém, a utilização e suporte a threads pela linguagem escolhida é de responsabilidade do(s) aluno(s), seja usado corretamente o conceito de threads. As bibliotecas usadas devem ser equivalentes a Pthreads. Bibliotecas que também implementem e que permitam usar conceitos de paralelismo também podem ser usadas, mas o aluno também é responsável pelo seu uso e apresentação

5. O sistema deve ser entregue funcionando corretamente. Sistemas não compilando e executando não serão aceitos. É de responsabilidade do(s) aluno(s) apresentar a execução funcionando corretamente.

6. Deve ser disponibilizado os códigos da implementação em repositório do(s) aluno(s) (github, bitbucket, etc...), deve ser fornecido o link e o repositório dever ser público.

7. O relatório deve seguir o formato de artigo científico ou seguindo as normas da ABNT e as orientações para produção de trabalhos acadêmicos da Univali contendo:

    - Identificação do autor e do trabalho.
    - Enunciado dos projetos.
    - Explicação e contexto da aplicação para compreensão do problema tratado pela solução.
    - Resultados obtidos com as simulações.
    - Códigos importantes da implementação.
    - Resultados obtidos com a implementação (tabelas, gráficos e etc).
    - Análise e discussão sobre os resultados finais.

8. Deverá ser gravado um vídeo explicando a implementação
    a. O vídeo deve ser postado no material em plataformas como Youtube, Dropbox ou Google Drive. O link para o vídeo e para o código no repositório devem ser disponibilizados no relatório.
    b. Deverá ser explicado a solução, códigos, análises, resultados e conclusões obtidas.
    c. Deverá ser compilado e executado o código na apresentação.

9. Deve ser disponibilizado os códigos da implementação juntamente com o relatório (salvo o caso da disponibilidade em repositório aberto do aluno, que deve ser fornecido o link). O repositório deve estar aberto do momento da entrega em diante, sendo que o professor não responsabiliza caso o projeto não esteja disponível para consulta no momento da correção, sendo do(s) aluno(s) essa responsabilidade de manter disponível. Cópias entre alunos implicará em nota zero para todos os envolvidos.

10. O trabalho deverá ser apresentado em data definida pelo professor. É de responsabilidade do(s) aluno(s) explicar os conceitos, comandos, bibliotecas usadas. É de responsabilidade do(s) aluno(s) fazer a solução funcionar e ela deverá ser baixada do local de entrega no momento da apresentação. Trabalhos não apresentados terão como nota máxima 5,0, além dos descontos aplicados no restante da avaliação da implementação. Todos os alunos poderão apresentar ou o professor poderá escolher um representante para apresentar o trabalho em nome do grupo.

Descrição do projeto a ser desenvolvido

Projeto 1

Problemática: uma Indústria de Alimentos de Santa Catarina chamada FoodSec S.A. possui a tarefa de escanear alimentos por meio de câmeras e verificar se os mesmos estão corretos. Os alimentos podem passar por uma das três esteiras disponíveis. As três esteiras são controladas por um por um único computador centralizado. Esse computador recebe dados de um sensor em cada uma das esteiras que captura a contagem de itens que são identificados como seguros. A contagem é exibida em um display perto das esteiras (todos os displays são controlados pela mesma lógica, é apenas uma replicação).

Diante disso, a empresa precisa que vocês implementem, por meio de aplicação para distribuição Linux/Windows, uma solução que consiga realizar as contagens nas três esteiras e exiba o resultado total (contagem esteira 1 + contagem esteira 2 + contagem esteira 3). A empresa pede que seja simulado a solução em um sistema multicore com sistema operacional com suporte a threads e IPC. A empresa solicita que um processo seja responsável pela contagem usando threads e outro processo seja responsável pela apresentação no display. Você deve usar pipe para que os dois processos troquem dados.

Além disso, os pesos dos itens que passam por cada uma das esteiras são armazenados em um único vetor de dados. A cada 1.500 unidades de produtos, das três esteiras, é necessário atualizar o peso total de itens processados. Sendo assim, a empresa aceita uma pausa na quantidade de itens sendo contados e pesados para realizar a pesagem total.

A empresa também fornece uma análise das três esteiras:

- Esteira 1: produtos de maior peso (5 Kg) – passa 1 item a cada segundo pelo sensor.
- Esteira 2: produtos de peso médio (2 Kg) – passa 1 item a cada 0,5 segundo pelo sensor.
- Esteira 3: produtos de menor peso (0,5 Kg) – passa 1 item a cada 0,1 segundo pelo sensor.
- A contagem deve acontecer initerruptamente.
- A exibição no display deve atualizar a cada 2 segundos para os operadores poderem acompanhar.
- Um operador pode usar um botão no equipamento para poder parar a contagem devido a um problema ocorrido.

Diante da problemática apresentada, vocês terão que implementar uma aplicação (em nível de MVP) que possa lidar com tal situação usando duas abordagens: Pthreads e IPC entre processos com pipe. No processo que utiliza threads, lembre-se de utilizar mutex ou semáforo para controlar o acesso das seções críticas.

Pontuação Extra na M1

A utilização da biblioteca OpenMP para contagem do peso total usando o vetor de 1500 valores irá receber de 0,5 à 1,5 pontos na prova da M1. A nota extra fica a critério do professor em avaliar o contexto de aplicação, bem como a qualidade do código e aplicabilidade.