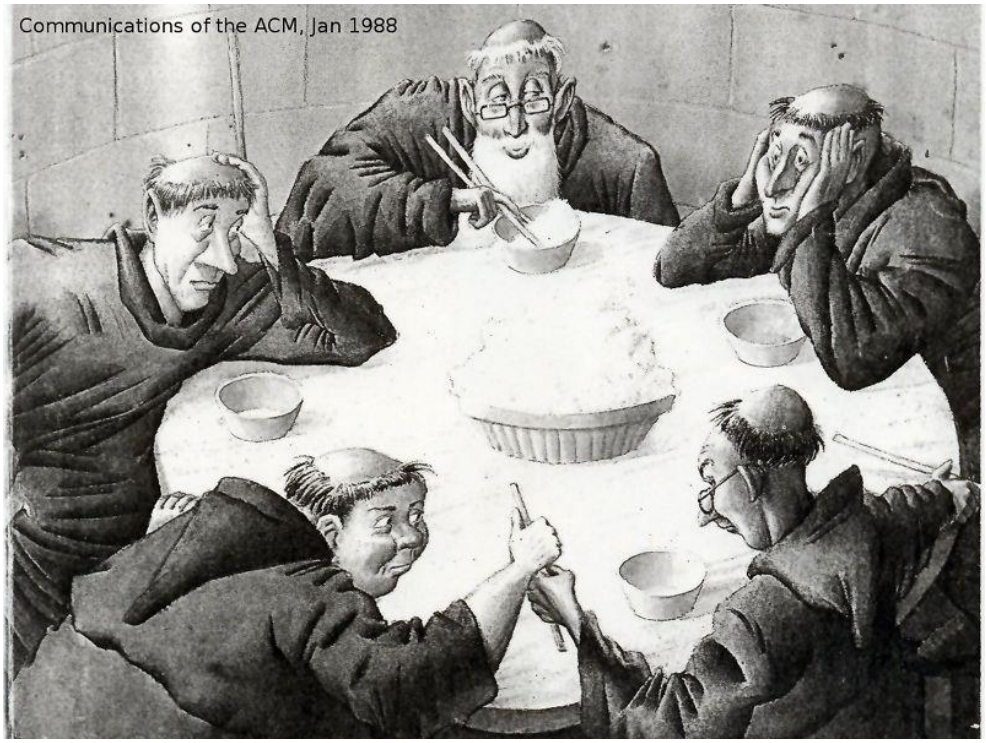
Trabalho de Concorrência T1: Jantar dos Filósofos

Objetivos:

O objetivo deste trabalho é aplicar os conceitos de concorrência por meio do estudo e comparação de soluções clássicas para o problema do Jantar dos Filósofos. Os principais pontos são: compreender os mecanismos de sincronização utilizados, identificar formas de evitar *deadlock* e *starvation*, e analisar o desempenho das soluções em sistemas *multicore*, comparando-as com uma versão sequencial.

Descrição do problema:



O problema do "Jantar dos Filósofos" foi originalmente formulado por Dijkstra em 1965 e é amplamente utilizado para exemplificar questões de concorrência e o acesso a recursos compartilhados. No cenário, cinco filósofos estão sentados ao redor de uma mesa, com um garfo entre cada par de filósofos adjacentes. Cada filósofo alterna entre pensar e comer. Para comer, um filósofo precisa dos dois garfos adjacentes (à esquerda e à direita). A solução para este problema deve garantir que não haja ***deadlock*** nem ***starvation***, ao mesmo tempo que mantém um bom nível de concorrência (permitindo que vários filósofos pensem ou comam simultaneamente).

Tarefa:

O grupo deve pesquisar e encontrar duas soluções concorrentes para o problema, escolhendo entre as soluções clássicas (duas dentre as seguintes: Dijkstra, Hierarquia de Recursos, Otimização de Tannenbaum, Arbitragem/Garçom, Chandy/Misra).

Os códigos podem ser obtidos na Internet e escritos em qualquer linguagem de programação, desde que sua execução seja reproduzida em uma máquina local (ou acessível ao grupo) com mais de um núcleo. Necessário incluir no relatório referências de onde o código foi obtido.

O grupo deve incluir uma versão sequencial do problema, onde os filósofos pensam e comem de forma sequencial, um de cada vez, para ser usada como base de comparação com as versões concorrentes.

Além da implementação, o grupo deve:

1. Descrever e explicar o funcionamento de cada solução.
2. Executar as soluções em uma máquina com múltiplos núcleos.
3. Analisar e comparar as soluções em relação aos seguintes aspectos:
    - Funcionamento geral da solução.
    - Mecanismos de sincronização usados.
    - Como esses mecanismos são implementados na linguagem escolhida.
    - Como as soluções evitam *deadlock* e *starvation* (discutir se houver limitações).
    - Nível de concorrência alcançado em cada solução (comparar com uma versão sequencial não concorrente). A avaliação do nível de concorrência pode ser feita através de estimativas de tempo, medindo o tempo total que os filósofos levam para pensar e comer 100 vezes cada. Outra opção é executar o programa durante um tempo determinado e verificar quantas vezes cada filosofo pensou e comeu no período.

Entrega e avaliação:

O trabalho deve ser realizado em grupos de até 4 integrantes. Os grupos precisam ser registrados no Moodle através da ferramenta de escolha de grupos.

A avaliação consiste no acompanhamento do desenvolvimento do trabalho durante as aulas de laboratório e no envio de um **relatório técnico** pelo Moodle, que deve contemplar as análises descritas acima. Apenas um membro do grupo deve fazer a submissão do relatório no Moodle. Atente para o prazo especificado na sala de entrega do Moodle.

Formato do relatório técnico:

- Arquivo em formato **.pdf**.
- Cabeçalho reduzido com identificação do grupo e do trabalho.
- Primeira página em **coluna dupla**, com margens de 1 cm e fonte tamanho 10.
- Segunda página com capturas de tela demonstrando a execução dos programas.
- A partir da terceira página, código-fonte formatado em **coluna simples** (sem limite de páginas).

Dicas para formatar o código:

- Copiar o código do VS Code e colar no Word, que mantém a formatação. Ajuste o tamanho da fonte e o espaçamento entre as linhas para economizar espaço.
- Outra opção é utilizar o Overleaf/Latex, que importa o código já formatado.