Problema de sincronização usando threads

José Luiz Maciel, Igor Antônio Pedroso, Gustavo Dias de Castro

[1]IFMG - Instituto Federal de Minas Gerais / Campus Formiga

Resumo. *Esta documentação é referente ao trabalho desenvolvido para a matéria de Sistemas Operacionais do curso de Ciência da Computação do IFMG - Campus Formiga, cujo o objetivo é trabalhar a sincronização entre processos e os efeitos da programação concorrente.*

1. Estrutura

As funções e as estruturas de dados do trabalho proposto estão incluidas todas em um só arquivo, chamado de Formigopolis.c.

1.1. Estrutura Geral

O arquivo formigopolis.c foi dividido nas seguintes seções:

- Constantes: As constantes definidas neste trabalho são as pessoas que irão utilizar a lotérica da cidade, o tamanho da fila e mais duas que recebem o valor 0 e 1 para fazer referência as definições de True e False. Cada pessoa que irá entrar na fila da loterica foi associada a um número: Vanda é a constante que carrega o número 0; Maria o número 1; Paula número 2; Sueli número 3; Valter número 4; Marcos número 5 e o Pedro que recebe o numéro 6.
- Estutura: Esta área do código é o local onde foi definido a estrutura dos dados utilizados. As estruturas utilizadas neste trabalho são os dados das threads e a fila circular, que representa a fila do caixa.
- Variáveis Globais: As variáveis globais deste trabalho são as duas variáveis mutex, uma para a fila e outra para o caixa; o vetor de condições; uma variável int responsável por idetificar o uso do caixa; a fila encadeada e mais outra variável int que guarda o valor passado por parametro na função main.
- Funções: Para tratar a sincronização e outros quesitos do trabalho foram criadas além da função principal (main), mais nove funções:
    1. main: A função main, aborta o processo caso o valor da variável passada por parametro seja inferior a dois, caso não seja essa, função cria as estruturas necessárias para o trabalho, inicializa os dados da thread associando ao vetor criado o nome, a prioridade e o aging de cada pessoa da lotérica.
    2. task_threads: Após inicializar a thread ela é criada invocando a função task_threads. Essa função cria a fila da lotérica, inicializa a flag de cada thread e após isso faz todo o tratamento de sincronização e inserção na fila. Além disso, essa função é responsável por mostrar os resultados no terminal: quem entrou a fila, quem esta sendo atendido, quem foi atendido e quem terminou.
    3. FilaVazia: Essa função confere se a fila da lotérica esta vazia, caso estiver ela retorna true;
    4. AvisaoProx: A AvisaoProx tem o papel de chamar a próxima pessoa da fila;

5. AdicionarFilaEncadeada: Essa função é responsável por adicionar a pessoa passada por parametro em uma fila encadeada, que representa a fila da lotérica;
6. ImprimeFila: Função que imprime a inical do nome das pessoas que estão na fila da lotérica;
7. MeuNomePrint: Retorna um caracter que é a letra inicial do nome passado por parametro;
8. Aging_Fila: responsável por acrescentar +1 nas prioridades das pessoas inseridas na fila que foi passada por parametro (envelhecimento);

2. Decisões Relevantes

Foi utilizado uma variável mutex para controlar a fila da lotérica. Toda vez que a fila é manipulada é dado um lock nessa variável e toda vez que a fila não esta sendo manipulada é dado unlock.

Outra decisão tomada foi a adição de uma flag, que é manipulada apenas por uma thread de cada vez e é responsável por informar se o caixa está ou não vazio

3. Conclusão

Este trabalho foi muito importante, pois com ele foi possível visualizar melhor qual o princípio da utilização das threads, além de identificar e corrigir os possiveis problemas da programação concorrente;