Trabalho de Sistemas Operacionais - Desenvolvimento de um escalonador

Fernando Ollé (18201068)
Gustavo Fernandes (18200553)
Thomazio Giacobbe (18201061)

Pelotas, 2021

1. Introdução

O trabalho envolve construir um programa que simula o escalonamento de um conjunto de tarefas fornecidas previamente em um arquivo de texto.

2. Descrição

A modelagem foi feita baseada em 3 funções: *ordenaCrescente, ordenaDecrescente e separaAsTarefasImprimeArquivo*.

O programa começa fazendo algumas verificações básicas de arquivo, em seguida os processos são lidos juntamente com seus tempos de execução em seguida a função do SJF é chamada, ela ordena baseado nos tempos de execução, sendo os menores primeiro, então a função que divide as tarefas entre os núcleos e imprime os resultados em um arquivo é chamada, a mesma coisa ocorre novamente, porém chamando a função oposta do SJF. Dois arquivos de saída serão criados, *menor_primeiro.txt* e *maior_primeiro.txt*, o nome desses arquivos podem ser modificados pelo último parâmetro na chamada da função.

3. Passos de execução

Para compilar o programa:

gcc funcoes_escalonador.c escalonador.c -o escalonador

Para executar o programa:

./escalonador tarefas.txt *numero_processadores*

Sendo *numero_processadores* um número inteiro, que representa a quantidade de processadores que serão utilizados para o cálculo dos algoritmos implementados.

Para uso do programa, será necessário primeiro escrever no arquivo *tarefas.txt* informações sobre as tarefas que serão escalonadas, em cada linha deve-se inserir o nome do processo e o tempo de cada execução separados por um espaço. Logo após a execução do programa, será possível visualizar a saída através dos arquivos gerados pelo mesmo. O *menor_primeiro.txt*, mostra o resultado quando é executada a função SJF (*Shortest Job First*) que executa as tarefas com o menor tempo de execução

primeiro. Já o segundo arquivo, *maior_primeiro.txt*, faz exatamente o contrário, mostrando primeiro as tarefas com o tempo de execução maior.

4. Dificuldades

A única dificuldade encontrada foi replicar a saída do *maior_primeiro* dado como exemplo na descrição do trabalho, o resultado foi próximo porém não igual.

5. Conclusão

Graças ao Visual Studio Code, que disponibiliza a extensão para pair programming, Live Share, o grupo conseguiu obter um fluxo de trabalho bem agradável onde o conhecimento flui entre integrantes com facilidade, assim gerando uma rápida adaptação ao problema que por sua vez foi resolvido em 2 encontros.

Em suma, este trabalho foi realizado com quase total êxito.