Trabalho Prático de Sistemas Operacionais

Leandro Santos Bergmann (14/0149007)

Gabriel Mesquita (130009121)

Carlos Joel Tavares (130007293)

Universidade de Brasília

Departamento de Ciência da Computação

Brasília, 29 de novembro de 2016

1. **Ferramentas e linguagens usadas**

Para desenvolver o trabalho foi utilizada a linguagem C99 e a biblioteca de *threads* da POSIX. Cada aluno utilizou um editor de texto a sua escolha para a edição. Além disso foi utilizado o GitHub para o versionamento do trabalho.

1. **Descrição teórica e prática da solução dada**

O trabalho foi dividido nos seguintes módulos:

1. Fila: módulo responsável pela estrutura de dados de fila que foi utilizada para fazer as filas de processos e as funções necessárias para manipular a fila;
2. Leitor: módulo responsável pela leitura do arquivo de entrada e criação dos processos;
3. Memoria: módulo responsável pela alocação e desalocação da memoria com base no processo recebido;
4. Processador: módulo responsável por escalonar os processos e também por simular sua execução;
5. Processo: módulo responsável por definir um processo, como imprimi-lo e como calcular seu *offset*;
6. Recurso: módulo responsável pela alocação e desalocação dos recursos de E/S oferecidos pelo SO;
7. psuedoSO: módulo principal que contém a main.

O trabalho funciona da seguinte forma: o programa começa lendo o arquivo e criando os processos. Depois disso, se cria uma *thread* para o processador e para cada processo criado. O único trabalho do processador é escalonar os processos, isto inclui verificar se existe algum processos em alguma das filas, pegar o processos, libera sua execução durante um *quantum*, e colocá-lo de volta na fila caso ele não tenha acabado, sendo que, caso o processo já tenha sido executado muitas vezes ele é posto em uma fila de prioridade menor, isso é feito para tentar garantir que não haja *starvation*. Esse processo é feito em laço até que se verifique que a quantidade de processos que foram executado é igual a que foi criada. Já cada processo tem um ciclo diferente, quando ele são criados é alocada a memoria necessária para eles, depois são alocados os recursos e então eles entram num laço de esperara a CPU e executar. Depois que eles terminam a memoria e os recursos são liberados.

São utilizados somente *threads* e *locks* no trabalho para simular o compartilhamento de recursos e para simular a concorrência dos processos na CPU onde cada recurso, como a memoria, CPU e dispositivos de entrada e saída possuem um *lock* para garantir que somente um processo o utilize por vez.

1. **Principais dificuldades encontradas durante a implementação**

As dificuldades encontradas resumem-se a detalhes sobre a implementação de projeto. Como seria a estrutura da fila, como que seria a memoria, como que seriam os dispositivos de E/S, etc.

1. **Soluções usadas para todas as dificuldades encontradas**

As soluções para as dificuldades são o próprio trabalho, com as divisões dos módulos e suas interfaces.

1. **Papel de cada aluno na realização do trabalho**

Depois de decidir os módulos, ele foram divididos entre os alunos...

1. **Bibliografia**