PUCRS - Escola Politécnica
Disciplina: Sistemas Operacionais - 2023/1 - Trabalho Prático - Escalonamento
Prof. Fernando Luís Dotti

Nesta fase do trabalho introduzimos o escalonamento de processos.

1. Antecedentes

Assume-se que você dispõe do trabalho 1, com paginação, operante. Poderemos carregar vários processos em memória e com um novo comando, seu Sistema Operacional escalonará todos os processos.

2. Escalonamento

Neste passo adotaremos uma política de escalonamento de processos. Assim, faz-se necessário poder parar e retomar a execução de processos, em qualquer ponto (após o final da execução da instrução corrente na CPU). Para isso, teremos que **salvar** o contexto da CPU, no momento que o processo é retirado da mesma. Este estado da CPU será **restaurado** quando o processo retorna à CPU para continuar sua execução. O contexto da CPU é salvo no PCB do processo, em um campo criado para isso.

O escalonamento em um sistema pode ser provocado por diferentes eventos que bloqueiam o processo atual, fazendo-se necessário escalonar outro. **Neste trabalho T2-A** exercitaremos apenas a **perda de processador por tempo de uso. (Em T2-B** exercitaremos perda de processador por **chamada de sistema).** Quando um processo "entra" na CPU, ele executa por um tempo *Delta,* então ocorre uma interrupção do relógio indicando fim de fatia de tempo e o escalonamento ocorre.

Em um sistema real existe um relógio independente no HW. Este relógio interrompe periodicamente a CPU. A CPU desvia para a rotina de tratamento do relógio. Entre outras funções, esta rotina avalia se o processo que está rodando deve ser trocado. Voce pode implementar o relógio como uma thread em loop, em que "dorme" delta tempo e então gera uma interrupção. Ou, para ficar mais fácil (e ter mais controle sobre o que ocorre no sistema), no nosso sistema você pode implementar o relógio como um contador de ciclos de CPU. A cada "delta" instruções liga-se a interrupção. Como para qualquer interrupção, ao final da execução da instrução corrente a CPU verifica se tem interrupção e desvia para a rotina específica.

A rotina de tratamento desta interrupção deve fazer o *salvamento* de contexto do processo interrompido, colocá-lo na fila de processos *prontos* a executar, escolher o próximo processo desta fila que deve ir para a CPU, *restaurar* o contexto deste na CPU, permitir então a CPU a executar o processo por *Delta* unidades - quando este ciclo explanado se repete.

Desta maneira, os processos ciclicamente ganham o direto de executar na CPU. Quando um processo acaba, este é desalocado do sistema e obviamente não será mais executado.

2.2. Fim de Processo

A instrução STOP é uma chamada de sistema. A rotina de tratamento respectiva deve deslocar o processo que executou STOP. Isso significa liberar a memória e desalocar o PCB. Além disso, o escalonamento de um novo processo para ocupar a CPU deve ser efetuado.

3**. Testes**

mantenha todos comandos existentes no T1 e crie um adicional: execAll.
Com este comando você manda executar de forma escalonada todos os processos em memória.

Carregue vários processos em memória. Estes processos estão todos parados, na fila ready.
Então dispare execAll. A partir disso, o ciclo exposto acima deverá acontecer até que todos processos acabem.
Deve ser possível acompanhar o progresso dos diferentes processos ao longo do tempo e as atividades de escalonamento.