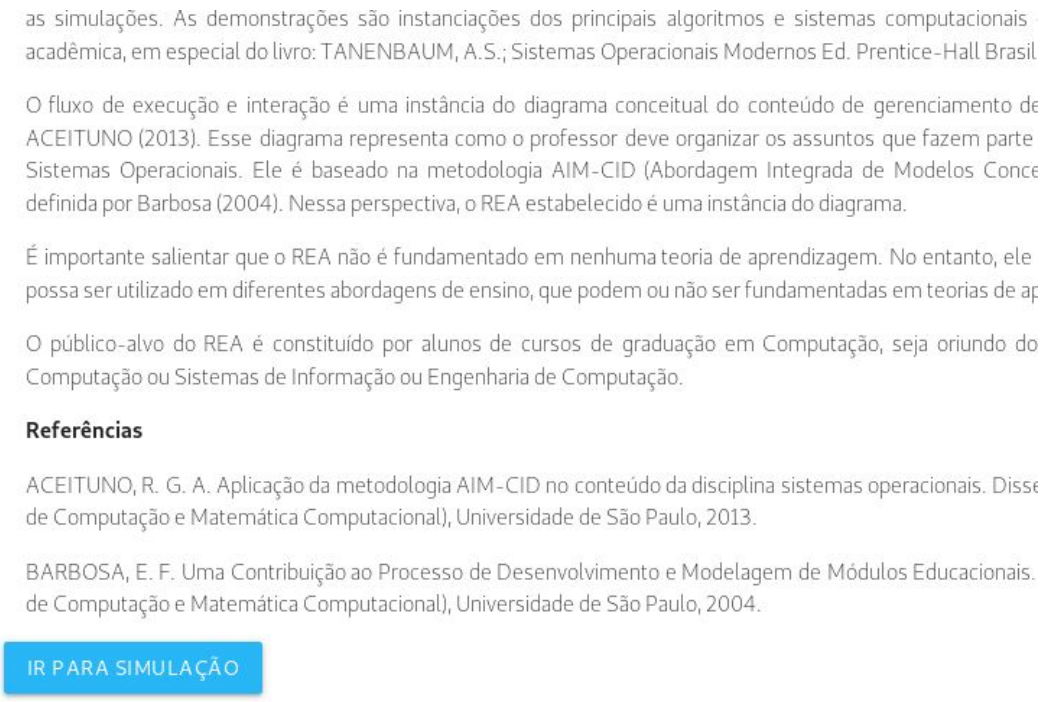
Bem Vindo ao tutorial do GerPro!

Esse tutorial tem como objetivo guiá-lo pelo simulador de escalonamento de processos.

Vamos começar!

Para iniciar a simulação, é necessário acessar a página de simulação, que pode ser encontrada na página inicial, pelo botão ir em simulação no canto inferior esquerdo ou pelo menu de acesso na barra de navegação.


as simulações. As demonstrações são instanciações dos principais algoritmos e sistemas computacionais
acadêmica, em especial do livro: TANENBAUM, A.S.; Sistemas Operacionais Modernos Ed. Prentice-Hall Brasil

O fluxo de execução e interação é uma instância do diagrama conceitual do conteúdo de gerenciamento de
ACEITUNO (2013). Esse diagrama representa como o professor deve organizar os assuntos que fazem parte
Sistemas Operacionais. Ele é baseado na metodologia AIM-CID (Abordagem Integrada de Modelos Conce
definida por Barbosa (2004). Nessa perspectiva, o REA estabelecido é uma instância do diagrama.

É importante salientar que o REA não é fundamentado em nenhuma teoria de aprendizagem. No entanto, ele
possa ser utilizado em diferentes abordagens de ensino, que podem ou não ser fundamentadas em teorias de ap

O público-alvo do REA é constituído por alunos de cursos de graduação em Computação, seja oriundo do
Computação ou Sistemas de Informação ou Engenharia de Computação.

Referências

ACEITUNO, R. G. A. Aplicação da metodologia AIM-CID no conteúdo da disciplina sistemas operacionais. Disse
de Computação e Matemática Computacional), Universidade de São Paulo, 2013.

BARBOSA, E. F. Uma Contribuição ao Processo de Desenvolvimento e Modelagem de Módulos Educacionais.
de Computação e Matemática Computacional), Universidade de São Paulo, 2004.

IR PARA SIMULAÇÃO

Ao iniciar a simulação, é necessário escolher qual tipo de sistema computacional que se deseja simular. O primeiro é o sistema *Batch,* ou em lotes, que se caracteriza pelo processamento em lotes dos processos, de forma que a próxima tarefa somente é iniciada pela CPU quando o processo anterior finalizar sua execução. Esse tipo de sistema computacional não permite interação com o usuário, logo um processo só pode ser interrompido por requisições de entrada ou saída.
O segundo tipo de sistema computacional simulado são os sistemas Interativos, que ao contrário do primeiro, ocorre interação com o usuário, sendo amplamente utilizado em computadores pessoais. Nesse sistema, algoritmos preemptivos são necessários para garantir maior ocupação da CPU. Nestes sistemas, é definido o *quantum* do processo, que trata-se do tempo limite para a utilização da CPU. A definição do tamanho do *quantum* é um desafio, pois o mesmo não pode ser muito grande a ponto da CPU passar muito tempo executando um único processo, porém não pode ser pequeno que a CPU tenha que lidar com o chaveamento constante de contexto.
Se o sistema *Batch* foi o escolhido, tem-se três opções de algoritmos de escalonamento. O primeiro é o First-Come First-Served (FCFS), que se caracteriza pela execução dos processos exatamente na ordem em que foram criados, não tendo um critério para a ordenação dos processos. Já o segundo algoritmo, é o Shortest Job First, que é definido

pela ordenação dos processos pelo tempo de execução estimado, em ordem crescente. O terceiro desses algoritmos, o Shortest Remaining Time Next é preemptivo, ou seja, o tempo de cada processo na CPU é limitado, pois a cada novo processo criado, o escalonador define qual a nova ordem de execução de acordo com o tempo que resta para o processo finalizar sua execução.
Caso o sistema escolhido for o Interativo, apresenta-se cinco algoritmos de escalonamento. O primeiro é o Round-Robin, que trata-se, basicamente, de uma lista circular, onde todos os processos têm igual prioridade de execução e são escalonados em função de sua ordem de chegada, de forma similar ao FCFS, porém com atribuição de um quantum para cada tarefa. O segundo algoritmo trata-se do escalonamento com prioridades, que trata-se de uma versão do Round-Robin com atribuição de diferente valores de prioridade para cada um dos processos. O terceiro algoritmo refere-se ao escalonamento com múltiplas filas, onde cada fila tem a sua prioridade, os sistemas operacionais do tipo linux utiliza uma adaptação desse algoritmo, com o agrupamento dos processos em filas de classes de prioridade, em que as mais altas referem-se a tarefas do próprio sistema operacional e os processos *I/O-bound* do usuário são priorizados em detrimento das demais tarefas. O quarto algoritmo constitui no Shortest Process Next que, de uma forma parecida com o SJF, ordena a fila/lista de processos prontos de forma em que a próxima tarefa a ser escalonada seja sempre aquela com o menor tempo estimado restante para processamento. Por último, temos o algoritmo de escalonamento por loteria, que trata-se de uma estratégia de sequenciamento baseada na atribuição de "bilhetes" para cada um dos processos, simbolizando sua prioridade de execução. Desta forma, os processos que acumulam mais bilhetes têm maior prioridade de serem escalonados em sequência.

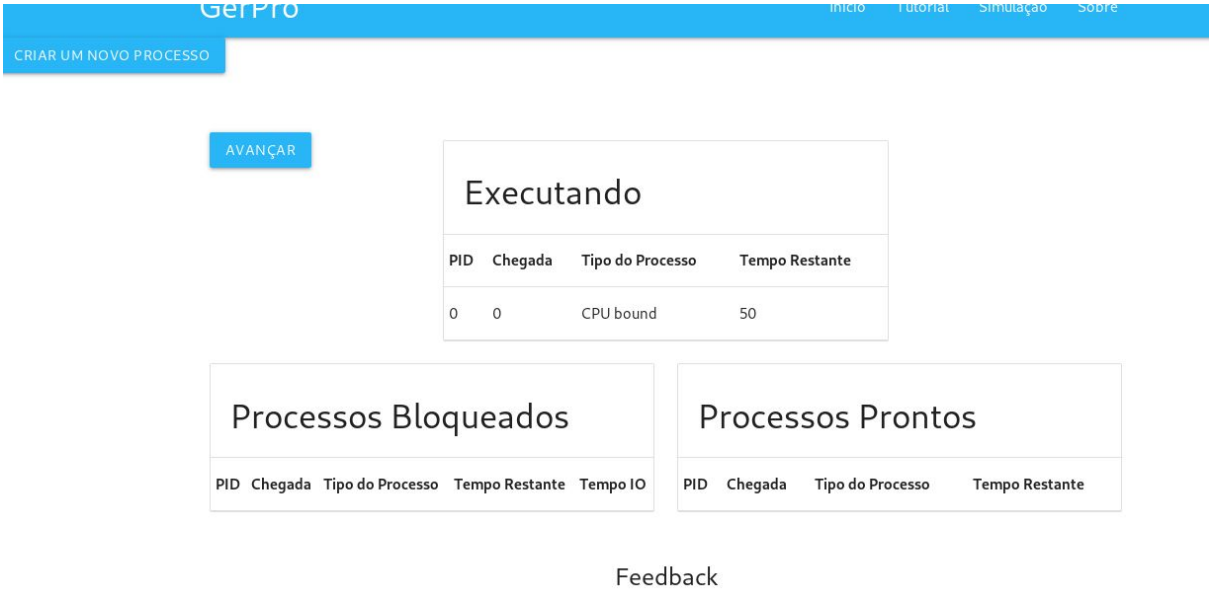
Para qualquer algoritmo escolhido, o simulador necessita que o usuário crie um novo processo, determine se o mesmo é *CPU-bound* ou *I/O-bound*. Nesse exemplo mostra o início da simulação do *FCFS*, que é do tipo *Batch.*

Após a definição do tipo do processo, uma nova página será aberta, onde é possível definir mais processos, ver a lista de processos já criados ou executar a simulação.

Se clicar em Verificar Informações sobre os Processos criados, uma página com a lista de processos criados, seu tipo (*CPU-bound* ou *I/O-bound*), seu *ID* e seu estado (**P**ronto, **B**loqueado, **E**xecutando) será mostrada.

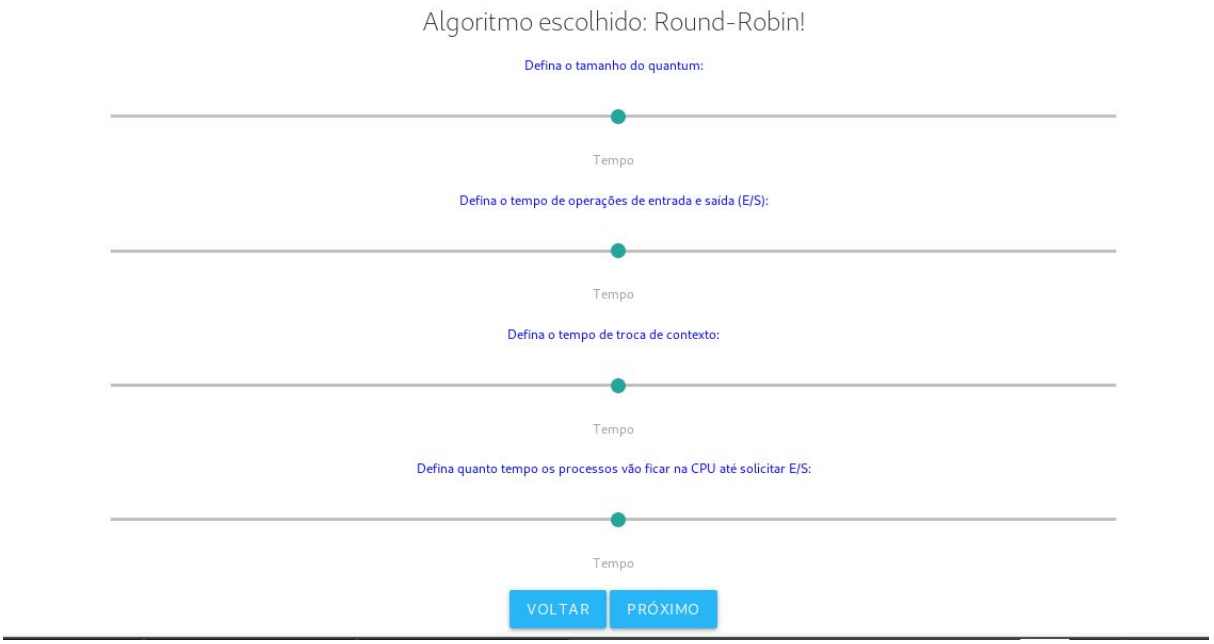
Essa página leva às opções anteriores.
Ao clicar em Executar Simulação, será iniciado a simulação do algoritmo escolhido. Para mostrar o passo a passo da execução, os processos são ilustrados nos seus possíveis estados: na lista de pronto, como processos que estão aguardando seu escalonamento; ou na lista de bloqueado, esperando finalizar o processo de *I/O*; ou como executando, quando o escalonador seleciona o processo para ser executado.

Estes passos são genéricos para todos os algoritmos selecionados. Como cada algoritmo apresenta suas próprias peculiaridades, algumas adaptações foram feitas de modo a representar melhor cada algoritmo.
Como para o Shortest Remaining Time Next, onde é possível criar um novo processo durante a execução corrente, no intuito de ilustrar como o escalonador age em caso de um novo processo ser criado possuir um tempo menor de execução do que aquele que está sendo processado.



Para ilustrar os sistemas Interativos, é necessário que o usuário defina o tamanho do *quantum*, lembrando que seu tamanho influencia no desempenho do CPU. Também são solicitados o tempo que as operações de entrada irá durar, o tempo de chaveamento de

contexto e o tempo em os processos vão ficar na CPU até serem bloqueados por operações de entrada e saída.



Para algoritmos em que os processos apresentam prioridade, também é requerido que o usuário escolha qual a prioridade de cada novo processo no momento de sua criação. Como ilustrado na Figura abaixo.

Para o algoritmo de Loteria, foi adicionado a opção de executá-lo de forma preemptiva, onde os processos possuem limite de *quantum* para serem executados na CPU, de forma que quando esse tempo é atingido, o escalonador retira o processo da CPU e o coloca na lista de processos prontos para serem executados, ou não-preemptiva, em que os processos só saem da CPU quando finalizados.

Tempo

Defina o tempo de operações de entrada e saída (E/S):

Tempo

Defina o tempo de troca de contexto:

Tempo

Defina quanto tempo os processos vão ficar na CPU até solicitar E/S:

Tempo

ATENÇÃO: Como você escolheu o algoritmo Loteria, você precisa definir se fará a simulação consideranção se preempeção ou não

- ○ Preemptivo
- ○ Não preemptivo

VOLTAR PRÓXIMO