AEDS II

Trabalho Prático I
Simulando um servidor de impressão

Entrega via Moodle

1 Contextualização

A interconexão de computadores através das redes permitiu, entre outras coisas, o compartilhamento de recursos computacionais, espaço de armazenamento, poder de computação e, não menos importante, as impressoras. Com o advento das redes, as impressoras deixaram para trás o papel de "periféricos" de um computador para se tornarem um recurso compartilhado por vários computadores. Em avanços mais recentes, as impressoras se desvincularam dos servidores de impressão, que gerenciavam os seus serviços, e passaram elas próprias executar tal gerenciamento com a implementação, em seus firmwares, de software de gerenciamento de suas filas de impressão. O gerenciamento de impressão pode ser visto como uma atividade de escolha do próximo trabalho de impressão a ser executado, em face de uma grande quantidade de trabalhos armazenado em um buffer (memória). Esses trabalhos chegam de diferentes pontos da rede e são escolhidos para serem processados segundo uma política de escalonamento pré-definida.

2 Objetivo

Neste trabalho, seu objetivo será implementar um simulador de um servidor de impressão com o intuito de determinar qual a melhor política e forma de implementação a ser instalada no firmware da impressora. O sistema simulado terá suporte ao gerenciamento de contas e à possibilidade de forçar impressões. O foco da simulação será verificar o funcionamento correto do sistema ao executar as diversas operações do servidor (descritas a seguir) em diferentes situações. Para tanto, você irá implementar 3 políticas de escalonamento das tarefas de impressão. Os escalonamentos estão descritos na Seção 4.

3 Problema

Seu simulador de um servidor de impressão deve obrigatoriamente dar suporte às seguintes operações com suas respectivas regras:

1. Cadastro de impressora

   Esta será sempre a primeira coisa a ser feita no seu simulador. Toda impressora tem as seguintes informações: nome, capacidade de impressão (em páginas por segundo).

2. Cadastro de novo usuário

Cada usuário no sistema possui uma prioridade de impressão em relação aos outros usuários. Podemos enxergar a prioridade de um usuário como seu perfil na empresa. Por exemplo, impressões enviadas pelo presidente da empresa têm prioridade sobre aquelas enviadas pelo estagiário. A empresa possui 5 perfis de usuário. Todo novo usuário no sistema possui as seguintes informações: nome, prioridade.

Importante: você não pode assumir um número pré-determinado de contas de usuário. Isso significa que o sistema não deve ter um número fixo ou máximo de contas de usuário. Cada conta criada deve ser alocada dinamicamente, e a memória utilizada por ela deve ser liberada se a conta for removida.

3. Remoção de usuário

   Quando um usuário é removido do sistema, todas as impressões associadas a ele serão também removidas. *Se um documento desse usuário estiver sendo impresso, a impressão deve ser concluída normalmente como se o usuário ainda estivesse no sistema*.

4. Nova Impressão

   Inserção de uma nova tarefa de impressão no sistema. Toda tarefa de impressão possui as seguintes informações: tempo que foi enviado, usuário, prioridade, número de páginas e tempo máximo de espera para começar a ser impresso *(tarefas que excedem o tempo de espera são contabilizadas como tarefas perdidas)*.

   Nota: Apenas usuários cadastrados no sistema podem imprimir documentos.

5. Relatório

   Seu sistema deverá ser capaz de gerar relatórios de impressão sempre que for requisitado. Para fins de simplificação, a tarefa que está sendo impressa no momento de gerar o relatório deve ser ignorada *(não conta como tarefa a ser processada, nem como tarefa concluída)*.

4 Políticas de escalonamento

Cada tarefa de impressão trás consigo a quantidade de páginas a serem impressas, o usuário dono daquela tarefa assim como a prioridade da tarefa. Note que a prioridade da tarefa não é a mesma do usuário. Quanto menor o valor da prioridade, maior é a prioridade (e.g.: prioridade 1 $>$ prioridade 2).

4.1 Primeiro a chegar, primeiro a sair

A abordagem mais simples de escolha de um trabalho é a *Primeiro a Chegar Primeiro a Sair (FIFO - First In First Out)*. Nesta abordagem, os trabalhos são processados à medida que chegam à impressora, não havendo assim nenhum privilégio entre eles, a não ser a ordem de chegada.

4.2 Primeiro usuários, impressão em segundo

Esta escolha é feita da seguinte forma: selecionam-se as tarefas cujo o usuário tenha a maior prioridade. Dentre essas tarefas, escolhe-se aquela com menor *tempo restante de espera*. Caso haja mais de uma tarefa com mesmo tempo restante de espera, escolhe-se àquela com maior prioridade. Caso

ainda sobre mais que uma, escolhe-se aquela que chegou primeiro. Entende-se por *tempo restante de espera* a seguinte equação:

$$temp\_rest = horario\_chegou + temp\_max\_espera - horario\_atual \quad (1)$$

4.3 Combinação das propriedades da tarefa

- O usuário de prioridade 0 tem prioridade total sobre os outros, de forma que as tarefas de usuários com essa prioridade são sempre as próximas a serem processadas. Caso já haja alguma tarefa com essa prioridade de usuário esperando para ser processada, é processada àquela com maior prioridade. Caso ainda haja mais de uma tarefa, escolhe-se aquela com menor tempo máximo de espera para começar a ser impressa.
- Para as tarefas cujo usuário não tem prioridade 0, é realizado um cálculo para determinar o nível de prioridade dessa tarefa. Tal cálculo é feito da seguinte forma:

$$p = p\_usuario * 0.2 + p\_impressao * 0.6 + \frac{temp\_max\_espera}{temp\_em\_espera} * 0.2 \quad (2)$$

. Sendo $temp\_em\_espera = horario\_atual - horario\_chegou$. Nesse caso, escolhe-se a tarefa que possua o menor p. Em caso de haver tarefas com mesma prioridade final, escolhe-se aquela com menor *tempo máximo de espera* para começar a ser impressa.

5 Entradas

A entrada deverá ser lida de um arquivo.

1. Cadastro de Impressora

   Primeira linha do arquivo de entrada, com o seguinte padrão:

   ```
   <IMPRESSORA> <CAPACIDADE> <ESCALONADOR>
   ```

2. Cadastro de novo usuário

   Pode acontecer em qualquer momento da simulação e possui o seguinte padrão de entrada:

   ```
   1 <USUARIO> <PRIORIDADE>
   ```

3. Remoção de usuário

   Possui o seguinte padrão de entrada:

   ```
   2 <USUARIO>
   ```

4. Nova impressão

   O envio de uma nova tarefa de impressão para o servidor é feito seguindo o padrão:

   ```
   4 <HORARIO> <USUARIO> <PRIORIDADE> <PAGINAS> <TEMP_MAX_ESPERA>
   ```

5. Relatório

   A impressão de relatório é feita em duas situações: quando o usuário solicita e quando a simulação é finalizada. A impressão de relatório é requisitada da seguinte forma:

   ```
   3
   ```

6. Finalizar simulação

   Quando for sinalizada a finalização da simulação, deve-se terminar o processamento de todas as tarefas e só então imprimir o relatório. A finalização da simulação será sinalizada pela seguinte linha:

   ```
   5
   ```

Tipos de dados:

- <IMPRESSORA>: string. Máximo 10 caracteres
- <CAPACIDADE>: inteiro
- <ESCALONADOR>:

  1: Primeiro a chegar, primeiro a sair

  2: Primeiro usuários, impressão em segundo

  3: Combinação das propriedades da tarefa
- <USUARIO>: string. máximo 20 caracteres
- <PRIORIDADE>: inteiro (0 - 4)
- <HORARIO>: inteiro. Padrão *Unix Time*
- <PAGINAS>: inteiro

  Nota: <PAGINAS>/<CAPACIDADE> → inteiro
- <TEMP_MAX_ESPERA>: inteiro. Tempo em segundos

6 Saídas

A saída deverá ser impressa em um arquivo. Seu simulador irá fornecer saída apenas em duas situações: quando for solicitado e quando finalizar a execução da simulação. A saída do relatório deve obrigatoriamente seguir o padrão:

```
## por prioridade de usuario
# total perdas:
<prioridade> <perdas> // Uma linha para cada prioridade
# tempo medio espera:
<prioridade> <tempo>(%.3f)
# tempo minimo espera:
<prioridade> <tempo>(%d)
# tempo maximo espera:
<prioridade> <tempo>(%d)
# numero de paginas impressas:
<prioridade> <paginas>
## por prioridade de tarefa
# total perdas:
<prioridade> <perdas>
# tempo medio espera:
```

```
<prioridade> <tempo>(%.3f)
# tempo minimo espera:
<prioridade> <tempo>(%d)
# tempo maximo espera:
<prioridade> <tempo>(%d)
# numero de paginas impressas:
<prioridade> <paginas>
## geral:
# total perdas: <perdas>
# tempo medio espera: <tempo>(%.3f)
# tempo minimo espera: <tempo>(%d)
# tempo maximo espera: <tempo>(%d)
# numero de paginas impressas: <paginas>
# quantidade de usuarios inseridos: <quantidade>
# quantidade de usuarios removidos: <quantidade>
# quantidade de tarefas removidas com os usuarios: <quantidade>
```

Notas:

- Os tempos das tarefas removidas com os usuários devem ser ignorados

7 Observações para entrada e saída

Conforme explicitado nas seções anteriores, tanto a entrada quanto a saída do seu programa serão feitas utilizando arquivos. Seu programa deverá executar através da seguinte linha de comando:

```
./main <entrada> <saida>
```

onde `<entrada>` e `<saida>` indicam os nomes dos arquivos de entrada e saída, respectivamente, os quais devem ser obtidos utilizando a variável `argv` na linguagem C.

8 Documentação

A documentação deve conter:

1. Introdução: descrição sucinta do problema a ser resolvido e visão geral sobre o funcionamento do programa.
2. Implementação: descrição sobre a implementação do programa. Devem ser detalhados as estruturas de dados utilizadas assim como o porquê da escolha de tais estruturas (de preferência com diagramas ilustrativos) e o funcionamento das principais funções.
3. Estudo de complexidade: estudo da complexidade do tempo de execução dos procedimentos implementados, em especial das operações de inserção de usuário, inserção de tarefa, remoção de usuário, e escolha da próxima tarefa(por escalonador) (notação O).
4. Conclusão: Qual escalonador teve o melhor desempenho e comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
5. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet, se for o caso.

9 Restrições

1. A implementação deve ser feita obrigatoriamente em C.
2. A implementação deverá compilar sem *warnings* em ambiente Linux (Ubuntu 14.04.4 LTS), utilizando a seguinte linha de comando:

   ```
   gcc -Wall *.c -o main
   ```

3. A saída deve obedecer rigorosamente ao padrão especificado (nota: a saída não contém acentuações).

10 Entregáveis

- Código
- Documentação (máximo 10 páginas). Seja direto, o número de páginas da sua documentação não é proporcional à nota da mesma.

Crie um diretório (nomeie o diretório com seu número de matrícula) contendo seu código fonte (todos os arquivos `.c` e `.h`) e a documentação do trabalho (`.pdf`). Comprima o diretório com o seguinte comando:

```
tar zcvf <diretorio>.tar.gz <diretorio>
```

onde `<diretorio>` indica o nome do diretório criado. Submeta o arquivo `.tar.gz` via Moodle.

11 Critérios de Correção

A pontuação do trabalho se divide nos seguintes critérios:

- (10%) Adequação às normas de envio.
- (10%) Funcionamento correto do programa no *valgrind*, ausência de erros de execução (e.g.: *segmentation fault*), validação da entrada (e.g.: checagem se o número de argumentos passados está correto).
- (50%) Código, tempo de execução, e saída correta do programa. A escolha das estruturas de dados utilizadas e forma de implementação e combinação das mesmas para que o escalonador tenha um baixo custo na escolha da próxima tarefa será o principal critério de avaliação desse quesito.
- (30%) Relatório

Comentários gerais

- Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar.
- Clareza, indentação e comentários no programa também vão valer pontos

- Mais uma vez: as saídas NÃO devem conter nenhum tipo de acentuação.
- A saída deve *obrigatoriamente* seguir o padrão especificado no item 6
- Trabalhos copiados serão penalizados com a nota zero.
- Política de atraso: será aceito até 24h depois do prazo, no entanto valerá apenas 70% da nota.