Trabalho
Ganância não tem limites

CI215 - Sistemas Operacionais
Primeiro Semestre de 2011
Prof. Bona

1. Descrição do Problema

O gerente de supermercado GANATELI deseja saber quais são os N conjuntos de 2 produtos mais vendidos a cada K compras efetuadas. Seu objetivo é desenvolver uma implementação de alto desempenho para o cálculo desta lista que será mostrada em um telão no supermercado para os funcionários do mercado que devem rearranjar os pares de produtos proximamente nas prateleiras em uma tentativa de promover o consumismo.

2. Dados de Entrada

Seu programa receberá como parâmetro de entrada o nome de um arquivo contendo, na primeira linha, o valor de N, na segunda linha, o valor de K e nas demais linhas, os códigos dos produtos adquiridos em cada compra. Cada linha é composta pelos produtos de uma única compra. Um exemplo de entrada pode ser visto abaixo:

```
2                   #N
10                  #K
1 2 3 6 7 8         #produtos da compra 1
4 7 8 1             #produtos da compra 2
1 3 6 9 10          # ...
11 12               # ...
1 5 10 3 4          #...
```

obs: Considerar que o código do produto é um inteiro de 32 bits e que cada linha (compra, tem no máximo 255 produtos)

3. Saída do Programa

A saída do programa é baseada em rodadas dadas em função do parâmetro K. A cada rodada R devem ser impressos os N's conjuntos de produtos mais vendidos referentes as ~~R*N~~ R*K primeiras linhas do arquivo de entrada.

Cada rodada inicia com a impressão da *string* "Rodada", seguida de um contador, nas N linhas seguintes são impressos os códigos do conjunto de produtos mais vendidos. Os código de produtos são separados por um único ~~zero~~ Espaço, nenhum zero a esquerda ou espaço adicional deve ser impresso. Abaixo um exemplo da saída esperada:

Exemplo (N=2, K=10):

```
Rodada 1
1 3
7 8
Rodada 2
1  7
5  4
```

Observação importante: Sua saída deve ser exatamente como a especificada porque a saída de seu programa será validada com o resultado esperado utilizando o comando **diff**

Outra observação importante: Se N for 0, então devem ser impressos a cada rodada todos os pares de produto encontrados.

4. Sobre a implementação

Você deve considerar que para resolver este problema você tem a disposição uma máquina com múltiplos núcleos que deve ser programada utilizando **Linguagem C e a biblioteca Pthreads (libpthreads).** Não devem ser usadas bibliotecas mais abstratas que escondam a libpthreads.

A impressão dos resultados de cada rodada deve ser feita com uma *thread*. Você deve tomar cuidado para evitar garantir a correta sincronização desta *thread* de impressão caso contrário os resultados apresentados podem acabar contendo mais ou menos compras do que R*K, alterando a saída esperada do programa.

5. Relatório

Você deve apresentar um relatório (em Latex) contendo:

1. Introdução
2. Arquitetura do Sistema
3. Experimentos
4. Conclusão

Na arquitetura você deve descrever a arquitetura do seu software, apresentando os diferentes grupos de *threads*, as principais estruturas de dados, bem como as técnicas de sincronização utilizadas. Figuras e diagramas são utéis nesta parte. Mas não deixe de usar texto para justificar e defender a arquitetura proposta.

Os experimentos devem apresentar execuções do programa com diferentes parâmetros para o número de threads dos pools e outros parâmetros considerados relevantes. Também é desejável resultados de experimentos realizados com versões iniciais do seu programa, supostamente mais lentas. Experimentos com diferentes cargas (conjunto distintos de arquivos a serem copiados) também são desejáveis.

A ideia é que os experimentem mostrem como as decisões de projeto melhoraram o desempenho do algoritmo. Você deve tentar demonstrar que teu programa é realmente eficiente.

Mais uma observação importante: Não menospreze o relatório, ele é parte da nota e importante. Como ele inclui experimentação você deve terminar a implementação antes da data limite para poder trabalhar no relatório final.

6. Apresentação

Além do relatório, no dia da apresentação será realizado um teste de funcionamento e desempenho do sistema. Todos os trabalhos serão executados utilizando o mesmo arquivo de entradada em uma máquina com 8 núcleos (sujeito a disponibilidade do HW) . A não execução, ou execução com erros, pode, potencialmente gerar problemas gerenciais e administrativos ao supermercado em

questão. Assim nestes casos a nota do trabalho é AUTOMATICAMENTE (0) ZERO. Um tempo de execução máximo (baseado na versão sequencial do programa) também será adotado. Caso o tempo de execução seja maior que este tempo máximo AUTOMATICAMENTE implicará em nota (0) ZERO.

7. Instruções para entrega

Enviar para o e-mail bona@inf.ufpr.br até as 13h30m do dia 23/04/2011 um tar.gz contendo:

23/06/2011

\src (com o codigo fonte e makefile)
\relatorio (com o relatorio em pdf)
\equipe.txt (nome da equipe e e-mails)
\instrucoes.txt (explicando como o programa deve ser executado, em especial qual os ajustes para rodar em uma maquina de 8 núcleos)

Referência

Kay A. Robbins, S. Robbins, Unix Systems Programming: Communication, Concurrency, and Threads.