Projeto PCAM

Alunos

Alexandre Galocha Pinto Junior **10734706**
Eduardo Pirro **10734665**
Fabio Fogarin Destro **10284667**
Fernando Gorgulho Fayet **10734407**
Matheus Sanchez **9081453**

Particionamento

Será realizado um particionamento de dados no vetor de tamanho N em N tarefas, cada uma delas responsável por uma posição da string.

Cada tarefa, inicialmente, será responsável por incrementar, num vetor de contagem, a frequência do caracter presente na sua posição.

Após isso, ocorre uma sincronização, fazendo um novo particionamento de dados, onde cada tarefa agora será responsável por uma posição do vetor de contagem e terá que comparar seu valor com um maior global, a fim de obter a frequência do caractere de maior frequência. A frequência máxima é obtida através de uma operação de redução de ordem logarítmica. Então apenas a tarefa principal imprime todos os caracteres que possuem frequência igual a máxima.

Comunicação

As tarefas serão inicializadas com um vetor de contagem e uma posição da string de entrada. Então elas incrementam a posição referente ao seu caractere, a fim de obter a contagem total após esse processo. Assim que a contagem é terminada é feita uma sincronização global no programa, para que todas as contagens tenham sido feitas.

Então as tarefas são novamente inicializadas (a nova partição de dados) cada uma recebendo uma posição do vetor de contagem, para buscar a maior frequência de letras por meio de uma operação de redução de ordem logarítmica sobre o vetor de contagem de tamanho constante.

Uma vez terminada a operação de redução, uma tarefa principal imprime de forma ordenada cada um dos caracteres que possuem frequência máxima.

Aglomeração

Para definir o número de processos (T) que serão utilizados é necessário saber a quantidade de processadores disponíveis para o uso. No geral o que é feito é escolher o valor de *threads* disponíveis no computador (quantos contadores de programas existem).

Considerando um número T de processos, cada um será responsável por N/T posições da string, contendo assim N/T tarefas por processo. O resto (R) dessa divisão deve ser distribuído adicionando uma tarefa nos primeiros R processos.

Cada processo tem acesso a um vetor de frequências local, com range de 0 a 255. Em seguida, o processo percorre sua parte da string, incrementando as respectivas posições no vetor de frequência a medida em que encontra caracteres na string.

Após todos os processos terem concluído o passo anterior, é feita uma operação de redução de ordem sequencial sobre o número de caracteres (256) entre os vetores de frequência local, gerando o vetor de frequência global.

Em seguida, ocorre um novo particionamento. A cada processo é atribuído 256/T posições do vetor de frequência. Cada processo percorre sua fração do vetor, buscando o maior número de ocorrências daquela fração. Concluído isto, as maiores frequências locais são comparadas, de forma a determinar a maior frequência global, em uma operação de redução de ordem logarítmica sobre o número de caracteres.

Por fim, o vetor de frequências é percorrido de forma sequencial, imprimindo os caracteres correspondentes às posições que possuem um número de ocorrência igual ao número de maior frequência global, identificado anteriormente.

Mapeamento

Cada um dos processos tem a mesma carga de trabalho portanto, eles poderiam ser distribuídos igualmente para cada elemento de processamento. Na prática, o número de processos é equivalente ao número de unidades de processamento, então se temos 4 processadores, por exemplo, poderíamos ter 16 processos, que seriam distribuídos 4 a 4 nos elementos de processamento. Nesse caso assumimos que cada um dos elementos de processamento possui um desempenho homogêneo.

Caso os processadores não possuam um desempenho homogêneo deveria ser levada em conta a diferença entre o desempenho dos processadores para a realização da divisão dos processos, elevando a concentração de processos nas unidades mais potentes.