==========
Universidade de São Paulo - ICMC/EESC - Curso de Eng. de Computação
SSC 0903 - Computação de Alto Desempenho (2021/1)
Primeira Avaliação Bimestral (AB 1) - Resolução Remota e em Grupo
==========

Data da Submissão: 12/06/2021

Nomes dos integrantes deste grupo que resolveram a avaliação:

João Pedro Fidelis Belluzzo - 10716661
Leonardo Cerce Guioto - 10716640
Luis Fernando Costa de Oliveira - 10716532
Rodrigo Augusto Valeretto - 10684792
Thiago Daniel Cagnoni de Pauli - 10716629

Resposta para Q01:

Particionamento: O primeiro passo é realizar a ordenação das notas em cada cidade, usando um algoritmo de ordenação. Nesse caso, será realizado um particionamento de dados, e um grupo de tarefas irá cuidar da ordenação de cada cidade. As tarefas do grupo de tarefas de uma cidade irão paralelizar o processo do algoritmo de ordenação.

Após a ordenação, será realizado um particionamento por dados e por função. O particionamento por dados irá alocar um grupo de tarefas para cada cidade, e o particionamento por função irá alocar as tarefas deste grupo de tarefas de cada cidade para calcular os diferentes valores solicitados da seguinte maneira:

Uma tarefa será responsável por pegar a maior nota da cidade, pegando o último elemento do vetor ordenado e o copiando para a posição equivalente da memória.

Uma tarefa será responsável por pegar a menor nota da cidade, pegando o primeiro elemento do vetor ordenado e o copiando para a posição equivalente da memória.

Uma tarefa (por cidade) será responsável por calcular a mediana, calculando a média entre os dois valores no centro se o tamanho do vetor for par, ou apenas pegando o valor no meio se o tamanho do vetor for ímpar.

Um grupo de tarefas será utilizado para obter a média, onde será utilizada redução logarítmica para obter o somatório do vetor e então armazená-lo, e no final uma tarefa será responsável por dividir o valor do somatório pelo tamanho do vetor.

O desvio padrão deve ser calculado sequencial à média através da subtração de cada elemento da cidade pela média calculada, cada subtração será elevada ao quadrado e somada, o resultado da soma é então dividido pelo número de elementos de cada cidade. Para isso, será utilizado um grupo de tarefas para calcular a diferença entre cada elemento e a média, elevando ao quadrado, e, após isso, os resultados serão somados em uma nova tarefa através de uma redução logarítmica.

Tudo que foi feito para a cidade (ordenação, mediana, média, desvio padrão, máximo e mínimo) será feito novamente, mas dessa vez para as regiões, tratando todas as cidades como um grande conjunto único de dados.

Finalmente iremos repetir isso mas agora para todo o país. Trataremos todas as notas (que correspondem a todo o país) como um único conjunto de dados, realizando a ordenação e cálculo dos valores relevantes.

Para o cálculo da cidade e da região ganhadora será realizado uma paralelização por função, e um grupo de tarefas irá encontrar a região com a maior média enquanto outro grupo de tarefas irá encontrar a cidade com a maior média.

Com todas as informações necessárias, uma única tarefa será responsável por imprimir a saída de forma sequencial.

Comunicação: Nos algoritmos de ordenação paralelos as tarefas vão comunicar entre si através da escrita de sua porção do vetor ordenado em um vetor compartilhado. Para garantir que todas cidades estejam com dados ordenados, será realizada uma sincronização antes do prosseguimento das tarefas.

As tarefas referentes às cidades comunicarão entre si através de um mesmo vetor compartilhado, que já está ordenado devido ao passo anterior. Cada tarefa alocará seu resultado em uma estrutura de dados referente àquela cidade, em uma posição de memória exclusiva para a tarefa. Após isso, será colocada uma barreira para garantir que todas tarefas tenham sido finalizadas antes do prosseguimento.

As tarefas referentes ao algoritmo de ordenação irão se comunicar de forma semelhante à comunicação ocorrida nos algoritmos de ordenação para as cidades. Uma nova barreira será estabelecida, garantindo que todas as regiões estejam ordenadas.

As tarefas referentes aos cálculos desejados para as regiões se comunicarão de forma semelhante às das tarefas para as cidades, realizando seus cálculos e atribuindo em uma estrutura de dados compartilhada, porém com regiões de escrita independentes, referentes a cada região.

No algoritmo de ordenação referente ao Brasil, teremos uma comunicação semelhante às dos algoritmos de ordenação anteriores. As tarefas lerão e escreverão em regiões exclusivas do vetor compartilhado. Uma nova barreira é necessária para garantir que todo vetor compartilhado esteja ordenado.

As tarefas referentes aos cálculos do Brasil se comunicarão entre si, escrevendo seus resultados em uma estrutura de dados compartilhada referente ao país, de forma semelhante à das regiões. Uma nova sincronização é necessária para garantir que todos cálculos sejam realizados.

A impressão será realizada de forma sincronizada, garantindo que os dados sejam impressos na ordem correta.

Aglomeração: Considere P o número de processadores, R o número de regiões, C o número de cidades e A o número de alunos por cidade. Caso alguma divisão do número de tarefas pelo número de threads não dê inteiro, as tarefas restantes serão distribuídas cada uma para uma diferente thread, começando da primeira e distribuindo sequencialmente.

Para a ordenação das notas para cada cidade será utilizado quicksort, e cada grupo de tarefas responsável pela ordenação de uma única cidade será unido e executado por uma única thread, para que não ocorra um número exageradamente grande de threads. Serão criadas P threads, onde cada thread será responsável por realizar a ordenação das notas de C / P cidades.

Para os cálculos do mínimo, máximo, mediana, média e desvio padrão referentes às cidades, as tarefas do particionamento por função serão aglomeradas, e novamente teremos P threads e cada thread cuidará de realizar C / P aglomerados de tarefas.

Para a ordenação das notas das regiões também será utilizado o quicksort. Cada quicksort será realizado de forma sequencial. Teremos a divisão da ordenação das regiões em P threads, sendo cada thread responsável por R / P regiões.

Para os cálculos do mínimo, máximo, mediana, média e desvio padrão referentes às regiões, as tarefas do particionamento por função serão aglomeradas, e novamente teremos P threads e cada thread cuidará de realizar R / P aglomerados de tarefas.

Para a ordenação das notas do país todo as tarefas responsáveis pela ordenação serão aglomeradas em apenas uma thread, e as notas serão ordenadas sequencialmente.

As tarefas referentes ao cálculo da melhor cidade e região serão aglomeradas e realizadas de forma sequencial.

Mapeamento: Considerando que os nós do cluster tem capacidade homogênea de processamento e que a aglomeração das tarefas foi utilizada de forma a se obter uma granulação grossa, apenas com paralelismo de dados, temos que os grupos de dados a serem processados são também homogêneos, desta forma, a distribuição de tarefas pode seguir um modelo estático.

No caso de PROCs heterogêneos, considera-se uma heurística de distribuição aos nós com menores cargas de trabalho, levando em conta a capacidade de cada unidade de processamento. A tarefa de ordenação dos dados do país como um todo, por exemplo, deveria ser executada pelo elemento de maior capacidade de processamento, já que é a mais custosa do processo.

Resposta para a Q02: A granulação obtida na etapa de aglomeração pode ser definida como grossa. Ao realizar o processo de aglomeração, diversos paralelismos tiveram que ser retirados, e o paralelismo por dados foi aglomerado, de forma que o volume de dados era distribuído entre os processadores da máquina. Desta forma, pode-se dizer que o resultado obtido foi uma granulação grossa, que mostrou-se mais eficiente.

Resposta para a Q03: Podemos considerar que a solução determinada no PCAM é escalável, em geral. Com um aumento do volume de dados e de processadores, a maioria das tarefas realizadas seriam otimizadas. Porém, o principal gargalo do processo é a ordenação dos dados do país como um todo. Essa tarefa é bem custosa e, na prática, não é paralelizada. Isso pode ser uma limitação considerável com uma quantidade de dados maior, já que, independentemente do número de elementos de processamento disponíveis, ela será executada por apenas um deles.

Resposta para a Q05: A máquina paralela que executará a solução é um *cluster* de processadores. Segundo a Taxonomia de *Flynn*, ela é conhecida como MIMD (*Multiple Instruction Multiple Data*), onde pode-se executar várias instruções diferentes sobre vários conjuntos de dados distintos.

Resposta para a Q06:

Entrada: 10
1000
1000
13473871237

	Código + Diretivas de compilação utilizadas							
Exec	studentsseq.c	studentspar.c	studentsseq.c -o3	studentspar.c -o3	studentsseq.c -ofast	studentspar.c -ofast	studentsseq.c -o3 -ofast	studentspar.c -o3 -ofast
1	1,61538833	0,78687088	1,66587149	1,24132644	1,61271911	0,79813197	1,66032435	0,84012993
2	1,67588609	0,88293261	1,64608701	0,89959226	1,66282357	0,83670296	1,67522573	0,86992355
3	1,61666308	0,86173847	1,63627474	0,89003691	1,65970421	0,85678278	1,63307471	0,94362054
4	1,63129317	0,80390563	1,64643416	0,91599782	1,66310739	0,94273699	1,62347835	0,88116264
5	1,65931976	0,85711208	1,64809978	0,85343475	1,64885231	0,91558439	1,67309918	0,84527891
6	1,62027747	0,85608389	1,66857606	1,18581372	1,71875379	0,88898936	1,65272362	0,86668819
7	1,62913354	0,82110717	1,61524856	1,27779848	1,65657939	0,88284351	1,65895299	0,81510419
8	1,63923121	0,78802131	1,66498731	0,83650811	1,64927848	0,86605868	1,69416195	0,81244567
9	1,69133294	0,79525907	1,63582578	0,85474306	1,64453363	0,94874034	1,62438812	0,79687923
10	1,62239372	0,84794623	1,66938708	0,81108254	1,80515416	0,93392275	1,65521924	0,79811871
Média	1,630213355	0,8345267	1,64726697	0,894814585	1,6581418	0,885916435	1,657086115	0,84270442
SpeedUp	1,953458595		1,84090313		1,871668404		1,966390677	
Eficiência	0,2441823244		0,2301128912		0,2339585505		0,2457988346	

Para calcular o *speedup* (absoluto), são necessários os tempos de execução do algoritmo sequencial e do paralelo, de forma que:

$$speedup_{abs} = T_{seqM} / T_{parM}$$

A eficiência é a relação entre o *speedup* e o número de processadores envolvidos na computação. Dessa forma, tem-se:

$$E_f = speedup_{abs} / N_{proc}$$

$$E_f = speedup_{abs} / 8$$