In The Name Of God

Assignment No.2 on

Multicore Programming

Dr. Hamid Sarbazi-Azad

Dr. Arash Tavakkol

Important Notes:

1. This assignment should be done individually.

2. You should write it using OpenMP.

3. The deadline for this assignment is on Thursday (30th Ordibehesht), Very soon delivery (till 26th Ordibehesht) gets 10% extra mark and a 10% penalty will be given for each one day latency. No delivery will be accepted after 5 days.

4. Cheating will make your grade -100 !

5. Your files should contain your codes and a report that explains your work. The grading policy includes: algorithm design, code implementation, correctness of implementation, clear comments and runtime execution performance.

6. You should zip all your files in this format: AssignmentSeriesX_StudentID.zip.

7. You should email your assignments to multicore16@gmail.com.

8. I will create a forum in CW about the assignments. If anyone has a question please ask me in the forum.

Regards,

Sina Daraby
1. **List Intersection**: In this assignment you are supposed to implement a parallel version of the List intersection algorithm and compare it to the serial version. Consider you have \( N \) Integer lists. Each list has \( M \) elements. Each element in any list is within a range. In other words we know that each number in any list is between 0 and \( \text{range} \). The algorithm should calculate the intersection of all lists. In mathematical notation it is: \( l_1 \cap l_2 \cap l_3 \ldots \cap l_n \). For example consider the following example:

Inputs:

\( N = 2, M = 3, \text{Range} = 10 \)

\( l_1 = \{3,2,7\}, l_2 = \{10,1,3\} \)

Output:

\( l = \{3\} \)

First you are expected to implement a serial version of this algorithm (it is so simple). Then you should write a parallel version. Obviously there may be more than 1 approach for implementing this algorithm in parallel. You are free to choose your own method. Then you should compare the parallel version speed up for various \( N \)s And \( M \)s and Threads. You should test the speed up for \( N = 10,50 \) and for \( M = 10k, 100k \) and threads=8,16,32,64. You should assume that for each configuration the value of the range is equal to \( M \). The lists should be initialize with random integer numbers (between 0 and range). Report files should have a brief description of parallelization method you used, and a result section.

Ex. Speedup table (the numbers are fake!)

<table>
<thead>
<tr>
<th>( N/M )</th>
<th>10/10K</th>
<th>10/100k</th>
<th>50/10K</th>
<th>50/100k</th>
</tr>
</thead>
<tbody>
<tr>
<td>#threads</td>
<td>---------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2.43</td>
<td>3.01</td>
<td>3.48</td>
<td>3.56</td>
</tr>
<tr>
<td>16</td>
<td>4.1</td>
<td>5.1</td>
<td>6.1</td>
<td>5.1</td>
</tr>
<tr>
<td>32</td>
<td>9.2</td>
<td>8.7</td>
<td>9.1</td>
<td>9.7</td>
</tr>
<tr>
<td>64</td>
<td>12</td>
<td>13</td>
<td>12.9</td>
<td>13.1</td>
</tr>
</tbody>
</table>