MAXIMUM SUM SUBSEQUENCE

Mahak Mukhi
Mohana Bhunekar
Determine a subsequence of data set that sums to the maximum value with respect to any other subsequence of the data set.

More formally, if we are given a sequence \( X = <x_0, x_1, x_2, \ldots, x_n> \) we are required to find a set of indices \( u \) and \( v, u \leq v \), such that the subsequence \( <x_u, x_{u+1}, \ldots, x_v> \) has the largest possible sum, \( x_u + x_{u+1} + \ldots + x_v \), among all subsequences of \( X \).

This problem is non-trivial only if the given sequence has both positive and negative values.
Protein and DNA sequence analysis

- To locate biologically meaningful segments, e.g. conserved segments, GC-rich regions, non-coding RNA genes, and transmembrane segments.
- A common approach is to assign a value to each residue, and then look for consecutive subsequences with high sum or average.

Traffic Monitoring

- Example: Say for a bridge, given the numbers of vehicles entering and exiting at various passes, you can determine the busiest routes.
• We worked this problem on a Linear Array

• But, we don’t have a Linear Array!
  • But we always can simulate the same behavior virtually.
  • We wrote the MPI code such that the program behaves as if it were running on a Linear Array.

• It comes with a cost though.
  • The communication time is more than what is intuitive.
The algorithm that I am considering to implement is from “Algorithms Sequential & Parallel: A Unified Approach” by Russ Miller and Laurence Boxer.

First compute the parallel prefix sums \( S = \{ p_0, p_1, \ldots, p_{n-1} \} \) of \( X = \{ x_0, x_1, \ldots, x_{n-1} \} \), where \( p_i = x_0 \otimes \ldots \otimes x_i \).

Next, compute the parallel postfix maximum of \( S \) so that for each index \( i \), the maximum \( p_j, j \geq i \), is determined, along with the value \( j \).

Let \( m_i \) denote the value of the postfix-max at position \( i \), and let \( a_i \) be the associated index, i.e., \( p_{a_i} = \max \{ p_i, p_{i+1}, \ldots, p_{n-1} \} \).
Next, for each $i$, compute $b_i = m_i - p_i + x_i$, the maximum prefix value of anything to the right minus the prefix sum plus the current value.

Finally, the solution corresponds to the maximum of the $b_i$’s, where $u$ is the index of the position where the maximum of the $b_i$’s is found and $v = a_u$. 
TIME COMPLEXITY

- This algorithm runs in $\Theta(n)$ time on a Linear Array.

- And the optimal cost of $\Theta(n)$ is achieved with $n^{1/2}$ processors.
<table>
<thead>
<tr>
<th>Process</th>
<th>Values</th>
<th>1000</th>
<th>10000</th>
<th>100000</th>
<th>1000000</th>
</tr>
</thead>
<tbody>
<tr>
<td>4(4x1)</td>
<td></td>
<td>0.0142</td>
<td>0.0832</td>
<td>1.1636</td>
<td>12.5648</td>
</tr>
<tr>
<td>8(8x1)</td>
<td></td>
<td>0.0271</td>
<td>0.1795</td>
<td>0.7540</td>
<td>19.2160</td>
</tr>
<tr>
<td>16(8x2)</td>
<td></td>
<td>0.0301</td>
<td>0.1571</td>
<td>2.8025</td>
<td>27.9722</td>
</tr>
<tr>
<td>32(8x4)</td>
<td></td>
<td>0.0306</td>
<td>0.5545</td>
<td>6.7950</td>
<td>92.8327</td>
</tr>
</tbody>
</table>
RUNNING TIME

Process vs Values

1000 10000 100000

4(4x1) 8(8x1) 16(8x2) 32(8x4)
CONCLUSION

- Linear Array is not a very efficient architecture to go with.

- With the increase in no of processes the running time increases too because of the communication diameter.

- After a while the communication time completely overshadows the execution time.
FUTURE GOALS

• For the rest of the semester, we would like to conduct experiments with even larger data.

• In the next phase we plan to work this out on various other architectures as well.

• A comparative analysis would be great!
ACKNOWLEDGEMENT AND REFERENCES

- “Algorithms Sequential & Parallel: A Unified Approach” by Russ Miller and Laurence Boxer

- [http://wordaligned.org/articles/the-maximum-subsequence-problem](http://wordaligned.org/articles/the-maximum-subsequence-problem)

- “Genomic Sequence Analysis: A Case Study in Constrained Heaviest Segments” by Kun-Mao Chao
LET’S TALK!

- Session open for discussion
Thank you