Assignment HPC-3

Title: Parallel Sorting Algorithm

Problem statement
For bubble sort and merge sort, based on existing sequential algorithms, design and implement parallel algorithm using all resources available

Objective
1. Understand concept of bubble sort and merge sort based on sequential algorithm
2. Understand concept of parallel programming
3. Compare performance by varying number of processors used and also with sequential algorithm

Outcome
To be able to display result for parallel bubble sort and merge sort
Analyze performance by varying number of processors used and also with sequential algorithm

Software and hardware requirements:
OS: Fedora 20 / Windows 10 (64-bit)
Python Scipy libraries / RStudio with R libraries
RAM: 4 GB, HDD: 500 GB

Theory related concepts

I Bubble sort algorithm

i Sequential
- Go through the list
- Compare consecutive elements and swap them if necessary
- Stop when no more are of ordered pair

Slow and efficient = $O(n^2)$

ii Parallel (odd even transposition)
- compare all pairs in list
- stop when sorted flag = true at beginning of each iteration (2 phase)
- if any processor swaps, sorted = false

Complexity : Do $(n-1)$ comparisons for each iteration (2 phase)
if unlimited processors → $O(n)$

II Merge sort

i Sequential
Divide and conquer
- use recursion
- combine solutions of sub problems

Complexity = $O(\log n)$

ii Parallel
Parallelize processing of sub problems max parallelization with one process per node (at each layer)
Complexity → $O(n \log n)$ → 'n' elements
'log n' for tree depth
if n processor → $O(\log n)$

Open MP
- For concurrent and synchronised data handling
- Conditional parallelization, eg: Scalar exp.
- Degree of concurrency. → num-threads (int exp)
- Data handling → private, first private shared

* program omp parallel if parallel ==1
num_thread (8) private (a) shared (b) ...
{ // * structured block * }

Open MP task group → specifies a wait on completion of child tasks of current tasks and their descendant tasks.

Test cases

Algorithms	Input size	Serial Time	Parallel Time	Speed up
Merge Sort	100000	0.0124	0.015	0.862
Bubble Sort	10000	0.338	0.423	0.798
Merge Sort	10000	0.0013	0.00146	0.905
Bubble Sort	1000	0.0027	0.0054	0.501

Conclusion
Thus successfully performed merge sort and bubble sort and analysed performance with serial algorithms using Open MP.