# HPC ASSIGNMENT 2

## Group 14

Tanisha Rana SE20UCSE202
Jahnavi Siripurapu SE20UCSE178
Harshit Reddy SE20UCSE051
Ridhi Bigala SE20UCSE138
Amrutha Manne SE20UARI017
Agastya Gandra SE20UARI011

### 1)Merge sort technique (by considering bottom up approach)

Suppose we wish to redesign merge sort to run on a parallel computing platform. Just as it it useful for us to abstract away the details of a particular programming language and use pseudocode to describe an algorithm, it is going to simplify our design of a parallel merge sort algorithm to first consider its implementation on an abstract PRAM machine. However, we will also consider the realities of practical platforms and discuss a likely version that would get implemented in practice.

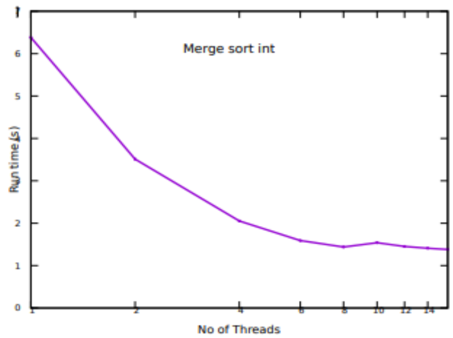
Suppose we have a PRAM machine with **n** processors. A theoretically possible case, but unlikely in practice, would be that that we would have plenty of processors, so that n >= N, the size of our data to be sorted. This relatively impractical case is referred to as fine-grained parallelism. Fine-grained case provides a useful starting point for eventually arriving at course-grained solutions.

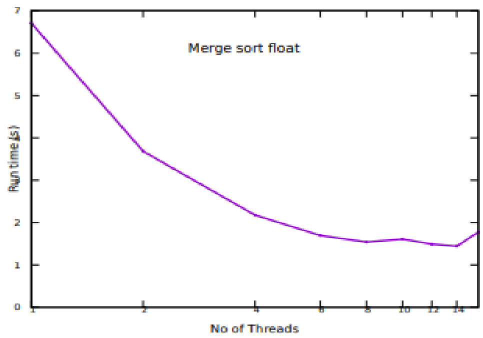
When thinking about the sequential merge sort algorithm, a way to visualize what is happening and reason about its complexity is to look at its recursion tree. For example, if there were 8 items to be sorted, The number of items each recursive call would be working on. The merge function is called on every node but the leaves of this tree, where the input m is a single element.

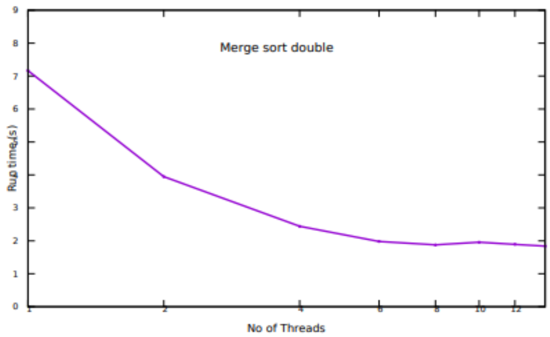
The sequential steps of this algorithm are taking place by following a depth-first traversal through this tree following the left children first. Take a moment to visualize, starting from the top node, which node begins executing the next merge sort function, and the next, and so on.

Now let’s examine which of these operations that were running one after the other in the sequential version could be run simultaneously on separate processors. A natural way to split the work that can be run ‘in parallel’ is to do the work required at each level of the tree. Note that when considering a parallel solution, we use an **iterative approach** and envision starting our work at the bottom of the tree, moving up one level at each iteration. Each individual process is simply executing a merge on a range of values in the array (a single array could be used to sort in place, or a result could be used if you wish to preserve the original input).

We have 8 elements to be sorted and what would result at each level of the tree. At the level of the leaves of the tree, there is no real work, and we could begin by envisioning that level as our N data items, all shared in memory by our processors. At the next level, each of 4 processors can work on disjoint sets of 2 separate data items and merge them together. Once those are done, 2 processors can each work on merging to create a list of 4 sorted items, and finally, one last processor can merge those 2 lists of 4 to create the final list of 8 sorted items. This is parallel merge sort algorithm.




**2) N-Queens on the N × N chessboard in non-attackable positions, N ∈ {10, 12, 14, 16}.**

N-queens Problem sequential Solution

The N-queens problem is a generalization of the well-known 8-queens problem, which consists in arranging 8 queens on a chessboard so that none can take another. A queen attacks another if they are located on the same diagonal, row or column. In the case of the N-queens, N queens are placed on a NxN board. There exists a known number of solutions; for instance, there are 92 solutions for placing 8 queens on a 8x8 board . Other problems are derivable from this. Among them, it is worth mentioning the problem which, given a NxN board, looks for the lesser quantity of queens that can be placed so that all the board squares are attacked by some queen . An initial solution to the N-queens problem, by way of a sequential algorithm, consists in testing all the possible placement combinations of queens on the board and choosing the valid ones. The combination in which no queen of the board is attacked by another is considered as valid. This solution can be upgraded by discarding, during the search, those ways by which a solution to the problem cannot be found . We used 03 optimization to get values faster.

Parallelization of the N-queens problem

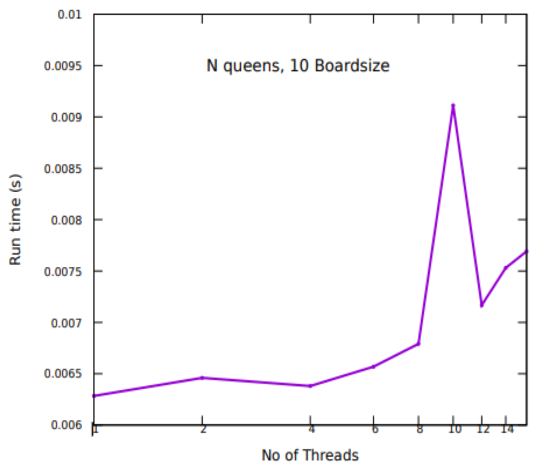
All the algorithms perform N stages where, in each stage I, they try to place queens on all the valid position of row i.

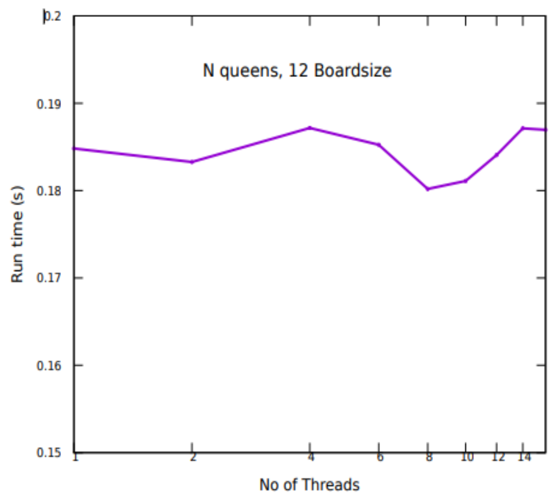
1- Solution with processor pipe: Let P1..PN be processors and Ti the set of boards with queens placed on valid positions on the first i rows. A processor Pi is in charge of placing the queens on row

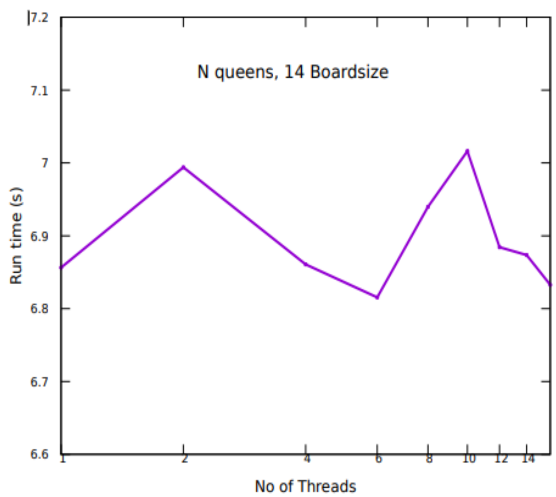
i. The work begins with processor P1, which places a queen on each possible position of row 1, thus obtaining a set of initial boards (T1). Each of these obtained boards passes to processor P2, which places a queen on all the valid positions of row 2 in each of them, passing the obtained boards (T2) to the next processor. The process is repeated until N processor is reached. This receives from processor PN-1 the boards with queens located accurately on the first N-1 rows (TN-1), and its task is to place the queens on row N, in order to obtain the set of solutions to the problem (TN).

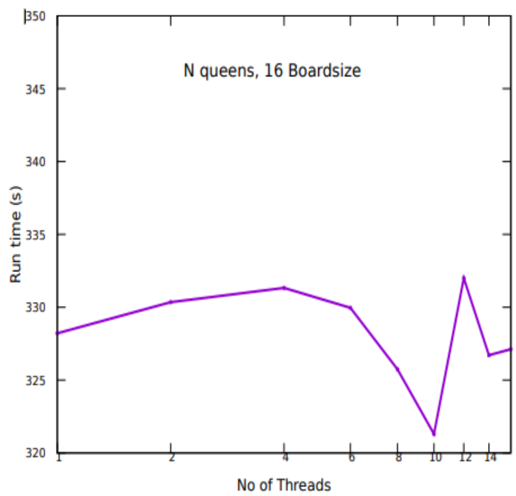
2.-Loosely coupled processor N Solution

Let P1..PN be processors and Ti,j the set of boards where i represents the column in which the queen was placed on the first row, and j the row up to which the board has queens. For instance: T2,4 is the set of all the boards with queens placed accurately on the first four rows, being the queen of the first row placed on the second column. A processor Pi is in charge of finding the set of boards (Ti,N) with all the possible solutions having placed the queen on the first row column i. In this case there are no communication between the processors during the search of the solutions. Each processor works independently.



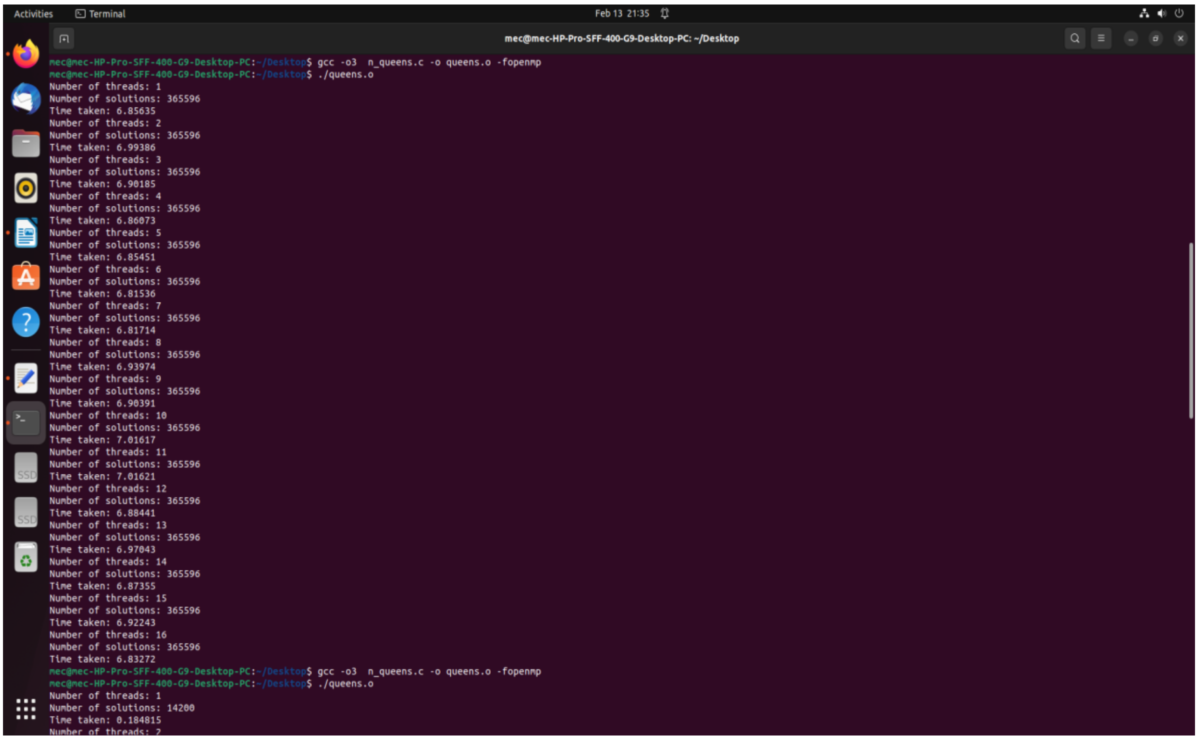



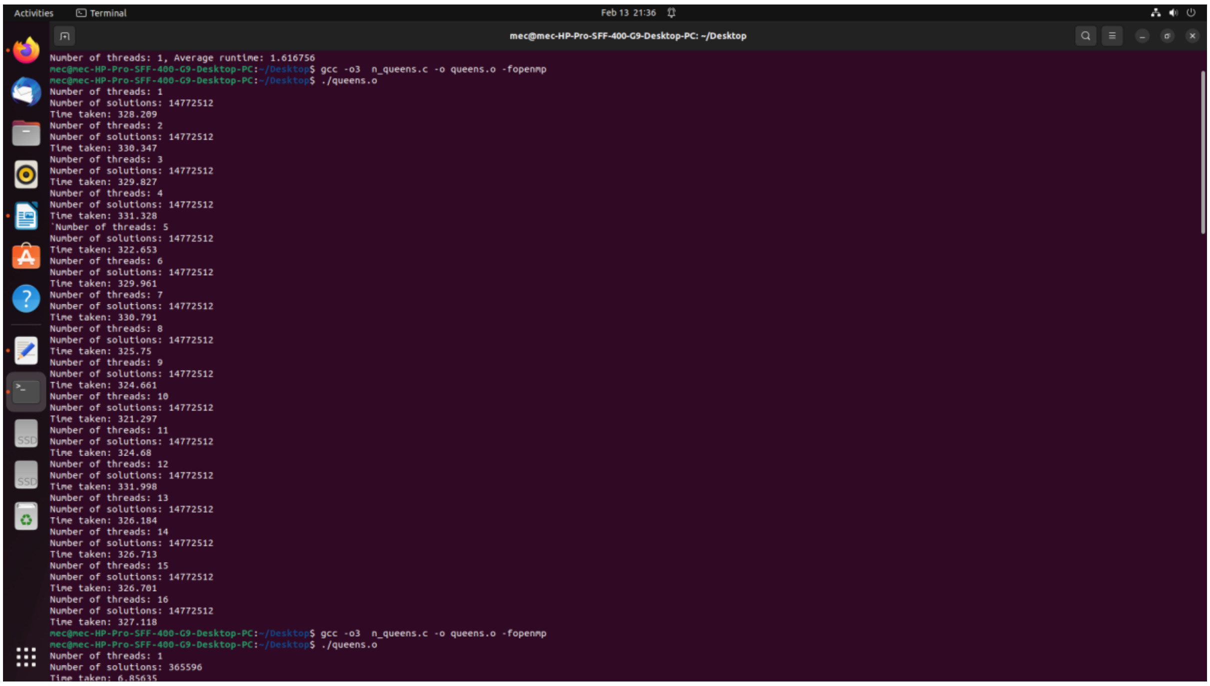



```
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 724
Time taken: 0.00628371
Number of threads: 2
Number of solutions: 724
Time taken: 0.0064601
Number of threads: 3
Number of solutions: 724
Time taken: 0.00645032
Number of threads: 4
Number of solutions: 724
Time taken: 0.0063797
Number of threads: 5
Number of solutions: 724
Time taken: 0.00660272
Number of threads: 6
Number of solutions: 724
Time taken: 0.00656818
Number of threads: 7
Number of solutions: 724
Time taken: 0.00673156
Number of threads: 8
Number of solutions: 724
Time taken: 0.00679116
Number of threads: 9
Number of solutions: 724
Time taken: 0.00703326
Number of threads: 10
Number of solutions: 724
Time taken: 0.00911315
Number of threads: 11
Number of solutions: 724
Time taken: 0.00703496
Number of threads: 12
Number of solutions: 724
Time taken: 0.00716667
Number of threads: 13
Number of solutions: 724
Time taken: 0.00763216
Number of threads: 14
Number of solutions: 724
Time taken: 0.00752991
Number of threads: 15
Number of solutions: 724
Time taken: 0.007615
Number of threads: 16
Number of solutions: 724
Time taken: 0.00769098
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$
```

```
Activities    Terminal    Feb 13 21:35
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC: ~/Desktop
Number of solutions: 365596
Time taken: 6.83272
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 14200
Time taken: 0.184815
Number of threads: 2
Number of solutions: 14200
Time taken: 0.183274
Number of threads: 3
Number of solutions: 14200
Time taken: 0.182681
Number of threads: 4
Number of solutions: 14200
Time taken: 0.187171
Number of threads: 5
Number of solutions: 14200
Time taken: 0.18108
Number of threads: 6
Number of solutions: 14200
Time taken: 0.185236
Number of threads: 7
Number of solutions: 14200
Time taken: 0.179982
Number of threads: 8
Number of solutions: 14200
Time taken: 0.180188
Number of threads: 9
Number of solutions: 14200
Time taken: 0.180457
Number of threads: 10
Number of solutions: 14200
Time taken: 0.181102
Number of threads: 11
Number of solutions: 14200
Time taken: 0.183579
Number of threads: 12
Number of solutions: 14200
Time taken: 0.184063
Number of threads: 13
Number of solutions: 14200
Time taken: 0.186302
Number of threads: 14
Number of solutions: 14200
Time taken: 0.187119
Number of threads: 15
Number of solutions: 14200
Time taken: 0.185838
Number of threads: 16
Number of solutions: 14200
Time taken: 0.18696
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 724
```


Activities
Terminal
Feb 13 21:35
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC: ~/Desktop
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 365596
Time taken: 6.85635
Number of threads: 2
Number of solutions: 365596
Time taken: 6.99386
Number of threads: 3
Number of solutions: 365596
Time taken: 6.90185
Number of threads: 4
Number of solutions: 365596
Time taken: 6.86073
Number of threads: 5
Number of solutions: 365596
Time taken: 6.85451
Number of threads: 6
Number of solutions: 365596
Time taken: 6.81536
Number of threads: 7
Number of solutions: 365596
Time taken: 6.81714
Number of threads: 8
Number of solutions: 365596
Time taken: 6.93974
Number of threads: 9
Number of solutions: 365596
Time taken: 6.90391
Number of threads: 10
Number of solutions: 365596
Time taken: 7.01617
Number of threads: 11
Number of solutions: 365596
Time taken: 7.01621
Number of threads: 12
Number of solutions: 365596
Time taken: 6.88441
Number of threads: 13
Number of solutions: 365596
Time taken: 6.97043
Number of threads: 14
Number of solutions: 365596
Time taken: 6.87355
Number of threads: 15
Number of solutions: 365596
Time taken: 6.92243
Number of threads: 16
Number of solutions: 365596
Time taken: 6.83272
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 14200
Time taken: 0.184815
Number of threads: 2


Activities
Terminal
Feb 13 21:36
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC: ~/Desktop
Number of threads: 1, Average runtime: 1.616756
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 14772512
Time taken: 328.209
Number of threads: 2
Number of solutions: 14772512
Time taken: 330.347
Number of threads: 3
Number of solutions: 14772512
Time taken: 329.827
Number of threads: 4
Number of solutions: 14772512
Time taken: 331.328
`Number of threads: 5
Number of solutions: 14772512
Time taken: 322.653
Number of threads: 6
Number of solutions: 14772512
Time taken: 329.961
Number of threads: 7
Number of solutions: 14772512
Time taken: 330.791
Number of threads: 8
Number of solutions: 14772512
Time taken: 325.75
Number of threads: 9
Number of solutions: 14772512
Time taken: 324.661
Number of threads: 10
Number of solutions: 14772512
Time taken: 321.297
Number of threads: 11
Number of solutions: 14772512
Time taken: 324.68
Number of threads: 12
Number of solutions: 14772512
Time taken: 331.998
Number of threads: 13
Number of solutions: 14772512
Time taken: 326.184
Number of threads: 14
Number of solutions: 14772512
Time taken: 326.713
Number of threads: 15
Number of solutions: 14772512
Time taken: 326.701
Number of threads: 16
Number of solutions: 14772512
Time taken: 327.118
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ gcc -o3  n_queens.c -o queens.o -fopenmp
mec@mec-HP-Pro-SFF-400-G9-Desktop-PC:~/Desktop$ ./queens.o
Number of threads: 1
Number of solutions: 365596
Time taken: 6.85635