CSC 345-02

Project #2 Multithreaded Programming

Chris Jenson and Brian Worts

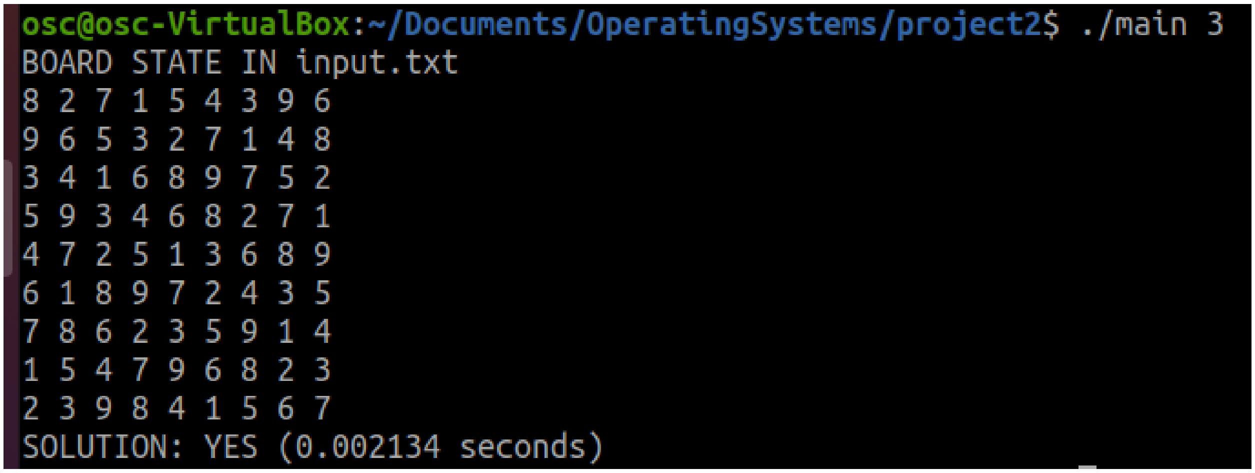
03/19/21

Program requirements(60 pts):

Implemented

•*[10pts] The program should take input from input.txt, 81 numbers listed up in 9 rows, and space separated. Your program should be able to read in this information and print out the read board states correctly. The program also prints out whether the board state is a solution or not.*

Our program properly takes input from a file named input.txt containing 81 numbers in rows and space separated. It is read in and printed to the user in the desired (space separated) format as seen in Figure 1. The program then checks the board using the indicated Method: 1, 2, or 3. Solution is printed based on the board as well as the time taken to execute. Method 1 was done as described in the assignment, 11 total threads.



```
osc@osc-VirtualBox:~/Documents/OperatingSystems/project2$ ./main 3
BOARD STATE IN input.txt
8 2 7 1 5 4 3 9 6
9 6 5 3 2 7 1 4 8
3 4 1 6 8 9 7 5 2
5 9 3 4 6 8 2 7 1
4 7 2 5 1 3 6 8 9
6 1 8 9 7 2 4 3 5
7 8 6 2 3 5 9 1 4
1 5 4 7 9 6 8 2 3
2 3 9 8 4 1 5 6 7
SOLUTION: YES (0.002134 seconds)
```

Figure 1: Output for Method 3 that matches the required output

Implemented

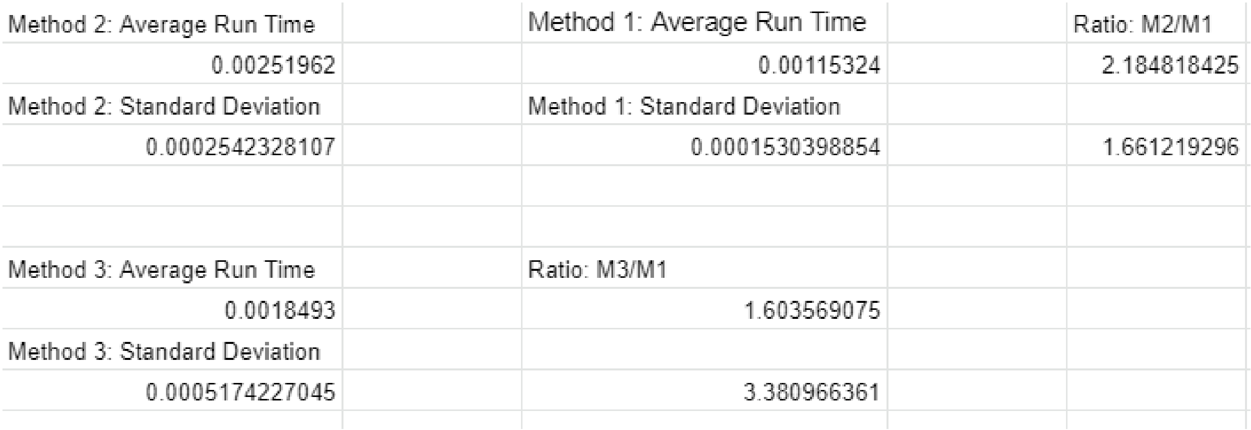
•*[5pts] Implement another way to set up threads as described in the attached description. That is, rather than creating one thread that checks all nine columns, you could create nine separate threads and have each of them check one column.To demonstrate this, your program should use the option “2” instead of “1”.*

A second method was implemented and denoted with the command line argument ‘2’. This method checked the rows and columns individually as well as the subgrids resulting in a total of 27 threads.

Implemented

•*[20 pts] Devise a statistical experiment that compares the two methods above.Your null hypothesis is “There is no statistically significant difference between two methods.*

To test Method 1 against Method 2, 50 runs were conducted for each method and the runtime recorded. The results of analyzing the results are shown in Figures 2 & 3.



Method 2: Average Run Time		Method 1: Average Run Time		Ratio: M2/M1
0.00251962		0.00115324		2.184818425
Method 2: Standard Deviation		Method 1: Standard Deviation		
0.0002542328107		0.0001530398854		1.661219296
Method 3: Average Run Time		Ratio: M3/M1		
0.0018493		1.603569075		
Method 3: Standard Deviation				
0.0005174227045		3.380966361		

Figure 2: Shows the runtime, standard deviation, and ratio for all comparisons

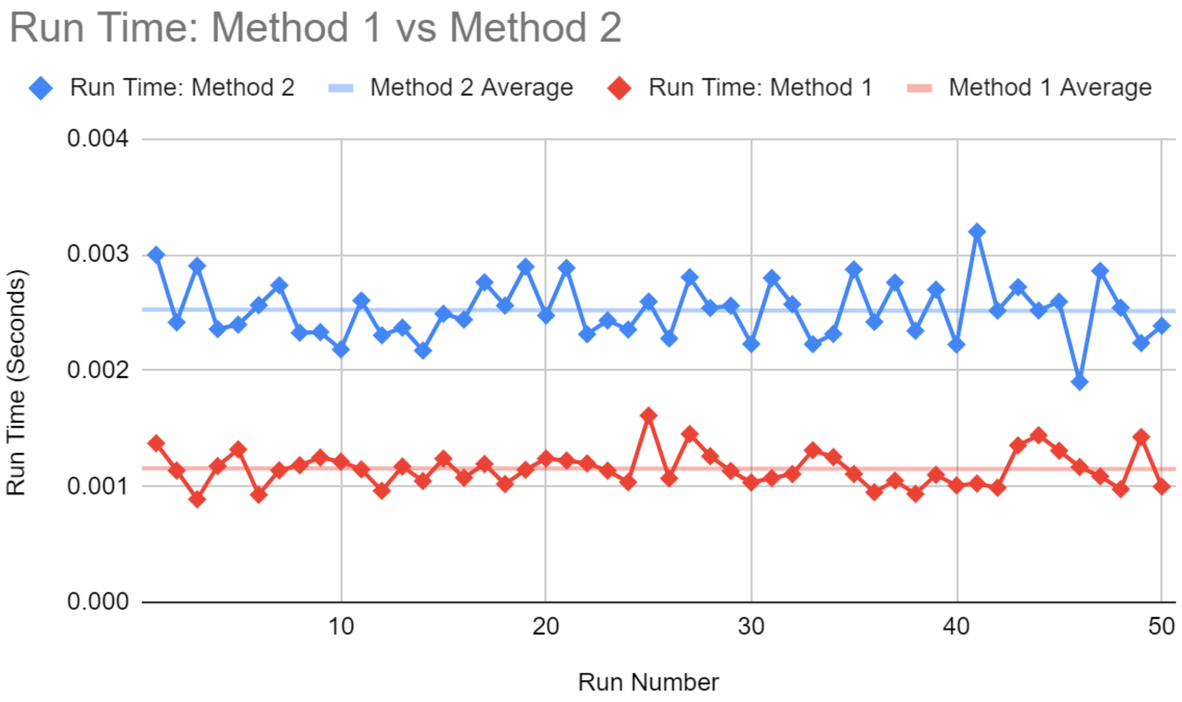


Figure 3: Plot of runtime for Method 1 versus Method 2

Figure 2 shows how the average runtime for Method 2 was 2.18 times that of Method 1. This was against the null hypothesis but what we expected as there were 2.45 times more threads to run in Method 2. Also interesting was that the standard deviation for Method 1 was considerably smaller than Method 2, this indicates that Method 1 was more consistent. This is also likely due to the larger number of threads

Figure 3 is the plot of Method 1 and 2. It visually shows how Method 1 ran faster and was more consistent than Method 2.

These clear statistical differences cause us to reject the null hypothesis.

Implemented

•*[25pts]Implement yet another, multi-process way to validate the board.When the option “3”is used, your program should create 11 child processes and the processes will behave in a similar manner as the option “1”except that processes are used instead of threads. Repeat the statistical experiments again comparing options “1”and “3.”Your null hypothesis is again “There is no statistically significant difference between two methods.”*

This method, Method “3” was implemented using 11 child processes. The first checks all columns, the second checks all rows, and the last 9 check each individual 3x3 subgrid of the Sudoku puzzle. Once each child has completed, the parent checks to see if each child passed its test successfully. After extensive testing, including giving incorrect values to rows, columns, and switching rows/columns to make the subgrids invalid, we determined that we had successfully implemented this requirement.

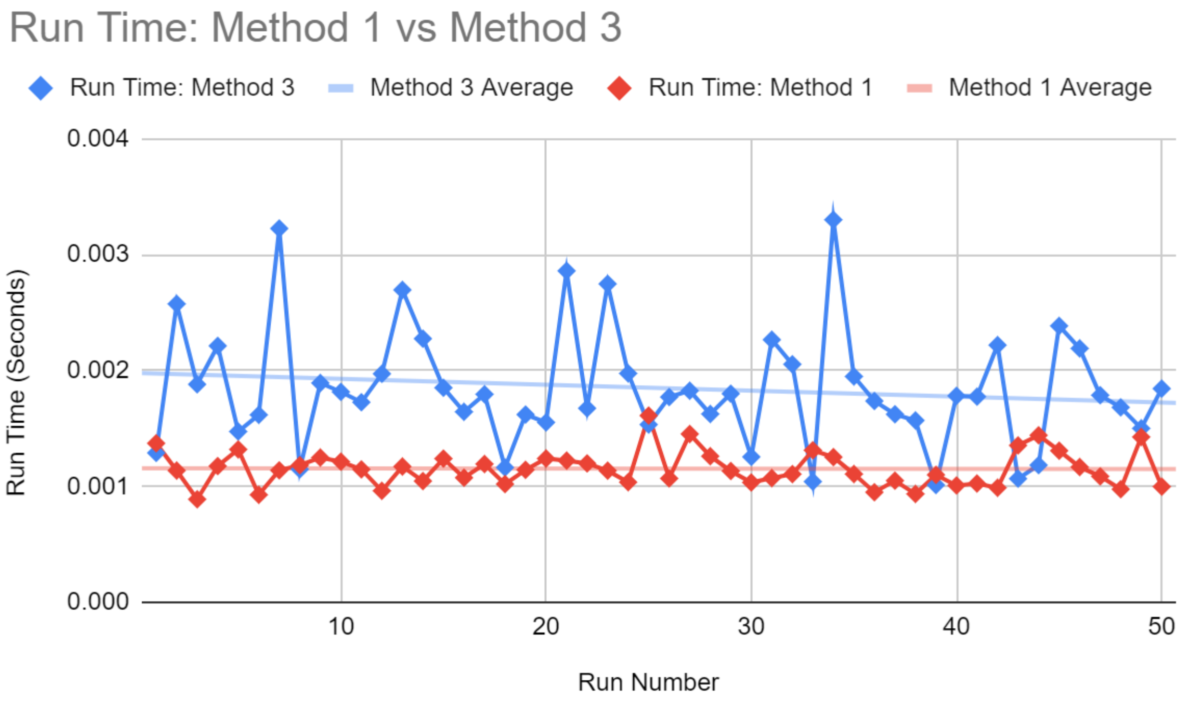


Figure 4: Plot of runtime for Method 3 vs Method 1

Figure 4 shows the graph of Method 3 vs Method 1 runtimes while Figure 2 shows the calculated average and standard deviation for Method 3. As one can see, Method 3 took 1.6 times as long as Method 1 and had a standard deviation over 3 times higher than Method 1.

From this data, it is clear that there is a statistically significant difference between the two methods where method 1 is significantly faster than method 2, so we can reject the null hypothesis.