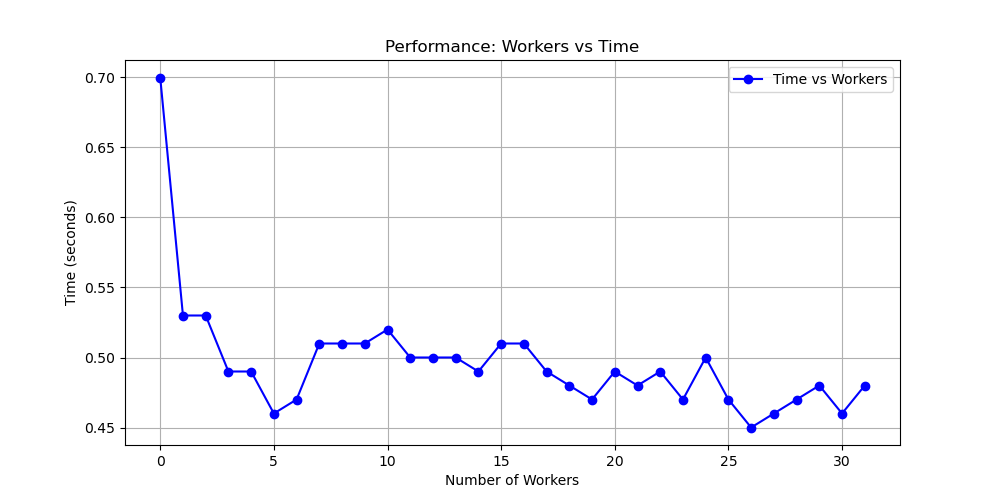
Question1**

The plot shows that the task distributing to 4 workers( the rest one is rank 0 ) has better preformance. This is because the time spent on managing (sending and receiving data) is equal to or greater than the time it takes for one worker to complete a task.

Question2.

We sort the profiler data by “tottime”, then we see that almost all functions were executed instantly (<=0.001 s) except for the trapezoid and my func and math.cos.

ncalls tottime percall cumtime percall filename:lineno(function)

1 11.144 11.144 25.368 25.368 /Users/hezhipeng/Desktop/venv/p5/week5/assignment1.py:9(trapezoid)

100000001 10.029 0.000 14.224 0.000 /Users/hezhipeng/Desktop/venv/p5/week5/assignment1.py:49(func)

100000001 4.194 0.000 4.194 0.000 {built-in method math.cos}

3 0.001 0.000 0.001 0.000 {built-in method \_imp.create\_dynamic}

6 0.001 0.000 0.001 0.000 {built-in method marshal.loads}

Actually, my func can be replaced by math. cos. If I Inline func directly in trapezoid to avoid the overhead of calling func repeatedly. It will improve the performance. It costs 11.14s for the implementation of the trapezoid function.Becuase it splits the distance of a to b into 100000000.

The math cos function runs 100000001 times and it costs 4.194. That is fine. So for this case, we can improve the performance by reducing the N, because the precision does not improve by increasing the N after N is already big, but it costs a lot of computation resources.