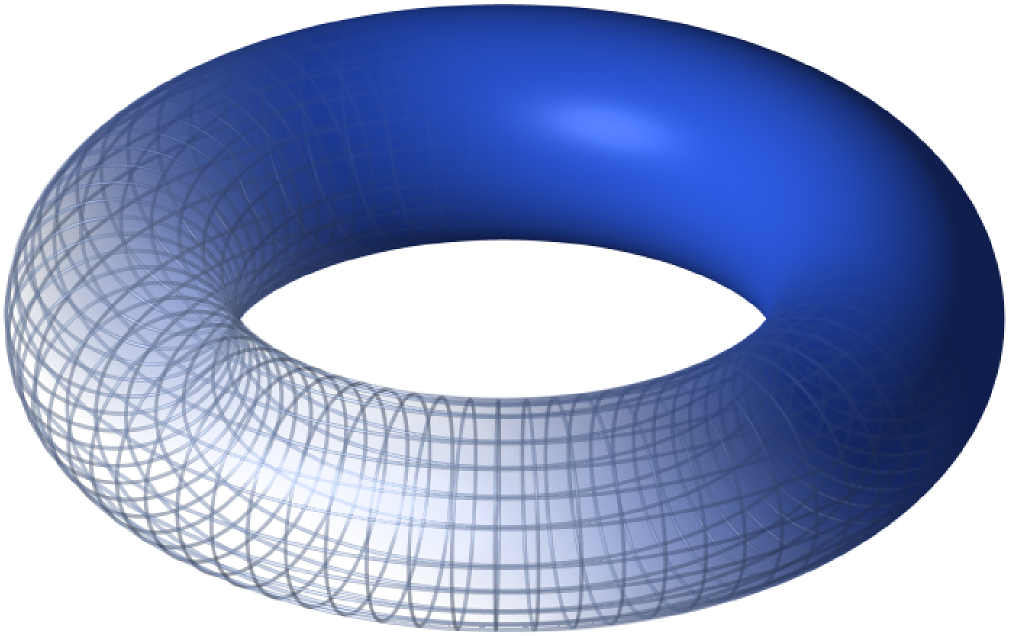


Wa-Tor Speedup Report

Joshua Wilkinson, 01/12/2023.

Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Purpose

This document is a report for the Wa-tor predator-prey simulation implemented using OpenMp, to see if there is a speedup in the parallelised code for the function of moving around sharks and fishes.

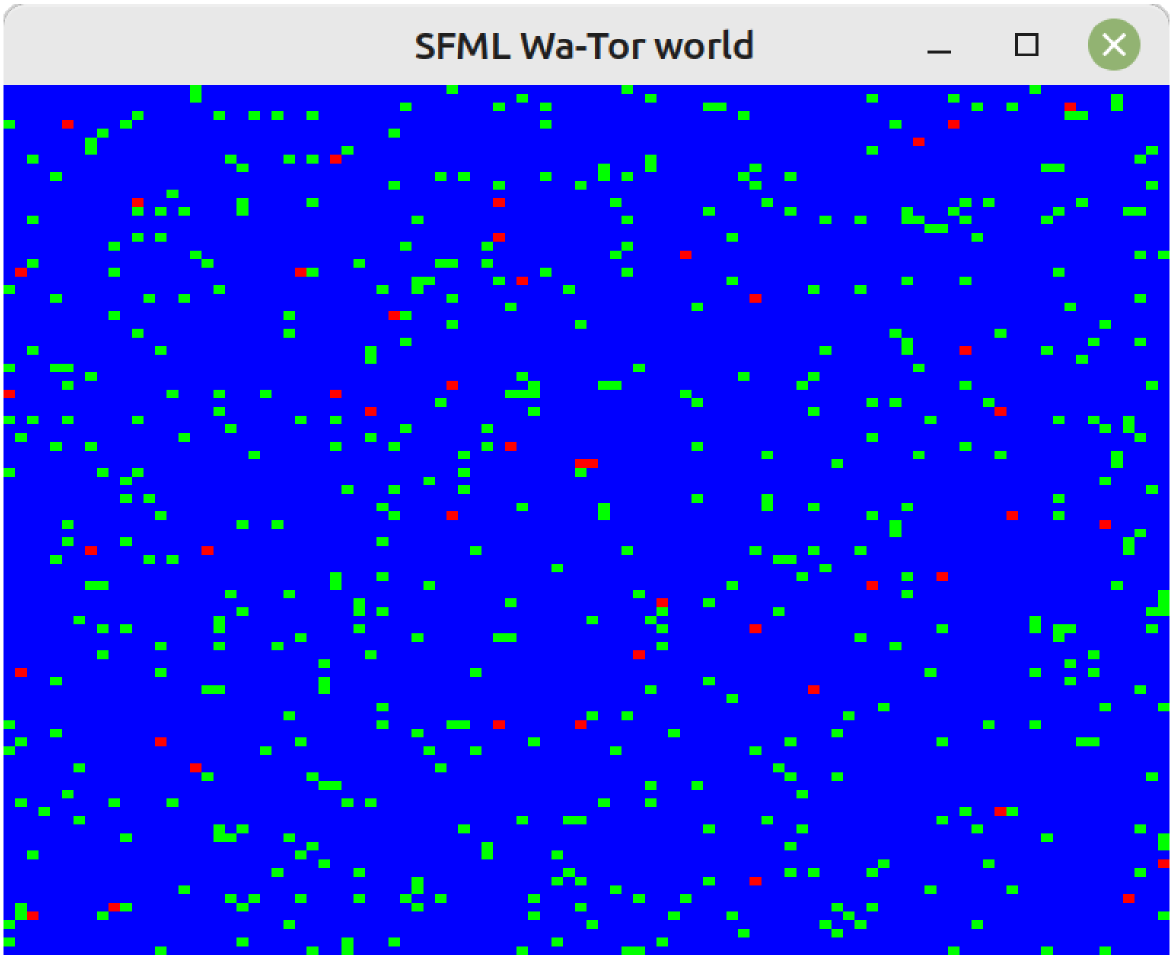


Fig 1: Screenshot of Wa-tor world in execution, **Source**: Author's own

PC Specs

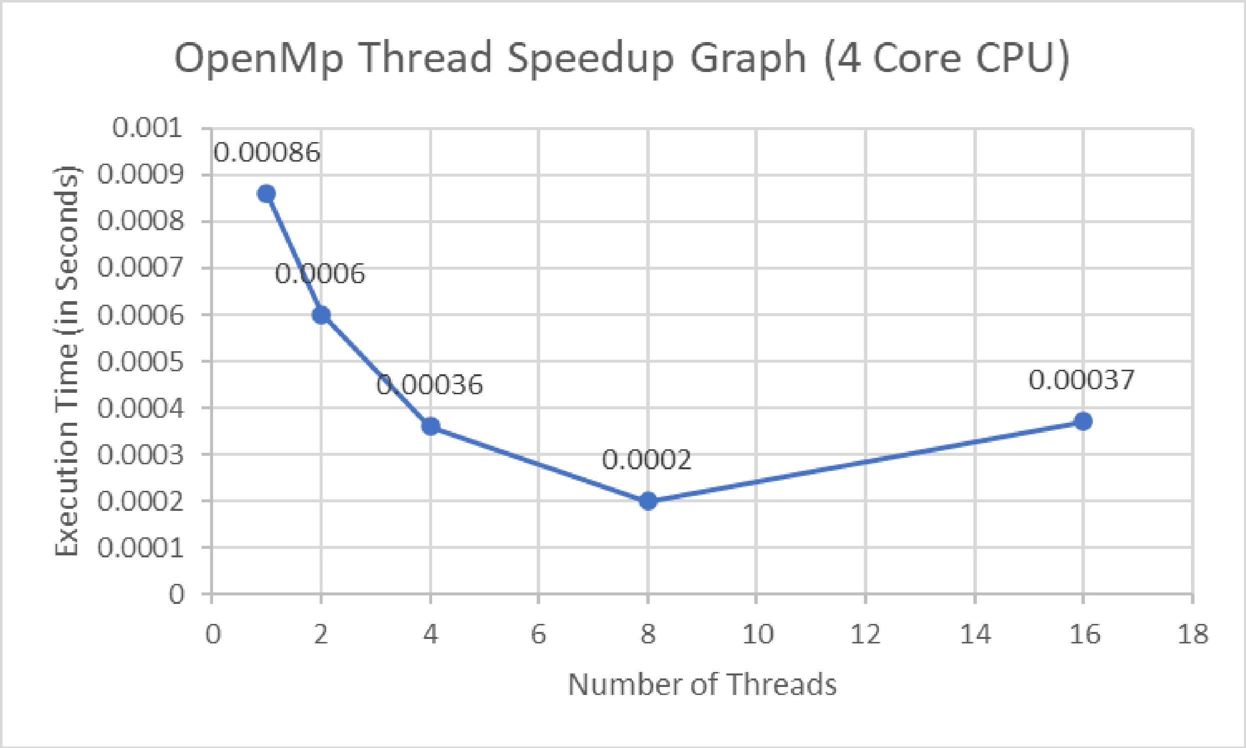
CPU: Intel i5 4 cores (8 thread)
RAM: 4GB

Results

For the results, the thread numbers 1, 2, 4, 8, and 16 were tested three times each to get the average/mean value. The OpenMp function '*omp_get_wtime()*' was used to time this.

Number of Threads	Execution Time Avg. (in Seconds)
1	0.00086
2	0.0006
4	0.00036
8	0.0002
16	0.00037

Here is a graph containing the results:



As we can see, there is a noticeable speedup when increasing the number of threads all the way up to 8 threads, but when we get to 16 threads the speed decreases. The reason I think that happens is because a single CPU core can efficiently handle up to 2 threads per core. For example, if a CPU is dual core (i.e., 2 cores) it will have 4 threads. And if a CPU has 8 cores it will have 16 threads and vice-versa. My CPU has 4 cores (8 threads), therefore it is reasonable to believe that is the reason for the slowdown.