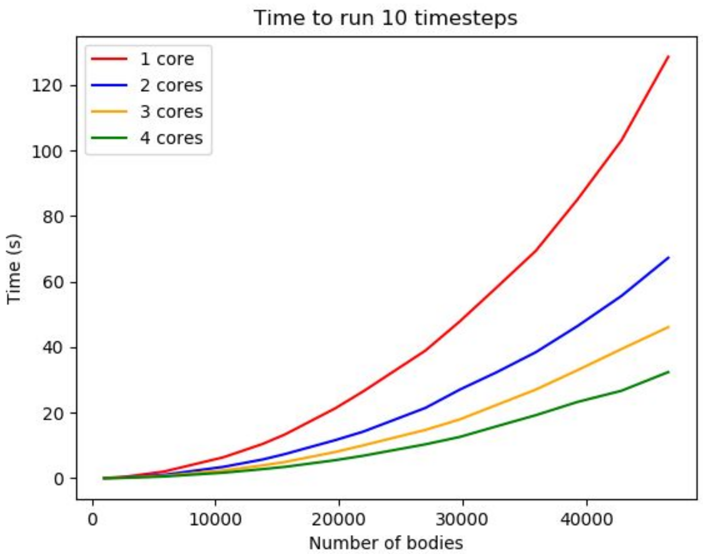
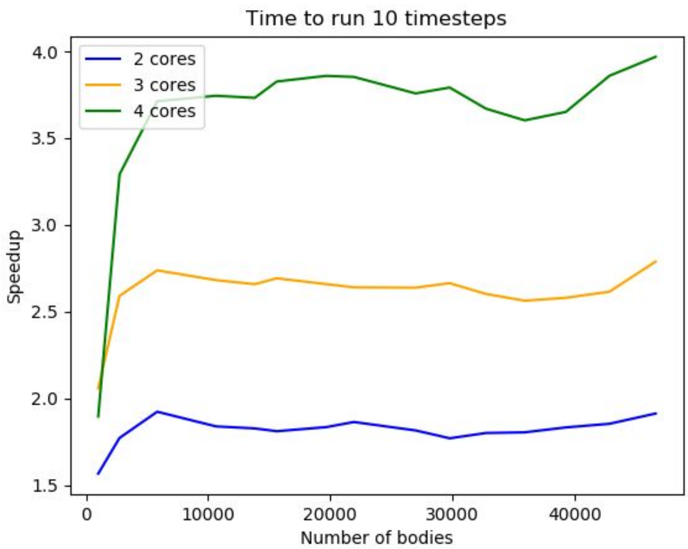
Setup

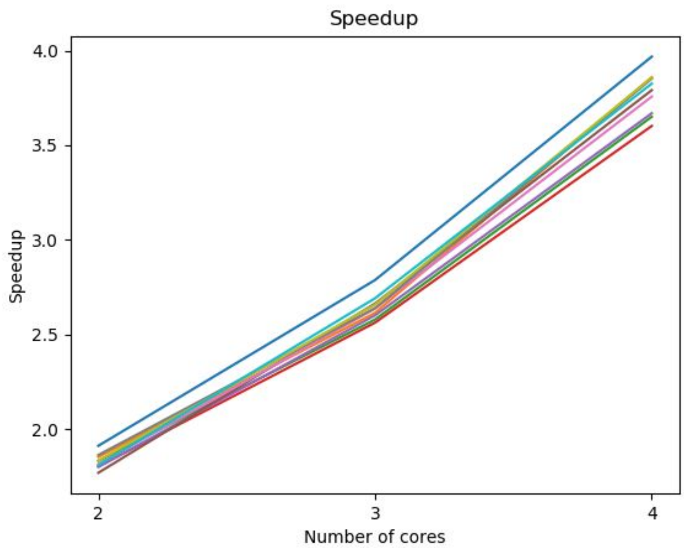
These experiments were run on a laptop with a 4-core Intel Core i7-6700HQ @2.60GHz and 16GB RAM. To study the scalability of my code I turned off file outputs and recorded the time taken to run 10 timesteps on increasing numbers of objects from 1000 to 46656 across 1, 2, 3 and 4 cores. This was achieved by setting the OMP_NUM_THREADS environment variable when running the code. The reason I have chosen not to run solution-step1.c as the baseline because it contains optimisations which only work when run on a single thread, so to get comparable results for scaling I have simply limited the number of threads for solution-step4.c to 1 to act as the baseline.

Results





On the top left plot we can see that the time taken to run each timestep grows polynomially, which is expected for this simulation. The plot on the top right shows the speedup relative to the single-core running time for 2, 3 and 4 cores. Once the number of objects goes past a certain threshold, the speedup remains largely static at around 2, 3 and 4 respectively. Due to the limited number of cores and the fact that both strong and weak scaling models have similar behaviour when the value of f is so small (fraction of the code which runs serially), it is hard to tell numerically whether a strong or weak scaling model is a better fit. However, from the definitions of each scaling model, strong scaling would be more appropriate in the case where the number of objects is fixed and we increase the number of cores, as shown in the plot on the right. In the case where we increase both the number of objects and the number of cores, weak scaling better describes the speedup.



If we apply the strong scaling model, the speedup is given by $S(p) = \frac{p}{f(p-1)+1}$. Fitting this to the data for the largest number of objects, 46656, using a least-squares method, we get an optimum value for f of 0.0103. This means that only 1% of the code runs serially, meaning it is highly scalable, and the theoretical lower limit for the running time is $0.0103 \cdot t(1) = 1.33s$.

If we apply the weak scaling law, $S(p) = f + (1-f)p$, and fit it to the same data, we get a value for f of 0.0433, also very low.