Space Bodies Assignment

cmkv68

February 19, 2018

Numerical Experiments

Consistency & Convergence

The table below describes the information related to the two bodies used for the collision experiment.

Body #	$s(x)$	$s(y)$	$s(z)$	$v(x)$	$v(y)$	$v(z)$	Mass
1	+0.1	+0.1	+0.1	-2.0	-2.0	-2.0	$1e^{-11}$
2	-1.0	-1.0	-1.0	+2.0	+2.0	+2.0	$1e^{-11}$

The bodies collide at $(x, y, z) = (0.155, 0.155, 0.155)$ at time $t = 0.275$. The error is calculated through calculating the difference between the position of one body and the other body upon collision. The following table shows the timestep used to calculate the collision, and the error value left over. We calculate the error of only one dimension; x, as the other dimensions (y, z) would follow the same values.

Timestep h	Error $\overline{u}_h$	x_a	x_b	Ratio	Steps	Range
Adaptive	0.000001997990	-0.449998	-0.450002	0.500503	1716457	0.000003994500
$10^{-6}/2^1$	0.000001999990	-0.449998	-0.450002	0.500003	275000	0.000003998500
$10^{-6}/2^2$	0.000001000020	-0.449999	-0.450001	0.499992	550000	0.000001998520
$10^{-6}/2^3$	0.000000499961	-0.45	-0.45	0.500039	1100000	0.000000998475
$10^{-6}/2^4$	0.000000250068	-0.45	-0.45	0.499865	2200000	0.000000498564
$10^{-6}/2^5$	0.000000125078	-0.45	-0.45	0.499688	4400000	0.000000248608
$10^{-6}/2^6$	0.000000062134	-0.45	-0.45	0.502946	8800000	0.000000123183
$10^{-6}/2^7$	0.000000031729	-0.45	-0.45	0.492452	17600000	0.000000061368
$10^{-6}/2^8$	0.000000014415	-0.45	-0.45	0.541966	35200000	0.000000028684
$10^{-6}/2^9$	0.000000006426	-0.45	-0.45	0.607915	70400000	0.000000012311
$10^{-6}/2^{10}$	0.000000002519	-0.45	-0.45	0.775391	140800000	0.000000004178

(0.000000014415 - 0.000000006426) / (0.000000006426 - 0.000000002519)

The convergence order is 0.7152957794....

The adaptive timestep works suitably and utilises less timesteps than $ts = 5e^-9$ by a significant margin, whilst having relatively comporable error residues.

Complexity

Under the assumption that the timestep and time limit is fixed, the most dominant function `updateBodies()` which utilises a nested loop that iterates through the number of bodies initiated. For each iteration, a force for a given

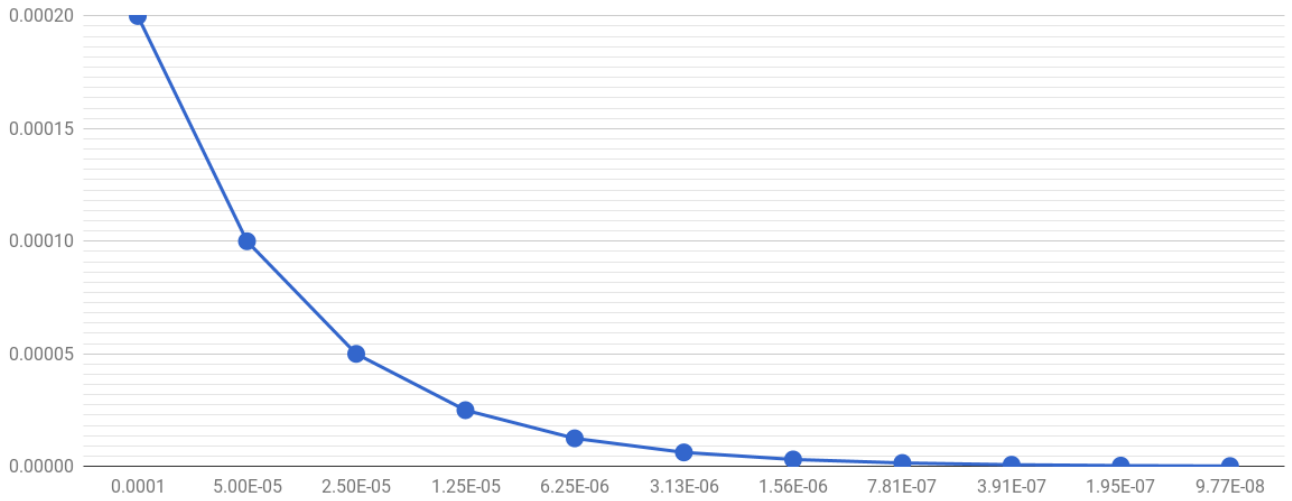

Figure 1: A chart showing the the timestep used against the error produced. The chart shows a clear convergence.

body is calculated by comparing its position against every other body in space. This results in `updateBodies()` to run in $O(n^2)$.

Procedures have been taken to reduce the constant; Each body only calculates its force against bodies that precede them in the order of initiation i.e. Body 2 calculates force from Body 1 and Body 0 whereas Body 3 calculates from $0, 1$ and 2.

Whilst `updateBodies()` would continue to run in $O(n^2)$, the hidden constant would be drastically reduced to a factor of $\frac{1}{2}$ of the original number of calculations needed.

Statistics

Scaling Experiments

The machine used consists of a `Intel i7 3770k` processor at a 3.7Ghz clock speed, powering 4 cores and 8 threads. It utilises 32GB of memory and the storage consists of a SSD hooked up via SATA3. It is using a fresh installation of Ubuntu and has no other major processes running.

Questions

1. How does the scalability for very brief simulation runs depend on the total particle count?

There is a lot of overhead involved in initiating a loop for a set of parallel processors, to the extent that *may take more time than the actual simulation itself.* This may include each processor initiating their own set of variables.

2. Calibrate Gustafson's law to your setup and discuss the outcome. Take your considerations on the algorithm complexity into account.

Gusafson's Law: Gustafson estimated the speedup S gained by using N processors (instead of just one) for a task with a serial fractions (which does not benefit from parallelism) as follows: S=N+(1-N)s

3. How does the parallel efficiency change over time if you study a long-running simulation?

There

Distributed Memory Simulation