Report on the optimization loops: avoid avoidable

Nicole Orzan

December 17, 2017

In this report you can find a description about the steps done to optimize the code provided avoid_avoidable.c This exercise was ran and measured on my laptop, which is a Lenovo with Intel CPU: Intel(R) Core(TM) i7-6500U CPU @ 2.50GHzi7. For each step of the optimization it was analyzed the time spent by the run of the program to be finished. For the analysis to be complete, for each optimization step was calculated the average execution time of the program, computed on a sample of 12 runs.
Then was used jupyter-notebook to create the a graph.

The optimization of the code was divided into 8 different steps, of which the first 6 will be examined. The examined steps include rewriting some parts of the code and clean some others. In practice some changes made are:

- The remotion of the roots calculations, substituting it with the use of squared powers, everywhere possible;
- The separation of the 3D calculus of the distances with the separate use of the coordinates x,y and z;
- The replacement of dx^2 with $dx * dx$, to avoid the use of powers;
- The replacement of repeated calculations of inverse numbers with new fixed variables;
- The insertion of some "register" variables, for the most often called ones.

I decided to use also some compiling options, such as O0, O1, O2 and O3, to take again the measurements and to compute the mean times, to examine the possible differences. With the -O option, the compiler tries to reduce code size and execution time, and turns on a lot of optimization flags.
You can find the table containing the mean calculated times in figure 1. I provide also a graph to see the trend of the different compiler optimizations for the different optimizations steps; you can see it in figure 1.

Optimiz Step	Hand optim	O0	O1	O2	O3
0	3.972505	4.09	1.49	1.07	1.06
1	3.485531	3.57	0.55	1.08	1.08
2	2.951075	3.01	0.55	1.08	1.09
3	2.113705	2.14	0.50	0.81	0.79
4	1.012711	1.04	0.49	0.80	0.80
5	1.172665	1.20	0.49	0.81	0.81
6	0.955033	0.96	0.50	0.82	0.87

Table 1: Mean computing times for each optimization step, without compiler options and with compiler options O0, O1, O2 and O3 on.

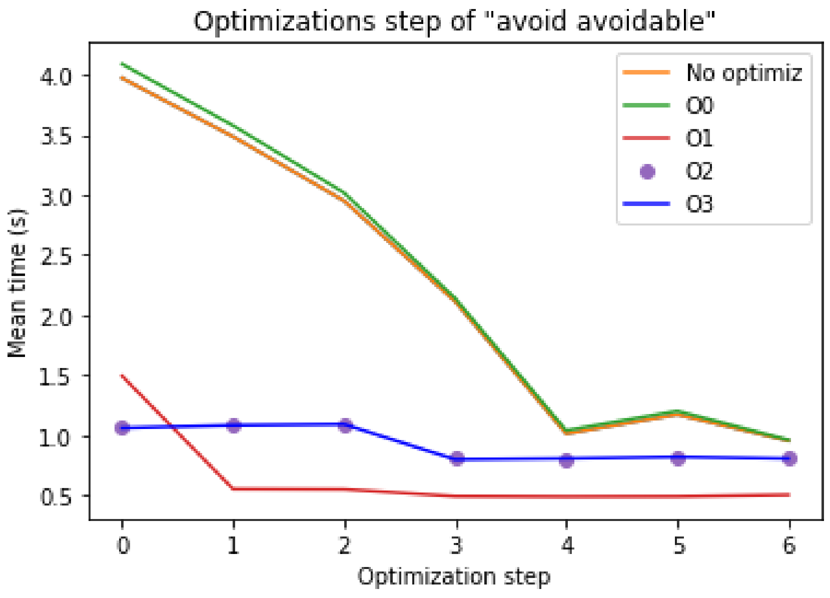

Figure 1: Mean computing time for each optimization step, without and with compiler optimizations O0, O1, O2 and O3 on.