Evolutionary Algorithms for Mechanical Structures

Tobias Jacob
Raffaele Galliera
Ali Muddasar

November 22, 2020

Project Report

During this first week of implementation, we worked on setting up the C++ project, implemented a framework of the project and a first running iteration.

- Setup of the project, Catch2 testing framework, GitHub, Makefile, Trello and the general code structure; [All]
- Implemented the creation of the FEM equation system, the CG solving method, the sparse matrix implementation and performance evaluation; [Tobias Jacob]
- First iteration of the evolutionary optimizer, implementing general concept of the Genetic approach and reproduction system; [Raffaele Galliera]
- Implemented a Floodfilling algorithm, in order to make sure that all the planes generated stay connected together; [Muddasar Ali]
- We planned how we want to parallelize the equation solver and distribute the evolution and which parts of it will have a focus on. [All]

Speedups

We want to solve an equation system of at least $n^2 \approx 80799$ equations. Possibly, even more. Solving time of an equation system grows in general as a cubic of equation size, so in our case n^6. We implemented the **CG method**, which reduces this time down to $O(n^4 i)$, with i being the required iterations, and n^4 the time for the vector matrix multiplication with n^2 rows / columns. A big advantage is, that we have sparse matrices with typically 8 values per row. By extending our matrix implementation to a **sparse matrix implementation**, we improved the execution speed to $O(n^2 i)$, which had the biggest impact on performance.

Using `-Ofast` instead of `-O3` improved the speed by roughly 25.4%. Using aligned memory did not improve the performance. Using a vector instead of a linked list for the matrix storage did not have a significant impact.

Profiling of the Simulation

Simulating the mechanical structures is taking up the most time. For profiling, we created a $n \times n$ grid, added supports on the bottom, and a force on the top. Then we tried to calculate the stress on the structure. We measured the time for different parts of the solver.

- A $n \times n$ grid has $2(n+1)^2 - 3$ degrees of freedom. The matrix we solve therefore has $O(n^2)$ equations.
- The estimated steps of the solution seems to be proportional to the maximum length between two points, so $i = O(n)$. An exact calculation is difficult because it is dependent on the condition of the matrix.
- For each degree of freedom the connections to the four neighbours have to be added to the equation. The equation setup time is therefore $O(n^2)$.
- The solving time is $O(n^2ci)$, with n^2 being the number of equations, $c = 8$ being the connections to different neighbour planes and i being the number of required iterations. Since c is constant, and $i = O(n)$, the total solving time is $O(n^3)$.

Currently, we are able to solve the equation for a 200×200 grid in roughly 22 seconds.

n	Equations	Steps for solution	Equation setup time (μs)	Solving time (μs)
-	$O(n^2)$	$O(n)$	$O(n^2)$	$O(n^3)$
200	80799	6555	98695	22344100
100	23187	2146	23187	1726910
50	5199	1039	6203	154818
25	1349	449	1626	17768
12	335	210	361	1786

Table 1: Execution time for FEM solver