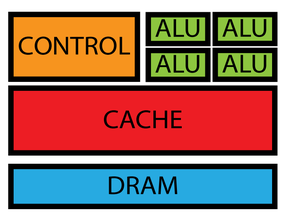
GPU-Based Topology Optimization on Unstructured Meshes

Objective

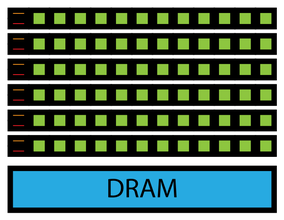
Investigates the feasibility of finite element methods and topology optimization for unstructured meshes in massively parallel computer architectures, more specically on GPUs.

Graphics Processing Unit (GPU)

The GPU is a many-core processor with a smaller and fast set of instructions (more specialized type of hardware), but capable of handling many concurrent threads.
A thread is an independent unit of processing that can handle and process a task.



CPU Schematic



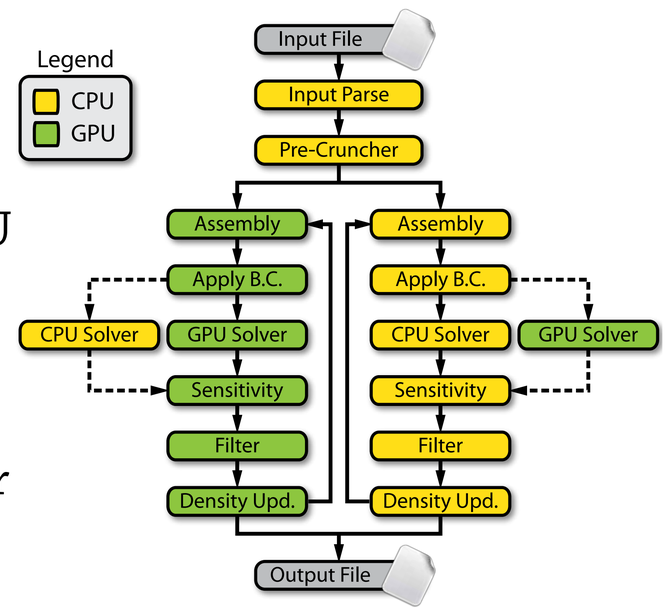
GPU Schematic

Why Unstructured Meshes?

Unstructured meshes remove the limitation on the user to define restrictions and loading further closing the gap between research code and real application.

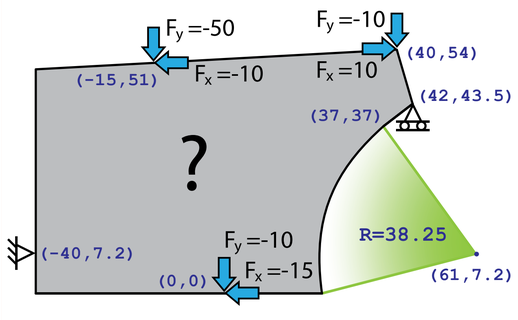
Topology Optimization Compute Chains

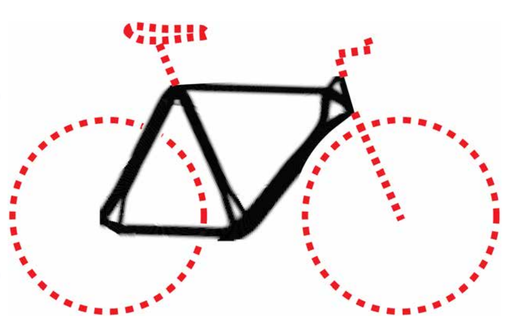
The code is organized in such a way that calculations can take place in the CPU, GPU or hybrids of both. Different machine precisions and operation order account for small diferences in the results.



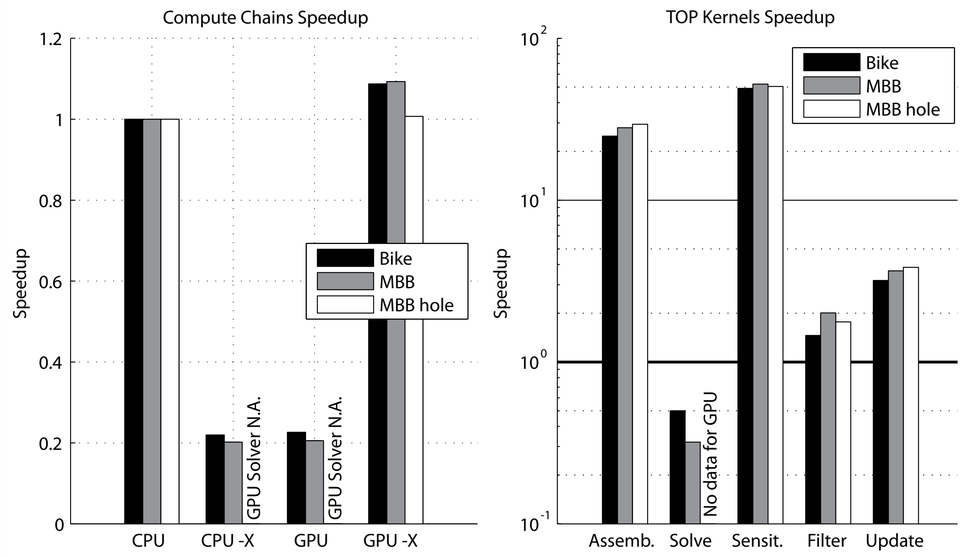
Example 1: Bike

Domain attempts to mimic common road bike restructions and plaussible loading scenario.
20378 Q4 elements and 20635 nodes.
30 Iterations compute time: 56 secs





Benchmark Results

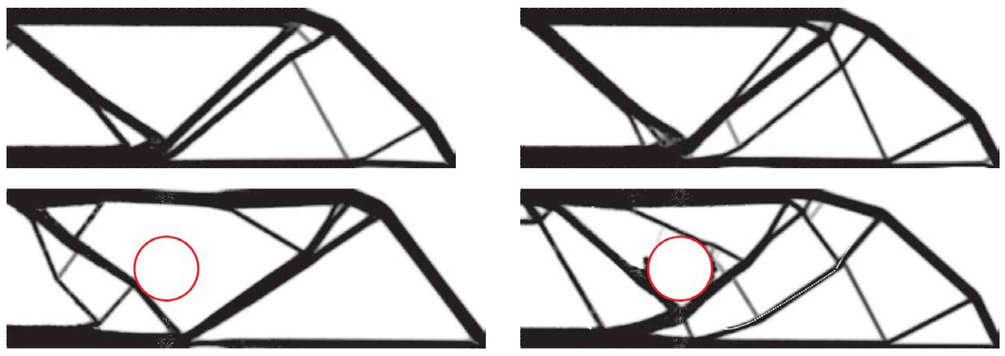


Runtime Speedup

Kernel Breakdown

Examples 2 & 3: MBB beam

The traditional Messerschmitt-Bölkow-Blohm (MBB) beam (43200 Q4 elements) and a variation with a circular hole (55200 Q4 elements) are tested, compared and benchmarked.
Results for 30 iterations:



CPU Results

GPU Results

GPU Solver: The Missing Link

The solver dominates the overall speedup taking approximately 90% of the runtime.
There is ongoing research in development of an efficient GPU solver with promising results.

Tomás Zegard , Glaucio H. Paulino