# Virtual Simulation Lab: Using Modern Distributed Cloud Clusters with Terabytes of RAM

To efficiently simulate the scattering from complex surface topologies up to 5 kHz, requires tens of thousands of boundary elements and terabytes of RAM. This is only possible using parallel processing in modern distributed cloud clusters. Dask is a flexible, open-source Python library for parallel computing. Dask scales Python code from multi-core local machines to large, distributed clusters in the cloud. Dask consists of three main components: a Client, a Scheduler, and one of more Workers.

In Figure 1, we show a schematic illustration of the Dask process used by a distributed cloud cluster consisting of hundreds of CPUs and terabytes of RAM. The Developer or user provides code to Dask's Client manager who analyzes the code and sends instructions to a Scheduler who coordinates the Worker's activity. The Workers are threads, processes, or separate machines in the cluster, who execute the computations. The three componenets communicate using messages about metrics, computations, or actual data. The results are sent back to the platform Client. All done via your laptop!

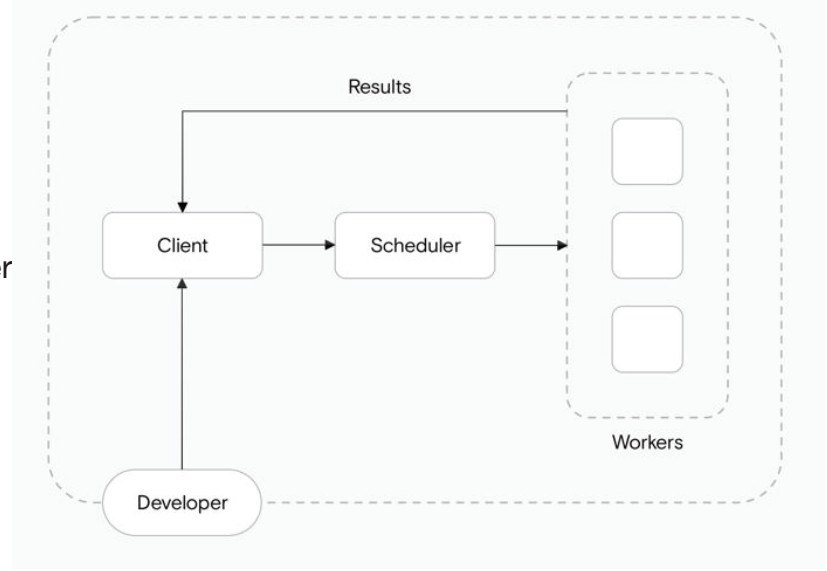


*Figure 1. Process used in a distributed cloud cluster.*

We seek to compare the run times achieved with VIRGO against the open-source package Bempp-cl. To do so, a performance comparison is drawn between a cloud cluster computer and a local personal computer. Figure 2 shows the device specifications for both computation devices.

| Device | CPU | Num CPUs | RAM |
|---|---|---|---|
| Cloud Cluster | Intel® Xeon® E7 8880 v3 | 128 | 4 TB |
| Personal Computer | Intel R©<br>Core(TM) i7-8750H | 6 | 32 GB |

*Figure 2. Comparison between personal computer and cloud cluster specifications.*

The data shown in Figure 3 correspond to a single-frequency, single-source benchmark on the run time using the different packages. As the mesh size increases, it becomes apparent that the cloud cluster approach is orders of magnitude faster than any local implementation. One can note that VIRGO on the local machine is still faster than Bempp-cl. Nonetheless, at the mesh size identified by the black star symbol, a shift from the general trend is noticed. This is due to the lack of RAM on the local machine, making the computation slower that the Bempp-cl implementation. This behavior is expected since VIRGO was designed to be used with cloud clusters.

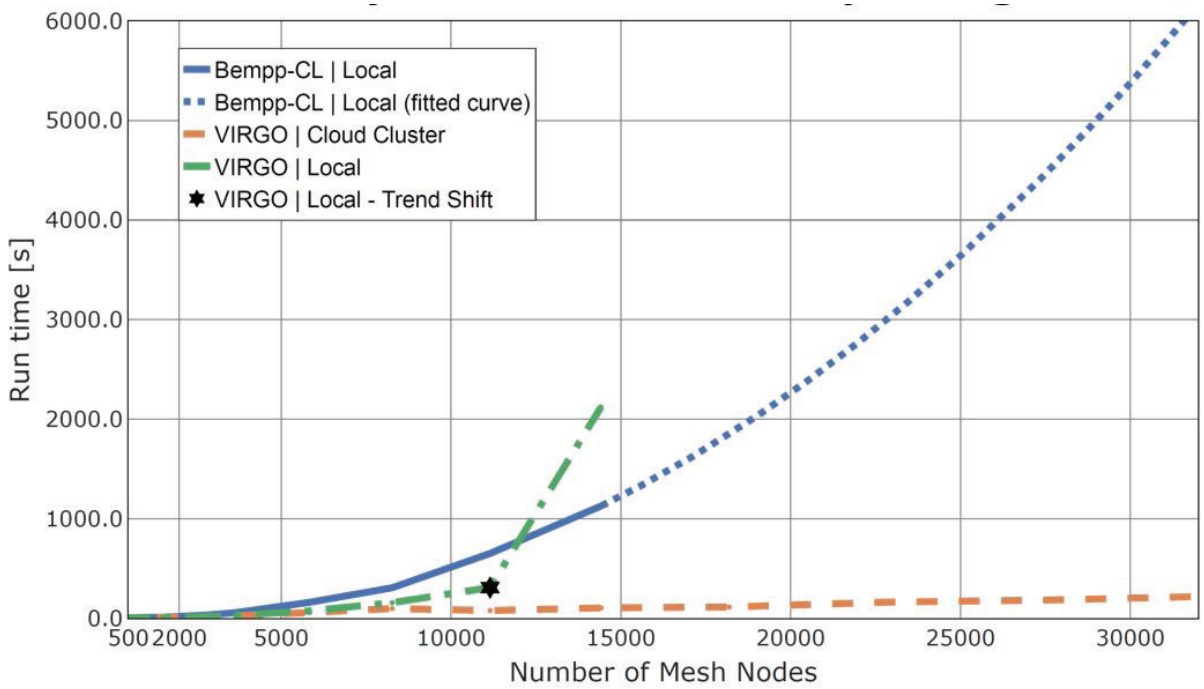


*Figure 3. Parallel python cloud computing run times.*

Even now, for larger samples or periodic/modulated arrays, the limitation of computational resources is still an obstacle to fast computing. To address this, we will explore new ways of distributing the problem to allow larger meshes (>50,000 nodes) to be computed efficiently.

In the next post, we will present simulations of RPG products with complex topology using VIRGO.

**Dr. Peter D'Antonio**
Director of Research
**Acoustical Research Center**