‘votess: A fast, multi-target voronoi tessellator using the SYCL framework’

Samridh Dev Singh [1], **Chris Byrohl** [2], and **Dylan Nelson** [2]

1 Grinnell College, 1115 8th Avenue, 50112 Grinnell, United States of America **2** Heidelberg University, Institute for Theoretical Astronomy, Albert-Ueberle-Str. 2, 69120 Heidelberg, Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Statement of need

Voronoi tessellation is a fundamental spatial decomposition technique enabling the partitioning of space into regions based on proximity to a discrete set of points, and is widely used in scientific domains such as astrophysics (Springel, 2010), earth sciences [], materials science [], and biochemistry []. Although performing Voronoi tessellations on large datasets has been feasible for some time, the increasing complexity and size of modern data have underscored the need for faster, more efficient computation. With the rise of powerful many-core CPU and GPU architectures, the computational power available today has greatly improved the ability to handle these large datasets more effectively.

However, despite these advancements, most existing implementations of Voronoi tessellations are tailored to specific processor architectures and platforms, limiting their portability. This results in the need for bespoke solutions for different hardware setups, creating inefficiencies and increasing development time. Additionally, many classic insertion algorithms do not fully leverage the performance potential of multi-core systems, further highlighting the need of a modern solution.

To address this problem, `votess` provides a portable solution that can operate across various accelerator architectures, without modification to the source code, enabling developers within various scientific fields to be able to make use of the new computing architectures.

To address this problem, votess leverages a SYCL single-source framework abstraction to provide a portable solution that runs efficiently across multiple accelerator architectures, including CPUs and GPUs, without requiring modifications to the source code.

Summary

`votess` is a library for performing parallel 3D Voronoi tessellations on heterogeneous platforms via the SYCL framework. `votess` was designed to be portable yet performant, with an easy-to-use interface.

The underlying algorithm is based on a paper (**?**), which highlights that many applications, such as in astrophysics [] or fluid simulations [], only require the geometry of the Voronoi cells and their neighboring information, rather than a full combinatorial mesh data structure. This observation allows for a simplified algorithm, as presented here, which avoids the need for classical mesh-based approaches like the Bowyer-Watson algorithm.

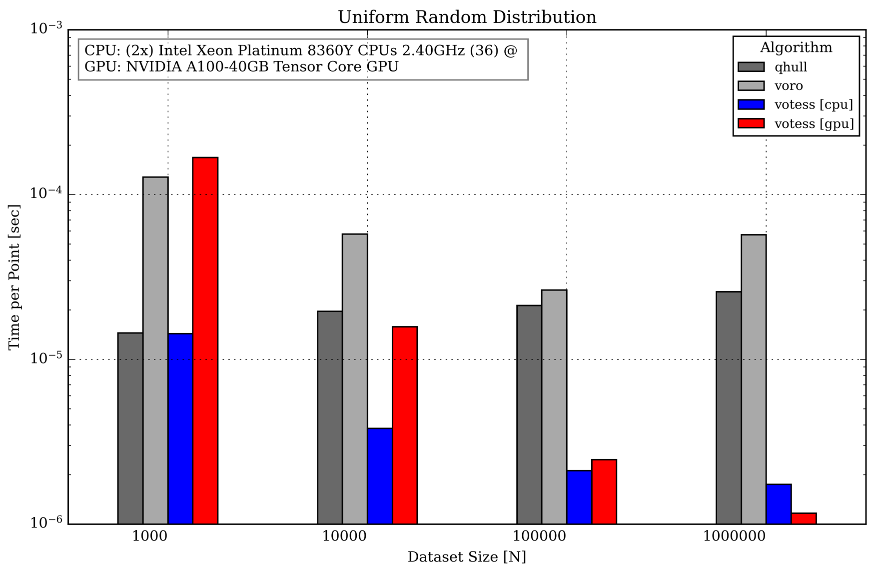
The core algorithm employed by `votess` consists of two main steps. First, the input set of points is sorted into a grid, and a k-nearest neighbors search is performed. Once the nearest neighbors are identified for each point, the Voronoi cell is computed by iteratively clipping a bounding box using the perpendicular bisectors between the point and its neighbors. To optimize the process and avoid iterating through all neighbors, a security radius condition is applied. If a Voronoi cell cannot be validated, a CPU fallback mechanism is used to ensures robustness.

This efficient algorithm allows for independent thread execution, making it highly suitable for GPU parallelism. Unlike previous algorithms that relied on sequential execution due to their mesh insertion methods [], `votess` leverages the independence of cell computations to achieve significant speedups in parallel environments.

Performance

With a working implementation of `votess`, it can be seen that it outperforms several single-threaded applications:

In Figure 1, we show the performance of `votess` compared to two other single-threaded Voronoi tessellation libraries: `QHULL` and `VORO++`. `QHULL` is a well-known computational geometry library that constructs convex hulls and Voronoi diagrams using an indirect projection method (Barber et al., 1996), while `Voro++` is a C++ library specifically designed for three-dimensional Voronoi tessellations, utilizing a cell-based computation approach that is well-suited for physical applications (Rycroft, 2009).



Here, it is clear `votess` performs best on large datasets. The CPU implementation outperforms other applications of atleast tenfold, and at most a hundred fold on large datasets. It must be noted, that the benchmarks were taken before either the CPU and GPU implementations have recieved optimizations.

There also exists other multithreaded Voronoi tesellelation codes, such as `ParVoro++` (Wu et al., 2023), `CGAL` (The CGAL Project, 2018), and `GEOGRAM` (Inria, 2018), and although they do not natively support GPU architectures, they are also widely used in large-scale computational geometry applications.

Features

`votess` provides a versatile and efficient tool for computing Voronoi tessellations, supporting multiple output formats including neighbor information for each Voronoi cell. It has been tested on various CPU and GPU architectures, delivering high performance on both platforms.

Users can leverage votess in three ways: through the C++ library, a command-line interface clvotess, and a Python interface pyvotess. The C++ library offers a simple interface with a primary function, tessellate, that computes the tessellation. Additionally, users can select the target device to run said tessellation. The Python wrapper, pyvotess, mirrors the functionality of the C++ version, providing the same ease of use for Python-based workflows.

To fine-tune the behavior of votess, the class vtargs is provided, allowing users to adjust parameters much like std::unordered_map from the STL. These parameters can be used to optimize runtime performance if needed. The tessellate function outputs a templated class dnn, representing a 2D jagged array of neighbors contributing to each particle's Voronoi cell of the sorted inpute dataset, as managed via vtargs.

Acknowledgements

CB and DN acknowledge funding from the Deutsche Forschungsgemeinschaft (DFG) through an Emmy Noether Research Group (grant number NE 2441/1-1).

Reference

Barber, C. B., Dobkin, D. P., & Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, *22*(4), 469–483. https://doi.org/10.1145/235815.235821

Inria, P. A.-G. (2018). *Geogram: A programming library of geometric algorithms*. http://alice.loria.fr/software/geogram/doc/html/index.html

Rycroft, C. (2009). *VORO++: A three-dimensional voronoi cell library in c++*.

Springel, V. (2010). Moving-mesh hydrodynamics with the AREPO code. *Proceedings of the International Astronomical Union*, *6*(S270), 203–206. https://doi.org/10.1017/S1743921311000378

The CGAL Project. (2018). *CGAL user and reference manual* (4.12.1 ed.). CGAL Editorial Board. https://doc.cgal.org/4.12.1/Manual/packages.html#PkgSpatialSortingSummary

Wu, G., Tian, H., Lu, G., & Wang, W. (2023). ParVoro++: A scalable parallel algorithm for constructing 3D voronoi tessellations based on kd-tree decomposition. *Parallel Computing*, *115*, 102995. https://doi.org/https://doi.org/10.1016/j.parco.2023.102995