Efficient and Rapid Grid Voronoi Diagram Computation

Mingeon Jeong, Minseop Lee, Dongwon Lee, Chaewoon Lee

August 14, 2023

// abstract //

1 Introduction

The Voronoi diagram is a fundamental geometric structure that has found diverse applications across fields ranging from computational geometry and computer graphics to geographic information systems and optimization. It serves as a powerful tool for spatial analysis, offering insights into proximity relationships, resource allocation, and nearest neighbor queries. As the scale and complexity of datasets continue to grow, the demand for efficient algorithms to compute Voronoi diagrams becomes increasingly crucial.[1]

This study addresses the challenge of computing Voronoi diagrams in the context of grid-based datasets. Grid Voronoi diagrams are particularly relevant in scenarios where spatial data is organized into a regular grid, such as image processing, terrain modeling, and geographic analysis.[2] However, as grid resolutions increase and the number of grid points escalates, traditional methods for Voronoi diagram computation exhibit limitations in terms of runtime efficiency and memory consumption.

The primary objective of this study is to propose an innovative approach for the efficient and rapid computation of grid Voronoi diagrams. By leveraging recent advancements in algorithm design and parallel processing techniques, we aim to develop a solution that not only scales gracefully with the size of the grid but also maintains a high level of accuracy and robustness. This research builds upon prior work in the field of computational geometry and contributes to the ongoing pursuit of practical solutions for real-world applications.

In this paper, we present the methodology, implementation details, and experimental evaluation of our proposed algorithm. We compare its performance against existing methods using a random test datasets, showcasing its advantages in terms of computation time, memory utilization, and scalability. The results obtained highlight the significant contributions of our approach in addressing the challenges posed by the computation of grid Voronoi diagrams.

The remainder of the paper is structured as follows: Section 2 provides an overview of related work in Voronoi diagram computation and existing techniques for grid-based datasets. Section 3 introduces the problem formulation and outlines the key components of our proposed algorithm. Section 4 elaborates on the technical implementation details, highlighting the novel strategies employed to achieve computational efficiency. Section 5 presents the experimental evaluation, offering quantitative analysis of our algorithm's performance. Finally, Section 6-8 discusses the implications of our findings, outlines future research directions, and concludes the study. References and Appendices can be found in the last section of this paper.

2 Related Work

2.1 Naive Approach

Naive approach of grid Voronoi Diagram is compute distance between grid points and cites.

Algorithm 1 Naive Voronoi Diagram Algorithm

```
for x = 1 to M do
    for y = 1 to M do                          ▷ For all grid point; O(M^2)
        d ← ∞                                  ▷ initialization
        ans[x][y] ← 0
        for i = 1 to N do                      ▷ Compute distance between all cites
            if d > dis((x_i, y_i), (x, y)) then
                d ← dis((x_i, y_i), (x, y))
                ans[x][y] ← i
            end if
        end for
    end for
end for
```

The time complexity is $O(NM^2)$.

2.2 Jump Flooding Algorithm [3]

Jump Flooding Algorithm is an algorithm used to obtain a Voronoi diagram or perform Distance transforms using a grid of M by M. The time complexity is $O(M^2 \log M)$ and the implementation of this algorithm is as follows:

1. Marking cells that include sites. We call these points to *seed*. Flooding is begin from these *seed*.
2. Start with $M/2$ and reduce the step size k by half, and mark the color according to the below steps.

   for each marked cell $P = (x, y)$, we consider ALL $Q = (x + i, y + j)$ where $i, j \in -k, 0, k$

   - if Q isn't marked, mark with the site of P
   - if Q is marked, mark with more closer site seed between *seed* of Q and *seed* of P
3. Repeating the above task $O(\log M)$ times results in a Voronoi diagram.

Figure 2.1 discribe this algorithm.

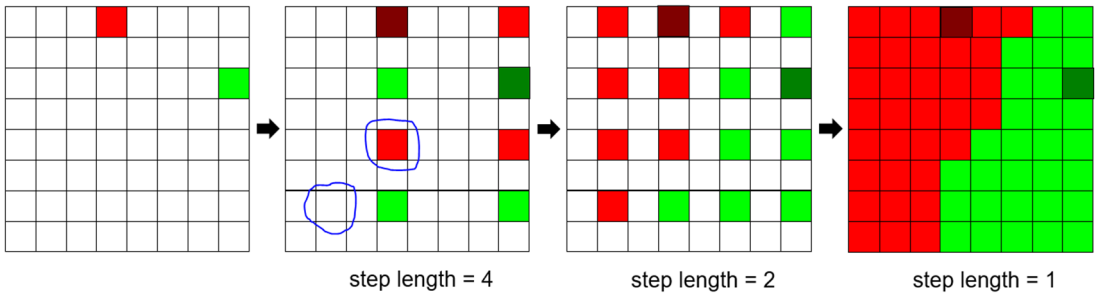

Figure 2.1

One of important feature of this algorihtm is that time complexity is not realted to N. In other word, if N is very bigger than M, this algorihtm is useful. However, if two or more cites is included in one cell, this algorihtm can't used to find Voronoi diagram. In that situation, we have to increase M.

Algorithm 2 Jump Flooding Algorithm

```
k ← M/2
for all (x, y), ans[x][y] ← 0
while k > 0 do                                              ▷ loop O(log M) times
    for x = 1 to M do
        for y = 1 to M do                                   ▷ For all grid point; O(M^2)
            if ans[x][y] is not zero then
                for (i, j) where i, j ∈ −k, 0, k do
                    if ans[x + i][y + j] is zero then       ▷ if Q isn't marked
                        ans[x + i][y + j] ← ans[x][y]
                    else if ans[x][y] is closer than ans[x + i][y + j] then   ▷ if Q is marked
                        ans[x + i][y + j] ← ans[x][y]
                    end if
                end for
            end if
        end for
    end for
end while
```

3 Problem Definition and Methodology

4 Implementation Details

5 Experimental Evaluation

5.1 Naive Algorithm

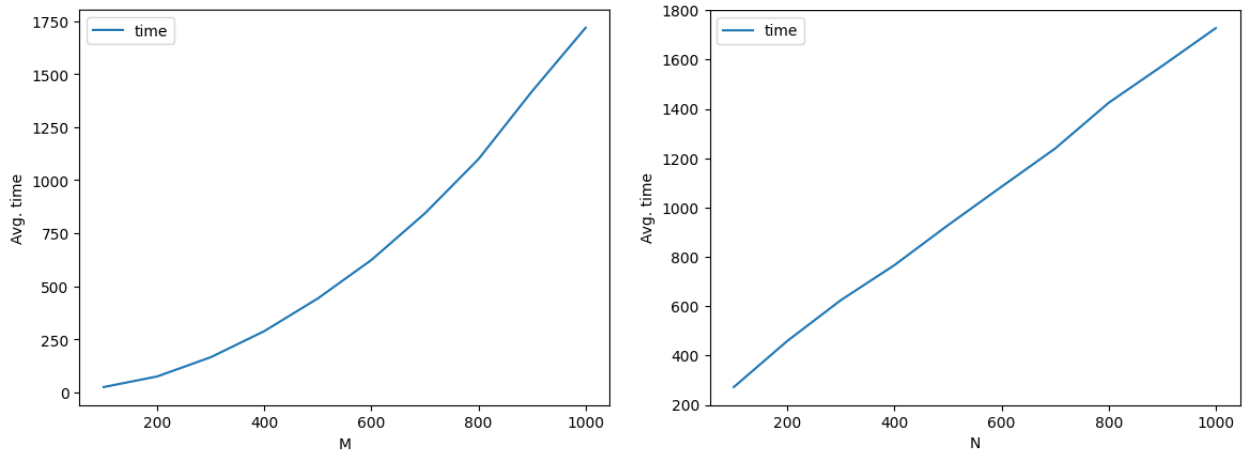

Figure 5.1

상단의 그래프를 보면 실행시간이 M^2과 N에 비례하는 것을 확인할 수 있다. 아래는 파이썬의 sklearn 모듈을 통해 추정한 식이다. 시간 단위는 ms이다.

$$T = 0.0017171M^2 + 8.5583150 \text{ where } N = 1000$$

$$T = 1.6047212N + 127.7733333 \text{ where } M = 1000$$

6 Discussion

7 Future Work

8 Conclusion

9 Acknowledgments

9.1 References

[1] Aurenhammer, F. (1991). Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Computing Surveys (CSUR), 23(3), 345-405.
[2] Lau, B., Sprunk, C., Burgard, W. (2010, October). Improved updating of Euclidean distance maps and Voronoi diagrams. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 281-286). IEEE.
[3] Proceedings of the 2006 symposium on Interactive 3D graphics and games - SI3D '06. I3D '06. Redwood City, California: Association for Computing Machinery. pp. 109–116.

9.2 Appendices