Дальневосточный федеральный университет

Алгоритмы и структуры данных

Алгоритм Форчуна

Выполнила Быкова Анна Вадимовна

Б9121-09.03.03ПИКД

Руководитель доцент ИМКТ Кленин Александр Сергеевич

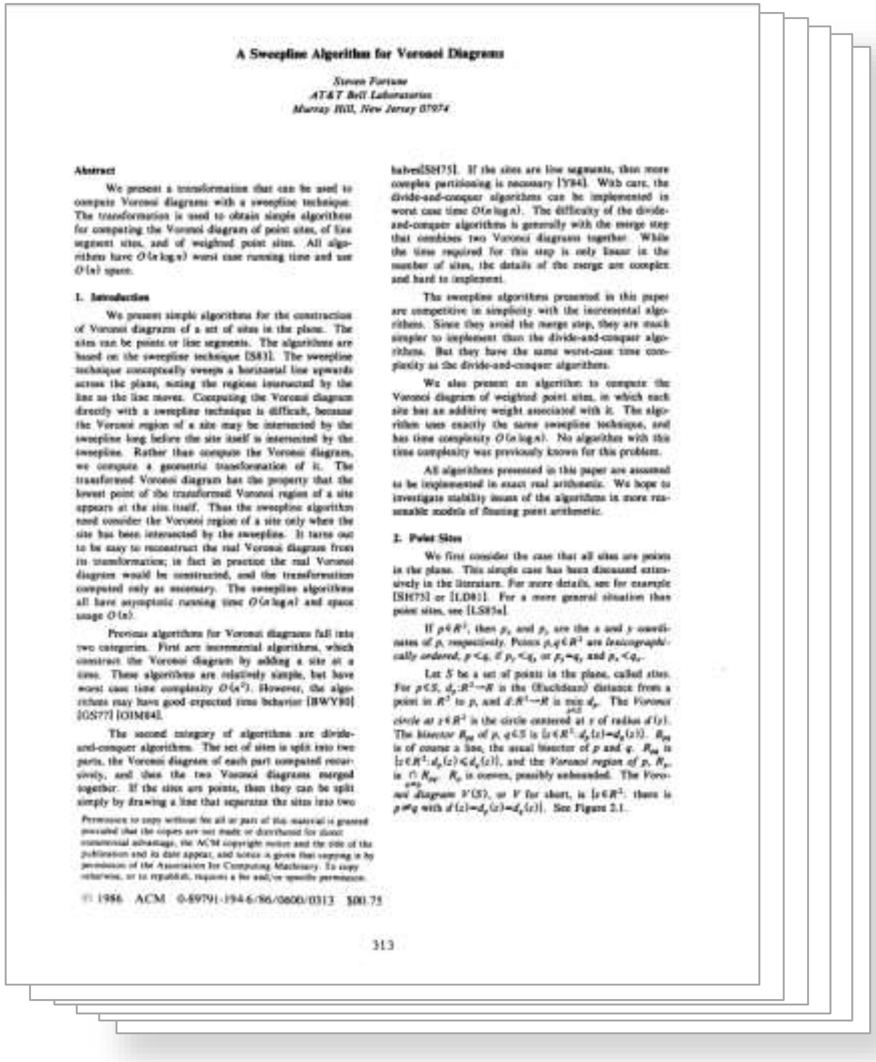
2022

Постановка задачи

1. Изучить алгоритм Форчуна и описать его в виде научного доклада
2. Реализовать алгоритм Форчуна
3. Провести исследование на ***
4. Результат работы выложить на GitHub

История создания

Опубликован Стивеном Форчуном(1985)



A Sweepline Algorithm for Voronoi Diagrams

Steven Fortune
AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract

We present a transformation that can be used to compute Voronoi diagrams with a sweepline technique. The transformation is used to obtain simple algorithms for computing the Voronoi diagram of point sites, of line segment sites, and of weighted point sites. All algorithms have $O(n \log n)$ worst case running time and use $O(n)$ space.

1. Introduction

We present simple algorithms for the construction of Voronoi diagrams of a set of sites in the plane. The sites can be points or line segments. The algorithms are based on the sweepline technique [S83]. The sweepline technique conceptually sweeps a horizontal line upwards across the plane, noting the regions intersected by the line as the line moves. Computing the Voronoi diagram directly with a sweepline technique is difficult, because the Voronoi region of a site may be intersected by the sweepline long before the site itself is intersected by the sweepline. Rather than compute the Voronoi diagram, we compute a geometric transformation of it. The transformed Voronoi diagram has the property that the lowest point of the transformed Voronoi region of a site appears at the site itself. Thus the sweepline algorithm need consider the Voronoi region of a site only when the site has been intersected by the sweepline. It turns out to be easy to reconstruct the real Voronoi diagram from its transformation; in fact in practice the real Voronoi diagram would be constructed, and the transformation computed only as necessary. The sweepline algorithms all have asymptotic running time $O(n \log n)$ and space usage $O(n)$.

Previous algorithms for Voronoi diagrams fall into two categories. First are incremental algorithms, which construct the Voronoi diagram by adding a site at a time. These algorithms are relatively simple, but have worst case time complexity $O(n^2)$. However, the algorithms may have good expected time behavior [BWY80] [GS77] [OIM84].

The second category of algorithms are divide-and-conquer algorithms. The set of sites is split into two parts, the Voronoi diagram of each part computed recursively, and then the two Voronoi diagrams merged together. If the sites are points, then they can be split simply by drawing a line that separates the sites into two halves[SH75]. If the sites are line segments, then more complex partitioning is necessary [Y84]. With care, the divide-and-conquer algorithms can be implemented in worst case time $O(n \log n)$. The difficulty of the divide-and-conquer algorithms is generally with the merge step that combines two Voronoi diagrams together. While the time required for this step is only linear in the number of sites, the details of the merge are complex and hard to implement.

The sweepline algorithms presented in this paper are competitive in simplicity with the incremental algorithms. Since they avoid the merge step, they are much simpler to implement than the divide-and-conquer algorithms. But they have the same worst-case time complexity as the divide-and-conquer algorithms.

We also present an algorithm to compute the Voronoi diagram of weighted point sites, in which each site has an additive weight associated with it. The algorithm uses exactly the same sweepline technique, and has time complexity $O(n \log n)$. No algorithm with this time complexity was previously known for this problem.

All algorithms presented in this paper are assumed to be implemented in exact real arithmetic. We hope to investigate stability issues of the algorithms in more reasonable models of floating point arithmetic.

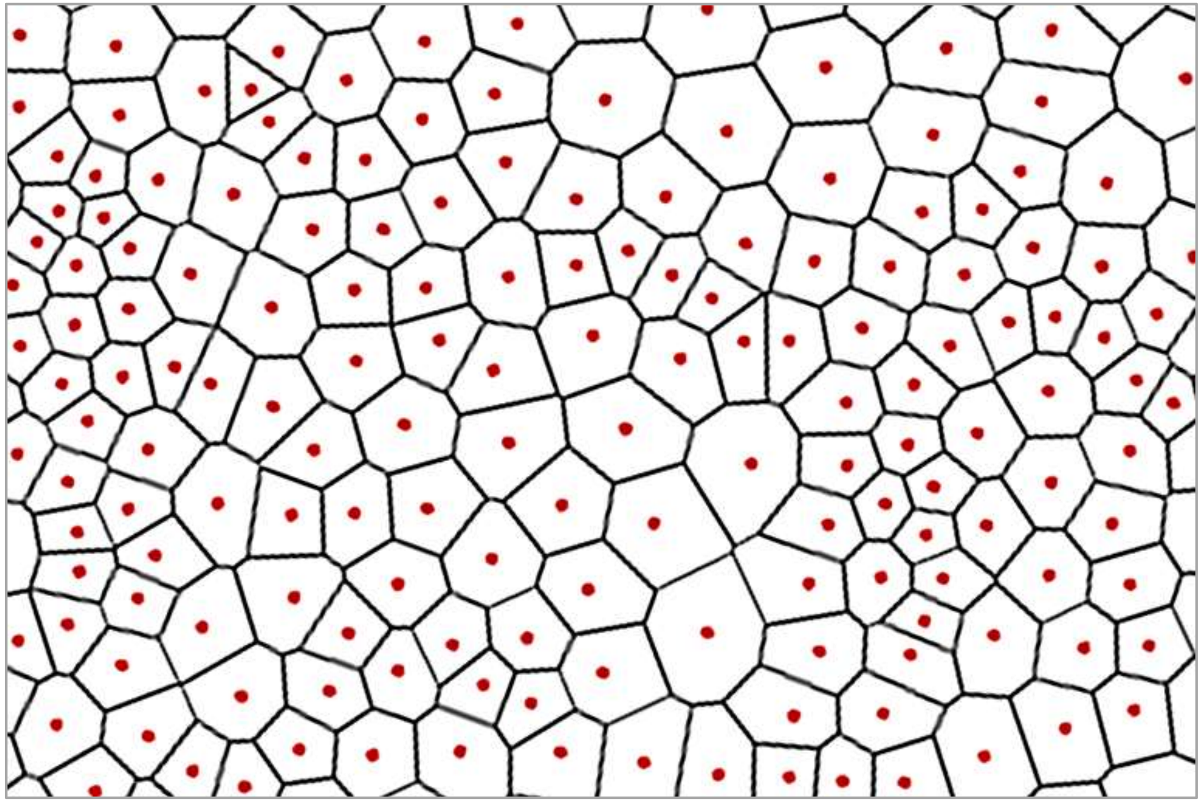
2. Point Sites

We first consider the case that all sites are points in the plane. This simple case has been discussed extensively in the literature. For more details, see for example [SH75] or [LD81]. For a more general situation than point sites, see [LS85a].

If $p \in R^2$, then p_x and p_y are the x and y coordinates of p, respectively. Points $p, q \in R^2$ are *lexicographically ordered*, $p < q$, if $p_y < q_y$ or $p_y = q_y$ and $p_x < q_x$.

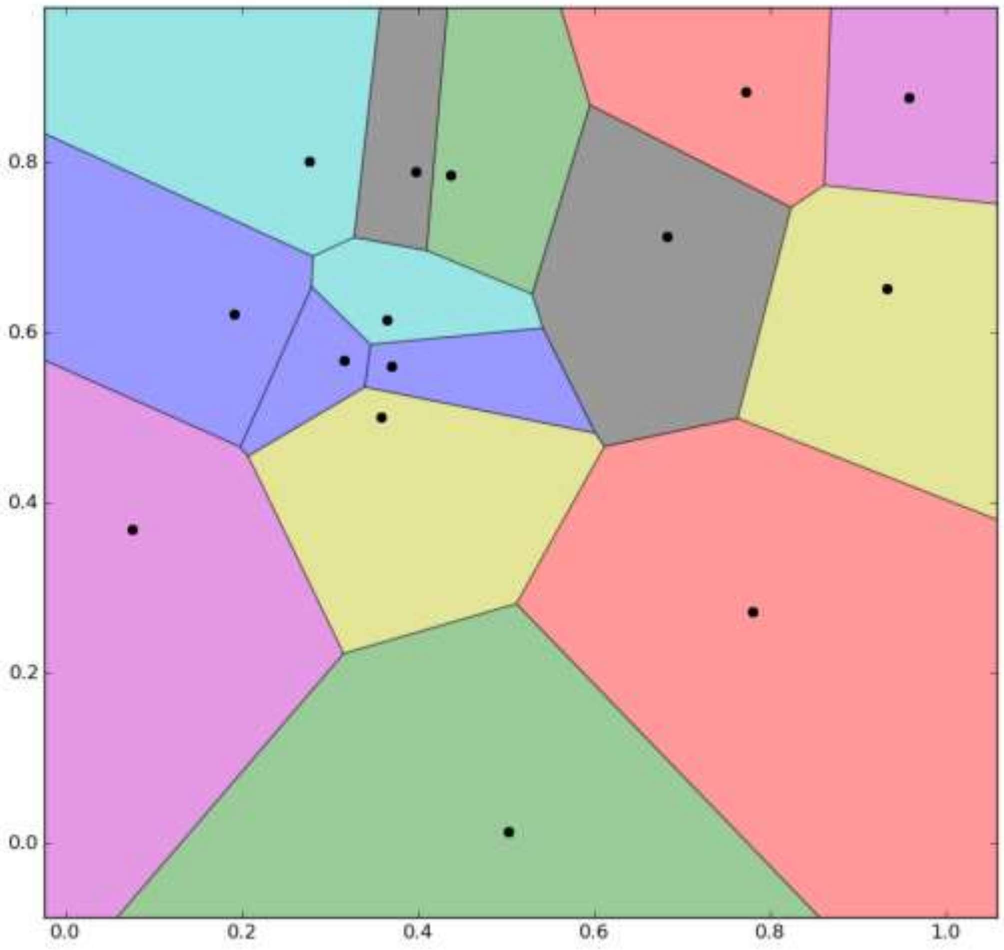
Let S be a set of points in the plane, called sites. For $p \in S$, $d_p : R^2 \to R$ is the (Euclidean) distance from a point in R^2 to p, and $d : R^2 \to R$ is $\min_{p \in S} d_p$. The *Voronoi circle at* $z \in R^2$ is the circle centered at z of radius $d(z)$. The *bisector* B_{pq} of $p, q \in S$ is $\{z \in R^2 : d_p(z) = d_q(z)\}$. B_{pq} is of course a line, the usual bisector of p and q. R_{pq} is $\{z \in R^2 : d_p(z) \le d_q(z)\}$, and the *Voronoi region of* p, R_p, is $\bigcap_{q \ne p} R_{pq}$. R_p is convex, possibly unbounded. The *Voronoi diagram* $V(S)$, or V for short, is $\{z \in R^2$: there is $p \ne q$ with $d(z) = d_p(z) = d_q(z)\}$. See Figure 2.1.

Диаграмма вороного



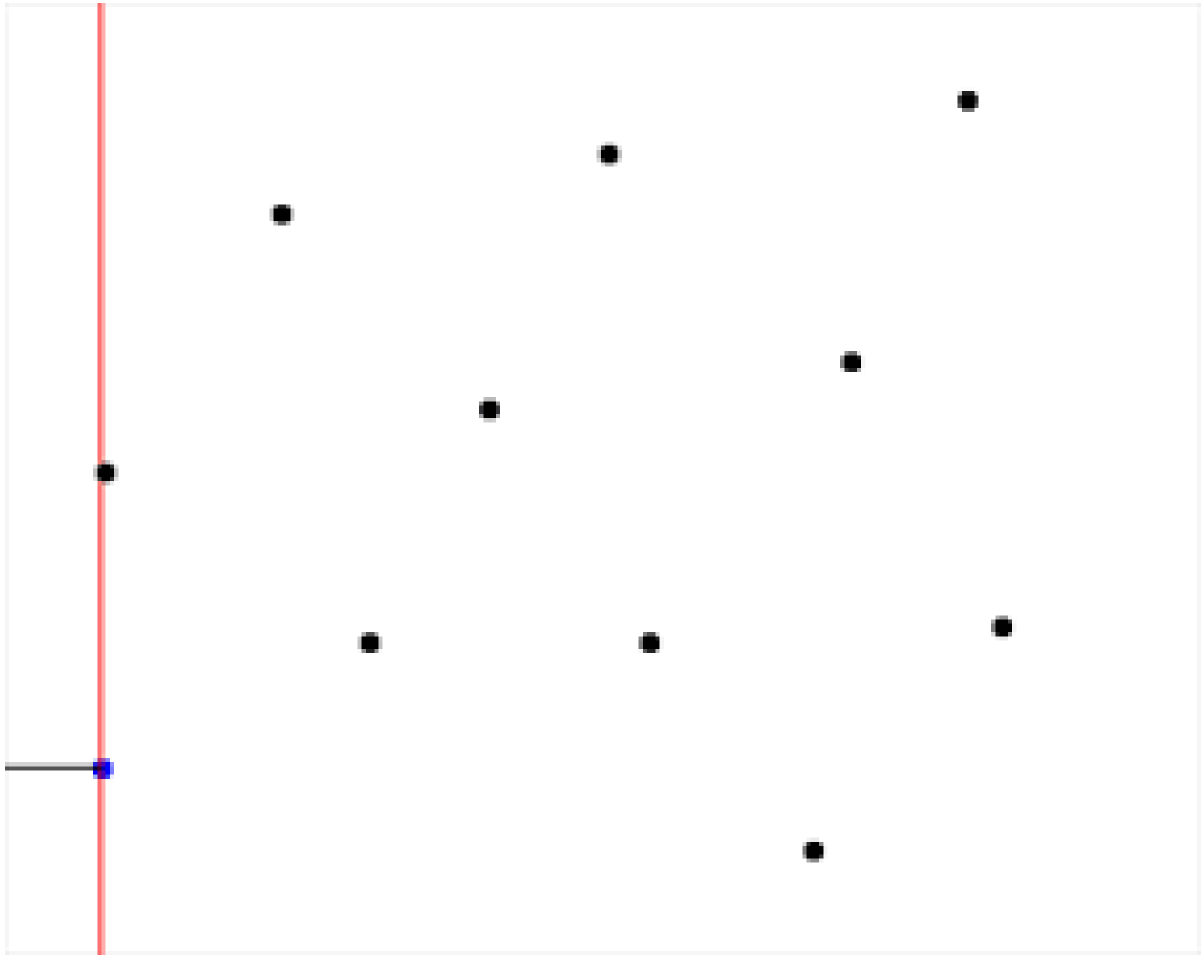
$P=\{p_1,p_2,\ldots,p_n\}$— множество точек на плоскости

Результат работы алгоритма Форчуна



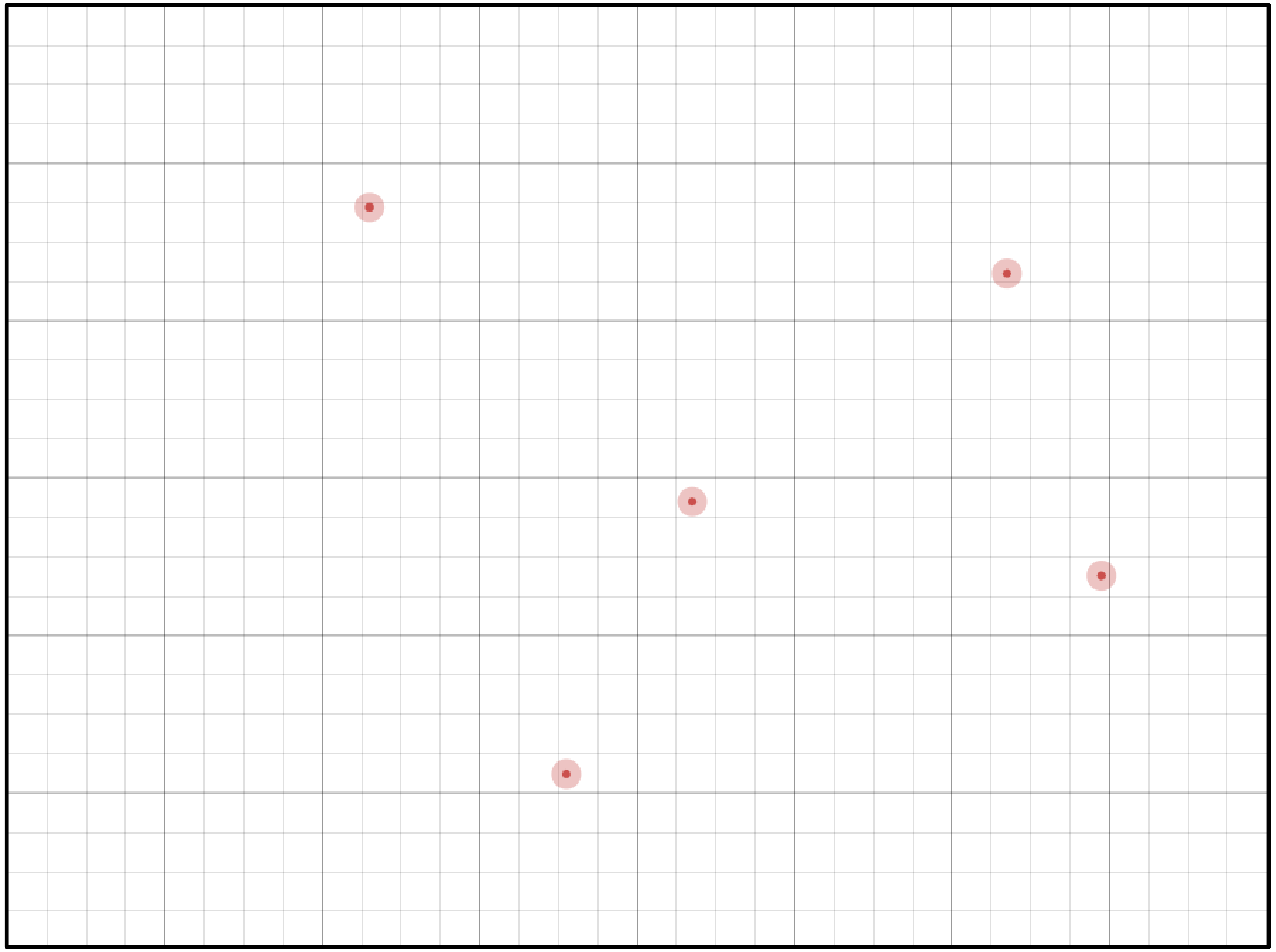
Готовая диаграмма Вороного

Движение заметающей прямой



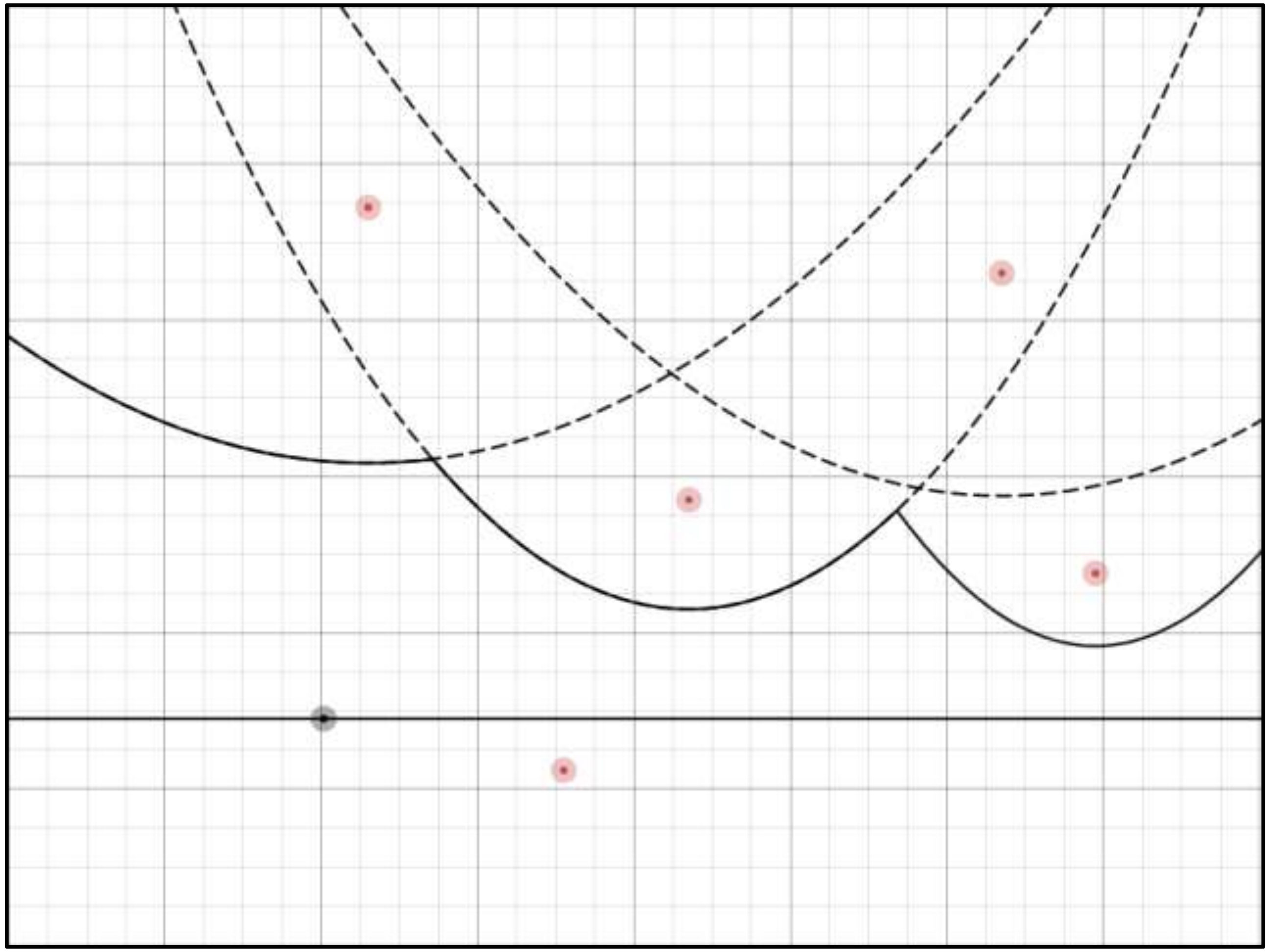
Принцип работы Алгоритма Форчуна

Описание работы. Исходные точки



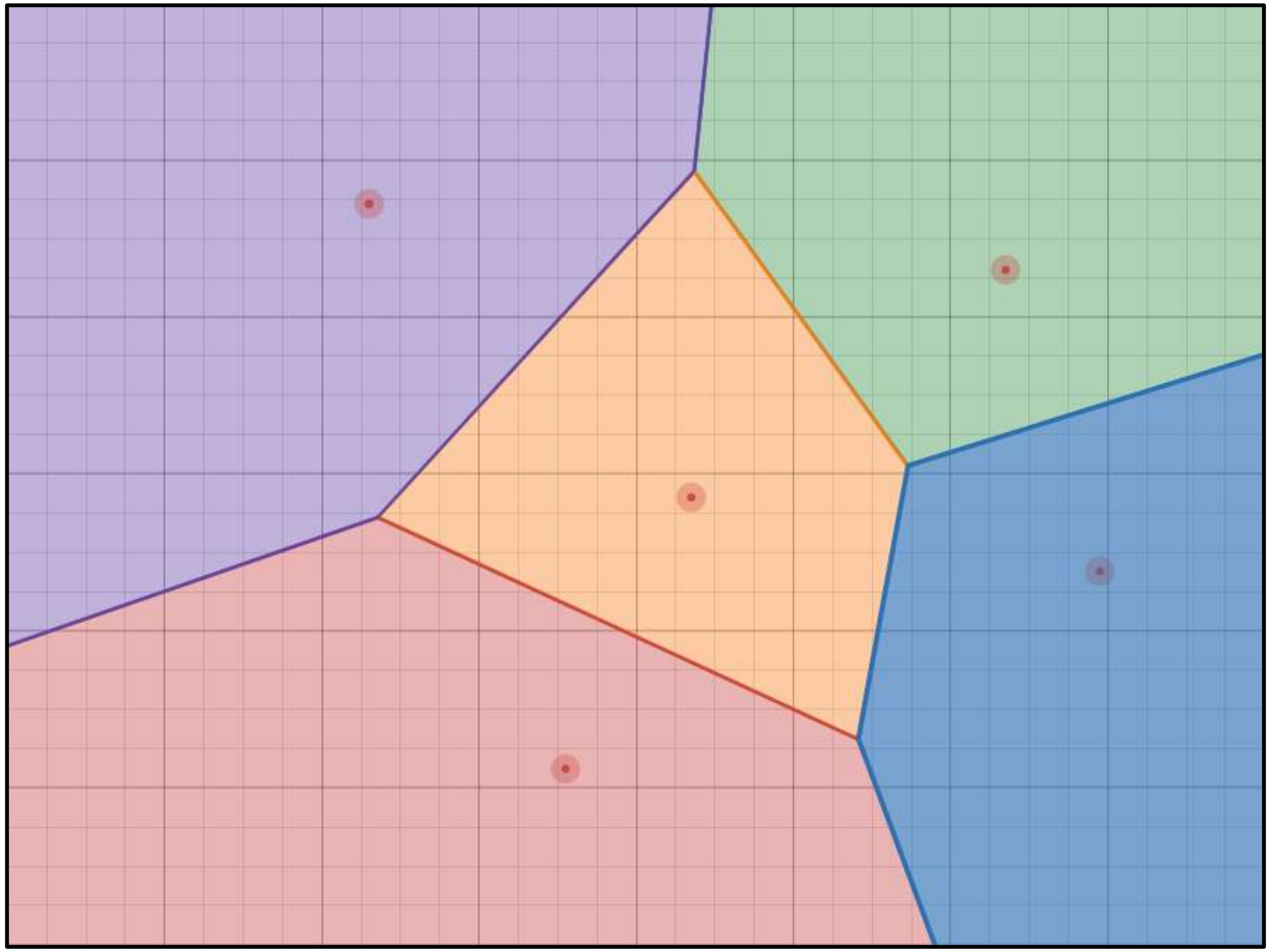
Имеется набор исходных точек – сайтов

Построение парабол, события



Строятся параболы
с фокусом в сайтах

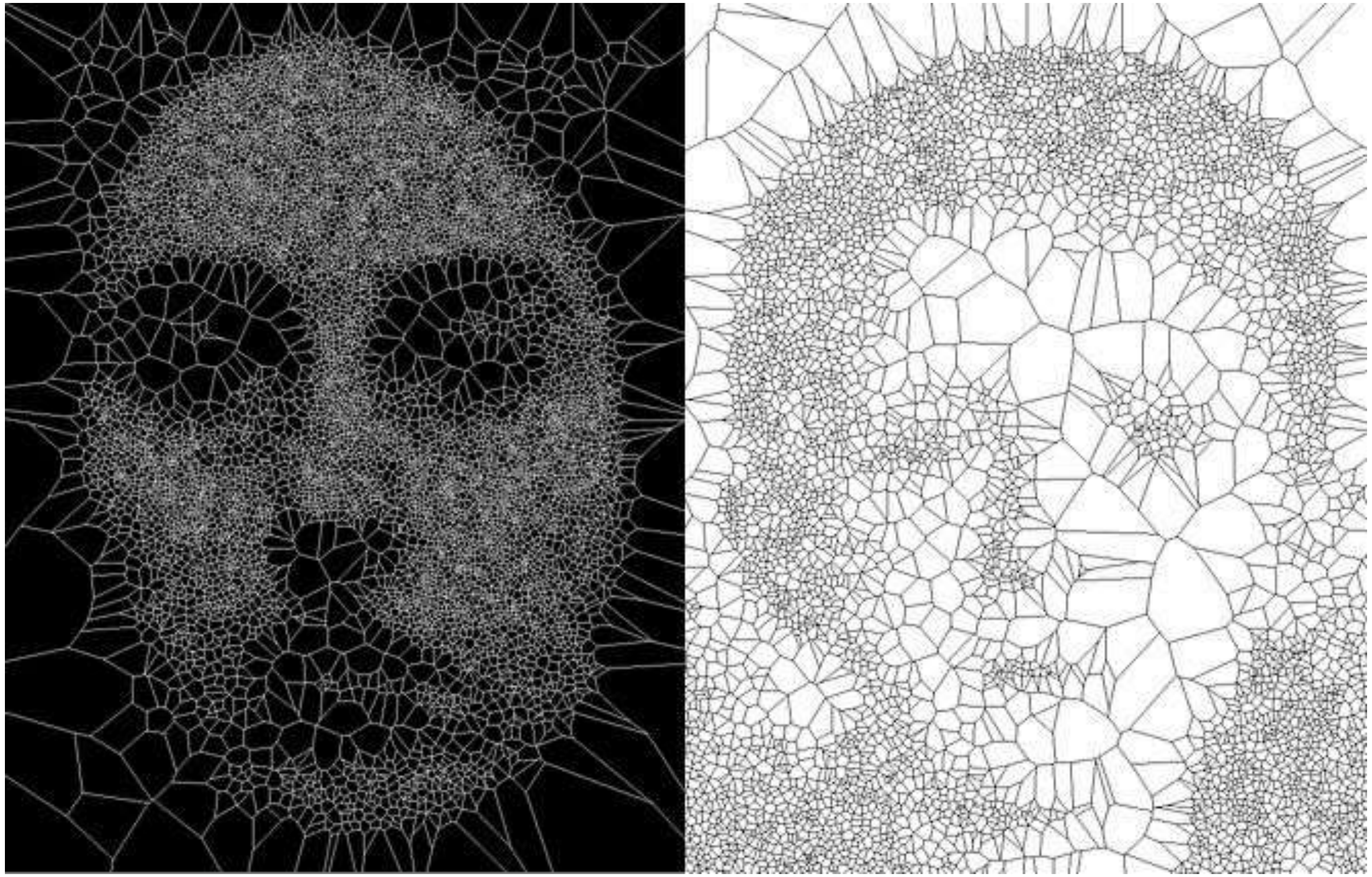
Ребра диаграммы Вороного



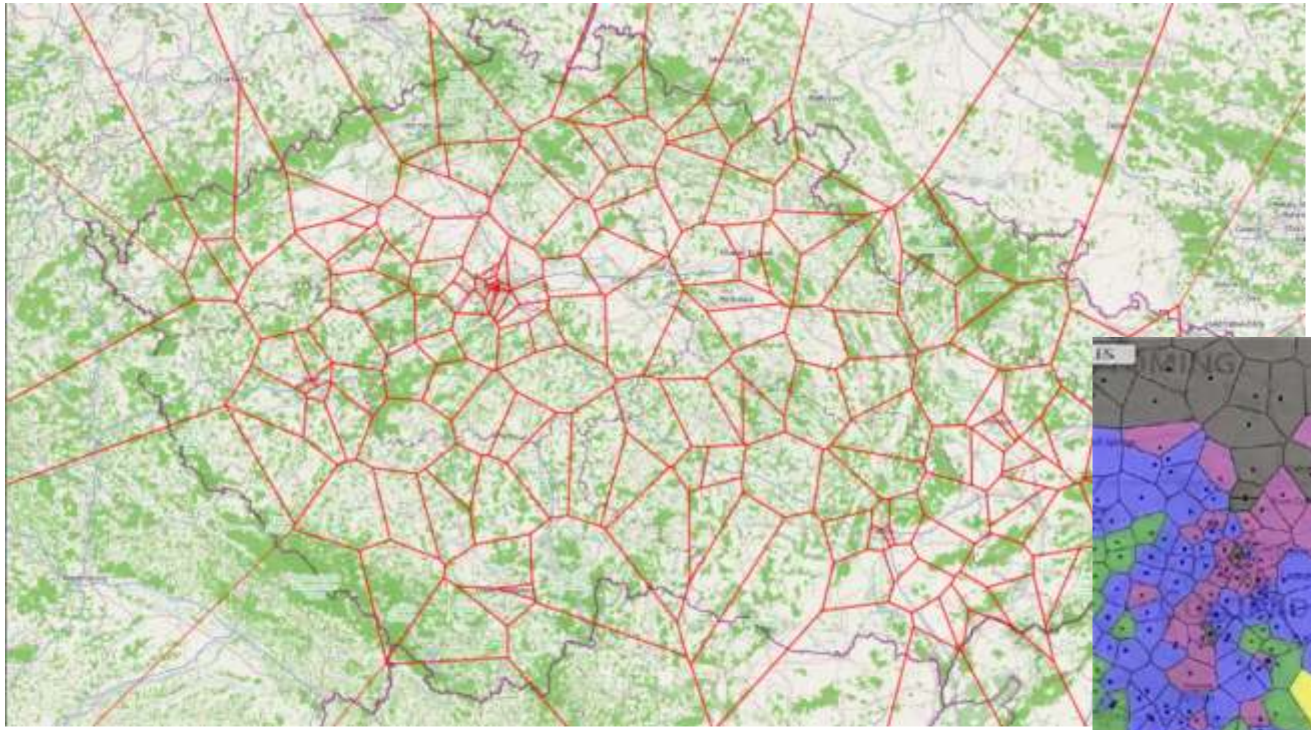
Пересечения ветвей парабол образуют ребра диаграммы Вороного

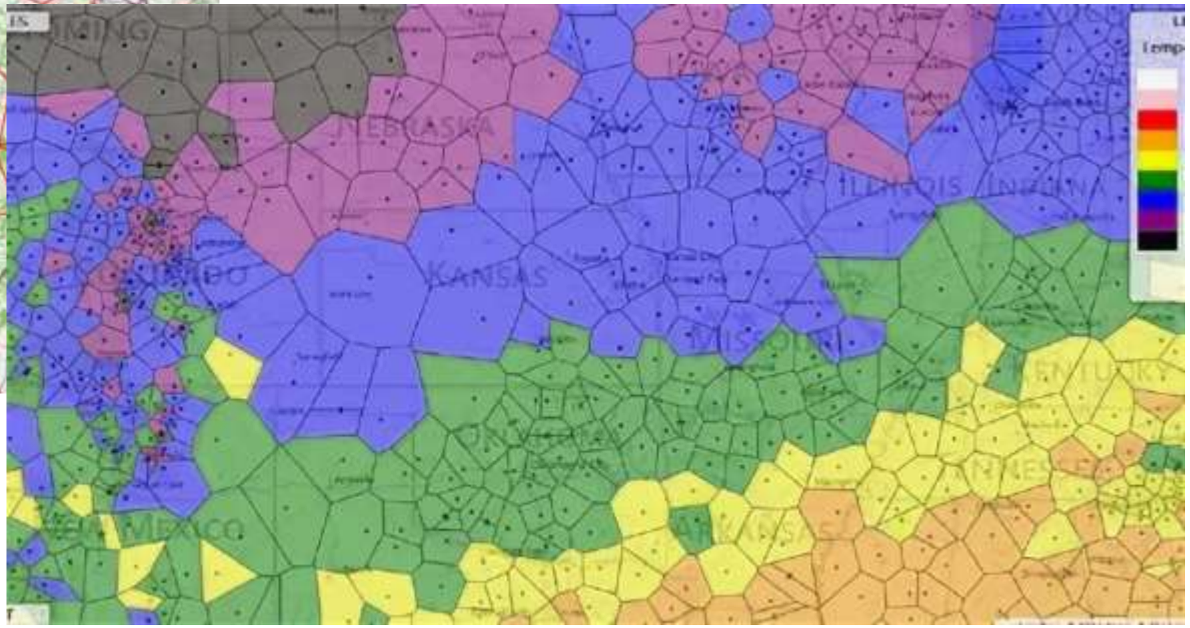
Применение в архитектуре

Применение в моделировании изображений

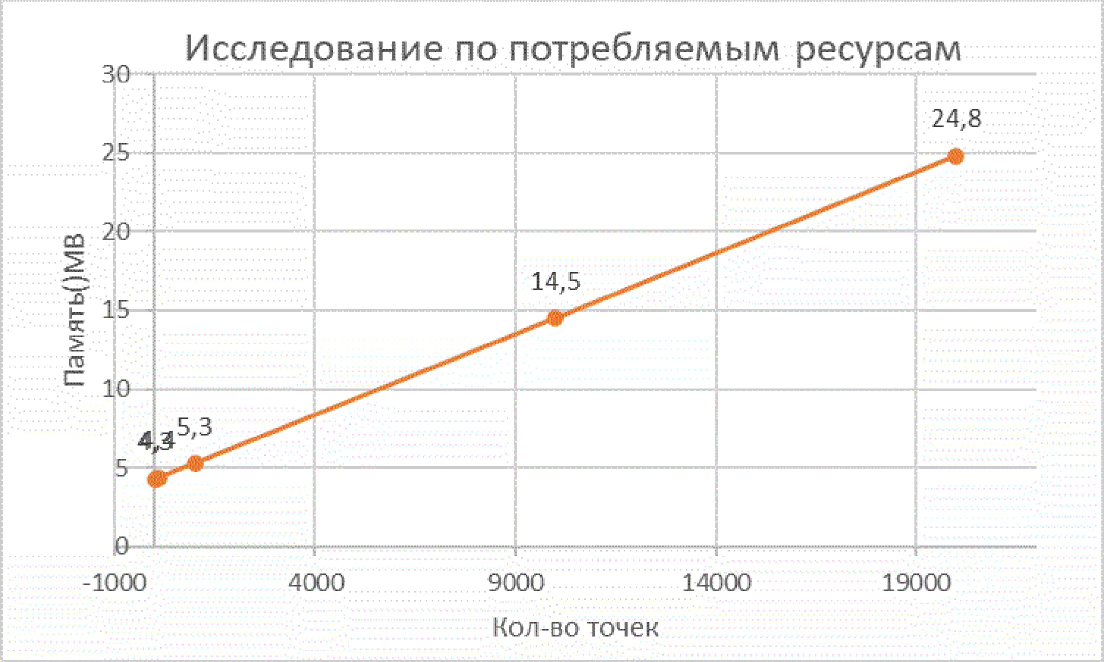


Применение в картографии





Исследование производительности



Заключение

<u>Алгоритм Форчуна:</u>

- изучен по литературным источникам
- изложен в удобной для понимания форме
- реализовано построение диаграммы Вороного по методу Форчуна
- предложена визуализация для получения общего представления
- исследована производительность для разного количества входных данных
    - 10
    - 100
    - 1000
    - 10000