

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

**(ДВФУ)**

|  |
| --- |
| **ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**  **Департамент математического и компьютерного моделирования** |

**ДОКЛАД**

**о практическом задание по дисциплине АИСД**

«Алгоритм Форчуна»

направление подготовки 09.03.03 «Прикладная информатика»

профиль «Прикладная информатика в компьютерном дизайне»

|  |  |  |
| --- | --- | --- |
|  |  | Выполнил студент  гр. Б9121-09.03.03пикд  Быкова Анна Вадимовна \_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Доклад защищен:  С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | *(подпись)*  Руководитель практики  Доцент ИМКТ А.С Кленин  *(должность, уч. звание)*  \_\_\_\_\_\_\_\_\_\_\_\_\_  *(подпись)*  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2022г. |
| Рег. № \_\_\_\_\_\_  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |  |  |

г. Владивосток

2022

**Содержание**

**Введение**

1)суть и назначение

Алгоритм Форчуна – это алгоритм заметающей прямой для генерации диаграммы Вороного из набора точек на плоскости за время O(*n* log *n*) с использованием памяти O(n).

2)авторство

Данный алгоритм был опубликован Стивеном Форчуном в 1986 году в Нью Джерси. Статья носит название «A Sweepline Algorithm for Voronoi Diagrams» *//в интернете нет биографии автора или истории алгоритма*

3)история развития(время, совершенствование, модернизации)

Алгоритм увидел свет в 1986 году.

4)перспектива использования

Алгоритм используется для построения диаграммы Вороного, которая в свою очередь имеет широкий спектр применения.

Она используется в разнообразном геолокационном софте, особенно те, которые показывают распределение чего-либо. Так же применим и в архитектуре, дизайне, поскольку образует красивые причудливые формы.

**Описание метода\алгоритма**

1)формальное  
Основная идея алгоритма — это так называемая **заметающая прямая (ЗП) (sweep line)**. Есть n сайтов (точек на плоскости). Есть заметающая прямая, которая двигается (например) «сверху вниз», то есть от сайта с наибольшей ординатой к сайту с меньшей (от события к событию, если быть точным). Сразу стоит отметить, что влияние на построение диаграммы оказывают *только* те сайты, которые находятся *выше или на* заметающей прямой.

Когда ЗП попадает на очередной сайт (происходит **событие точки (point event)**), создаётся новая парабола (arch), **фокусом** которой является данный сайт, а **директрисой** — заметающая прямая. Эта парабола делит плоскость на две части — «внутренняя» область параболы соответствует точкам, которые сейчас ближе к сайту, а «внешняя» область — точкам, которые ближе к sweep line, ну а точки, лежащие на параболе — равноудалены от сайта и ЗП. Парабола будет меняться в зависимости от положения ЗП к сайту — чем дальше ЗП уходит от сайта вниз, тем больше расширяется парабола, однако в самом начале она вообще является отрезком («направленным» вверх).

Далее парабола расширяется, у неё появляются две **контрольные точки (break points)** — точки её пересечения с остальными параболами («береговой линией»). В «береговой линии» мы храним дуги парабол от одной точки пересечения их друг с другом до другой, так и получается beach line. По сути, в этом алгоритме мы моделируем движение этой «береговой линии». потому как эти самые break point`ы движутся аккурат по рёбрам ячеек Вороного (ведь получается, что контрольные точки равноудалены от обоих сайтов, которым соответствуют эти параболы, да ещё и от ЗП).

И как раз-таки в тот момент, когда две контрольные точки — по одной из разных парабол — «встречаются», то есть как бы превращаются в одну, эта точка и становится вершиной ячейки Вороного (происходит **событие круга (circle event)**), причём в это время та дуга, которая находилась между этими двумя точками — «схлопывается» и удаляется из «береговой линии». Далее мы просто соединяем эту точку с предыдущей соответствующей ей и получаем ребро ячейки Вороного.

2)математическое

Псевдокод:

добавляем событие места в очередь событий для каждого места

пока очередь событий не пуста

извлекаем верхнее событие

если событие является событием места

вставляем в линию побережья новую дугу

проверяем наличие новых событий окружностей

иначе

создаём в диаграмме вершину

удаляем из береговой линии стянутую дугу

удаляем недействительные события

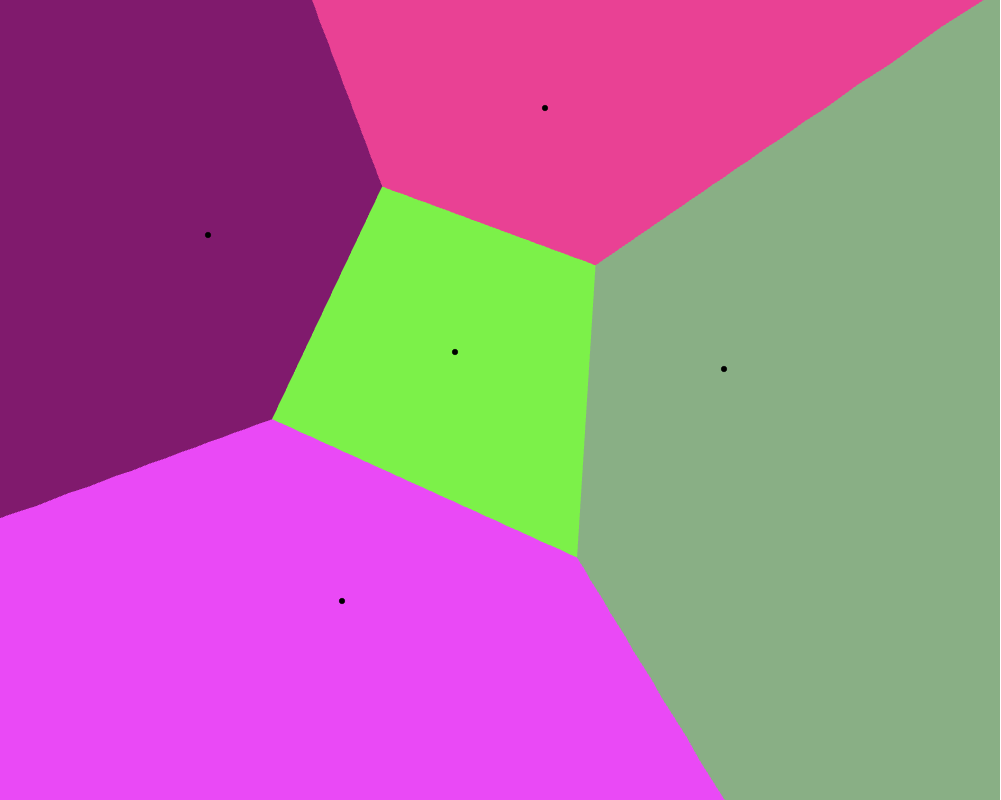
проверяем наличие новых событий окружностей

3)с примером

У нас имеется поле с размерами: ширина – 1000, высота – 800.

Добавим 5 точек с координатами (208,235), (545,108), (342,601), (724,369), (455,352) соответственно.

\*более подробное описание работы алгоритма\*

После работы алгоритма мы увидим такую картину: 

**ФПЗ(формальная постановка задачи)**

1)реализовать, исследовать метод

Даны координаты точек …. Для указанных точек требуется построить диаграмму Вороного, использовуя Алгоритч Форчуна.

2)формат входных данных

На вход подаются координаты точек, по которым нужно постоить диаграмму Вороного.

3)формат выходных данных

На выходе должна получиться визуализированная диаграмма Вороного.

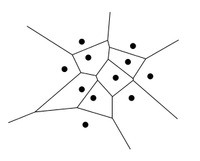
**Список литературы(27 источников)**

1. <https://habr.com/ru/post/309252/>
2. <https://habr.com/ru/post/112581/>
3. <https://neerc.ifmo.ru/wiki/index.php?title=Диаграмма_Вороного>
4. <https://www.cs.hmc.edu/~mbrubeck/voronoi.html>
5. <https://proxy-c-sharp.herokuapp.com/ru/post/430628/>
6. <https://en.wikipedia.org/wiki/Fortune%27s_algorithm>
7. <https://itnan.ru/post.php?c=1&p=315658>
8. <https://habr.com/ru/post/315658/>
9. <https://www.youtube.com/watch?v=ImJ6IC1a_Sg>
10. <https://www.youtube.com/watch?v=3G5d8ob_Lfo>
11. <https://www.boost.org/doc/libs/1_52_0/libs/polygon/doc/voronoi_main.htm>
12. <https://www.pvsm.ru/matematika/211589>
13. <http://obi2ru.blogspot.com/2012/12/spatial-voronoi-diagram-by-java.html>
14. <http://www.raymondhill.net/voronoi/rhill-voronoi.html>
15. <https://codeforces.cc/blog/entry/85638>
16. <https://pvigier.github.io/2018/11/18/fortune-algorithm-details.html>
17. <https://jacquesheunis.com/post/fortunes-algorithm/>
18. <https://www.codeproject.com/Articles/11275/Fortune-s-Voronoi-algorithm-implemented-in-C>
19. <https://www.youtube.com/watch?v=dgEt9Go7GvE>
20. <https://dspace.spbu.ru/bitstream/11701/13589/1/diploma_Ivanova.pdf>
21. <https://temofeev.ru/info/articles/kak-ispolzovat-diagrammy-voronogo-dlya-upravleniya-ii/>
22. <https://www.liveinternet.ru/community/rss_habr/post444611206/>
23. <https://rykovodstvo.ru/other/145548/index.html?page=6>
24. <https://lektsii.net/3-59738.html>
25. <https://dic.academic.ru/dic.nsf/ruwiki/328641>
26. <https://pandia.ru/text/79/483/36865.php>
27. <https://blog.ivank.net/fortunes-algorithm-and-implementation.html>
28. <https://jacquesheunis.com/post/fortunes-algorithm/>
29. <https://blog.ivank.net/fortunes-algorithm-and-implementation.html#impl_cpp>
30. <https://sci-hub.hkvisa.net/10.1145/10515.10549> (оригинал)

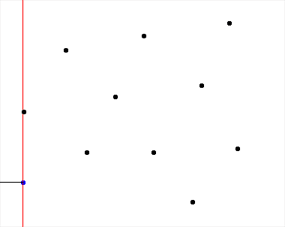
**Доп инфа:**

**Пример**

Пример диаграммы Вороного Пусть есть карта города, на которой точками обозначены почтовые отделения. Человек хочет отправить письмо, и он пойдёт на ближайшую почту. Ему интересно знать, какое отделение ближе, для любой точки города — необходимость отправить письмо может наступить неожиданно. Для этого он может взять карту и расчертить её на ячейки так, чтобы внутри каждой ячейки находилось только одно отделение, а для всех остальных точек ячейки именно эта почта была ближайшей. Полученная картинка и будет диаграммой Вороного для точек

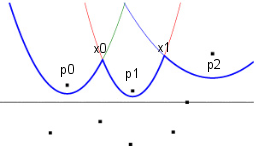
[](https://neerc.ifmo.ru/wiki/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB:Voronoi-diagram.png)

Перед тем, как начать говорить про то, как и что мы будем делать, давайте посмотрим, как движется sweep line (взято [отсюда](http://blog.ivank.net/fortunes-algorithm-and-implementation.html)):



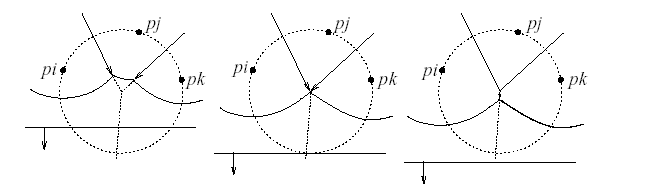
В реализации всё примерно так же, только ЗП обычно движется сверху вниз, а не слева направо, и на самом деле всё не так плавно, а происходит от события к событию (см. ниже), то есть *дискретно*.

##### **Событие точки (point event)**

Событие точки — это попадание ЗП на один из сайтов, поэтому мы создаём новую параболу, соответствующую данному сайту, а также добавляются две **контрольные точки (break points)** (на самом деле сначала — одна, а при расширении арки уже две) — точки пересечения этой параболы с береговой линией (то есть с фронтом уже существующих парабол). Стоит отметить, что в данном алгоритме парабола (а точнее её часть, принадлежащая «береговой лини» — **арка**) «вставляется в береговую линию» **только** в случае события точки, то есть новая арка может появиться только при обработке **события точки**.  
  
Кстати, на следующих картинках видно, почему объединения таких «кусков парабол» называют «береговой линией».  
  
 

##### **Событие круга (circle event)**

Событие круга — это возникновение новой вершины ячейки Вороного вместе с удалением одной арки, потому что возникновение новой вершины диаграммы здесь означает, что было три арки, левая, средняя и правая, средняя «схлопывается» вследствие сближения левой и правой точек арок и получается новая вершина диаграммы Вороного. Стоит отметить, что в данном алгоритме парабола (арка) удаляется из «береговой линии» **только** в случае события круга, то есть арка может удалиться только при обработке **события круга**.



Есть теорема, в которой говорится, что вершина диаграммы Вороного всегда лежит на пересечении ровно трёх рёбер диаграммы, и есть следствие из этой теоремы, которое гласит, что вершина диаграммы является центром окружности, проходящей через три сайта и расстояние от этой точки до заметающей прямой тоже равно радиусу этой окружности (это свойство точек, лежащих на «береговой линии»). Это — **ключевой момент**, потому что именно когда **самая нижняя точка окружности**, проходящей через три сайта, лежит **ниже или на** заметающей прямой, мы пушим в очередь событий событие круга с этой самой нижней точкой, потому что когда прямая на неё попадёт, мы получим вершину диаграммы Вороного.  
  
**Важно**, что с любым событием (точки или круга) связана одна конкретная арка, и наоборот. Это пригодится при обработке событий. Также надо не забыть, что нужно вовремя добавлять рёбра в РСДС (DCEL) (пункт 1 в структурах, см. ниже), так что надо понимать связь арок с рёбрами.  
  
Таким образом, движение прямой дискретно — прямая в любой момент времени **либо на сайте**, **либо в нижней точке окружности**, проходящей через три сайта, центр которой — новая вершина диаграммы Вороного. Прекрасно.  
  
**Общий алгоритм**:

1. Создаём очередь (с приоритетом) событий, изначально инициализируя событиями точки — данным множеством сайтов (ведь каждому сайту соответствует событие точки);
2. Пока очередь не пуста:  
     
   а). Берём из неё событие;  
   б). Если это — *событие точки*, то *обрабатываем событие точки;*  
   в). Если это — *событие круга*, то *обрабатываем событие круга;*
3. Закончить все оставшиеся рёбра (поработать с border\_box).