УДК 519.688

А.В. СКВОРЦОВ

ПОСТРОЕНИЕ СВЕРХБОЛЬШОЙ ТРИАНГУЛЯЦИИ ДЕЛОНЕ

Рассматривается задача построения триангуляции Делоне, заведомо не помещающейся в оперативную память. Предлагается клеточный алгоритм, позволяющий снизить размерность решаемой задачи с N до $\sim 3,23 \cdot N^{2/3}$.

Введение

Проблема построения триангуляции по заданному набору точек на плоскости часто возникает при аппроксимации различного рода физических поверхностей и непрерывных полей, разбиения плоскости на элементарные части в методах конечных элементов, учёта взаимного влияния смежных тел и в других физических задачах. В связи с продолжающимся быстрым ростом производительности вычислительной техники стремительно растут и объемы обрабатываемых данных, в том числе и размеры триангуляционных моделей. В настоящее время реально используемые триангуляционные модели могут содержать миллиарды треугольников.

Среди всех возможных видов триангуляции на практике наиболее часто используется триангуляция Делоне [1], обладающая рядом оптимальных свойств. Для ее построения разработан ряд эффективных алгоритмов, работающих в среднем за время $O(N)$ и требующих $O(N)$ оперативной памяти. Однако зачастую реально возникающие объемы исходных данных N настолько велики, что они не помещаются в оперативной памяти, а поэтому их применение становится невозможным.

Другой проблемой, связанной с огромными массивами исходных данных, является обработка триангуляционных моделей. В большинстве известных алгоритмов требуется полного представления всей структуры триангуляции в оперативной памяти, что не всегда приемлемо. Поэтому необходимо создание алгоритмов, позволяющих извлекать для обработки только необходимые части триангуляции.

Целью настоящей работы является разработка структур данных и создание алгоритмов построения и обработки триангуляции Делоне, работающих за время, близкое к линейному, но затрачивающих только $o(N)$ оперативной памяти.

Клеточный алгоритм

В предлагаемом клеточном алгоритме необходимо выполнить разбиение множества всех исходных точек на некоторые части (клетки), построить триангуляции в отдельных частях, а затем соединить отдельные триангуляции в единое целое.

Основная идея алгоритма заключается в том, чтобы выбрать размеры и количество частей так, чтобы количество точек в каждой части и количество треугольников в зоне слияния триангуляций было примерно равно друг другу и составляло $o(N)$. Это позволит снизить размерность нашей задачи и свести ее к решению множества маленьких задач.

Рассмотрим предлагаемый алгоритм по шагам.

Алгоритм построения сверхбольшой триангуляции Делоне. Дано множество точек на плоскости в виде одного файла, причем занимаемый ими объем памяти превышает объемы доступной оперативной памяти. Требуется построить триангуляцию Делоне.

Шаг 1. Разбиваем множество исходных точек на K частей с помощью клеточного разбиения. Для этого считываем из входного файла координаты точек и записываем их в один из K файлов, соответствующих разным частям (рис. 1,*а*).

Шаг 2. Строим триангуляцию Делоне для множеств точек в каждом из K файлов (рис. 1,*б*).

Если при этом количество точек в очередном файле слишком велико, то можно применить данный алгоритм рекурсивно, иначе нужно применить любой обычный алгоритм построения триангуляции Делоне [1, 2].

Шаг 3. В каждой из K построенных триангуляций определяем, какие треугольники гарантированно не будут перестроены при объединении триангуляции, а какие могут быть потенциально перестроены. Для этого находим для каждого треугольника описанную окружность. Если она хотя бы частично выходит за границы текущей клетки, то условие Делоне в данном треугольнике может быть нарушено точками соседних клеток, а поэтому помечаем этот треугольник как перестраиваемый (рис. 1,*в*)

Шаг 4. В каждой триангуляции определяем точки, попадающие на границу триангуляции либо входящие в состав перестраиваемых треугольников, определенных на предыдущем шаге. Все перестраиваемые треугольники удаляем (рис. 1,*г*).

Шаг 5. Множества всех точек, найденных на предыдущем шаге, подаем на вход некоторого алгоритма построения сшивающей триангуляции Делоне (рис. 1,*д*). Если количество точек в этом множестве велико, то используем данный алгоритм рекурсивно, иначе применяем любой обычный алгоритм построения триангуляции Делоне.

Шаг 6. В полученной сшивающей триангуляции удаляем треугольники, которые пересекаются с треугольниками в отдельных триангуляциях. Для этого выполняем поочередное наложение на сшивающую триангуляцию всех отдельных триангуляций. *Конец алгоритма.*

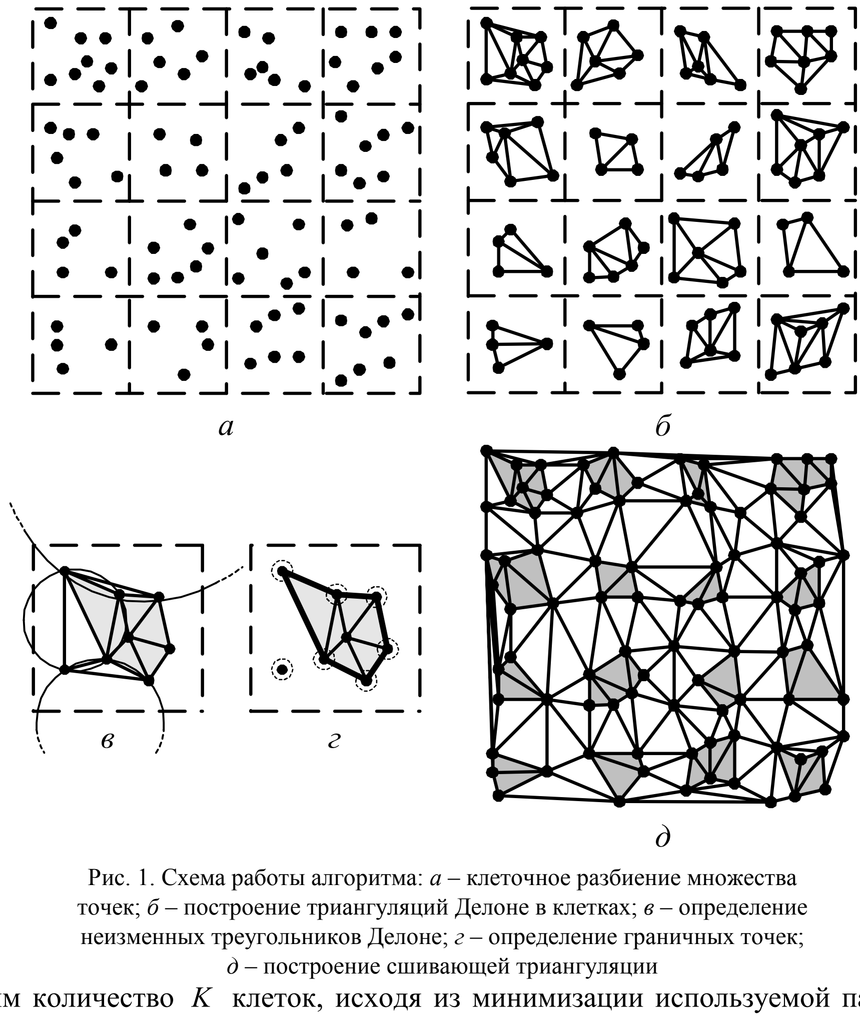

Рис. 1. Схема работы алгоритма: *а* – клеточное разбиение множества точек; *б* – построение триангуляций Делоне в клетках; *в* – определение неизменных треугольников Делоне; *г* – определение граничных точек; *д* – построение сшивающей триангуляции

Определим количество K клеток, исходя из минимизации используемой памяти. Наибольшие затраты памяти возникают тогда, когда мы строим триангуляции отдельных частей (шаг 2), сшивающую триангуляцию (шаг 5), а также когда выполняем оверлей двух триангуляций на шаге 6. Потребуем, чтобы размеры сшивающей триангуляции и триангуляций отдельных частей были равны.

Пусть N – общее количество исходных точек. Тогда, подобрав соответствующее разбиение, можно получить, чтобы в среднем в каждой клетке было $n = N/K$ точек.

Попробуем определить, какое количество точек M войдет в сшивающую триангуляцию. Для этого нужно найти число точек m, попадающих на границу триангуляции только одной из клеток (шаг 4 алгоритма). Тогда $M = mK$.

К сожалению, это число m и даже тип его функциональной зависимости очень сильно зависят от вида распределения исходных точек. Автором было проведено экспериментальное определение m для равномерного распределения исходных точек в единичном квадрате в виде функциональной зависимости $m(n) = k\sqrt{n}$. Выбранный вид зависимости был подтвержден экспериментально с помощью метода наименьших квадратов, а значение коэффициента $k = 5{,}8$ было установлено с точностью до 2 значащих цифр с доверительной вероятностью 0,95. Таким образом, для равномерного распределения в единичном квадрате $m = 5{,}8\sqrt{n}$.

Отсюда получаем $M = mK = 5{,}8\sqrt{N/K} \times K = 5{,}8\sqrt{NK}$.

Из условия равенства размера сшивающей триангуляции и триангуляций отдельных частей найдем количество клеток разбиения в алгоритме:

$$n = M,\ \Rightarrow\ N/K = 5{,}8\sqrt{NK},\ \Rightarrow\ N^2/K^2 = 33{,}64 \cdot NK,\ \Rightarrow$$
$$\Rightarrow\ N = 33{,}64 \cdot K^3 = (3{,}23 \cdot K)^3,\ \Rightarrow\ K \approx 0{,}31\sqrt[3]{N}.$$

Тогда в среднем в каждой клетке и в сшивающей триангуляции окажется по $n = M = 3{,}23 \cdot N^{2/3}$ точек.

Например, если исходные $N_1 = 1\,000\,000\,000$ точек распределены равномерно в единичном квадрате, то $K_1 = 18 \times 18 = 324$ клеток, а $n_1 = 3\,086\,420$, $M_1 = 3\,301\,418$. Такие количества точек уже могут быть обработаны на современных компьютерах обычными алгоритмами триангуляции. Тем не менее, если объемы оперативной памяти этого не позволяют, то предложенный алгоритм можно использовать повторно для построения триангуляции на n_1 и M_1 точках. Тогда на следующем уровне рекурсии будет не более $N_2 = M_1 = 3\,301\,418$ точек, $K_2 = 7 \times 7 = 49$ клеток, $n_2 = 67\,376$, $M_2 = 73\,769$, а такие объемы точек уже точно можно обработать большинством современных алгоритмов [1].

Теперь рассмотрим вопрос трудоемкости работы предложенного алгоритма.

К сожалению, в худшем случае большинство точек может оказаться сосредоточенными в одной из клеток, или все точки окажутся в сшивающей триангуляции. Тогда размерность задачи триангуляции не будет уменьшена, и алгоритм будет работать бесконечно. Однако на практике это вряд ли возникнет, поэтому рассмотрим случай, когда координаты исходных точек распределены равномерно и независимо в единичном квадрате.

Трудоемкость шагов 1, 3 и 4 очевидно в среднем является линейной. Трудоемкость шагов 2 и 5 зависит от сложности используемого там алгоритма триангуляции Делоне. Если используется, например, алгоритм динамического кэширования [1, 2], то в среднем эти шаги будут работать также линейное время. Шаг 6 может быть выполнен с помощью алгоритма наложения, описанного в [3], который также работает за линейное время.

Таким образом, приведенный алгоритм при условии использования на шагах 2 и 5 обычных алгоритмов триангуляции имеет трудоемкость в среднем $O(N)$, при этом затраты по памяти составляют в среднем $O(N^{2/3})$. В случае использования одного уровня рекурсии алгоритм будет также работать в среднем $O(N)$, при этом затраты памяти составят только $O(N^{4/9}) = o(N)$.

В заключение отметим, что важным достоинством данного алгоритма является то, что он позволяет эффективно работать с большинством распространенных структур данных триангуляций [1]. В результате работы алгоритма будут сгенерированы отдельные файлы, соответствующие триангуляциям в отдельных клетках, и один файл сшивающей триангуляции. При использовании двойной нумерации всех узлов и треугольников (номер клетки + номер узла или треугольника) можно достаточно легко работать со сгенерированной структурой данных. При этом по мере необходимости в оперативную память должны подгружаться те или иные части общей триангуляции.

Заключение

Приведенный в работе алгоритм был использован автором для построения тестовой триангуляции на наборе в 1 млрд точек, чьи координаты были распределены равномерно и независимо в единичном квадрате. Общий объем тестовых данных составил 8 Гб. В результате работы было сгенерировано 325 отдельных файлов триангуляции примерно равного размера и общим объемом 64 Гб.

Общее время работы на компьютере с Athlon 1 ГГц, Windows 2000 и 640 Мб оперативной памяти составило около 12 часов, при этом основную долю времени (10 часов) заняло построение триангуляции Делоне на шагах 2 и 5, а также выполнение наложения триангуляций на шаге 6. Остальное время (2 часа) использовалось для выполнения дисковых операций.

СПИСОК ЛИТЕРАТУРЫ

1. Скворцов А.В. // Вычислительные методы и программирование. 2002. Т. 3. Разд. 1. С. 14–39. (http://num-meth.srcc.msu.su)
2. Скворцов А.В. // Изв. вузов. Физика. 1999. №3. С. 120–126.
3. Midtbø T. Spatial Modeling by Delaunay Networks of Two and Three Dimensions: Dr. Ing. Thesis. – Department of Surveying and Mapping, Norwegian Institute of Technology, University of Trondheim, February 1993.

Томский госуниверситет

Поступила в редакцию 01.04.02.