Скелет многосвязной многоугольной фигуры[1]

Л.М.Местецкий
Московский государственный университет, Москва, Россия
l.mest@ru.net

Аннотация

Рассматривается задача построения непрерывного скелета многоугольной фигуры – замкнутой области, граница которой состоит из конечного числа простых непересекающихся многоугольников. Предлагается алгоритм вычисления скелета за время *O(n log n)* в худшем случае, где *n* – общее число вершин фигуры. Отличительной особенностью алгоритма является построение двойственного к диаграмме Вороного графа смежности сайтов, образующих границу фигуры. В основе решения лежит построение дерева смежности всех многоугольников методом плоского заметания. При этом сайты и многоугольники считаются смежными, если они имеют общую касательную пустую окружность. Предложенный подход позволяет обобщить известный алгоритм Ли, используемый для построения скелета простого многоугольника, на случай многосвязной многоугольной фигуры.

Ключевые слова: *многоугольник с дырами, скелет, граф Делоне, алгоритм Ли, смежность многоугольников.*

1. ВВЕДЕНИЕ

Многоугольная многосвязная фигура (ММФ) – это замкнутая ограниченная область в евклидовой плоскости, граница которой состоит из простых замкнутых ломаных линий (многоугольников). ММФ представляет собой «многоугольник с многоугольными дырами» и служит удобной моделью для описания формы объектов в компьютерной графике, анализе и распознавании изображений. В частности, любое бинарное растровое изображение может быть аппроксимировано с помощью ММФ с точностью, равной размеру пиксела, в смысле метрики Хаусдорфа [1].

Мы рассматриваем задачу построения скелета – множества серединных осей ММФ (рис.1). Скелетом замкнутой области называется геометрическое место точек области, являющихся центрами максимальных по включению вписанных окружностей. Скелет используется для исследования топологических и метрических свойств области, что находит применение во многих методах анализа и распознавания изображений. Применение скелетов в системах машинного зрения ставит высокие требования к вычислительной эффективности алгоритмов их построения. Теоретическая оценка вычислительной сложности задачи построения скелета ММФ есть *O(n log n)*, где *n* – число вершин ММФ. Эта оценка получается на основе сведения задачи к построению диаграммы Вороного множества отрезков на плоскости [2]. Из диаграммы Вороного скелет можно получить путем удаления части ребер за время *O(n)*.

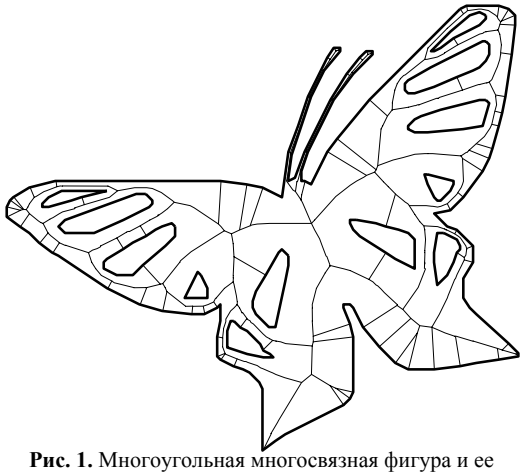

Рис. 1. Многоугольная многосвязная фигура и ее скелет

Среди алгоритмов построения диаграммы Вороного для множества отрезков следует выделить алгоритмы Fortune [3] и Yap [4], имеющие сложность *O(n log n)*. Оба эти алгоритма весьма трудны для реализации. Кроме того, в силу универсальности, они строят диаграмму Вороного не только внутри, но и вне ММФ, что является лишней работой. Более простой (и хронологически более ранний) *O(n log n)* алгоритм Ли [5] строит диаграмму Вороного внутри многоугольника. Он основан на алгоритмической парадигме «разделяй и властвуй». Исходный многоугольник разбивается на две разомкнутые ломаные линии, для каждой из них рекурсивно строится диаграмма Вороного, а затем осуществляется объединение обеих диаграмм. Нужно отметить, что такой подход не вполне корректен, поскольку понятие диаграммы Вороного, сформулированное для внутренности многоугольника, не определено для ломаной линии. Это приводит к тому, что критерий завершения процесса слияния двух диаграмм Вороного, используемый в алгоритме Ли, иногда не срабатывает, что ведет к ошибке алгоритма. Пример, демонстрирующий такую ситуацию, приведен в [6]. Тем не менее, на практике алгоритм Ли широко используется и работает достаточно надежно [7]. Предпринимались неоднократно попытки обобщить алгоритм Ли на случай многоугольника с дырами. Известные решения этой задачи [7,8] имеют сложность $O(kn + n \log n)$, где k – количество дыр. Проблема возникает в том случае, когда число многоугольных дыр велико. Поскольку ситуация $k=O(n)$ является вполне реальной, получается, что вычислительная сложность таких обобщений алгоритма Ли составит $O(n^2)$ в худшем случае, что весьма далеко от оптимального теоретического значения.

[1] Работа выполнена при поддержке РФФИ, гранты 04-01-08058, 05-01-00542.

Предлагаемое в настоящей работе решение задачи представляет собой обобщение алгоритма Ли с сохранением вычислительной сложности *O(n log n)* в худшем случае. Решение основывается на введении понятия смежности контуров границы ММФ и построении так называемого дерева смежности контуров границы ММФ. Кроме того, в основе алгоритма лежит более корректная математическая модель, использующая вместо диаграммы Вороного ее двойственный граф.

2. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Граница ММФ может быть представлена в виде объединения конечного числа точечных множеств, называемых сайтами.

Сайтом называется вершина ММФ (сайт-точка) либо сторона граничного многоугольника ММФ (сайт-сегмент).

Будем называть сайт-точку и сайт-сегмент *соседними*, если они имеют непустое пересечение.

Условимся, что граница ММФ ориентирована таким образом, что внутренность фигуры находится от нее слева. Это значит, что внешний многоугольник границы обходится против, а все внутренние многоугольники – по часовой стрелке.

Для каждого сайта определено понятие внутренней стороны. Для сайта-сегмента внутренней стороной является полуплоскость, лежащая от него слева. А для сайта-точки – это сектор, заключенный между двумя соседними сайтами-сегментами и лежащий слева от них.

Пусть S – множество сайтов, образованных границей ММФ, а S'- некоторое его подмножество $S' \subseteq S$.

Окружность ненулевого радиуса называется *пустой* относительно множества сайтов S', если внутри нее нет точек, принадлежащих сайтам из S'.

Окружность называется *инцидентной* сайту, если ее центр лежит с внутренней стороны от сайта и существует прямая линия, содержащая сайт и являющаяся касательной к окружности в точке, принадлежащей этому сайту.

Два сайта называются *смежными*, если они являются соседними либо имеют общую пустую окружность. Определенное таким образом отношение смежности задает *граф смежности сайтов*. Граф смежности представляет собой двойственный граф диаграммы Вороного ММФ. Естественно называть его графом Делоне, по аналогии с триангуляцией Делоне, являющейся графом смежности сайтов в обычной диаграмме Вороного [9].

Графом Делоне (ГД) ММФ будем называть граф (S,V), у которого множество ребер $V \subseteq S \otimes S$ содержит все пары смежных сайтов из S.

Аналогично графом Делоне для подмножества сайтов S' называется граф (S',V'), в котором $V' \subseteq S' \otimes S'$ и V' содержит все пары смежных сайтов из S'.

В примере на рис.2 изображены диаграмма Вороного ММФ и соответствующий граф Делоне. Вершины графа Делоне соответствуют сайтам ММФ. Сайты-точки изображены кружками, а сайты-сегменты – квадратиками. Ребра соответствуют парам смежных сайтов.

Поскольку диаграмма Вороного является плоским графом, то двойственный ей граф Делоне тоже является плоским, т.е. допускает плоскую укладку. Как для всякого плоского графа, для ГД определено понятие грани. На рис.2 все грани являются треугольными, т.е. образуются тремя попарно инцидентными сайтами. При этом для каждой грани существует пустой круг, инцидентный всем сайтам грани.

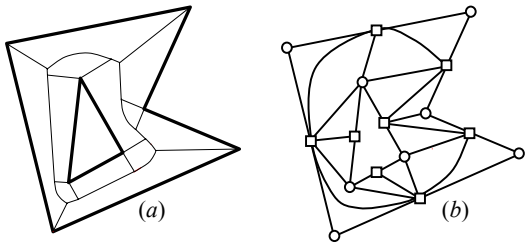

Рис. 2. Диаграмма Вороного (*a*) и граф Делоне ММФ (*b*)

Таким образом, имеет место очевидная аналогия между графом Делоне для ММФ и обычной триангуляцией Делоне для конечного множества точек. В частности, по ГД можно построить диаграмму Вороного для ММФ за время *O(n)*. Поэтому получается, что для эффективного решения задачи скелетизации ММФ вполне достаточно построить *O(n log n)*-алгоритм вычисления графа Делоне для ММФ.

Главное преимущество этого подхода перед традиционным состоит в том, что граф Делоне корректно определен для любого подмножества сайтов ММФ, чего нельзя сказать об определении диаграммы Вороного. Это позволяет корректно сформулировать алгоритм Ли в терминах слияния ГД частей границы ММФ.

3. ГРАФ ДЕЛОНЕ МНОГОУГОЛЬНИКА

Рассмотрим односвязную многоугольную фигуру, т.е. простой многоугольник без дыр. Вершины многоугольника, углы в которых меньше 180° назовем выпуклыми, а остальные вершины – вогнутыми.

Из n-угольника образуется $2n$ сайтов. Пронумеруем все сайты вдоль направления обхода границы.

Далее разобьем последовательность сайтов на элементарные цепочки следующим образом. Во-первых, элементарной цепочкой является сайт, соответствующий выпуклой вершине. Кроме этого, элементарной является цепочка из одного или нескольких сайтов, лежащих между двумя последовательными выпуклыми вершинами. Во втором случае это либо один сайт-сегмент, либо несколько сайтов-сегментов, перемежаемых вогнутыми вершинами. Очевидно, что такое разбиение является единственным. На рис.3 показан пример разбиения многоугольника на цепочки. Выпуклые сайты-точки обозначены черными кружками, вогнутые – белыми, а сайты-сегменты – квадратиками.

Для каждой элементарной цепочки построим граф Делоне. Легко видеть, что в цепочке смежными являются только лишь соседние сайты. Поэтому ГД цепочки, составленной из k сайтов, является связным и имеет k-1 ребро.

Дальнейшее построение ГД многоугольника осуществляется итерационно путем последовательного слияния ГД цепочек. Пусть $C_1, C_2, \ldots, C_p$ – элементарные цепочки сайтов, а $GD(C_1)$, $GD(C_2)$, …, $GD(C_p)$ – ГД этих цепочек. Очередная итерация состоит в том, чтобы слить попарно цепочки, стоящие подряд, и из их ГД построить ГД объединенных цепочек. В результате первой итерации получаем следующие ГД (при четном p): $GD(C_1 \cup C_2)$, $GD(C_2 \cup C_3)$, …, $GD(C_{p-1} \cup C_p)$. Если p

нечетное, то GD(C_p) остается без изменений. В результате полного цикла слияний в очереди образуются $p_1 = \lceil p/2 \rceil$ ГД. Далее цикл слияний повторяется снова, в результате чего образуется $\lceil p_1/2 \rceil$ ГД. Процесс завершится, когда в очереди останется единственный ГД, который и является решением задачи. Очевидно, что общее число циклов слияния ГД составит *O(log p)*.

В качестве примера рассмотрим построение ГД для многоугольника, изображенного на рис.3*a*. Для восьми элементарных цепочек построены графы Делоне GD(0), GD(1), GD(2), GD(3), GD(4), GD(5,6,7), GD(8), GD(9). Итерационный процесс слияния ГД состоит из трех циклов. В первом цикле получаются ГД GD(0,1), GD(2,3), GD(4,5,6,7), GD(8,9), во втором – GD(0,1,2,3) и GD(4,5,6,7,8,9), а в третьем – GD(0,1,2,3,4,5,6,7,8,9).

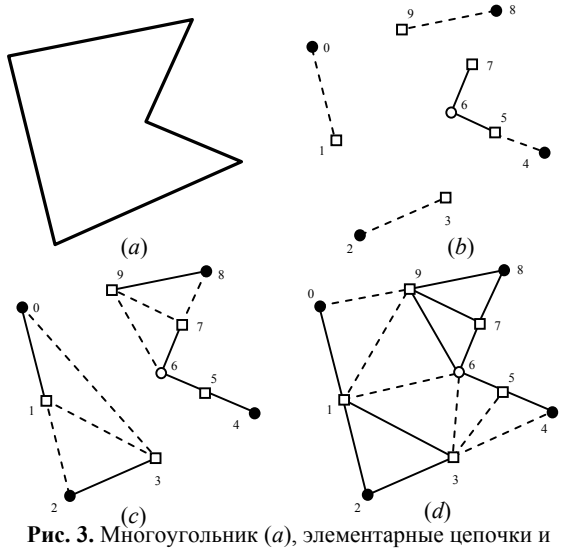

Рис. 3. Многоугольник (*a*), элементарные цепочки и первый цикл слияния (*b*), второй (*c*) и третий (*d*) циклы.

Для того чтобы общая вычислительная эффективность алгоритма составила *O(n log n)*, достаточно обеспечить слияние двух ГД цепочек за время, пропорциональное общему числу сайтов в этих цепочках. Действительно, в этом случае время, требуемое для всех слияний одного цикла, составит *O(n)*. Поскольку число циклов, как было показано, составляет *O(log p)*, общее требуемое время будет *O(n log p)*. Из того, что $p<n$, и получится требуемая оценка.

Таким образом, задача состоит в том, чтобы обеспечить слияние двух ГД за линейное время. Алгоритм, дающий такое решение, чрезвычайно похож на хорошо известный алгоритм Ли-Шехтера слияния обычных триангуляций Делоне [9]. Поясним его работу на примере слияния двух ГД, изображенных на рис.3*c*.

Процесс слияния двух ГД состоит в построении новых ребер и в удалении некоторых старых ребер. Этот процесс напоминает сшивание двух лоскутов. Поскольку сшиваются всегда две последовательно расположенные цепочки сайтов, то последний сайт первой цепочки и первый сайт второй цепочки являются соседними. Это значит, что начальный стежок сшивания соединяет эти два сайта. В примере это сайты 3 и 4. Сайт 4 является выпуклой вершиной, поэтому всегда найдется достаточно маленькая пустая окружность, инцидентная сайтам-сегментам 3 и 5. А это значит, что ребро (3,5) тоже должно быть включено в объединенный ГД. Аналогичным образом строится первая пара новых ребер при сшивке любых ГД последовательных цепочек сайтов.

Следующим шагом должно стать построение нового ребра, инцидентного сайтам 3 или 5 (рис.4*a*). К новому ребру (3,5) примыкают два ребра: (3,0) из левого и (5,6) из правого ГД. А у ребра (3,0) есть инцидентная грань (3,0,1). С этой гранью связана пустая окружность, инцидентная сайтам 3, 0, 1 – вершинам этой грани. Проверка показывает, что сайт 5 попадает внутрь этой окружности, нарушая ее «пустоту». А это значит, что ребро (3,0) должно быть разрушено, потому что в объединенной цепочке сайтов не существует пустой окружности, инцидентной сайтам 3 и 0.

В результате, ребром, примыкающим к сайту 2, становится (3,1). Теперь кандидаты на образование нового сшивающего ребра будут ребра (3,6) и (5,1). Для того, чтобы выбрать одно из них рассмотрим тройки сайтов (3,5,1) и (3,5,6) и попытаемся построить инцидентные им пустые окружности (рис.4*b*). Оказывается, что существует пустая инцидентная окружность только для тройки (3,5,6). Поэтому в качестве очередного сшивающего ребра выбирается ребро (3,6). Дальнейший процесс сшивки осуществляется аналогичным образом. Общий результат представлен на рис.2*b*.

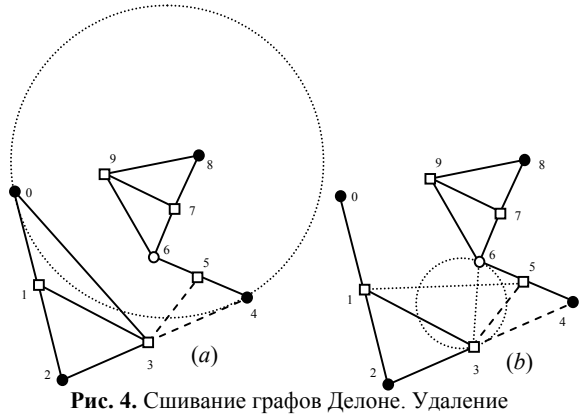

Рис. 4. Сшивание графов Делоне. Удаление лишнего ребра (*a*) и построение нового ребра (*b*).

Следует отметить две особенности, которые имеет описанный алгоритм по сравнению с алгоритмом Ли-Шехтера [9]. Во-первых, построение инцидентной окружности для трех произвольных сайтов является существенно более сложной задачей, чем для трех точек. Здесь возникают 7 различных геометрических задач в зависимости от типа и соседства сайтов [6], для каждой из которых приходится использовать свой метод решения. Во-вторых, при выборе очередного сшивающего ребра из двух кандидатов может возникнуть ситуация, когда существуют пустые окружности для каждого ребра, чего не может быть в обычной триангуляции Делоне. В этом случае из двух окружностей следует выбрать ту, центр которой лежит ближе к центру пустой окружности, построенной на предыдущем шаге.

Линейная оценка сложности алгоритма сшивки двух ГД основывается, как и в алгоритме [9], на том, что общее число операций пропорционально числу удаленных ребер двух исходных ГД и числу построенных ребер нового ГД. Из линейного времени слияния ГД следует, что общая

вычислительная сложность алгоритма построения ГД n-угольника составляет *O(n log n)*.

Таким образом, формулировка алгоритма Ли в терминах построения графа Делоне позволила преодолеть сложности, возникающие при его традиционном использовании для построения диаграмм Вороного. Однако, проблема обобщения этого алгоритма на случай ММФ («многоугольника с дырами») остается. Предлагаемый ниже подход к ее решению основывается на понятии дерева смежности многоугольников, составляющих границу ММФ.

4. ДЕРЕВО СМЕЖНОСТИ МНОГОУГОЛЬНИКОВ

Рассмотрим множество многоугольников, составляющих границу ММФ. Два многоугольника из этого множества будем называть смежными, если в них существует пара смежных сайтов, принадлежащих разным многоугольникам. Другими словами, многоугольники являются смежными, если существует пустая окружность, инцидентная им обоим. Заданное таким образом отношение смежности контуров определяет граф смежности контуров. Очевидно, что этот граф является связным. Поэтому его остов (минимальный связный покрывающий подграф) является деревом. Такое дерево будем называть деревом смежности контуров границы ММФ. Очевидно, что дерево смежности не является единственным. На рис.6*a* представлена ММФ, имеющая 13 граничных контуров. Пустые круги, касающиеся пар контуров, демонстрируют смежность контуров. На рис.5*b* показан граф смежности контуров. Ребра одного из деревьев смежности контуров («вертикального дерева») выделены жирными линиями.

Дерево смежности контуров границы ММФ дает возможность свести задачу построения ГД ММФ к рассмотренной выше задаче построения ГД простого многоугольника.

5. ГРАФ ДЕЛОНЕ МНОГОУГОЛЬНОЙ ФИГУРЫ

Имея дерево смежности контуров границы ММФ, можно построить маршрут обхода всех сайтов ММФ. Этот маршрут позволяет выстроить все сайты ММФ в единую цепь.

Пусть $M_0,M_1,\ldots,M_k$ – контура ММФ, причем M_0 – единственный внешний контур, а остальные – внутренние. В дереве смежности контуров выберем в качестве корня внешний контур M_0.

Для каждого контура M_i, i=0,…,k, построим цепочку из k_i сайтов $\{s_{i1}, s_{i2}, \ldots, s_{ik_i}\}$, упорядоченную по направлению контура, т.е. против часовой стрелки для M_0 и по часовой стрелке для остальных контуров. В дереве смежности контуров каждый элемент является потомком другого элемента. Это значит, что в каждом контуре-потомке имеется сайт, для которого существует смежный сайт в контуре-предке. Не нарушая общности, будем считать, что в цепочке сайтов контура M_i, i=1,…,k этот «соединительный» сайт стоит первым, т.е. это сайт s_{i1}.

Теперь преобразуем все цепочки сайтов ММФ в единую цепочку путем «врезки» друг в друга цепочек отдельных контуров. Этот процесс осуществляется рекурсивно, начиная с контуров, являющихся листьями в дереве смежности. Цепочка каждого контура «врезается» в цепочку его предка.

Пусть контур M_i является потомком контура M_j, причем пара смежных сайтов – это первый сайт s_{i1} в цепочке M_i, а также некоторый сайт s_{jt} в цепочке контура M_j. Тогда объединенная

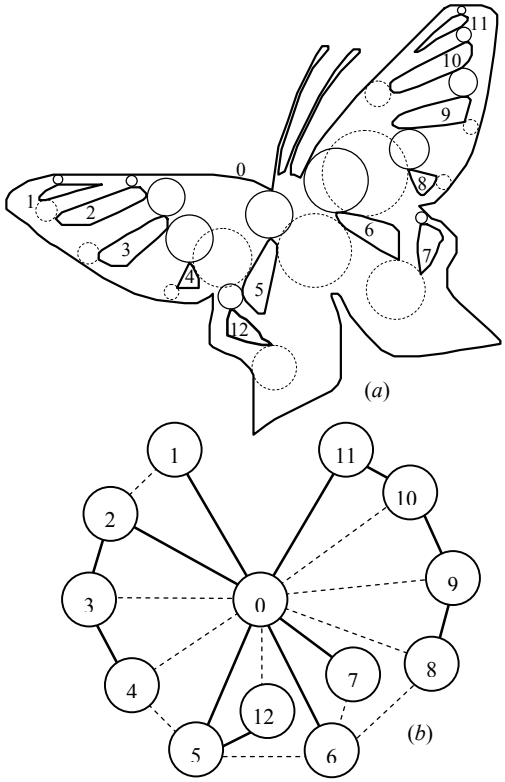

Рис. 5. Пустые окружности, инцидентные смежным контурам (*a*), граф смежности контуров (*b*) (дерево смежности выделено сплошными

цепочка имеет следующий вид:

$$\{s_{j1}, s_{j2}, \ldots, s_{jt}, s_{i1}, s_{i2}, \ldots, s_{jk_i}, s_{i1}, s_{jt}, s_{jt+1}, \ldots, s_{jk_j}\}$$

Цепочка контура M_i вставляется в цепочку контура M_j после элемента s_{jt}, а после последнего ее сайта ставится пара сайтов s_{i1},s_{jt}, образующая «мост» для обратного перехода с контура M_i на контур M_j.

У контура M_j может быть несколько потомков, и цепочки сайтов всех их должны быть вставлены таким образом в цепочку сайтов M_j. После этого цепочка самого M_j вставляется в цепочку его предка в дереве смежности. В результате этого процесса будет выполнена вставка всех цепочек в цепочку сайтов внешнего контура. Очевидно, что длина этой объединенной цепочки сайтов составит *O(n)*.

Теперь к объединенной цепочке сайтов может быть применен алгоритм построения ГД, который был использован для простого многоугольника. При этом необходимо сделать следующие модификации, связанные с появлением в цепочке пар смежных сайтов, не являющихся соседними.

1. Процесс образования элементарных цепочек теперь строится в два этапа. Сначала образуются элементарные цепочки из пар смежных сайтов разных контуров. А затем строятся элементарные цепочки из сайтов одного контура так, как это делается в случае простого многоугольника.

2. После того, как будет построен ГД объединенной цепочки, ребра, образованные связующими парами сайтов (из дерева смежности контуров), окажутся продублированными, т.е. войдут в ГД дважды. Поэтому необходимо провести «склейку» таких ребер.

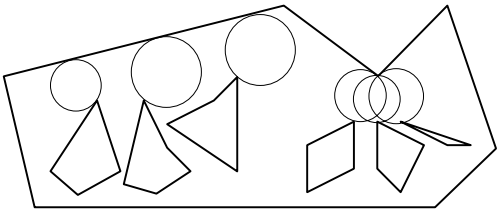

Рис. 6. Случай присоединения нескольких контуров-потомков к одному и тому же сайту контура-предка.

Следует отметить особый случай, когда несколько контуров имеют общего предка в дереве смежности и при этом их пустые окружности инцидентны одному и тому же сайту в контуре-предке (рис.6). В этом случае врезка цепочек контуров-потомков в цепочку контура-предка осуществляется строго в той последовательности, в которой происходит касание окружностей общего сайта в контуре-

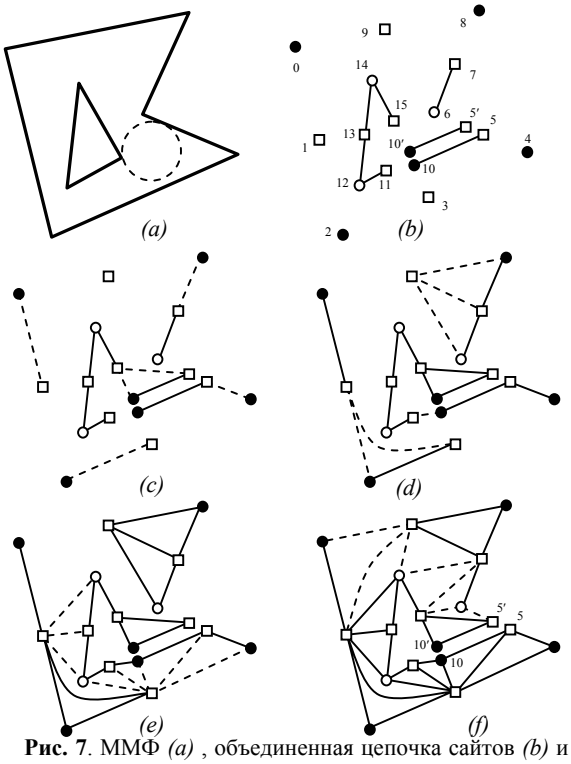

Рис. 7. ММФ *(a)* , объединенная цепочка сайтов *(b)* и 4 цикла слияния ГД цепочек *(c-f)*.

предке.

Пример работы алгоритма представлен на рис.7. Исходная ММФ (рис.7*a*) имеет два граничных контура, которые образуют 16 сайтов. Предположим, что найдена пустая окружность, показывающая, что сайты 5 и 10 являются смежными. В результате объединения цепочек двух контуров получается объединенная цепочка сайтов:

{0, 1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 10 ,5 ,6 , 7, 8, 9}.

В ней выделяются элементарные цепочки сайтов. При этом образуются две элементарные цепочки из пары смежных сайтов, обозначенные (5,10) и (10′,5′) на рис.7*b*. Далее строятся ГД 11 элементарных цепочек, которые последовательно сливаются в общий ГД за 4 цикла (рис.7*c-f*). В полученном ГД (рис.7*f*) остается склеить ребра (5,10) и (10′,5′), в результате чего получается искомый ГД, изображенный на рис.2*b*.

Таким образом, при наличии дерева смежности контуров построение графа Делоне ММФ можно осуществить с использованием того же самого алгоритма, который строит ГД простого многоугольника, за время *O(n log n)*. Естественно, что ценность этого результата зависит от того, какое время потребуется для построения дерева смежности контуров. Предлагаемый ниже алгоритм позволяет построить это дерево за время *O(n log n)* в худшем случае.

6. ПОСТРОЕНИЕ ДЕРЕВА СМЕЖНОСТИ КОНТУРОВ ГРАНИЦЫ ФИГУРЫ

6.1 Вертикальное дерево смежности контуров

Среди всех возможных деревьев смежности мы будем строить так называемые «ориентированные по направлению» деревья. Не нарушая общности, будем рассматривать в качестве такого направления ось ординат («вертикальное направление»). Для каждого внутреннего граничного контура выделим вершину, имеющую максимальную проекцию на это направление. Для вертикального направления это просто вершина с максимальной ординатой. В случае если таких вершин в контуре несколько, выберем одну из них. Эту вершину будем называть *макушкой* многоугольника. С каждой макушкой свяжем *луч*, выходящий из нее вверх.

Рассмотрим множество пустых окружностей инцидентных макушке, центры которых расположены на ее луче. Среди таких окружностей всегда найдется максимальная. Такую окружность будем называть *пузырем*. Пузырь инцидентен макушке и сайту из другого многоугольника. Этот сайт назовем *опорным* для макушки и *контактным* для пузыря. Если таких сайтов несколько, выберем в качестве опорного лишь один из них (любой). Многоугольник, в котором расположен опорный сайт, назовем верхним, поскольку он всегда лежит выше макушки. Легко видеть, что каждый многоугольник является смежным со своим верхним многоугольником. Таким образом, построенные пузыри определяют отношение частичного порядка на множестве многоугольников: у каждого внутреннего многоугольника существует единственный верхний многоугольник. Единственный многоугольник, не имеющий верхнего – это внешний граничный многоугольник. Следовательно, нами получено дерево смежности контуров границы ММФ, в котором пустыми окружностями, инцидентными парам контуров, являются пузыри многоугольников. Такое дерево смежности будем называть *вертикальным деревом смежности*. Пример вертикального дерева смежности контуров для ММФ «бабочка» представлен на рис.5*b*.

Можно предложить следующий (наивный) алгоритм построения вертикального дерева смежности контуров. Для каждой макушки перебрать все сайты ММФ и для каждого сайта построить пустую окружность с центром на вертикальном луче, инцидентную макушке и сайту (если такая окружность существует). А потом отобрать окружности

минимального радиуса для всех макушек. Очевидно, что вычислительная сложность такого алгоритма составит $O(p \cdot n)$, где p – количество контуров в границе ММФ. Поскольку в худшем случае $p=O(n)$, получаем сложность наивного алгоритма $O(n^2)$, что является неприемлемым. Требуется более эффективный алгоритм.

6.2 Алгоритм плоского заметания

Источником низкой эффективности наивного алгоритма является перебор для каждой макушки всех сайтов, в том числе и тех, которые находятся далеко и заведомо не могут оказать влияния на построение ее пузыря. Для того, чтобы исключить анализ таких несущественных сайтов, воспользуемся идеей плоского заметания.

Рассмотрим горизонтальную прямую, перемещаемую снизу вверх, последовательно заметающую ММФ. Процесс заметания условно развивается во времени, с каждым моментом времени связано некоторое положение заметающей прямой. При этом множество сайтов, пересекаемых заметающей прямой, естественным образом упорядочено слева направо по абсциссам точек пересечения. Рассмотрим также множество вертикальных лучей, выходящих из макушек. В момент пересечения макушки заметающей прямой соответствующий луч может занять свое место в упорядоченном множестве пересекаемых сайтов. Нетрудно убедиться в справедливости следующего утверждения:

Если сайт является опорным для пузыря, то существует такое положение заметающей прямой, при котором между этим сайтом и лучом нет других сайтов.

Это необходимое условие позволяет организовать поиск опорного сайта для макушки среди тех сайтов, которые оказались соседними с лучом макушки в процессе заметания.

Процесс заметания реализуется, как это обычно принято [10,11], с помощью двух структур данных: статуса заметающей прямой и очереди событий. Статус заметающей прямой представляет собой упорядоченное множество сайтов и лучей, пересекаемых заметающей прямой. Статус динамически меняется по мере перемещения прямой. Моменты изменения статуса называются событиями. События естественным образом упорядочиваются по времени наступления. Упорядоченное множество событий образует очередь событий.

Статус заметающей прямой и очередь событий реализуются в виде сбалансированных деревьев [12] (АВЛ-, 2-3- или красно-черные деревья), в результате чего выполнение операций поиска, включения и исключения элементов в структурах осуществляется за время O(log K), где K – число элементов, находящихся в структуре.

Изменения статуса состоят в том, что заметающая прямая «наехала» на очередной сайт, либо «съехала» с какого-то сайта. Очевидно, что это происходит при пересечении заметающей прямой вершины ММФ, следовательно, общее число таких событий составляет $O(n)$. Поскольку общее число элементов в статусе есть также $O(n)$, время, требуемое на операции включения-исключения сайта или луча, составляет $O(\log n)$. Для каждого события необходимо проверить, не образовалась ли пара луч-сайт такая, что между ними в статусе нет других лучей. Если образовалась, то нужно проверить их на контакт.

Практическую реализацию этого процесса продемонстрируем на примере (рис.8). Для макушки многоугольника S_0 в момент, когда она попала в статус, строим вертикальный луч R_0 и пузырь бесконечного размера, т.е. полуплоскость C_0. При перемещении заметающей прямой в положение L_1 в статус попадает сайт S_1, который оказывается соседом с лучом R_0. Инцидентная сайтам S_0 и S_1 окружность C_1 имеет радиус меньше, чем C_0, поэтому она становится кандидатом в пузыри для макушки S_0.

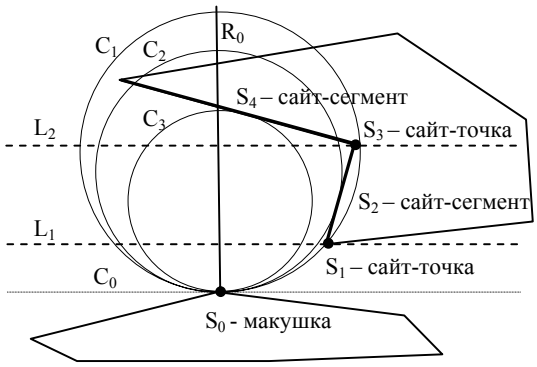

Рис. 8. Построение пузыря для макушки.

Следующими событиями, которые происходят при том же положении заметающей прямой L_1, является исключение из статуса сайта S_1 и включение вместо него сайта S_2. Окружность C_2, инцидентная сайтам S_0 и S_2, меньше окружности C_1, поэтому она занимает место пузыря макушки S_0. Далее при положении заметающей прямой L_2 в статусе происходит замена сайта S_2 сайтом S_3, которая не влечет за собой коррекции пузыря. А вот следующее событие – замена в статусе сайта S_3 сайтом S_4 – приводит к появлению новой пустой окружности C_3, которая и становится окончательным пузырем макушки S_0. Следует еще отметить, что в момент, когда заметающая прямая станет верхней касательной для окружности C_3, луч R_0 будет исключен из статуса.

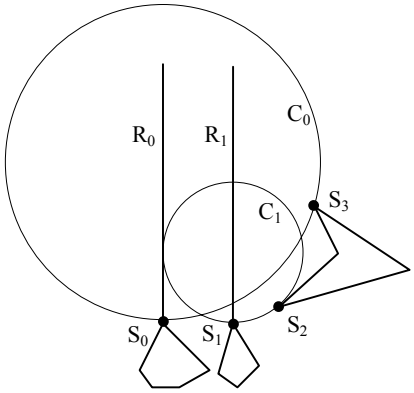

Рис.9. Контакт пузыря не с соседним сайтом.

Поскольку проверка, является ли сайт опорным для пузыря, осуществляется за конечное время $O(1)$, может сложиться впечатление, что общее время, необходимое для нахождения всех пузырей таким способом, составит $O(n \log n)$. Однако так будет не всегда. Картину портит ситуация, когда между опорным сайтом и лучом оказывается другой луч (рис.9). В этом примере пузырь C_0 для макушки S_0 является

инцидентной окружностью для сайта S_3. Но сайт S_3 никогда не окажется непосредственным соседом луча R_0 в статусе.

Возможность возникновения подобных ситуаций приводит к усложнению обработки события соседства сайта и луча. Получается, что если в статусе стоят подряд несколько лучей, то необходимо проверить на смежность этот сайт и макушки всех этих лучей. В результате может сложиться ситуация (рис.10), когда общее время, необходимое для проверки, становится равным $O(p \cdot q)$, где p –число соседних лучей в статусе, а q – число сайтов, оказавшихся с ними соседями в процессе заметания. Очевидно, что в худшем случае p и q могут иметь порядок $O(n)$, что опять приводит к общему времени всех проверок $O(n^2)$.

Конечно, случай, подобный изображенному на рис.10, явно экзотический. В практических случаях, когда количество стоящих подряд лучей в статусе не превосходит некоторого небольшого числа, общее время работы останется $O(n \log n)$. Однако для гарантированного достижения такой эффективности в любом случае алгоритм нуждается еще в одном усовершенствовании.

6.3 Заметание с поглощением пузырей

В ситуации, показанной на рис.9, контакт сайта S_3 с пузырем C_0 стал возможен из-за того, что окружность C_0 «накрыла» окружность C_1, лежащую между S_3 лучом R_0. Такое взаимное положение пузырей будем называть *доминированием*. Формально это означает, что верхняя точка пузыря C_1 (доминируемого) оказалась внутри пузыря C_0 (доминирующего).

Для того, чтобы корректно обработать ситуации, подобные этой, необходимо сначала выявить все случаи доминирования пузырей. Для этого процесс заметания, описанный выше, дополняется функцией контроля доминирования пузырей. Этот контроль осуществляется в процессе заметания и состоит в выполнении следующих правил:

1) Если два луча оказались соседними в статусе, то осуществляется проверка их на доминирование, т.е. на попадание вершины пузыря внутрь соседнего пузыря.
2) В случае, если доминирование имеет место, то планируется событие «поглощение» на момент, когда один пузырь целиком окажется внутри другого. Соответствующий момент вычисляется как ордината точки пересечения окружностей.
3) При наступлении события поглощения доминируемый пузырь (его луч) удаляется из статуса и помещается в доминирующий пузырь. Для этого в доминирующем пузыре заводится список-магазин для поглощенных пузырей, работающий по принципу «последним пришел – первым вышел».
4) В процессе заметания коррекция размеров доминирующих пузырей сопровождается проверкой, сохраняется ли доминирование над поглощенными пузырями. Если доминирования больше нет, то поглощенный пузырь возвращается в статус на свое место.
5) При выявлении контакта пузыря и сайта коррекция осуществляется только для доминирующего пузыря. Контакты сайта с поглощенными пузырями выявляются в процессе постобработки после завершения заметания.

Основная экономия времени в этом алгоритме для ситуаций типа изображенной на рис.10 достигается за счет правила 5, поскольку при соседстве сайта и луча не производится полный перебор всех остальных лучей, стоящих подряд. Общая вычислительная сложность процесса заметания остается $O(n \log n)$, поскольку число событий поглощения составляет $O(n)$, а обработка, связанная с выполнением правил 1-5, требует фиксированного времени $O(1)$.

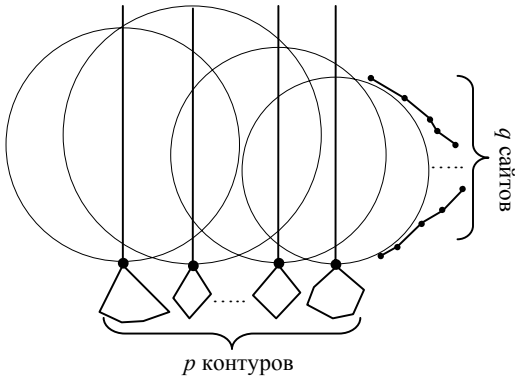

Рис.10. Соседство нескольких лучей в статусе.

Модифицированный таким образом алгоритм плоского заметания на выходе дает вложенные цепочки доминируемых пузырей. Каждая такая цепочка имеет вид доминирующего пузыря со своим списком-магазином поглощенных им доминируемых пузырей. Те пузыри, которые не были поглощены, можно рассматривать как цепочки длины 1. При этом все пузыри имеют контактные сайты, определяющие их размеры. Однако, правило 5, позволившее сэкономить время на переборе, привело к тому, что среди доминируемых пузырей могут оказаться такие, для которых контактный сайт найден неправильно. Причина этого состоит в том, что поскольку после попадания такого пузыря в список поглощенных, проверка его на контакт с сайтами не производилась. Для того, чтобы восстановить правильные контакты для доминируемых пузырей, после завершения заметания требуется дополнительный анализ списков поглощенных пузырей.

6.4 Определение опорных сайтов для поглощенных пузырей

Рассмотрим цепочку пузырей $\{C_0,C_1,\ldots,C_k\}$, полученную следующим образом. Пузырь C_0 является «крайним», т.е. его луч C_0 непосредственно соседствует с каким-то сайтом в статусе. Он доминируется пузырем C_1, тот в свою очередь доминируется пузырем C_2 и т.д. Наверху в этой последовательности находится доминирующий пузырь C_k. В процессе заметания для всех этих пузырей были получены все сайты, оказавшиеся с ними соседними. Все такие соседства прошли через очередь событий, причем в последовательности снизу-вверх. Это значит, что известна последовательность сайтов $\{S_0,S_1,\ldots,S_m\}$, которые только и могут быть контактными для пузырей из цепочки. Остается лишь сопоставить пузыри и сайты из этих множеств между собой. Для того, чтобы это было сделано эффективно, воспользуемся тем очевидным соображением, что такие контакты обладают следующим свойством монотонности.

Если пузыри C_i и C_j имеют контактные сайты S_p и S_q и $i<j$, то $p \le q$.

С учетом этого процесс установки соответствия пузырей и контактных сайтов может быть построен следующим образом.

Выберем пузырь C_r из середины цепочки, т.е. $r=\lceil k/2 \rceil$, и найдем для него контактный сайт полным перебором всех сайтов из $\{S_0,S_1,\ldots,S_m\}$. Предположим, что это сайт S_t.

Теперь можем воспользоваться тем, что для пузырей из нижней части цепочки $\{C_0,C_1,\ldots,C_{r-1}\}$ контактные сайты нужно искать среди $\{S_0,S_1,\ldots,S_t\}$, а для пузырей из $\{C_{r+1},\ldots,C_k\}$ – среди сайтов $\{S_t,\ldots,S_m\}$. Это значит, что для двух средних пузырей подмножеств $\{C_0,C_1,\ldots,C_{r-1}\}$ и $\{C_{r+1},\ldots,C_k\}$ поиск контактных сайтов потребует перебора в общей сложности тех же $\{S_0,S_1,\ldots,S_m\}$ сайтов. Очевидно, что обработка всех пузырей потребует *log(k)* шагов при числе операций на каждом шаге *O(m)*, т.е. затраты на определение опорных сайтов для всей цепочки составят *O(m log k)*.

Пусть l – номер цепочки доминирования пузырей, включающей k_l пузырей, и имеющей m_l соседних сайтов. Тогда общее время постобработки всех цепочек составит $O(\sum m_l \log k_l)$. Поскольку общее число пузырей $\sum k_l \le n$, получаем

$$\sum m_l \log k_l \le \sum m_l \log n = \log n \cdot \sum m_l.$$

Величина $\sum m_l$ представляет собой общее число событий соседства сайтов и пузырей в процессе заметания. Очевидно, что эта величина есть *O(n)*. С учетом этого получаем, что общее время на описываемый процесс составит *O(n log n)*.

7. ЗАКЛЮЧЕНИЕ

Итак, вычислительная сложность предложенного алгоритма построения дерева смежности граничных контуров ММФ составляет *O(n log n)*. Алгоритм построения ГД ММФ на основе этого дерева также имеет сложность *O(n log n)*. Сложность построения скелета ММФ из ГД составляет *O(n)*. Таким образом, предлагаемое общее решение задачи скелетизации ММФ имеет сложность *O(n log n)* для худшего случая.

Описанный алгоритм реализован почти полностью (построение дерева смежности контуров выполнено в варианте перебора всех соседних лучей в статусе без поглощения пузырей). Алгоритм проверен на реальных задачах обработки изображений ($n \sim 10^4$).

8. БИБЛИОГРАФИЯ

[1] Местецкий Л.М. Непрерывный скелет бинарного растрового изображения. Труды межд. конф. "Графикон-98". Москва, 1998.

[2] Kirkpatrick D.G. Efficient computation of continuous skeletons. In Proceedings of the 20th Annual IEEE Symposium on FOCS, 1979, 18-27.

[3] Fortune S. A sweepline algorithm for Voronoi diagrams. Algorithmica, 2 (1987), 153-174.

[4] Yap C.K. An O(n log n) algorithm for the Voronoi diagram of the set of simple curve segments. Discrete Comput. Geom., 2(1987), 365-393.

[5] Lee D.T. Medial axes transform of planar shape. IEEE Trans. Patt. Anal. Mach. Intell. PAMI-4 (1982), 363-369.

[6] Местецкий Л. М. Скелетизация многоугольной фигуры на основе обобщенной триангуляции Делоне. Программирование, 1999, № 3, с.16-31.

[7] Лагно Д., Соболев А. Модифицированные алгоритмы Форчуна и Ли скелетизации многоугольной фигуры. Труды межд. конф. "Графикон-2001". Москва, 2001.

[8] Srinivasan V., Nackman L.R., Tang J.-M., Meshkat S.N., Automatic mesh generation using the symmetric axis transform of polygonal domains, Proc. of the IEEE 80 (9) (1992) 1485–1501.

[9] Lee D.T., Schachter B.J. Two algorithms for constructing a Delaunay triangulation. Int.J.Comput.Inf.Sci.,9(1980), 219-242.

[10] Препарата Ф., Шеймос М. Вычислительная геометрия: введение. Москва, Мир, 1989.

[11] Dehne F., Klein R. “The big sweep”: On the power of the wavefront approach to Voronoi Diagrams. Algorithmica, 17(1997), 19-32.

[12] Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

Об авторе

Леонид Моисеевич Местецкий – доктор технических наук, профессор факультета вычислительной математики и кибернетики Московского Государственного Университета им. М.В. Ломоносова.

Адрес: Москва, 119899, Воробьевы горы, МГУ, 2-й учебный корпус, факультет ВМиК, кафедра математических методов прогнозирования.
E-mail: l.mest@ru.net

Skeleton of multiply-connected polygonal domain

Abstract

The problem of a continuous skeleton construction for a multiply-connected polygonal domain is considered. Polygonal domain is a closed domain, which border consists of finite number of simple polygons. The *O(n log n)* - algorithm for the worse case is offered, where n - number of polygonal domain vertexes. Proposed algorithm creates dual for Voronoi diagram of domain boundary. The solution is based at the adjacent tree of all boundary polygons created by a sweepline method. Sites and polygons called adjacent if they have common contact empty circle. The offered approach generalizes Lee skeleton algorithm from a simple polygon to the polygon with holes with the same running time.

Keywords: *multiplay-connected polygonal domain, skeleton, Delaunay graph, Lee algorithm, polygon ajacence.*

About the author

Leonid Mestetskiy is a professor at Moscow State University, Department of Mathematical methods of Forecast. His contact email is l.mest@ru.net.