



DOI: <https://doi.org/10.15688/jvolsu1.2016.5.8>

УДК 517.518.85+517.27

ББК 22.144

О ПАРАЛЛЕЛЬНОМ АЛГОРИТМЕ ПЕРЕЧИСЛЕНИЯ ТРИАНГУЛЯЦИЙ МНОГОУГОЛЬНИКА НА ПЛОСКОСТИ

Владимир Валентинович Попов

Кандидат физико-математических наук, доцент кафедры компьютерных наук и экспериментальной математики,

Волгоградский государственный университет

popov_v_v@ Rambler.ru, knem03@mail.ru

просп. Университетский, 100, 400062 г. Волгоград, Российская Федерация

Аннотация. В статье описывается параллельный алгоритм перечисления всех триангуляций многоугольника на плоскости. Дается оценка необходимой для реализации алгоритма памяти. Обсуждается быстрое действие алгоритма и возможность его применения для компактной записи списка всех триангуляций.

Ключевые слова: триангуляция, число триангуляций, дерево триангуляций, оценка объема памяти, число Каталана, выпуклая оболочка.

1. Задача построения всех триангуляций многоугольника на плоскости

Вопрос о построении триангуляций множества точек на плоскости рассматривался многими авторами (см., например, [5–8]). В работе [1] предложен алгоритм построения всех триангуляций конечного множества на плоскости. Соответствующий алгоритм для трехмерного пространства описан в [2]. В работе [8] изучается граф, вершинами которого являются триангуляции конечного набора точек $P \subset \mathbf{R}^2$, причем две триангуляции связаны ребром в том и только том случае, когда найдутся четыре точки множества P , являющиеся вершинами выпуклого четырехугольника, такие, что в одной триангуляции этот четырехугольник разбит на два треугольника одной диагональю, а в другой триангуляции — другой диагональю (все остальные треугольники в этих триангуляциях одинаковы). Переход от одной триангуляции к другой в этом случае называется перестроением (или флипом, flipping). В работе [3] строится другое дерево, вершинами которого являются триангуляции (а также другие прямолинейные геометрические графы) с вершинами в заданном конечном подмножестве плоскости.

В данной статье рассматривается следующая задача.

На плоскости дан замкнутый многоугольник F и конечное множество точек $P = \{p_1, p_2, \dots, p_N\}$, включающее все вершины этого многоугольника. Необходимо перечислить все триангуляции многоугольника F относительно набора точек P .

Ниже предлагается параллельный алгоритм решения этой задачи. Часть результатов объявлена в [4].

Под многоугольником понимается область плоскости, ограниченная замкнутой несампересекающейся ломаной. Триангуляция T многоугольника F относительно набора точек P — это набор треугольников S_1, S_2, \dots, S_m , удовлетворяющий следующим условиям:

- (1) Каждая точка $p_i \in P$ является вершиной хотя бы одного из треугольников S_j .
- (2) Вершины любого треугольника S_j лежат во множестве P .
- (3) Если $i \neq j$, то треугольники S_i и S_j или не пересекаются, или имеют только общую вершину, или только общее ребро.
- (4) Объединение треугольников S_1, S_2, \dots, S_m совпадает с многоугольником F .

Через $Tr(F, P)$ будем обозначать множество всех триангуляций набора точек P , удовлетворяющих условиям (1)–(4). Таким образом, необходимо перечислить все триангуляции $T \in Tr(F, P)$. При построении параллельного алгоритма решения этой задачи используется понятие перегородки.

Пусть l — прямая, пересекающая многоугольник F и не содержащая точек множества P . Назовем перегородкой, определяемой тройкой F, P, l , набор треугольников Π , который удовлетворяет таким условиям:

- (a) Вершины любого треугольника $S \in \Pi$ лежат во множестве P .
- (b) Различные треугольники $S, S' \in \Pi$ или не пересекаются, или пересекаются по общей вершине или по общему ребру.
- (c) Объединение всех треугольников $S \in \Pi$ содержит множество $F \cap l$ и само содержится в F .

На рисунке 1 слева изображен многоугольник F , множество P и прямая l , в центре — одна из триангуляций $T \in Tr(F, P)$, а справа — соответствующая этой триангуляции перегородка Π (треугольники, составляющие перегородку, изображены серым цветом).

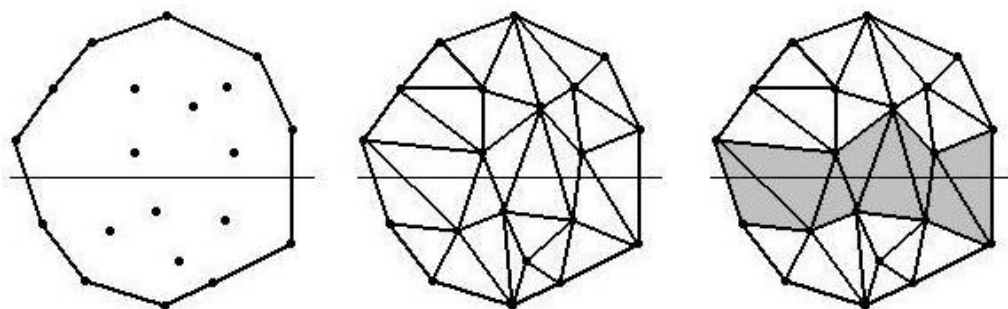


Рис. 1. Многоугольник F , множество P и прямая l (слева), одна из триангуляций (в центре) и соответствующая этой триангуляции перегородка Π (справа)

Как уже указывалось выше, в [1] описан алгоритм перечисления всех триангуляций конечного множества на плоскости. Опишем более компактный алгоритм, который можно применять не только к многоугольникам, но и к объединениям конечного числа многоугольников. Последнее важно для построения параллельного алгоритма.

2. Последовательный алгоритм построения триангуляций многоугольника F относительно набора точек P

На каждом шаге алгоритма будет определен набор треугольников

$$S_1, S_2, \dots, S_{k-1}, \quad (1)$$

который можно достроить до некоторой триангуляции многоугольника F относительно набора точек P . Через a_i , b_i и c_i будем обозначать номера точек множества P , являющихся вершинами треугольника S_i , $i = 1, 2, \dots, k-1$.

Будем использовать также следующие обозначения:

- (i, j) — ребро (прямолинейный отрезок), соединяющее вершины множества P с номерами i и j . Ребра считаем неориентированными, поэтому ребро (i, j) совпадает с ребром (j, i) ;
- k — номер строящегося треугольника;
- m — число треугольников в любой триангуляции множества P . Это число определяется после того, как построена первая триангуляция;
- ktr — число всех триангуляций множества P относительно многоугольника F , то есть $ktr = |Tr(F, P)|$. В начале работы алгоритма $ktr = 0$.

Функция $Df(i, j)$ позволяет узнать — является ли (i, j) стороной (граничным отрезком) многоугольника F или нет. В первом случае $Df(i, j)$ равно 1, а во втором 0. Если (i, j) — некоторое ребро, то будем называть его кратностью число треугольников в последовательности (1), для которых это ребро является стороной. Если кратность ребра (i, j) равна 1 и $Df(i, j) = 0$, то будем называть это ребро граничным. Если граничное ребро появилось при добавлении треугольника S_k , то оно будет запоминаться как ребро (u_k, v_k) . Если же при добавлении треугольника S_k граничное ребро не появилось, то переменной u_k присваивается значение 0. Если $e = (u_k, v_k)$ — граничное ребро (возникшее при добавлении треугольника S_k) и в дальнейшем к последовательности (1) был добавлен треугольник S_l , $l > k$, для которого ребро e является стороной, то e перестанет быть граничным. Этот факт фиксируется путем присвоения переменной r_k значения l . Для граничного ребра $r_k = 0$.

На каждом шаге работы алгоритма будет определено базовое ребро (i_1, i_2) , к которому будет подыскиваться число j такое, что треугольник с номерами вершин i_1 , i_2 и j можно добавить к последовательности (1) в качестве S_k . Это будет осуществляться с помощью функции $f(i_1, i_2, i_3)$, которая и возвращает нужное j с дополнительным условием $i_3 < j \leq N$. Если такого j не найдено, то возвращается значение 0.

Функция $g(k)$ находит наименьшее целое число j , для которого $1 \leq j < k$ и (u_j, v_j) является граничным ребром для последовательности (1). Если такого j не найдено, то $g(k)$ возвращает значение 0.

Алгоритм построения списка триангуляций

- ДАНО: многоугольник F на плоскости (не обязательно выпуклый);
 множество P из N точек плоскости, лежащее в F и содержащее все вершины многоугольника F ;
 базовое ребро (i_1, i_2) , являющееся стороной многоугольника F .
- ПОЛУЧИТЬ: список всех триангуляций многоугольника F относительно множества P .

- 1) Полагаем $k = 1$, $m = 0$ и $ktr = 0$. При всех $i, j = 1, 2, \dots, N$ кратность ребра (i, j) полагаем равной 0. Полагаем также $a_1 = i_1$ и $b_1 = i_2$.

- 2) Полагаем $c_k = 0$.
- 3) Полагаем $j = f(a_k, b_k, c_k)$. Если $j \leq 0$, то идем на пункт 8.
- 4) Полагаем $c_k = j$. Кратности ребер треугольника S_k (номера вершин которого a_k , b_k и c_k) увеличиваем на 1.
- 5) Если $m > 0$ и $k \geq m$, то идем на пункт 10.
- 6) Полагаем $u_k = 0$.
- 7) а) Если $Df(a_k, c_k) = 0$ и кратность ребра (a_k, c_k) равна 1, то полагаем $a_{k+1} = a_k$ и $b_{k+1} = c_k$. Если, дополнительно, $Df(b_k, c_k) = 0$ и кратность ребра (b_k, c_k) равна 1, то полагаем $u_k = b_k$, $v_k = c_k$ и $r_k = 0$.
Увеличиваем k на 1 и идем к пункту 2.
б) Если $Df(b_k, c_k) = 0$ и кратность ребра (b_k, c_k) равна 1, то полагаем $a_{k+1} = b_k$ и $b_{k+1} = c_k$. Увеличиваем k на 1 и идем к пункту 2.
в) Увеличиваем k на 1 (и переходим на пункт 8).
- 8) Полагаем $j = g(k)$. Если $j \leq 0$, то идем на пункт 10.
- 9) Полагаем $a_k = u[j]$, $b_k = v[j]$, $c_k = 0$, $r[j] = k$ и идем на пункт 3.
- 10) Если $ktr = 0$, то полагаем $m = k - 1$.
Увеличиваем ktr на 1. Записываем список троек номеров вершин треугольников из последовательности (1), образующих полученную триангуляцию, в общий список триангуляций (под номером ktr).
- 11) Уменьшаем на 1 кратности ребер треугольников S_m и S_{m-1} и полагаем $k = m - 1$ (тем самым эти треугольники удаляются из списка (1)).
- 12) При каждом $i = 1, 2, \dots, k$, при котором $r[i] \geq k$, полагаем $r[i] = 0$.
- 13) Полагаем $j = f(a_k, b_k, c_k)$. Если $j > 0$, то идем к пункту 4.
- 14) Если $k = 1$, то завершаем работу алгоритма.
- 15) Уменьшаем k на 1. Уменьшаем на 1 кратности ребер треугольника S_k .
Идем на пункт 12.

При компьютерной реализации описанного алгоритма необходимо хранить в памяти 8 одномерных массивов и 2 двумерных. Поэтому требуемая память — $O(N^2)$. Однако можно уменьшить эту оценку до $O(N)$ (за счет некоторого снижения быстродействия).

3. Параллельный алгоритм построения триангуляций многоугольника

Пусть F — многоугольник на плоскости; $P \subset F$ — конечное множество, содержащее все вершины F , а l — прямая, не содержащая точек P и пересекающая F . Пусть $T \in Tr(F, P)$ — триангуляция многоугольника F относительно множества P . Обозначим через Π множество треугольников $S \in T$, пересекающих l , а через $\tilde{\Pi}$ — объединение этих треугольников. Тогда Π является перегородкой, определяемой тройкой F , P и l . Прямая l разбивает плоскость на две полуплоскости H_1 и H_2 . При $i = 1, 2$ обозначим через T_i множество треугольников $S \in T$, лежащих в H_i и не входящих в перегородку Π , а через F_i — объединение треугольников, входящих в T_i . Тогда T_i — триангуляция F_i относительно множества $P \cap F_i$. Таким образом, если задана некоторая триангуляция $T \in Tr(F, P)$, то можно однозначно определить перегородку Π и триангуляции T_1 , T_2 многоугольников F_1 и F_2 относительно множеств $F_1 \cap P$ и $F_2 \cap P$ соответственно. Наоборот, по триангуляциям T_1 и T_2 однозначно восстанавливается триангуляция T —

достаточно взять все треугольники, входящие в T_1 , T_2 и Π .

Следовательно, для построения всех триангуляций $T \in Tr(F, P)$ можно поступить так: выбрать прямую l , пересекающую F и не пересекающую P , а затем перебрать все перегородки, определяемые тройкой F, P, l , и для каждой такой перегородки Π построить все триангуляции многоугольников F_1 и F_2 (определяемых по F и Π) относительно множеств $F_1 \cap P$ и $F_2 \cap P$ соответственно. Триангуляции многоугольников F_1, F_2 можно перечислять независимо друг от друга по описанному выше последовательному алгоритму. Осталось описать алгоритм построения всех перегородок, определяемых тройкой F, P и l . Сделаем это.

Заметим, что прямую l следует выбирать так, чтобы по обе стороны от нее находилось приблизительно одинаковое число точек множества P . Кроме того, расчеты упрощаются, если эта прямая — горизонтальная или вертикальная. Остановимся на случае горизонтальной прямой.

На каждом шаге алгоритма будет определен набор треугольников вида (1), который можно достроить до перегородки, определяемой многоугольником F , множеством P и горизонтальной прямой l . Через kpr будем обозначать число всех перегородок. В начале работы алгоритма $kpr = 0$.

Многоугольник F может быть невыпуклым и потому может пересекать прямую l по нескольким отрезкам. Нам потребуется список всех ребер многоугольника F , пересекающих прямую l , отсортированный по возрастанию x -координаты точки пересечения ребра с прямой l :

$$e_1, e'_1, e_2, e'_2, \dots, e_q, e'_q. \quad (2)$$

Здесь q — число отрезков, на которые распадается пересечение $F \cap l$. Через t будем обозначать порядковый номер такого отрезка. Если точка движется слева направо по прямой l , то после пересечения ребра e_t она попадает в многоугольник F , а после пересечения ребра e'_t выходит из него. Остальные обозначения — такие же, как в предыдущем алгоритме.

Алгоритм построения списка горизонтальных перегородок

ДАНО: многоугольник F на плоскости (не обязательно выпуклый);
множество P из N точек плоскости, лежащее в F и содержащее все вершины многоугольника F ;
горизонтальная прямая l , пересекающая многоугольник F и не содержащая точек множества P .

ПОЛУЧИТЬ: список всех перегородок, определяемых тройкой F, P и l .

- 1) Составляем список ребер (2).
- 2) Полагаем $k = 1$, $t = 1$ и $kpr = 0$.
- 3) Полагаем $a_k = i_1$ и $b_k = i_2$, где i_1, i_2 — номера вершин ребра e_t .
- 4) Полагаем $c_k = 0$.
- 5) Полагаем $j = f(a_k, b_k, c_k)$.
- 6) Полагаем $c_k = j$. Если ребро (a_k, c_k) пересекает прямую l , то полагаем $a_{k+1} = a_k$ и $b_{k+1} = c_k$. В противном случае полагаем $a_{k+1} = c_k$ и $b_{k+1} = b_k$.
- 7) Если ребро (a_{k+1}, b_{k+1}) не совпадает с ребром e'_t , то увеличиваем k на 1 и идем к пункту 4.
- 8) Если $t < q$, то t увеличиваем на 1, k увеличиваем на 1 и идем к пункту 3.

- 9) Увеличиваем kpr на 1. Записываем список троек номеров вершин треугольников, составляющих полученную перегородку, в общий список перегородок (под номером kpr).
- 10) Полагаем $j = f(a_k, b_k, c_k)$. Если $j > 0$, то идем к пункту 6.
- 11) Если $k = 1$, то завершаем работу алгоритма.
- 12) Если ребро (a_k, b_k) совпадает с ребром e_t , то уменьшаем t на 1, уменьшаем k на 1 и идем к пункту 10.
- 13) Уменьшаем k на 1 и идем к пункту 10.

При реализации описанного алгоритма необходимо хранить в памяти только 3 одномерных массива длины N каждый (и список ребер (2), размер которого обычно существенно меньше N). Поэтому требуемая память — $O(N)$.

4. Сравнение параллельного и последовательного алгоритмов

Рассмотрим вопрос об объеме памяти, необходимой для хранения списка всех триангуляций $T \in Tr(F, P)$ многоугольника F относительно конечного множества P . Прежде всего надо описать F и P , для чего можно, например, указать число N элементов множества P , а затем выпisać набор координат x_i, y_i точек этого множества, после чего привести список номеров вершин многоугольника F . Информацию же о триангуляциях можно записывать различными способами. Один из них таков. Для каждой триангуляции будем хранить список номеров вершин треугольников, составляющих эту триангуляцию. Тогда список всех триангуляций будет содержать $ktr \cdot (3 \cdot N)$ чисел в промежутке от 1 до N (напомним, что $ktr = |Tr(F, P)|$ — число всех триангуляций многоугольника F относительно множества P , а m — число треугольников в любой такой триангуляции).

Другой способ заключается в следующем. Пусть Π — некоторая перегородка, определяемая многоугольником F , множеством P и некоторой (заранее выбранной) прямой l . Тогда Π однозначно определяет две части F_1 и F_2 многоугольника F (см. начало предыдущего раздела). Нам потребуется разделяющий символ, который обозначим R (за R можно принять, например, число $N + 1$). Фрагмент записи, кодирующий список всех триангуляций $T \in Tr(F, P)$, до которых можно достроить перегородку Π , имеет такой вид:

- 1) Список номеров вершин треугольников, составляющих перегородку Π , а затем разделяющий символ R .
- 2) Число m_1 треугольников в любой триангуляции первой части F_1 многоугольника F .
- 3) Список триангуляций первой части F_1 многоугольника F . Для каждой триангуляции записывается список номеров вершин треугольников, составляющих эту триангуляцию.
- 4) Символ R .
- 5) Число m_2 треугольников в любой триангуляции второй части F_2 многоугольника F .
- 6) Список триангуляций второй части F_2 многоугольника F (как в пункте 3).
- 7) Символ R .

Пусть при $i = 1, 2$ часть F_i многоугольника F имеет t_i триангуляций. Пусть также перегородка Π состоит из m_Π треугольников. Тогда имеется $t_1 \cdot t_2$ триангуляций $T \in Tr(F, P)$, до которых можно достроить перегородку Π . При этом любая триангуляция состоит из $m = m_1 + m_\Pi + m_2$ треугольников. Поэтому для записи этих триангуляций по первому способу необходимо $V_1 = (t_1 \cdot t_2) \cdot (3m) = 3t_1t_2m$ чисел в диапазоне от 1 до N . Вторым же способом требуется

$$V_2 = (3m_\Pi + 1) + (1 + t_1 \cdot (3m_1)) + 1 + (1 + t_2 \cdot (3m_2)) + 1 = 3m_\Pi + 3t_1m_1 + 3t_2m_2 + 5$$

чисел. Разность $V_1 - V_2$ можно записать в виде

$$V_1 - V_2 = 3m_\Pi(t_1t_2 - 1) + 3m_1t_1(t_2 - 1) + 3m_2t_2(t_1 - 1) - 5.$$

Если $t_1t_2 \geq 1$ и $m_\Pi \geq 2$, то $V_1 - V_2 > 0$, поэтому вторым способом записи списка триангуляций требуется меньше памяти, чем первым. Если же $t_1 = t_2 = 1$, то вторым способом требуется запомнить на 5 чисел больше, чем первым. Однако это бывает только тогда, когда части F_1 и F_2 многоугольника F допускают только одну триангуляцию каждая. Подобные ситуации случаются, но достаточно редко (см. ниже примеры 2 и 3). В подавляющем большинстве случаев (когда $t_1t_2 \geq 2$) V_2 меньше V_1 :

Утверждение 1. Пусть перегородка Π и целое число $n \geq 4$ таковы, что каждая из частей F_1, F_2 многоугольника F содержит по n точек множества P , которые являются вершинами выпуклого n -угольника. Тогда $\frac{V_2}{V_1} \leq \frac{1}{C_{n-2}} + \frac{5}{3(2n-4)(C_{n-2})^2}$, где $C_k = \frac{1}{k+1}C_{2k}^k$ — число Каталана (см., например, [9]). В частности, при $n = 4$ верно $\frac{V_2}{V_1} \leq \frac{5}{8}$, а при $n = 5$ выполнено $\frac{V_2}{V_1} \leq \frac{1}{4}$.

Доказательство. Так как части F_1 и F_2 многоугольника F не имеют общих точек, получаем, что $N = |P| \geq 2n$, поэтому число треугольников m в любой триангуляции $T \in Tr(F, P)$ не меньше $2n - 4$. Так как выпуклый n -угольник допускает ровно C_{n-2} триангуляций (см.: [8; 9]), несложно заключить, что $t_i \geq C_{n-2}$ при $i = 1, 2$. Пусть для определенности $1 \leq t_1 \leq t_2$. Тогда $V_2 \leq 3t_2(m_\Pi + m_1 + m_2) + 5 = 3t_2m + 5$. Поэтому

$$\frac{V_2}{V_1} \leq \frac{3t_2m + 5}{3mt_1t_2} = \frac{1}{t_1} + \frac{5}{3mt_1t_2} \leq \frac{1}{C_{n-2}} + \frac{5}{3(2n-4)(C_{n-2})^2}.$$

Остановимся теперь на вопросе о сравнении быстродействия последовательного и параллельного алгоритмов. Чтобы перечислить все триангуляции $T \in Tr(F, P)$, до которых можно достроить перегородку Π , параллельный алгоритм должен построить t_1 и t_2 триангуляций частей F_1 и F_2 многоугольника F соответственно, а перед этим — саму перегородку Π . Таким образом, надо найти $1 + t_1 + t_2$ триангуляций отдельных частей многоугольника F . Последовательный же алгоритм должен строить $t_1 \cdot t_2$ триангуляций всего многоугольника F . Если $t_1, t_2 \geq 2$, то первое число меньше второго, то есть параллельный алгоритм работает быстрее последовательного.

Если части F_1 и F_2 , определяемые горизонтальной перегородкой Π , достаточно велики, то каждую из них можно разбить на две части вертикальной перегородкой и перечислить триангуляции получившихся четырех частей многоугольника F . Ниже (в примере 1) приведены результаты численных экспериментов.

Приведем теперь результаты, полученные с помощью описанных алгоритмов.

Пример 1. Рассмотрим на плоскости множество из $n \cdot m$ точек:

$$L_{m,n} = \{(i, j) : i = 1, 2, \dots, n, j = 1, 2, \dots, m\},$$

где $n, m \geq 1$ — целые числа. Пусть F — выпуклая оболочка множества $P = L_{m,n}$. Число триангуляций в $Tr(F, P)$ при некоторых n и m подсчитывалось тремя способами: с помощью последовательного алгоритма (алгоритм 1), по алгоритму с одной горизонтальной перегородкой (алгоритм 2), а также по алгоритму с одной горизонтальной и двумя вертикальными перегородками (алгоритм 3). Получены следующие результаты:

n	3	4	4	4
m	6	4	5	6
Алгоритм 1	12	6	3 963	Более $2 \cdot 10^5$
Алгоритм 2	7	3	85	3 511
Алгоритм 3	2	2	40	1 231
Число триангуляций	182 132	46 456	2 822 648	182 881 520

В таблице указано время работы компьютерной программы (в секундах).

Рассмотрим теперь примеры перегородок, для которых числа t_1, t_2 малы и потому параллельный алгоритм не дает существенного преимущества по сравнению с последовательным (примеры 2 и 3), или же, наоборот, эти числа велики и преимущество очень существенно (пример 4).

Пример 2. Пусть F и P — такие, как в примере 1. Выберем прямую l с уравнением $y = 5/2$. При $i = 1, 2, \dots, n - 1$ проведем отрезки, соединяющие точки с координатами $(i, 1)$ и $(i + 1, m)$. Эти отрезки можно единственным образом дополнить другими отрезками так, чтобы получилась триангуляция многоугольника F относительно множества P . Соответствующая этой триангуляции перегородка изображена (при $n = 4$ и $m = 5$) на рисунке 2 слева (входящие в нее треугольники заштрихованы). Для этой перегородки $t_1 = t_2 = 1$.

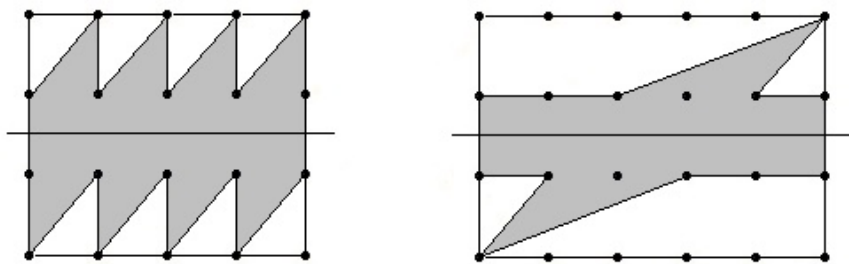


Рис. 2. Многоугольники из примера 2 (слева) и примера 3 (справа), а также перегородки, соответствующие описанным в этих примерах триангуляциям

Пример 3. Пусть снова F и P — такие, как в примере 1, причем $m = 4$, а n имеет вид $n = 6k$ или $n = 6k + 2$, где $k \geq 1$ — целое. Пусть l — прямая с уравнением $y = 5/2$. Пусть также I — отрезок, соединяющий точки q_1 и q_2 с координатами $(1, 1)$ и $(n, 4)$ соответственно. Пусть \mathcal{I}_1 — множество отрезков с концами q_1 и p , где точка p принадлежит множеству P и лежит ниже отрезка I . Пусть также \mathcal{I}_2 — множество отрезков с концами q_2 и p , где $p \in P$ и точка p лежит выше отрезка I . Тогда множество

отрезков $\mathcal{I}_1 \cup \mathcal{I}_2 \cup \{I\}$ можно единственным образом дополнить другими отрезками так, чтобы получилась триангуляция многоугольника F относительно множества P . Пусть Π — соответствующая перегородка и $\tilde{\Pi}$ — объединение треугольников из Π . Если начать обход границы многоугольника $\tilde{\Pi}$ (против часовой стрелки) из вершины q_1 , то встретятся вершины с координатами $(\frac{n}{2} + 1, 2)$, $(n, 2)$, $(n, 3)$, $(\lceil \frac{2n+1}{3} \rceil, 3)$, $q_2 = (n, 4)$, $(\lceil \frac{n+1}{2} \rceil, 3)$, $(1, 3)$, $(1, 2)$, $(\lceil \frac{n+2}{3} \rceil, 2)$, после чего мы вернемся в вершину q_1 . Соответствующая этой триангуляции перегородка изображена (при $n = 6$) на рисунке 2 справа (входящие в нее треугольники заштрихованы). Для этой перегородки $t_1 = t_2 = 1$.

Пример 4. Пусть F и P — такие, как в примере 1, причем $n = 5$ и $m = 6$. Пусть Π — перегородка, заполняющая горизонтальную полосу $3 \leq y \leq 4$. Тогда части F_1 и F_2 многоугольника F будут иметь по 12 170 триангуляций каждая и отношение $\frac{V_2}{V_1}$ будет меньше числа 0,0001.

Пример 5. Пусть $m, n \geq 1$ — целые числа, а F и P — многоугольник и множество из примера 1 для этих m и n . Пусть \tilde{V}_1 и \tilde{V}_2 — количество целых чисел (в границах от 1 до $N + 1$, где $N = m \cdot n$) при первом и втором способе записи списка всех триангуляций многоугольника F относительно множества P . В следующей таблице указаны приближенные значения \tilde{V}_1 и \tilde{V}_2 (а также их отношение) при некоторых m и n :

n	3	4	4	4
m	6	4	5	6
Число триангуляций	182 132	46 456	2 822 648	182 881 520
Число V_1	$10,93 \cdot 10^6$	$2,51 \cdot 10^6$	$203,23 \cdot 10^6$	$164,6 \cdot 10^8$
Число V_2	$0,82 \cdot 10^6$	$0,29 \cdot 10^6$	$9,18 \cdot 10^6$	$2,67 \cdot 10^8$
Отношение $\frac{\tilde{V}_2}{\tilde{V}_1}$	0,075	0,115	0,045	0,016

Пример 6. При перечислении триангуляций можно учитывать только те из них, в которых нет треугольников с длинными сторонами или же треугольников с маленькими углами (задача фильтрации). Укажем число триангуляций многоугольника из примера 1, для которых длины всех сторон треугольников не больше заданного числа L :

n	3	4	4	4
m	6	4	5	6
$L = 2$	1 024	512	4 096	32 768
$L = 3$	84 592	28 608	1 168 516	47 733 448
$L = 4$	163 330	46 456	2 700 598	159 422 068

СПИСОК ЛИТЕРАТУРЫ

1. Клячин, В. А. Метод цепей для организации хранения многомерных триангуляций / В. А. Клячин, В. В. Попов // Вестник Волгоградского государственного университета. Серия 1, Математика. Физика. — 2013. — № 2 (19). — С. 71–79.
2. Попов, В. В. Об алгоритме перечисления триангуляций / В. В. Попов // Вестник Волгоградского государственного университета. Серия 1, Математика. Физика. — 2014. — № 5 (24). — С. 40–45.
3. Попов, В. В. Об алгоритме перечисления остовов связного графа / В. В. Попов // Вестник Волгоградского государственного университета. Серия 1, Математика. Физика. — 2015. — № 2 (27). — С. 6–16.

4. Попов, В. В. О параллельном алгоритме перечисления триангуляций множества точек плоскости / В. В. Попов // Геометрический анализ и его приложения : материалы III Международной школы-конференции. — Волгоград : Изд-во ВолГУ, 2016. — С. 168–170.
5. Препарата, Ф. Вычислительная геометрия: Введение / Ф. Препарата, М. Шеймос. — М. : Мир, 1989. — 478 с.
6. Скворцов, А. В. Триангуляция Делоне и ее применение / А. В. Скворцов. — Томск : Изд-во Том. ун-та, 2002. — 128 с.
7. On the Number of Plane Geometrical Graphs / O. Aichholzer, T. Hackl, B. Vogtenhuber, C. Huemer, F. Hurtado, H. Krasser // *Graphs and Combinatorics*. — 2007. — Vol. 23. — P. 67–84. — DOI: 10.1007/s00373-007-0704-5.
8. Hurtado, F. Graph of triangulations of a convex polygon and tree of triangulations / F. Hurtado, M. Noy // *Computational Geometry*. — 1999. — № 13. — P. 179–188.
9. Polia, G. On picture-writing / G. Polia // *The American Mathematical Monthly*. — 1956. — № 63. — P. 689–697.

REFERENCES

1. Klyachin V.A., Popov V.V. Metod tsepey dlya organizatsii khraneniya mnogomernykh triangulyatsiy [Chanes Method for Storage of Multidimensional Triangulation]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 1, Matematika. Fizika* [Science Journal of Volgograd State University. Mathematics. Physics], 2013, no. 2 (19), pp. 71-79.
2. Popov V.V. Ob algoritme perechisleniya triangulyatsiy [On Algorithm of Numbering of Triangulations]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 1, Matematika. Fizika* [Science Journal of Volgograd State University. Mathematics. Physics], 2014, no. 5 (24), pp. 40-45.
3. Popov V.V. Ob algoritme perechisleniya ostovov svyaznogo grafa [On Algorithm of Numbering the Spanning Trees in a Connected Graph]. *Vestnik Volgogradskogo gosudarstvennogo universiteta. Seriya 1, Matematika. Fizika* [Science Journal of Volgograd State University. Mathematics. Physics], 2015, no. 2 (27), pp. 6-16.
4. Popov V.V. O parallelnom algoritme perechisleniya triangulyatsiy mnozhestva toчек ploskosti [On Parallel Algorithm of Numbering of the Triangulations of a Finite Subset of the Plane]. *Geometricheskii analiz i ego prilozheniya: materialy III Mezhdunarodnoy shkoly-konferentsii*. Volgograd, Izd-vo VolGU, 2016, pp. 168-170.
5. Preparata F., Shamos M. *Vychislitel'naya geometriya: Vvedenie* [Computational Geometry: An Introduction]. Moscow, Mir Publ., 1989. 478 p.
6. Skvortsov A.V. *Triangulyatsiya Delone i ee primenenie* [Delaunay Triangulation and Its Application]. Tomsk, Izd-vo Tom. un-ta, 2002. 128 p.
7. Aichholzer O., Hackl T., Vogtenhuber B., Huemer C., Hurtado F., Krasser H. On the Number of Plane Geometrical Graphs. *Graphs and Combinatorics*, 2007, vol. 23, pp. 67-84. DOI: 10.1007/s00373-007-0704-5.
8. Hurtado F., Noy M. Graph of Triangulations of a Convex Polygon and Tree of Triangulations. *Computational Geometry*, 1999, no. 13, pp. 179-188.
9. Polia G. On Picture-Writing. *The American Mathematical Monthly*, 1956, no. 63, pp. 689-697.

ON THE PARALLEL ALGORITHM OF NUMBERING TRIANGULATIONS OF THE POLYGON IN THE PLANE

Vladimir Valentinovich Popov

Candidate of Physical and Mathematical Sciences, Associate Professor,
Department of Computer Sciences and Experimental Mathematics,
Volgograd State University

popov_v_v@rambler.ru, knem03@mail.ru
 Prosp. Universitetsky, 100, 400062 Volgograd, Russian Federation

Abstract. The task of constructing a triangulation of a finite subset of the plane is considered by many authors (see, for example, [5–8]). In [1] it was offered the algorithm for constructing all the triangulations of a finite set on the plane. The corresponding algorithm for three-dimensional space is described in [2]. In this paper we consider the following problem:

Let P be a (closed) polygon in the plane and $P = \{p_1, p_2, \dots, p_N\}$ is a finite subset of \mathbf{R}^2 , including all the vertices of the polygon F . It is necessary to numbering all the triangulations of the polygon F relative to a set P .

In this paper we propose a parallel algorithm for solving this problem. Some of the results are announced in [4].

The triangulation T of a polygon F relative to a set P is a collection rS_1, S_2, \dots, S_m of triangles, satisfying the following conditions:

- (1) Each point $p_i \in P$ is a vertex of at least one of the triangle S_j .
- (2) The vertices of any triangle S_j lie in the set P .
- (3) If $i \neq j$, then the triangles S_i and S_j either have no common points, or have (only) a common vertex, or have (only) a common edge.
- (4) The union of triangles S_1, S_2, \dots, S_m coincides with the polygon F .

Let $Tr(F, P)$ be the set of all triangulations, satisfying conditions (1)–(4). Thus, it is necessary to list all the triangulation $T \in Tr(F, P)$. To describe the parallel algorithm for solving this problem we need to use the notion of septum. Let l be such a straight lines in the plane, which intersect F and disjoint with P . We call the septum defined by the triple F, P, l , a set Π of triangles, which satisfies the following conditions:

- (a) All vertices of any triangle $S \in \Pi$ lie in the set P .
- (b) If $S, S' \in \Pi$ and $S \neq S'$, then $S \cap S' = \emptyset$, or S and S' have (only) a common vertex or have (only) a common edge.
- (c) The union of all triangles $S \in \Pi$ contains the set $F \cap l$ and itself is contained in F .

Let $T \in Tr(F, P)$ is a triangulation of the polygon F relative to the set P and l is a straight line, $l \cap F \neq \emptyset$ and $l \cap P = \emptyset$. Then $\Pi = \{S \in T : S \cap l \neq \emptyset\}$ is a septum defined by the triple F, P, l . The line l divides the plane \mathbf{R}^2 into two half-planes H_1 and H_2 . Let, for $i = 1, 2$, T_i be the set of all triangles $S \in T$, that lies in H_i , and let F_i be the union of triangles $S \in T_i$. Then T_i is a triangulation of polygon F_i relative to the set $P \cap F_i$. Thus, if we are given a triangulation $T \in Tr(F, P)$, it is possible to uniquely identify the septum Π and the triangulations T_1, T_2 . On the contrary, if we have Π, T_1 and T_2 , then the family of all triangles from Π, T_1 and T_2 gives us triangulation T . In this way, we can list all the triangulation $T \in Tr(F, P)$: we need only to renumber all the the septa Π , defined by the triple F, P, l , and for each such Π to renumber the triangulations $T_1 \in Tr(F_1, P \cap F_1)$ and $T_2 \in Tr(F_2, P \cap F_2)$.

We now present some of the results obtained by the described methods.

Example 1. Consider the set $L_{m,n}$ on the plane, consisting of $n \cdot m$ points:

$$L_{m,n} = \{(i, j) : i = 1, 2, \dots, n, j = 1, 2, \dots, m\},$$

where $n, m \geq 1$ — integers.

Let F be the convex hull of $P = L_{m,n}$. Then the number of triangulations in $Tr(F, P)$ for some n and m we calculated in three ways:

1) Using algorithm from [1] for constructing all triangulations $T \in Tr(F, P)$ (Algorithm 1).

2) Using algorithm we describe above, where l is a horizontal line (Algorithm 2).

3) Using the modification of the algorithm from 2) in which we need horizontal line and vertical line (Algorithm 3).

n	3	4	4	4
m	6	4	5	6
Algorithm 1	12	6	3 963	More then $2 \cdot 10^5$
Algorithm 2	7	3	85	3 511
Algorithm 3	2	2	40	1 231
The number of triangulations	182 132	46 456	2 822 648	182 881 520

The table shows the operating time of the computer program (in seconds).

Example 6. The following problem arises often: for given F and P we need to numbering only those triangulation $T \in Tr(F, P)$, that satisfy certain additional conditions (Filtering problem). The following table shows (for some m, n and $L \in \mathbf{R}$) the number of such a triangulation T from Example 1, that for any triangle $S \in T$ the length of the sides of S does not exceed L .

n	3	4	4	4
m	6	4	5	6
$L = 2$	1 024	512	4 096	32 768
$L = 3$	84 592	28 608	1 168 516	47 733 448
$L = 4$	163 330	46 456	2 700 598	159 422 068

Key words: triangulation, the number of triangulations, the tree of triangulations, memory volume estimate, Catalan number, convex hull.