DOI: http://dx.doi.org/10.15688/jvolsu1.2015.2.1

УДК 517.518.85+517.27
ББК 22.144

ОБ АЛГОРИТМЕ ПЕРЕЧИСЛЕНИЯ ОСТОВОВ СВЯЗНОГО ГРАФА

Попов Владимир Валентинович

Кандидат физико-математических наук, доцент кафедры компьютерных наук и экспериментальной математики,
Волгоградский государственный университет
popov_v_v@rambler.ru, kiem@volsu.ru
просп. Университетский, 100, 400062 г. Волгоград, Российская Федерация

Аннотация. Описывается алгоритм перечисления всех остовных деревьев (остовов) связного графа с конечным числом вершин. Приводятся результаты работы компьютерной программы, составленной по этому алгоритму. Обсуждается также вопрос о перечислении всех триангуляций плоского графа.

Ключевые слова: связный граф, планарный граф, остовное дерево, число остовных деревьев, триангуляция, число триангуляций, выпуклая оболочка.

1. Задача построения всех остовных деревьев связного графа

Пусть дан связный неориентированный граф $G = (V, E)$ с конечным числом вершин, не содержащий петель и кратных ребер. Требуется перечислить все его остовные деревья. В работе [2, с. 180, Следствие 2, с. 191] описан метод решения этой задачи, основанный на переборе миноров матрицы инцидентности графа. Там же (с. 191–193) приведен алгоритм четырех японских математиков (И. Касахара, К. Тезука, Линг Шун Тонг и Т. Китахаши [6]), сводящийся к раскрытию скобок в произведениях формальных сумм ребер графа. В данной работе предлагается алгоритм, основанный на переборе последовательностей ребер графа.

Напомним, что остовным деревом (или остовом) связного графа называется дерево, связывающее все вершины графа и составленное из его ребер. Дерево — это связный

граф без циклов. Остовное дерево называют также охватывающим деревом. Все необходимые определения можно найти в работах [1; 2; 5; 6].

Будем считать, что вершинами графа G являются числа 1, 2, 3, ..., n, где n — число вершин графа.

На множестве $\mathbf{Z}\times\mathbf{Z}$ упорядоченных пар целых чисел рассмотрим лексикографический порядок:

$$(a,b) < (a',b') \iff \begin{cases} a < a' \text{ или} \\ a = a' \text{ и } b < b'. \end{cases}$$

Пусть $m \geq 1$ — целое число. Обозначим через S_m множество последовательностей e_1, e_2, ..., e_m длины m, где $e_1 < e_2 < \ldots < e_m$ и $e_i \in \mathbf{Z} \times \mathbf{Z}$ при всех i. На множестве S_m рассмотрим лексикогрфический порядок:

$$e_1, e_2, \ldots, e_m \prec e'_1, e'_2, \ldots, e'_m \iff \begin{cases} e_1 < e'_1 \text{ или} \\ e_1 = e'_1, e_2 < e'_2 \text{ или} \\ e_1 = e'_1, e_2 = e'_2, e_3 < e'_3 \text{ или} \\ \qquad \ldots \end{cases}$$

Через (a,b) будем обозначать ребро, соединяющее вершины a и b. При этом считаем, что $a < b$, поскольку G — неориентированный граф, не имеющий петель.

Пусть T — остовное дерево в графе G. Известно, что число ребер остовного дерева графа на n вершинах равно $n-1$ [1; 5]. Поэтому множество ребер дерева T можно представить в виде упорядоченной последовательности $E' = E'_T$, состоящей из ребер графа:

$$E' = e_1, e_2, e_3, \ldots, e_{n-1}, \tag{1}$$

где

$$e_i = (a_i, b_i), a_i, b_i \in \mathbf{Z} \text{ при } i = 1, 2, \ldots, n-1.$$

Не теряя общности, будет считать, что $e_1 < e_2 < \ldots < e_{n-1}$, то есть

$$a_1 \leq a_2 \leq a_3 \leq \ldots \leq a_{n-1} \tag{2}$$

и при любом $i < n-1$ верно $a_i \leq b_i$, а также

$$a_i < a_{i+1} \text{ или } a_i = a_{i+1} \text{ и } b_i < b_{i+1}. \tag{3}$$

При этом $a_1 = 1$ и $b_1 > 1$, поскольку вершина 1 связана некоторым ребром дерева T хотя бы с одной вершиной графа.

Наименьшим (в смысле порядка «$\prec$») остовным деревом будет дерево с ребрами $(1,2)$, $(1,3)$, $(1,4)$, ..., $(1,n)$ при условии, что $(1,b) \in E$ при $b = 2, 3, \ldots, n$. Наибольшим остовным деревом будет дерево с ребрами $(1,n)$, $(2,n)$, $(3,n)$, ..., $(n-1,n)$ при аналогичном условии. Перебирая все элементы множества S_{n-1} в порядке возрастания между указанными наборами ребер и проверяя отсутствие циклов и наличие связности у соответствующих наборов, сможем перечислить все остовные деревья графа.

Алгоритм перебора остовных деревьев связного графа

ВХОД: число n вершин связного графа $G = (V, E)$ и множество E его ребер (множество E может задаваться явно — как список ребер, или неявно — через матрицу смежности или через матрицу инцидентности).

ВЫХОД: список S всех остовных деревьев графа G.

В процессе работы алгоритма формируется и корректируется упорядоченная последовательность E', состоящая из ребер графа. В начале работы алгоритма число k членов этой последовательности равно 0. Через nom обозначается порядковый номер строящегося остовного дерева. Наборы ребер остовных деревьев записываются в список S остовных деревьев. В начале работы алгоритма этот список пуст.

1) Полагаем $S = \emptyset$, $k = 0$, $nom = 0$, $x = 1$ и $y = 2$.
2) Если $y < n$ и $(x, y) \notin E$, то увеличиваем y на 1 и идем к пункту 2.
3) Если $k > 0$, $y \leq n$ и добавление ребра (x, y) к графу $G_k = (V, E')$ создает цикл, то увеличиваем y на 1 и идем к пункту 2.
4) Если $y > n$, то увеличиваем x на 1, полагаем $y = x + 1$ и идем к пункту 2.
5) Если $x \geq n$ и $k = 0$, то завершаем работу.
6) Если $x \geq n$ и $k > 0$, то полагаем $x = a_k$, $y = b_k + 1$, уменьшаем k на 1 и идем к пункту 2.
7) Увеличиваем k на 1, полагаем $a_k = x$, $b_k = y$, увеличиваем y на 1 и идем к пункту 2.
8) Если $k < n - 1$, то увеличиваем k на 1 и идем к пункту 2.
9) Построенная последовательность ребер $E' = ((a_1, b_1), (a_2, b_2), \ldots, (a_{n-1}, b_{n-1}))$ образует остовное дерево. Увеличиваем счетчик nom на 1 и добавляем E' в список S остовных деревьев (под номером nom).
10) Уменьшаем k на 1, полагаем $x = a_k$, $y = b_k + 1$ и идем к пункту 2.

Можно дать альтернативное описание алгоритма перечисления остовов в виде обхода вершин некоторого ориентированного дерева.

Пусть задано ориентированное дерево $\mathcal{D}$ с конечным числом вершин и корнем d. Пусть в каждую вершину этого дерева можно попасть из корня, перемещаясь по дугам. Пусть для каждой вершины v на множестве дуг, исходящих из v, задан некоторый линейный порядок «$\prec$». Пусть также $\mathcal{D}_0$ — множество листьев дерева, то есть таких вершин u, из которых нет исходящих дуг. Тогда можно обойти вершины дерева $\mathcal{D}$, побывав в каждом листе ровно один раз. Для этого надо использовать поиск в глубину [1; 5; 6], начиная поиск с корня и придерживаясь таких правил:

- Если мы попадаем по дуге (u, v) в вершину v и существует хотя бы одна непройденная дуга, исходящая из вершины v, то выбираем наименьшую (в смысле порядка «$\prec$») непройденную дугу (v, w) и по ней перемещаемся в вершину w.
- Если мы попали в вершину v по дуге (u, v) и из вершины v нет выходящих дуг или все выходящие дуги уже пройдены, то возвращаемся назад в вершину u.
- Если мы возвратились в вершину u по некоторой дуге и из этой вершины выходит хотя бы одна ранее не пройденная дуга, то движемся вперед по наименьшей (в смысле порядка «$\prec$») из таких дуг.

Чтобы получить список всех остовов графа G, надо определить дерево $\mathcal{D}$ следующим образом. Его вершинами являются упорядоченные наборы упорядоченных пар целых чисел

$$e_1 = (a_1, b_1), e_2 = (a_2, b_2), e_3 = (a_3, b_3), \ldots, e_k = (a_k, b_k),$$

где $1 \le k < n$, $e_1 < e_2 < \ldots < e_k$ и $a_i < b_i$ при всех i. Пусть также $\mathcal{D}$ содержит пустой набор $\emptyset$, который является корнем дерева.

Пусть $u = e_1, e_2, \ldots, e_k$ и $v = e'_1, e'_2, \ldots, e'_l$ две вершины дерева $\mathcal{D}$. Из вершины u в вершину v идет дуга тогда и только тогда, когда $l = k + 1$ и $e_i = e'_i$ при $i = 1, 2, \ldots, k$. Иными словами, (u, v) — дуга в том и только том случае, когда набор v получается из набора u путем добавления в конец набора u некоторой упорядоченной пары e_{k+1} целых чисел.

Пусть $u = e_1, e_2, \ldots, e_k$ — вершина дерева $\mathcal{D}$ и $k < n - 1$. Тогда из u выходит хотя бы одна дуга. Введем на множестве дуг, выходящих из u, отношение порядка «$\prec$». Пусть имеется две дуги (u, v) и (u, v'). Тогда v и v' получаются из набора u путем добавления в конец набора u некоторых различных пар целых чисел e_{k+1} и e'_{k+1} соответственно.

Тогда считаем, что $(u, v) \prec (u, v')$ в том и только том случае, когда $e_{k+1} < e'_{k+1}$. Ясно, что листьям построенного дерева будут соответствовать остовные деревья исходного графа.

Приведем результаты работы компьютерной программы, составленной по описанным выше алгоритмам.

Пример 1. Для полных графов на n вершинах при $n \le 9$ получены списки всех остовных деревьв. Число остовных деревьев для таких графов по теореме Кэли равно n^{n-2} [1; 4–6]. Например, при $n = 9$ число остовов равно 4 782 969.

Пример 2. Рассмотрим граф на 12 вершинах, изображенный слева на рисунке 1. Он имеет 2 415 остовных деревьев. Приведем списки первых трех остовов этого графа, а также последних трех остовов (в левой колонке указан порядковый номер остова при лексикографическом упорядочении, в правой — список ребер остова):

1	(1,2), (1,5), (2,3), (2,6), (3,4), (3,7), (4,8), (5,9), (6,10), (7,11), (8,12)
2	(1,2), (1,5), (2,3), (2,6), (3,4), (3,7), (4,8), (5,9), (6,10), (7,11), (11,12)
3	(1,2), (1,5), (2,3), (2,6), (3,4), (3,7), (4,8), (5,9), (6,10), (8,12), (10,11)
...	... ...
2 413	(1,5), (2,6), (3,7), (4,8), (5,9), (6,10), (7,8), (7,11), (9,10), (10,11), (11,12)
2 414	(1,5), (2,6), (3,7), (4,8), (5,9), (6,10), (7,8), (8,12), (9,10), (10,11), (11,12)
2 415	(1,5), (2,6), (3,7), (4,8), (5,9), (6,10), (7,11), (8,12), (9,10), (10,11), (11,12)

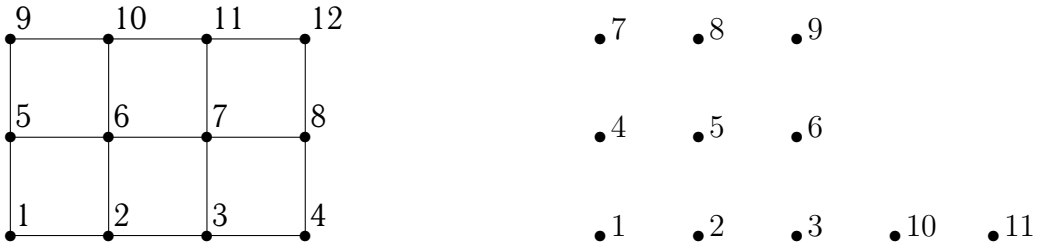

Рис. 1

Пусть P — конечный набор точек плоскости. Геометрический граф $G = (V, E)$ с носителем P — это граф, вершинами которого являются точки из P, а ребрами — прямолинейные отрезки, соединяющие эти вершины, причем отрезки могут пересекаться только в точках набора P. Такие графы изучались многими авторами (в иностранных работах — под названием crossing-free geometrical graph, см., например, [5; 6]). Если

геометрический граф является деревом и соединяет все вершины P, то он называется остовным деревом (spanning tree) с носителем P. Изменив соответствующим образом алгоритм $\mathcal{A}$, можно получить алгоритм перебора всех таких деревьев с заданным носителем P.

Пример 3. Пусть P — множество вершин квадрата, а также его центр. Тогда имеется 45 остовов.

Пример 4. Пусть P состоит из вершин квадрата, а также середин его сторон. Тогда имеется 3 273 остова.

Пример 5. Пусть P состоит из вершин квадрата, середин его сторон, а также центра квадрата. Тогда число остовов равно 24 965.

Пример 6. Рассмотрим множество точек, изображенное на рисунке 1 справа. Если оставить в этом множестве первые n вершин, то для полученного множества $P = P_n$ число остовов таково:

n	7	8	9	10	11
Число остовов	1 018	6 164	24 965	169 458	854 692

Приведем списки ребер некоторых остовов для множества P_{11} (в левой колонке указан порядковый номер остова при лексикографическом упорядочении, в правой — список ребер остова):

1	$(1,2)$, $(1,4)$, $(1,5)$, $(1,6)$, $(1,8)$, $(2,3)$, $(3,10)$, $(4,7)$, $(5,9)$, $(6,11)$
2	$(1,2)$, $(1,4)$, $(1,5)$, $(1,6)$, $(1,8)$, $(2,3)$, $(3,10)$, $(4,7)$, $(5,9)$, $(9,11)$
3	$(1,2)$, $(1,4)$, $(1,5)$, $(1,6)$, $(1,8)$, $(2,3)$, $(3,10)$, $(4,7)$, $(5,9)$, $(10,11)$
...	... ...
100 000	$(1,2)$, $(1,4)$, $(2,7)$, $(3,5)$, $(5,6)$, $(5,11)$, $(6,7)$, $(7,8)$, $(8,9)$, $(10,11)$
100 001	$(1,2)$, $(1,4)$, $(2,7)$, $(3,5)$, $(5,6)$, $(5,11)$, $(6,7)$, $(7,8)$, $(9,11)$, $(10,11)$
...	... ...
200 000	$(1,2)$, $(1,6)$, $(2,3)$, $(4,5)$, $(4,7)$, $(5,6)$, $(6,9)$, $(7,8)$, $(9,10)$, $(10,11)$
200 001	$(1,2)$, $(1,6)$, $(2,3)$, $(4,5)$, $(4,7)$, $(5,6)$, $(6,9)$, $(7,8)$, $(9,11)$, $(10,11)$
...	... ...
300 000	$(1,2)$, $(2,3)$, $(3,4)$, $(4,5)$, $(4,10)$, $(6,7)$, $(7,10)$, $(8,9)$, $(9,10)$, $(9,11)$
300 001	$(1,2)$, $(2,3)$, $(3,4)$, $(4,5)$, $(4,10)$, $(6,7)$, $(7,10)$, $(8,9)$, $(9,10)$, $(10,11)$
...	... ...

Пример 7. Рассмотрим на плоскости множество из $n \cdot m$ точек:

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

где $n, m \geq 1$ — целые числа. Таким образом, $L_{n,m}$ — равномерная решетка, точки которой лежат на n горизонтальных прямых и на m вертикальных прямых. Число остовов этой решетки при небольших n и m указано в следующей таблице:

n	2	2	2	2	2	3	3
m	2	3	4	5	6	3	4
Число остовов	12	169	2 665	44 329	759 114	24 965	4 595 581

При перечислении остовных деревьев можно осуществлять фильтрацию. Например, можно оставлять в итоговом списке только те деревья, степени вершин которых ограничены заданным числом. Так, полный граф на 8 вершинах содержит $8^6 = 262\,144$ остова. Среди них 201 600 остовов, степени вершин которых не превосходят 3 и 20 160 остовов со степенями вершин, не большими 2.

Незначительная модификация описанного выше алгоритма обхода дерева позволяет получить список всех геометрических графов (crossing-free geometrical graph), определяемых множеством точек на плоскости. Для этого надо к набору приведенных выше правил добавить следующее:

- Если мы возвратились в вершину u по некоторой дуге, и все дуги, выходящие из этой вершины, уже пройдены, то заносим в итоговый список очередной геометрический граф, который состоит из ребер, составляющих последовательность ребер u.

Например, если P состоит из вершин квадрата, а также середин его сторон, то число определяемых множеством P геометрических графов равно 21 795 (в это число включен и пустой граф). Если же добавить к P и центр квадрата, то число геометрических графов будет равно 167 570.

2. Распараллеливание алгоритма перечисления остовов

Пусть поставлена задача о перечислении всех остовных деревьев заданного связного графа. Требуется разбить ее на $p \geq 2$ подзадач.

Способ 1. Разбиваем область поиска (то есть множество S_{n-1} всех последовательностей из $n-1$ ребра) на p частей (отрезков относительно лексикографического порядка «$\prec$»), состоящих приблизительно из одинакового числа последовательностей, и пусть каждый процессор обрабатывает свою часть.

При таком способе каждый процессор должен знать только список ребер графа и границы отрезка, на котором надо осуществлять поиск. Кроме того, надо сбрасывать в общий список набор найденных деревьев. Однако число остовных деревьев на разных отрезках может оказаться различным и часть процессоров будет простаивать после завершения своей части работы. Рассмотрим, например, множество точек P, изображенное слева на рисунке 2.

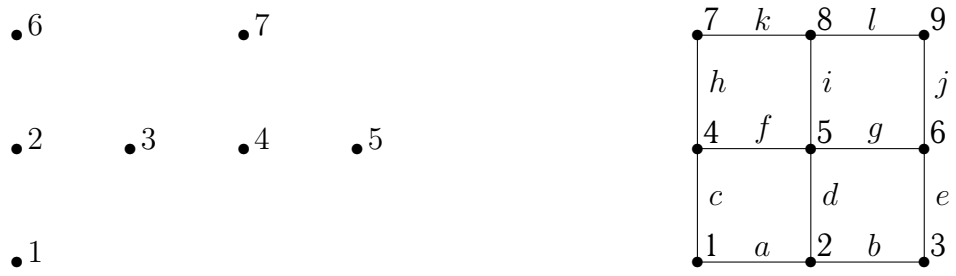

Рис. 2

Для этого множества существует 1 302 остовных дерева. Однако если область поиска разбить на 4 отрезка, состоящих из примерно одинакового числа последовательностей, то на одном из участков будет 618 остовных деревьев, а на других — соответственно 345, 207 и 132.

Способ 2. Разбиваем область поиска на $K \cdot p$ частей (отрезков), где $K > 2$ — целое число (например, $K = 10$). Первые $p - 1$ процессоров начинают обрабатывать первые $p - 1$ отрезков множества S_{n-1}, а процессор с номером p будет координировать работу остальных: если какой-то процессор завершает поиск на своем участке, то он сообщает об этом процессору с номером p и получает от него границы нового участка, на котором надо провести поиск.

3. Сравнение методов перечисления остовных деревьев графа

Напомним описание упомянутого в первой части работы алгоритма построения всех остовов графа, предложенного четырьмя японскими математиками в 1962 г. (И. Касахара, К. Тезука, Линг Шун Тонг и Т. Китахаши, [6], см. также [2, с. 192]). Пусть дан граф G с множеством вершин $V = \{v_1, v_2, \ldots, v_n\}$ и множеством ребер $E = \{u_1, u_2, \ldots, u_m\}$. Алгоритм предполагает выполнение следующих шагов:

1) Составляем матрицу инцидентности A графа G и выбираем такое множество вершин $V_0 \subset V$, что строки, соответствующие вершинам из V_0, образуют базис системы строк матрицы A над полем $Z_2 = \{0, 1\}$ из двух элементов.
2) Каждой вершине $v \in V_0$ сопоставляем формальную сумму

$$S(v) = u_{i_1} + u_{i_2} + \ldots + u_{i_k},$$

где $u_{i_1}, u_{i_2}, \ldots, u_{i_k}$ — список всех ребер, инцидентных вершине v, а k — число таких ребер, то есть степень вершины v.
3) Составляем произведение P сумм $S(v)$ по все вершинам $v \in V_0$, заключив каждую такую сумму в скобки.
4) Раскрываем скобки в произведении P и приводим подобные члены над полем Z_2 при дополнительных соотношениях $u_1^2 = 0$, $u_2^2 = 0$, …, $u_m^2 = 0$. Тогда P будет представлено в виде слагаемых вида $\alpha \cdot u_{j_1} \cdot u_{j_2} \cdot \ldots \cdot u_{j_{n-1}}$, где α равно 0 или 1, а множители u_{j_1}, u_{j_2}, …, $u_{j_{n-1}}$ различны. При этом каждое такое слагаемое, для которого $\alpha = 1$, будет соответствовать ровно одному остову графа, составленному из ребер u_{j_1}, u_{j_2}, …, $u_{j_{n-1}}$.

Рассмотрим, например, граф, изображенный на рисунке 2 справа. Ребра этого графа обозначены (для упрощения записи) переменными без индексов a, b, c и т. д. (вместо u_1, u_2, u_3, …). При естественной нумерации вершин и ребер графа его матрица инцидентности имеет вид

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Последняя строка равна сумме всех остальных строк (в поле Z_2), а первые восемь строк линейно независимы. Поэтому за V_0 можно принять множество вершин $\{1, 2, 3, \ldots, 7, 8\}$. В этом случае

$$P = (a + c)(a + b + d)(b + e)(c + f + h)(d + f + g + i)(e + g + j)(h + k)(i + k + l).$$

При раскрытии всех скобок возникнет 2 592 слагаемых, из которых 192 не содержат квадратов переменных и поэтому определяют ровно по одному из остовов графа. При реализации в виде компьютерной программы процесс раскрытия скобок соответствует поиску в ширину на графе, поэтому при работе такая программа требует много дополнительной памяти (по сравнению с поиском в глубину) и имеет более сложную структуру. Кроме того, неясно, как применить этот алгоритм при поиске остовов геометрических графов.

4. Триангуляции многоугольника на плоскости

Пусть F — замкнутый многоугольник на плоскости и $P = \{p_1, p_2, \ldots, p_n\}$ — такое его конечное подмножество, которое содержит все вершины F. Необходимо перечислить все триангуляции многоугольника F, вершинами которых являются точки множества P. Алгоритм решения этой задачи описан в [3]. Дадим альтернативное описание этого алгоритма. Не теряя общности, будем считать, что отрезок $[p_1, p_2]$ является стороной многоугольника F.

Рассмотрим дерево $\mathcal{D}$, вершинами которого являются такие упорядоченные наборы $\Delta_1, \Delta_2, \ldots, \Delta_k$ треугольников с вершинами из множества P, объединение которых является связным множеством и которые можно достроить до триангуляции на P путем добавления (справа) других треугольников (с вершинами из P), или эти наборы уже сами являются триангуляциями на P. Кроме того, дерево $\mathcal{D}$ содержит пустой набор $\emptyset$, который является корнем дерева. Пусть $u = \Delta_1, \Delta_2, \ldots, \Delta_k$ и $v = \Delta'_1, \Delta'_2, \ldots, \Delta'_l$ две вершины дерева $\mathcal{D}$. Тогда из вершины u в вершину v идет дуга тогда и только тогда, когда $l = k + 1$ и $\Delta_i = \Delta'_i$ при $i = 1, 2, \ldots, k$. Иными словами, (u, v) — дуга в том и только том случае, когда набор v получается из набора u путем добавления в конец набора u некоторого треугольника. Триангуляциям многоугольника F (с дополнительными точками из P) соответствуют листья дерева $\mathcal{D}$.

Пусть $u = \Delta_1, \Delta_2, \ldots, \Delta_k$ — вершина дерева $\mathcal{D}$, не являющаяся триангуляцией. Тогда из u выходит хотя бы одна дуга. Введем на множестве дуг, выходящих из u, отношение порядка «$\prec$». Пусть имеется две дуги (u, v) и (u, v'). Тогда v и v' получаются из набора u путем добавления в конец набора u некоторых треугольников Δ_{k+1} и Δ'_{k+1} соответственно. Эти треугольники имеют общие отрезки с некоторыми треугольниками набора u. Пусть a, b и a', b' — номера вершин этих отрезков. Не теряя общности, считаем, что $a < b$ и $a' < b'$. Пусть также c и c' — номера третьих вершин треугольников Δ_{k+1} и Δ'_{k+1}. Тогда считаем, что $(u, v) \prec (u, v')$ в том и только том случае, когда $a < a'$ или когда $a = a'$ и $b < b'$ или когда $a = a'$, $b = b'$ и $c < c'$. Таким образом, «$\prec$» является модификацией лексикографического порядка.

Напомним, что нумерация вершин множества P такова, что отрезок $[p_1, p_2]$ является стороной многоугольника F. Поэтому из корня $\emptyset$ дерева $\mathcal{D}$ идут дуги вида $(\emptyset, v)$, где набор v состоит из единственного треугольника Δ_1, номера вершин которого 1, 2 и j, где $2 < j \leq n$. Если другая дуга $(\emptyset, v')$ определяет тройку чисел 1, 2, j', то считаем, что $(\emptyset, v) \prec (\emptyset, v)$ тогда и только тогда, когда $j < j'$.

Теперь для построения всех триангуляций на P надо осуществить обход всех вершин дерева $\mathcal{D}$ поиском в глубину, начиная поиск с корня дерева. Таким способом можно получить следующие результаты.

Пример 8. Пусть $P = P_{11}$ — множество из 11 точек, изображенное на рисунке 1 справа, и пусть F — выпуклая оболочка P. Тогда число триангуляций равно 302.

Пример 9. Пусть $L_{n,m}$ — равномерная решетка на плоскости (см. пример 7). Число ее триангуляций при небольших n и m указано в следующей таблице:

n	2	2	2	3	3	4
m	2	3	8	6	7	5
Число триангуляций	2	6	3 432	182 132	2 801 708	2 822 648

СПИСОК ЛИТЕРАТУРЫ

1. Зыков, А. А. Основы теории графов / А. А. Зыков. — М. : Наука, 1987. — 384 с.

2. Зыков, А. А. Теория конечных графов / А. А. Зыков. — Новосибирск : Наука, 1969. — 554 с.

3. 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.

4. Aichholzer, O. On the Number of Plane Geometrical Graphs / O. Aichholzer, T. Hackl, B. Vogtenhuber, C. Huemer, F. Hurtado, H. Krasser // Graphs and Combinatorics. — 2007. — № 23. — P. 67–84.

5. Diestel, R. Graph Theory / R. Diestel. — N. Y. : Springer-Verlag, 2000. — 384 p.

6. Kasahara, Y. Topological evaluation of a system determinants / Y. Kasahara, K. Tezuka, S. T. Ling, T/ Kitahashi // Technol. Repts. Osaka Univ. — 1962. — № 12. — P. 239–248.

REFERENCES

1. Zykov A.A. *Osnovy teorii grafov* [Introduction to Graph Theory]. Moscow, Nauka Publ., 1987. 384 p.

2. Zykov A.A. *Teoriya konechnykh grafov* [Finite Graph Theory]. Novosibirsk, Nauka Publ., 1969. 554 p.

3. 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.

4. Aichholzer O., Hackl T., Vogtenhuber B., Huemer C., Hurtado F., Krasser H. On the Number of Plane Geometrical Graphs. *Graphs and Combinatorics*, 2007, no. 23, pp. 67-84.

5. Diestel R. *Graph Theory*. N. Y., Springer-Verlag, 2000. 384 p.

6. Kasahara Y., Tezuka K., Ling S.T., Kitahashi T/ Topological evaluation of a system determinants. *Technol. Repts. Osaka Univ.*, 1962, no. 12, pp. 239-248.

ON ALGORITHM OF NUMBERING THE SPANNING TREES IN A CONNECTED GRAPH

Popov Vladimir Valentinovich

Candidate of Physical and Mathematical Sciences, Associate Professor, Department of Computer Sciences and Experimental Mathematics,
Volgograd State University
popov_v_v@rambler.ru, kiem@volsu.ru
Prosp. Universitetsky, 100, 400062 Volgograd, Russian Federation

Abstract. Let $G = (V, E)$ be a finite connected graph without multiple edges or loops. We consider the task about the numbering of all the spanning trees of G. In [2, p. 180, p. 191] described a method of solution of this task, based on the properties of minors of an incidence matrix of G. In the same place (p. 191–193) gave algorithm of four Japanese mathematicians (Kasahara Y., Tezuka K., Ling Shun Tong, Kitahashi T., [6]), reduced the task to removal of brackets in the product of formal sums of edges of G. Here we concider the method based on lexicographical order on the set of all sequences of edges of G.

We will remind that a spanning tree T of the graph G is a tree consisting of edges of G and connecting all the vertices of G.

We shall assume that $V = \{1, 2, \ldots, n\}$, where n is a number of vertices of G. If $a, b \in V$, then (a, b) will be designated the edge with end-points a and b.

Let T be a spanning in G. Then the set of his edges can be written in the form

$$(a_1, b_1), (a_2, b_2), \ldots, (a_{n-1}, b_{n-1}), \qquad (A)$$

where

$$a_i < a_{i+1} \text{ or } a_i = a_{i+1} \text{ and } b_i < b_{i+1}, \text{ where } i = 1, 2, \ldots, n-2. \qquad (B)$$

On a set of sequences of in the form of (A) with the additional condition (B) we will consider a lexicographic order. The smallest (with respect to this order) spanning tree will be the tree T_0 with edges $(1, 2)$, $(1, 3)$, $(1, 4)$, ..., $(1, n)$ provided $(1, b) \in E$ for $b = 2, 3, \ldots, n$. The greatest spanning tree T_1 will be $(1, n)$, $(2, n)$, $(3, n)$, ..., $(n-1, n)$ under a similar condition. Touching all sequences of a look (A) in ascending order between T_0 and T_1 and checking lack of cycles at the corresponding sets of edges, we will be able to get a list of all spanning trees.

We will give results of work of the appropriate computer program.

Example 1. For complite graph K_n on n vertices for $n \leq 9$ the lists of all the spanning trees are obtained. Due to Kely theorem K_n has n^{n-2} spanning trees. For $n = 9$ thos number is equal $4\,782\,969$.

Example 2. Let G be a graph on 12 vertices in figure 1 (in the left). Then G has $2\,415$ spanning trees.

Let P be a finite set of points in the plane R^2. Then $G = (V, E)$ is a plane geometrical graph on top of P, if $V \subset R^2$ and egdes of G are stright-line segments with the end-points in P and two edges of G intersects only in points of P [3]. If G is a tree and $V = P$, then G called a spanning tree on top of P.

Example 3. Let P be a set of square tops, and also its center. Then there are 45 spanning trees on top of P.

Example 4. Let P consists of square tops, and also middle of its parties. Then there are 3 273 spanning trees on top of P.

Example 7. We will consider a set $P = L_{n,m}$ consisting of $n \cdot m$ points on the plane:

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

where $n, m \geq of1$ are integers. Thus, $L_{n,m}$ is a uniform lattice which points lie on n horizontal straight lines and for m vertical straight lines. The numbers $St(P)$ of a spanning trees on top of this lattice for small n and m it is specified in the following table:

n	2	2	2	2	2	3	3
m	2	3	4	5	6	3	4
$St(P)$	12	169	2 665	44 329	759 114	24 965	4 595 581

At transfer the spanning trees it is possible to carry out a filtration. For example, it is possible to keep only those trees in the final list, which degrees of vertices not exceeded some number r. So, the complete graph on 8 vertices has $8^6 = 262\,144$ spanning trees. Among them 201 600 spanning trees which degree of vertices don't exceed 3, and 20 160 spanning trees with degree of vertices ≤ 2.

Some modification of the main algorihm allows to receive the list of all crossing-free geometrical graph on top of P [3]. For example, if P consists of square tops, and also middle of its parties, the number of such a graph on top of P is equal to 21 795 (this number included also the empty graph). If to add to P the center of a square, this number will become to 167 570.

In work is considered also a task about creation of all triangulations of a finite set $P \subset R^2$. The algorithm of the solution of this task is available in [3]. The alternative description of algorithm of search is provided in this work.

Example 9. Let $P = L_{n,m}$ be a uniform lattice on the plane (see example 7). Then the number $Tr(P)$ of triangulations of $L_{n,m}$ for small n and m is specified in the following table:

n	2	2	2	3	3	4
m	2	3	8	6	7	5
$Tr(P)$	2	6	3 432	182 132	2 801 708	2 822 648

Key words: connected graph, planar graph, spanning tree, the number of spanning trees, triangulation, the number of triangulations.