Кибернетический сборник

НОВАЯ СЕРИЯ

ВЫПУСК

27

Перевод с английского

под редакцией

О. Б. ЛУПАНОВА И
О. М. КАСИМ-ЗАДЕ

МОСКВА «МИР» 1990

ББК 32.81
К38
УДК 519.95

Кибернетический сборник. Новая серия. Вып. 27. Сб.
К38 статей: Пер. с англ. — М.: Мир, 1990. — 200 с., ил.
ISSN 0234—1921

Продолжение известной серии сборников, начатой издательством в 1965 г. В данном выпуске содержатся оригинальные и обзорные работы зарубежных ученых по актуальным проблемам теоретической кибернетики и ее приложениям. Большой интерес вызовут статьи К. Пападимитриу и др. о построении геодезических и Р. Тарьяна и др. о сверхбыстрых алгоритмах триангуляции. Применению новых методов перечислительной комбинаторики посвящена статья Ж. Вьенно. Включены также работы по теории сложности, алгебраическим схемам отношений и обзор по методам преобразования многоуровневых изображений в двухуровневые. Среди авторов — специалисты из Канады, США, Франции.

Для научных работников, инженеров-исследователей, аспирантов и студентов университетов.

К $\frac{1402010000-377}{041(01)-90}$ 1—90 **ББК 32.81**

Редакция литературы по математическим наукам

ISSN 0234—1921

Дискретная геодезическая задача[1)]

Джозеф Митчел[2)], *Дэвид Маунт*[3)] *и Кристос Пападимитриу*[4)]

Предлагается алгоритм определения кратчайшего пути между двумя точками на произвольной (возможно, невыпуклой) многогранной поверхности. Путь должен проходить по поверхности; метрика предполагается евклидовой. Сложность алгоритма по времени составляет $O(n^2 \log n)$, а по объему памяти $O(n^2)$, где n — число ребер поверхности. После того как алгоритм отработает, расстояние от начальной точки до любой концевой точки поверхности может быть определено с помощью стандартных методов за время $O(\log n)$ локализацией положения конечной точки в полученном разбиении поверхности. Кратчайший путь от начальной до конечной точки может быть найден за время $O(k + \log n)$, где k — число граней, через которые он проходит. Алгоритм можно обобщить на случай нескольких начальных точек, что позволяет строить на поверхности диаграмму Вороного, при этом n равно максимуму из числа вершин поверхности и числа начальных точек.

1. ВВЕДЕНИЕ

Недавние исследования алгоритмических аспектов построения роботов и навигации в средах с препятствиями привели к нескольким интересным вариантам постановки задачи поиска кратчайшего пути. Задача определения кратчайшего пути между двумя точками в трехмерном пространстве при наличии препятствий в виде многогранников оказывается очень сложной, и для ее решения известны лишь чрезвычайно неэффективные

[1)] The Discrete Geodesic Problem. SIAM J. Comput., vol. 16, No. 4, August 1987.

[2)] Department of Operations Research, Stanford University and Hughes Artificial Intelligence Center, Stanford, California 94305.

[3)] Department of Computer Science, University of Maryland, Baltimore, Maryland 21201.

[4)] Department of Operations Research and Department of Computer Science, Stanford University, Stanford, California 94305.

[16, 17] или приближенные [14] алгоритмы. Соответствующая двумерная задача с препятствиями в виде многоугольников может быть легко решена за время $O(n^2 \log n)$ путем построения *графа видимости* [4, 6, 8, 9, 17], а в некоторых особых случаях, когда кратчайшие траектории обладают рядом свойств монотонности, сложность алгоритма удается снизить до $O(n \log n)$ [5, 9, 17] (здесь n — суммарное число вершин препятствий многоугольников). Недавно было сообщено [16] об алгоритме, решающем двумерную задачу нахождения кратчайшего пути, который имеет сложность $O(nk + n \log n)$ (где k — число отдельных простых многоугольных препятствий).

В этой статье мы рассмотрим особый случай трехмерной постановки задачи поиска кратчайшего пути, известный под названием *дискретной геодезической задачи*. На поверхности заданного многогранника даны две точки, и требуется найти кратчайший путь между ними, проходящий по этой поверхности. Геодезическая задача представляет определенный интерес для наземной навигации на пересеченной местности, где движущийся аппарат должен оставаться на поверхности, которую можно смоделировать многогранником. Заметим, что длина полученного пути вовсе не обязана быть минимальной для свободного от многогранника пространства, поскольку в нашем случае мы вынуждены оставаться на многогранной поверхности. Чтобы убедиться в том, что приведенная формулировка есть частный случай общей трехмерной постановки, представим себе два многогранных препятствия: открытый многогранник и дополнение к его замыканию. Тогда путь между двумя точками, не пересекающий оба многогранника, есть как раз путь, лежащий на общей для них обоих поверхности. С другой стороны, решаемая нами задача представляет собой обобщение двумерной постановки, поскольку каждое многоугольное препятствие можно представить в виде очень высокой призмы, выступающей над плоскостью, в которой лежит заданный многоугольник.

Впервые задача поиска кратчайшего пути на поверхности была поставлена в работе [17], где описан алгоритм ее решения для случая *выпуклого* многогранника, имеющий временную сложность $O(n^3 \log n)$. При этом n — это мера сложности сцены, которую в нашей ситуации можно оценить, к примеру, числом ребер многогранной поверхности. В работе [10] Маунт предложил алгоритм для выпуклого многогранника, который улучшил оценку до $O(n^2 \log n)$. Для *невыпуклого* многогранника в работе [13] приведен алгоритм, имеющий сложность $O(n^5)$.

В настоящей работе перечисленные результаты улучшаются следующим образом. Мы рассматриваем многогранник общего вида (возможно, невыпуклый или даже более сложной кон-

струкции) и получаем для него оценку сложности $O(n^2 \log n)$. При этом решается задача нахождения кратчайшего пути с одним источником (а с помощью простого обобщения и задачи с несколькими источниками, см. [11]) путем построения такого разбиения поверхности, которое позволяет определить длину кратчайшего пути до любой концевой точки простой локализацией положения этой точки в построенном разбиении. Таким образом, длина кратчайшего пути может быть определена за время $O(\log n)$, а сам путь затем находится за время $O(k + \log n)$, где k — число ребер, которые он пересекает. Тем самым наш алгоритм служит обобщением решений двумерной задачи построения кратчайшего пути среди препятствий, данных в работах [4] и [17]. Здесь мы сосредоточим внимание на случае *ограниченных* многогранников. Однако алгоритм непосредственно обобщается так, чтобы можно было работать и с неограниченными гранями.

Алгоритм основан на методе, который мы назвали «непрерывный Дейкстра», поскольку по своей структуре он очень напоминает известный алгоритм Дейкстры нахождения кратчайшего пути на графе [1]. Ребра многогранника соответствуют вершинам графа с той лишь разницей, что в нашем случае расстояние от исходной точки до ребра определено неоднозначно. Вместо этого мы имеем *функцию*, служащую как бы меткой узла, и храним дискретное описание минимума этой функции. Это требует запоминания для каждого ребра «интервалов оптимальности», разбивающих ребро на части, для которых кратчайший путь к точкам области имеет одинаковую дискретную структуру, проходя через одну и ту же последовательность вершин и ребер. Эти интервалы аналогичны понятию «срезов», использованному в работе [17].

2. ПРЕДВАРИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

Пусть $\mathcal{P}$ — заданная многогранная поверхность, определенная множеством своих граней, ребер и вершин, причем каждое ребро является общим для двух граней, а любые две грани либо не имеют общих точек, либо пересекаются по общему ребру или вершине. Будем считать грани *замкнутыми* (т. е. включающими свою границу) многоугольниками, а ребра — замкнутыми отрезками (включающими концы, которые являются вершинами). Кроме того, нам даны две особые точки s и t (*источник* и *цель* соответственно). Не ограничивая общности, можно считать, что все грани представляют собой треугольники (поскольку за время $O(n \log n)$ можно произвести триангуляцию всех граней [2], и при этом число порожденных ребер линейно зависит от числа

вершин), и что s и t — вершины нашей многогранной поверхности. Нам требуется найти кратчайший путь от начальной до конечной точки, целиком лежащий на поверхности.

Дискретная геодезическая задача (ДГЗ).
Дано: Две точки s и t на многогранной поверхности $\mathscr{P}$, которая была подвергнута триангуляции.
Требуется: Найти кратчайший в смысле евклидовой метрики путь из s в t, целиком лежащий на поверхности $\mathscr{P}$.
В действительности предлагаемый ниже алгоритм решает поставленную задачу в более общей формулировке.

Дискретная геодезическая задача с одним источником (ДГЗ-ОИ).
Дано: Две точки s и t на многогранной поверхности $\mathscr{P}$, которая была подвергнута триангуляции.
Требуется: Построить структуру, позволяющую вычислять кратчайший путь из s в любую конечную точку t, целиком лежащий на поверхности $\mathscr{P}$.
Таким образом, мы можем определить кратчайший путь от источника до любой точки на поверхности, и нам неизвестен

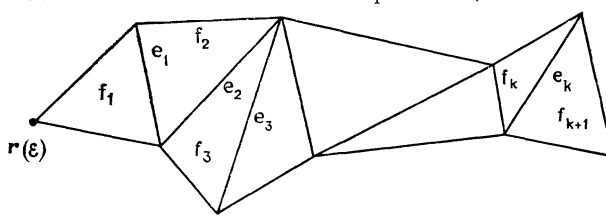


Рис. 1. Последовательность смежных по ребру граней.

алгоритм, решающий ДГЗ за более короткое время (в наихудшем случае асимптотически), чем метод ДГЗ-ОИ. Ввиду сказанного в последующем изложении мы не будем рассматривать какую-либо конкретную конечную точку.

Две грани f и f' называются *смежными по ребру*, если они имеют общее ребро e. *Последовательность смежных по ребру граней* — это список $\mathscr{F} = (f_1, f_2, \ldots, f_{k+1})$, включающий одну или более граней, таких что грань f_i смежна по ребру (обозначим его через e_i) с гранью f_{i+1} (рис. 1). В дальнейшем список ребер (возможно, пустой) $\mathscr{E} = (e_1, e_2, \ldots, e_k)$ мы будем называть *реберной последовательностью*, а вершину грани f_1, противоположную ребру e_1 — *корнем* $r(\mathscr{E})$ ($r(\varnothing)$ неопределен). Если каждая грань (соответственно ребро) встречается лишь один раз в $\mathscr{F}$ (соответственно в $\mathscr{E}$), мы говорим, что последовательность является *простой*.

С каждой гранью связана двумерная система координат. Для двух смежных по ребру граней f и f' с общим ребром e определим *плоскую развертку грани f' на грань f* как образ точек грани f' при ее повороте вокруг прямой, проходящей через e, до совмещения с плоскостью грани f, причем так, что точки f' оказываются с *противоположной* стороны от e по отношению к f (т. е. f' *не накрывает* грань f).

Координаты точек плоской развертки грани f' на грань f записываются в системе координат, связанной с гранью f. Обобщая введенное понятие, мы говорим, что развертываем реберную последовательность $\mathscr{E} = (e_1, e_2, \ldots, e_k)$ следующим образом: поворачиваем грань f_1 вокруг ребра e_1 до тех пор, пока ее плоскость не совпадет с плоскостью грани f_2; поворачиваем грани f_1 и f_2 вокруг e_2 до тех пор, пока их общая плоскость не совпадет с плоскостью грани f_3, и т. д. до тех пор, пока все грани $f_1, f_2, \ldots, f_k$ не лягут в плоскость грани f_{k+1}. Окончательная плоская развертка вдоль $\mathscr{E}$ приводит к представлению точек всех граней $f_1, f_2, \ldots, f_k$ в системе координат, связанной с гранью f_{k+1}. Если x — точка, принадлежащая грани f_i $(1 \leqslant i \leqslant k+1)$, то через $U_{\mathscr{E}}(x)$ обозначим *образ точки x при развертке вдоль $\mathscr{E}$* (представленный в системе координат, связанной с гранью f_{k+1}).

Под *путем* на $\mathscr{S}$ будем всегда подразумевать *простой* (т. е. самонепересекающийся) путь, пересечение которого с любой гранью есть объединение непересекающихся прямолинейных отрезков. *Геодезический путь* — это путь, являющийся локально оптимальным и, следовательно, не поддающийся укорачиванию посредством небольших вариаций. Мы говорим, что путь p *связывает* реберную последовательность $\mathscr{E} = (e_1, e_2, \ldots, e_k)$, если p состоит из отрезков, соединяющих внутренние точки ребер $e_1, e_2, \ldots, e_k$ (именно в таком порядке). (Заметим, что такой путь не может проходить через концы какого-либо из ребер e_i, поскольку в противном случае два разных последовательных ребра из $\mathscr{E}$ должны были бы быть коллинеарны.) Путь p *проходит через* реберную последовательность $\mathscr{E}$, если существует его часть, связывающая $\mathscr{E}$ (заметьте, что таких частей может быть несколько). Если p связывает реберную последовательность $\mathscr{E}$, то *плоская развертка p вдоль $\mathscr{E}$* есть просто образ p при плоской развертке вдоль $\mathscr{E}$.

3. ХАРАКТЕРИСТИКА ГЕОДЕЗИЧЕСКИХ И ОПТИМАЛЬНЫХ ПУТЕЙ

Характеристику геодезических и оптимальных путей мы начнем с простой леммы о существовании, утверждающей, что такие пути *существуют* (но они необязательно *единственны*).

Лемма 3.1. *Всегда существует геодезический путь из точки s до любой другой точки $x \in \mathcal{P}$. Более того, среди геодезических путей, идущих из s в x, всегда существует по крайней мере один путь наименьшей длины* [1].

Мы предположим, что рассматриваются только простые пути, т. е. пути, которые не проходят ни через какую точку более одного раза. (Для кратчайших путей это, конечно, справедливо.) В соответствии со следующей леммой оптимальные пути обладают дополнительным свойством: они не проходят ни по какой грани более одного раза.

Лемма 3.2. *Пересечение оптимального пути p с произвольной гранью f представляет собой отрезок (или пустое множество).*

Доказательство. Пусть $p \cap f \neq \varnothing$ и пусть y — первая точка p, принадлежащая f, а y' — последняя. Тогда путь из точки y в точку y' также должен быть оптимальным (иначе p можно было бы улучшить). Кратчайший путь из точки y в точку y' на f есть отрезок, их соединяющий. Следовательно, пересечение p с f должно быть прямолинейным отрезком. ■

Обратите внимание на то, что пересечение *геодезического* пути с гранью может быть объединением нескольких отдельных прямолинейных отрезков. Например, если несколько раз обмотать туго натянутую струну вокруг длинного узкого прямоугольного параллелепипеда, то путь, по которому она пройдет, локально будет оптимален, хотя глобально, конечно, нет. Отсюда также видно, что не все геодезические являются оптимальными путями и что геодезических может быть бесконечно много (хотя лишь конечное их число будет пересекать каждую грань не более одного раза).

Сейчас мы сформулируем одно простое свойство геодезических (оно упоминается в работах [7] и [17]), заключающееся в том, что при развертке геодезического пути вдоль реберной последовательности он переходит в отрезок прямой.

Лемма 3.3. *Если p — геодезический путь, связывающий реберную последовательность $\mathcal{E}$, то плоская развертка p вдоль реберной последовательности $\mathcal{E}$ есть отрезок прямой.*

Доказательство. Если бы образ *отличался* от отрезка прямой, то нашлись бы два отрезка $\overline{\alpha x}$ и $\overline{x\alpha'}$ пути p, такие что x есть внутренняя точка ребра $e = f \cap f' \subset \mathcal{E}$, и угол $\angle \alpha x \alpha'$ не

[1] В дальнейшем такой путь называется кратчайшим или оптимальным.— *Прим. перев.*

равен π. Но тогда путь p можно укоротить за счет выбора точек $\beta \in \alpha x$, $\beta' \in x\alpha'$, определяющих отрезок $\overline{\beta\beta'}$, целиком принадлежащий граням f и f'. Это противоречит локальной оптимальности p. ■

В случае выпуклых многогранников оптимальный путь никогда не пройдет через вершины многогранников, за исключением граничных точек, и это обстоятельство играет существенную роль в алгоритме, предложенном в [17]. Однако в случае невыпуклых многогранников оптимальный путь может проходить через множество других вершин. Например, на рис. 2 оптимальный путь из s в t проходит через вершину v. (Вершина v обладает тем свойством, что сумма углов при вершине v по всем граням, смежным с ней, больше 2π.) Оптимальный путь может также включать длинную цепочку смежных ребер (рис. 3).

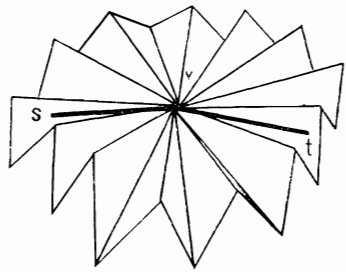


Рис. 2. Оптимальный путь, проходящий через вершину.

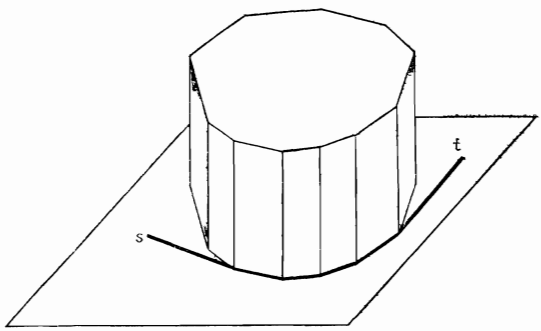


Рис. 3. Оптимальный путь, включающий цепочку ребер.

Мы можем охарактеризовать то, каким образом геодезический путь проходит через вершину. Предположим, что путь p проходит через вершину v при переходе с грани f на грань f'. Обозначим через $\overline{\alpha v}$ отрезок пути p на f, входящий в v, а через $\overline{v\alpha'}$ — отрезок пути p на f', выходящий из v. Определим угол пути p при вершине v в направлении *по часовой стрелке* как угол, отсчитываемый *от* $\overline{\alpha v}$ *к* $\overline{v\alpha'}$ *в направлении по часовой*

стрелке при плоской развертке всех граней от f до f' (включительно), содержащих v (просто нужно сложить углы, образованные соответствующими ребрами граней; в результате угол может оказаться значительно больше 2π). Аналогично определим угол в направлении *против часовой стрелки* от $\overline{\alpha v}$ к $\overline{v\alpha'}$. Определим *угол, образованный путем p при прохождении через вершину v*, как минимум из этих двух углов. Тогда мы имеем следующий результат.

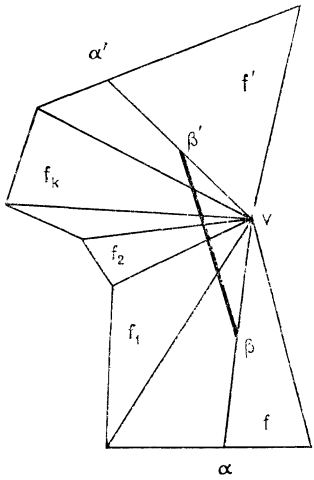


Рис. 4. Спрямление пути, проходящего через вершину.

Лемма 3.4. *Если геодезический путь p проходит через вершину v, то угол, образованный p при прохождении v, больше или равен π.*

Доказательство. Допустим, что угол меньше π и что (без ограничения общности) это угол в направлении по часовой стрелке. Если грани f, f_1, f_2, ..., f_k, f' развернуть в одной плоскости вокруг вершины v, то мы получим ситуацию, изображенную на рис. 4. Теперь заметим, что на отрезке $\overline{\alpha v}$ существует точка β, а на отрезке $\overline{v\alpha'}$ — точка β', такие что отрезок $\beta\beta'$ целиком принадлежит объединению развернутых граней f, f_1, f_2, ..., f_k, f'. Ясно, что отрезок $\overline{\beta\beta'}$ «спрямляет» путь, проходящий через вершину, так что p не может быть геодезическим путем. ■

Часть пути p между любыми двумя последовательными вершинами v и v' связывает некоторую реберную последовательность $\mathscr{E}$ с корнем $v = r(\mathscr{E})$ (если $\mathscr{E} = \varnothing$, то v и v' — концевые точки ребра e, являющегося частью p). Если p — геодезический путь, то, согласно лемме 3.3, часть его от вершины v до вершины v' полностью определяется заданной реберной последовательностью $\mathscr{E}$. Следовательно, мы можем описать геодезический путь из s в x в виде списка $p = (v_1 = s, \mathscr{E}_1, v_2, \mathscr{E}_2, v_3, \ldots, v_K, \mathscr{E}_K, x)$, где $v_1, v_2, \ldots, v_K$ — вершины (упорядоченные), через которые проходит p, а $\mathscr{E}_1, \mathscr{E}_2, \ldots, \mathscr{E}_K$ — реберные последовательности (возможно, пустые), которые связывает p. Отметим, что $v_i = r(\mathscr{E}_i)$ для всякого i, такого что $\mathscr{E}_i \neq \varnothing$. Мы назовем v_K ***корнем пути*** p, а $\mathscr{E}_K$ — последней реберной последовательностью p. Для геодезических путей p реберные последователь-

ности необязательно простые (вспомните пример со струной, намотанной на прямоугольный параллелепипед); однако если p оптимален, то каждая $\mathscr{E}_i$ должна быть простой, и никакое ребро не должно входить более чем в одну реберную последовательность $\mathscr{E}_i$, $1 \leqslant i \leqslant K$ (иначе p в противоречии с леммой 3.2 более одного раза пересечет ребро).

Подводя итог сказанному, мы можем следующим образом охарактеризовать геодезические и оптимальные пути на невыпуклых поверхностях.

Лемма 3.5. *В общем случае геодезический путь есть путь, который проходит через чередующуюся последовательность вершин и (возможно, пустых) реберных последовательностей, таких что образ пути при развертке вдоль любой реберной последовательности представляет собой прямолинейный отрезок, а угол, образованный путем при прохождении через вершины, больше или равен* π. *Общий вид оптимального пути такой же, как и геодезического, за исключением того, что никакое ребро не может появиться более чем в одной реберной последовательности и каждая реберная последовательность должна быть простой.*

4. f-СВОБОДНЫЕ ПУТИ И ИНТЕРВАЛЫ ОПТИМАЛЬНОСТИ

Для произвольной точки x обозначим через $p(x)$ оптимальный путь из s в x, и пусть $d(x)$ — длина $p(x)$. Множество всех точек y ребра e, таких, что существует оптимальный путь в y с корнем r и последней последовательностью $\mathscr{E}$, представляет собой некоторое подмножество ребра e, которое в общем случае может содержать до $O(n)$ отрезков. Это приводит к заметным трудностям, когда наш алгоритм пытается «распространить» сигналы, идущие из r, через ребро e. В случае выпуклого многогранника подобной проблемы не возникает. В этом случае все «клинья» на плоской развертке поверхности выпуклы, поскольку их границы прямолинейны (это следует из того факта, что оптимальные пути не проходят через вершины, отличные от s, и, значит, разделяющие «клинья» кривые являются прямыми). В невыпуклом случае кривые раздела будут гиперболическими дугами, так что «клинья» будут невыпуклыми.

Чтобы преодолеть указанную трудность, мы вводим понятие *f-свободного пути*. Если задана грань f и e — одно из трех ее ребер (сокращенно это будем обозначать так: (e, f) — *пара ребро — грань*), то *f-свободный путь* в точку $y \in e$ определяется как путь на $\mathscr{S}$ из s в y, не пересекающий внутренней части грани f. *Кратчайший* f-свободный путь в y (обозначаемый $p_f(y)$)

есть f-свободный путь в y минимальной длины. Нам нужно показать следующее.

Лемма 4.1. *Оптимальный путь $p(x)$ в точку $x \in e$ пересекает внутреннюю часть не более чем одной грани, содержащей x.*

Доказательство. Если x — вершина, то содержащих ее граней может быть несколько; если x — внутренняя точка e, то их ровно две. Пусть $f_1, f_2, \ldots, f_k$ — грани, содержащие x. Если $p(x)$ не пересекает внутреннюю часть ни одной из них, то все в порядке. В противном случае пусть f_i будет первой из этих граней, внутреннюю часть которой пересекает оптимальный путь $p(x)$, и пусть y — точка на $p(x)$, внутренняя для грани f_i. Тогда часть пути $p(x)$ от y до x должна представлять собой оптимальный путь. Но кратчайший путь из y в x на $\mathscr{S}$ есть отрезок прямой, соединяющий y и x, поскольку он целиком лежит на грани f_i и, значит, на поверхности $\mathscr{S}$. Следовательно, никакая другая грань $f_j \neq f_i$, содержащая x, не имеет общих внутренних точек с путем p. ■

Лемма 4.2 указывает нам способ построения оптимального пути во внутреннюю точку x ребра $e = f \cap f'$ по кратчайшему f-свободному и кратчайшему f'-свободному пути в x.

Лемма 4.2. *Оптимальный путь во внутреннюю точку x ребра e совпадает с кратчайшим из путей $p_f(x)$ и $p_{f'}(x)$.*

Лемма 4.3 устанавливает другое важное свойство кратчайших f-свободных путей: они не могут пересекать друг друга.

Лемма 4.3. *Если $p(x)$ и $p(y)$ — оптимальные пути из s в точки x и y, то они могут пересекаться только в вершинах $\mathscr{S}$, причем если они пересекаются в вершине v, то часть пути $p(x)$ от s до v имеет ту же длину, что и часть пути $p(y)$ от s до v. Аналогичное утверждение справедливо по отношению к f-свободным путям $p_f(x)$ и $p_f(y)$.*

Доказательство. Если $p(x)$ и $p(y)$ пересекаются в точке z, то, конечно, длина их частей до точки z должна быть одинакова (иначе один из них можно сделать короче, заменив его часть от точки s до z). Если бы z была внутренней точкой грани, то оба пути можно было бы локально улучшить с помощью обхода точки z. Аналогичное улучшение обоих путей было бы возможно, если бы z была внутренней точкой ребра (достаточно представить себе развертку путей и действовать тем же способом, как если бы z была внутренней точкой грани). Те же рассуждения применимы к кратчайшим f-свободным путям, поскольку подобные локальные улучшения не влияют на свойства обоих путей быть f-свободными. ■

Обратите внимание на то, что, хотя $p_f(x)$ и $p_f(y)$ не могут пересекаться, кратчайшие f-свободные пути для *различных* граней пересекаться могут (например, на рис. 5 $p_{f_1}(x)$ пересекается с $p_{f_2}(x)$). Проблема в том, что если мы попытаемся улучшить один из путей (скажем, $p_{f_1}(x)$) с помощью замены его части частью пути p_{f_2}, то он может перестать быть f_1-свободным.

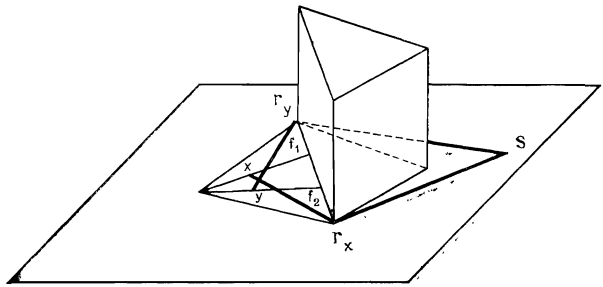


Рис. 5. Путь $p_{f_1}(x)$ может пересекать путь $p_{f_2}(x)$.

Важность введенного понятия кратчайших f-свободных путей для разрабатываемого нами алгоритма становится очевидной из следующей леммы.

Лемма 4.4. *Если заданы пара ребро — грань* (e, f), *реберная последовательность* $\mathscr{E}$ *и некоторая вершина* r [1], *то множество* $\mathscr{Y}$ *точек* x *на* e, *для которых существует кратчайший* f-*свободный путь в* x *с корнем* r *и последней реберной последовательностью* $\mathscr{E}$, *является связным* (*и, следовательно, представляет собой интервал, принадлежащий* e).

Доказательство. Если на e нет точек, в которые ведет кратчайший f-свободный путь с корнем r и последней реберной последовательностью $\mathscr{E}$, то доказывать нечего. В противном случае пусть x и x' — точки на e, такие, что $p_f(x)$ и $p_f(x')$ имеют корень r и последнюю реберную последовательность $\mathscr{E}$. Мы хотим доказать, что если y — точка на e, расположенная между x и x', то $p_f(y)$ также имеет корень r и последнюю реберную последовательность $\mathscr{E}$. Для этого построим развертку последовательности $\mathscr{E}$ и обозначим через $\bar{r} = U_{\mathscr{E}(r)}$ образ r при этой развертке. Поскольку, согласно лемме 3.3, образы путей $p_f(x)$ и $p_f(x')$ являются отрезками, отрезки $\overline{\bar{r}x}$, $\overline{\bar{r}x'}$ и $\overline{xx'}$ образуют на развертке треугольник. Никакой f-свободный путь $p_f(y)$ не может пересекать внутренность f; следовательно, он должен

[1]) Очевидно, что r является корнем $\mathscr{E}$: $r = r(\mathscr{E})$. — *Прим. перев.*

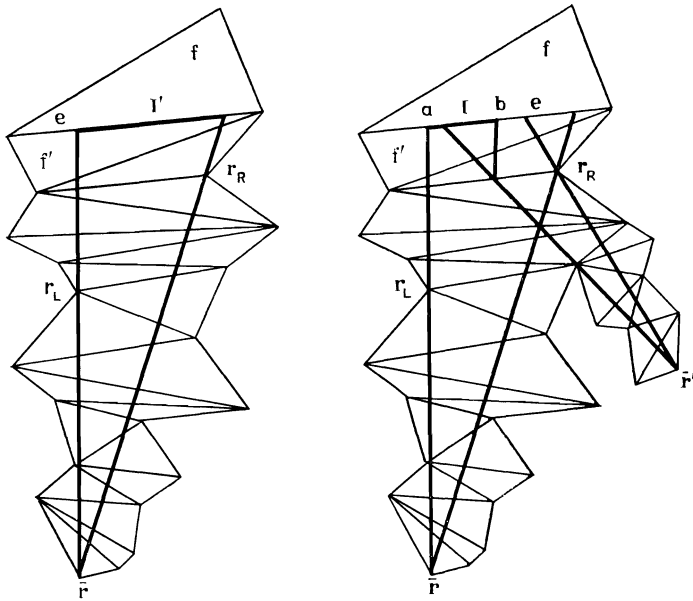


Рис. 6. Развертки и интервалы оптимальности.

проходить внутри указанного треугольника. Далее, поскольку кратчайшие $\hat{f}$-свободные пути не пересекаются (лемма 4.3), образ $p_{\hat{f}}(y)$ при развертке не может пересекать отрезки $\overline{\bar{r}x}$ и $\overline{\bar{r}x'}$ во внутренних точках. Поэтому образ $p_{\hat{f}}(y)$ должен пересекать треугольник в точке $\bar{r}$, а это и означает, что $p_{\hat{f}}(y)$ имеет корень r и последнюю реберную последовательность $\mathscr{E}$.

Интервал J может быть замкнутым, открытым или полуоткрытым. (Концевая точка может не принадлежать интервалу в вырожденном случае, при котором отрезок, представляющий собой развертку кратчайшего $\hat{f}$-свободного пути, проходит через две или более вершины (рис. 6). То, что результирующий интервал имеет при этом открытый конец, в действительности есть следствие данного нами определения корня пути как *первой* вершины, отсчитываемой при трассировке пути в обратном направлении.) Пусть отрезок $I = [a, b]$ представляет собой замыкание $\mathscr{I}$. (Обычно геометрический отрезок с концами a и b мы будем записывать как $I = \overline{ab}$; однако здесь мы пишем $[a, b]$, желая подчеркнуть тот факт, что точки ребра e можно снабдить координатами, и тогда интервал I можно рассматривать как интервал этих координат.) Точка a (соответственно b) представляет собой левый (соответственно правый) конец ин-

тервала, если смотреть на грань f с наружной относительно e стороны. Мы будем называть I *интервалом оптимальности* для r и $\mathscr{E}$ *относительно пары* (e, f). Альтернативное задание интервала оптимальности состоит в указании *образа корня при развертке* $\bar{r} = U_{\mathscr{E}}(r)$, его *глубины* $d = d(r)$ и соответствующей пары ребро — грань (e, f). Заметьте, что в вырожденном случае r может быть концевой точкой e (и, следовательно, I). (Условимся, что $\mathscr{E}$ не включает ребро e, так что $U_{\mathscr{E}}(r)$ есть представление корня r в системе координат грани f'. Если $\mathscr{E} = \varnothing$, то r будет одной из вершин f'.) Точка x является элементом интервала оптимальности для $\bar{r}$ относительно (e, f), если существует кратчайший f-свободный путь в x, образ которого при развертке (вдоль его последней реберной последовательности) в плоскости f содержит отрезок $\overline{\bar{r}x}$.

Примеры разверток и интервалов оптимальности приведены на рис. 6. Слева показан интервал I' точек ребра e, которые «достижимы» из корня r вдоль путей, проходящих через заданную реберную последовательность и являющихся при развертке прямыми линиями. Обратите внимание на то, что I' определяется левой и правой «загораживающими» вершинами r_L и r_R. (I' есть попросту та часть e, которая видима из r внутри многоугольника, получаемого разверткой последовательности смежных граней.) На правой стороне рисунка показан интервал оптимальности $I = [a, b]$. (Заметьте, что здесь $\mathscr{I} = (a, b]$ — это полуоткрытый интервал, поскольку корнем пути в a служит не r, а r_L.) Интервал слева от I есть просто тот, корнем которого является r_L. Правая концевая точка b находится как точка пересечения I' с гиперболической кривой раздела между вершинами $\bar{r}$ и $\bar{r}'$. Хотя точки справа от b и достижимы из r, более короткий путь к ним проходит через r'.

Легко видеть, что интервалы оптимальности относительно (e, f) образуют покрытие e и что они могут касаться, но не перекрывать друг друга.

Лемма 4.5. *Интервалы оптимальности относительно заданной пары ребро — грань* (e, f) *образуют покрытие* e, *причем их внутренние части не перекрываются.*

Доказательство. Прежде всего заметим, что каждая точка $x \in e$ должна принадлежать *хотя бы одному* интервалу оптимальности относительно (e, f) (если $p_f(x)$ — любой кратчайший f-свободный путь в x, то x принадлежит интервалу оптимальности для корня этого пути относительно (e, f)). Далее, если точка $x \in e$ принадлежит интервалам оптимальности относительно (e, f) для r и r', то она должна удовлетворять уравнению $d(r) + |\bar{r}x| = d(r') + |r'x|$. Но это уравнение задает гиперболу

(которая вырождается в прямую при $d(r) = d(r')$). Пересечение этой гиперболы с e есть точка (в вырожденном случае прямая не может содержать e, потому что тогда точки $\bar{r}$ и $\bar{r}'$ оказались бы по разные стороны от прямой, инцидентной e). Таким образом, не существует открытого подынтервала e, все точки которого принадлежат двум или более интервалам оптимальности. А это и означает, что последние не имеют общих внутренних точек. ■

Точка c на I, ближайшая к r, называется *фронтальной точкой* интервала. Поскольку минимум расстояния от точки r до замкнутого отрезка $[a, b]$ достигается ровно в одной точке, мы можем быть уверены, что такая точка c существует и единственна. Она может быть как концевой, так и внутренней точкой I в зависимости от того, лежит ли проекция перпендикуляра из r на прямую, инцидентную e, соответственно вне или внутри интервала I.

Пересечение f-свободного пути, идущего в c, с гранью f' (являющейся смежной с гранью f по ребру e) есть отрезок $\overline{\beta c}$ на грани f'. Мы будем называть β *точкой доступа* интервала I. Заметьте, что если c — вершина, то β может совпадать с c, так что $\overline{\beta c}$ вырождается в точку. Если нарисовать все отрезки $\overline{\beta c}$ для каждого интервала оптимальности относительно (e, f), то мы получим разбиение грани f' на *каналы доступа* (поскольку никакие два отрезка не могут пересекаться). Нетрудно видеть, что канал доступа представляет собой либо треугольник, либо четырехугольник, либо пятиугольник. Канал доступа мы будем обозначать, задавая пару (I_1, I_2), где I_1 и I_2 — интервалы оптимальности, определяющие граничные отрезки $\overline{\beta_1 c_1}$ и $\overline{\beta_2 c_2}$. Если $I_1 = \mathrm{NIL}$, то канал есть просто часть f', расположенная слева от $\overline{\beta_2 c_2}$; если же $I_2 = \mathrm{NIL}$, то канал — это часть f', расположенная справа от $\overline{\beta_1 c_1}$.

5. АЛГОРИТМ

Наш алгоритм работает в полном соответствии с духом алгоритма Дейкстры [1]. Он представляет собой важный пример мощности метода, который мы назвали «непрерывный Дейкстра». «Сигнал» распространяется из источника к остальным точкам поверхности. Если он попадает в точку x поверхности впервые, то ей присваивается *окончательная метка*, значение которой равно $d(x)$ — времени распространения сигнала (равному минимальному расстоянию от начальной точки до x), а сам сигнал передается дальше.

К счастью, такую разметку и дальнейшее распространение сигнала достаточно производить лишь для конечного числа (как мы покажем, для $O(n^2)$) точек поверхности, называемых *точками событий*. (Фактически мы даем «ориентированную» реализацию рассматриваемого метода, при которой точки ребра помечаются длинами путей, входящих в ребро с каждой из его сторон. Это следует из того, что интервалы оптимальности были определены по отношению к f-свободным путям.)

В алгоритме используется несколько простых структур данных. Мы заводим список ILIST для *кандидатов* в интервалы оптимальности. *Кандидатом в интервал* (или для краткости иногда просто *интервалом*) считается подынтервал ребра, который содержит некоторый (возможно, пустой) интервал оптимальности и имеет в точности ту же структуру, что и этот интервал (т. е. в его структуре данных содержится информация того же типа). Мы называем их *кандидатами*, поскольку на завершающей стадии алгоритма все оставшиеся кандидаты в интервалы будут уже интервалами оптимальности. С кандидатом в интервал I связана следующая информация: его замыкание $[a, b]$; пара ребро — грань (e, f); корень r; проекция корня при развертке $\bar{r}$; глубина корня d; фронтальная точка c; точка доступа β; предшественник $\bar{I}$, являющийся кандидатом в интервалы (или вершиной), «распространение» которого привело к созданию интервала I. В дальнейшем мы будем использовать запись $I = ([a, b], (e, f), r, \bar{r}, d, c, \beta, \bar{I})$.

Паре ребро — грань (e, f) ставится в соответствие упорядоченный список кандидатов в интервалы относительно (e, f), которые заносятся в порядке, определяемом направлением на ребре (направление на ребре выбирается так, чтобы грань f была расположена *слева*). Такое упорядочивание оказывается возможным благодаря тому, что внутренние части кандидатов в интервалы никогда не перекрываются. При инициации и в процессе работы алгоритма *интервалы из списка*, связанного с парой (e, f), необязательно *покрывают* ребро e (это означает, что существуют точки на e, не лежащие ни в одном из кандидатов в интервалы), но при завершении работы, как мы увидим, они будут образовывать покрытие.

С каждой вершиной v связываются указатели на всех кандидатов в интервалы, для которых она является корнем. Кроме того, ей присваивается *окончательная метка* $d(v)$, представляющая собой длину кратчайшего пути в вершину v. Исходно $d(v) = +\infty$ для каждой вершины v. По мере выполнения алгоритма вершины становятся *окончательно помеченными*: им присваиваются метки $d(v) < +\infty$. Мы будем также помечать

некоторые точки, не являющиеся вершинами (фронтальные точки, лежащие во внутренней части ребер). Будем считать, что и для них исходные значения меток равны бесконечности.

Далее, в алгоритме используется очередь с $\log n$ приоритетами (называемая *очередью событий*), входами в которую служат точки некоторого кандидата в интервалы (концевая или фронтальная точки) с уже присвоенными им метками, которые представляют собой длины кратчайших на данный момент путей, ведущих к источнику. (В действительности нет необходимости хранить информацию о всех концевых точках кандидатов в интервалы. Вполне достаточно было бы это делать для событий, связанных с вершинами и фронтальными точками, лежащими внутри интервалов. Однако принятая избыточность не влияет на оценку эффективности алгоритма.)

Алгоритм работает следующим образом.

Алгоритм

(0) (Инициализация.) Присвоить s окончательную метку $\emptyset$. Инициализировать очередь событий; список ILIST сделать пустым. Для каждой пары ребро — грань связанный с ней список интервалов сделать пустым. Для каждого ребра, противолежащего s, создать кандидата в интервалы, замыкание которого есть все ребро, а корень равен s. Найти точку c для каждого такого интервала и поставить ее вместе с концевыми точками ребра в очередь событий, причем каждая точка должна быть помечена ее расстоянием от s. Занести полученные интервалы в список ILIST и в списки интервалов, связанные с соответствующими парами ребро — грань.

(1) (Основной цикл.) Если очередь событий не пуста, выбрать из нее точку с наименьшей меткой и окончательно пометить ее. Если эта точка есть фронтальная точка некоторого кандидата в интервалы I, то выполнить процедуру *Propagate*(I) (распространение I).

Распространение интервала I интуитивно означает следующее: «волновой фронт» сигналов, идущих из корня интервала I, пропускается через I, прослеживается движение сигналов через два других ребра той же грани и на этих ребрах запоминаются те интервалы точек, которые лежат на оптимальных путях, проходящих через I. Один интервал может породить два (по одному на каждом из противолежащих ребер), поэтому при распространении интервалов необходимо соблюдать осторожность, чтобы не получить экспоненциального роста их числа. Это оказывается возможным благодаря тому, что оптимальные пути и кратчайшие f-свободные пути *не пересекаются* (лемма 4.3).

Отсюда также следует, что распространение интервалов может произойти лишь через каналы доступа, уже образованные интервалами на противолежащих ребрах. Теперь мы приведем детали процедуры Propagate(I).

Процедура Propagate(I)

(0) Допустим, что I — интервал относительно пары (e, f), где $e = f \cap f'$. Пусть $c \in e$ — фронтальная точка I. Пусть $e_1 = f \cap f_1$ и $e_2 = f \cap f_2$ — ребра грани f, противолежащие e (рис. 7).

(1) (c не является вершиной). Пусть $I_i = Project(I, e_i)$. Тогда, если $I_i \neq$ NIL, выполнить процедуру $Insert\text{-}Interval(I_i, c)$ для $i = 1, 2$.

(2) (c — вершина). Для каждого противолежащего вершине c ребра e_0, на котором найдется точка, составляющая угол, больший чем π, с отрезком $\overline{\beta c}$, создать кандидата в интервал I_0 на e_0, замыкание которого совпадает со всем ребром e_0, а корень есть c. Выполнить процедуру $Insert\text{-}Interval(I_0, c)$.

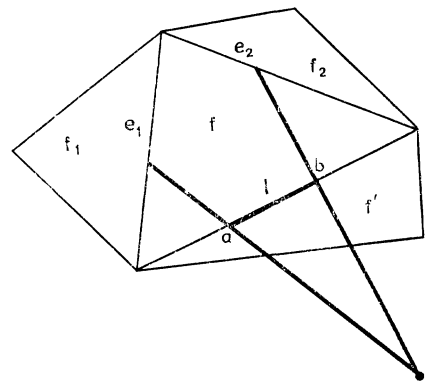


Рис. 7. Распространение интервала.

Чтобы определить, в каком случае целесообразно в качестве корня кандидата в интервал на противолежащем ребре принять вершину c, на шаге (2) применяется лемма 3.4. (Это не влияет на корректность алгоритма или его сложность, но позволяет несколько ускорить его работу.) Мы создаем интервалы только на тех ребрах, для которых угол при вершине c, образованный построенными в итоге путями, будет больше π.

Функция $Project(I, e_1)$ находит подмножество ребра e_1, на которое проектируются пути, идущие через I. Здесь для определения того, как «клинья» путей распространяются через следующую грань, используется критерий локальной оптимальности (заключающийся в том, что развертки путей представляют собой прямые — лемма 3.3). Функция возвращает кандидата в интервал I_A на ребре e_1, точки которого доступны при прохождении через I. (Если таких точек на e_1 нет, возвращается NIL.) Корень интервала I_A тот же, что и у интервала I, а проекция корня I_A получается из проекции корня I просто преобра-

зованием его в систему координат грани f' (поворотом вокруг e).

Процедура *Insert-Interval*(I, c) вставляет нового кандидата в интервал, I, между интервалами I_1 и I_2, которые определяют канал доступа, содержащий c. Интервал I может «поглотить» интервал I_1 или интервал I_2 (если только фронтальная точка интервала I_1 или соответственно интервала I_2 не была еще окончательно помечена). В этом случае поглощенный интервал должен быть удален. Кроме того, I, возможно, будет «обрезан» в «точках стыковки» между корнем интервала I и корнями интервалов I_1 и I_2. В результате та часть интервала, которая останется после обрезания, будет состоять из точек ребра, которые достигаются первыми при распространении сигналов из корня интервала I через последнюю реберную последовательность этого интервала.

Процедура Insert-Interval(I, c)

(0) Допустим, что $I = [a, b]$ — интервал относительно пары (e, f). Пусть он имеет корень r с проекцией $\bar{r}$ и глубиной d. Точка c — фронтальная точка интервала $\bar{I}$, предшественника I.

(1) **(Определение положения точки c в канале доступа.)** Найти канал доступа, которому принадлежит точка c. Пусть он определяется интервалами (I_1, I_2) относительно пары (e, f). Фронт сигнала может попасть в I только через этот канал. Пусть $I_i = [a_i, b_i]$, $i = 1, 2$ (при условии, что $I_i \neq \mathrm{NIL}$), и пусть r_i — корень, $\bar{r}_i$ — его проекция, d_i — глубина.

(2) **(Удаление поглощенных кандидатов в интервалы.)** Если $I_1 \neq \mathrm{NIL}$, $a_1 \in I$ и $d_1 + |\bar{r}_1 a_1| \geqslant d + |\bar{r} a_1|$, то выполнить *Delete*$(I_1)$, включить I_1 в качестве предшественника I в список интервалов ребра e_1 (если предшественника нет, то I_1 положить равным NIL) и вернуться на начало шага (2). Аналогично, если $I_2 \neq \mathrm{NIL}$, $b_2 \in I$ и $d_2 + |\bar{r}_2 a_2| \geqslant d + |\bar{r} a_2|$, то выполнить *Delete*$(I_2)$, включить I_2 в качестве преемника I в списке интервалов ребра e_1 (если преемника нет, то I_2 положить равным NIL) и вернуться на начало шага (2).

(3) **(Обрезание в точках стыковки.)** Если $I_1 \neq \mathrm{NIL}$ и $b_1 \in I$, то найти точку a, принадлежащую $I \cap I_1$, такую, что $d_1 + |\bar{r}_1 a| = d + |\bar{r} a|$ (если такой точки не существует, то занести в a точку b_1). Точка a достигается за одинаковое время как при прохождении через интервал I, так и через интервал I_1. Удалить b_1 из очереди событий, положить b_1 равной a и, если нужно, перевычислить точку c_1. Если $I_2 \neq \mathrm{NIL}$ и $a_2 \in I$, то найти точку b, принадлежащую $I \cap I_2$, такую что $d_2 + |\bar{r}_2 b| = d + |\bar{r} b|$ (если такой точки не существует, то занести в b точку a_2). Точка b достигается за одинаковое время как при прохождении через

интервал I, так и через интервал I_2. Удалить a_2 из очереди событий, положить a_2 равной b и, если нужно, перевычислить точку c_2. Если $a \neq b$, то перейти к шагу (4); иначе — останов (никакого интервала вставлять не нужно).

(4) (Окончательные корректировки.) Вычислить фронтальную точку c_I и точку доступа для интервала $I = [a, b]$. Занести I

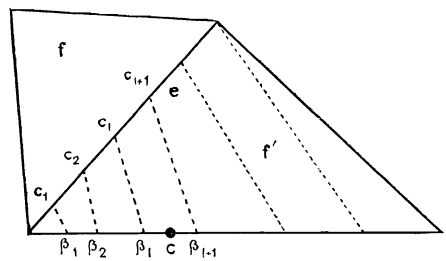


Рис. 8. Определение положения точки c в канале.

в список ILIST и вставить его в список интервалов ребра e (между интервалами I_1 и I_2). Занести точки a, b и c_I (если $c_I \neq a$, b) в очередь событий, снабдив их соответственно метками $d + |\bar{r}a|$, $d + |\bar{r}b|$ и $d + |\bar{r}c_I|$.

Процедура *Delete*(I) удаляет I из списка ILIST и из списка интервалов, связанного с соответствующей парой ребро — грань, а также удаляет все точки, принадлежащие I, из очереди событий.

На шаге (1) приведенной выше процедуры нам необходимо найти положение точки c в канале доступа пары (e, f). Это легко сделать посредством бинарного поиска следующим образом. Мы имеем список интервалов для (e, f) с хранящимися в нем отрезками $\overline{\beta_i c_i}$, которые упорядочены вдоль ребра e. Отрезки не пересекаются. Поэтому для нахождения положения точки c в канале доступа грани f' можно произвести бинарный поиск, в процессе которого каждая проверка определяет, по какую сторону от отрезка $\overline{\beta_i c_i}$ точка c расположена (рис. 8).

6. ДОКАЗАТЕЛЬСТВО КОРРЕКТНОСТИ

Теперь мы приступим к доказательству работоспособности алгоритма. Наша первая цель состоит в доказательстве следующего предложения.

Предложение 6.1. *После каждой итерации основного цикла алгоритма в списке кандидатов в интервал* (ILIST) *содержатся*

интервалы, отвечающие кратчайшим на текущий момент f-свободным путям из s в точки этих интервалов, где «кратчайший на текущий момент» означает следующее: если точка $x \in e = f \cap f'$ принадлежит интервалу I для корня r относительно пары (e, f), то f-свободный путь в x, имеющий корень r и длину $d(r) + |\bar{r}x|$, является оптимальным среди тех f-свободных путей в x, которые пересекают границу грани f' через интервал, содержащийся в текущем списке ILIST.

Доказательство. Доказательство производится индукцией по счетчику основного цикла. На первой итерации доказываемое утверждение справедливо, поскольку единственными кандидатами служат тривиальные интервалы, расположенные на ребрах, противолежащих источнику s. Допустим теперь, что утверждение справедливо для первых k циклов (предположение индукции). Для индуктивного перехода нам необходимо доказать, что если на $(k+1)$-й итерации мы вводим в рассмотрение интервал I для корня r относительно пары (e, f), то кратчайший из всех построенных к настоящему моменту f-свободных путей пройдет через r. Это будет показано в двух приведенных ниже леммах. ■

Допустим теперь, что индуктивное предположение выполняется.

Судьба кандидата в интервалы может быть двоякой: либо его фронтальная точка становится окончательно помеченной (при этом она оказывается в вершине очереди событий), тем самым обеспечивая «выживаемость» некоторой его части, которая войдет в окончательный интервал оптимальности с теми же атрибутами, что и сам кандидат, либо кандидат в интервалы удаляется на шаге 2 процедуры *Insert-Interval*; в этом случае соответствующий этому кандидату интервал оптимальности окажется пустым.

Лемма 6.2. *Если на шаге* (2) *процедуры Insert-Interval удаляется интервал I_1, то интервал оптимальности для корня r_1 относительно пары (e, f) не имеет внутренних точек. Аналогичный вывод относится и к результату удаления интервала I_2.*

Доказательство (рис. 9). Допустим, что $a_1 \in I$ и $d_1 + |\bar{r}_1 a_1| \geqslant d + |\bar{r}a_1|$. Тогда кратчайший на текущий момент f-свободный путь в точку a_1 проходит через корень r. Пусть β есть пересечение отрезка $\bar{r}a$ с ребром e'. Поскольку $a_1 \in I$, то, согласно способу построения I по его предшественнику $\bar{I}$, мы знаем, что $\beta \in \bar{I}$. Предположим, что существует внутренняя точка y кандидата в интервалы для корня r_1 относительно (e, f)

(так что путь $p_f(y)$ имеет корень r_1). Это приведет нас к противоречию.

Во-первых, заметим, что y есть внутренняя точка отрезка $[a_1, b_1]$ (иначе y либо недостижима из r_1, либо может быть достигнута из некоторого другого корня пары (e, f) по более короткому f-свободному пути). Далее, вспомним, что интервал

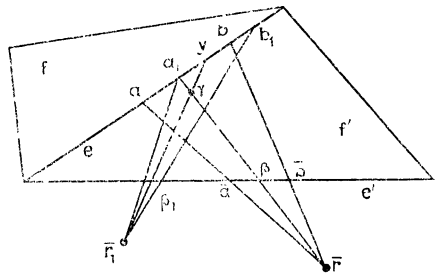


Рис. 9. Иллюстрация к доказательству леммы 6.2.

I_1 был создан в результате распространения интервала $\bar{I}_1$. Поскольку точка $c \in \bar{I}$ расположена в канале (I_1, I_2), то c лежит справа от отрезка $\overline{\bar{r}_1 c}$, а потому справа и от отрезка $\overline{\bar{r}_1 b}$ (иначе $c \in \bar{I}_1$). Поскольку множество $\bar{I}$ связно, все точки $\bar{I}$ лежат справа от отрезка $\overline{\bar{r}_1 z}$ для любого $z \in [a_1, b_1]$ и лежат строго справа от отрезка $\overline{\bar{r}_1 z}$ для любой внутренней точки z отрезка $[a_1, b_1]$. В частности, β лежит строго справа от отрезка $\overline{\beta_1 y}$, где $\beta_1 y$ есть пересечение $\overline{\bar{r}_1 y}$ с гранью f'. Но a_1 лежит строго слева от отрезка $\overline{\beta_1 y}$, так что отрезки $\overline{\beta a_1}$ и $\overline{\beta_1 y}$ должны пересекаться в некоторой точке γ, которая является внутренней для каждого из них (рис. 9). Поскольку γ принадлежит пути $p_f(y)$, f-свободный путь через r_1 в γ не длиннее f-свободного пути через r в γ. Но это означает, что мы можем укоротить путь в a_1, пройдя в a_1 напрямую через корень r_1 (поскольку $|\bar{r}_1\gamma| + |\gamma a_1| > |\bar{r}_1 a_1|$). Это противоречит нашему предположению, что $d_1 + |\bar{r}_1 a_1| \geqslant d + |\bar{r} a_1|$.

Аналогичное рассуждение справедливо и в случае удаления интервала I_2. ■

Теперь мы должны доказать, что интервал, вставленный процедурой *Insert-Interval*, «корректен» в следующем смысле.

Лемма 6.3. *После вставки процедурой Insert-Interval интервала I между интервалами I_1 и I_2 получающийся список интервалов для ребра e правильно отражает картину разбиения точек e в соответствии с кратчайшими на текущий момент f-свободными путями.*

Доказательство. Пусть $[a_1, b_1]$ и $[a_2, b_2]$ обозначают первоначальные границы интервалов I_1 и I_2, а через $[a_1, b_1']$ и $[a_2', b_2]$ обозначим интервалы, полученные в результате обрезания после выполнения процедуры *Insert-Interval*. Кратчайший на текущий момент путь в точку, лежащую вне отрезка $[a_1, b_2]$, не может проходить через r, поскольку в противном случае он должен был бы пересечь граничный отрезок $\overline{\beta_i c_i}$ канала доступа.

Пусть теперь $x \in I = [a, b]$, где $[a, b]$ — замыкание I после выполнения всех необходимых операций обрезания. Поскольку по построению $I \subseteq [a_1, b_2]$, то кратчайший на текущий момент путь в x должен проходить через r, r_1 или r_2. Если точка x не принадлежит $[a_1, b_1]$, то более короткий путь в нее проходит через корень r, а не через r_1. Если x — внутренняя точка $[a_1, b_1]$, то и a является внутренней точкой (поскольку $[a, b] \subseteq [a_1, b_2]$). Но тогда путь в a через корень r не длиннее, чем путь в a через корень r_1, и поэтому он является на текущий момент оптимальным. Если бы кратчайший путь в x проходил через корень r_1, а не через корень r, то он пересек бы путь, проходящий через r в a (поскольку отрезки $\overline{ra}$ и $\overline{r_1 x}$ должны пересекаться). Приходим к противоречию. Аналогично можно показать, что кратчайший путь в x не может проходить через r_2.

Если $x \in [a_1, b_1']$ (соответственно $[a_2', b_2]$), те же рассуждения показывают, что кратчайший путь в x проходит через r_1 (соответственно r_2), а не через r (и, конечно, лучше идти в x через r_1 (соответственно r_2), а не через r_2 (соответственно r_1), поскольку $[a_1, b_1'] \subseteq [a_1, b_1]$ и $[a_2', b_2] \subseteq [a_2, b_2]$). ■

Этим завершается доказательство предложения 6.1.

Можно показать, что если в процессе выполнения алгоритма Дейкстры вершина оказывается «окончательно помеченной» (т. е. занесенной в список CLOSED), то ее метка равна длине кратчайшего пути от источника. Аналогично, справедлива следующая лемма.

Лемма 6.4. *Если точка, расположенная внутри ребра, оказывается окончательно помеченной, являясь при этом частью кандидата в интервал относительно пары* (e, f), *то ее метка равна длине кратчайшего* f*-свободного пути в нее. При присвоении окончательной метки вершине ее значение равно длине кратчайшего пути в вершину.*

Доказательство. Пусть x есть внутренняя точка ребра $e = f \cap f'$. Предположим, что ей только что была присвоена окончательная метка δ и что длина кратчайшего f-свободного

пути $p_f(x)$ меньше δ. Поскольку $p_f(x)$ пересекает по крайней мере одного кандидата в интервал (ведь во время инициализации списка ILIST они «окружают» точку s), мы можем обозначить последнего такого кандидата, скажем, через $I_l = [a_l, b_l]$ и считать, что он является кандидатом в интервалы относительно пары (e_l, f_l). (Заметьте, что в случае, когда $p_f(x)$ пересекает I_l в концевых точках, таких кандидатов может быть несколько. В этом случае можно выбрать любого из них. Если $p_f(x) \cap I_l = v$ — вершина, нужно выбрать такой I_l, чтобы $p_f(x) \cap f_l \neq v$.) Ввиду того что длина $p_f(x)$ меньше δ, первая общая точка пути $p_f(x)$ и грани f' не может принадлежать ни одному кандидату в интервалы (это вытекает из предложения 6.1, поскольку x была помечена длиной кратчайшего на текущий момент пути). Таким образом, $f_l \neq f'$. Пусть $\overline{\beta x'} = f_l \cap p_f(x)$. Тогда точка x' принадлежит ребру $e_0 = f_l \cap f_0$. Обозначим через $p_{f_0}(x')$ часть пути $p_f(x)$ от s до x'. Ясно, что $p_{f_0}(x')$ есть кратчайший f_0-свободный путь в x'. Интервал I_l *должен* был быть распространен (поскольку расстояние до фронтальной точки I_l меньше или равно $d(\beta) < d(x) < \delta$, а окончательная метка присваивается алгоритмом всегда ближайшей точке событий). Но тогда, согласно лемме 6.3, на ребре e_0 должен был бы существовать кандидат в интервалы, содержащий точку x'. Это противоречит тому, что I — *последний* из кандидатов в интервалы, которые пересекают путь $p_f(x)$.

Аналогичное рассуждение позволит показать, что окончательные метки, присваиваемые вершинам, равны длинам кратчайших путей в них. ■

В нашей воображаемой картине процесса исполнения алгоритма мы считаем, что «волновой фронт» распространяется из источника s, и, по мере того как он захватывает новые ребра или границы кандидатов в интервалы, происходят события. Если на определенном расстоянии от s происходит событие, то, согласно приведенной ниже лемме 6.5, все точки, расположенные ближе к s, оказываются уже охваченными фронтом.

Лемма 6.5. *Если некоторой точке алгоритмом только что была присвоена окончательная метка* δ, *то для любой точки* x *ребра* e, *такой что* $d(x) < \delta$, *в списке* ILIST *найдется интервал* I (*для корня* r *относительно пары* (e, f)), *содержащий эту точку, причем оптимальный путь в* x *имеет корень* r *и длину* $d(r) + |\bar{r}x|$.

Доказательство. Пусть $x \in e$ и $d(x) < \delta$. Допустим, что x не принадлежит кандидату в интервалы, который определяет оптимальный путь в эту точку. Как и при доказательстве леммы 6.4,

обозначим через $I_l = [a_l, b_l]$ последний из кандидатов в интервалы текущего списка ILIST, пересекаемый путем $p(x)$, и допустим, что I_l есть интервал относительно пары (e_l, f_l). Из нашего предположения относительно x следует, что $x \notin I_l$. Пусть $\overline{\beta x'} = f_l \cap p(x)$. Тогда точка x' лежит на ребре $e_0 = f_l \cap f_0$. Пусть $p_{f_0}(x')$ есть часть пути $p(x)$ от s до x'. Ясно, что $p_{f_0}(x')$ представляет собой кратчайший f_0-свободный путь в x'. Интервал I_l *должен* был быть распространен (поскольку расстояние до фронтальной точки I_l меньше или равно $d(\beta) < d(x) < \delta$, а окончательная метка присваивается алгоритмом всегда ближайшей точке событий). Но тогда, согласно лемме 6.3, на ребре e_0 должен был бы существовать кандидат в интервалы, содержащий точку x'. Это противоречит тому, что I — последний из кандидатов в интервалы, которые пересекает путь $p(x)$. ■

По завершении работы алгоритма волновой фронт охватит *все* точки поверхности $\mathscr{S}$:

Лемма 6.6. *По завершении работы алгоритма список интервалов относительно пары (e, f) образует покрытие ребра e.*

Доказательство. Допустим, что существует точка $x \in e = f \cap f'$, которая не принадлежит ни одному из кандидатов в интервалы относительно (e, f). Пусть $p_f(x)$ — кратчайший f-свободный путь в x. Как и при доказательстве леммы 6.4, обозначим через $I_l = [a_l, b_l]$ последний из кандидатов в интервалы, пересекаемый путем $p_f(x)$, и допустим, что I_l — интервал относительно пары (e_l, f_l). Поскольку точка x не покрыта ни одним из интервалов, имеем $e_l \neq e$. Пусть $\overline{\beta x'} = f_l \cap p_f(x)$. Тогда точка x' лежит на ребре $e_0 = f_l \cap f_0$. Пусть $p_{f_0}(x')$ есть часть пути $p(x)$ от s до x'. Ясно, что $p_{f_0}(x')$ представляет собой кратчайший f_0-свободный путь в x'. Интервал I_l должен был быть распространен (иначе его фронтальная точка не была бы удалена из очереди событий, и алгоритм никогда бы не завершился). Но тогда, согласно лемме 6.3, на ребре e_0 должен был бы существовать кандидат в интервалы, содержащий точку x'. Это противоречит тому, что I — *последний* из кандидатов в интервалы, которые пересекают путь $p_f(x)$. ■

Лемма 6.7. *Следующие два утверждения относительно точки x, лежащей на ребре e грани f, эквивалентны.*

(1) *По завершении работы алгоритма x принадлежит кандидату в интервалы для корня r (с проекцией $\bar{r}$) относительно пары (e, f).*

(2) *Существует кратчайший f-свободный путь из s в x с корнем r длиной $d(r)+|\bar{r}x|$; иными словами, x принадлежит интервалу оптимальности для корня $\bar{r}$ относительно пары (e, f).*

Доказательство. Доказательство следует непосредственно из предложения 6.1 и леммы 6.6. ■

7. АНАЛИЗ СЛОЖНОСТИ

Чтобы установить время работы алгоритма, нам прежде всего необходимо найти верхнюю границу числа событий.

Лемма 7.1. *Алгоритмом генерируется не более $O(n^2)$ кандидатов в интервалы.*

Доказательство. Пусть $I_1, I_2, \ldots, I_K$ — список интервалов относительно некоторой пары ребро — грань (e, f). Обозначим через x_i внутреннюю точку интервала I_i, а ребро e представим в виде отрезка $e = x_0x_{K+1}$. Проведем теперь кратчайшие f-свободные пути в каждую из точек x_i $(0 \leqslant i \leqslant K+1)$. (Напомним, что ни один из этих путей не пересекает ребро e.) Ясно, что в результате мы получаем разбиение поверхности $\mathscr{P}$ на $K+1$ частей (каждая часть представляет собой область, ограниченную замкнутой кривой, идущей из s в x_i (вдоль $p_f(x_i)$), из x_i в x_{i+1} (вдоль e) и затем из x_{i+1} назад в s (вдоль $p_f(x_{i+1})$, но в противоположном направлении). Заметьте, что пути $p_f(x_i)$ и $p_f(x_{i+1})$ могут иметь некоторые общие вершины. Внутри каждой из указанных частей должна находиться по крайней мере одна вершина [1]). Следовательно, $K \leqslant n-1$. Поскольку пар ребро — грань не более чем $O(n)$, то кандидатов в интервалы может быть не более чем $O(n^2)$.

Предыдущее рассуждение применимо лишь к многогранникам нулевой связности, поскольку в нем неявно использовалась теорема Жордана о кривой, из которой вытекало, что число частей разбиения поверхности равно $K+1$. В случае ненулевой связности можно применить следующее альтернативное доказательство. Будем прослеживать пути из x_i и x_{i+1} в обратном направлении, отмечая места, где они впервые начинают расходиться, проходя через различные ребра (или где один из путей пройдет через вершину — его корень). Определим *точку ветвления* v_i как вершину, ответственную за это расхождение. Она может быть корнем одного из путей или вершиной между двумя соседними ребрами, внутренние части которых пересекаются рассматриваемыми путями. Обозначим через f_i грань, на которой

[1]) Корень соответствующего интервала. — *Прим. перев.*

началось расхождение. Таким образом, оба пути, $p_f(x_i)$ и $p_f(x_{i+1})$, проходят по внутренней части f_i, а v_i есть вершина f_i, которая «расщепляет» эти пути (она может лежать на одном из них). В такой ситуации мы «припишем» интервал I_i паре вершина — грань (v_i, f_i). Нетрудно видеть, что к каждой паре вершина — грань можно приписать не более двух интервалов: если v_i является корнем одного из путей (скажем, $p_f(x_i)$), то паре (v_i, f_i) могут соответствовать максимум два интервала (один, относящийся к паре путей $p_f(x_{i-1})$, $p_f(x_i)$, и другой, относящийся к путям $p_f(x_i)$, $p_f(x_{i+1})$); в противном случае — только один. (Доказательство последнего в сущности сводится просто к утверждению, что кратчайшие f-свободные пути не пересекаются.) Между парами вершина — грань и ребрами существует взаимно однозначное соответствие. Поэтому каждому ребру может быть приписано не более двух интервалов. Это приводит к линейной оценке числа K в зависимости от числа *ребер* поверхности $\mathcal{S}$ (эта оценка, видимо, является более подходящей мерой сложности для многогранников с высоким порядком связности). ■

Мы можем также оценить число обращений к процедуре *Delete* внутри процедуры *Insert-Interval*.

Лемма 7.2. *Число обращений к процедуре Delete будет не более чем* $O(n^2)$.

Доказательство. Согласно лемме 6.2, мы никогда не уничтожаем интервал, если какой-либо его точке присвоена окончательная метка. Если интервал исключается, то распространения его в дальнейшем не происходит. В случае распространения интервала его фронтальной точке присваивается окончательная метка и создается не более двух новых интервалов на противолежащих сторонах грани. Эти два интервала являются лишь кандидатами в интервалы оптимальности, и поэтому в дальнейшем они могут быть исключены. Но таких кандидатов может быть не более $O(n^2)$, поскольку, согласно лемме 7.1, число интервалов с точками, имеющими окончательную метку, не превышает $O(n^2)$. ■

Легко видеть, что объем структур, необходимых для работы алгоритма, квадратично зависит от n.

Лемма 7.3. *Объем памяти, необходимый для размещения структур данных, не превышает* $O(n^2)$.

Доказательство. Каждый из $O(n^2)$ кандидатов в интервалы требует постоянного объема памяти. (Напомним, что мы *не* храним реберную последовательность, соответствующую каж-

дому интервалу, а запоминаем только образ ее корня.) Поскольку каждое ребро инцидентно лишь двум граням, имеется не более $O(n)$ пар ребро — грань, с каждой из которых связан список не более чем $O(n)$ кандидатов в интервалы (см. доказательство леммы 7.1). Следовательно, структура данных для хранения пар ребро — грань требует объема памяти лишь $O(n^2)$. Очередь событий в худшем случае может содержать утроенное число кандидатов в интервалы [1], так что и эта структура ограничена размером $O(n^2)$. ■

Теперь мы можем дать окончательное заключение о корректности предложенного алгоритма и о заявленных показателях его эффективности.

Теорема 7.4. *Вышеприведенный алгоритм корректно строит разбиение каждого ребра на интервалы оптимальности и при этом требует $O(n^2 \log n)$ времени и $O(n^2)$ памяти.*

Доказательство. Корректность следует из леммы 6.7. Оценка памяти обоснована в лемме 7.3. Что касается оценки времени, то, согласно лемме 7.1, число событий в очереди не превосходит $O(n^2)$. Поскольку каждое событие вставляется или удаляется не более одного раза, полное время, необходимое для операций с очередью, составляет $O(n^2 \log n)$. Кроме того, по лемме 7.1 число вызовов процедуры *Propagate* оценивается как $O(n^2)$. Если фронтальная точка, которой присваивается метка, не является вершиной, то время работы процедуры *Propagate* составляет $O(\log n)$ (это время расходуется на определение канала доступа, которому принадлежит c), а затем происходит вызов процедуры *Insert-Interval*. На каждый вызов этой процедуры расходуется постоянное время, за исключением того случая, когда вызывается процедура *Delete*. Но вызов процедуры *Delete* требует максимум $O(\log n)$ времени (на удаление элемента из списка интервалов); следовательно, по лемме 7.2 суммарное время составляет $O(n^2 \log n)$. В случае когда фронтальная точка есть вершина, само по себе обращение к процедуре *Propagate* могло бы потребовать $O(n \log n)$ времени (поскольку число граней, инцидентных c, может составлять $O(n)$, а каждое обращение к процедуре *Insert-Interval* занимает $O(\log n)$ времени). Однако общее число вызовов процедуры *Insert-Interval* не может превзойти числа пар (v, e), где v — вершина, а e — ребро, противоположное v. Ясно, что таких пар не более $O(n)$. Таким образом, суммарное время, необходимое

[1] Поскольку с каждым интервалом связано не более трех событий. — *Прим. перев.*

на обращение к процедуре *Insert-Interval*, не превосходит $O(n^2 \log n)$. Это дает общую оценку времени работы алгоритма в виде $O(n^2 \log n)$. ■

8. ПОСТРОЕНИЕ РАЗБИЕНИЯ

Предложенный нами алгоритм позволяет найти длину $d(x)$ кратчайшего пути из s в любую точку x ребра $e = f \cap f'$. Если x — вершина, то $d(x)$ есть просто окончательно присвоенная ей метка. В противном случае мы находим (за время $O(\log n)$) соответствующий точке x интервал оптимальности I (соответственно I') для корня r (соответственно r') относительно пары (e, f) (соответственно (e, f')). Согласно лемме 4.2, наименьшая из двух сумм $d(r) + |x\bar{r}|$ и $d(r') + |xr'|$ и является искомой длиной кратчайшего пути.

Мы можем обобщить наш алгоритм для нахождения наикратчайшего пути из s в любую точку многогранной поверхности следующим образом. Пусть e_1, e_2 и e_3 — три ребра грани f. Построим три разбиения ($\mathcal{T}_1$, $\mathcal{T}_2$ и $\mathcal{T}_3$) этой грани. Разбиение $\mathcal{T}_i$ представляет собой разбиение грани f на ячейки, связанные с интервалами оптимальности относительно (e_i, f).

Обозначим через $I_1, I_2, \ldots, I_K$ список интервалов относительно пары (e_1, f). Ячейка C_j разбиения $\mathcal{T}_1$, связанная с соответствующим интервалом оптимальности I_j $(1 \leqslant j \leqslant K)$, определяется как множество точек x, таких что существует путь (на $\mathcal{P}$) из s в x через интервал I_j (т. е. через корень I_j и его реберную последовательность), являющийся кратчайшим среди всех путей (на $\mathcal{P}$) из s в x, пересекающих ребро e_1.

Пусть r_j — корень, соответствующий интервалу I_j. Тогда точка x будет лежать на границе ячеек C_j и $C_{j'}$ в том и только в том случае, когда $d(r_j) + |x\bar{r}_j| = d(r_{j'}) + |x\bar{r}_{j'}|$, где $\bar{r}_j = U_{(\mathcal{E}_{j'}, e)}(r_j)$ есть образ корня r_j при развертке в плоскости f (это несколько отличается от принятого нами до сих пор соглашения о том, что $\bar{r} = U_{\mathcal{E}}(r)$ есть образ r при развертке в плоскости, расположенной по другую, чем f, сторону ребра e_1). Иными словами, «*взвешенное* расстояние» [1]) от x до $\bar{r}_j$ должно равняться аналогичному «расстоянию» от x до $\bar{r}_{j'}$. Под «взвешенным расстоянием» (имеющим *вес* d) между точкой x и другой точкой y мы подразумеваем сумму $d + |xy|$. Из аналитической геометрии известно, что геометрическое место точек, «взвешенные расстояния» от которых до двух фиксированных точек

[1]) В ориг. weighted distance. Аксиомам расстояния указанное выражение не удовлетворяет. Поэтому этот термин взят в кавычки. — *Прим. перев.*

r и r' равны, представляет собой гиперболу. Таким образом, границами ячеек разбиения являются гиперболические дуги.

Теперь мы опишем процедуру *Build-Subdivision* (e, f), которая строит разбиение грани f на ячейки C_j, определяющие кратчайшие пути через e к точкам x на f. Суть процедуры сводится к «распространению» интервалов оптимальности $(I_1, I_2, \ldots, I_K)$ относительно пары (e, f) во внутреннюю часть грани f с прослеживанием определенных точек «событий» (которые будут точками пересечений гиперболических дуг с границами ячеек).

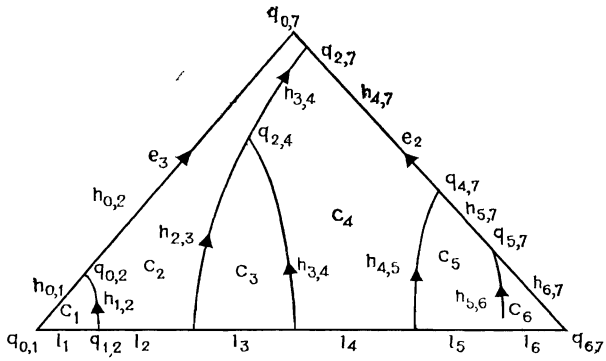


Рис. 10. Построение разбиения внутренней части грани f.

Обозначим через $h_{i,j}$ ветвь гиперболы, содержащую границу между ячейкой C_i (связанной с интервалом $I_i = [a_i, b_i]$) и ячейкой C_j (связанной с интервалом $I_j = [a_j, b_j]$). Она определяется уравнением $d(r_i) + |x\bar{r}_i| = d(r_j) + |x\bar{r}_j|$ относительно переменной x (в системе координат грани f). Заметим, что если $h_{i,j}$ существует, то с необходимостью $|d(r_i) - d(r_j)| \leqslant |\bar{r}_i\bar{r}_j|$, причем в случае равенства гипербола вырождается в прямую. Мы будем рассматривать только ту часть $h_{i,j}$, которая лежит в полуплоскости, определяемой прямой, проходящей через e и содержащей грань f. Кроме того, будем считать $h_{i,j}$ *ориентированной* в направлении увеличения расстояния от r_i и r_j. Через $h_{0,1}$ обозначим луч, идущий из a_1 вдоль ребра e_3, а через $h_{K,K+1}$ — луч, идущий из b_K вдоль ребра e_2. Если ячейки C_i и C_j имеют общую границу, то через $q_{i,j}$ $(0 \leqslant i < j \leqslant K+1)$ обозначим точку этой границы, ближайшую к точке r_i (и r_j). На рис. 10 приведен пример для случая, когда $K = 6$. Наконец, пусть u_i — верхняя грань расстояний от s до точек C_i (u_i может быть равно $+\infty$).

Процедура Build-Subdivision (e, f)

(0) Инициализировать список INTLIST: $(I_0, I_1, I_2, \ldots, I_K, I_{K+1})$. Для каждого i $(1 \leqslant i \leqslant K)$ положить $u_i = d_i + |\bar{r}_i w_i|$[1]), где w_i — точка пересечения $h_{i-1,\,i}$ и $h_{i,\,i+1}$ (в случае, если они пересекаются), и положить $u_i = +\infty$ в противном случае. Величины u_i занести в очередь с приоритетами ULIST.

(1) **(Определение следующего события.)** Если очередь ULIST пуста (это означает, что список INTLIST имеет вид (I_0, I_{K+1})), то выход из процедуры. В противном случае введем индексацию элементов INTLIST: $(I_0, I_{i_1}, \ldots, I_{i_J}, I_{K+1})$. Пусть $u = u_{ij}$ — элемент из списка ULIST с минимальным значением (если значения совпадают, то выбираем элемент с наименьшим индексом i_j). Удалить I_{i_j} из INTLIST, а u_{i_j} из ULIST.

(2) **(Корректировка $u_{i_{j-1}}$ и $u_{i_{j+1}}$.)** Если $j \neq 1$, то $u_{i_{j-1}}$ положить равным $d_{i_{j-1}} + |\bar{r}_{i_{j-1}} w_{j-1}|$, где w_{j-1} — точка пересечения $h_{i_{j-2},\,i_{j-1}}$ и $h_{i_{j-1},\,i_{j+1}}$, если они пересекаются, и положить $u_{i_{j-1}} = +\infty$ в противном случае. Если $j \neq J$, то $u_{i_{j+1}}$ положить равным $d_{i_{j+1}} + |\bar{r}_{i_{j+1}} w_{j+1}|$, где w_{j+1} есть точка пересечения $h_{i_{j-1},\,i_{j+1}}$ и $h_{i_{j+1},\,i_{j+2}}$, если они пересекаются, и положить $u_{i_{j+1}} = +\infty$ в противном случае. Дуги $h_{i_{j-1},\,i_j}$ и $h_{i_j,\,i_{j+1}}$ перестают быть «активными», а дуга $h_{i_{j-1},\bar{r}_{i_{j+1}}}$, наоборот, «активизируется» на этом этапе.

(3) **(Корректировка разбиения.)** Пусть $q_{i_{j-1},\,i_{j+1}}$ — точка событий, соответствующая u_{i_j}. (Точка $q_{i_{j-1},\,i_{j+1}}$ есть точка ячейки C_{i_j}, наиболее удаленная от r_{i_j}.) Установить $q_{i_{j-1},\,i_{j+1}}$ в качестве одной из вершин разбиения, через которую проходят дуги $h_{i_{j-1},\,i_j}$, $h_{i_j,\,i_{j+1}}$ (входящие в $q_{i_{j-1},\,i_{j+1}}$) и $h_{i_{j-1},\,i_{j+1}}$ (выходящая из $q_{i_{j-1},\,i_{j+1}}$). Перейти к шагу (1).

Процедура вычисляет вершины разбиения $\mathcal{T}_1$ в порядке их удаленности от источника. Каждое событие соответствует смыканию границ ячейки C_{i_j} в момент, когда мы удаляем I_{i_j} из списка интервалов INTLIST, определяющих фронт волны. При этом мы следим за дугой гиперболы, являющейся границей

[1]) Здесь авторы вводят более короткое обозначение (см. также п. 5): $d_i = d(r_i)$. — *Прим. перев.*

ячеек, связанных с интервалами, которые расположены по обе стороны I_{i_j} в списке INTLIST. Рис. 11 иллюстрирует типичную ситуацию на одном цикле работы процедуры. Следующая лемма показывает, что процедура *Build-Subdivision*(e, f) делает то, что от нее требуется.

Лемма 8.1. *Процедура Build-Subdivision*(e, f) *правильно вычисляет ячейки* C_j, *связанные с интервалами оптимальности относительно пары* (e, f).

Доказательство. Доказательство проводится индукцией по числу итераций в основном цикле. Допустим, что δ_k есть расстояние от точки события до s на k-й итерации. Прежде всего

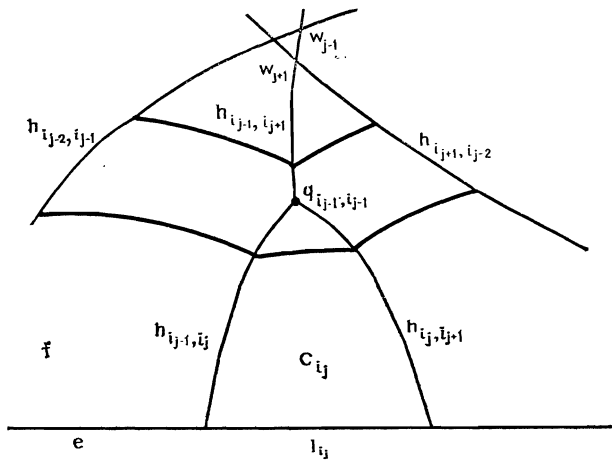


Рис. 11. Событие в точке $q_{i_{j-1},\ i_{j+1}}$.

заметим, что исходное разбиение, заданное первоначальными гиперболическими дугами при инициализации, справедливо для всех точек, расстояние от которых до s не превосходит δ_1 (поскольку по определению расстояния до точки события гиперболические дуги не могут пересечься на расстоянии, не превосходящем δ_1). Допустим теперь, что оно справедливо для всех точек, расстояние от которых до s не превосходит δ_k, и рассмотрим эффект возникновения события на расстоянии δ_k. Событию номер k отвечает смыкание границ ячейки C_I и возникновение контакта между ячейками C_L и C_R, которые становятся соседями на расстоянии, большем δ_k. (До наступления события ячейки C_L и C_R расположены соответственно слева и справа от C_I.) Если точка x находится на расстоянии δ от s,

где $\delta_k \leqslant \delta < \delta_{k+1}$, то утверждается, что она будет правильно расположена по отношению к активным дугам, т. е. внутри разбиения, построенного процедурой для точек между k-м и $(k+1)$-м событиями. Действительно, кратчайший путь в x не может проходить через интервал I, поскольку x лежит по другую сторону от общей границы ячеек C_I и C_L или C_I и C_R. Положение x относительно новой активной дуги (являющейся общей границей ячеек C_L и C_R) верно отражает ее расположение относительно соответствующих интервалов. Наконец, положение x относительно всех остальных интервалов также должно быть правильным, поскольку точки грани, находящиеся от s на расстоянии, лежащем в диапазоне от δ_k до δ_{k+1}, не влияют на расположение соответствующих разделяющих дуг, и, кроме того, согласно предположению индукции, то же относится к точкам грани, расстояние до которых от s не превосходит δ_k. Тем самым, высказанное утверждение вытекает из предположения индукции о том, что построенное до этого разбиение корректно. ■

Лемма 8.2. *Процедура Build-Subdivision*(e, f) *требует времени* $O(K \log K)$ *и памяти* $O(K)$, *где* K — *число интервалов оптимальности относительно пары* (e, f).

Доказательство. Поскольку на каждом цикле процедуры из списка INTLIST удаляется один интервал, шаги (1) и (2) выполняются не более K раз. Но на каждом из них число операций вставки и удаления элемента из списка ULIST одно и то же. Отсюда ясно, что общая временна́я сложность процедуры равна $O(K \log K)$. ■

Необходимо заметить, что разбиения $\mathcal{T}_1$, $\mathcal{T}_2$ и $\mathcal{T}_3$ необязательно строить по отдельности. (Это было сделано для простоты изложения.) Можно распространять сразу все интервалы относительно пар (e_1, f), (e_2, f) и (e_3, f) во внутреннюю часть грани f. Соответствующая процедура аналогична процедуре для одного ребра, приведенной выше, но теперь список INTLIST должен быть циклическим списком интервалов, расположенных вдоль периметра f (тем самым необходимость в «фиктивных» гиперболах на ребрах грани f отпадает).

Процедура *Build-Subdivision*(e, f) выполняется для каждой пары ребро — грань (e, f). Это приводит к построению трех разбиений каждой грани и требует $O(n^2 \log n)$ суммарного времени (поскольку общее число интервалов оптимальности относительно всех пар ребро — грань оценивается как $O(n^2)$). Чтобы определить длину кратчайшего пути из s в любую точку $x \in \mathcal{P}$, мы просто должны найти грань f, которой принадлежит x, а затем

три ячейки разбиений f, в которых лежит эта точка. Тем самым мы получаем длины кратчайших путей в x, проходящих через каждое из трех ребер f. Выбирая наименьшую из них, получаем длину кратчайшего пути в x. Все четыре задачи о положении точки могут быть решены стандартными методами за время $O(\log n)$ ([15], [3]). (Единственное небольшое отличие нашей ситуации состоит в том, что границы клеток являются гиперболами, а не прямыми. Это, однако, не вызывает серьезных затруднений.) Сам оптимальный путь в x находится обратной трассировкой указателей на предшественников (от I к $\bar{I}$, потом к $\bar{\bar{I}}$ и т. д.). Понятно, что обратная трассировка потребует $O(K)$ времени, где K — число граней, через которые проходит оптимальный путь. В итоге мы имеем следующую теорему.

Теорема 8.3. *После предобработки длина кратчайшего пути из s в любую точку $x \in \mathscr{P}$, лежащую на многогранной поверхности $\mathscr{P}$, может быть определена за время $O(\log n)$, а сам кратчайший путь может быть найден за время $O(K + \log n)$, где K — число граней, которые он пересекает. Предобработка может быть осуществлена за время $O(n^2 \log n)$ (и требует памяти объема $O(n^2)$).*

9. ЗАКЛЮЧЕНИЕ

В работе приведен алгоритм сложности $O(n^2 \log n)$, строящий такое разбиение поверхности произвольного многогранника, что длина кратчайшего пути из заданной начальной точки s (источника) до произвольной точки t поверхности может быть определена с помощью локализации положения точки t в разбиении. Для работы алгоритма требуется объем памяти $O(n^2)$. Определение положения точки может быть осуществлено стандартными методами за время $O(\log n)$, после чего сам путь находится обратной трассировкой за время $O(K)$, где K — число граней, через которые он проходит.

Наш алгоритм основывается на методе, который мы назвали «непрерывный Дейкстра». Этот метод очень напоминает оригинальный алгоритм Дейкстры нахождения кратчайшего пути, но при этом включает новые понятия о распространяющемся *волновом фронте* и дискретных *событиях*. В некотором смысле получается сочетание алгоритма Дейкстры и принципа заметания, используемого в вычислительной геометрии.

Алгоритм легко обобщается на случай нескольких источников, а потому может быть применен для построения диаграммы Вороного на поверхности многогранника [11]. Для этого достаточно при инициализации очереди событий занести в нее

события, связанные с ребрами, противоположными каждому источнику, а затем при распространении интервалов действовать точно так же, как и ранее. Для m точек-источников, заданных на поверхности многогранника, диаграмма Вороного строится алгоритмом за время $O(N^2 \log N)$ и требует обьема памяти $O(N^2)$, где $N = \max(n, m)$. После этого определение ближайшего к произвольной точке источника требует $O(\log N)$ времени, а нахождение пути до него осуществляется с помощью обратной трассировки за время, пропорциональное числу изломов искомого пути.

Нетрудно также представить себе, как предложенный алгоритм может быть обобщен на случай поверхностей других типов (не являющихся многогранными). Если поверхность задана порциями, образующими ее грани, то при определенном типе сложности поверхности (как говорят, «достаточно хорошем поведении») мы можем рассчитать, как проходят по ней геодезические. Так, в случае плоских порций поверхности мы имеем задачу, рассмотренную в настоящей работе, и участки геодезических на каждой грани являются в этом случае прямыми линиями. Для сферических порций поверхности геодезическими будут дуги больших кругов. Поэтому нам было бы желательно обобщить локальный критерий оптимальности, состоящий в том, что геодезические, пересекающие границы порций поверхности, при развертке переходят в прямые: развертки геодезических только *локально* (вблизи границ порций поверхности) должны быть прямыми. А это есть просто условие на непрерывность изменения касательной вдоль геодезической. Метод «непрерывный Дейкстра» позволяет распространять фронт волны поперек границ порций поверхности, не нарушая критерия локальной оптимальности. Детализация рассмотренного обобщения требует некоторой дополнительной работы.

Нам бы хотелось улучшить временну́ю сложность предложенного алгоритма. Как нам кажется, ее можно свести к оценке $O(n^2)$ (такая оценка является в настоящее время наилучшей для алгоритмов обхода препятствий на плоскости), но до тех пор, пока для плоского случая не будут найдены более эффективные решения, мы не надеемся добиться еще большего продвижения [1].

[1]) Недавно опубликован алгоритм построения кратчайших путей на плоскости в среде с прямоугольными препятствиями, имеющий сложность $O(n \log n)$. Используемый в нем метод близок к предложенному авторами настоящей работы. См. P. J. de Rezende, D. T. Lee, G. F. Wu. Rectilinear Shortest Paths in the Presence of Rectangular Barriers. *Discrete & Comput. Geometry*, v. 4 (1989), N 1, 41—53. — *Прим. перев.*

Что касается оценки объема памяти $O(n^2)$, она бесспорно поддается улучшению. Предложенный алгоритм является громоздким в том плане, что в процессе его работы запоминаются точки пересечения карты кратчайших путей с каждым ребром, а число таких точек квадратично зависит от n. В то же время размер графа, расположение которого на поверхности образует нужное нам разбиение, лишь линейно зависит от числа ребер исходного многогранника. Представляется целесообразным хранить информацию не о концевых точках интервалов, а о кривых раздела и распространять эти кривые через ребра, не создавая при этом каждый раз новые структуры. (Это также наводит на мысль о построении всего разбиения (включая разбиение внутренней части граней) «по ходу дела» и хранении информации о пересечении кривых раздела подобно тому, как мы это делали в процедуре, описанной в § 8.) При этом нужно следить за тем, чтобы структуры данных, используемые для хранения построенного разбиения, позволяли эффективно отвечать на запросы о положении точки (т. е. за время $O(\log n)$). В работе [12] Маунт продемонстрировал способ построения подобного разбиения многогранной поверхности с использованием структуры данных размера $O(n \log n)$, позволяющей отвечать на запрос о положении точки за время $O(\log n)$. Может ли такая конструкция быть построена по ходу исполнения предложенного алгоритма? Укажем, что для случая выпуклых многогранников Маунту удалось снизить требования к объему памяти до $O(n \log n)$ [10].

Другая очевидная открытая проблема заключается в нахождении эффективного алгоритма определения кратчайших путей в трехмерном пространстве при наличии произвольной совокупности многогранных препятствий. Геодезическая задача является особым случаем, для которого эффективный алгоритм существует. Однако мы убеждены, что имеются и другие специальные постановки, допускающие простые и эффективные решения.

Благодарность. Нам хотелось бы поблагодарить референтов за тщательное прочтение текста и замечания, улучшившие форму подачи материала.

ЛИТЕРАТУРА

[1] Dijkstra E. W. A note on two problems in connection with graphs. Numer. Math., 1 (1959), p. 269—271. [См. кн. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. — М.: Мир, 1979, с. 235—238.]

[2] Garey M. R., Johnson D. S., Preparata F. P., Tarjan R. E. Triangulating a simple polygon. Inform. Process. Lett., 7 (1978), p. 175—179.

[3] Kirkpatrick D. G. Optimal search in planar subdivisions. SIAM J. Comp., 12 (1983), p. 28—35.
[4] Lee D. T. Proximity and reachability in the plane. Ph. D. thesis, Technical Report ACT-12, Coordinated Science Laboratory, Univ. of Illinois, Chicago, IL, November 1978.
[5] Lee D. T., Preparata F. P. Euclidean shortest paths in the presence of rectilinear boundaries. Networks, 14 (1984), p. 393—410.
[6] Lozano-Perez T., Wesley M. A. An algorithm for planning collision-free paths among polyhedral obstacles. Comm. ACM, 22 (1979), p. 560—570.
[7] Lyusternik L. A .Shortest Paths: Variational Problems. Macmillan, New York, 1964. [Ориг.: Люстерник Л. А. Кратчайшие линии: вариационные задачи. — М.: Гостехиздат, 1955.]
[8] Mitchell J. S. B. Shortest paths in the plane in the presence of obstacles. Manuscript, Dept. of Operations Research, Stanford Univ., Stanford, CA, 1984.
[9] Mitchell J. S. B. Planning shortest paths. Ph. D. thesis, Dept. of Operations Research, Stanford Univ., Stanford, CA, August, 1986. (Available as Research Report 561, Artificial Intelligence Series, No. 1, Hughes Research Laboratories, Malibu, CA.)
[10] Mount D. M. On finding shortest paths on convex polyhedra. Technical Report 1495, Dept. of Computer Science, Univ. of Maryland, Baltimore, MD, 1985.
[11] Mount D. M. Voronoi diagrams on the surface of a polyhedron. Technical Report 1496, Dept. of Computer Science, Univ. of Maryland, Baltimore, MD, 1985.
[12] Mount D. M. Storing the subdivision of a polyhedral surface. Proc. 2nd ACM Symposium on Computational Geometry. Yorktown Heights, NY, June 2—4, 1986.
[13] O'Rourke J., Suri S., Booth H. Shortest paths on polyhedral surfaces. Manuscript. The John Hopkins Univ., Baltimore, MD, 1984.
[14] Papadimitriou C. H. An algorithm for shortest-path motion in three dimensions. Inform. Process. Lett., 20 (1985), p. 259—263.
[15] Preparata F. P. A new approach to planar point location. SIAM J. Comp., 10 (1981), p. 473—482.
[16] Reif J. H., Storer J. A. Shortest paths in Euclidean space with polyhedral obstacles. Technical Report CS—85—121, Computer Science Dept., Brandeis Univ., Waltham, MA, April, 1985.
[17] Sharir M., Schorr A. On shortest paths in polyhedral spaces. SIAM J. Comp., 15 (1986), p. 193—215.

Алгоритм триангуляции простого многоугольника с временной сложностью $O(n \log \log n)$[1)]

Р. Э. Тарьян[2)], *К. Дж. Ван Вик*[3)]

Задача триангуляции простого многоугольника с n вершинами состоит в выборе $n-3$ непересекающихся диагоналей, разбивающих его внутренность на $n-2$ треугольника. Предлагается алгоритм решения этой задачи, имеющий временную сложность $O(n \log \log n)$, что лучше прежней оценки $O(n \log n)$. Тем самым показывается, что задача триангуляции не такая сложная, как задача сортировки. Полученный результат позволяет улучшить алгоритмы для решения некоторых других задач вычислительной геометрии, включая проверку многоугольника на простоту.

1. ВВЕДЕНИЕ

Пусть P — простой многоугольник с n вершинами, заданный списком своих вершин $v_0, v_1, \ldots, v_{n-1}$, перечисляемых в порядке обхода границы многоугольника по часовой стрелке. (При движении по границе многоугольника в направлении по часовой стрелке внутренность многоугольника будет находиться справа.) Обозначим ∂P границу многоугольника P. В этой статье предполагается, что все вершины многоугольника P имеют различные y-координаты (это предположение не умаляет общности). Кроме того, для удобства положим $v_n = v_0$. *Стороны* многоугольника P — это открытые отрезки с концами в вершинах v_i и v_{i+1}, $0 \leqslant i < n$. *Диагонали* многоугольника P суть открытые отрезки, целиком лежащие внутри многоугольника P, концами которых являются вершины многоугольника. *Задача триангуляции* заключается в выборе $n-3$ непересекаю-

---

[1)] An $O(n \log \log n)$-time algorithm for triangulating a simple polygon. SIAM Journal on computing, v. 17, n. 1, 1988, pp. 143—178.

[2)] AT&T Bell Laboratories, Murray Hill, New Jersey 07974. Department of Computer Science, Princeton, New Jersey 08544.

[3)] AT&T Bell Laboratories, Murray Hill, New Jersey 07974.

щихся диагоналей многоугольника P, разбивающих его внутренность на $n-2$ треугольников.

Если многоугольник P выпуклый, то любая пара его вершин определяет диагональ и триангуляция многоугольника P довольно просто может быть выполнена за время $O(n)$. Если P не является выпуклым, то не все пары вершин определяют диагонали, и сама по себе задача определения одной диагонали, не говоря уже о задаче триангуляции, нетривиальна. В 1978 г. Гэри, Джонсон, Препарата и Тарьян [10] предложили алгоритм триангуляции с временно́й сложностью $O(n \log n)$. Дальнейшее исследование этой задачи велось по двум направлениям. Некоторые исследователи разработали алгоритмы триангуляции с линейной временной сложностью для специальных классов многоугольников, таких как монотонные [10] и звездные [31] многоугольники [32, 33]. Другие разработали алгоритмы триангуляции с временной сложностью $O(n \log k)$, где k — параметр, некоторым образом характеризующий сложность многоугольника, такой, например, как число невыпуклых углов [13] или «извилистость» [5]. Так как для каждой такой меры сложности существует класс многоугольников, для которых $k = \Omega(n)$, то в худшем случае время выполнения этих алгоритмов составляет $O(n \log n)$. Вопрос, можно ли произвести триангуляцию многоугольника за время $o(n \log n)$, т. е. асимптотически быстрее, чем сортировку, был одним из главных открытых вопросов вычислительной геометрии.

В этой статье предлагается алгоритм триангуляции с временной сложностью $O(n \log \log n)$ и тем самым показывается, что задача триангуляции и в самом деле более легкая, чем задача сортировки. Данная статья представляет исправленный вариант доклада, опубликованного в трудах конференции [27], где ошибочно утверждалось о существовании алгоритма с временной сложностью $O(n)$. Все попытки разработать алгоритм с линейной сложностью по-прежнему остаются безуспешными, но предлагаемый нами подход определяет несколько направлений для дальнейшего поиска и проясняет те трудности, которые при этом потребуется преодолеть.

Отправной точкой нашего алгоритма является сведение задачи триангуляции к задаче вычисления информации о видимости по какому-то одному направлению, в качестве которого мы возьмем горизонтальное направление. Будем говорить, что вершина и сторона многоугольника P образуют *ВС-пару видимости*, или просто *ВС-пару*, если их можно соединить открытым горизонтальным отрезком, целиком лежащим внутри многоугольника P. Аналогично, две стороны образуют *СС-пару видимости*, или просто *СС-пару*, если их можно соединить открытым

горизонтальным отрезком, целиком лежащим внутри P. Фурнье и Монтуно [9] доказали, что задача триангуляции простого многоугольника сводима за линейное время к задаче вычисления всех BC-пар этого многоугольника (см. дополнение переводчика). Независимо от них этот результат получен в работе [5]. Учитывая сказанное, в этой статье мы представим лишь

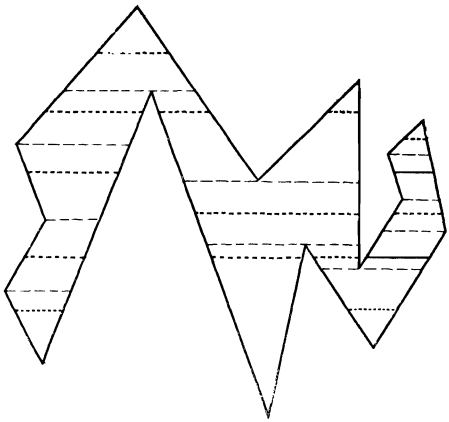


Рис. 1. Иллюстрация к определению пар видимости в простом многоугольнике. Штриховые горизонтальные прямые соответствуют BC-парам; точечные горизонтальные прямые соответствуют CC-парам.

алгоритм вычисления всех BC-пар многоугольника, имеющий сложность $O(n \log\log n)$. С помощью упомянутого сведения получаем алгоритм триангуляции простого многоугольника с временной сложностью $O(n \log\log n)$.

Наш алгоритм вычисления пар видимости определяет не только пары видимости, образованные вершиной и стороной многоугольника, но и некоторые пары видимости, образованные сторонами многоугольника (CC-пары). Утверждается, что полное число пар видимости каждого типа (BC- или CC-пар) линейным образом зависит от числа вершин многоугольника.

Лемма 1. *В простом многоугольнике с n вершинами имеется не более $2n$ BC-пар и не более $2n$ CC-пар.*

Доказательство. Каждая вершина многоугольника может принадлежать не более чем двум BC-парам. Следовательно, в целом по всем вершинам получаем не более $2n$ BC-пар. Разобьем многоугольник P на трапеции и треугольники, проведя горизонтальные отрезки, целиком лежащие внутри P и соединяющие элементы каждой BC- и CC-пар (рис. 1). Каждая CC-пара соответствует нижней стороне в точности одной такой

трапеции или треугольника. В случае трапеции ее верхняя сторона состоит из одного или двух отрезков, соответствующих BC-парам. В случае треугольника его верхняя вершина не входит ни в одну BC-пару. Любая вершина многоугольника порождает не более двух отрезков, служащих верхней стороной трапеций, или она является верхней вершиной треугольника. Таким образом, имеются не более $2n$ трапеций и треугольников, нижние стороны которых соответствуют CC-парам, и, следовательно, не более $2n$ таких пар. ■

Другим краеугольным камнем нашего метода служит очень тесная взаимосвязь между вычислением видимости и задачей о жордановой сортировке. *Задача жордановой сортировки* для простого многоугольника P и горизонтальной прямой L состоит в упорядочении точек пересечения ∂P и L в соответствии со значением x-координаты в случае, когда в качестве исходных данных служит лишь список этих точек пересечения, перечисленных в порядке их прохождения при движении вдоль ∂P в направлении по часовой стрелке. (Заметим, что список вершин многоугольника P *не является* частью исходных данных.) Хофмен, Мелхорн, Розенстил и Тарьян [14] представили в своей работе алгоритм для задачи жордановой сортировки, имеющий линейную сложность и решающий задачу для произвольной простой кривой, как открытой, так и замкнутой. В этом алгоритме, который мы называем алгоритмом жордановой сортировки, требуется, чтобы всякий раз, когда ∂P доходит до L, она пересекала ее. Но этот алгоритм можно легко изменить с тем, чтобы можно было обработать и точки касания, при условии, что про каждую точку в исходных данных известно, является ли она точкой пересечения или точкой касания. Вычисление пар видимости является по крайней мере такой же сложной задачей, как и задача жордановой сортировки. Более точно это сформулировано в следующей лемме.

Лемма 2. *Используя алгоритм вычисления BC-пар, можно решить задачу жордановой сортировки для n-угольника P, затратив на это дополнительное время $O(n)$, при условии, что в качестве исходных данных задан многоугольник и прямая L (точки пересечения прямой L с многоугольником P не заданы).*

Доказательство. Вычислим все BC-пары многоугольника P. После этого «вывернем» многоугольник P, «разорвав» его границу ∂P в вершине с наименьшей y-координатой и «склеив» ее с границей произвольного прямоугольника, охватывающего многоугольник P, как это показано на рис. 2. В результате получится многоугольник Q, имеющий $n+5$ вершин. Вычислим

BC-пары многоугольника *Q*. Эти пары определяют видимость вершин многоугольника *P* снаружи и указывают, какие его вершины будут видны из внешних точек, расположенных достаточно далеко слева или справа от многоугольника *P*. Разобьем внутренность и внешность многоугольника *P* на трапеции,

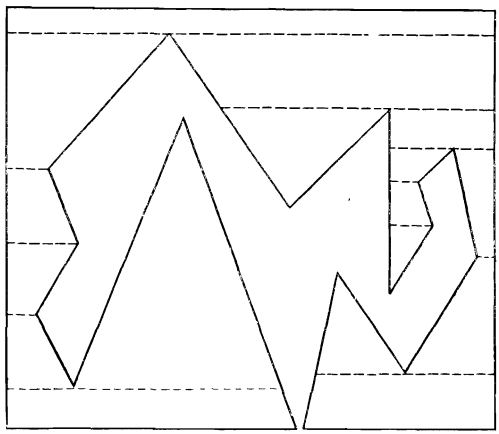


Рис. 2. «Вывернутый наизнанку» многоугольник с рис. 1 и соответствующие ему *BC*-пары видимости.

треугольники и неограниченные трапецеидальные области, проведя горизонтальные отрезки, соответствующие всем парам видимости. Если задана некоторая прямая *L*, то точки пересечения ∂P и *L* в порядке увеличения их *x*-координаты можно получить, просматривая слева направо области, через которые проходит прямая *L*. Полное время выполнения этого алгоритма (без учета времени на двухкратное вычисление пар видимости) составляет $O(n)$. ■

Так как при любом вычислении данных о видимости неявно выполняется жорданова сортировка, то вполне естественно попытаться явным образом использовать жорданову сортировку для вычисления пар видимости. Это соображение приводит к следующему алгоритму, основанному на методе «разделяй и властвуй» (рис. 3).

Шаг 1. В заданном многоугольнике *P* выбираем одну из его вершин *v*, *y*-координата которой не является ни максимальной, ни минимальной среди всех вершин многоугольника *P*. Если такой вершины не существует, то в *P* нет ни одной пары видимости, и на этом применение алгоритма к многоугольнику *P*

заканчивается. В противном случае проведем через вершину v горизонтальную прямую, обозначив ее L.

Шаг 2. Определим точки пересечения ∂P с L в порядке их расположения на границе многоугольника P.

Шаг 3. Выполним жорданову сортировку найденных точек пересечения и перечислим все пары видимости, определяемые последовательными точками пересечения в порядке их расположения на L.

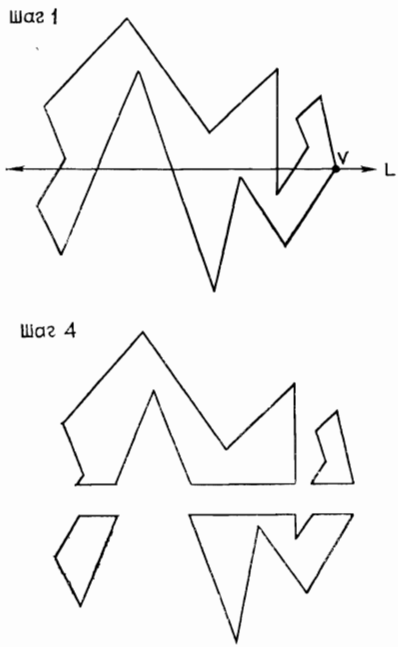


Рис. 3. Иллюстрация к выполнению шагов 1 и 4 алгоритма вычисления пар видимости.

Шаг 4. Разобьем многоугольник P по прямой L на части, каждая из которых является простым многоугольником.

Шаг 5. К каждому из многоугольников, получившемуся в результате разбиения P, применим рекурсивно данный алгоритм.

К счастью, алгоритм жордановой сортировки в качестве побочного эффекта вычисляет достаточно много дополнительной информации, облегчающей выполнение шага 4 приведенного выше алгоритма. Шаг 2 является наиболее трудной частью алгоритма. В алгоритме имеются два узких места, непродуманная реализация каждого из которых приведет к тому, что алгоритм будет иметь квадратичную временную сложность. Во-первых, в ряде случаев алгоритм будет многократно порождать некоторые пары видимости. Действительно, для многоугольника, показанного на рис. 4, алгоритм может найти $\Omega(n^2)$ повторяющихся пар видимости.

Эту неприятность можно обойти, изменив шаг 2 алгоритма таким образом, чтобы вычислять не все точки пересечения ∂P с L, а лишь некоторые из них. Однако это может привести к появлению ошибок при выполнении жордановой сортировки на шаге 3. Эти ошибки возникают из-за того, что в такой ситуации сортируемая последовательность может не содержать

всех точек пересечения простого многоугольника с прямой. К счастью, алгоритм сортировки является итеративным, и при обнаружении ошибки можно продолжить его выполнение с некоторого допустимого состояния, вычислив для этого дополнительно несколько точек пересечения и сделав локальные изменения в используемой им структуре данных. Будем называть этот

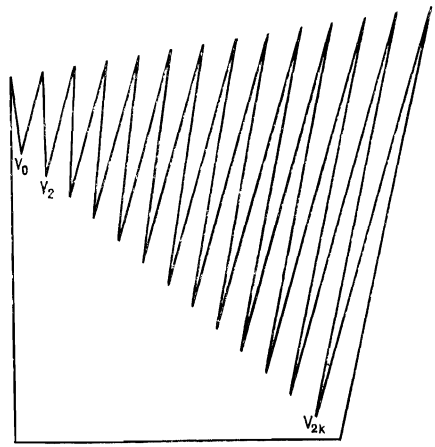


Рис. 4. Обрабатывая многоугольники, подобные приведенному на рисунке, «прямолинейный» алгоритм порождает пары видимости в количестве, пропорциональном квадрату числа вершин многоугольника. Первый разрез, проходящий через вершину v_0, отсекает $k+1$ треугольников. Последующие разрезы, проходящие через вершины v_{2i}, $i = 1, 2, \ldots, k$, порождают каждый раз $k-i+1$ пар видимости, но лишь две из них являются новыми.

модифицированный метод сортировки *жордановой сортировкой с исправлением ошибок.*

Вычисляя лишь часть точек пересечения ∂P с L и используя жорданову сортировку с исправлением ошибок, получаем алгоритм определения пар видимости, вычисляющий лишь $O(n)$ пар видимости и требующий для этого времени $O(n)$ (без учета времени, необходимого для определения точек пересечения). Такой подход требует использования двухуровневой структуры данных для представления границы многоугольника, но при этом остается полная свобода во всем остальном, касающемся этой структуры данных.

Второе и значительно более существенное узкое место в нашем алгоритме — это проблема определения точек пересечения. Прямая L разбивает ∂P на части. Если бы каждая из этих частей образовывала границу отдельного подмногоугольника, то можно было (без особого труда) получить оценку $O(n)$ для

временной сложности алгоритма вычисления пар видимости, воспользовавшись для представления границ многоугольников структурой данных, называемой деревом поиска с указателями, и учитывая линейность следующего рекуррентного соотношения [20]:

$$(1)\quad T(n)=\begin{cases} O(1), & \text{если } n=1; \\ \max\limits_{1\leqslant k<n}\{T(k)+T(n-k)+O(1+\log\min\{k,\ n-k\})\}, \\ & \text{если } n>1. \end{cases}$$

К сожалению, части исходной границы многоугольника нельзя рассматривать каждую по отдельности, так как они группируются, образуя границы подмногоугольников. Чтобы улучшить оценку $O(n\log n)$ временной сложности задачи триангуляции простого многоугольника, даваемую существующими алгоритмами, необходима новая идея, какой является идея *сбалансированного разбиения на части*. Мы усовершенствуем алгоритм вычисления пар видимости, сделав более разумным выбор прямой L, так чтобы каждая из границ образующихся подмногоугольников содержала относительно небольшое число частей исходной границы многоугольника. Метод сбалансированного разбиения на части в сочетании с использованием деревьев поиска с указателями в качестве структуры данных для представления границ многоугольников позволяет получить алгоритм вычисления пар видимости с временной сложностью $O(n\log\log n)$.

Оставшаяся часть данной статьи включает пять разделов и приложение. В разд. 2 рассматривается алгоритм жордановой сортировки и его модификация, позволяющая исправлять ошибки. В разд. 3 обсуждается базовый алгоритм вычисления пар видимости, использующий жорданову сортировку с исправлением ошибок и выдающий $O(n)$ пар видимости. В разд. 4 этот алгоритм получает дальнейшее развитие на основе метода сбалансированного разбиения на части. В разд. 5 описывается структура данных для представления границы многоугольника. Эта структура данных состоит из двух уровней деревьев поиска с указателями. Показывается, что с использованием такой структуры данных временная сложность алгоритма вычисления пар видимости, рассматриваемого в разд. 4, равна $O(n\log\log n)$. В заключение в разд. 6 приводятся некоторые замечания, обсуждаются вопросы, касающиеся приложений, и формулируются открытые проблемы. В приложении приведено описание деревьев поиска с указателями, которые используются не только в алгоритме, непосредственно вычисляющем пары видимости, но и в алгоритме жордановой сортировки.

2. ЖОРДАНОВА СОРТИРОВКА С ИСПРАВЛЕНИЕМ ОШИБОК

Пусть P — простой многоугольник, а v — одна из его вершин. Обозначим L горизонтальную прямую, проходящую через вершину v, а точки пересечения этой прямой с границей многоугольника P обозначим $x_0 = v, x_1, \ldots, x_{m-1}$. Точки пересечения перечислены в порядке обхода границы многоугольника в направлении по часовой стрелке. Так как в данной работе предполагается, что вершины многоугольника P имеют различные y-координаты, то множество точек пересечения L и ∂P конечно. Точки $x_1, x_2, \ldots, x_{m-1}$ являются точками пересечения ∂P и L; точка x_0 является либо точкой пересечения, либо точкой касания. Введем на множестве точек пересечения x_i полный порядок, соответствующий упорядоченности этих точек по значению их x-координаты. Мы хотим упорядочить точки $x_1, x_2, \ldots, x_{m-1}$ в соответствии с этим полным порядком.

Последовательность $x_0, x_1, \ldots, x_{m-1}$ порождает два леса. Ниже обсуждается, как это происходит. Положим для удобства $x_m = x_0$. Не ограничивая общности, будем считать, что часть границы ∂P от x_0 до x_1 находится выше прямой L. Обозначим $l_i = \min\{x_{i-1}, x_i\}$ и $r_i = \max\{x_{i-1}, x_i\}$, где $0 < i \leqslant m$. Будем говорить, что пара $\{x_{i-1}, x_i\}$ *охватывает* точку x, если $l_i \leqslant x \leqslant r_i$. Две пары $\{x_{i-1}, x_i\}$ и $\{x_{j-1}, x_j\}$ называются *пересекающимися*, если $\{x_{i-1}, x_i\}$ охватывает в точности одну из точек x_{j-1} и x_j; пара $\{x_{i-1}, x_i\}$ *охватывает* пару $\{x_{j-1}, x_j\}$, если она охватывает одновременно обе точки x_{j-1} и x_j. Из условия простоты многоугольника P следует, что если $i \equiv j \bmod 2$, то две пары $\{x_{i-1}, x_i\}$ и $\{x_{j-1}, x_j\}$ не пересекаются. Будем называть это свойство *свойством непересекаемости*. Диаграмма Хассе отношения «охватывает» на множестве пар $\{\{x_{2i}, x_{2i+1}\} \mid 0 \leqslant i < m/2\}$ является лесом, называемым далее *верхним лесом*. Диаграмма Хассе отношение «охватывает» на множестве пар $\{\{x_{2i-1}, x_{2i}\} \mid 0 < i \leqslant m/2\}$ также является лесом, называемым *нижним лесом*. Упорядочим каждое множество потомков в каждом из двух лесов, поместив $\{x_{i-1}, x_i\}$ перед $\{x_{j-1}, x_j\}$, если $r_i \leqslant l_j$. Эта операция делает каждый лес упорядоченным лесом. Преобразуем оба леса в деревья, добавив к каждому из них фиктивную пару $\{-\infty, \infty\}$. В результате получим два дерева, называемые *верхнее дерево* и *нижнее дерево* (рис. 5). Множество, состоящее из родительской вершины в каждом из деревьев и ее потомков, называется *семьей*.

Здесь мы представим алгоритм жордановой сортировки [14] в форме, удобной для его последующего расширения для применения при вычислении пар видимости. Алгоритм обрабатывает точки $x_1, x_2, \ldots, x_m$ итеративно, добавляя на каждом

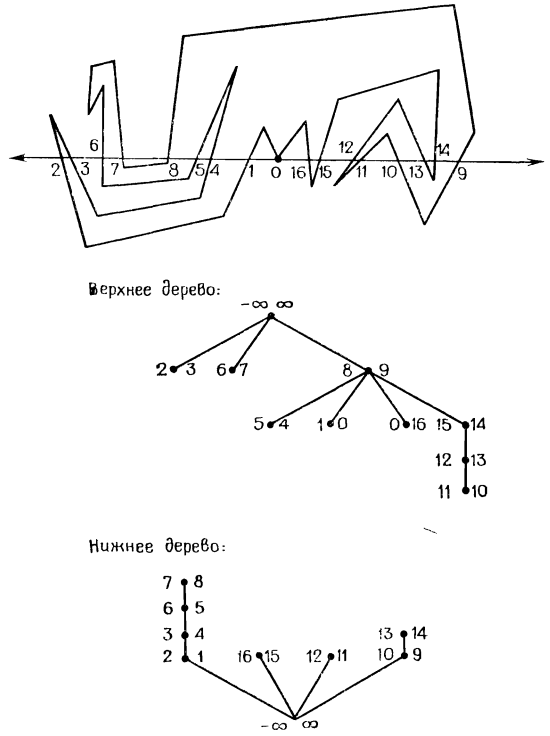


Рис. 5. Диаграмма Хассе отношения «охватывает», определяемого прямой L. (На рисунке каждая точка x_i отмечена своим номером i.)

шаге по одной точке и постепенно создавая верхнее дерево, нижнее дерево и упорядоченный список этих точек пересечения. В начале работы оба дерева содержат единственную пару $\{-\infty, \infty\}$, а упорядоченный список имеет вид $\{-\infty, x_0, \infty\}$. Основной этап обработки очередной точки x_i включает выполнение следующих шагов. Предположим, что i — нечетное число. Это значит, что пара $\{x_{i-1}, x_i\}$ должна добавляться к верхнему дереву. Будем считать, что $x_{i-1} < x_i$. (Случай $x_{i-1} > x_i$ является симметричным.)

Шаг 1. Найти точку x, следующую за x_{i-1} в упорядоченном списке.

Шаг 2. В верхнем дереве найти такую пару $\{x_{j-1}, x_j\}$, что $x \in \{x_{j-1}, x_j\}$.

Шаг 3. В зависимости от ситуации, выполнить одну из следующих четырех групп действий (рис. 6):

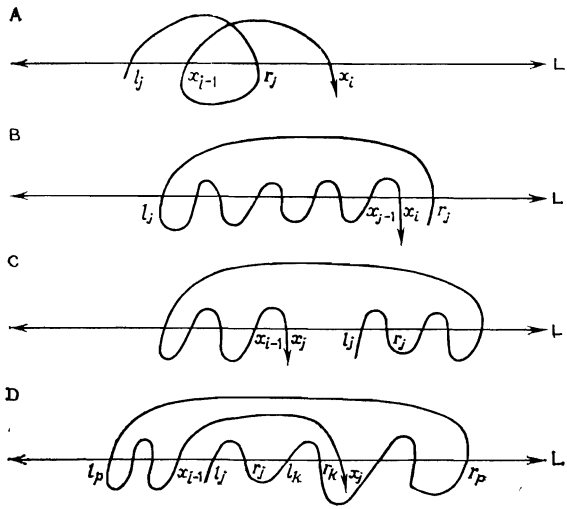


Рис. 6. Четыре случая, возникающие при жордановой сортировке.

Случай A ($l_j < x_{i-1} < r_j < x_i$). Завершить обработку, так как пары $\{x_{i-1}, x_i\}$ и $\{x_{j-1}, x_j\}$ пересекаются.

Случай B ($l_j < x_{i-1} < x_i \leqslant r_j$). Сделать пару $\{x_{i-1}, x_i\}$ новым последним сыном пары $\{x_{j-1}, x_j\}$. Если $i < m$, то вставить x_i в упорядоченный список после x_{i-1}.

Случай C ($x_i \leqslant l_j$). Вставить $\{x_{i-1}, x_i\}$ в список братьев пары $\{x_{j-1}, x_j\}$ непосредственно перед $\{x_{j-1}, x_j\}$. Если $i < m$, то вставить x_i в упорядоченный список после x_{i-1}.

Случай D ($x_{i-1} < l_j < x_i$). В списке братьев пары $\{x_{j-1}, x_j\}$ найти такую последнюю пару (назовем ее $\{x_{k-1}, x_k\}$), для которой $l_k < x_i$. Если $r_k > x_i$, то завершить обработку, так как пары $\{x_{i-1}, x_i\}$ и $\{x_{k-1}, x_k\}$ пересекаются. В противном случае, если пара $\{x_{k-1}, x_k\}$ является последним сыном родителей пары $\{x_{p-1}, x_p\}$ и $r_p < x_i$, то завершить обработку, так как пары $\{x_{i-1}, x_i\}$ и $\{x_{p-1}, x_p\}$ пересекаются. Если ни одно из этих пересечений не имеет места, то удалить из списка братьев пары $\{x_{j-1}, x_j\}$ подсписок, начиная с $\{x_{j-1}, x_j\}$ и кончая $\{x_{k-1}, x_k\}$ включительно, и вставить вместо него пару $\{x_{i-1}, x_i\}$. Удаленный подсписок пар сделать списком сыновей пары $\{x_{i-1}, x_i\}$. Если $i < m$, то вставить x_i в упорядоченный список после r_k.

Заметим, что если $\{x_{i-1}, x_i\}$ и $\{x_{j-1}, x_j\}$ — две такие пары, что $i > j$ и $i \equiv j \bmod 2$, то все четыре точки x_{i-1}, x_i, x_{j-1}, x_j различны, если только не имеет место случай $i = m$, и m — нечет-

ное число, и в этом случае возможно $x_i \in \{x_{j-1}, x_j\}$. Это значит, что случаи A — D исчерпывают все возможные варианты упорядочения четырех точек.

Случай, когда i — четное число, а значит, пара $\{x_{i-1}, x_i\}$ добавляется к нижнему дереву, обрабатывается аналогично тому, как описано выше, с той лишь разницей, что необходимо сделать небольшие изменения, чтобы учесть тот факт, что точка x_0 не принадлежит ни одной паре в нижнем дереве до тех пор, пока не будет обработана точка $x_m = x_0$ (если это произойдет). Эти изменения состоят в следующем:

(1) Непосредственно перед шагом 2 проверить, не выполняется ли $x = x_0$; если да, то заменить x точкой, следующей за x_0 в упорядоченном списке.

(2) На шаге 2 пара $\{x_{j-1}, x_j\}$, содержащая точку x, ищется в нижнем дереве (а не в верхнем дереве, как выше).

(3) На шаге 3 в случаях B и C, если $x_{i-1} < x_0 < x_i$, то точка x_i вставляется в упорядоченный список после x_0 (а не после x_{i-1}, как выше).

(4) На шаге 3 в случае D, если $r_k < x_0 < x_i$, то точка x_i вставляется в упорядоченный список после точки x_0 (а не после r_k, как выше).

Алгоритм сортировки в том виде, как он представлен выше, выполняет проверки с целью обнаружения пересечений пар. Если гарантируется корректность исходных данных (т. е. имеет место свойство непересекаемости), то алгоритм можно упростить. В этом случае можно исключить полностью случай A и две проверки на пересечение в случае D.

Чтобы рассмотренный алгоритм выполнял сортировку за линейное время, необходимо использовать в нем соответствующие структуры данных. А именно, каждый список братьев в верхнем и нижнем деревьях будем представлять с помощью однородного дерева поиска с указателями (см. приложение), каждый лист которого соответствует паре из списка братьев. Кроме того, между каждой парой и ее первым и последним сыновьями имеются двунаправленные указатели (в любом дереве, содержащем эту пару). Таким образом, каждая семья образует дважды связный кольцевой список, обладающий дополнительно тем свойством, что любая пара в списке может быть достигнута из любой другой пары, отстоящей от нее на d позиций в любом направлении, за время $O(1 + \log d)$. Более того, приведенное время [1]), необходимое для вставки пары рядом с заданной па-

[1]) *Приведенное время* — это время, приходящееся на одну операцию, усредненное по последовательности операций, выполняемой в худшем случае, при условии, что исходная структура данных является пустой. Подробное обсуждение этого понятия содержится в обзоре [26].

рой в список семьи, составляет $O(1)$, а приведенное время, необходимое для удаления подсписка из d пар из списка, содержащего s пар, в случае, когда заданы первая и последняя пары подсписка, составляет $O(1+\log(\min\{d, s-d\}+1))$.

Время выполнения алгоритма жордановой сортировки определяется в первую очередь временными затратами на выполнение операций типа «удалить подсписок» и «вставить пару», выполняемых над списками семей. Обозначим $T(p, s)$ максимальное значение приведенного времени, необходимое для выполнения совокупности из p таких операций над исходным списком, содержащим s элементов, и над удаленными подсписками. $T(p, s)$ удовлетворяет следующему рекуррентному соотношению:

$$(2)\quad T(p, s)=\begin{cases} 0, & \text{если } p=0; \\ \max\limits_{\substack{0\leqslant i<p \\ 0\leqslant d<s}} \{T(i, d+1)+T(p-i-1, s-d+1)+ \\ \quad +O(1+\log(\min\{d, s-d\}+1))\}, & \text{если } p>0. \end{cases}$$

По индукции можно доказать, что $T(p, s)=O(p+s)$. На обработку списка в алгоритме жордановой сортировки тратится время не более $T(\lceil m/2\rceil, 1)+T(\lfloor m/2\rfloor, 1)$, откуда следует, что время выполнения алгоритма равно $O(m)$. Детальное описание самого алгоритма и его анализ можно найти в исходной работе [14]. (В представленном в настоящей работе алгоритме несколько иная структура данных. Основное изменение состоит в устранении кольцевых ссылок по уровню в деревьях поиска с указателями. Эти модификации не влияют на полученную оценку временной сложности $O(m)$.)

Теперь мы хотим расширить алгоритм жордановой сортировки таким образом, чтобы при обнаружении двух пересекающихся пар он мог в определенных случаях возобновлять работу с некоторого скорректированного состояния, представляющего частично отсортированную последовательность, измененную так, чтобы устранить обнаруженное пересечение. При использовании алгоритма жордановой сортировки в задаче вычисления пар видимости алгоритм сортировки получает в качестве входных данных только некоторую подпоследовательность последовательности точек пересечения. При этом элементы подпоследовательности выбираются из последовательности не обязательно подряд. В этой подпоследовательности некоторые пары определяются как *особые*. (Все остальные пары являются *нормальными*.)

Чтобы приспособить алгоритм триангуляции к работе с такими данными, мы наложим на каждую особую пару $\{x_{i-1}, x_i\}$ дополнительное требование, заключающееся в том, чтобы из

данных точек пересечения она не охватывала никаких других, кроме x_{i-1} и x_i. Мы назовем это свойство *свойством пустоты*. С другой стороны, особые пары потенциально могут подлежать изменению: если $\{x_{i-1}, x_i\}$ — особая пара, то между точками x_{i-1} и x_i на границе ∂P могут иметься дополнительные точки пересечения. Алгоритм жордановой сортировки имеет возможность запрашивать такие дополнительные точки пересечения, если он обнаружит пересекающиеся пары или нарушения свойства пустоты.

Механизм, обеспечивающий получение дополнительных точек пересечения, реализуется процедурой корректировки refine. Входными данными для этой процедуры служат особая пара $\{x_{i-1}, x_i\}$ и охватываемая этой парой точка x. Процедура возвращает так называемую *ограничивающую пару* точек x' и x'', являющихся точками пересечения ∂P и L. При этом порядок следования точек вдоль L будет следующим: для четырех точек x_{i-1}, x', x'', x_i, для пяти точек x_{i-1}, x', x, x'', x_i (либо обратным). Если такой ограничивающей пары точек не существует, то refine ничего не возвращает. Если процедура возвращает пару точек, то новая последовательность точек для сортировки получается из старой последовательности путем вставки между точками x_{i-1} и x_i точек x' и x'', при этом точка x'' следует после точки x'. Из трех пар точек, заменяющих пару $\{x_{i-1}, x_i\}$, две пары, $\{x_{i-1}, x'\}$ и $\{x'', x_i\}$, являются особыми, а пара $\{x', x''\}$ — нормальной. (Это значит, что $\{x', x''\}$ — *ближайшая* к x пара точек пересечения.)

Мы модифицируем алгоритм жордановой сортировки так, чтобы он мог обрабатывать особые пары, используя процедуру refine всегда, когда это возможно, чтобы устранить нарушение свойств непересекаемости и пустоты. Соответствующее изменение алгоритма достигается путем введения следующих трех дополнений (рис. 7):

(1) На шаге 1, если $\{x_{i-1}, x_i\}$ — особая пара и $x < x_i$, выполнить refine$(\{x_{i-1}, x_i\}, x)$. Если процедура refine ничего не возвратит (в качестве своего результата), то завершить обработку, так как пара $\{x_{i-1}, x_i\}$ нарушает свойство пустоты. Если refine возвратит пару $\{x', x''\}$, то вставить x' и x'' в упорядочиваемую последовательность точек пересечения между точками x_{i-1} и x_i и возобновить обработку с точки x'.

(2) На шаге 3, случай B, если пара $\{x_{i-1}, x_j\}$ — особая, то закончить обработку, так как нарушено свойство пустоты. Даже если бы его можно было восстановить применением процедуры refine к $\{x_{j-1}, x_j\}$, это привело бы впоследствии к нарушению свойства непересекаемости. (То же самое имеет место на шаге 3 в случае A: даже если пара $\{x_{j-1}, x_j\}$ — особая и пересечение

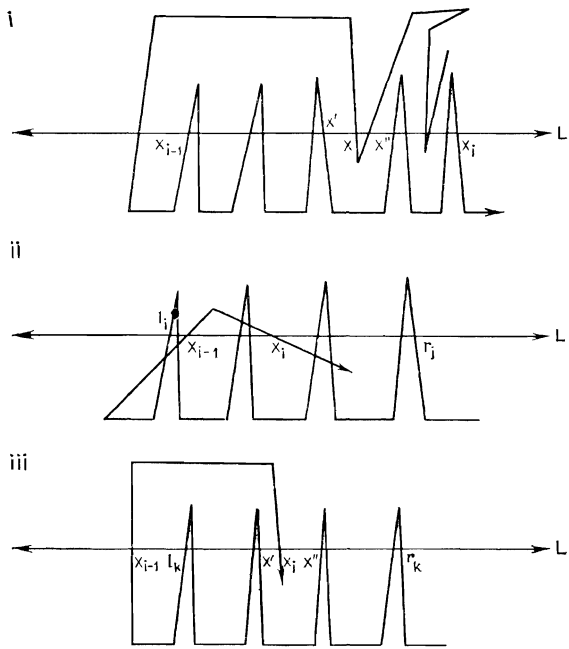


Рис. 7. Иллюстрация к изменениям, вносимым в алгоритм жордановой сортировки, позволяющим исправлять возникающие ошибки.

$\{x_{i-1}, x_i\}$ и $\{x_{j-1}, x_j\}$ может быть устранено путем корректировки $\{x_{j-1}, x_j\}$, это приведет к появлению новой пересекающейся пары.)

(3) На шаге 3, случай D, если пара $\{x_{k-1}, x_k\}$ — особая и $r_k > x_i$, то завершать выполнение алгоритма не нужно, а вместо этого нужно выполнить процедуру refine $(\{x_{k-1}, x_k\}, x_i)$. Если процедура refine ничего не возвратит, то завершить алгоритм: пара $\{x_{k-1}, x_k\}$ нарушает свойство пустоты. Если refine возвратит пару (x', x''), то в верхнем лесе заменить пару $\{x_{k-1}, x_k\}$ на $\{x_{k-1}, x'\}$, за которой следует $\{x'', x_k\}$. Вставить $\{x', x''\}$ в надлежащее место в нижнем лесе (в качестве брата или сына пары, содержащей x_k, в зависимости от ситуации). Дальнейшую обработку вести как для случая D на шаге 3, подставив пару $\{x_{k-1}, x'\}$ вместо $\{x_{k-1}, x_k\}$, а также вместо пары $\{x_{j-1}, x_j\}$, если $\{x_{j-1}, x_j\} = \{x_{k-1}, x'\}$. А именно, если $\{x_{j-1}, x_j\} = \{x_{k-1}, x_k\}$, то заменить пару $\{x_{j-1}, x'\}$ в списке ее братьев парой $\{x_{i-1}, x_i\}$ и сделать $\{x_{j-1}, x'\}$ сыном пары $\{x_{i-1}, x_i\}$. Если $\{x_{j-1}, x_j\} \neq \neq \{x_{k-1}, x_k\}$, то заменить подсписок от $\{x_{j-1}, x_j\}$ до $\{x_{k-1}, x'\}$ (включительно) парой $\{x_{i-1}, x_i\}$, а заменяемый подсписок

сделать списком сыновей пары $\{x_{i-1}, x_i\}$. В обоих случаях точку x_i необходимо вставить в сортируемый список после точки x' (или после x_0 или $x' < x_0 < x_i$).

Корректность алгоритма жордановой сортировки с исправлением ошибок следует из того факта, что в работе алгоритма особая пара $\{x_{j-1}, x_j\}$ может охватывать самое большее одну точку пересечения $x_i \notin \{x_{j-1}, x_j\}$. Чтобы убедиться в этом, предположим, не умаляя общности, что пара $\{x_{j-1}, x_j\}$ принадлежит верхнему дереву. Точка пересечения $x_i \notin \{x_{j-1}, x_j\}$ может быть вставлена в сортируемый список между x_{j-1} и x_j, так как пара $\{x_{i-1}, x_i\}$ добавляется к нижнему дереву, но в этом случае, как показано на рис. 7(ii), при обработке точки x_{i+1} будет обнаружено нарушение свойства пустоты.

Дополнения, которые необходимо сделать к алгоритму, чтобы он мог исправлять ошибки, приводят к дополнительным временным затратам, составляющим лишь $O(1)$ на обрабатываемую точку и на одну корректировку, без учета времени, затрачиваемого внутри вызовов процедуры refine. (Одна корректировка порождает не более двух операций вставки в списке братьев.) Таким образом, алгоритм сортировки с исправлением ошибок имеет временную сложность $O(m)$, где m — число точек пересечения в окончательной скорректированной последовательности. В следующем разделе будет показано, как алгоритм жордановой сортировки с исправлением ошибок может быть использован для вычисления пар видимости.

3. ЭФФЕКТИВНЫЙ АЛГОРИТМ ВЫЧИСЛЕНИЯ ПАР ВИДИМОСТИ

Наш алгоритм вычисления пар видимости следует общей схеме, представленной в разд. 1, основанной на методе «разделяй и властвуй»: исходный многоугольник разбивается на части, также представляющие собой многоугольники. Эти многоугольники в свою очередь подвергаются разбиению на меньшие многоугольники, и т. д. до тех пор, пока ни один из получившихся многоугольников не сможет быть подвергнут дальнейшему разбиению. Прежде чем подробно обсуждать этот метод, необходимо рассмотреть структуру многоугольников, получаемых при разбиении, которые мы можем называть *областями видимости*. Внутренность области видимости представляет связанное подмножество внутренности исходного многоугольника, заключенное между двумя горизонтальными прямыми $y = y_{\min}$ и $y = y_{\max}$, где $y_{\min} < y_{\max}$. Мы потребуем, чтобы граница области видимости реально пересекала обе эти прямые (рис. 8).

Граница области видимости состоит из связанных частей границы исходного многоугольника, называемых *граничными*

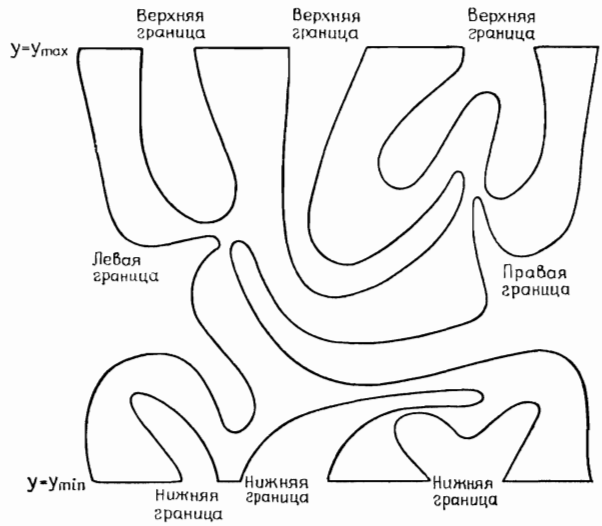


Рис. 8. Схематичное представление области видимости. Каждая кривая представляет сегмент, являющийся частью границы многоугольника.

сегментами, чередующихся с отрезками прямых $y = y_{\min}$ и $y = y_{\max}$. Каждый такой горизонтальный отрезок, не вырождающийся в точку, соответствует паре видимости. На каждой из прямых $y = y_{\min}$ и $y = y_{\max}$ лежит не более чем одна вершина исходного многоугольника. Хотя область видимости сама является простым многоугольником, тем не менее всякий раз, когда мы будем говорить о вершинах области видимости, мы будем иметь в виду лишь те из них, которые являются вершинами исходного многоугольника P. Граничный сегмент начинается вершиной или частью стороны исходного многоугольника, называемой *усеченной стороной*, и заканчивается вершиной или усеченной стороной. Сторона исходного многоугольника, содержащая усеченную сторону, называется *концевой стороной* сегмента.

Разобьем граничные сегменты на три группы в зависимости от их типа:

верхний сегмент: ни одна концевая сторона сегмента и ни одна его вершина не пересекает прямую $y = y_{\min}$;

нижний сегмент: ни одна концевая сторона сегмента и ни одна его вершина не пересекает прямую $y = y_{\max}$;

боковой сегмент: одна концевая сторона или вершина пересекает прямую $y = y_{\max}$, а другая пересекает прямую $y = y_{\min}$.

Вырожденный случай верхнего или нижнего граничного сегмента представляет сегмент, содержащий одну вершину и не содержащий усеченных сторон. Вырожденный случай бокового граничного сегмента представляет сегмент, содержащий одну усеченную сторону и не содержащий ни одной вершины. Граница области видимости состоит из четырех следующих друг за другом частей. Далее перечислены эти части в порядке их следования при обходе границы области по часовой стрелке. Множество верхних сегментов вместе со смежными с ними частями прямой $y = y_{\max}$ образует *верхнюю границу*. Затем следует боковой сегмент, образующий *правую границу* области. Множество нижних сегментов вместе со смежными с ними частями прямой $y = y_{\min}$ образует *нижнюю границу* области. И наконец, второй *боковой сегмент* образует *левую границу* области. В границе области обязательно присутствуют оба боковых сегмента; верхняя и/или нижняя границы могут быть пустыми.

Мы будем представлять область видимости, задав $y_{\min}$ и $y_{\max}$, а также четыре части границы (левую, правую, верхнюю и нижнюю). Верхнюю и нижнюю границы области будем представлять списками входящих в них граничных сегментов в порядке их следования при обходе границы по часовой стрелке. Левая и правая границы представляются одним сегментом каждая. В свою очередь каждый граничный сегмент будем представлять списком его вершин, перечисляя их в порядке следования при обходе границы по часовой стрелке, и концевыми сторонами (если они имеются). Мы не конкретизируем реализацию (организацию) списков для представления граничных сегментов, а также верхней и нижней границ области. Этот вопрос обсуждается в разд. 5.

Обсудив структуру областей видимости, рассмотрим вопрос о пересечении границы области с горизонтальной прямой и введем ряд связанных с этим понятий (рис. 9). Пусть V — область видимости, а L — горизонтальная прямая, пересекающая внутренность этой области. В зависимости от взаимного расположения прямой L и граничных сегментов области V разобьем все граничные сегменты на три категории:

мелкий: сегмент не пересекает прямую L;

глубокий: верхний сегмент, все вершины которого лежат строго ниже прямой L, или нижний сегмент, все вершины которого лежат строго выше прямой L;

смешанный: любой другой сегмент.

В соответствии с этой классификацией боковой сегмент является смешанным, а верхний или нижний сегмент может иметь любой тип. Определим М-границу как непрерывную часть списка граничных сегментов, перечисляемых в порядке обхода

∂V по часовой стрелке, содержащую лишь мелкие граничные сегменты. Аналогичным образом определяется Г-граница. Г-граница и М-граница состоят либо целиком из верхних сегментов, либо целиком из нижних сегментов. Каждая усеченная сторона Г-границы пересекает прямую L, и других точек пересечения

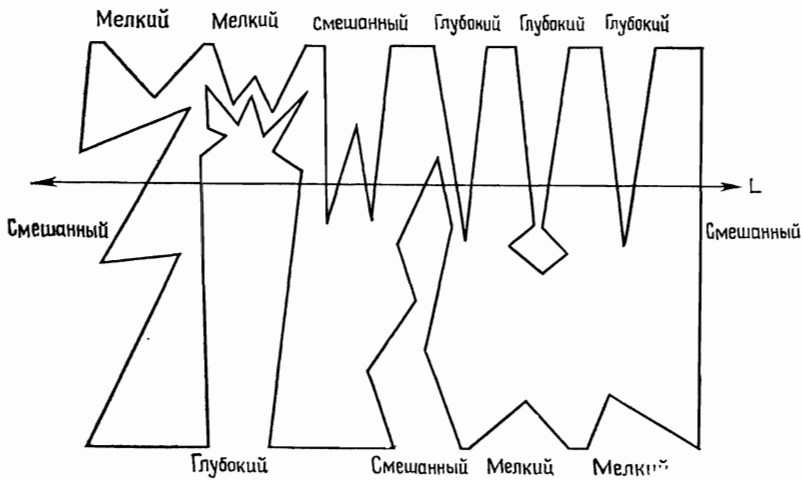


Рис. 9. Иллюстрация трех типов граничных сегментов. Верхняя группа сегментов включает Г-границу, состоящую из трех сегментов. Верхняя и нижняя группы сегментов включают М-границы, каждая из которых состоит из двух сегментов.

этой Г-границы с L нет. Введем следующую классификацию точек пересечения ∂V с L:

пересечение с максимальной Г-границей, не являющееся ни первым, ни последним пересечением с этой Г-границей, будем называть *несущественным*;

любое другое пересечение будем называть *существенным*

Последнее, что необходимо обсудить, прежде чем перейти к описанию алгоритма вычисления пар видимости, это понятие *особой пары* и действие процедуры корректировки (refine); они влияют на выполнение алгоритма жордановой сортировки с исправлением ошибок. Пара точек пересечения ∂V с L называется *особой*, если одна из них является первой, а другая — последней точками пересечения некоторой Г-границы (в порядке следования вдоль ∂V), иначе пара называется *нормальной*. (Заметим, что порядок следования точек пересечения Г-границы с прямой L при просмотре вдоль L либо совпадает с порядком следования этих точек при просмотре вдоль ∂V, либо точки вдоль L идут в обратном порядке по сравнению с ∂V.)

В результате обращения к процедуре *refine* $(\{x_{i-1}, x_i\}, x)$ происходит следующее. Точки x_{i-1} и x_i являются первой и последней точками пересечения некоторой Г-границы S с прямой L. Если S можно разбить на две части (Г-границы) S_1 и S_2, первой и последней точками пересечения которых будут точки x_{i-1}, x' и x'', x_i соответственно, и при этом $\{x', x''\}$ охватывает точку x, то процедура *refine* возвращает пару точек (x', x''). Если такое разбиение S выполнить нельзя, то процедура ничего не возвращает. Заметим, что если процедура *возвратила* пару точек (x', x''), то $\{x_{i-1}, x'\}$ и $\{x'', x_i\}$ являются особыми парами, а (x', x'') — нормальная пара, как того требует алгоритм жордановой сортировки.

Отметим еще один существенный момент, касающийся особых пар. Рассмотрим особую пару $\{x_{i-1}, x_i\}$, содержащую первую и последнюю точки пересечения одного глубокого граничного сегмента T с прямой. Если этот сегмент является верхним, то T вместе с некоторым отрезком прямой $y = y_{\max}$ составляют простую замкнутую кривую, содержащую внутри отрезок, соединяющий точки x_{i-1} и x_i, вне которой находится вся граница области видимости V, отличная от T. Согласно теореме о жордановой кривой, $\{x_{i-1}, x_i\}$ не может охватывать никаких других точек пересечения, кроме x_{i-1} и x_i. Таким образом, каждая особая пара либо может быть скорректирована, либо обладает свойством пустоты, как того требует алгоритм жордановой сортировки.

Теперь мы готовы к обсуждению самого алгоритма вычисления пар видимости. Входными данными для алгоритма служит описание одной области видимости V. Чтобы алгоритм можно было применить к исходному многоугольнику, мы преобразуем этот многоугольник в область видимости, разбив его границу на два боковых граничных сегмента, концевыми вершинами которых являются вершины исходного многоугольника с минимальной и максимальной y-координатой; y-координаты этих двух вершин выбираются в качестве значений $y_{\min}$ и $y_{\max}$ для получаемой таким образом области. Приводимый ниже алгоритм аналогичен алгоритму из разд. 1, за тем исключением, что он вычисляет лишь существенные точки пересечения ∂V и L (шаг 2) и использует жорданову сортировку с исправлением ошибок (шаг 3). Алгоритм включает следующие шаги:

Шаг 1. Задана область V, ограниченная прямыми $y = y_{\min}$, $y = y_{\max}$. Выберем некоторую вершину v области V с y-координатой, равной y_{cut}, удовлетворяющей условию $y_{\min} < y_{\text{cut}} < y_{\max}$. Если такой вершины v не существует, то останов: об-

ласть V не дает пар видимости. В противном случае обозначим через L прямую $y = y_{\text{cut}}$.

Шаг 2. Найти существенные точки пересечения ∂V с L в том порядке, в каком они встречаются при движении вдоль ∂V.

Шаг 3. Применить алгоритм жордановой сортировки с исправлением ошибок для сортировки по x-координате существенных точек пересечения и точек пересечения, полученных при выполнении процедуры корректировки (refine). Перечислить все пары видимости, соответствующие последовательным упорядоченным точкам пересечения в порядке следования вдоль L.

Шаг 4. Разрезать область V по прямой L, разбив ее на некоторую совокупность подобластей.

Шаг 5. Применить рекурсивно данный алгоритм к каждой подобласти, полученной на шаге [4].

Из сделанного выше замечания, касающегося особых пар и процедуры корректировки, следует, что шаг жордановой сортировки выполняется корректно. Всякая несущественная точка пересечения лежит при движении вдоль ∂V между точками, входящими в особую пару, и в случае необходимости может быть получена с помощью процедуры корректировки.

Последний вопрос, который необходимо обсудить, прежде чем переходить к анализу приведенного алгоритма, касается выполнения шага 4. Границы подобластей образуются следующим образом (рис. 10). Разрезать ∂V в каждой точке пересечения, входящей в упорядоченный список, полученный на шаге 3. Каждой из полученных таким образом частей соответствует пара в одном из деревьев, созданных алгоритмом жордановой сортировки, верхнем или нижнем. Каждая семья в каждом из деревьев, родительная вершина которой имеет нечетную глубину (считая, что фиктивные корни деревьев имеют нулевую глубину), соответствует подобласти. Граница этой подобласти состоит из частей ∂V, которым соответствуют пары, входящие в семью. Порядок следования этих частей совпадает с порядком соответствующих им пар в списке семьи в нижнем дереве или, в случае верхнего дерева, является обратным. Части границы ∂V чередуются с соответствующими отрезками прямой $y = y_{\text{cut}}$. В этом порядке следования есть *одно важное исключение*. В особом случае, когда некоторая часть D границы ∂V соответствует Г-границе, прежде чем включить D в качестве составляющей в границу подобласти, необходимо каждый отрезок прямой $y = y_{\min}$ или $y = y_{\max}$ заменить соответствующим отрезком прямой $y = y_{\text{cut}}$. Отметим, что это не влияет на представление

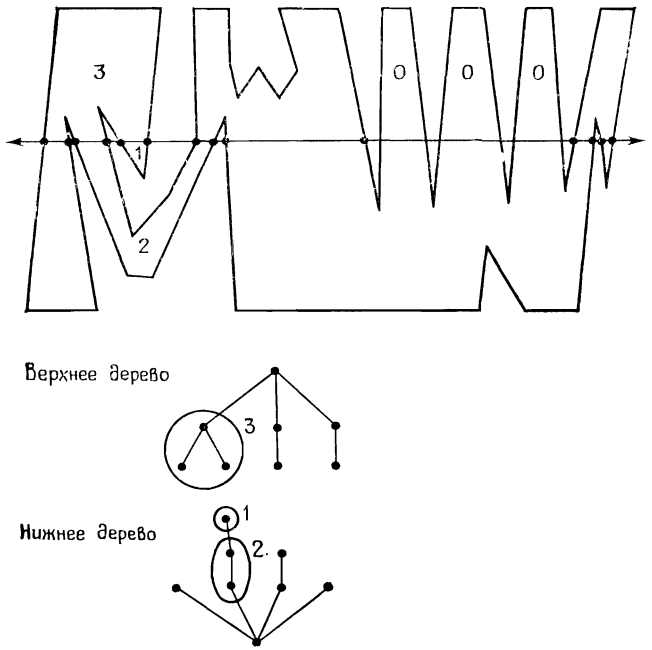


Рис. 10. Образование подобластей. На верхнем и нижнем деревьях выделены семьи вершин дерева, соответствующих подобластям. Трапеции, отмеченные на рисунке цифрой 0, полностью игнорируются.

Г-границы, а значит, не требуется никаких изменений в структуре данных, представляющей Г-границу. Более того, области видимости, отсекаемые в результате такой замены и, следовательно, игнорируемые алгоритмом при обработке, представляют собой трапеции. Если бы алгоритм вычисления пар видимости обрабатывал их, он завершил бы свою работу на шаге 1. Эффективность алгоритма вычисления пар видимости достигается за счет того, что он избегает каких-либо вычислений, связанных с такими тривиальными областями видимости.

Рассмотрим теперь, как определяются значения $y_{\min}$ и $y_{\max}$ для получаемых подобластей. Возьмем некоторую подобласть, лежащую над прямой L. Ей соответствует семья в верхнем дереве. Значение $y_{\min}$ для этой подобласти устанавливается равным y_{cut}. Значение $y_{\max}$ для подобласти равно $y_{\max}$ для исходной области, если родительская вершина семьи имеет глубину 1 в верхнем дереве. В противном случае $y_{\max}$ для подобласти берется равным максимальному значению y-координаты среди всех вершин, принадлежащих части границы ∂V, соответствую-

щей родительской вершине семьи. В последнем случае часть границы ∂V, соответствующая родительской вершине, состоит из одной части граничного сегмента области V. Разделим эту часть на две, разорвав ее в вершине с максимальной y-координатой. Две получившиеся части образуют боковые сегменты подобласти. Верхняя граница подобласти оказывается пустой. Для подобластей, лежащих ниже прямой L, значения $y_{\min}$ и $y_{\max}$ определяются симметричным образом.

Давайте вновь сформулируем, в чем состоит различие между приведенным выше алгоритмом и алгоритмом, описанным в разд. 1. В первом из них максимальная Г-граница рассматривается так, как если бы она имела лишь две точки пересечения с L (существенные точки пересечения), до тех пор пока не будет обнаружена точка пересечения, не принадлежащая этой Г-границе и находящаяся между точками пересечения Г-границы с L. Такой подход оказывается возможным благодаря тому, что порядок точек пересечения Г-границы с L оказывается одним и тем же (или обратным) при просмотре вдоль границы и вдоль прямой L. Если таких «чужих» точек пересечения нет, то алгоритм из разд. 1 просто рассекал бы границу между каждой парой смежных граничных сегментов на шаге 2 и опять соединял бы их в точности в том же самом порядке на шаге 4. Новый алгоритм избегает этой ненужной работы.

Теперь мы хотим оценить количественно, насколько новый алгоритм эффективнее. Наш основной результат состоит в том, что полное число пар видимости, обнаруженное при обработке n-угольника, равно $O(n)$. Отсюда следует, что время, затрачиваемое на жорданову сортировку, без учета обращений к процедуре *refine* равно $O(n)$.

Лемма 3. *При обработке n-угольника требуется не более $n-2$ раз выполнить шаги* 2—4 *алгоритма.*

Доказательство. Каждая вершина многоугольника, за исключением вершин с максимальной и минимальной y-координатой, может быть выбрана в качестве вершины v на шаге 1 не более одного раза. ■

Лемма 4. *Рассмотрим однократное выполнение шага* 3. *Пусть k — число пар видимости, найденных при этом выполнении шага* 3, *которые не были найдены ранее. Полное число точек пересечения, отсортированных на шаге* 3, *включая точки пересечения, полученные в результате выполнения процедуры корректировки* refine, *равно $O(k+1)$.*

Доказательство. Подсчитаем сначала существенные пересечения. Назовем существенную точку пересечения *x хорошей,*

если x является вершиной многоугольника (т. е. $x = v$) или если две вершины, предшествующая x и следующая за x вдоль ∂V, которые мы обозначим v' и v'', находятся в одной и той же части ∂V (верхней, нижней, левой или правой) и не лежат строго по одну сторону от L. В противном случае точку пересечения x будем называть *плохой*. Утверждается, что если x — хорошая точка пересечения, отличная от v, то сторона многоугольника, содержащая x, принадлежит некоторой впервые найденной паре видимости. Это утверждение справедливо, если часть границы ∂V от v' до v'' состоит от отрезка, соединяющего v' и v'', так как пара видимости, которая содержит сторону многоугольника с вершинами v' и v'' и которая будет обнаружена алгоритмом, является первой из уже найденных пар видимости, содержащей эту сторону многоугольника. Утверждение также справедливо, если часть границы V от v' до v'' состоит из усеченной стороны многоугольника, идущей из v' до прямой $y = y_{\max}$ (или $y = y_{\min}$), отрезка прямой $y = y_{\max}$ (или соответственно $y = y_{\min}$) и усеченной стороны многоугольника, идущей от прямой $y = y_{\max}$ (или соответственно $y = y_{\min}$) в вершину v''. Чтобы убедиться в этом, без ограничения общности предположим, что вершина v' находится строго ниже прямой L и ∂V содержит усеченную сторону многоугольника, идущую из v' до прямой $y = y_{\max}$ (рис. 11). Точка пересечения x лежит на этой усеченной стороне. Если смотреть вдоль направления прямой L со стороны, выходящей из v', то будет видно нечто иное по сравнению с тем, что видно со стороны, входящей в v'', и это различие будет отражено в новых парах видимости. Таким образом, в обоих случаях сделанное утверждение оказывается справедливым. Из этого утверждения следует, что число хороших точек пересечения равно $O(k + 1)$.

Рассмотрим теперь плохие точки пересечения. Любой смешанный граничный сегмент содержит не более двух плохих точек пересечения (первое и последнее пересечение в сегменте) и, если он считается верхним или нижним сегментом, по крайней мере одну хорошую точку пересечения. Любая максимальная Г-граница содержит не более двух плохих точек пересече-

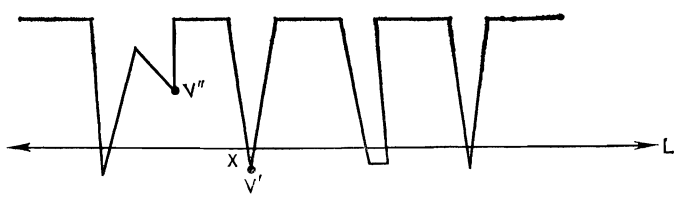


Рис. 11. Сторона многоугольника, содержащая точку пересечения x, принадлежит новой паре видимости.

ния (первое и последнее пересечения в Г-границе). Такая Г-граница либо (i) предшествует мелкому сегменту, либо (ii) предшествует смешанному верхнему или нижнему граничному сегменту, либо (iii) содержит последний сегмент на своей стороне. В случае (i) последнее пересечение является хорошим. Число плохих пересечений в случае (ii) равно $O(k+1)$, так как смешанный сегмент содержит по крайней мере одно хорошее пересечение. На случай (iii) приходится не более четырех плохих пересечений (два последних пересечения внутри каждой верхней и нижней границ). Кроме того, имеется не более двух плохих пересечений внутри как левой, так и правой границ. Таким образом, число плохих пересечений равно $O(k+1)$.

Остается подсчитать несущественные точки пересечения, получаемые при корректировке. Предположим, что при обращении к процедуре *refine* $(\{x_{i-1}, x_i\}, x)$ она возвращает пару x', x''. Каждая из сторон многоугольника, содержащая x' или x'', будет входить в новые пары видимости, обнаруженные алгоритмом, так как вдоль прямой L с каждой из этих сторон видно не только другую сторону. Таким образом, число точек пересечения, полученных при корректировке, равно $O(k)$.

Следующая теорема подводит итог проведенному анализу алгоритма.

Теорема 1. *При обработке n-угольника алгоритм вычисления пар видимости выдает $O(n)$ пар видимости, затратив при этом на жорданову сортировку время $O(n)$ без учета времени на обращение к процедуре корректировки refine.*

Доказательство. Доказательство этой теоремы непосредственно следует из лемм 1, 3 и 4. ■

4. ИСПОЛЬЗОВАНИЕ МЕТОДА СБАЛАНСИРОВАННОГО РАЗБИЕНИЯ НА ЧАСТИ

Алгоритм вычисления пар видимости, приведенный в разд. 3, в худшем случае может порождать области, содержащие в общей сложности $\Omega(n^2)$ граничных сегментов (граничный сегмент учитывается каждый раз, когда он входит в некоторую область). Как будет показано в разд. 5, для того чтобы данный алгоритм имел временную сложность $O(n \log\log n)$, необходимо, чтобы суммарное число граничных сегментов не превышало $O(n \log n)$. Мы достигаем этого, модифицировав алгоритм таким образом, чтобы использовать в нем стратегию сбалансированного разбиения на части (сбалансированный метод «разделяй и властвуй»). Предлагаемая модификация заключается в более тщательном выборе вершины v на шаге 1. Грубо говоря, мы хотим разрезать

область с большим числом граничных сегментов таким образом, чтобы по крайней мере нужная нам доля сегментов заканчивалась в какой-либо одной из подобластей. Процедура, которая обеспечивает такой выбор вершины и может быть быстро выполнена, состоит в следующем.

Выбор вершины для разреза. Пусть область V имеет t верхних и b нижних граничных сегментов. Предположим, что $t \geqslant b$. (В противном случае обработка ведется симметричным образом.) Если $t \leqslant 2$, то выбрать любую вершину v, y-координата которой не входит в множество $\{y_{\min}, y_{\max}\}$. В противном случае разделить список граничных сегментов на три подсписка так, чтобы первый и последний содержали по $\lfloor t/3 \rfloor$ сегментов, а средний подсписок — все оставшиеся. Среди вершин сегментов среднего подсписка выбрать вершину v, имеющую наименьшую y-координату.

Будем называть модифицированный алгоритм вычисления пар видимости, использующий такую стратегию выбора вершины, *алгоритмом со сбалансированным разбиением.* Алгоритм, использующий некоторую произвольную стратегию выбора, будем называть *исходным* алгоритмом. В следующем ниже анализе два граничных сегмента будут считаться различными лишь в случае, если различны множества их вершин или они имеют различные концевые стороны (если таковые вообще имеются).

Лемма 5. *При обработке n-угольника исходный алгоритм вычисления пар видимости создает в целом $O(n)$ различных граничных сегментов.*

Доказательство. Единственный способ, которым алгоритм может создать новые граничные сегменты, — это разрезать старый граничный сегмент на два в точке пересечения, являющейся исходными данными для алгоритма жордановой сортировки, или в вершине с максимальной или минимальной y-координатой среди вершин подобласти. Согласно теореме 1, число таких вершин, по которым разрезаются сегменты, равно $O(n)$. Следовательно, имеется $O(n)$ различных граничных сегментов. ■

Лемма 6. *Пусть V — область с s граничными сегментами. Если V разбивается на части с использованием стратегии сбалансированного разбиения, то ни одна из получившихся подобластей не будет содержать более $7s/8$ исходных граничных сегмента области V.*

Доказательство. Пусть V содержит t верхних и b нижних граничных сегментов. Имеет место равенство $s = b + t + 2$. Без потери общности предположим, что $t \geqslant b$. Если $t \leqslant 2$, то спра-

ведливость леммы очевидна, так как некоторый граничный сегмент разрезается на новые сегменты, отличные от исходных, так что исходный сегмент не появится ни в одной подобласти. Предположим, что $t \geqslant 3$. Рассмотрим, где заканчиваются верхние сегменты области V после разбиения ее на части. Сегмент, являющийся смешанным относительно разбивающей прямой L, разбивается на несколько новых граничных сегментов, отличных от исходных, так что исходный сегмент не появится ни в одной подобласти. Имеется не более $\lfloor t/3 \rfloor$ глубоких верхних граничных сегментов. Отсюда следует, что каждая подобласть, лежащая ниже L, содержит не более $b+2t/3+2 \leqslant 7s/8$ граничных сегментов области V. Глубокий верхний сегмент не может заканчиваться в подобласти, лежащей выше прямой L; это возможно лишь для части этого сегмента, образующей отдельный граничный сегмент. Таким образом, среди верхних сегментов лишь мелкие сегменты могут заканчиваться в подобластях, лежащих выше L. Множество мелких сегментов может заканчиваться в одной и той же подобласти лишь в том случае, если оно образует M-границу. С учетом выбора прямой L любая такая M-граница может содержать не более $t - \lfloor t/3 \rfloor - 1 \leqslant 2t/3$ верхних граничных сегментов области V. Таким образом, любая подобласть, лежащая выше L, содержит не более $b + 2t/3 + 2 \leqslant 7s/8$ граничных сегментов области V. ■

Теорема 2. *При применении алгоритма вычисления пар видимости, использующего стратегию сбалансированного разбиения, к n-угольнику сумма по всем областям числа граничных сегментов в области равна* $O(n \log n)$.

Доказательство. Один из способов доказательства этой теоремы состоит в том, чтобы, основываясь на лемме 6, выписать рекуррентное соотношение и решить его. Вместо этого мы применим «амортизационный» метод доказательства, основанный на анализе кредитов (см. [26]). При создании алгоритмом нового граничного сегмента мы будем приписывать этому сегменту значение $15\log_{16/15} n$, называемое кредитом. В дальнейшем всякий раз, когда этот сегмент будет появляться в некоторой области, кредит будет уменьшаться на единицу. Мы покажем, что сумма кредитов всегда остается неотрицательной. Отсюда, согласно лемме 5, следует, что сумма по всем областям числа граничных сегментов этих областей равна $O(n \log n)$.

Чтобы показать, что сумма кредитов остается неотрицательной, мы в действительности докажем следующее более сильное утверждение об *инвариантности кредита*: область с s граничными сегментами имеет кредит не менее $s \log_{16/15} s$. Очевидно, что исходно это утверждение истинно. Предположим, что

оно истинно и в некоторый момент обработки перед началом выполнения шагов 1—4. Пусть V — область, подвергаемая разбиению, а s — число граничных сегментов этой области. До разбиения область V имеет кредит не менее $s \log_{16/15} s$, который мы распределяем по $\log_{16/15} s$ на каждый граничный сегмент. Одна единица из этой величины вычитается как «плата» за появление сегмента в области V. Оставшийся кредит $(\log_{16/15} s) - 1 = \log_{16/15}(15s/16)$ представляет собой вклад сегмента в кредит подобласти, в которой он появляется. Рассмотрим подобласть V', образующуюся при разбиении V. Предположим, что ее граница содержит p граничных сегментов области V и q вновь созданных граничных сегментов. Положим $s' = p + q$. Чтобы проверить, что V' имеет кредит $s' \log_{16/15} s'$, рассмотрим два случая. Если $q > s'/15$, то сумма всех кредитов не меньше суммы кредитов новых сегментов:

$$15q \log_{16/15} n \geqslant s' \log_{16/15} n \geqslant s' \log_{16/15} s'.$$

Если $q \leqslant s'/15$, то поскольку ,согласно лемме 6, $p \leqslant 7s/8$, имеем $s' \leqslant 15s/16$. Сумма всех кредитов равна

$$\begin{aligned} &p \log_{16/15}(15s/16) + 15q \log_{16/15} n \geqslant \\ &\geqslant p \log_{16/15}(15s/16) + 15q \log_{16/15}(15s/16) \geqslant \\ &\geqslant s' \log_{16/15}(15s/16) \geqslant s' \log_{16/15} s'. \end{aligned}$$

Применив индукцию по числу шагов, убеждаемся в истинности утверждения об инвариантности кредита. С учетом сказанного ранее это доказывает теорему. ■

5. ПРЕДСТАВЛЕНИЕ ГРАНИЦЫ ОБЛАСТИ С ИСПОЛЬЗОВАНИЕМ ДЕРЕВЬЕВ ПОИСКА С УКАЗАТЕЛЯМИ

Мы почти завершили описание алгоритма вычисления пар видимости. Единственная задача, которую нам осталось решить, это выбрать структуру данных для списков, представляющих граничные сегменты и их группы, и проанализировать последствия этого выбора. Для представления списков обоих типов мы используем неоднородные деревья поиска с указателями (см. приложение). Как мы увидим, это обеспечивает временную сложность $O(n \log\log n)$ для алгоритма вычисления пар видимости, использующего стратегию сбалансированного разбиения.

Мы представляем каждый граничный сегмент как неоднородное дерево поиска с указателями, в котором каждый лист содержит вершину, входящую в этот сегмент. Порядок следования вершин при просмотре дерева слева направо соответствует по-

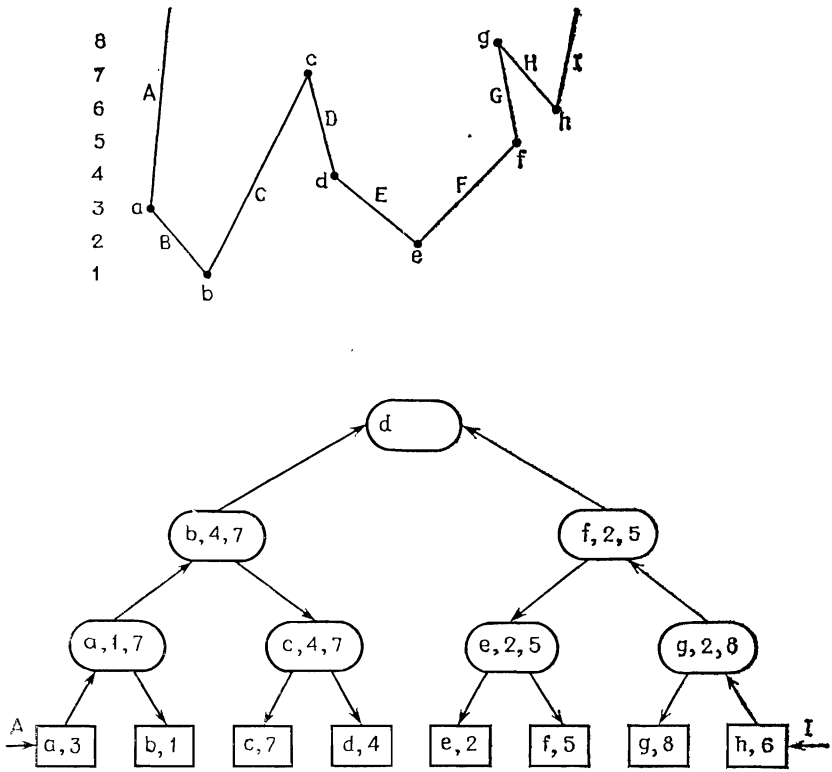


Рис. 12. Граничный сегмент и его представление с помощью неоднородного дерева поиска с указателями. (Обозначения, принятые при изображении дерева, объясняются на рис. 22.)

рядку прохождения вершин при движении вдоль сегмента. Кроме того, если сегмент имеет одну или две концевые стороны, то эти стороны запоминаются вместе с деревом. В дереве поддерживаются два вторичных порядка, оба связанные со значением y-координаты вершин. Первый определяет упорядоченность вершин по возрастанию y-координаты, а второй — по уменьшению y-координаты. Это значит, что каждая вершина дерева содержит максимальное и минимальное значения y-координаты вершин многоугольника, соответствующих листьям дерева, достижимых из этой вершины дерева (рис. 12). Это позволяет для любого заданного значения y найти самую левую (или самую правую) вершину многоугольника, y-координата которой меньше (или больше) заданного значения, за время $O(1 + \log(\min\{d, s - d\} + 1))$, где s — полное число вершин

многоугольника, запомненных в дереве, а d — позиция найденной вершины. За это же время можно найти вершину многоугольника с максимальной (или минимальной) y-координатой. Приведенное время операции расщепления дерева по позиции с номером d также равно $O(1 + \log(\min\{d, s - d\} + 1))$.

Для представления каждого списка верхних и нижних граничных сегментов, составляющих верхнюю или нижнюю границу области, используются неоднородные деревья поиска с указателями, в которых каждый лист представляет граничный сегмент. При просмотре дерева слева направо порядок следования граничных сегментов совпадает с порядком их прохождения при обходе границы в направлении по часовой стрелке. Каждый лист дерева содержит указатель на дерево, представляющее соответствующий граничный сегмент, а также на концевые вершины или стороны этого сегмента и максимальное и минимальное значения y-координаты вершин многоугольника, входящих в сегмент. Мы рассматриваем каждую вершину дерева как представляющую подсписок граничных сегментов, соответствующих множеству листьев, достижимых из этой вершины дерева. Для каждого такого подсписка запоминаются также первая и последняя концевые вершины или стороны, число входящих в него сегментов и максимальное и минимальное значения y-координаты вершин, входящих в сегменты подсписка (рис. 13). Вся эта информация может быть изменена, при этом обновление информации в вершинах дерева проводится в порядке движения снизу вверх для внутренних вершин дерева и сверху вниз для вершин дерева, лежащих на левом и правом путях.

Все перечисленные ниже операции могут быть выполнены за время $O(1 + \log(\min\{d, s - d\} + 1))$, где d — позиция найденного элемента, а s — общее число сегментов, запомненных в дереве.

(1) Найти в дереве самый левый (или самый правый) сегмент, содержащий вершину многоугольника, y-координата которой меньше (или больше) заданного значения.

(2) Найти сегмент в позиции d.

(3) Предположим, что все вершины сегмента лежат строго выше (или строго ниже) заданной горизонтальной прямой L и все концевые стороны сегментов пересекают L. Для заданного значения x найти в дереве самый левый (или самый правый) сегмент, сторона которого пересекает прямую L, и при этом x-координата точки пересечения меньше (или больше) x. (Возможны все четыре варианта поиска: самый левый сегмент с x-координатой точки пересечения, меньшей x, самый левый сегмент с x-координатой точки пересечения, большей x, и т. д.)

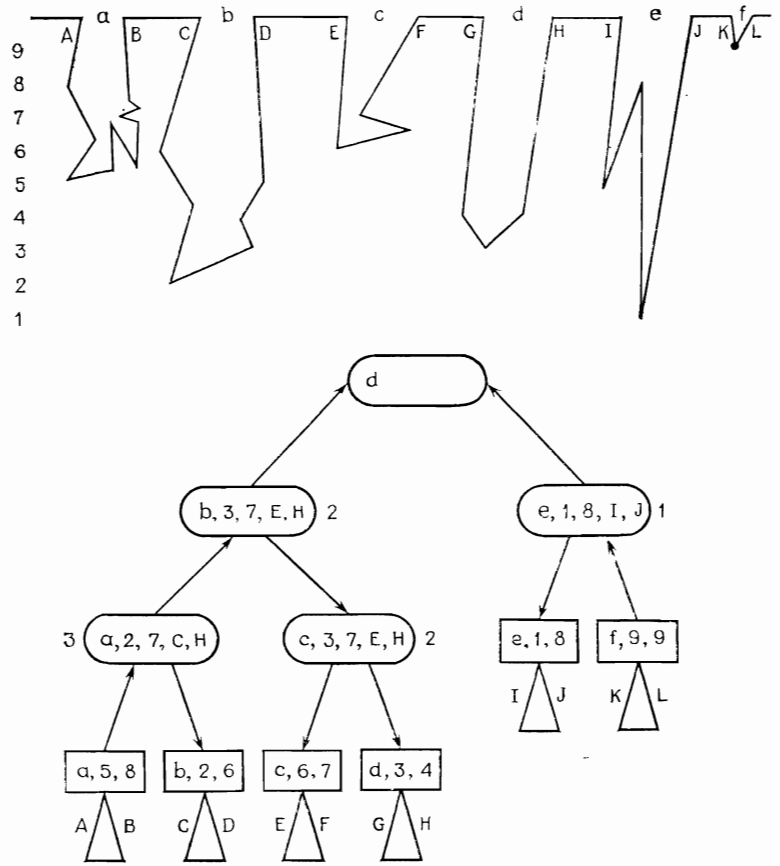


Рис. 13. Группа граничных сегментов и ее представление с помощью неоднородного дерева поиска с указателями. (Обозначения, используемые при изображении дерева, объясняются на рис. 22; прописными буквами обозначены концевые стороны.)

Кроме того, приведенное время операции вставки нового сегмента вслед за элементом в позиции d составляет $O(1 + \log(\min\{d, s-d\}+1))$, где s — общее число сегментов, запомненных в дереве. Приведенное время операции конкатенации двух деревьев равно $O(1)$.

Теперь рассмотрим, какие операции над структурами данных, представляющими границы областей, требуется выполнить на каждом шаге алгоритма вычисления пар видимости, использующего стратегию сбалансированного разбиения. Мы

последовательно опишем выполнение каждого шага алгоритма, приводя в некоторых случаях временные оценки.

Шаг 1. Пусть задана область V. Определим число верхних и нижних граничных сегментов в границе области V, обозначив их соответственно t и b (время $O(1)$). Предположим, что $t \geqslant b$. (Альтернативный случай рассматривается симметричным образом.) Если $t \leqslant 2$, выберем сегмент границы, содержащий некоторую вершину многоугольника, y-координата которой лежит строго между $y_{\min}$ и $y_{\max}$, и выберем в качестве v первую такую вершину в этом сегменте (время $O(1)$). Если $t \geqslant 3$, то разделим список верхних граничных сегментов на три части. В средней части найдем вершину с минимальной y-координатой, которую и возьмем в качестве v (затраты времени на эту операцию будут проанализированы ниже). Тем самым определяется прямая L, по которой производится разбиение области v.

Шаг 2. Расщепим дерево, представляющее верхнюю границу, между каждой парой сегментов, имеющих различные типы: *мелкий*, *глубокий*, *смешанный* (затраты времени на эту операцию будут проанализированы ниже). Такое расщепление разбивает список верхней границы на три части: смешанные сегменты, максимальная Г-граница и максимальная М-граница. Повторим эту операцию с деревом, представляющим нижнюю границу (затраты времени анализируются ниже). Расщепим дерево, содержащее вершину, лежащую на L, по этой вершине, поместив ее в оба полученных в результате расщепления дерева (затраты времени анализируются ниже). Каждому из выполненных расщеплений соответствует существенная точка пересечения границы области с прямой L. Сформируем список этих точек пересечения в том порядке, в каком они располагаются вдоль границы области. Это делается путем просмотра списков деревьев, представляющих Г-границы и новые граничные сегменты, образованные при расщеплении смешанных сегментов (время $O(1)$ на одно существенное пересечение).

Шаг 3. Для того чтобы выполнить вызов процедуры *refine* $(\{x_{i-1}, x_i\}, x)$, рассмотрим дерево, представляющее Г-границу, первой и последней точками пересечения которой с L являются соответственно x_{i-1} и x_i. Предположим, что $x_{i-1} < x_i$ (альтернативный случай симметричен). Если x-координата точки последнего пересечения этого сегмента с L больше x, то процедура *refine* ничего не возвращает (время $O(1)$). В противном случае расщепить Г-границу между этим и следующим за ним сегментами и возвратить в качестве x' и x'' соответственно последнюю точку пересечения этого сегмента в первую точку пересечения

следующего за ним сегмента (время выполнения этой операции будет проанализировано ниже).

Шаг 4. Для каждой пары в верхнем дереве, глубина которой нечетна и больше единицы, выполним расщепление дерева, представляющего соответствующий граничный сегмент, по вершине этого сегмента, имеющей максимальную y-координату (время

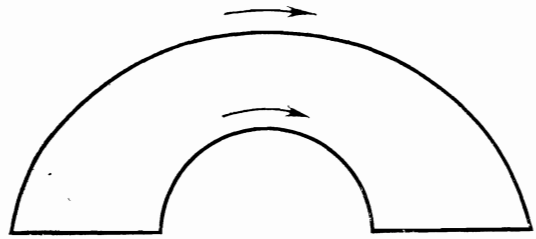


Рис. 14. Подобласть, приводящая к ошибке на шаге 4 алгоритма. Стрелки указывают направление прохождения двух граничных сегментов в исходном многоугольнике. Подобные подобласти не могут появиться при обработке простого многоугольника.

выполнения этой операции будет проанализировано ниже). Поместим вершину, по которой произведено расщепление, в оба получившихся в результате расщепления дерева. Аналогичным образом (с учетом симметрии) обработаем граничные сегменты, соответствующие парам в нижнем дереве. Для каждой семьи в обоих деревьях выполним конкатенацию граничных сегментов и Г-границ, соответствующих парам этой семьи, получив в результате границу подобласти, соответствующей семье (время $O(1)$ на одну пару). Так как обрабатываемый многоугольник является простым, то порядок расположения вершин многоугольника на границе подобласти согласуется с их порядком в исходном многоугольнике (рис. 14). Это значит, что при конкатенации никогда не надо будет изменять на обратный порядок граничных сегментов.

Теперь давайте проанализируем время выполнения этой реализации алгоритма сбалансированного разбиения. Заметим, что для каждого сегмента, найденного среди сегментов верхней или нижней границы, и для каждой вершины, найденной в сегменте, расщепление производится по этому сегменту или вершине. Так как временные оценки для операций поиска и расщепления одинаковы, то время, затраченное на расщепление, доминирует над временем, затрачиваемым на поиск таких сегментов и вершин, включая время на поиск y-координат вершин многоугольника, по которым производится расщепление, на шаге 1. Таким

образом, учитывая оценки, приведенные выше, и теорему 1, получаем, что полное время работы алгоритма равно $O(n)$ плюс время выполнения операций вставки в дерево, расщепления и конкатенации деревьев.

Остается оценить временные затраты, связанные с операциями, изменяющими структуры данных. Каждая операция вставки производится с одного из концов дерева поиска с указателями, и, следовательно, приведенное время этой операции равно $O(1)$. Изменения деревьев, представляющих сегменты, связаны лишь операциями расщепления и вставки, число которых равно $O(n)$. Полное время, затрачиваемое на эти изменения, удовлетворяет рекуррентному соотношению, в сущности совпадающему с соотношением (1) (см. разд. 1), и, следовательно, равно $O(n)$.

Изменения, производимые над деревьями, представляющими верхнюю и нижнюю границы, включают в конкатенацию, а значит, рекуррентное соотношение (1) в этом случае уже неприменимо. Для анализа этих операций заметим сначала, что приведенное время одной операции конкатенации или вставки равно $O(1)$. Так как полное число таких операций равной $O(n)$, то полное время выполнения всех операций составляет $O(n)$.

Для анализа $O(n)$ операций расщепления рассмотрим некоторую область видимости V_1, полное число граничных сегментов которой равно s_i. Полное время, затрачиваемое на расщепление двух деревьев, представляющих верхнюю и нижнюю границы, равно

$$O\left(k_i + \sum_{j=0}^{k_i+1} \log s_{i,j}\right),$$

где k_i — полное число расщеплений, а $s_{i,0}, s_{i,1}, \ldots, s_{i,k_i+1}$ — число сегментов в каждом из деревьев, созданных в результате выполнения операций расщепления. (Начав с двух деревьев, в результате k_i расщеплений получим k_i+2 деревьев; если исходно имеется только одно дерево, то мы считаем, что $s_{i,k_i+1}=1$.) Для всех j имеем $s_{i,j} \geqslant 1$ и

$$\sum_{j=0}^{k_i+1} s_{i,j} \leqslant s_i + 1.$$

Так как логарифмическая функция является вогнутой, то оценка времени, затрачиваемого на расщепление, имеет максимум, когда все получаемые части имеют одинаковый «размер», что дает оценку $O(k_i + (k_i+2)\log((s_i+1)/(k_i+2)))$.

Оценивая полное время выполнения всех $O(n)$ операций расщепления, равное $O(\sum_i (k_i + (k_i+2)\log((s_i+1)/(k_i+2))))$,

мы представим его в виде суммы двух слагаемых и оценим каждое из них. Назовем область V_i *хорошей областью*, если $k_i \leqslant s_i/(\log n)^2$, и *плохой* в противном случае. Согласно теореме 2, $\sum_i s_i = O(n \log n)$, так что число расщеплений, имеющих место при обработке хорошей области, равно

$$\sum_{k_i \leqslant s_i/(\log n)^2} k_i = O(n/\log n).$$

Взяв $O(\log n)$ в качестве завышенной оценки времени выполнения этих операций расщепления, получаем для полного времени расщепления хороших областей оценку $O(n)$. Остается получить оценку для времени расщепления плохих областей:

$$O\left(\sum_{k_i \geqslant s_i/(\log n)^2} (k_i + (k_i + 2)\log(s_i + 1)/(k_i + 2))\right) =$$

$$= O\left(\sum_{k_i \geqslant s_i/(\log n)^2} (k_i + (k_i + 2)\log\log n)\right) = O(n \log\log n),$$

так как $\sum_i k_i = O(n)$.

На основании проведенного выше анализа мы приходим к выводу, что полное время выполнения алгоритма вычисления пар видимости, использующего стратегию сбалансированного разбиения, равно $O(n \log\log n)$.

6. ЗАМЕЧАНИЯ, ПРИЛОЖЕНИЯ И ОТКРЫТЫЕ ПРОБЛЕМЫ

Мы описали алгоритм вычисления СВ-пар видимости в горизонтальном направлении для простого многоугольника, имеющий временную сложность $O(n \log\log n)$. Путем линейного по времени сведения задачи триангуляции к задаче вычисления пар видимости [5, 9] мы получаем алгоритм с временной сложностью $O(n \log\log n)$, решающий задачу триангуляции. Главными составляющими нашего алгоритма являются жорданова сортировка, метод сбалансированного разбиения на части и деревья поиска с указателями. Представляет интерес следующее замечание: и алгоритм жордановой сортировки, и алгоритм вычисления пар видимости используют деревья поиска с указателями, но только разных типов. Алгоритм жордановой сортировки требует быстрого доступа в окрестность элемента, занимающего произвольную позицию в дереве, но при этом нет необходимости искать элементы по вторичному порядку. Этим требованиям удовлетворяют однородные деревья поиска с указателями. Алгоритм вычисления пар видимости сам по себе не нуждается в использовании поиска по вторичному порядку, но вместе с тем требует быстрого доступа лишь в окрестность

первого и последнего элементов. Этим требованиям удовлетворяют неоднородные деревья поиска с указателями. Наш алгоритм в полной мере использует указанные свойства этих структур.

Деревья поиска с указателями довольно сложны для использования в практической реализации алгоритма. Слитор и Тарьян [23] выдвинули гипотезу, что если вместо деревьев поиска с указателями использовать сплей-деревья (разновидность саморегулирующихся деревьев поиска), то оценка $O(n \log \log n)$ для временно́й сложности алгоритма остается справедливой. Использование сплей-деревьев, возможно, приведет к практически полезной реализации нашего алгоритма, хотя это предположение следует проверить экспериментально. С практической точки зрения в алгоритм следует внести еще некоторые менее значительные изменения. Этот вопрос мы оставим в качестве темы для дальнейших исследований.

Рассмотренный алгоритм вычисления пар видимости может быть изменен с тем, чтобы можно было обрабатывать многоугольники, имеющие вершины с одинаковыми y-координатами. Для обработки возникающих при этом касательных точек пересечения в процессе жордановой сортировки (см. разд. 2) мы представляем такую точку x_i фиктивной парой (x_i, x_i), добавляемой к нижнему дереву, если касание происходит с верхней стороны прямой L, по которой производится разбиение, или к верхнему дереву в противном случае. Другие связанные с этим изменения в алгоритме очевидны (см., например, [30]).

Эффективный алгоритм триангуляции имеет многочисленные применения в вычислительной геометрии. Обычно требуется выполнить триангуляцию (возможно, несколько раз) и некоторую линейную по времени предобработку или постобработку. В этом случае наш алгоритм триангуляции дает оценку $O(n \log \log n)$ временной сложности решения всей задачи в целом. Любое улучшение времени триангуляции будет давать соответствующее улучшение в прикладной задаче. Среди задач, в которых используется триангуляция, отметим следующие:

1) некоторые задачи разбиения многоугольников на части [9] (когда дополнительно не налагается условие минимальности разбиения, как, например, в [17]);

2) регуляризация (или триангуляция) некоторого разбиения плоскости, задаваемого связным планарным графом [8];

3) вычисления внутреннего расстояния между двумя точками внутри многоугольника и определение видимости многоугольника из точки, находящейся внутри многоугольника [3];

4) решение задачи о кратчайшем пути внутри многоугольника и вычисление информации о видимости внутренности мно-

гоугольника с некоторого отрезка, лежащего внутри многоугольника [11];

5) проверка многоугольников на пересечение и представление (декомпозиция) простых сплайн-многоугольников [24] в виде объединения разностей объединений выпуклых множеств [7];

6) определение трансляционной отделимости двух простых многоугольников [1];

7) определение кратчайшего пути наблюдателя при обходе простого многоугольника [6, 3].

Одно из важных применений нашего алгоритма вычисления пар видимости касается проверки, является ли многоугольник P

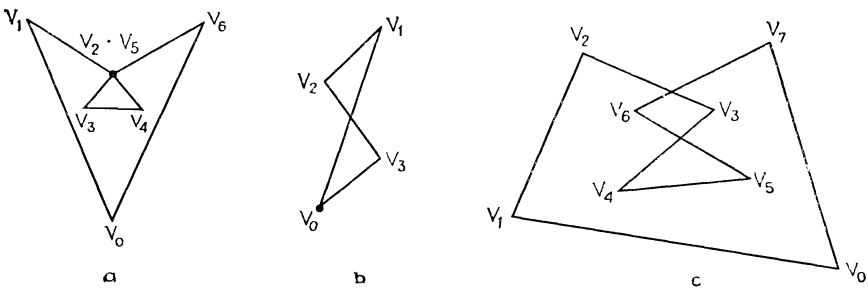


Рис. 15. Многоугольники, не являющиеся простыми и не вызывающие ошибок при применении к ним алгоритма вычисления пар видимости. Если в многоугольнике, показанном на рисунке (*c*), сделать разрез, проходящий через вершину v_5, то он «распадется» на три простых многоугольника.

с n вершинами простым, и определения точек самопересечения границы ∂P, если многоугольник не является простым. Мы покажем, как следует изменить наш алгоритм для того, чтобы можно было решать эти задачи за время $O(n \log \log n)$.

Хотя алгоритм жордановой сортировки с исправлением ошибок и обнаруживает в некоторых случаях, что многоугольник не является простым, столкнувшись с неподдающимися корректировке пересечениями или нарушениями свойства пустоты, однако успешное завершение алгоритма вычисления пар видимости не доказывает, что он действительно прост. Среди проблем, с которыми должен успешно справляться алгоритм, гарантированно проверяющий простоту многоугольника, укажем следующие: совпадение двух вершин — многоугольник касается самого себя (рис. 15, *a*); ни при какой согласованной разметке ребер, определяющей их ориентацию при обходе границы, невозможно определить внутренность многоугольника (рис. 15, *b*); наличие самопересечений границы, когда многоугольник, не

являясь простым, может быть разбит на части, каждая из которых является простым многоугольником (рис. 15, *c*).

Наш алгоритм проверки простоты многоугольника состоит в следующем. Сначала мы проверяем, что любые две последовательные стороны многоугольника P имеют пересечение, состоящее из одной точки — их общей вершины. Затем применяем алгоритм вычисления пар видимости к многоугольнику P и многоугольнику Q, полученному из P путем «выворачивания», как это определено при доказательстве леммы 2. Выполнение алгоритма прерывается и объявляется, что многоугольник P не является простым, если имеет место один из перечисленных ниже случаев.

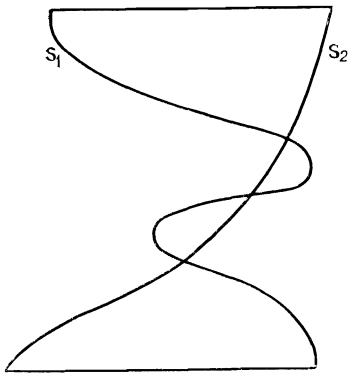


Рис. 16. Порядок четырех углов области видимости, показанной на рисунке, позволяет утверждать о том, что эта область не является простым многоугольником.

1) На шаге жордановой сортировки обнаруживается точка пересечения двух сторон многоугольника, отличная от общей вершины этих сторон (рис. 15, *a*).

2) На шаге жордановой сортировки обнаруживается неподдающееся корректировке пересечение или нарушение свойства пустоты.

3) На шаге 4 строится подобласть с граничными сегментами S_1 и S_2, про которые известно, что они пересекаются, так как порядок точек пересечения S_1 и S_2 с верхней ограничивающей прямой отличается от порядка точек пересечения этих сегментов с нижней ограничивающей прямой (рис. 16).

Если алгоритм вычисления пар видимости успешно завершает свою работу на обоих многоугольниках P и Q, то объявляется, что многоугольник P простой.

При обсуждении шага 4 алгоритм в разд. 5 мы отметили, что если многоугольник простой, то порядок вершин вдоль граничных сегментов области совместим с порядком вершин вдоль собранной из частей граничных сегментов области границы подобласти. Ясно, что если многоугольник не является простым, то на шаге 4 может появиться подобласть, граничные сегменты которой входят в границу в неправильном порядке, как показано на рис. 14. К счастью, такая ситуация не может реально произойти: если такая вложенная пара существует, то, согласно теореме о жордановой кривой, текущая обрабатываемая

область имеет непростую границу, и на самом деле сегменты границы, определяемые разбивающей прямой, не удовлетворяют свойству непересекаемости. Следовательно, тот факт, что многоугольник не является простым, будет обнаружен на шаге 3.

Теорема 3. *Алгоритм проверки многоугольника на простоту корректен.*

Доказательство. Несомненно, что если алгоритм проверки многоугольника на простоту выдает ответ, что P не является простым многоугольником, то ∂P имеет самопересечения. Предположим, что алгоритм определил, что P — простой многоугольник. В процессе вычисления пар видимости алгоритм породил два множества областей. Пусть **P** и **Q** — множества областей, порожденных при применении алгоритма вычисления пар видимости соответственно к многоугольникам P и Q. Каждая область в **P** и **Q** является либо трапецией, либо треугольником. Некоторые области из **Q** ограничены одной или более сторонами из множества $Q—P$, т. е. сторонами, добавленными к «вывернутому» многоугольнику P. Обозначим **Q′** множество областей, получаемых из областей множества **Q** путем их продолжения в бесконечность в направлении сторон, входящих в $Q—P$. Области, входящих в **Q′**, могут быть трапециями, треугольниками, полуплоскостями или бесконечными областями, ограниченными двумя горизонтальными прямыми и частью некоторой стороны многоугольника P.

Соблазнительно было бы сказать, что области из **P** образуют разбиение «внутренности» многоугольника P, но мы не можем так сказать, так как мы пока не знаем, *имеет ли* многоугольник P внутренность. Однако, учитывая, каким образом были получены области из **P**, мы знаем, что их можно склеить вместе по горизонтальным сторонам, определяющим пары видимости, таким образом, что получившаяся область будет топологически эквивалентна кругу. Это необходимое, но не достаточное условие для того, чтобы многоугольник P был простым (рассмотрите в качестве примера многоугольник на рис. 15, *c*). Алгоритм вычисления пар видимости легко модифицировать так, чтобы он выполнял такое «склеивание», или, более правильно, порождал двойственный граф «склеенной» области, в котором области представляются вершинами, и вершины, соответствующие областям, разделяемым горизонтальной стороной, определяющей пару видимости, соединяются ребром.

Вследствие успешности завершения алгоритма вычисления пар видимости также известно, что области, входящие в **Q**, могут быть склеены друг с другом по горизонтальным отрезкам, определяющим пары видимости. Получаемая в результате

такого склеивания область топологически эквивалентна кругу. Эта операция склеивания может быть естественным образом расширена на области, входящие в **Q**; при этом получается область, топологически эквивалентная плоскости с выколотой точкой. Теперь мы используем области, входящие в **P** и **Q′**, для построения отображения плоскости на себя.

Пусть C — окружность в плоскости, на которой выбраны n различных точек, соответствующих вершинам многоугольника P. Это позволяет естественным образом установить соответствие между точками границы многоугольника ∂P и точками окружности C. Для каждой BC- или CC-пары видимости в **P**, порождаемой алгоритмом вычисления пар видимости, соединим соответствующие точки некоторой кривой, лежащей внутри окружности C. Кривые будем проводить таким образом, чтобы они нигде не пересекались, за исключением концевых точек. (Двойственный граф областей, входящих в **P**, дает естественный и конструктивный способ реализации этой операции. Обрабатывая вершину двойственного графа, степень которой равна единице, мы должны провести кривую между двумя точками, лежащими на C. Эта кривая разбивает круг на две части, одна из которых может быть исключена из рассмотрения на последующих этапах построения отображения. Таким образом, полное построение требуемой совокупности кривых может быть осуществлено в результате последовательной обработки с удалением вершин двойственного графа, имеющих степень, равную единице, выполняемой до тех пор, пока двойственный граф не станет пустым.) Эти кривые разбивают область, лежащую внутри окружности C, на области, соответствующие областям **P**. Аналогичным образом для каждой пары видимости, связанной с **Q′**, проведем кривую, соединяющую две точки окружности C и лежащую вне C. Если один из элементов пары видимости находится в бесконечности, то соответствующая этой паре кривая, начинаясь на C, уходит в бесконечность. При этом кривые вне C будем проводить таким образом, чтобы не было пересечений. Эти кривые разбивают область вне окружности C на области, соответствующие области из **Q′** (рис. 17).

Можно построить непрерывное отображение h плоскости на себя, которое отображает каждую область внутри C на соответствующую область из **P** и каждую область, лежащую вне C, на соответствующую область из **Q′**. Отображение является отображением на, так как каждая точка плоскости принадлежит одной из областей, входящих в **P** или **Q′**. Действительно, пусть x — произвольная точка. Будем двигаться из точки x вправо в горизонтальном направлении до тех пор, пока не достигнем границы многоугольника ∂P. Если мы достигли ∂P изнутри, то

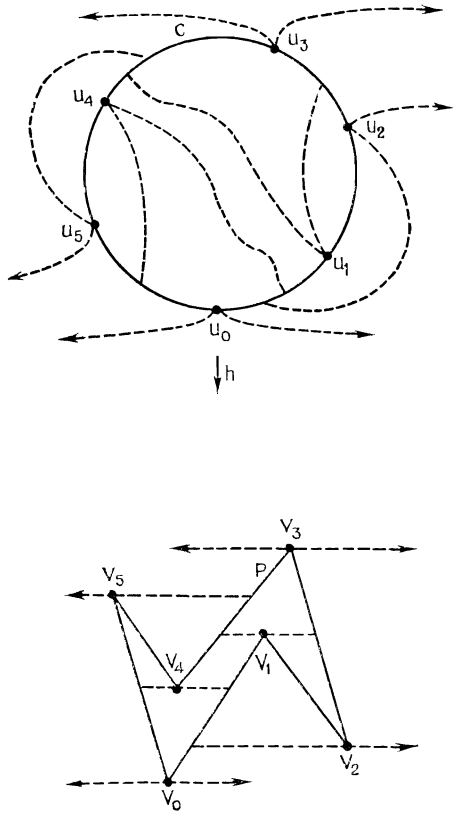


Рис. 17. Иллюстрация к построению отображения h в доказательстве теоремы 3. Для $0 \leqslant i \leqslant 5$ имеет место $h(u_i) = v_i$. Пунктирные кривые на верхнем рисунке соответствуют BC-парам видимости, показанным на нижнем рисунке. Стрелки на конце пунктирных линий означают, что эти линии уходят на бесконечность.

точка x принадлежит некоторой области из **P**. В противном случае точка x принадлежит некоторой области из $\mathbf{Q}'$. Если, двигаясь таким образом, мы не достигнем ∂P, то x принадлежит некоторой области из $\mathbf{Q}'$.

Так как множества **P** и $\mathbf{Q}'$ конечны и прообразы различных областей имеют непересекающиеся внутренности, то отображение h является накрывающим отображением плоскости на себя: каждая точка плоскости x имеет открытую окрестность N, такую что $h^{-1}(N)$ является непересекающимся объединением конечного числа открытых множеств [21]. Более того, так как область определения отображения h связна, то каждая точка

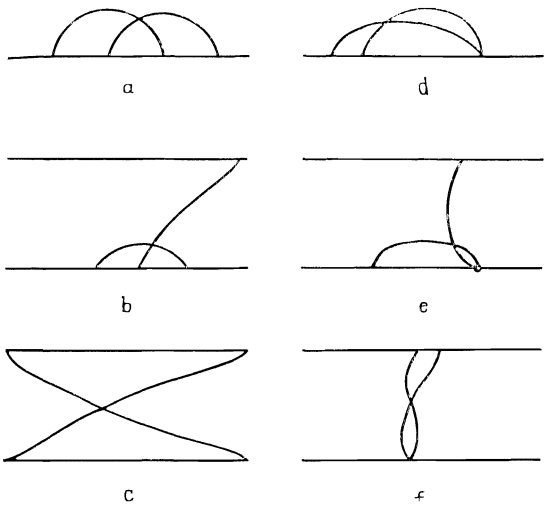


Рис. 18. Возможные варианты взаимного расположения пересекающихся граничных сегментов: (*a*) четыре точки лежат на одной ограничивающей прямой; (*b*) три точки лежат на одной ограничивающей прямой, а четвертая на другой; (*c*) по две точки на каждой ограничивающей прямой. Варианты (*d*), (*e*) и (*f*) представляют вырожденные случаи конфигураций вариантов (*a*), (*b*) и (*c*) соответственно. Многоугольник, показанный на рис. 15, *b*, представляет вырожденный случай варианта (*f*). Вырожденные случаи могут быть обнаружены в результате анализа внутреннего угла в точке (или в точках) вырождения.

в области значений h является образом (при отображении h) одного и того же числа точек из области значений [21, примеры 8—3.5]. Учитывая, что каждая точка, принадлежащая открытой полуплоскости, лежащей выше горизонтальной прямой, проходящей через вершину многоугольника P с наибольшей y-координатой, является образом (при отображении h) лишь одной точки, то отображение h является взаимно однозначным соответствием. Таким образом, h является гомеоморфизмом плоскости на себя. Тем самым доказано, что ∂P гомеоморфна окружности C и, следовательно, многоугольник P — простой. ■

Если алгоритм проверки на простоту обнаруживает, что многоугольник P не является простым, то конкретное подтверждение этого можно получить, затратив дополнительное время $O(n)$. Если имеет место случай (1), то мы непосредственно получаем точку самопересечения границы многоугольника. В других случаях ((2) и (3)) мы можем найти одну или две ограничивающие прямые и два граничных сегмента S_1 и S_2 с концами

на этих ограничивающих прямых, которые заведомо пересекаются. При этом возможна одна из семи конфигураций взаимного расположения сегментов. Эти конфигурации показаны на рис. 18 и 15, *b*. Для непосредственного выявления точки пересечения сегментов применим, возможно многократно, следующую последовательность действий.

Шаг 1. Выберем некоторую вершину, лежащую на одном из сегментов S_1 и S_2, y-координата которой не является ни минимальной, ни максимальной. Пусть L — горизонтальная прямая, проходящая через эту вершину.

Шаг 2. Найти все точки пересечения сегментов S_1 и S_2 с прямой L в том порядке, в каком они расположены вдоль S_1, а затем вдоль S_2.

Шаг 3. Применим алгоритм жордановой сортировки к полученным точкам пересечения. В результате либо будет явно определена точка самопересечения, либо будет обнаружено нарушение свойства непересекаемости. Если обнаружено нарушение свойства непересекаемости, то обозначим через S_1' и S_2' пересекающиеся подсегменты. Заменим S_1 и S_2 на S_1' и S_2'. Если и S_1', и S_2' состоят лишь из одной усеченной стороны каждый, то выдать точку пересечения этих сторон.

Анализ, аналогичный проведенному в разд. 3—5, показывает, что такая постобработка требует времени $O(n)$, если для представления сегментов используются деревья поиска с указателями. (При этом не требуется выполнять конкатенацию деревьев и производить сбалансированное разбиение.)

Алгоритмы проверки многоугольника на простоту и получение фактов, подтверждающих непростоту многоугольника, легко могут быть расширены для обработки связных незамкнутых ломаных. В качестве первого шага такого расширения проведем через концы ломаной прямую. Эта прямая разбивает ломаную на граничные сегменты, целиком лежащие по одну сторону от прямой. Выполним жорданову сортировку точек пересечения ломаной с прямой. Если алгоритм жордановой сортировки обнаружит нарушение свойства непересекаемости, то алгоритм поиска пересечения может быть применен непосредственно к двум обрабатываемым сегментам. В противном случае дальнейшую обработку будем производить аналогично шагу 4 алгоритма вычисления пар видимости (см. разд. 3), дополнив его таким образом, чтобы можно было учесть ситуации с противоположно направленными граничными сегментами (рис. 14). В результате мы построим многоугольники, целиком лежащие по одну сторону от прямой. Исходная ломаная является простой тогда и

только тогда, когда каждый из этих многоугольников является простым.

Алгоритмы вычисления информации о горизонтальной видимости, проверки на простоту и получения фактов, подтверждающих непростоту, могут быть расширены для обработки кривых, удовлетворяющих некоторым довольно слабым ограничениям (см., например, [22, 24]). Для этого нужно выполнить предобработку исходной кривой, разбив ее на части, каждая из которых представляет монотонную по y-координате кривую. Точки разбиения будут играть роль вершин ломаной, а дуги кривой между последовательными точками разбиения — роль сторон. К представленной таким образом кривой можно непосредственно применять указанные алгоритмы при условии, что имеются процедуры вычисления точек пересечения кривой с прямой. Корректность такого подхода вытекает из того факта, что если некоторая совокупность «сторон» кривой пересекает две горизонтальные прямые, то порядок пересечения обеих прямых «сторонами» кривой одинаков [1]). При применении алгоритма проверки на простоту нужно расширить его следующим образом: необходимо для каждой тривиальной области видимости проверять, не пересекаются ли ее левая и правая границы. Если все получаемые области обладают этим свойством, то исходная кривая является простой, иначе мы непосредственно получаем подтверждение ее непростоты. И предварительная обработка кривой, и постобработка получаемых областей видимости могут быть выполнены за время $O(n)$.

Ряд проблем, связанных с триангуляцией, по-прежнему остается открытым. Первая и, конечно, самая важная из них касается сложности задачи триангуляции: возможно ли разработать алгоритм триангуляции простого многоугольника с временной сложностью $O(n)$ или хотя бы алгоритм со сложностью $o(n \log\log n)$? Для решения этой проблемы, по-видимому, требуется новая идея. Один из возможных подходов заключается в разработке структуры данных для представления ломаной, позволяющей быстро вычислять точки пересечения ломаной с произвольным горизонтальным отрезком или хотя бы с произвольным горизонтальным лучом. Возможно, такую структуру данных можно построить, используя информацию, получаемую алгоритмом вычисления пар видимости, применяемым рекурсивно к небольшим участкам границы, имеющим размер, скажем, $O(\log n)$. Результатом такого подхода мог бы стать алгоритм вычисления пар видимости с временной сложностью $O(n \log^* n)$. Другая открытая проблема касается сложности

[1]) При условии, что кривая является простой. — *Прим. перев.*

определения всех точек самопересечения ломаной: можно ли для этой задачи получить лучшие оценки по сравнению с оценками для задачи поиска всех пересечений произвольной совокупности отрезков [4, 32*, 33*]?

Приложение. Деревья поиска с указателями. *Дерево поиска с указателями* — это разновидность сбалансированного дерева поиска, в котором с целью повышения эффективности доступа к элементам, находящимся в окрестности некоторых выделенных элементов, используются специальные *указатели*. Деревья поиска с указателями впервые ввели Гибас, Маккрейт, Плэс и Робертс [12], а затем они получили дальнейшее развитие в работах других исследователей [2, 15, 18, 28, 29]. Мы рассмотрим два вида деревьев поиска с указателями, обладающие несколько различными свойствами: *неоднородные* и *однородные деревья*. Наше обсуждение основано на использовании сбалансированного дерева специального вида, называемого *красно-черным деревом* [20, 25], хотя с таким же успехом можно было бы использовать другие разновидности сбалансированных деревьев, например a, b-деревья [16, 19]. Нас будет интересовать главным образом оценка приведенной сложности, связанной с использованием таких деревьев, а не оценки сложности в худшем случае.

Здесь мы будем подразумевать под *двоичным деревом поиска* полное двоичное дерево, в котором каждая внешняя вершина содержит элемент, выбранный из некоторого полностью упорядоченного универсума, причем порядок следования внешних вершин согласуется с общей упорядоченностью этого универсума. Каждая внутренняя вершина дерева содержит *ключ*, который также является элементом универсума. Ключ не меньше любого элемента в левом поддереве соответствующей вершины и меньше любого элемента в правом поддереве. Ключи можно использовать для поиска наибольшего элемента в дереве, не превосходящего заданного элемента. Поиск начинается с корня дерева и продолжается в левом поддереве, если ключ в текущей вершине больше или равен заданному элементу, в противном случае поиск продолжается в правом поддереве. Поиск продолжается до тех пор, пока не будет достигнута какая-либо внешняя вершина дерева. Искомым элементом является либо элемент в первой достигнутой внешней вершине, либо элемент, содержащийся в предыдущей внешней вершине, которую можно найти, если вернуться назад по пути, пройденному при поиске, до внутренней вершины, в которой для поиска было выбрано правое поддерево, в этой точке выбрать левое и снова двигаться вниз до внешней вершины, каждый раз выбирая

правую ветвь. Время поиска элемента пропорционально глубине дерева.

Красно-черное дерево — это двоичное дерево поиска, все вершины которого раскрашены в два цвета: *красный* и *черный*. Цвета вершин удовлетворяют следующим условиям (рис. 19):

(1) все внешние вершины являются черными;

(2) все пути из корня в любую внешнюю вершину содержат одинаковое число черных вершин;

(3) родительская вершина (если она существует) любой красной вершины является черной вершиной.

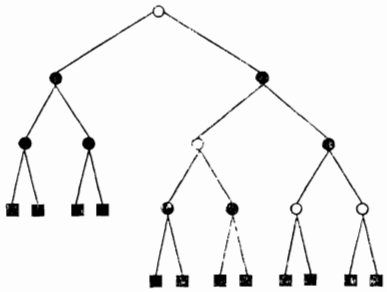


Рис. 19. Красно-черное дерево. Закрашенные вершины имеют черный цвет, а незакрашенные — красный. (На рисунке показаны лишь цвета вершин.)

Глубина красно-черного дерева, содержащего n элементов, равна $O(\log n)$. Чтобы вставить новый элемент в красно-черное дерево, мы сначала ищем в нем наибольший элемент, меньший вставляемого. Достигнув в процессе поиска внешней вершины, мы заменяем ее внутренней вершиной, имеющей двух сыновей (потомков): старую внешнюю вершину и новую внешнюю вершину, содержащую вставляемый элемент. Новая внутренняя вершина содержит в качестве ключа наименьший из двух элементов, содержащихся в вершинах-сыновьях этой вершины. Эта новая внутренная вершина окрашивается в красный цвет. Это может привести к нарушению условия (3), которое должно выполняться для красных вершин. Чтобы это условие вновь выполнялось, применим следующую процедуру перекраски вершин. Двигаясь снизу вверх по пути, пройденному при поиске, изменяем цвета вершин по правилам, показанным на рис. 20, *a*. Если эти преобразования больше неприменимы, то в случае необходимости один раз выполняется одно из преобразований, представленных на рис. 20, *b*, *c*, *d*.

Исключение элемента производится аналогичным образом. При удалении элемента сначала ищется внешняя вершина, содержащая этот элемент. Затем вершина-родитель этой вершины заменяется братом удаляемой вершины. Это может привести к нарушению условия (2) для черных вершин, а именно может появиться так называемая *короткая* вершина, определяемая тем, что все пути, проходящие через нее к внешним вершинам, содержат на одну черную вершину меньше, чем пути, проходящие через ее брата. Чтобы условие (2) вновь выполнялось, при-

меним следующую процедуру перекраски вершин. Двигаясь снизу вверх по пути, пройденному при поиске, повторно применяем (до тех пор, пока это возможно) преобразование, показанное на рис. 21, *a*. После этого, если требуется, применяем один раз преобразование, показанное на рис. 21, *b*, а затем,

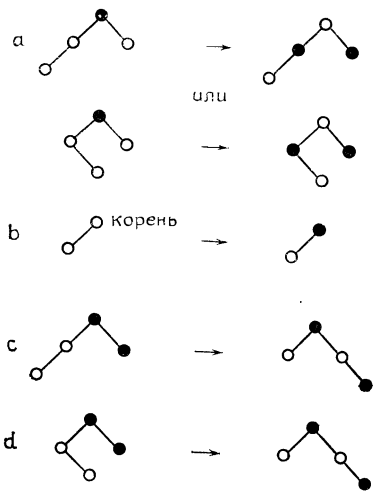


Рис. 20. Преобразования, применяемые для балансировки красно-черного дерева после операции вставки вершины. Симметричные случаи опущены. Все неизвестные сыновья красных вершин имеют черный цвет. В случаях (*c*) и (*d*) самая нижняя из показанных черная вершина может быть внешней.

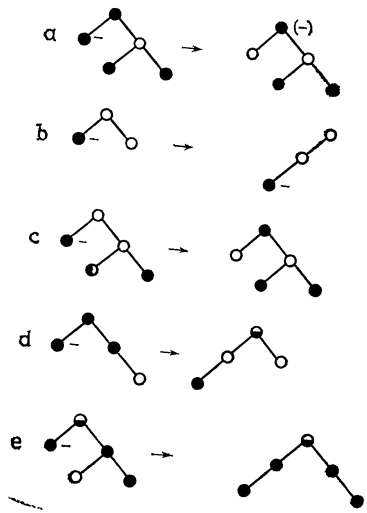


Рис. 21. Преобразования, применяемые для балансировки красно-черного дерева после операции удаления вершины. В случае (*d*) две вершины, закрашенные наполовину, имеют одинаковый цвет. То же самое имеет место и в случае (*e*). Знаком минус отмечены короткие вершины. В случае (*a*) верхняя вершина после применения преобразования является короткой, если только она не является корнем.

возможно, один раз одно из преобразований, показанных на рис. 21, *a*, *c*, *d* или *e*.

Для выполнения операций вставки и удаления, выполняемых над красно-черным деревом, сдержащим n элементов, в худшем случае требуется время $O(\log n)$. Но время, затрачиваемое непосредственно на вставку и удаление элемента, без учета времени на поиск места (вершины), где производится операция вставки или удаления, равно $O(1)$. (Это переформулировка результата, полученного Хадлестоном и Мелхорном [16] и Майером и Салветером [19] для a, b-деревьев.) Для доказа-

тельства этого нам потребуется ряд понятий. *Потенциал* красно-черного дерева определяется как число черных вершин, имеющих двух черных сыновей, плюс удвоенное число черных вершин с двумя красными сыновьями. *Фактическое время* операции вставки или удаления определим как увеличенное на единицу число примененных локальных преобразований, а *приведенное время* — как фактическое время плюс увеличение потенциала, обусловленное операцией. С учетом этих определений можно сказать, что если начать с пустого дерева, то полное фактическое время для последовательности операций вставки и удаления не превышает сумму приведенных времен для каждой из операций, так как потенциал пустого дерева равен нулю и потенциал дерева не может быть отрицательным. Кроме того, приведенное время операции вставки или удаления равно $O(1)$, так как любое из преобразований, представленных на рис. 20 и 21, увеличивает потенциал на $O(1)$, а каждое из преобразований, показанных на рис. 20, *а* и 21, *а*, уменьшает потенциал по крайней мере на единицу.

В обычном двоичном дереве поиска каждая вершина содержит указатели на двух своих сыновей. Мы преобразуем такое дерево в *неоднородное дерево поиска с указателями*, заменив указатели у ряда вершин. У всех вершин левого пути [1]) указатель на левого сына будет теперь указывать на родителя, а у всех вершин правого пути указатель на правого сына будет указывать на родителя.

Доступ к дереву осуществляется с помощью двух указателей, указывающих на самую левую и самую правую внешние вершины дерева (рис. 22).

В неоднородном дереве поиска, содержащем n элементов, элемент, отстоящий на d позиций от одного из концов, может быть найден за время $O(1 + \log(\min\{d, n - d\} + 1))$ в результате выполняемых параллельно просмотра снизу вверх левого и правого путей до тех пор, пока не будет найдено одно или два поддерева, гарантированно содержащие требуемый элемент, с последующим поиском сверху вниз в найденных поддеревьях. Кроме того, за приведенное время $O(1 + \log(\min\{d, n - d\} + 1)$ можно вставить или удалить элемент, отстоящий на d позиций от одного из концов. Поиск элемента можно также осуществлять по занимаемой им позиции или по вторичному порядку. Чтобы приспособить дерево для поиска по позиции, в каждой

[1]) В двоичном дереве *левый путь* определяется как путь, ведущий из корня в некоторую внешнюю вершину, при этом на каждом уровне путь идет по ссылке на левого сына. Правый путь определяется симметричным образом.

внутренней вершине дерева запоминается число достижимых из нее внешних вершин (т. е. внешних вершин, в которые можно попасть из данной вершины, пройдя некоторый путь по указателям). Чтобы приспособить дерево для поиска по вторичному порядку, предположим, что каждый элемент имеет некоторое связанное с ним вторичное значение. В каждой внутренней вершине дерева запоминается минимальное и максимальное вторичные значения элементов, достижимых из этой вершины по

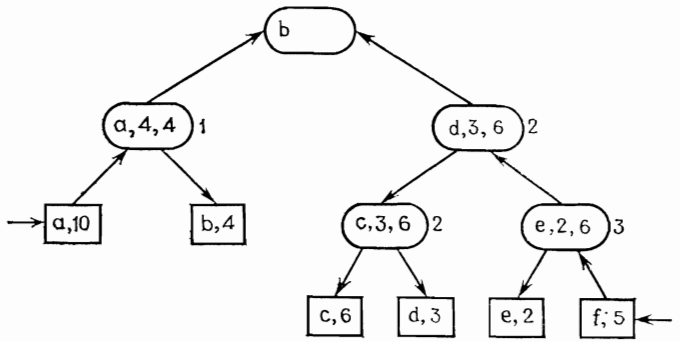


Рис. 22. Неоднородное красно-черное дерево поиска с указателями. Цвета вершин на рисунке не показаны. Элементами являются буквы от *a* до *f*. Числа, указанные во внешних вершинах, представляют вторичные значения. Числа, указанные во внутренних вершинах, представляют минимальное и максимальное вторичные значения вершин, достижимых из соответствующей вершины. Числа, указанные рядом с вершинами, представляют число достижимых внешних вершин.

путям указателей (рис. 22). Просматривая снизу вверх левый и правый пути дерева, а затем сверху соответствующие поддеревья (или одно поддерево) за время $O(1+\log(\min\{d, n-d\}+1))$, можно выполнить следующие операции поиска:

(1) Найти в дереве элемент, находящийся в позиции с номером d;

(2) Найти самый левый (или самый правый) элемент, вторичное значение которого не меньше (не больше) заданного значения, если найденный элемент находится в позиции d.

Вспомогательная информация о позиции и вторичном значении должна соответствующим образом изменяться при выполнении операций вставки или удаления элементов. Эти изменения можно сделать, просматривая в обратном порядке путь поиска, т. е. двигаясь снизу вверх внутри дерева и сверху вниз по его левому или правому пути. Приведенное время, необходимое для вставки или удаления элемента с номером d, включая время поиска, составляет $O(1+\log(\min\{d, n-d\}+1))$.

Теперь нам хочется расширить диапазон операций, связанных с изменением дерева, и включить в него операции конкатенации и расщепления деревьев. Мы рассмотрим лишь, как эти операции изменяют структуру дерева; довольно легко можно проверить, что указатели, ключи и вспомогательная информация о позиции и вторичном значении могут быть изменены надлежащим образом за время, удовлетворяющее приведенным выше оценкам. Определим *ранг* вершины красно-черного дерева как число черных внутренних вершин на любом пути, идущем из этой вершины в некоторую внешнюю вершину. Ранг внешней вершины равен нулю. Ранг вершины можно вычислить за время, пропорциональное рангу вершины, двигаясь вниз в соответствующем поддереве.

Операцию конкатенации деревьев описать проще, чем расщепление. Предположим, что нужно объединить два дерева T_1 и T_2 в одно дерево. При этом предполагается, что все элементы в дереве T_1 меньше любого элемента в дереве T_2. Пусть x_1 и x_2 — корневые вершины деревьев T_1 и T_2 соответственно, r_1 и r_2 — ранги этих вершин. Будем считать, что $r_1 \leqslant r_2$. (Случай $r_2 \leqslant r_1$ рассматривается аналогичным образом.) Чтобы выполнить конкатенацию T_1 и T_2, будем просматривать снизу вверх левый путь дерева T_2 до тех пор, пока не будет найдена вершина (которую обозначим y), ранг которой равен r_1. Заменим вершину y в T_2 новой красной вершиной, левым сыном которой является вершина x_1, а правым сыном — вершина y. При нарушении условия (3) для красных вершин коррекция цветов вершин производится так же, как это делалось при вставке вершины. Приведенное время выполнения операций конкатенации равно $O(1 + \min\{r_1, r_2\})$. Если изменить определение потенциала дерева, определив его как сумму ранга корня дерева и числа черных вершин, не имеющих черных сыновей, то приведенное время операции конкатенации составит $O(1)$. Приведенное время операций вставки или удаления элемента в позиции d при общем числе элементов n останется по-прежнему равным $O(1 + \log(\min\{d, n - d\} + 1))$.

Пусть нам нужно расщепить дерево T, содержащее n элементов, разделив его на два дерева: T_1, содержащее d первых элементов, и T_2, содержащее $n - d$ последних элементов. Прежде всего найдем элемент с номером d. Затем будем подниматься вверх по пути поиска до тех пор, пока не достигнем некоторой вершины x, лежащей либо на левом, либо на правом пути. При продвижении по пути поиска будем удалять все лежащие на нем вершины, за исключением внешней вершины, содержащей элемент с номером d. Предположим, что вершина x принадлежит левому пути дерева. Произведем конкатенацию деревьев,

лежащих слева от пути поиска, включая дерево, состоящее из единственной вершины, содержащей элемент с номером d. Конкатенация деревьев производится в порядке справа налево. Дерево, получившееся в результате конкатенации, является искомым деревом T_1. Выполним конкатенацию деревьев, лежащих справа от пути поиска, корни которых являются потомками вершины x. Результат конкатенации обозначим T_2'. Если вершина x не имеет родителя (предка), то дерево T_2' является искомым деревом T_2. В противном случае остается еще другое дерево T_2'', содержащее вершину y, являющуюся родителем вершины x. В дереве T_2'' недостает вершины, так вершина x была удалена. Мы заменяем вершину y ее правым сыном, и если при этом появляется короткая вершина, нарушающая условие (2), то выполняется процедура перекраски вершин (так же, как это делалось при удалении вершин). После этого, выполнив конкатенацию деревьев T_2' и T_2'', получаем необходимое дерево T_2. Тщательный анализ (см., например, [20]) показывает, что приведенное время операции расщепления дерева составляет $O(1+\log(\min\{d, n-d\}+1))$, как при старом, так и при новом определении потенциала дерева.

Неоднородные деревья поиска с указателями используются в алгоритме вычисления пар видимости для представления частей границ областей. Термин «неоднородный» указывает на тот факт, что указатели структуры обеспечивают особо эффективный доступ к некоторым частям структуры, а именно к двум ее концам. В противоположность этому *однородные* деревья поиска с указателями обеспечивают быстрый доступ в окрестность *любого* элемента. Однородное дерево поиска с указателями получается из красно-черного дерева путем добавления новых указателей. А именно, каждая вершина содержит указатели на двух своих сыновей и на родительскую вершину. Кроме того, каждая черная вершина имеет указатели на своего *левого* и *правого соседей* — черные вершины, имеющие тот же самый ранг, одна из которых предшествует данной вершине, а другая следует за ней при симметричном упорядочении (рис. 23). Эти дополнительные указатели называются *ссылками по уровню*.

Ссылки по уровню обеспечивают поиск заданного элемента за время $O(1+\log\{\min(d, n-d\}+1))$ при начале поиска с произвольного элемента. В этом случае d — это число элементов между искомым элементом и элементом, с которого начинается поиск. Поиск ведется вверх по указателям на родительские вершины и по ссылкам по уровню до тех пор, пока не будет найдено одно или два поддерева, содержащее требуемый

элемент. Затем поиск продолжается сверху вниз обычным образом. Одновременно с поиском из заданной начальной вершины выполняется поиск еще из двух вершин — из вершин, содержащих первый и последний элемент в списке. Все три процесса поиска завершаются при первом обнаружении искомого элемента (или как только становится известным, что он не содержится в дереве).

Однородные деревья поиска с указателями обеспечивают быстрый поиск лишь в случае, когда поиск ведется на основе

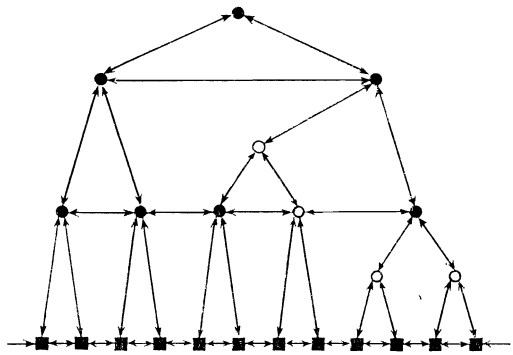


Рис. 23. Организация указателей в однородном красно-черном дереве поиска с указателями.

полного порядка на множестве элементов и не дает никаких преимуществ при поиске элемента по номеру позиции или при поиске на основе порядка, заданного на множестве вторичных значений. С другой стороны, дополнительное время, необходимое для изменения ссылок по уровню при выполнении операций над деревом, составляет лишь $O(1)$ на одно локальное преобразование дерева (типа показанных на рис. 19 и 20), и оценки приведенного времени для операций вставки, удаления, конкатенации и расщепления одни и те же для однородных и неоднородных деревьев. Более того, однородные деревья позволяют выполнить более «масштабную» операцию расщепления, так называемое *тройное расщепление*. Суть этой операции состоит в следующем. В дереве T заданы два элемента x и y. Требуется удалить из T подсписок элементов, начиная с x и кончая y (включительно), получив в результате два дерева. Первое дерево представляет удаленный подсписок, а второе — остальные элементы. Приведенное время выполнения операции тройного расщепления равно $O(1 + \log \min\{d, n - d\} + 1))$, где d — число элементов в удаляемом подсписке. Более подробное обсуждение

вопроса о реализации операции тройного расщепления за указанное время можно найти в работе [14].

Благодарности. Мы благодарны Бренде Бекер за ее многочисленные замечания к предварительному варианту этой статьи и Бернарду Чазелле, с большой тщательностью прочитавшему окончательный вариант статьи.

ЛИТЕРАТУРА

[1] B. K. Bhattacharya and G. T. Toussaint, A linear algorithm for delerming the translation separability of two simple polygons, Technical report SOCS 861, School of Computer Science, McGill University, Montreal, Canada, 1986.
[2] M. R. Brown and R. E. Tarjan, Desing and analysis of a data structure for representing sorted lists, SIAM Journal on Computing, 9 (1980), pp. 594—614.
[3] B. Chazelle, A theorem on polygon cutting with applications, Proc. 23rd Annual Symp. on Found. of Comput. Sci., 1982, pp. 339—349.
[4] B. Chazelle, Intersecting is easier than sorting, Proc. Sixteenth Annual ACM Symp. on Theory of Comput., 1984, pp. 125—134.
[5] B. Chazelle and J. Inperpi, Triangulation and shape complexity, ACM Trans. on Graphics, 3 (1984), pp. 135—152.
[6] W. P. Chin and S. Ntafos, Optimum watchman routes, Proc. Second Annual Symp. on Computational Geometry, 1986, pp. 24—33.
[7] D. P. Dobkin, D. L. Souvaine and C. J. Van Wyk, Decomposition and intersection of simple splinegons, Algorithmica, to appear.
[8] H. Edelsbrunner, L. J. Guibas and J. Stolfi, Optimal point location in a monotone subdivision, SIAM Journal on Computing, 15 (1986), pp. 317—340.
[9] A. Fournier and D. Y. Montuno, Triangulating simple polygons and equivalent problems, ACM Trans. on Graphics, 3 (1984), pp. 153—174.
[10] M. R. Garey, D. S. Johnson, F. P. Preparata and R. E. Tarjan, Triangulating a simple polygon, Inform. Process. Lett., 7 (1978), pp. 175—180.
[11] L. Guibas, J. Hershberger, D. Leven, M. Sharir and R. E. Tarjan, Linear time algorithms for visibility and shortest path problems inside triangulated simple polygons, Algorithmica, to appear.
[12] L. J. Guibas, E. M. McCreight, M. F. Plass and J. R. Roberts, A new representation for linear lists, Proc. Ninth Annual ACM Symp. on Theory of Comput., 1977, pp. 49—60.
[13] S. Hertel and K. Mehlhorn, Fast triangulation of a simple polygon, Proc. Conf. Found. of Comput. Theory, New York, Springer-Verlag, Berlin, 1983, pp. 207—218.
[14] K. Hoffman, K. Mehlhorn, P. Rosenstiehl and R. Tarjan, Sorting Jordan sequences in linear time using level-linked search trees, Inform. and Control, 68 (1986), pp. 170—184.
[15] S. Huddleston, An efficient scheme for fast local updates in linear lists, Dept. on Information and Computer Science, University of California, Irvine, CA, 1981.
[16] S. Huddleston and K. Mehlhorn, A new data structure for representing sorted lists, Acta Inform., 17 (1982), pp. 157—184.
[17] J. M. Keil, Decomposing a polygon into simpler components, SIAM Journal on Computing, 14 (1985), pp. 799—817.

[18] S. R. Kosaraju, Localized search in sorted lists, Proc. Thirteenth Annual ACM Symp. on Theory of Comput., 1981, pp. 62—69.
[19] D. Maier and C. Salveter, Hysterical B-trees, Inform. Process. Lett., 12 (1981), pp. 199—202.
[20] K. Mehlhorn, Data Structures and Efficient Algorithms, Volute 1: Sorting and Searching, Springer-Verlag, Berlin, 1984.
[21] J. R. Munkres, Topology: A First Course, Prentice-Hall, Englewood Cliffs, NJ, 1975.
[22] A. A. Schäffer and C. J. Van Wyk, Convex hulls of piecewise smooth Jordan curves, J. Algorithms, 8 (1987), pp. 66—94.
[23] D. D. Sleator and R. E. Tarjan, Self-adjusting binary search trees, J. Assoc. Comput. Mach., 32 (1985), pp. 652—686.
[24] D. L. Souvaine, Computational Geometry in a Curved World, Ph. D. dissertation, Department of Computer Science, Princeton University, 1986.
[25] R. E. Tarjan, Data Structures and Network Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983.
[26] R. E. Tarjan, Amortized computational complexity, SIAM J. Algebraic and Discrete Methods, 6 (1985), pp. 306—318.
[27] R. E. Tarjan and C. J. Van Wyk, A lineal-time algorithm for triangulating simplev polygons, Proc. Eighteenth Annual ACM Symp. on Theory of Comput., 1986, pp. 380—388.
[28] A. K. Tsakalidis, AVL-trees for localized search, Automata, Languages, and Programming, 11th Colloquium, J. Paredaens, ed., Lecture Notes in Computer Science 172, Springer-Verlag, Berlin, 1984, pp. 473—485.
[29] A. K. Tsakalidis, An optimal implementation for localized search, A84/06 Fachbereich Angewandte Mathematik und Informatik, Universität des Saarlandes, Saarbrücken, West Germany, 1984.
[30] C. J. Van Wyk, Clipping to the boundary of a circular — arc polygon, Computer Vision, Graphics, and Image Processing, 25 (1984), pp. 383—392.
[31] T. C. Woo and S. Y. Shin, A linear time algorithm for triangulating a point — visible polygon, ACM Trans. on Graphics, 4 (1985), pp. 60—69.
[32*] Д. Ли, Ф. Препарата, Вычислительная геометрия. Обзор. — Кибернетический сборник, вып. 24. — М.: Мир. 1987, с. 5—96.
[33*] Ф. Препарата, М. Шеймос, Вычислительная геометрия. Введение. — М.: Мир, 1989.

ДОПОЛНЕНИЕ ПЕРЕВОДЧИКА

Хотя в название статьи авторы вынесли тему триангуляции простого многоугольника, в самой статье они лишь ограничились ссылкой на работы, содержащие описание сведения задачи триангуляции к задаче вычисления пар видимости. Несмотря на относительную простоту этого сведения, здесь мы ради полноты изложения дадим набросок алгоритма, позволяющего за линейное время выполнить триангуляцию простого многоугольника, основываясь на информации, полученной в результате выполнения алгоритма вычисления пар видимости.

Алгоритм использует тот факт, что триангуляцию простого монотонного многоугольника можно выполнить за линейное время [32, 33]. Простой многоугольник называется монотонным, если существует такая прямая L, что граница P может быть разбита

на две ломаные таким образом, что для каждой из них порядок проекций вершин ломаной на L (при проектировании в направлении, перпендикулярном к прямой L) совпадает с порядком вершин вдоль ломаной. Идея алгоритма состоит в разбиении исходного многоугольника на совокупность монотонных подмногоугольников с последующей триангуляцией последних. Разбиение производится путем проведения ряда диагоналей в исходном многоугольнике.

Результатом применения алгоритма вычисления пар видимости, представленного в статье, является разбиение многоугольника на некоторое множество трапеций, верхнее и нижнее основания которых параллельны оси Ox. Получившиеся трапеции

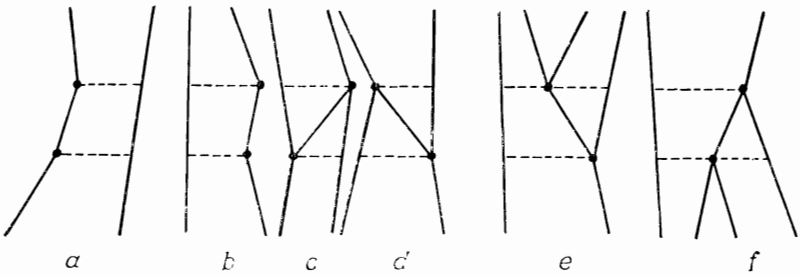


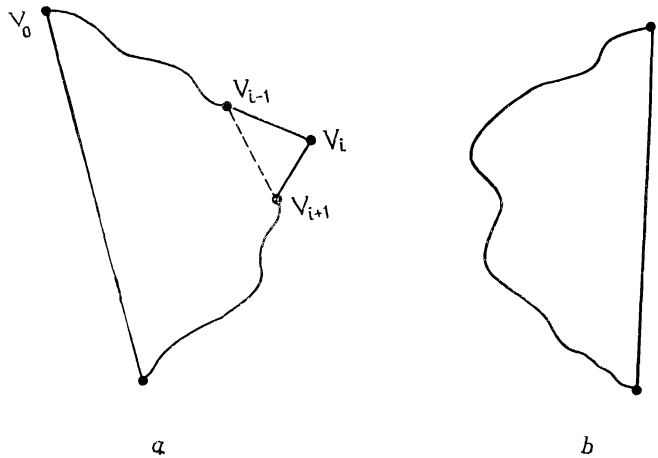
Рис. 1.

делятся на два класса (рис. 1). К первому классу относятся трапеции, у которых обе определяющие их вершины являются смежными в исходном многоугольнике (рис. 1*a*, 1*b*). Все остальные трапеции относятся ко второму классу (рис. 1*c*—1*f*). В каждой трапеции второго класса проведем отрезок, соединяющий вершины, определяющие эту трапецию. Каждый такой отрезок является диагональю исходного многоугольника P. Совокупность этих диагоналей разбивает P на подмногоугольники.

Довольно легко можно показать, что все полученные таким образом подмногоугольники являются монотонными относительно вертикальной прямой. Более того, в каждом подмногоугольнике одна из монотонных ломаных, на которые в силу монотонности подмногоугольника разбивается его граница, состоит из единственного отрезка (рис. 2).

Рассмотрим триангуляцию подмногоугольника типа показанного на рис. 2*a*. Триангуляция осуществляется в процессе обхода с возвратом границы подмногоугольника в направлении по часовой стрелке. Обход начинается с вершины, следующей за вершиной с максимальной y-координатой. Будем продвигаться по границе до тех пор, пока не достигнем вершины V_i внутренний

угол которой меньше 180°. Такая вершина, как легко можно убедиться, всегда существует. Соединим диагональю вершины V_{i-1} и V_{i+1}, получив очередной треугольник триангуляции $V_{i-1}V_iV_{i+1}$. Исключим вершину V_i из границы и продолжим движение по скорректированной границе с вершины V_{i-1} (возврат на один шаг), если $V_{i-1} \neq V_0$, и с вершины V_{i+1} в противном



a *b*

Рис. 2.

случае. Эта процедура продолжается до тех пор, пока в границе не останутся лишь две вершины.

Легко видеть, что при триангуляции подмногоугольника число возвратов равно $O(n)$, так как возврат осуществляется лишь при удалении вершины, а число шагов в прямом направлении также составляет $O(n)$. Таким образом, триангуляция подмногоугольника выполняется за линейное время.

Триангуляция подмногоугольника, показанного на рис. 2*b*, производится аналогично с той лишь разницей, что обход начинается с вершины, следующей за вершиной с минимальной y-координатой.

Реализация описанной выше схемы позволяет получить алгоритм, выполняющий триангуляцию простого многоугольника с использованием информации, полученной в результате применения алгоритма вычисления пар видимости, за время $O(n)$. Это с учетом оценки сложности алгоритма вычисления пар видимости и дает суммарную оценку $O(n \log\log n)$ сложности алгоритма триангуляции простого многоугольника.

М. М. Комаров

Отрицание малоэффективно для булевых слой-функций[1)]

Л. Дж. Вальянт[2)]

Резюме. Показано, что для любой слой-функции от n переменных сложность монотонной схемы не превосходит суммы удвоенной сложности схемы в полном базисе и аддитивного члена порядка $O(n(\log n)^2)$.

1. ВВЕДЕНИЕ

Вопрос о том, какую экономию дает полный базис $\{\&, \vee, \neg\}$ по сравнению с монотонным базисом $\{\&, \vee\}$, долгое время оставался нерешенной проблемой теории сложности булевых схем. В 1981 г. С. Берковиц [3] сделал замечательное наблюдение, показывающее, что при подходящей формулировке эта проблема имеет простое решение. Для произвольной булевой функции $f(x_1, \ldots, x_n)$ он ввел k-ю *слой-функцию* для функции f как функцию, равную:

1) $f(x_1, \ldots, x_n)$, если ровно k переменных равны 1,
2) 0, если меньше чем k переменных равны 1,
3) 1, если больше чем k переменных равны 1.

Ясно, что для многих целей функцию f можно представлять ее $n+1$ слой-функциями. Пусть $g(x_1, \ldots, x_n)$ — любая слой-функция (т. е. она является таковой для некоторого k) и $X(x_1, \ldots, x_n)$ — схема в базисе $\{\&, \vee, \neg\}$, вычисляющая ее. Известно, что схема X может быть преобразована в монотонную схему $X^*(x_1, \ldots, x_n, z_1, \ldots, z_n)$, содержащую не более чем в 2 раза больше элементов [6], которая будет вычислять g, если вместо каждого z_j подставить $\bar{x}_j$. Известно также [1], что существует монотонная схема сложности $O(n \log n)$ для булевой сортировки и, следовательно, для вычисления любой пороговой функции

[1)] Valiant L. G. Negation is powerless for Boolean slice functions, SIAM Journal on Computing, v. 15, N 2 (1986), 531—535.

[2)] Aiken Computation Laboratory, Harvard University, Cambridge, Massachusetts 02138.

$\mathrm{Th}_k(x_1, \ldots, x_n)$. Здесь Th_k — функция, которая равна 1 тогда и только тогда, когда по крайней мере k переменных равны 1. Вот это неожиданное открытие, справедливость которого читатель может легко проверить:

Лемма Берковица. *Если g — k-я слой-функция, то схема $X^*(x_1, \ldots, x_n, z_1, \ldots, z_n)$ будет вычислять $g(x_1, \ldots, x_n)$, если вместо z_j подставить $\mathrm{Th}_k(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$, $1 \leqslant j \leqslant n$.*

Пусть $L_{\&, \vee}$ и $L_{\&, \vee, \neg}$ — величины сложности минимальных схем в соответствующих базисах.

Следствие 1. *Если g — слой-функция, то*

$$L_{\&, \vee}(g) \leqslant 2L_{\&, \vee, \neg}(g) + O(n^2 \log n).$$

Цель данной работы — улучшить аддитивный член до $O(n(\log n)^2)$, показав, каким образом функции, выполняющие роль $\bar{x}_1, \ldots, \bar{x}_n$, могут быть одновременно вычислены монотонной схемой указанной сложности.

Определение. *Монотонным (k, n)-инвертором* называется монотонная схема с входами $x_1, \ldots, x_n$ и выходами $y_1, \ldots, y_n$, такая что:

a) Если ровно k входов равны 1, то $y_j = \bar{x}_j$, $1 \leqslant j \leqslant n$.
b) Если больше чем k входов равны 1, то $y_j = 1$, $1 \leqslant j \leqslant n$.
c) Если меньше чем k входов равны 1, то $y_j = 0$, $1 \leqslant j \leqslant n$.

Теорема. *Существует константа α, такая что для любых n и k существует монотонный (k, n)-инвертор, содержащий не более чем $\alpha n(\log_2 n)^2$ элементов.*

Главный недостаток следствия 1 в том виде, в каком оно сформулировано, состоит в том, что оно слишком слабо по отношению к известным нижним оценкам монотонной сложности, так как эти оценки имеют порядок $O(n^2)$[1]), а такой рост подавляется аддитивным членом $\theta(n^2 \log n)$. В противоположность этому наша теорема может быть успешно применена к таким проблемам, как умножение матриц и булева свертка, которые рассматривались в работах [4], [7]—[11], [13], [15]. Например, в случае умножения булевых матриц размера $m \times m$ известно, что любая монотонная схема содержит по крайней мере

[1]) В 1985 г. А. А. Разборовым в работе «Нижние оценки монотонной сложности некоторых булевых функций» (ДАН СССР, 1985, т. 281, № 2, с. 798—801) и А. Е. Андреевым в работе «Об одном методе получения нижних оценок сложности индивидуальных монотонных функций» (ДАН СССР, 1985, т. 282, № 5, с. 1033—1037) были получены значительно более высокие (выше полиномиальных) нижние оценки монотонной сложности. Впоследствии и эти оценки улучшались. — *Прим. ред.*

m^3 элементов, в то время как в полном базисе достаточно $O(m^{2,81})$ или даже $O(m^{2,5})$ элементов [5], [12]. Следствием нашей теоремы является тот факт, что, например, (m^2)-я слой-функция, соответствующая умножению матриц размера $m \times m$, может быть вычислена монотонной схемой из $O(m^{2,81})$ или даже $O(m^{2,5})$ элементов. На интуитивном уровне это сводится к тому, что известные методы доказательства нижних оценок для монотонных схем в равной мере выражают и особенности конкретной постановки задачи, подлежащей вычислению, и свойства модели вычислений, а также самой этой задачи.

Мы сформулируем наше усиленное следствие для систем слой-функций, поскольку это единственный случай, для которого в настоящее время получены нелинейные нижние оценки [1]).

Определение. *Системой слой-функций* называется система булевых функций $f_1(x_1, \ldots, x_n), \ldots, f_r(x_1, \ldots, x_n)$, каждая из которых является k-й слой-функцией для одного и того же k.

Следствие 2. *Для любой системы F слой-функций от переменных $x_1, \ldots, x_n$*

$$L_{\&, \vee}(F) \leqslant 2L_{\&, \vee, \neg}(F) + O(n(\log n)^2).$$

Дальнейшие результаты о слой-функциях получены недавно Вегенером [14].

2. КОНСТРУКЦИЯ

Конструкция основана на монотонной подсхеме, которая упорядочивает совокупность двух предварительно упорядоченных наборов длины m и содержит $O(m \log m)$ элементов. Здесь достаточно алгоритма раздельного упорядочения чисел с нечетными и четными номерами, принадлежащего Бэтчеру [2].

Первое упрощение. Заметим, что достаточно построить инвертор, работающий правильно, когда на входах ровно k единиц. По выходам $\{y_i\}$ такой схемы правильные выходы $\{y_i^*\}$ могут быть вычислены так:

$$y_i^* = (y_i \,\&\, \mathrm{Th}_k(x_1, \ldots, x_n)) \vee \mathrm{Th}_{k+1}(x_1, \ldots, x_n)$$

для каждого i. Ясно, что все выходы y^* будут равны 0, если $\mathrm{Th}_k = 0$, и будут равны 1, если $\mathrm{Th}_{k+1} = 1$. Это добавит только $O(n(\log n)^2)$ элементов к общей сложности, если использовать алгоритм Бэтчера, или $O(n \log n)$, если использовать алгоритм из [1].

[1]) В работах А. А. Разборова и А. Е. Андреева, упомянутых выше, каждая оценка получена для одной функции. — *Прим. ред.*

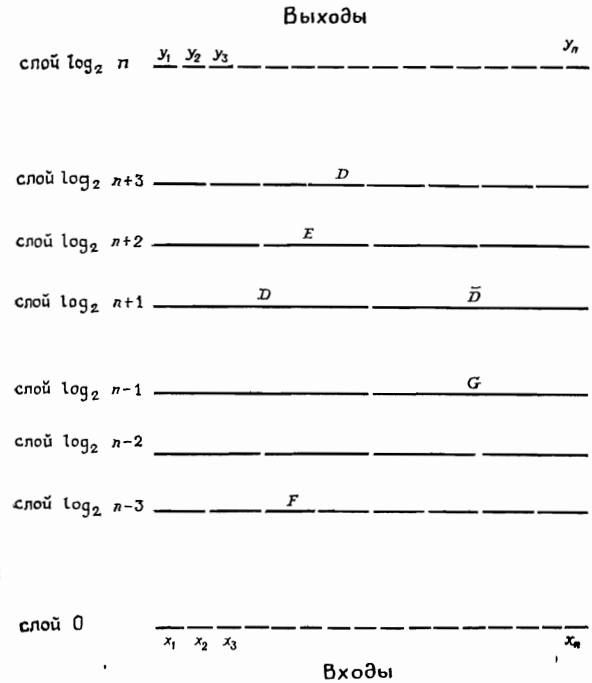


Рис. 1. Схематический вид конструкции.

Следующее упрощение. Рассмотрим только случай $k \leqslant n/2$. Чтобы получить (k, n)-инвертор для $k > n/2$, достаточно построить $(k, 2n)$-инвертор и на половину его входов подать 0. По аналогичной причине мы можем также считать n степенью 2.

Мы построим схему глубины $\theta((\log n)^2)$. Выделим элементы на $2\log_2 n$ уровнях и назовем эти уровни *ярусами*. Ярусы пронумеруем числами из множества $\{i \mid 0 \leqslant i < \log_2 n\} \cup \{2\log_2 n - i \mid 0 \leqslant i < \log_2 n\}$. Ярусы с номерами i и $2\log_2 n - i$ состоят из $n/2^i$ *блоков*, содержащих по 2^i элементов. Каждый блок A *охватывает* множество $\mathrm{span}(A) \subseteq \{x_1, \ldots, x_n\}$ с индексами переменных $r2^i + 1, \ldots, r2^i + 2^i$ для некоторого $r (0 \leqslant r < n/2^i)$. Ярус с номером 0 состоит из n одиночных блоков-входов $x_1, \ldots, x_n$, ярус с номером $2\log_2 n$ также состоит из одиночных блоков-выходов $y_1, \ldots, y_n$, причем $\mathrm{span}(\{y_i\}) = \{x_j\}$. *Дополнение* блока A, обозначаемое $\bar{A}$, есть объединение всех блоков, отличных от A, лежащих в том же ярусе, что и A.

Согласно сказанному выше, блоки состоят из элементов или из входов. Но, когда входы принимают булевы значения, выходы

элементов тоже принимают булевы значения. Для удобства отождествим блоки с наборами булевых значений, которые они порождают. Каждый такой блок будет упорядочен в порядке возрастания, потому что в конструкции не будет других преобразований, кроме слияния уже упорядоченных блоков. Схематический вид конструкции дан на рис. 1.

Будем обозначать блоки через A, B, C, ..., а номера их ярусов — через $N(A)$, $N(B)$, ... Схему можно описать так:

1) Если $N(A)=0$ и $\operatorname{span}(A)=\{x_j\}$, то A — это вход x_j.

2) Если $N(A)=i$ и $1\leqslant i<\log_2 n$, то A — это упорядоченный блок, полученный слиянием блоков B и C, где $N(B)=N(C)=i-1$ и $\operatorname{span}(A)=\operatorname{span}(B)\cup\operatorname{span}(C)$.

3) Если $N(D)=\log_2 n+1$, то D — это блок, полученный приписыванием справа к последним k цифрам блока G $n/2-k$ единиц, где $N(G)=\log_2 n-1$ и $\operatorname{span}(G)\neq\operatorname{span}(D)$. (Заметим, что G и D имеют размер $n/2$ и $\operatorname{span}(G)\cup\operatorname{span}(D)=\{x_1, \ldots, x_n\}$.)

4) Если $N(D)=2\log_2 n-i$ и $0\leqslant i<\log_2 n$, то D — это блок, состоящий из средних 2^i разрядов упорядоченного блока длины 3×2^i, полученного слиянием блоков E и F, где $N(E)=2\log_2 n-i-1$, $N(F)=i$ и $\operatorname{span}(E)=\operatorname{span}(D)\cup\operatorname{span}(F)$.

5) Если $N(D)=2\log_2 n$ и $D=\{y_j\}$, то y_j — это выход, соответствующий входу x_j $(1\leqslant j\leqslant n)$.

3. ДОКАЗАТЕЛЬСТВО КОРРЕКТНОСТИ

Очевидно, что первые $\log_2 n$ ярусов образуют схему упорядочения. В частности, если $N(A)=i$ и $0\leqslant i<\log_2 n$, то A — упорядоченный блок для $\operatorname{span}(A)$. Обозначим через $\#_a(A)$ (соответственно через $\#_a(\operatorname{span}(A))$) число разрядов с цифрой a в A (соответственно в $\operatorname{span}(A)$), где $a=0$ или 1. Главную идею конструкции можно выразить так:

Утверждение [1]. *Если* $N(D)=2\log_2 n-i$ *и* $0\leqslant i<\log_2 n$, *то*

$$\#_0(D)=k-\#_1(\operatorname{span}(\bar{D})).$$

Заметим, что из справедливости утверждения для яруса с номером $2\log_2 n$ следует и истинность теоремы, так как если $D=\{y_j\}$, то $\operatorname{span}(\bar{D})$ — это множество всех входов, отличных от x_j, и, следовательно, $y_j=0$ тогда и только тогда, когда $x_j=1$.

[1]) Особо не оговорено, что рассматривается только случай, когда на входы подается ровно k единиц. — *Прим. перев.*

Доказательство утверждения. Доказательство основано на индукции по i от $\log_2 n - 1$ до $i = 0$.

Основание индукции, т. е. случай $i = \log_2 n - 1$, соответствует ярусу с номером $\log_2 n + 1$. В силу п. 3) описания конструкции и того, что $\#_1(G) \leqslant k$, имеем

$$\#_1(D) = \#_1(G) + (n/2 - k).$$

Поскольку $\#_1(G) = \#_1 \operatorname{span}(\bar{D})$ и $\#_0(D) = n/2 - \#_1(D)$, $\#_0(D) = k - \#_1(\operatorname{span} \bar{D})$.

Шаг индукции. Пусть утверждение верно для некоторого i $(0 < i \leqslant \log_2 n - 1)$. Докажем, что тогда оно верно и для $i - 1$. Рассмотрим упорядоченный блок, полученный в результате слияния E и F (см. п. 4) описания конструкции). По индуктивному предположению

$$\#_0(E) = k - \#_1(\operatorname{span}(\bar{E})).$$

Так как F — упорядоченный блок для $\operatorname{span}(F)$, то

$$\#_0(F) = 2^i - \#_1(\operatorname{span}(F)).$$

Следовательно,

$$\#_0(E \cup F) = 2^i + k - \#_1(\operatorname{span}(\bar{E})) - \#_1(\operatorname{span}(F)) =$$
$$= 2^i + k - \#_1(\operatorname{span}(\bar{D})),$$

так как $\operatorname{span}(\bar{D}) = \operatorname{span}(\bar{E}) \cup \operatorname{span}(F)$. Упорядоченный блок, полученный в результате слияния блоков E и F, состоит из 2^i нулей, следующих за ними еще $k - \#_1(\operatorname{span}(\bar{D}))$ нулей и единиц в оставшихся разрядах. Так как всего на входы схемы подается ровно k единиц, то $\#_1(\operatorname{span}(\bar{D})) \geqslant k - 2^i$ и, следовательно, $k - \#_1(D) \leqslant 2^i$. Поэтому в средних 2^i разрядах блока, полученного в результате слияния блоков E и F, ровно $k - \#_1(\operatorname{span}(\bar{D}))$ нулей, что и требовалось доказать. ■

ЛИТЕРАТУРА

[1] M. Ajtai, J. Komlós, E. Szemerédi, An $O(n \log n)$ sorting network, Proc. 15th ACM Symposium on Theory of Computing, 1983, pp. 1—9.

[2] K. Batcher, Sorting networks and their applications, AFIPS Spring Joint Computing Conference, 32, 1968, p. 307—314.

[3] S. J. Berkowitz, Personal communication (1981). Also, On some relationships between monotone and non-monotone circuit complexity, manuscript, University of Toronto, Computer Science Department.

[4] N. Blum, An $\Omega(N^{3/4})$ lower bound on the monotone network complexity of N-th degree convolution, Proc. 22nd ACM Symposium on Foundations of Computer Science, 1981, pp. 101—108.

[5] D. Coppersmith and S. Winograd, On the asymptotic complexity of matrix multiplication, Proc. 22nd ACM Symposium on Foundations of Computer Science, 1981, pp. 82—90.
[6] M. J. Fischer, The complexity of negation-limited networks, Lecture Notes in Computer Science 33, Springer-Verlag, Berlin, 1974, pp. 71—82.
[7] E. A. Lamagna and J. E. Savage, Combinational complexity of some monotone functions, Proc. 15th IEEE Symposium on Switching and Automata Theory, 1974, pp. 140—144.
[8] K. Mehlhorn and Z. Galil, Monotone switching circuits and Boolean matrix product, Computing, 16 (1976), pp. 99—111.
[9] M. S. Paterson, Complexity of monotone networks for Boolean matrix product, Theoret. Comput. Sci, 1 (1975), pp. 13—20. [Имеется перевод: М. С. Патерсон. Сложность монотонных схем для булева умножения матриц. Кибернетический сборник, № 15 (1978), с. 28—38.]
[10] N. J. Pippenger and L. G. Valiant, Shifting graphs and their applications, J. Assoc. Comput. Mach., (1976), pp. 423—433.
[11] V. R. Pratt, The power of negative thinking in multiplying Boolean matrices, this Journal, 4 (1975), pp. 326—330.
[12] V. Strassen, Gaussian elimination is not optimal, Numer. Math., 13 (1969), pp. 354—356. [Имеется перевод: В. Штрассен, Исключение по Гауссу не оптимально, Математика, сб. переводов, 1970, т. 14, № 3, с. 127—128.]
[13] I. Wegener, Boolean functions whose monotone complexity is size $n^2/\log n$, Theoret. Comput. Sci., 21 (1982), pp. 213—224. [Имеется перевод: И. Вегенер. Булевы функции, чья монотонная сложность имеет величину порядка $n^2/\log n$, Кибернетический сборник, № 21. — М.: Мир, 1984, с. 69—85.]
[14] Wegener, On the complexity of slice functions, Lecture Notes in Computer Science 176, Springer-Verlag, Berlin, 1984, pp. 553—561. (Also Theoret. Comput. Sci., to appear.)
[15] J. Weiss, An $\Omega(n^{3/2})$ lower bound on the monotone complexity of Boolean convolution, Inform. and Control, to appear.
[16*] P. E. Dunne, The complexity of central slice functions, Theoretical Computer Science, v. 44, No. 3 (1986), 247—257.
[17*] I. Wegener, On the complexity of slice functions, Theoretical Computer Science, v. 38, No. 1 (1985), 55—68.
[18*] I. Wegener, More on the complexity of slice functions, Theoretical Computer Science, v. 43, No. 2/3 (1986), 201—212.
[19*] Е. А. Окольнишникова. О сведении оценок сложности в полном базисе к оценкам сложности в неполном базисе. — В кн.: Методы дискретного анализа в теории графов и схем. Новосибирск, 1985, вып. 42, с. 80—90.

* Добавлено при переводе.

Характеристика всех оптимальных схем из функциональных элементов для одновременного вычисления AND и NOR[1])

Н. Блюм[2]), *М. Сейсен*[3])

В работе характеризуются оптимальные схемы из функциональных элементов в базисе из всех 16 двуместных булевых операций для одновременного вычисления AND и NOR. Мы покажем, что оптимальные схемы для AND и NOR — это в точности схемы, каждая из которых является объединением оптимальной схемы для AND и оптимальной схемы для NOR, не имеющих общих элементов.

ВВЕДЕНИЕ

Мы охарактеризуем оптимальные схемы из функциональных элементов для одновременного вычисления AND и NOR. Из [3—5] известно, что сложность схем для булевых функций тесно связана с величиной произведения времени работы и объема программы машин Тьюринга, вычисляющих их. Поэтому представляет интерес получение нижних оценок сложности схем для конкретных булевых функций. Обычно трудно вычислить точное значение сложности схемы для булевой функции. Нетривиальный пример см. в [6].

Если несколько булевых функций вычисляются одной схемой, то интересно выяснить, что лучше: вычислять ли все функции независимо или же экономичнее вычислить промежуточные результаты, которые можно использовать для вычисления нескольких функций.

Пауль [2] показал[4]), что существуют две функции F_n, G_n: $\{0, 1\}^n \to \{0, 1\}^n$ равной сложности, зависящие от непересе-

[1]) Blum N., Seysen M. Characterization of all Optimal Networks for a Simultaneous Computation of AND and NOR, Acta Informatica, 21 (1984), 171—181.

[2]) Fachbereich 10, Angewandte Mathematik und Informatik, Universität des Saarlandes, D-6600 Saarbrücken, (Fed. Rep.).

[3]) Fachbereich 12, Mathematik, Johann Wolfgang v. Goethe-Universität, D-6000 Frankfurt a. M., (Fed. Rep.).

[4]) Ранее аналогичный результат был получен в работе Д. Улига «О синтезе самокорректирующихся схем из функциональных элементов с малым числом надежных элементов», Матем. заметки, т. 15, вып. 6, 1974, 937—944. — *Прим. перев.*

кающихся множеств переменных, такие что:

$$L\{F_n, G_n\} \leqslant L\{F_n\}(1+\varepsilon), \quad \text{где} \quad \varepsilon \to 0 \quad \text{при} \quad n \to \infty.$$

С другой стороны, мы покажем, что функции

$$\mathrm{AND}_n = \bigwedge_{i=1}^{n} x_i \quad \text{и} \quad \mathrm{NOR}_n = \bigwedge_{i=1}^{n} \bar{x}_i,$$

зависящие от одного множества переменных, независимы в том смысле, что

$$L\{\mathrm{AND}_n, \mathrm{NOR}_n\} = L\{\mathrm{AND}_n\} + L\{\mathrm{NOR}_n\}.$$

Более того, мы покажем, что в каждой оптимальной схеме, вычисляющей эти две функции одновременно, элементы, используемые для вычисления AND_n, отличны от элементов, используемых для вычисления NOR_n.

1. СИСТЕМА ОБОЗНАЧЕНИЙ

Пусть $B = \{0, 1\}$, $B_n = \{f \mid f\colon B^n \to B\}$, и пусть x_i — это i-я переменная.

Схемой называется ориентированный ациклический граф β, такой что:

(i) Входная степень каждой вершины графа β равна 0 или 2 (вершины с входной степенью 0 называются *входами* схемы). Каждому входу v приписана функция $\mathrm{op}(v) \in \{x_1, x_2, \ldots, x_n\} := X_n$.

(ii) Каждой вершине v с входной степенью 2 приписана операция $\mathrm{op}_v \in B_2$ (подразумевается, что входы вершины v упорядочены).

Каждой вершине v схемы β сопоставим функцию $\mathrm{res}_{\beta, v}$, определяемую следующим образом:

$$\mathrm{res}_{\beta, v} = \begin{cases} x_i, & \text{если } v \text{ — вход схемы с } \mathrm{op}(v) = x_i, \\ \mathrm{op}_v(\mathrm{res}_{\beta, u}, \mathrm{res}_{\beta, w}), & \text{если } u \text{ — первый вход вершины } v \text{ и } w \text{ — второй вход вершины } v. \end{cases}$$

Будем говорить, что система функций $F \subset B_n$ *реализуется* схемой β, если

$$\forall f \in F\colon \exists v \in \beta\colon \mathrm{res}_{\beta, v} = f.$$

Для любой булевой переменной x положим $x^1 = x$ и $x^0 = \bar{x}$. Будем говорить, что функция $f \in B_n$ *зависит* от x_1, если суще-

ствует набор $(a_1, \ldots, a_n) \in B^n$, такой что

$$f(a_1, \ldots, a_i, \ldots, a_n) \neq f(a_1, \ldots, a_{i-1}, \bar{a}_i, a_{i+1}, \ldots, a_n).$$

В противном случае говорят, что f не зависит от x_i.

По определению $f \in B_2$ — *функция* $\wedge$-типа, если

$$\exists a, b, c \in B: f(x, y) = (x^a \wedge y^b)^c;$$

$f \in B_2$ — *функция* $\oplus$-*типа*, если

$$\exists a \in B: f(x, y) = (x \oplus y)^a.$$

16 функций $f \in B_2$ классифицируются следующим образом: 2 константы (0 и 1), 4 функции, зависящие в точности от одной переменной, 8 функций $\wedge$-типа и 2 функции $\oplus$-типа.

Введем обозначения:

$$\mathrm{AND}_n = \bigwedge_{i=1}^{n} x_i, \quad \mathrm{NOR}_n = \bigwedge_{i=1}^{n} \bar{x}_i.$$

Для схемы β положим $L(\beta)$ равным числу всех вершин схемы, не являющихся ее входами. *Сложность* $L(F)$, $F \subseteq B_n$, определим следующим образом:

$$L(F) = \min \{L(\beta) \mid \beta \text{ реализует } F\}.$$

Схема β, реализующая F и удовлетворяющая равенству $L(\beta) = L(F)$, называется *оптимальной* схемой для F.

Вершина u называется *преемником* (*предшественником*) вершины v, если существует путь из v в u (из u в v).

Пусть v — произвольная вершина схемы β. Обозначим множество непосредственных преемников вершины v через $\mathrm{suc}_\beta(v)$, а ее выходную степень через $\mathrm{out}_\beta(v)$. Вершина с выходной степенью $\geqslant 2$, не являющаяся входом схемы, называется *элементом ветвления*. Вершину с выходной степенью 0, не являющуюся входом схемы, назовем *выходом схемы*. Вершины, не являющиеся входами схемы, называются *элементами*.

Для каждой схемы β введем следующее обозначение:

$$x \deg(\beta) = \sum_{x_i \in X_n} \mathrm{out}_\beta(x_i).$$

2. ДОКАЗАТЕЛЬСТВО НИЖНЕЙ ОЦЕНКИ

Следующая лемма представляет собой незначительное обобщение леммы, доказанной Паулем [1].

Лемма 1. *Каждая схема* β *с* S *элементами ветвления и* (*в точности*) A *выходами удовлетворяет неравенству*

$$L(\beta) \geqslant x \deg(\beta) + S - A.$$

Доказательство совпадает с доказательством в работе [1].

Лемма 2. *Для каждой оптимальной схемы* β, *реализующей* AND_n (*соответственно* NOR_n), *выполняются следующие* 3 *условия*:
(1) β — *бинарное дерево*[1]) *с* n *входами и* $n-1$ *элементом*;
(2) *все элементы в* β *являются элементами* $\wedge$*-типа*;
(3) *для каждой вершины* v *схемы* β *существуют множество переменных* M *и константа* $c \in B$, *такие что*:

$$\mathrm{res}_{\beta, v} = \Big(\bigwedge_{x_i \in M} x_i^k\Big)^c,$$

где $k = 1$ (*соответственно* $k = 0$) *при реализации функции* AND_n (*соответственно* NOR_n).

Доказательство. Мы докажем лемму для AND_n. Доказательство для NOR_n аналогично.

(1) следует из следующих фактов:

(i) AND_n зависит от всех n переменных.

(ii) Существует оптимальная схема β для AND_n, такая что $L(\beta) = n - 1$.

(iii) Единственный способ связать n входов при помощи $n-1$ элемента — это бинарное дерево.

Для доказательства (2) и (3) положим, что β — оптимальная схема для AND_n. Допустим, что в β существует элемент v $\oplus$-типа с непосредственными предшественниками u, w. Так как β — бинарное дерево и для каждого $l \in \{1, \ldots, n\}$, $\mathrm{out}_\beta(x_l) = 1$, имеет место следующее: существуют x_i и $x_j \neq x_i$, такие что $\mathrm{res}_{\beta, u}$ зависит от x_i, но не зависит от x_j и $\mathrm{res}_{\beta, w}$ зависит от x_j, но не зависит от x_i. Поскольку $\mathrm{AND}_n|_{x_\nu = 1,\ \nu \neq k}$ зависит от x_k при любом $k = 1, \ldots, n$, имеем

$$\mathrm{res}_{\beta, u}(1, \ldots, 1) = \overline{\mathrm{res}_{\beta, u}(1, \ldots, 1, \underset{i}{0}, 1, \ldots, 1, \underset{j}{0}, 1, \ldots, 1)},$$

$$\mathrm{res}_{\beta, w}(1, \ldots, 1) = \overline{\mathrm{res}_{\beta, w}(1, \ldots, 1, \underset{i}{0}, 1, \ldots, 1, \underset{j}{0}, 1, \ldots, 1)}.$$

Следовательно,

$$\mathrm{res}_{\beta, v}(1, \ldots, 1) = \mathrm{res}_{\beta, v}(1, \ldots, 1, \underset{i}{0}, 1, \ldots, 1, \underset{j}{0}, 1, \ldots, 1),$$

[1]) Бинарным деревом называется дерево с корнем, в котором ребра ориентированы по направлению к корню и при этом в каждую вершину дерева входит либо 0 ребер (тогда это концевая вершина), либо ровно 2 ребра. — *Прим. перев.*

что невозможно, так как β — бинарное дерево. Это доказывает (2).

Предположим [1]), что некоторая вершина v в β не удовлетворяет (3). Из (1) следует, что β — бинарное дерево с n входами. Так как AND_n зависит от всех n переменных, все n входов схемы попарно различны, и, следовательно, из (2) мы имеем

$$\mathrm{res}_{\beta, v} = \Big(\bigwedge_{x_i \in M} x_i^{k_i}\Big)^c, \quad k_i,\ c \in B,$$

где $|M| \geqslant 2$ и $\exists x_j \in M$: $k_j = 0$.

Следовательно, $\mathrm{res}_{\beta, v}(1, \ldots, 1) = 0^c$. Выберем произвольное $x_l \in M$; $l \neq j$. Тогда

$$\mathrm{res}_{\beta, v}(1, \ldots, 1, \underset{l}{0}, 1, \ldots, 1) = \mathrm{res}_{\beta, v}(1, \ldots, 1).$$

Это невозможно, так как β — бинарное дерево. Утверждение (3) доказано. ■

Определение. Пусть β_1 — оптимальная схема для AND_n и β_0 — оптимальная схема для NOR_n. Отождествляя входы x_i схемы β_1 с входами x_i схемы β_0, получаем схему β для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$. Такая схема называется *стандартной схемой* для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$.

Теорема 1. *Каждая оптимальная схема для* $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ *либо является стандартной схемой, либо в ней существует вход* x_i *с выходной степенью* 1.

Доказательство. Пусть β — оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, причем $\mathrm{out}_\beta(x_i) \geqslant 2$ при всех $i \in \{1, \ldots, n\}$. Тогда $x \deg(\beta) \geqslant 2n$. Пусть S — количество элементов ветвления и A — количество выходов схемы β. В силу леммы 2

$$\begin{aligned} 2n - 2 &\geqslant L(\beta) \geqslant \\ &\geqslant x \deg(\beta) + S - A \geqslant \text{ (по лемме 1)} \\ &\geqslant 2n + S - A. \end{aligned}$$

---

[1]) Утверждение (3) доказано авторами не полностью. Его можно доказывать, например, так. Пусть v — произвольная вершина схемы β и пусть $\mathrm{res}_{\beta, v} = \mathrm{res}_{\beta, v}(x_{i_1}, \ldots, x_{i_s})$. Без ограничения общности считаем, что $\mathrm{res}_{\beta, v} = \mathrm{res}_{\beta, v}(x_1, \ldots, x_s)$. Пусть $(\sigma_1, \ldots, \sigma_s)$ — произвольный набор, отличный от $(1, \ldots, 1)$. Предположим, что $\mathrm{res}_{\beta, v}(\sigma_1, \ldots, \sigma_s) = \mathrm{res}_{\beta, v}(1, \ldots, 1)$. Тогда в силу того, что β — бинарное дерево, получим $\mathrm{AND}_n(\sigma_1, \ldots, \sigma_s, 1, \ldots, 1) = \mathrm{AND}_n(1, \ldots, 1)$. Поскольку это неверно, $\mathrm{res}_{\beta, v}(\sigma_1, \ldots, \sigma_s) \neq \mathrm{res}_{\beta, v}(1, \ldots, 1)$. Отсюда следует утверждение (3). — *Прим. перев.*

Значит, $A - S \geqslant 2$, и следовательно, $A \geqslant 2$, так как S неотрицательно. Поскольку β — оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, $A \leqslant 2$. Следовательно, $A = 2$ и $S = 0$. Пусть t_1 — элемент, вычисляющий AND_n, и t_0 — элемент, вычисляющий NOR_n (t_0, t_1 суть 2 выхода схемы β). Пусть β_m ($m = 0, 1$) — схема, состоящая из всех (непосредственных и отдаленных) предшественников t_m. Так как β не содержит элементов ветвления (т. е. $S = 0$), две схемы β_0, β_1 не имеют общих элементов. Из этого следует, что β — стандартная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$. Это доказывает теорему 1. ■

В оставшейся части работы мы докажем следующее утверждение.

Теорема 2. *Пусть β — оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$. Тогда каждый вход x_i схемы β имеет выходную степень $\geqslant 2$.*

Из теорем 1 и 2 непосредственно следует

Главная теорема. *Каждая оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ — это стандартная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$.*

3. ДОКАЗАТЕЛЬСТВО ТЕОРЕМЫ 2

Пусть $p_{\beta, i}$ — путь от x_i ($\mathrm{out}_\beta(x_i) = 1$) до первого элемента ветвления или выхода.

Лемма 3. *Пусть β — оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ с $\mathrm{out}_\beta(x_i) = 1$ для некоторого $i \in \{1, \ldots, n\}$. Тогда последний элемент пути $p_{\beta, i}$ — это элемент ветвления.*

Доказательство. Без ограничения общности положим $i = 1$.

Допустим, что последний элемент c на пути $p_{\beta, 1}$ — не элемент ветвления. Тогда c вычисляет или AND_n, или NOR_n. Пусть t — другой элемент схемы β, вычисляющий одну из двух функций AND_n или NOR_n. Элемент t принадлежат $p_{\beta, 1}$, так как $\mathrm{res}_{\beta, t}$ зависит от x_1. Следовательно, существует функция $h \in B_n$, такая что

$$\mathrm{res}_{\beta, c} = h(\mathrm{res}_{\beta, t}, x_2, \ldots, x_n).$$

Без ограничения общности положим $\mathrm{res}_{\beta, t} = \mathrm{AND}_n$, $\mathrm{res}_{\beta, e} = \mathrm{NOR}_n$. Имеем

$$\mathrm{NOR}_n = h(\mathrm{AND}_n, x_2, \ldots, x_n).$$

Это дает

$$\mathrm{NOR}_n(0, \ldots, 0) = h\left(\bigwedge_{i=1}^{n} 0, 0, \ldots, 0\right) = h(0, 0, \ldots, 0),$$

$$\mathrm{NOR}_n(1, 0, \ldots, 0) = h\left(1 \wedge \bigwedge_{t=2}^{n} 0, 0, \ldots, 0\right) = h(0, 0, \ldots, 0).$$

Следовательно, $\mathrm{NOR}_n(0, 0, \ldots, 0) = \mathrm{NOR}_n(1, 0, \ldots, 0)$. Получили противоречие. ■

Весьма полезна следующая лемма.

Лемма 4. *Пусть* β *— произвольная схема для* $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, *и пусть* g *— элемент на пути* $p_{\beta, i}$ $(\mathrm{out}_\beta(x_i) = 1)$. *Пусть* c *— непосредственный предшественник* g *на* $p_{\beta, i}$ *(это может быть* x_i*) и пусть* d *— другой элемент, предшествующий* g. *Если* $\mathrm{res}_{\beta, g} = (\mathrm{res}^k_{\beta, c} \wedge \mathrm{res}^l_{\beta, d})^m$, $k, l, m \in B$ *(т. е.* g *— элемент* $\wedge$*-типа), то*

$$\mathrm{res}^l_{\beta, d}\, 0, \ldots, (0) = \mathrm{res}^l_{\beta, d}(1, \ldots, 1) = 1.$$

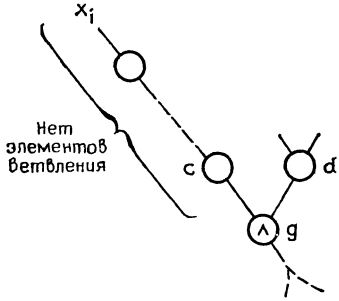


Рис. 1.

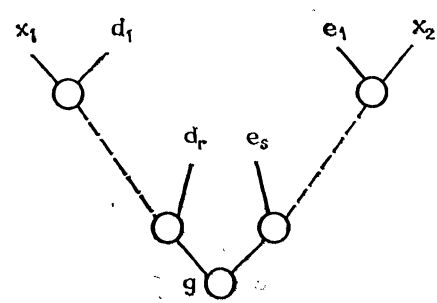


Рис. 2.

Доказательство. В схеме β есть фрагмент, изображенный на рис. 1. Отметим, что функция $\mathrm{res}_{\beta, d}$ не зависит от x_i.

Допустим, что $\mathrm{res}^l_{\beta, d}(0, \ldots, 0) = 0$ или $\mathrm{res}^l_{\beta, d}(1, \ldots, 1) = 0$. Тогда по крайней мере в одном из этих двух случаев $\mathrm{res}_{\beta, g}$ не зависит от x_i, и, следовательно, или AND_n, или NOR_n не зависит от x_i. Получили противоречие. ■

Приведем простое следствие леммы 4.

Лемма 5. *Пусть* β *— произвольная схема для* $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$. *Пусть* $p_{\beta, i}$, $p_{\beta, j}$ $(i \neq j, \mathrm{out}_\beta(x_i) = \mathrm{out}_\beta(x_j) = 1)$ *— два пути в* β, *не содержащие элементов* $\oplus$*-типа. Тогда* $p_{\beta, i}$ *и* $p_{\beta, j}$ *не имеют общих элементов.*

Доказательство. Без ограничения общности положим $i = 1$, $j = 2$. Допустим, что $p_{\beta, 1}$ и $p_{\beta, 2}$ имеют общий элемент. Тогда мы имеем ситуацию, показанную на рис. 2 (g — элемент $\wedge$-типа, в котором пути $p_{\beta, 1}$ и $p_{\beta, 2}$ встречаются; $d_1, \ldots, d_r$ (соответственно $e_1, \ldots, e_s$) — входные вершины для элементов на $p_{\beta, 1}$ (соответственно $p_{\beta, 2}$), не принадлежащие $p_{\beta, 1}$ (соответственно $p_{\beta, 2}$)).

Из конструкции ясно, что функция $\mathrm{res}_{\beta, d_l}$, $l = 1, \ldots, r$ (соответственно $\mathrm{res}_{\beta, e_l}$, $l = 1, \ldots, s$) не зависит от x_1 (соответ-

ственно от x_2)[1]). Отсюда в силу леммы 4 следует, что существуют $a, b, c \in B$, такие что

$$\mathrm{res}_{\beta, g}(x_1, x_2, 0, \ldots, 0) = \mathrm{res}_{\beta, g}(x_1, x_2, 1, \ldots, 1) = (x_1^a \wedge x_2^b)^c.$$

Следовательно, либо $\mathrm{res}_{\beta, g}(x_1, 0, 0, \ldots, 0)$, либо $\mathrm{res}_{\beta, g}(x_1, 1, 1, \ldots, 1)$ не зависит от x_1. Значит, или $\mathrm{NOR}_n(x_1, 0, \ldots, 0)$, или $\mathrm{AND}_n(x_1, 1, \ldots, 1)$ не зависит от x_1. Получили противоречие. ■

Утверждение 1. *Пусть* β *— оптимальная схема для* $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, *которая имеет* $l \geqslant 1$ *входов с выходной степенью* 1. *Пусть* m *— число элементов* $\oplus$*-типа в* β. *Тогда существует оптимальная схема* β' *для* $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, *обладающая следующими двумя свойствами*:

(i) *В* β' *существует элемент* d, *такой, что*

$$\mathrm{res}_{\beta', d}(0, \ldots, 0) = \mathrm{res}_{\beta', d}(1, \ldots, 1).$$

(ii) *Выполняется по крайней мере одно из следующих двух условий*:

a) *В* β' *не больше* $l - 1$ *входов с выходной степенью* 1 *и не больше* m *элементов* $\oplus$*-типа.*

b) *В* β' *не больше* l *входов с выходной степенью* 1 *и не больше* $m - 1$ *элемента* $\oplus$*-типа.*

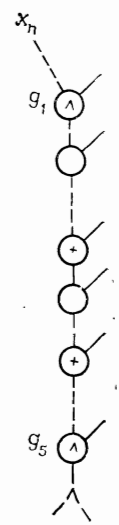


Рис. 3.

Доказательство утверждения 1

Случай 1. Существует вход x_j с выходной степенью 1, такой, что на пути $p_{\beta, j}$ есть элемент $\oplus$-типа.

Без ограничения общности положим $j = n$. Тогда этот путь имеет вид, показанный на рис. 3.

Пусть $g_1, \ldots, g_s$ — элементы $\wedge$-типа на $p_{\beta, n}$ и пусть d_μ $(\mu = 1, \ldots, s)$ — элемент, являющийся предшественником g_μ и не лежащий на $p_{\beta, n}$. Пусть $l_\mu = \mathrm{res}_{\beta, d_\mu}(0, \ldots, 0)$, $\mu = 1, \ldots, s$.

Преобразуем схему β в схему $\overline{\beta}$ следующим образом.

Вычислим $G = \bigwedge_{\mu=1}^{s} \mathrm{res}_{\beta, d_\mu}^{l_\mu}$, используя $s - 1$ дополнительных элементов и вершины $d_1, \ldots, d_s$.

Сначала на входы элементов g_μ подадим l_μ вместо $\mathrm{res}_{\beta, d_\mu}$ для всех $\mu = 1, \ldots, s$. (Это позволяет удалить s элементов $g_1, \ldots, g_s$.) Затем мы вычислим $\mathrm{AND}_n \wedge G$ и $\mathrm{NOR}_n \wedge G$. (Это требует двух дополнительных элементов.) В результате нужен

[1]) Более того, каждая из этих функций не зависит от их x_1 ни от x_2. — *Прим. перев.*

один дополнительный элемент, т. е. $L(\bar{\beta}) = L(\beta) + 1$. Отметим, что

$$\mathrm{res}^{l_\mu}_{\beta,\, d_\mu}(0, \ldots, 0) = \mathrm{res}^{l_\mu}_{\beta,\, d_\mu}(1, \ldots, 1) = 1 \text{ при всех } \mu = 1, \ldots, s$$

(по лемме 4 и определению l_μ). Отсюда следует, что новая схема $\bar{\beta}$ также вычисляет $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ [1]).

По построению все элементы на $p_{\bar{\beta},\, i}$ — элементы $\oplus$-типа (рис. 4).

Так как $(f \oplus h)^0 = (f \oplus h^0)$ и операция $\oplus$ ассоциативна, без потери общности будем считать, что элементом ветвления в

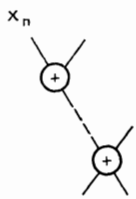


Рис. 4.

Рис. 5.

$p_{\bar{\beta},\, n}$ является преемник вершины x_n, обозначаемый через g. Пусть $d \neq x_i$ [2]) — другой элемент, предшествующий g.

Тогда имеет место ситуация, показанная на рис. 5. Так как $\bar{\beta}$ вычисляет $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, существуют функции $h_0,\ h_1 \in B_n$, такие что

$$\bigwedge_{i=1}^{n} x_i^n = h_m\left(x_1, \ldots, x_{n-1}, x_n \oplus \mathrm{res}_{\bar{\beta},\, d}(x_1, \ldots, x_{n-1})\right), \quad m = 0,\ 1.$$

Если мы заменим x_n на $\left[x_n \oplus \mathrm{res}_{\bar{\beta},\, d}(x_1, \ldots, x_{n-1})\right]$, то последний аргумент функции h_m станет просто x_n. В итоге получаем

$$h_m(x_1, \ldots, x_n) = \bigwedge_{i=1}^{n-1} x_i^m \wedge \left[x_n \oplus \mathrm{res}_{\bar{\beta},\, d}(x_1, \ldots, x_{n-1})\right]^m =$$

$$= \bigwedge_{i=1}^{n-1} x_i^m \wedge \left(x_n^m \oplus \mathrm{res}_{\bar{\beta},\, d}(m, \ldots, m)\right).$$

---

[1]) Надо рассмотреть два случая: а) при всех $\mu = 1, \ldots, s$ $\mathrm{res}_{\beta,\, d_\mu}(\sigma_1, \ldots, \sigma_n) = l_\mu$; b) $\exists \mu$: $\mathrm{res}_{\beta,\, d_\mu}(\sigma_1, \ldots, \sigma_n) \neq l_\mu$. Очевидно, что в этом случае $(\sigma_1, \ldots, \sigma_n) \neq (0, \ldots, 0)$ и $(\sigma_1, \ldots, \sigma_n) \neq (1, \ldots, 1)$. — *Прим. перев.*

[2]) Неясно, почему авторы считают возможным не рассматривать случай $d = x_i$ $(1 \leqslant i \leqslant n-1)$. Из последующих примечаний будет видно, что доказательство проходит и в этом случае. — *Прим. перев.*

(Заметим, что если $(x_1, \ldots, x_{n-1}) \neq (m, \ldots, m)$, то $\bigwedge_{i=1}^{n-1} x_i^m = 0$ и, следовательно, $h_m(x_1, \ldots, x_n) = 0$.)

Случай 1.1. $\mathrm{res}_{\bar{\beta}, d}(0, \ldots, 0) = \mathrm{res}_{\bar{\beta}, d}(1, \ldots, 1) = k \in B$. Получим β' из $\bar{\beta}$, подав k вместо $\mathrm{res}_{\bar{\beta}, d}$ на вход элемента g. Схема β' вычисляет

$$h_m(x_1, \ldots, x_{n-1}, x_n \oplus k) = \bigwedge_{i=1}^{n-1} x_i^m \wedge [(x_n \oplus k)^m \oplus \mathrm{res}_{\bar{\beta}, d}(m, \ldots, m)] =$$

$$= \bigwedge_{i=1}^{n-1} x_i^m \wedge (x_n^m \oplus k \oplus k) = \begin{cases} \mathrm{AND}_n & \text{при } m = 1, \\ \mathrm{NOR}_n & \text{при } m = 0. \end{cases}$$

Таким образом, удален один элемент и в итоге $L(\beta') = L(\beta)$.

Случай 1.2. $\mathrm{res}_{\bar{\beta}, d}(0, \ldots, 0) \neq \mathrm{res}_{\bar{\beta}, d}(1, \ldots, 1)$. Пусть $\mathrm{res}_{\bar{\beta}, d}(m, \ldots, m) = k^m$, $k \in B$. Получим β^* из $\bar{\beta}$, заменяя всюду в схеме $\mathrm{res}_{\bar{\beta}, g}$ на $\bar{k}$. Схема β^* вычисляет

$$h_m(x_1, \ldots, x_{n-1}, \bar{k}) = \bigwedge_{i=1}^{n-1} x_i^m \wedge (\bar{k}^m \oplus k^m) = \bigwedge_{i=1}^{n-1} x_i^m \wedge 1 =$$

$$= \begin{cases} \mathrm{AND}_{n-1} & \text{при } m = 1, \\ \mathrm{NOR}_{n-1} & \text{при } m = 0. \end{cases}$$

Схема β^* для $\{\mathrm{AND}_{n-1}, \mathrm{NOR}_{n-1}\}$ может быть легко дополнена до схемы β' для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ путем введения двух новых элементов, которые вычисляют соответственно

$$\mathrm{AND}_{n-1} \wedge x_n = \mathrm{AND}_n \quad \text{и} \quad \mathrm{NOR}_{n-1} \wedge \bar{x}_n = \mathrm{NOR}_n.$$

При замене $\mathrm{res}_{\bar{\beta}, g}$ на $\bar{k}$ удаляются по крайней мере элемент g и его преемники. Напомним, что g — элемент ветвления. Поэтому в конечном счете удален по крайней мере один элемент. В итоге нам не потребовались дополнительные элементы для преобразования β в β' (т. е. $L(\beta') \leqslant L(\beta)$).

Замечание. (1). В обоих случаях при преобразовании $\bar{\beta}$ в β' удален один элемент, и, следовательно, в силу оптимальности β'

на пути $p_{\beta', i}$ сушествует элемент $\wedge$-типа[1]). По лемме 4 β' содержит элемент t, такой что

$$\mathrm{res}_{\beta', t}(0, \ldots, 0) = \mathrm{res}_{\beta', t}(1, \ldots, 1).$$

Тем самым доказано (i).

(2) В обоих случаях число входов с выходной степенью 1 не увеличивалось[2]), а число элементов $\oplus$-типа уменьшилось по крайней мере на один. Следовательно, (ii) также доказано. Случай 1 разобран.

Случай 2. Для всех входов x_j с выходной степенью 1 все элементы на пути $p_{\beta, j}$ — это элементы $\wedge$-типа.

По лемме 3 последний элемент на любом из этих путей $p_{\beta, j}$ — это элемент ветвления, и по лемме 5 все эти элементы

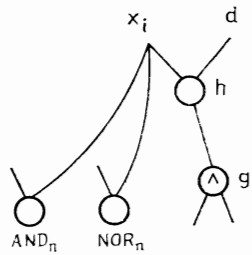


Рис. 6.

попарно различны. Так как $L(\beta) \leqslant 2n - 2$, других элементов ветвления не существует. Следовательно, существует путь $p_{\beta, i}$ в β, такой, что последний его элемент g — это элемент ветвления и среди преемников элемента g нет элементов ветвления.

Сначала преобразуем схему β в схему $\bar{\beta}$ введением двух новых элементов, вычисляющих $\mathrm{AND}_n \wedge x_i$ и $\mathrm{NOR}_n \wedge \bar{x}_i$. Ясно, что два новых выхода вычисляют AND_n и NOR_n. Благодаря этому преобразованию после последующих шагов, удаляющих элементы, NOR_n будет правильно вычисляться при $x_i = 1$ и AND_n — правильно вычисляться при $x_i = 0$.

Рассмотрим $p_{\beta, i}$ как подграф $\bar{\beta}$. Мы собираемся последовательно удалить все элементы на $p_{\beta, i}$. На каждом шаге рассматриваем непосредственного преемника h входа x_i на $p_{\beta, i}$. Имеем следующую ситуацию (рис. 6).

[1]) Для доказательства последнего утверждения достаточно повторить преобразования, примененные к схеме $\bar{\beta}$ в случае 1. — *Прим. перев.*

[2]) В случае 1.1 выходная степень уменьшается лишь у вершины d; но очевидно, что в этом случае $d \neq x_i$. В случае 1.2 возможное увеличение числа входов с выходной степенью 1 (если $d = x_i$) в схеме β' компенсируется тем, что выходная степень x_n становится равной 2. — *Прим. перев.*

Пусть $d \neq x_i$ [1]) — другой элемент, предшествующий h, и пусть $l = \mathrm{res}_{\beta, d}(0, \ldots, 0)$. По лемме 4 существует $a \in B$, такое, что

$$\mathrm{res}_{\bar{\beta}, h}(0, \ldots, 0, x_i, 0, \ldots, 0) = \\ = \mathrm{res}_{\bar{\beta}, h}(1, \ldots, 1, x_i, 1, \ldots, 1) = x_i^a.$$ [2])

Теперь удалим элемент h, подав l вместо $\mathrm{res}_{\bar{\beta}, d}$ на вход элемента h. Затем введем новый элемент, который вычисляет

$$\mathrm{AND}_n \wedge \mathrm{res}^l_{\bar{\beta}, d}, \text{ если } a = 1,$$

или

$$\mathrm{NOR}_n \wedge \mathrm{res}^l_{\bar{\beta}, d}, \text{ если } a = 0.$$

В итоге на этом шаге не нужно дополнительных элементов. По лемме 4

$$\mathrm{res}^l_{\bar{\beta}, d}(0, \ldots, 0) = \mathrm{res}^l_{\bar{\beta}, d}(1, \ldots, 1) = 1.$$

Кроме того, $\mathrm{res}_{\bar{\beta}, h}$ зависит от $\mathrm{res}_{\bar{\beta}, d}$ только при $x_i = a$. Отсюда следует, что обе функции AND_n и NOR_n вычисляются правильно [3]).

Пусть β^* — схема, полученная после удаления последнего элемента g из $p_{\beta, i}$. Тогда по построению $L(\beta^*) \leqslant L(\beta) + 2$ и $\mathrm{out}_{\beta^*}(x_i) = 4$.

Так как ни на каком шаге новые элементы ветвления не вводились, среди преемников x_i нет элементов ветвления. Следовательно, по построению β^* существуют путь от x_i к выходу, вычисляющему AND_n, и путь от x_i к выходу, вычисляющему NOR_n, возникшие при преобразовании β в $\bar{\beta}$. Кроме того, существует другой путь p от x_i к выходу, вычисляющему AND_n, и существует другой путь p' от x_i к выходу, вычисляющему NOR_n, которые являются остатками путей схемы $\bar{\beta}$, проходивших от x_i через $p_{\beta, i}$ к выходам схемы.

Подавая 1 на ребро, выходящее из x_i в p, и подавая 0 на ребро, выходящее из x_i в p', экономим два элемента.

---

[1]) Например, в силу леммы *4*. — *Прим. перев.*

[2]) В дальнейшем авторы подразумевают, что op_h имеет вид $x_i^a \wedge d^l$, считая, что последующее отрицание (если оно есть) «включено» в операции непосредственных преемников элемента h. — *Прим. перев.*

[3]) Действительно, пусть, например, $a = 1$. Тогда очевидно, что AND_n вычисляется правильно (см. прим. 1 на с. 112). Функция NOR_n тоже вычисляется правильно, так как при $x_i = 0$ $\mathrm{res}_{\bar{\beta}, h}$ не зависит от $\mathrm{res}_{\bar{\beta}, d}$, а при $x_i = 1$ обращается в 0 выход введенного ранее элемента, вычисляющего $\mathrm{NOR}_n \wedge \bar{x}_i$. — *Прим. перев.*

Благодаря преобразованию, сделанному при переходе от β к $\bar{\beta}$, AND_n продолжает вычисляться верно при $x_i = 0$, а NOR_n — при $x_i = 1$.

Таким образом получена схема β', вычисляющая $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, для которой $L(\beta') \leqslant L(\beta)$, т. е. β' — оптимальная схема.

Замечание. Как следует из леммы 4, β' содержит элемент t, такой что

$$\mathrm{res}_{\beta', t}(0, \ldots, 0) = \mathrm{res}_{\beta', t}(1, \ldots, 1).^{1})$$

Число входов с выходной степенью 1 уменьшилось на единицу, а число элементов $\oplus$-типа не увеличилось.

Утверждение 1 доказано.

Доказательство теоремы 2.

Допустим, что существует оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$, имеющая по крайней мере один вход с выходной степенью 1. В силу утверждения 1 можно построить схему β' из схемы β со следующими свойствами:

(1) β' — оптимальная схема для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$.
(2) Все входы в β' имеют выходную степень 2.
(3) Существует элемент g в β', такой что

$$\mathrm{res}_{\beta', g}(0, \ldots, 0) = \mathrm{res}_{\beta', g}(1, \ldots, 1).$$

Из леммы 2 следует, что β' не является стандартной схемой для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$. С другой стороны, по теореме 1 или β' — стандартная схема, или в β' существует вход x_i с выходной степенью 1. Получили противоречие.

Следовательно, оптимальная схема β' для $\{\mathrm{AND}_n, \mathrm{NOR}_n\}$ не может иметь входа с выходной степенью 1.

Теорема 2 доказана.

ЛИТЕРАТУРА

[1] Paul, W., A 2.5n lower bound on the combinatorial complexity of Boolean functions. SIAM J. Comput. 6, 427—433 (1977). [Имеется перевод: Пауль В. И., Нижняя оценка для комбинационной сложности булевых функций. Киб. сб. № 16, с. 23—44. — М.: Мир, 1979.]
[2] Paul, W., Realizing Boolean functions on disjont sets of variables. Theor. Comp. Sci. 2, 383—396 (1976).

---

[1]) Например, можно взять $t = d$. — *Прим. перев.*

[3] Pipenger, N., Fisher, M. J., Relation among complexity measures. JACM 26, 361—381 (1979).
[4] Savage, J. E., Computational work and time on finite machines. JACM 19, 660—674 (1972).
[5] Schnorr, C. P., The network complexity of finite functions. Acta Informat. 7, 95—107 (1976).
[6] Schnorr, C. P., The combinatorial compexity of equvalence. Theor. Comp. Sci. 1, 289—295 (1976). [Имеется перевод: Шнорр К. П., Комбинационная сложность эквивалентности, Киб. сб. № 16, с. 74—81. — М.: Мир, 1979.]

Характеризация P- и Q-полиномиальных схем отношений[1)]

П. Тервиллигер[2)]

Пусть $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ — симметричная схема отношений с d классами, с числами пересечений p_{ij}^h и параметрами Крейна q_{ij}^h $(0 \leqslant h, i, j \leqslant d)$. Для каждого i $(0 \leqslant i \leqslant d)$ определим i-ю (приведенную) диаграмму пересечений D_i (соответственно диаграмму представлений D_i^*) на вершинах $0, 1, \ldots, d$, соединяя неориентированным ребром каждую пару вершин h, j, для которых $p_{ij}^h > 0$ (соответственно $q_{ij}^h > 0$). Схема Y называется P-полиномиальной (соответственно Q-полиномиальной), если некоторая D_i (соответственно D_i^*) является путем. Мы получим положительно полуопределенные матрицы $G(i)$ и $G(i)^*$ $(0 \leqslant i \leqslant d)$, которые дадут новые неравенства для чисел p_{ij}^h и q_{ij}^h. Кроме того, мы покажем, что при каждом i $(0 \leqslant i \leqslant d)$ эквивалентны следующие утверждения: диаграмма D_i^* является лесом, матрица $G(i)$ обращается в нулевую, существует определенная геометрическая интерпретация множества X. Аналогичные результаты справедливы для $G(i)^*$ и D_i. Называя листом произвольной диаграммы вершину, смежную ровно с одной другой вершиной, мы покажем, что в любой связной диаграмме D_i^* для P-полиномиальной схемы помимо 0-вершины имеется самое большее один лист. Объединив этот и приведенный выше результаты, мы получим некоторую интерпретацию свойства Q-полиномиальности для P-полиномиальных схем. В заключение мы используем уравнения, возникающие в силу обращения в нулевую некоторой $G(i)$ для получения простого доказательства теоремы Д. Леонарда о том, что числа пересечений P- и Q-полиномиальной схемы могут быть найдены с помощью 5 параметров.

[1)] Paul Terwilliger, A Characterization of P- and Q-Polynomial Association Schemes, Journal of Combinatorial Theory, Series A 45, 8—26 (1987).

[2)] Department of Mathematics, The Ohio State University, Columbus, Ohio 43210.

1. ВВЕДЕНИЕ

Пусть $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ — симметричная схема отношений с d классами (см. определение 1.2), и пусть p_{ij}^h и q_{ij}^h $(0 \leqslant h, i, j \leqslant d)$ — числа пересечений и параметры Крейна схемы Y. Для каждого i $(0 \leqslant i \leqslant d)$ определим i-ю (приведенную) *диаграмму пересечений* D_i (соответственно *диаграмму представлений* D_i^*) на вершинах $\{0, 1, \ldots, d\}$, соединяя неориентированные ребром любые две *различные* вершины h и j, для которых $p_{ij}^h > 0$ (соответственно $q_{ij}^h > 0$). Схему Y называют *P-полиномиальной* (соответственно Q-полиномиальной), если диаграмма D_i (соответственно D_i^*) является *путем* при некотором i. В этой статье мы свяжем геометрию схемы Y со структурой диаграммы D_i и D_i^* $(0 \leqslant i \leqslant d)$. В нашем первом основном результате, теореме 2.3, мы найдем множество положительно полуопределенных матриц $G(i)$ и $G(i)^*$ $(0 \leqslant i \leqslant d)$, которые дадут новые неравенства для чисел пересечений и параметров Крейна произвольной схемы отношений Y. Эти неравенства обусловлены неотрицательностью чисел пересечений и параметров Крейна (условие Крейна см.[1]) в [1], с. 73) и очень полезны. В теореме 2.5 мы показываем, что при каждом i $(0 \leqslant i \leqslant d)$ эквивалентны три следующих утверждения: диаграмма D_i^* является лесом (объединением непересекающихся деревьев), матрица $G(i)$ обращается в нулевую, существует определенная геометрическая интерпретация схемы Y. Далее мы замечаем, что обращение матрицы $G(i)$ (соответственно $G(i)^*$) в нулевую, когда D_i^* (соответственно D_i) является лесом, дает много равенств, связывающих числа пересечений (соответственно параметры Крейна) схемы Y. Можно надеяться, что эти равенства прольют достаточно света на схемы отношений, для которых как некоторая диаграмма пересечений D_i $(i \neq 0)$, так и некоторая диаграмма представлений D_j^* $(j \neq 0)$ являются лесами, чтобы мы могли классифицировать эти схемы. Это есть обобщение Баннаи и Ито [1, стр. 156] о классификации P- и Q-полиномиальных схем отношений.

В разд. 3 мы сконцентрируем внимание на P-полиномиальных схемах. Так как Баннаи и Ито [1] и Стантон [12] заметили, что при $d \geqslant 13$ все известные P-полиномиальные схемы либо являются Q-полиномиальными схемами, либо связаны с одной из них с помощью «деления пополам» двудольных графов или образования антиподальных частных, то возникает

[1]) Здесь и далее указываются номера страниц в русском переводе книги [1]. — *Прим. перев.*

вопрос, какие диаграммы представлений могут быть у P-полиномиальных схем. Наша теорема 3.3 дает частичный ответ на этот вопрос. Называя *листом* диаграммы любую вершину, смежную ровно с одной другой вершиной, мы докажем, что в любой связной диаграмме представлений имеется самое большее один лист, не считая O-вершины (которая всегда является листом). В частности, если некоторая диаграмма представлений является деревом, то она является путем, и эта схема является Q-полиномиальной. Мы соединяем эту информацию с теоремой 2.5 и получаем следующий результат, который дает геометрическую интерпретацию свойства Q-полиномиальности для P-полиномиальных схем.

Теорема 1.1. *Симметричная схема отношений* $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ *обладает некоторой приведенной диаграммой представлений, которая является деревом, тогда и только тогда, когда множество* X *может быть представлено множеством* $\{x^* \mid x \in X\}$ *различных единичных векторов, порождающих некоторое евклидово пространство* U, $<$, $>$ *таким, что*

1) $\langle x^*, y^* \rangle$ *зависит только от* i, *для которого* $(i, y) \in R_i$; (1.1)

2) *для всех* $x \in X$ *и всех* i $(0 \leqslant i \leqslant d)$ *вектор* $\sum_{y \in X,\, (x, y) \in R_i} y^*$ пропорционален вектору x^*; (1.2)

3) *для всех* x, $y \in X$ *и всех* i, j $(0 \leqslant i,\ j \leqslant d)$ *вектор* $\sum_{y \in X,\, (x, y) \in R_i,\, (z, y) \in R_j} z^* - \sum_{w \in X,\, (x, w) \in R_j,\, (w, y) \in R_i} w^*$ *пропорционален вектору* $x^* - y^*$. (1.3)

Кроме того, если схема Y *является* P*-полиномиальной, то приведенная выше фраза «обладает приведенной диаграммой представлений, которая является деревом» может быть заменена на «является* Q*-полиномиальной».*

Упомянем об одном важном следствии из (1.3). Мы говорим, что четверки (x_1, x_2, x_3, x_4) и (y_1, y_2, y_3, y_4) элементов из X имеют один и тот же *тип*, если (x_i, x_j) и (y_i, y_j) принадлежат одному и тому же классу схемы Y при всех i, j $(1 \leqslant i,\ j \leqslant 4)$. Обозначим через n_T число четверок из X типа T. Кроме того, для любых x, $y \in X$ и любых i, j $(0 \leqslant i \leqslant d)$ обозначим через $r_{ij}(x, y)$ вектор в левой части (1.3). С помощью вычисления различных $\langle r_{ij}(x, y), r_{st}(x, y) \rangle$ $(0 \leqslant i,\ j,\ s,\ t \leqslant d)$ мы покажем в последующей статье, что намного больше информации о числах n_T получается из чисел пересечений схемы, удовлетворяющей условиям теоремы 1.1, по сравнению с общим случаем. В частности,

для P- и Q-полиномиальных схем мы покажем, что все числа n_r могут быть определены заданными числами пересечений и $[d/2]$ свободными параметрами (которые должны удовлетворять определенным неравенствам). Этот результат должен помочь при классификации схем числами пересечений (см., например, Егава [5] для графа Хэмминга и Хуанг [6] для его q-аналога, Неймайер [9] и Тервиллигер [15] для графа Джонсона и Спраги [11] для его q-аналога).

В нашем последнем результате, теореме 3.4, мы используем уравнение, возникающее в силу обращения некоторой матрицы $G(i)$ в нулевую, для получения простого доказательства теоремы Леонарда [8] о том, что числа пересечений P- и Q-полиномиальной схемы могут быть определены с помощью пяти параметров.

В оставшейся части этого раздела мы введем термины и приведем некоторые основные сведения. За подробностями мы отсылаем читателя к работам Баннаи и Ито [1], Дельсарта [4] или Слоэна [10]. Для произвольного целого положительного числа d обозначим через $[d]$ множество $\{0, 1, \ldots, d\}$ и положим

$$S_d = \{(i, j) \mid i, j \in [d],\ i < j\}. \tag{1.4}$$

Определение 1.2. *Симметричная схема отношений с d классами* (или просто *схема*) есть совокупность $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$, состоящая из конечного множества X и симметричных отношений R_0, R_1, R_d на X, где

1) $R_0 = \{(x, x) \mid x \in X\}$ — тождественное отношение;

2) для любых x, $y \in X$ имеет место $(x, y) \in R_i$ для единственного $i \in [d]$;

3) для любых h, i, $j \in [d]$ и любых x, $y \in X$, таких что $(x, y) \in R_h$, и число элементов $z \in X$, для которых $(x, z) \in R_i$ и $(z, y) \in R_j$, зависит только от h, i и j. Мы обозначим это число через p_{ij}^h.

Мы называем числа p_{ij}^h *числами пересечений*, а отношения R_i — *классами схемы* Y.

В дальнейшем будем считать, что $n = |X|$. Пусть *матрицы смежности* $A_0 = I$, A_1, $\ldots$, A_d схемы Y имеют строки и столбцы, занумерованные элементами множества X, и удовлетворяют соотношению

$$(A_i)_{xy} = \begin{cases} 1, & \text{если } (x, y) \in R_i, \quad (x, y \in X,\ i \in [d]). \\ 0 & \text{в противном случае.} \end{cases}$$

В силу пункта 3) определения 1.2 эти матрицы удовлетворяют соотношению

$$A_i A_j = \sum_{h=0}^{d} p_{ij}^h A_h \quad (i, j \in [d]),$$

и, следовательно, они порождают некоторую алгебру $\mathscr{A}(Y)$ над R, называемую алгеброй *Боуза-Меснера*. Пусть $E_0, E_1, \ldots, E_d$ — примитивные идемпотенты алгебры $\mathscr{A}(Y)$, упорядоченные таким образом, что $nE_0 = J$, где J — матрица, все элементы которой равны единице, и пусть P и Q — матрицы порядка $d+1$, чьи (i, j)-элементы $p_j(i)$ и $q_j(i)$ определяются соотношениями

$$A_j = \sum_{i=0}^{d} p_j(i) E_i$$

и

$$E_j = n^{-1} \sum_{i=0}^{d} q_j(i) A_i. \tag{1.5}$$

Параметры Крейна q_{ij}^h $(h, i, j \in [d])$ схемы Y определяются соотношениями

$$E_i \cdot E_j = n^{-1} \sum_{h=0}^{d} q_{ij}^h E_h, \tag{1.6}$$

где $\cdot$ есть адамарово (поэлементное) умножение матриц. Полагая $k_i = p_{ii}^0$ и $m_i = q_{ii}^0$ $(i \in [d])$, мы получаем, согласно [1, стр. 69], что $k_i q_i(j) = m_i p_j(i)$ $(i, j \in [d])$, и, следовательно, можно определить матрицу *косинусов* C порядка $d+1$, чей (i, j)-й элемент удовлетворяет равенству

$$\begin{aligned} c_j(i) &= q_j(i) m_j^{-1} \\ &= p_i(j) k_i^{-1} \quad (i, j \in [d]). \end{aligned} \tag{1.7}$$

Из хорошо известных равенств, которые имеются в указанных выше источниках, немедленно следуют следующие соотношения:

$$\begin{aligned} k_h p_{ij}^h &= k_i p_{hj}^i \quad (h, i, j \in [d]), \\ m_h q_{ij}^h &= m_i q_{hj}^i \quad (h, i, j \in [d]), \end{aligned} \tag{1.8}$$

$$q_{i0}^h = \delta_{hi} \quad (h, i \in [d]), \tag{1.9}$$

$$\begin{aligned} \sum_{j=0}^{d} p_{ij}^h c_t(j) &= k_i c_t(i) c_t(h) \quad (h, i, t \in [d]), \\ \sum_{j=0}^{d} q_{ij}^h c_j(t) &= m_i c_i(t) c_h(t) \quad (h, i, t \in [d]), \end{aligned} \tag{1.10}$$

$$p_{ij}^h = k_i k_j n^{-1} \sum_{f=0}^{d} m_f c_f(h) c_f(i) c_f(j) \quad (h, i, j \in [d]), \tag{1.11}$$

$$q_{ij}^h = m_i m_j n^{-1} \sum_{f=0}^{d} k_f c_h(f) c_i(f) c_j(f) \geqslant 0 \quad (h, i, j \in [d]). \tag{1.12}$$

Ради упрощения обозначений будем считать, что $c_i(j) = 0$ для любых целых положительных чисел i и j, для которых i или j не принадлежит $[d]$. Нам понадобится следующее утверждение о диаграммах пересечений и представлений, определенных в начале этой статьи.

Лемма 1.3. *Пусть* $i \in [d]$. *Тогда число компонент связности диаграммы* D_i (*соответственно* D_i^*) *равно числу тех* $j \in [d]$, *для которых* $c_j(i) = 1$ (*соответственно* $c_i(j) = 1$).

Доказательство. Мы рассмотрим только диаграммы D_i, поскольку для D_i^* рассуждения аналогичны. Определим полную диаграмму пересечений FD_i на вершинах $\{0, 1, \ldots, d\}$, проводя ориентированное ребро от вершины h к вершине j всякий раз, когда $p_{ij}^h > 0$, и припишем этому ребру пометку p_{ij}^h. В этой диаграмме возможны петли, и в силу (1.8), если имеется ребро из j в h ($h, j \in [d]$), то имеется ребро из h в j. Диаграмма D_i получается из FD_i удалением пометок, петель и объединением пар ребер (h, j) и (j, h), которые могут существовать в FD_i, в одно неориентированное ребро. Очевидно, что число компонент в D_i и FD_i одно и то же. Далее пусть B_i — матрица смежности для FD_i, т. е. матрица порядка $d+1$, чей (h, j)-й элемент равен пометке на ребре от h к j, если оно существует, и равен 0, если такого ребра нет. В силу (1.10)

$$B_i C = C k_i \operatorname{diad}\{c_0(i) = 1,\ c_1(i),\ \ldots,\ c_d(i)\}. \tag{1.13}$$

В частности, первый столбец матрицы C, состоящий из одних единиц, является собственным вектором для B_i. Так как все элементы в B_i неотрицательны, то по теореме Фробениуса (см., например [7]) кратность максимального собственного значения k_i матрицы B_i равна числу компонент FD_i. Утверждение леммы следует теперь из (1.13). ■

Пусть $\mathscr{A}(Y)$ действует на евклидовом пространстве V, $\langle , \rangle$, обладающем ортонормированным базисом, который мы отождествим с элементами множества X, и пусть $V = V_0 + V_1 + \ldots + V_d$ есть ортогональная прямая сумма его максимальных подпространств, инвариантных относительно $\mathscr{A}(Y)$. Обозначим через π_j ортогональную проекцию V на V_j ($j \in [d]$) и занумеруем инвариантные подпространства таким образом, что E_j является матричным представлением π_j относительно упомянутого выше базиса пространства V для всех $j \in [d]$. Так как $E_j^2 = E_j^t = E_j$ ($j \in [d]$), то можно рассматривать E_j как матрицу Грама векторов $\{\pi_j(x) \mid x \in X\}$, откуда в силу (1.5) следует, что

$$\langle \pi_j(x),\ \pi_j(y) \rangle = m_j n^{-1} c_j(i), \text{ когда } (x,\ y) \in R_i\ (x,\ y \in X). \tag{1.14}$$

Кроме того,

$$\pi_j(x) = \sum_{y \in X} (E_j)_{yx}\, y = \sum_{y \in X} \langle \pi_j(x),\ \pi_j(y) \rangle\, y. \tag{1.15}$$

2. ДИАГРАММЫ ПЕРЕСЕЧЕНИЙ И ПРЕДСТАВЛЕНИЙ

В этом разделе мы исследуем фиксированную схему $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ и найдем новые неравенства для чисел пересечений и параметров Крейна. Мы покажем важность достижения в них равенства, что происходит, когда некоторая диаграмма пересечений или представлений схемы Y является лесом, и этот случай приводит к некоторому геометрическому представлению схемы Y. Векторное пространство V и проекции $\pi_0, \pi_1, \ldots, \pi_d$ упомянуты после леммы 1.3. Символ $[d]$ введен перед определением (1.4) множества S_d, а константы $c_i(j)$ определены равенством (1.7).

Определение 2.1. Для любого $i \in [d]$ положим

$$[d]_i = \{j \mid j \in [d],\quad c_j(i) \neq 1\},$$
$$[d]_i^* = \{j \mid j \in [d],\quad c_i(j) \neq 1\}.$$

Определение 2.2. Для любого $t \in [d]$ обозначим через $G(t)$ и $G(t)^*$ матрицы, строки и столбцы которых занумерованы элементами множества S_d, такие что

$$G(t)_{ij,\,i'j'} = \sum_{h=0}^{d} k_h c_t(h)\,(p_{ii'}^h p_{jj'}^h - p_{ij'}^h p_{i'j}^h) -$$
$$- \sum_{h \in [d]_t^*} \frac{k_h (c_t(i) - c_t(j))\,(c_t(i') - c_t(j'))\, p_{ij}^h p_{i'j'}^h}{1 - c_t(h)}$$
$$((i, j) \in S_d,\ (i', j') \in S_d);$$

$$G(t)^*_{ij,\,i'j'} = \sum_{h=0}^{d} m_h c_h(t)\,(q_{ii'}^h q_{jj'}^h - q_{ij'}^h q_{i'j}^h) -$$
$$- \sum_{h \in [d]_t} \frac{m_h (c_i(t) - c_j(t))\,(c_{i'}(t) - c_{j'}(t))\, q_{ij}^h q_{i'j'}^h}{1 - c_h(t)}$$
$$((i, j) \in S_d,\ (i', j') \in S_d).$$

Теорема 2.3. *Матрицы $G(t)$ и $G(t)^*$ из определения 2.2 являются положительно полуопределенными при всех $t \in [d]$.*

В частности,

$$\sum_{h=0}^{d} k_h c_t(h)\,(p_{ii}^h p_{jj}^h - (p_{ij}^h)^2) - \sum_{h\in[d]_t}^{*} \frac{k_h\,(c_t(i)-c_t(j))^2\,(p_{ij}^h)^2}{1-c_t(h)} \geqslant 0, \quad (2.1)$$

$$\sum_{h=0}^{d} m_h c_h(t)\,(q_{ii}^h q_{jj}^h - (q_{ij}^h)^2) - \sum_{h\in[d]_i} \frac{m_h\,(c_i(t)-c_j(t))^2\,(q_{ij}^h)^2}{1-c_h(t)} \geqslant 0. \quad (2.2)$$

Доказательство этого утверждения будет проведено после следствия 2.17. Мы теперь дадим геометрическую интерпретацию обращения в нуль некоторой из матриц $G(t)$ или $G(t)^*$.

Определение 2.4. Для любых $i, j\in[d]$ и любых $x, y\in X$ обозначим через $P_{ij}(x, y)$ и $Q_{ij}(x, y)$ векторы в V, такие что

$$P_{ij}(x, y) = \sum_{\substack{z\in X\\(x,z)\in R_i\\(z,y)\in R_j}} z$$

и

$$Q_{ij}(x, y) = \sum_{z\in X} \langle \pi_i(x),\ \pi_i(z)\rangle\langle \pi_j(y),\ \pi_j(z)\rangle z.$$

Теорема 2.5. *При любом фиксированном $t\in[d]$ эквивалентны следующие утверждения:*

1) *диаграмма представления D_t^* является лесом,*

2) *вектор $\pi_t(P_{ij}(x, y) - P_{ji}(x, y))$ пропорционален вектору $\pi_t(x-y)$ при всех $x, y\in X$ и всех $i, j\in[d]$,*

3) *матрица $G(t)$ является нулевой.*

Кроме того, лес в пункте 1) является деревом тогда и только тогда, когда векторы $\{\pi_t(x)\mid x\in X\}$ различны.

Теорема 2.6. *При любом фиксированном $t\in[d]$ эквивалентны следующие утверждения:*

1) *диаграмма пересечений D_t является лесом,*

2) *вектор $\pi_r(Q_{ij}(x, y) - Q_{ji}(x, y))$ пропорционален вектору $\pi_r(x-y)$ при всех $x, y\in X$, для которых $(x, y)\in R_t$ и всех $i, j, r\in[d]$,*

3) *матрица $G(t)^*$ является нулевой.*

Кроме того, лес в пункте 1) является деревом тогда и только тогда, когда $\pi_r(x)\neq\pi_r(x)$ при всех $r\in[d]$ и всех $x, y\in X$, для которых $(x, y)\in R_t$.

Доказательства следуют из леммы 2.19. Нам будут необходимы следующие предварительные результаты.

Определение 2.7. Пусть V — тензорное произведение $V = V \otimes V \otimes V$. Введем на V скалярное произведение $\langle , \rangle$ так, чтобы множество $\{x \otimes y \otimes z \mid x, y, z \in X\}$ образовывало некоторый ортонормированный базис. Заметим, что

$$\langle u \otimes v \otimes w, u' \otimes v' \otimes w' \rangle = \langle u, u' \rangle \langle v, v' \rangle \langle w, w' \rangle$$

для всех $u, v, w, u', v', w' \in V$. Для каждого $t \in [d]$ определим подпространство Y_i пространства $V \otimes V$ формулой $Y_i = \langle \{x \otimes y \mid x, y \in X, (x, y) \in R_i\} \rangle$ (где $\langle\ \rangle$ обозначает линейную оболочку) и положим $V_i = Y_i \otimes V$ и $V_i^* = V \otimes V \otimes V_i$ $(i \in [d])$. Здесь V_i есть i-е инвариантное подпространство V. Наконец, обозначая через p_i и p_i^* $(i \in [d])$ ортогональные проекции из V на V_i и V_i^* соответственно, заметим, что они удовлетворяют следующим условиям:

$$p_i(x \otimes y \otimes z) = \begin{cases} x \otimes y \otimes z, & \text{если } (x, y) \in R_i \quad (x, y, z \in X,\ i \in [d]) \\ 0 & \text{в противном случае,} \end{cases} \tag{2.3}$$

$$p_i^*(x \otimes y \otimes z) = x \otimes y \otimes \pi_i(z) \quad (x, y, z \in X,\ i \in [d]). \tag{2.4}$$

Заметим также, что p_i и p_j^* коммутируют при всех $i, j \in [d]$.

Теперь мы определим подпространство W и W_1 пространства V и рассмотрим два базиса для каждого из них: H и H^* для W и H_1 и H_1^* для W_1. Затем мы изучим действие p_i и p_i^* на этих базисах.

Определение 2.8. Для любых $i, j \in [d]$ положим

$$e_{ij} = \sum_{z \in X} \sum_{\substack{x \in X \\ (x, z) \in R_i}} \sum_{\substack{y \in X \\ (y, z) \in R_j}} x \otimes y \otimes z \tag{2.5}$$

и определим множества H и H_1 формулами

$$H = \{e_{ij} \mid i, j \in [d]\},$$
$$H_1 = \{e_{ij} - e_{ji} \mid (i, j) \in S_d\}. \tag{2.6}$$

Положим также $W = \langle H \rangle$ и $W_1 = \langle H_1 \rangle$.

Определение 2.9. Для любых $s, t \in [d]$ положим

$$e_{s,t}^* = n \sum_{z \in X} \pi_s(z) \otimes \pi_t(z) \otimes z, \tag{2.7}$$

определим множества H^* и H_1^* формулами

$$H^* = \{e^*_{s,t} \mid s,\ t \in [d]\},$$
$$H_1^* = \{e^*_{st} - e^*_{ts} \mid (s,\ t) \in S_d\}.$$

Лемма 2.10. *Множества H^* и H_1^* содержатся в W и W_1 соответственно.*

Доказательство. Применяя (1.14) и (1.15) к (2.7), получаем

$$e^*_{st} = n \sum_{z \in X} \sum_{x \in X} \sum_{y \in X} \langle \pi_s(x),\ \pi_s(z)\rangle \langle \pi_t(y),\ \pi_t(z)\rangle\, x \otimes y \otimes z =$$

$$= m_s m_t n^{-1} \sum_{i=0}^{d} \sum_{j=0}^{d} c_s(i)\, c_t(j)\, e_{ij} \quad (s,\ t \in [d])$$

и

$$e^*_{st} - e^*_{ts} = m_s m_t n^{-1} \times$$
$$\times \sum_{i=0}^{d-1} \sum_{j=i+1}^{d} (c_s(i)\, c_t(j) - c_t(i)\, c_s(j))\,(e_{ij} - e_{ji}) \quad ((s,\ t) \in S_d). \blacksquare$$

Лемма 2.11. *При всех i, j, s, $t \in [d]$ имеет место*

$$\langle e_{ij},\ e_{st}\rangle = \delta_{is}\delta_{jt} n k_i k_j, \tag{2.8}$$
$$\langle e^*_{ij},\ e^*_{st}\rangle = \delta_{is}\delta_{jt} n m_i m_j. \tag{2.9}$$

В частности, множества H и H^* являются ортогональными базисами для W, а H_1 и H_1^* являются ортогональными базисами для W_1.

Доказательство. Равенства (2.8) следуют непосредственно из (2.2). Чтобы получить (2.9), используем (1.14), (2.2) и ортогональность инвариантных подпространств пространства V:

$$\langle e^*_{ij},\ e^*_{st}\rangle = \Big\langle n \sum_{z \in X} \pi_i(z) \otimes \pi_j(z) \otimes z,\ n \sum_{w \in X} \pi_s(w) \otimes \pi_t(w) \otimes w \Big\rangle =$$
$$= n^2 \sum_{z \in X} \langle \pi_i(z),\ \pi_s(z)\rangle \langle \pi_j(z),\ \pi_t(z)\rangle = \delta_{is}\delta_{jt} n m_i m_j. \blacksquare$$

Перейдем теперь к вычислению матриц Грама множеств векторов $p_t(H_1)$, $p_t(H_1^*)$, $p_t^*(H_1)$ и $p_t^*(H_1^*)$ $(t \in [d])$. Вычислим сначала действие p_t и p_t^* на H_1 на H_1^*.

Лемма 2.12. *При всех* $i, j, t \in [d]$ *имеет место*

$$p_t(e_{ij}) = \sum_{z \in X} \sum_{\substack{x \in X \\ (x,z) \in R_i}} \sum_{\substack{y \in X \\ (x,z) \in R_t \\ (y,z) \in R_j}} x \otimes y \otimes z, \tag{2.10}$$

$$p_t^*(e_{ij}) = \sum_{z \in X} \sum_{\substack{x \in X \\ (x,z) \in R_i}} \sum_{\substack{y \in X \\ (y,z) \in R_t}} x \otimes y \otimes \pi_t(z), \tag{2.11}$$

$$p_t(e_{ij}^*) = n \sum_{z \in X} \sum_{x \in X} \sum_{\substack{y \in X \\ (x,y) \in R_t}} \langle \pi_i(x), \pi_i(z) \rangle \langle \pi_j(y), \pi_j(z) \rangle x \otimes y \otimes z, \tag{2.12}$$

$$p_t^*(e_{ij}^*) = n \sum_{z \in X} \pi_i(z) \otimes \pi_j(z) \otimes \pi_t(z). \tag{2.13}$$

Доказательство. Следует непосредственно из (1.15) и (2.3)–(2.7). ■

Лемма 2.13. *При всех* $i, j, r, s, t \in [d]$ *имеет место*

$$\langle p_t(e_{ij}), p_t(e_{et}) \rangle = \delta_{ir}\delta_{js} n k_t p_{ij}^t.$$

В частности, для любого $t \in [d]$ *имеет место*

$$W \cap \mathrm{Ker}(p_t) = \langle \{ e_{ij} \mid i, j \in [d],\ p_{ij}^t = 0 \} \rangle. \tag{2.14}$$

Доказательство. Следует непосредственно из (2.10). ■

Лемма 2.14. *При всех* $i, j, r, s, t \in [d]$ *имеет место*

$$\langle p_t^*(e_{ij}^*), p_t^*(e_{rs}) \rangle = \delta_{ir}\delta_{js} n m_t q_{ij}^t.$$

В частности, для любого $t \in [d]$ *имеет место*

$$W \cap \mathrm{Ker}(p_t^*) = \langle \{ e_{ij}^* \mid i, j \in [d],\ q_{ij}^t = 0 \} \rangle. \tag{2.15}$$

Доказательство. В силу (2.13), (2.2), (1.14) и (1.12)

$$\langle p_t^*(e_{ij}^*), p_t^*(e_{rs}^*) \rangle =$$

$$= n^2 \Big\langle \sum_{z \in X} \pi_i(z) \otimes \pi_j(z) \otimes \pi_t(z), \sum_{w \in X} \pi_r(w) \otimes \pi_s(w) \otimes \pi_t(w) \Big\rangle =$$

$$= n^2 \sum_{z \in X} \sum_{w \in X} \langle \pi_i(z), \pi_r(w) \rangle \langle \pi_j(z), \pi_s(w) \rangle \langle \pi_t(z), \pi_t(w) \rangle =$$

$$= \delta_{ir}\delta_{js} m_i m_j m_t \sum_{f=0}^{d} k_f c_i(f) c_j(f) c_t(f) =$$

$$= \delta_{ir}\delta_{js} n m_t q_{ij}^t,$$

что и требовалось доказать. ■

Лемма 2.15. *Для любого* $t \in [d]$ *имеет место*

$$W_1 \cap \mathrm{Ker}\,(p_t) = \langle \{e_{ij} - e_{ji} \mid (i,\ j) \in S_d,\ p_{ij}^t = 0\} \rangle,$$
$$W_1 \cap \mathrm{Ker}\,(p_l^*) = \langle \{e_{rs}^* - e_{sr}^* \mid (r,\ s) \in S_d,\ q_{rs}^t = 0\} \rangle. \qquad (2.16)$$

Доказательство. Следует непосредственно из (2.14)—(2.15). ■

Лемма 2.16. *При всех* $i,\ j,\ r,\ s,\ t \in [d]$ *имеют место равенства*

$$\langle p_t^*(e_{ij}),\ p_t^*(e_{rs}) \rangle = m_t \sum_{h=0}^{d} k_h p_{ir}^h p_{js}^h c_t(h), \qquad (2.17)$$

$$\langle p_t(e_{ij}^*),\ p_t(e_{rs}^*) \rangle = k_t \sum_{h=0}^{d} m_h q_{ir}^h q_{js}^h c_h(t). \qquad (2.18)$$

Доказательство. Чтобы получить (2.17), мы используем (2.11) и (1.14) и, вспоминая, что $\langle w,\ w' \rangle = \delta_{w,\,w'}$ $(w,\ w' \in X)$, получаем

$$\langle p_t^*(e_{ij}),\ p_t^*(e_{rs}) \rangle =$$
$$= \sum_{z \in X} \sum_{\substack{x \in X \\ (x,\ z) \in R_i}} \sum_{\substack{y \in X \\ (y,\ z) \in R_j}} \sum_{z' \in X} \sum_{\substack{x' \in X \\ (x',\ z') \in R_r}} \sum_{\substack{y' \in X \\ (y',\ z') \in R_s}} \langle x,\ x' \rangle \langle y,\ y' \rangle \times$$
$$\times \langle \pi_t(z),\ \pi_t(z') \rangle =$$
$$= \sum_{z \in X} \sum_{z' \in X} \sum_{\substack{x \in X \\ (x,\ z) \in R_i \\ (x,\ z') \in R_r}} \sum_{\substack{y \in X \\ (y,\ z) \in R_j \\ (y,\ z') \in R_s}} \langle \pi_t(z),\ \pi_t(z') \rangle =$$
$$= \sum_{z \in X} \sum_{h=0}^{d} \sum_{\substack{z' \in X \\ (z,\ z') \in R_h}} \langle p_{ir}^h p_{is}^h \pi_t(z),\ \pi_t(z') \rangle = m_t \sum_{h=0}^{d} k_h p_{ir}^h p_{js}^h c_t(h),$$

что и требовалось доказать. Чтобы установить (2.18), мы вспоминаем, что, согласно (1.11),

$$p_{ef}^t = k_e k_f n^{-1} \sum_{h=0}^{d} m_h c_h(e)\, c_h(f)\, c_h(t) \quad (e,\ f,\ t \in [d]),$$

и, используя (1.14) и (2.13), получаем

$$\langle p_t(e_{ij}^*),\ p_t(e_{rs}^*) \rangle =$$
$$= n^2 \sum_{z \in X} \sum_{x \in X} \sum_{\substack{y \in X \\ (x,\ y) \in R_t}} \langle \pi_i(x),\ \pi_i(z) \rangle \langle \pi_j(y),\ \pi_j(z) \rangle \langle \pi_r(x),\ \pi_r(z) \rangle \times$$
$$\times \langle \pi_s(y),\ \pi_s(z) \rangle =$$

$$= m_i m_j m_r m_s n^{-1} \sum_{e=0}^{d} \sum_{f=0}^{d} k_t p_{ef}^t c_i(e)\, c_j(f)\, c_r(e)\, c_s(f) =$$

$$= k_t \sum_{h=0}^{d} m_h c_h(t) \left(m_i m_r n^{-1} \sum_{e=0}^{d} k_e c_h(e)\, c_i(e)\, c_r(e) \right) \times$$

$$\times \left(m_j m_s n^{-1} \sum_{f=0}^{d} k_f c_h(f)\, c_j(f)\, c_s(f) \right) =$$

$$= k_t \sum_{h=0}^{d} m_h c_h(t)\, q_{ir}^h q_{js}^h,$$

что дает (2.18). ■

Следствие 2.17. *При всех* $i, j, r, s, t \in [d]$ *имеют место равенства*

$$\langle p_t^*(e_{ij} - e_{ji}),\ p_t^*(e_{0h} - e_{h0}) \rangle = 2k_h m_t p_{ij}^h (c_t(i) - c_t(j)),$$
$$\langle p_t(e_{ij}^* - e_{ji}^*),\ p_t(e_{0h}^* - e_{h0}^*) \rangle = 2m_h k_t q_{ij}^h (c_i(t) - c_j(t)).$$

В частности,

$$\langle p_t^*(e_{0j} - e_{j0}),\ p_t^*(e_{0h} - e_{h0}) \rangle = 2\delta_{jh} k_h m_t (1 - c_t(j)), \tag{2.19}$$

$$\langle p_t(e_{0j}^* - e_{j0}^*),\ p_t(e_{0h}^* - e_{h0}^*) \rangle = 2\delta_{jh} m_h k_t (1 - c_j(t)), \tag{2.20}$$

так что каждое из множеств $\{p_t^*(e_{0j} - e_{j0}) \mid j \in [d]_t^*\}$ *и* $\{p_t(e_{0j}^* - e_{j0}^*) \mid j \in [d]_t\}$ *состоит из ортогональных векторов. Заметим также, что из равенств* (2.19) и (2.20) *следует, что векторы* $p_t^*(e_{0j})$ *и* $p_t^*(e_{j0})$ *(соответственно* $p_t(e_{0j}^*)$ *и* $p_t(e_{j0}^*)$*) различны тогда и только тогда, когда* $j \in [d]_t^*$ *(соответственно* $j \in [d]_t$*).*

Доказательство. В силу (1.8) доказательство следует непосредственно из (2.17) и (2.18). ■

Теперь мы готовы доказать теорему 2.3, показывающую, что матрица $G(t)$ (соответственно $G(t)^*$) является матрицей Грама множества векторов, полученных из $p_t^*(H_1)$ (соответственно $p_t(H_1^*)$).

Доказательство теоремы 2.3. Мы рассмотрим только матрицу $G(t)$, так как в случае матрицы $G(t)^*$ рассуждения аналогичны. Пусть задано число $t \in [d]$ и F_t — подпространство пространства V_t^*, определенное следующим образом:

$$F_t = \langle \{p_t^*(e_{0j} - e_{j0}) \mid j \in [d]_t^*\} \rangle. \tag{2.21}$$

Для любых $(i, j) \in S_d$ обозначим через f_{ij} проекцию $p_t^*(e_{ij} - e_{ji})$ на ортогональное дополнение F_t в V_t^*. Это дает

$$f_{ij} = p_t^*(e_{ij} - e_{ji}) - \sum_{h \in [d]_t^*} \lambda_{ij}^h p_t^*(e_{0h} - e_{h0}), \tag{2.22}$$

где в силу ортогональности векторов из (2.21) и следствия 2.17 имеет место

$$\lambda_{ij}^h = \frac{\langle p_t^*(e_{ij} - e_{ji}),\ p_t^*(e_{0h} - e_{h0})\rangle}{\langle p_t^*(e_{0h} - e_{h0}),\ p_t^*(e_{0h} - e_{h0})\rangle} =$$

$$= \frac{p_{ij}^h(c_t(i) - c_t(j))}{1 - c_t(h)} \quad (h \in [d]_t^*,\ i,\ j \in [d]).$$

Используя (2.22), лемму 2.16 и следствие 2.17, можно теперь проверить, что

$$2m_t G(t)_{ij,\, i'j'} = \langle f_{ij},\ f_{i'j'}\rangle \quad ((i, j) \in S_d,\ (i', j') \in S_d).$$

Тем самым матрица $G(t)$ представлена в виде матрицы Грама и, следовательно, является положительно полуопределенной. Неравенства (2.1) следуют теперь из неотрицательности диагональных элементов матрицы $G(t)$. ■

Далее мы собираемся показать, что размерность образа W_1 при отображении $p_r p_t^*$ сужается самое большее до единицы в предельном случае, когда диаграмма пересечений D_r или диаграмма представлений D_t^* является лесом. Мы сделаем это, подбирая подмножества в H_1 и H_1^* так, чтобы получилось независимое множество в W_1, которое становится базисом для W_1 в упомянутых выше предельных случаях и на котором легко описать действие $p_r p_t^*$.

Лемма 2.18. *Для любого* $t \in [d]$ *векторы*

$$\{e_{0j} - e_{j0} \mid j \in [d]_t^*\} \cup \{e_{rs}^* - e_{sr}^* \mid (r, s) \in S_d,\ q_{rs}^t = 0\} \tag{2.23}$$

независимы. Векторы

$$\{e_{0s}^* - e_{s0}^* \mid s \in [d]_t\} \cup \{e_{ij} - e_{ji} \mid (i, j) \in S_d,\ p_{ij}^t = 0\} \tag{2.24}$$

также независимы.

Доказательство. Если векторы (2.23) зависимы, то, согласно (2.16), подпространство пространства W_1, образованное линейной оболочкой первого множества в (2.23), имеет нетривиальное пересечение с $\mathrm{Ker}(p_t^*)$. Тогда векторы $\{p_t^*(e_{0j} - e_{j0}) \mid j \in [d]_t^*\}$

не являются независимыми. Однако в силу следствия 2.17 эти векторы ортогональны, и, следовательно, независимость векторов (2.23) установлена. Аналогичным образом доказывается независимость векторов (2.24). ■

Для доказательства теоремы 2.5 нам необходима еще одна лемма.

Лемма 2.19. *Для любого* $t \in [d]$ *число ребер в* D_t *(соответственно в* D_t^**) не меньше* $|[d]_t|$ *(соответственно* $|[d]_t^*|$*), причем равенство достигается тогда и только тогда, когда* D_t *(соответственно* D_t^**) является лесом.*

Доказательство. Утверждение следует непосредственно из леммы 1.3 и хорошо известной теоремы о графах, состоящей в том, что неориентированного графа без петель с e ребрами, v вершинами и c компонентами справедливо неравенство $e \geqslant v - c$, которое становится равенством тогда и только тогда, когда граф является лесом (см. [3]). ■

Доказательство теоремы 2.5. 1)→2). Предположим, что D_*^t является лесом. В силу (2.16) число элементов S_d, которые не являются ребрами в D_t^*, равно $\dim(W_1 \cap \mathrm{Ker}(p_t^*))$, а в силу леммы 2.19 число элементов, которые являются ребрами, равно $|[d]_t^*|$. В частности, векторы (2.23) образуют базис для W_1. Зафиксируем теперь произвольное $h \in [d]$. Применяя лемму 2.15 к векторам (2.23), получаем, что образ W_1 при отображении $p_h p_h^*$ имеет размерность 1 или 0 в зависимости от того, принадлежит h множеству $[d]_t^*$ или нет. В частности, (2.6) означает, что вектор $p_h p_t^*(e_{ij} - e_{ji})$ пропорционален вектору $p_h p_t^*(e_{0h} - e_{h0})$ при всех $i, j \in [d]$. Так как ввиду (2.3)—(2.5) имеют место равенства

$$p_h p_t^*(e_{ij} - e_{ji}) = \sum_{x \in X} \sum_{\substack{y \in X \\ (x, y) \in R_h}} x \otimes y \otimes \pi_t(P_{ij}(x, y) - P_{ij}(x, y)), \tag{2.25}$$

$$p_h p_t^*(e_{0h} - e_{h0}) = \sum_{x \in X} \sum_{\substack{y \in X \\ (x, y) \in R_h}} x \otimes y \otimes \pi_t(x - y), \tag{2.26}$$

то мы заключаем, что 2) справедливо при всех $x, y \in X$, для которых $(x, y) \in R_h$. В силу произвольности выбора h отсюда следует 2).

2)→3). Выберем произвольные $x, y \in X$ и $i, j \in [d]$ и предположим, что $(x, y) \in R_h$ при некотором $h \in [d]$. Пусть λ есть

скаляр, такой что

$$\pi_t(P_{ij}(x, y) - P_{ji}(x, y)) = \lambda \pi_t(x - y). \tag{2.27}$$

Вычисляя скалярное произведение каждой из частей последнего равенства с $\pi_t(x)$, находим, что

$$p_{ij}^h(c_t(i) - c_t(j)) = \lambda(1 - c_t(h)).$$

В частности, либо $h \notin [d]_t^*$ (и в этом случае обе части равенства (2.27) равны нулю), либо λ определяется числами h, i и j, что позволяет нам написать $\lambda = \lambda_{ij}^h$. В силу (2.25) и (2.26) мы заключаем, что

$$p_h p_t^*(e_{ij} - e_{ji}) = \begin{cases} \lambda_{ij}^h p_h p_t^*(e_{0h} - e_{h0}), & \text{если } h \in [d]_t^*, \\ 0 & \text{в противном случае.} \end{cases}$$

Так как из (2.10) следует $p_h(e_{0h} - e_{h0}) = e_{0h} - e_{h0}$ для всех $h \in [d]$ и так как $\sum_{h \in [d]} p_h$ есть тождественное отображение, то

$$\begin{aligned} p_t(e_{ij} - e_{ji}) &= \sum_{h \in [d]} p_h p_t^*(e_{ij} - e_{ji}) = \\ &= \sum_{h \in [d]_t^*} p_h p_t^*(e_{ij} - e_{ji}) = \\ &= \sum_{h \in [d]_t^*} \lambda_{ij}^h p_h^*(e_{0h} - e_{h0}). \end{aligned}$$

В частности, образ W_1 при отображении p_t^* содержится в подпространстве F_t, определенном в (2.21). Согласно абзацу, следующему за (2.21), мы заключаем, что матрица $G(t)$ является нулевой.

3)$\rightarrow$1). Согласно доказательству теоремы 2.3, равенство $G(t) = 0$ означает, что образ W_1 при отображении p_t^* содержится в F_t. В силу (2.21) F_t имеет размерность $|[d]_t^*|$, поэтому $\dim(\mathrm{Ker}(p_t^*) \cap W_1)$, т. е. число векторов в правой части (2.16) не меньше $|H_1| - |[d]_t^*|$. Это означает, что D_t^* имеет не более $|[d]_t^*|$ ребер. По лемме 2.19 число ребер в D_t^* не меньше этого числа, что приводит к равенству и выводам о том, что D_t^* является лесом.

Последнее утверждение теоремы 2.5 следует из леммы 1.3 и того факта, что $\pi_t(x) = \pi_t(y)$ для произвольных x, y ($(x, y) \in R_h$) тогда и только тогда, когда $c_t(h) = 1$. ■

Доказательство теоремы 2.6 полностью аналогично, и мы его опускаем. В следующем разделе мы ограничимся рассмотре-

нием P-полиномиальных схем и покажем, что все диаграммы представлений, которые являются деревьями, в действительности являются путями. Это есть следствие теоремы 3.3.

3. P- И Q-ПОЛИНОМИАЛЬНЫЕ СХЕМЫ

В этом разделе мы усилим теорему 2.5 для случая P-полиномиальных схем. В теореме 3.3 мы покажем, что помимо 0-вершины имеется самое большее один лист в любой связной диаграмме представлений P-полиномиальной схемы. В нашем последнем результате, теореме 3.4, мы приводим короткое доказательство теоремы Леонарда [8], использующее матрицу $G(t)$ из определения 2.2. Сначала мы сделаем обзор некоторых фактов о P-полиномиальных схемах.

Определение 3.1. Схема Y называется *P-полиномиальной*, если некоторая ее диаграмма пересечений является путем, и Q-полиномиальной, если некоторая ее диаграмма представлений является путем (более широко известное, но эквивалентное определение см. в [1, стр. 159]). Заметим, что понятия P-полиномиальной схемы и *дистанционно-регулярного графа* эквивалентны.

Мы всегда будем предполагать, что в *P-полиномиальной* (соответственно Q-полиномиальной) схеме классы занумерованы таким образом, что D_1 есть путь (соответственно примитивные идемпотенты занумерованы таким образом, что D_1^* есть путь), в котором вершины $i-1$ и i являются смежными при всех i $(1 \leqslant i \leqslant d)$.

Определение 3.2. *Пусть* $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ является P-полиномиальной схемой. Для каждого $i \in [d]$ обозначим через c_i, a_i и b_i числа пересечений $p^i_{1,i-1}$, $p^i_{1,i}$ и $p^i_{1,i+1}$ соответственно. Заметим, что число b_0 есть в точности k_1, и мы в дальнейшем будем обозначать его просто через k. Согласно [2], константы $\{c_i, a_i, b_i \mid i \in [d]\}$ определяют все числа пересечений. Вот один из примеров, который нам понадобится:

$$p^i_{2,i-1} = c_2^{-1} c_i (a_i + a_{i-1} - a_1) \quad (1 \leqslant i \leqslant d). \tag{3.1}$$

В этом легко убедиться непосредственно или посмотреть в [14]. Нам также будут необходимы следующие известные результаты:

$$k = c_i + a_i + b_i \quad (i \in [d]), \tag{3.2}$$

$$k c_t(i)\, c_t(1) = c_i c_t(i-1) + a_i c_t(i) + b_i c_t(i+1) \quad (i, t \in [d]). \tag{3.3}$$

Заметим, что индукцией по i в (3.3) можно показать, что $c_t(i)$ и $c_t(i+1)$ не обращаются в нуль одновременно $(i, t \in [d])$ и, следовательно, $c_t(d) \neq 0$ при всех $t \in [d]$.

Теорема 3.3. *Пусть* $Y = (X, \{R_i\}, 0 \leqslant i \leqslant d)$ *есть P-полиномиальная схема и некоторая диаграмма представлений* D_t^* *является связной. Тогда* D_t^* *имеет самое большее один лист, не считая 0-вершины. В частности, если диаграмма* D_t^* *является деревом, то она является путем и, следовательно, схема Y Q-полиномиальна.*

Доказательство. Пусть f $(f \neq 0)$ является листом в D_t^*, и пусть h — единственная вершина, смежная f. В силу (1.6) мы имеем $nE_t \cdot E_f = q_t f_f E_f + q_{tf}^h E_h$, и, таким образом, (1.5) означает, что

$$q_t(i)\, q_f(i) = q f_{tf} q_f(i) + q_{tf}^h q_h(i) \quad (i \in [d]). \tag{3.4}$$

Чтобы упростить наши обозначения, положим $\alpha_i = C_\alpha(i) = q_\alpha(i)\, m_\alpha^{-1}$ при $\alpha = t$, f и h и также положим $a = q_{tf}^f m_t^{-1}$ и $b = q_{ht}^f m_t^{-1}$. Тогда (3.4) приводится к виду

$$t_i f_i = a f_i + b h_i \quad (i \in [d]) \tag{3.5}$$

или

$$h_i = b^{-1} f_i (t_i - a) \quad (i \in [d]). \tag{3.6}$$

Полагая $i = 0$ и 1 в (3.5), находим, что $b = 1 - a$ и

$$a = \frac{h_1 - f_1 t_1}{h_1 - f_1}. \tag{3.7}$$

Заменим a_i в (3.3) с помощью (3.2):

$$c_i(\alpha_{i-1} - \alpha_i) - B_i(a_i - \alpha_{i+1}) = k(\alpha_1 - 1)\, d_1$$
$$(i \in [d],\ \alpha = h,\ f \text{ или } t). \tag{3.8}$$

Полагая далее $i = 1$, $i = d$ и $\alpha = f$, $\alpha = t$ в (3.8), получаем

$$1 - f_1 - b_1(f_1 - f_2) = k(f_1 - 1) f_1, \tag{3.9}$$
$$c_d(f_{d-1} - f_d) = k(f_1 - 1) f_d, \tag{3.10}$$
$$1 - t_1 - b_1(t_1 - t_2) = k(t_1 - 1) t_1, \tag{3.11}$$
$$c_d(t_{d-1} - t_d) = k(t_1 - 1) t_d. \tag{3.12}$$

Полагая $\alpha = h$ и $i = 1$ в (3.8) и используя для всех, кроме одного, значений h_i равенство (3.6) с учетом того, что $b = 1 - a$,

получаем

$$k(h_1-1)(f_1t_1-f_1a)=$$
$$=-a(1-f_1-b_1f_1+b_1f_2)+1-f_1t_1-b_1f_1t_1+b_1f_2t_2.$$

Исключение обоих членов b_1f_2 с помощью (3.9), приведение членов и применение (3.7) дает

$$f_1(-t_1-b_1(t_1-t_2))+t_2(kf_1+1)(f_1-1)+1=$$
$$=kf_1(h_1t_1+f_1t_1-h_1-t_1).$$

Теперь вычислим первый член с помощью (3.11):

$$(1-t_2)(1-f_1)=kf_1(h_1t_1+f_1t_1-h_1-t_2f_1-t_1^2+t_2). \quad (3.13)$$

Далее, полагая $\alpha=h$ и $i=d$ в (3.8) и исключая h_{d-1} и h_d с помощью (3.6), получаем

$$-a(c_df_{d-1}-c_df_d)+c_df_{d-1}t_{d-1}-c_df_dt_d=k(h_1-1)(f_dt_d-f_da).$$

Исключение обоих членов c_df_{d-1} с помощью (3.10), приведение членов и применение (3.7) дает

$$f_dc_d(t_{d-1}-t_d)-f_dk(h_1t_d+f_1t_1-h_1-t_d-t_{d-1}f_1+t_{d-1})=0.$$

Вычисляя первый член с помощью (3.12), получаем

$$kf_d(f_1(t_1-t_{d-1})-h_1(1-t_d)+t_{d-1}-t_1t_d)=0.$$

По лемме 1.3 $t_d\neq 1$, а по замечанию, следующему за (3.3), $f_d\neq 0$. Следовательно,

$$h_1=\frac{f_1(t_1-t_{d-1})+t_{d-1}-t_1t_d}{1-t_d}.$$

Подставляя h_1 в (3.13), используя это равенство и проводя тождественные преобразования, получаем

$$(f_1-1)(kf_1(t_2t_d-t_1t_d-t_1t_{d-1}+t_{d-1}-t_2+t_1^2)+$$
$$+(1-t_2)(1-t_d))=0$$

Так как t_2 и t_d отличны от 1 по лемме 1.3, то это квадратное уравнение относительно f_1 не вырождается. В частности, имеется только одно возможное значение для f_1, кроме 1. Так как константы $\{C_i(1)\,|\,i\in[d]\}$ различны для любой P-полиномиальной схемы [1, с. 269], то f_1 однозначно определяет вершину f.

Доказательство теоремы 1.1. Если множество X представимо множеством единичных векторов, удовлетворяющих условиям (1.1)—(1.2), то их матрица Грама должна быть пропорциональна некоторому примитивному идемпотенту E_t алгебры

$\mathscr{A}(Y)$, и, следовательно, введенное векторное пространство U можно рассматривать в качестве V_t. Утверждение следует теперь из теоремы 2.5 и 3.3. ■

Теорема 3.4 (Леонард [8]). *Числа пересечений P- и Q-полиномиальной схемы Y определяются пятью параметрами. Если обозначить $C_i(i)$ $(i \in [d])$ через r_i, то числа пересечений схемы Y могут быть получены из k, a_2, r_1, r_2 и r_3 с помощью следующих соотношений:*

$$(r_1 - r_{i-1})(r_1 - r_{i+1}) = (1 - r_i)(r_2 - r_i) \quad (1 \leqslant i \leqslant d), \tag{3.14}$$

$$a_i \frac{(r_1 - r_{i-1})(r_i - r_{i+1})}{1 - r_i} =$$

$$= a_{i-1} \frac{(r_1 - r_i)(r_{i-2} - r_{i-1})}{1 - r_{i-1}} - kr_1^2 + kr_2 - r_2 + 1 \quad (1 \leqslant i \leqslant d), \tag{3.15}$$

$$c_i(r_{i-1} - r_{i+1}) = k(r_1 r_i - r_{i+1}) - a_i(r_i - r_{i+1}) \quad (1 \leqslant i \leqslant d), \tag{3.16}$$

$$b_i = k - a_i - c_i, \quad (1 \leqslant i \leqslant d). \tag{3.17}$$

Доказательство. Если Y является P- и Q-полиномиальной схемой, то, согласно [1, с. 269], $r_i \neq r_j$ при $i \neq j$ $(i, j \in [d])$. Поэтому если равенства (3.14)—(3.17) справедливы, то они действительно позволяют определить c_i, a_i, b_i $(i \in [d])$. Равенство (3.14) получается упрощением равенства $G(1)_{1,\, i+1,\, i-1} = 0$ $(1 \leqslant i \leqslant d-1)$. С помощью (3.1) и (3.2) равенство $G(1)_{1,\, i-1,\, 1,\, i} = 0$ $(i \in [d])$ сводится к

$$a_i\left(\frac{(r_1 - r_{i-1})(r_1 - r_i)}{1 - r_i} + r_i - r_2\right) +$$

$$+ a_{i-1}\left(\frac{(r_1 - r_{i-1})(r_1 - r_i)}{1 - r_{i-1}} + r_{i-1} - r_2\right) - a_1(r_1 - r_2) = 0$$

$$(1 \leqslant i \leqslant d). \tag{3.18}$$

Сочетая (3.2) и (3.3) при $i = 1$, получаем

$$a_1(r_1 - r_2) = kr_1^2 - kr_2 + r_2 - 1.$$

Исключая a_1 из (3.18), используя это равенство и упрощая коэффициенты при a_{i-1} и a_i с помощью (3.14), получаем (3.15). Равенство (3.16) есть просто равенство (3.3), в котором b_i заменено на $k - a_i - c_i$, а равенство (3.17) есть просто равенство (3.2). ■

Заметим, что уравнения, эквивалентные (3.16)—(3.19), были решены Баннаи и Ито [1, с. 206] при получении явных выражений для чисел пересечений P- и Q-полиномиальных схем.

ЛИТЕРАТУРА

1. Bannai E. and Ito T. Algebraic Combinatorics I: Association Schemes. Benjamin Cummings Lecture Note Series, Menlo Park, Cal., 1984. [Имеется перевод: Баннаи Э., Ито Т. Алгебраическая комбинаторика. Схемы отношений. — М.: Мир, 1987.]
2. Biggs N. Algebraic Graph Theory. Cambridge Univ. Press, Cambridge, 1974.
3. Bollobas B. Graph Theory, an Introductury Course. Graduate Texts in Mathematics Vol. 63, Springer-Verlag, New York, 1979.
4. Delsarte P. An algebraic approach to the association schemes of coding theory. Phillips Res. Rep. Suppl. 10 (1973). [Имеется перевод: Дельсарт Ф. Алгебраический подход к схемам отношений теории кодирования. — М.: Мир, 1976.]
5. Egawa Y. Characterization of $H(n, q)$ by the parameters. J. Combin. Theory Ser. A 31 (1981), 108—125.
6. Huang T. Ph. D. Thesis, The Ohio State University, 1985.
7. Lancaster P. Theory of Matrices. Academic Press, New York/London, 1969. [Имеется перевод: Ланкастер П. Теория матриц. М.: Наука, 1982.]
8. Leonard D. Parameters of association schemes that are both P- and Q-polynomial, J. Combin. Theory Ser. A 36 (1984), 355—363.
9. Neumaier A. A characterization of a class of distance-regular graphs. J. Reine Angew. Math., 357 (1985), 182—192.
10. Sloane N. An introduction to association schemes and coding theory, in Theory and Applications of Special Functions (R. A. Askey, Ed.), 225—260, Academic Press, New York, 1975.
11. Sprague A. Characterization of projective graphs. J. Combina Theory Ser. B 24 (1978), 294—300.
12. Stanton D. Some q-Krawtchouk polynomials on Chevalley groups. Amer. J. Math. 102 (1980), 625—662.
13. Terwilliger P. A class of distance-regular graphs with the Q-polynomial property, J. of Comb. Theory, 40B (1986), 213—223.
14. Terwilliger P. Distance — Regular Graphs and Generalizations, Ph. D. Thesis, University of Illinois, Urbana, 1982.
15. Terwilliger P. Root systems and the Johnson and Hamming graphs, European J. Comp., 8 (1987), 73—102.

Обзор по пороговым методам[1)]

П. К. Саху[2)], *С. Солтани*[3)], *А. К. С. Вонг*[4)], *И. С. Чень*[5)]

В области цифровой обработки изображений пороговая операция широко применяется при сегментации изображений. Ввиду ее широкой применимости и в других задачах обработки изображений в последние годы были разработаны многочисленные пороговые методы. В данной работе дан обзор пороговых методов и обновлены предыдущие обзоры по этой тематике, выполненные Уэшка (Comput. Vision, Graphics & Image Process., 7, 1978, 259—265) и Фу и Му (Pattern Recognition, 13, 1981, 3—16). Предпринята попытка оценить качество некоторых автоматических глобальных пороговых методов с помощью таких критериев, как меры однородности и показатель формы. Проведенные оценки основаны на некоторых изображениях, взятых из реального мира.

1. ВВЕДЕНИЕ

Пороговая операция является известным средством сегментации изображений. В этой статье представлен обзор ряда методов порогового типа (также известных как методы приведения к бинарному виду), включая как глобальные, так и локальные методы.

Ради удобства обсуждения глобальные методы в свою очередь разделяются на точечно-зависимые и регионально-зависимые. Некоторые глобальные пороговые методы исследованы подробно, с тем чтобы оценить их качество на основе данного множества тестовых изображений. В работе сделана попытка

[1)] P. K. Sahoo, S. Soltani and A. K. Wong. A survey of thresholding techniques. — Computer vision, graphics and image processing 41, 233—260, 1988.

[2, 3, 4)] Department of Systems Design Engineering, University of Waterloo, Waterloo, Canada.

[5)] Department of Electrical Engineering, Univ. of Waterloo, Waterloo, Canada.

оценить качество ряда глобальных методов, выбирая в качестве критериев показатель формы и меру однородности области. Процесс оценки осуществляется на множестве изображений, полученных в результате оцифровки фотографий портрета и естественных сцен. Хотя главной целью остается обзор двухуровневых пороговых методов, также рассмотрены и обобщения этих методов на случай решения задачи сегментации со многими порогами.

Пусть N — множество натуральных чисел, (x, y) — пространственные координаты точек цифрового изображения, $G = \{0, 1, \ldots, l-1\}$ — множество неотрицательных целых чисел, представляющих уровни яркости. Тогда изображение может быть определено как отображение f: $N \times N \to G$. Яркость (т. е. полутоновый уровень) элемента изображения с координатами (x, y) будет обозначаться как $f(x, y)$.

Пусть $t \in G$ — некоторый порог, а $B = \{b_0, b_1\}$ — пара бинарных уровней яркости и $b_0, b_1 \in G$. Результат применения пороговой операции к функции $f(\cdot, \cdot)$ (изображению) на уровне яркости t приводит к двузначной функции f_t: $N \times N \to B$, такой что

$$f_t(x, y) = \begin{cases} b_0, & \text{если } f(x, y) < t, \\ b_1, & \text{если } f(x, y) \geqslant t. \end{cases}$$

В общем случае пороговый метод — это такой метод, который определяет значение величины t на основе некоторого критерия. Если t^* задается только по уровню яркости каждого элемента изображения, то пороговый метод считается точечно-зависимым. Если же t^* определяется исходя из локального свойства (например, по локальному распределению яркостей) в окрестности каждого элемента изображения, то пороговый метод считается регионально-зависимым. Глобальным пороговым методом называется метод, в котором все изображение обрабатывается по одному значению порога. Локальным пороговым методом называется такой метод, в котором производится разбиение данного изображения на подызображения и определяется значение порога для каждого из этих подызображений.

Пусть n_i — число элементов изображения с уровнем яркости i. Тогда общее число элементов изображения равно

$$n = \sum_{i=0}^{l-1} n_i.$$

Вероятность встречаемости уровня яркости i задается величиной

$$p_i = n_i/n.$$

По соглашению будем считать, что уровень яркости 0 соответствует наиболее темному, а уровень яркости $l-1$ — наиболее светлому концам диапазона яркости.

2. ГЛОБАЛЬНЫЕ ПОРОГОВЫЕ ТОЧЕЧНО-ЗАВИСИМЫЕ МЕТОДЫ

A. *p-клеточный метод*

Одним из первых пороговых методов является p-клеточный метод [10]. В нем предполагается, что изображение состоит из темных объектов на светлом фоне. При условии, что известна доля площади объекта в процентах, порог задается как наибольший уровень яркости, при выборе которого по крайней мере $(100-p)\,\%$ элементов изображения отображаются в точки объектов на результирующем изображении. Например, пусть объект занимает 20 % изображения, тогда в качестве порога следует выбрать наибольший уровень яркости, такой что по крайней мере 20 % элементов изображения должны будут отобразиться в точки объекта. Ясно, что если площади объектов на изображении неизвестны, то данный метод неприменим.

B. *Метод мод*

Для изображений, состоящих из различных объектов и фона, их гистограммы яркостей должны быть бимодальными. В этом случае значение порога можно выбрать равным уровню яркости, соответствующему «впадине» гистограммы. Данный метод называется методом мод [31]. Хотя он и прост, он неприменим к изображениям с очень неравными пиками или широкими и пологими впадинами на гистограммах.

C. *Метод Осту*

Этот метод, предложенный в [27], основан на дискриминантном анализе. В нем пороговая операция рассматривается как способ разбиения элементов изображения на два класса C_0 и C_1 (например, на объекты и фон) на уровне яркости t. Таким образом, $C_0 = \{0, 1, \ldots, t\}$ и $C_1 = \{t+1, t+2, \ldots, l-1\}$. Пусть σ_W^2, σ_B^2 и σ_T^2 — соответственно дисперсия внутри класса, дисперсия между классами и полная дисперсия. Оптимальный порог можно определить исходя из минимизации одной из следующих (эквивалентных друг другу) функций-критериев относительно t:

$$\lambda = \frac{\sigma_B^2}{\sigma_W^2}, \quad \eta = \frac{\sigma_B^2}{\sigma_T^2} \quad \text{и} \quad \varkappa = \frac{\sigma_T^2}{\sigma_W^2}.$$

Среди приведенных трех функций η наиболее простая. В этом случае оптимальный порог — это

$$t^* = \operatorname{Arg} \min_{t \in G} \eta,$$

где

$$\sigma_T^2 = \sum_{i=0}^{l-1} (i - \mu_T)^2 p_i, \quad \mu_T = \sum_{i=0}^{l-1} i p_i,$$

$$\sigma_B^2 = \omega_0 \omega_1 (\mu_1 \mu_0)^2, \quad \omega_0 = \sum_{i=0}^{t} p_i, \quad \omega_1 = 1 - \omega_0,$$

$$\mu_1 = \frac{\mu_T - \mu_t}{1 - \omega_0}, \quad \mu_0 = \mu_t / \omega_0, \quad \mu_t = \sum_{i=0}^{t} i p_i.$$

D. *Метод анализа вогнутости гистограмм*

Для изображений, содержащих различные объекты и фоновые части, значение порога можно выбрать на гистограмме яркостей с помощью метода мод. Для некоторых изображений, гистограммы яркостей которых не имеют выраженных минимумов, часто удается задать хорошее значение порога на «хвосте» гистограммы. Поскольку как долины, так и хвосты соответствуют вогнутостям гистограммы, значение порога можно определить в результате анализа вогнутостей гистограммы [35].

Пусть HS — гистограмма, заданная на множестве уровней яркостей $g_0, g_1, \ldots, g_{l-1}$. Обозначим высоты гистограммы для указанных уровней яркостей через $h(g_0), h(g_1), \ldots, h(g_{l-1})$, где $h(g_i) \neq 0$ при всех i. Таким образом, HS можно рассматривать как двумерную область.

Для определения вогнутостей гистограммы HS строится ее выпуклая оболочка. Она представляет собой наименьший выпуклый многоугольник $\overline{HS}$, содержащий HS. Вогнутости HS выделяются по теоретико-множественной операции разности $HS - \overline{HS}$. Пусть $\bar{h}(g_i)$ — высота $\overline{HS}$ для уровня яркости g_i. Возможными пороговыми значениями являются уровни яркости, для которых $\bar{h}(g_i) - h(g_i)$ имеет локальный максимум. Однако не все из этих максимумов равноценны при выборе значения порога, так как большие вогнутости также могут возникать из-за шумовых выбросов. В работе [35] Розенфельда и Дела Торре эти вогнутости были названы ложными. Для устранения максимумов, порождаемых ложными вогнутостями, вводится мера равновесия

$$E_i = \left\{ \sum_{j=g_0}^{g_{i-1}} h(j) \right\} \cdot \left\{ \sum_{j=g_i}^{g_{l-1}} h(j) \right\}.$$

E_i измеряет меру сбалансированности гистограммы относительно уровня яркости g_i. Для ложных вогнутостей, которые обычно присутствуют лишь на одной стороне гистограммы, значения E_i будут малы. Таким образом, ложные вогнутости могут исключаться, если игнорировать максимумы $\bar{h} - h$ при малых E_i. Остальные максимумы указывают возможные значения порога, хотя они могут и не быть оптимальными. Другие уровни яркости, расположенные вблизи точек максимума, также могут рассматриваться для возможного улучшения выбора значений порога.

E. *Энтропийные методы*

В этих методах, разработанных в последние годы, гистограмма яркостей рассматривается как источник l символов. Оптимальное значение порога находится применением теории информации.

(*i*) *Методы Пуна*

В данном подразделе будут рассматриваться два алгоритма, недавно разработанные Пуном [32, 33].

Пусть t — значение порога. Определим две апостериорные энтропии [1]

$$H_b' = -\sum_{i=0}^{t} p_i \ln p_i,$$

$$H_w' = -\sum^{t} p_i \ln p_i,$$

где H_b' и H_w' могут соответственно трактоваться как меры апостериорной информации, связанной с черными и белыми элементами изображения после применения пороговой операции.

Исходя из априорной энтропии гистограммы яркостей, Пун [32] предложил алгоритм для определения оптимального значения порога путем максимизации верхней границы апостериорной энтропии

$$H' = H_b' + H_w'.$$

Пун [32] показал, что максимизация H' равносильна максимизации оценочной функции

$$f(t) = \frac{H_t}{H_T} \frac{\ln P_t}{\ln \max\{p_0, \ldots, p_t\}} + \left[1 - \frac{H_t}{H_T}\right] \frac{\ln (1 - P_t)}{\ln \max\{p_{t+1}, \ldots, p_{l-1}\}}$$

относительно t, где $H_t = -\sum_{i=0}^{t} p_i \ln p_i$,

$$H_T = -\sum_{i=0}^{l-1} p_i \ln p_i, \quad P_t = \sum_{i=0}^{t} p_i.$$

Во втором алгоритме Пун [33] предложил использовать в пороговой операции коэффициент анизотропии α, где

$$\alpha = \frac{\sum_{i=0}^{m} p_i \ln p_i}{\sum_{i=0}^{l-1} p_i \ln p_i}$$

и m — наименьшее целое число, такое что

$$\sum_{i=0}^{m} p_i \geqslant 0.5.$$

Оптимальное значение порога t^* выбирается по условию

$$\sum_{i=0}^{t^*} p_i = \begin{cases} 1-\alpha, & \text{если } \alpha \leqslant 0.5, \\ \alpha, & \text{если } \alpha > 0.5. \end{cases}$$

Однако Капур и др. [16] установили, что данный алгоритм всегда будет характеризоваться порогом $t^* \geqslant m$, что означает возникновение нежелательных смещений на изображении.

(ii) *Метод Капура, Саху и Вонга*

В этом методе [16] по первоначальному распределению уровней яркости изображения строятся два распределения вероятностей (например, распределение объекта и распределение фона) следующим образом:

$$\frac{p_0}{P_t}, \frac{p_1}{P_t}, \ldots, \frac{p_t}{P_t}$$

и

$$\frac{p_{t+1}}{1-P_t}, \frac{p_{t+2}}{1-P_t}, \ldots, \frac{p_{l-1}}{1-P_t},$$

где t — значение порога и $P_t = \sum_{i=0}^{t} p_i$. Определим

$$H_b(t) = -\sum_{i=0}^{t} p_i/P_t \ln (p_i/P_t),$$

$$H_w(t) = -\sum_{i=t+1}^{l-1} \frac{p_i}{1-P_t} \ln \frac{p_i}{1-P_t}.$$

Тогда оптимальное значение порога t^* определяется как значение яркости, которое максимизирует величину $H_b(t)+H_w(t)$, т. е.

$$t^* = \operatorname{Arg}\max_{t\in G}\{H_b(t)+H_w(t)\}.$$

Метод Йоханнсена и Билле

В данном методе [15] используется энтропия гистограммы яркостей цифрового изображения. В сущности он состоит в разбиении множества уровней яркости на два подмножества с тем условием, чтобы минимизировать взаимозависимость между ними (в теоретико-информационном смысле). Метод Йоханнсена и Билле предлагает выбор значения порога t^* по соотношению

$$t^* = \operatorname{Arg}\min_{t\in G}\{S(t)+\bar{S}(t)\},$$

где

$$S(t) = \ln\left(\sum_{i=0}^{t} p_i\right) - \frac{1}{\sum_{i=0}^{t} p_i}\left[p_t \ln p_t + \left(\sum_{i=0}^{t-1} p_i\right)\ln\left(\sum_{i=0}^{t-1} p_i\right)\right]$$

и

$$\bar{S}(t) = \ln\left(\sum_{i=t}^{l-1} p_i\right) - \frac{1}{\sum_{i=t}^{l-1} p_i}\left[p_t \ln p_t + \left(\sum_{i=t}^{l-1} p_i\right)\ln\left(\sum_{i=t+1}^{l-1} p_i\right)\right].$$

F. *Метод сохранения моментов*

В этом методе [40] значения порога вычисляются детерминированным способом, так чтобы моменты для изображения, подвергаемого пороговому преобразованию, сохранялись для выходного (бинарного) изображения. Момент m_i порядка i вычисляется по формуле

$$m_i = \frac{1}{n}\cdot\sum_{g=0}^{l-1} g^i\, h(g), \quad i = 1,\ 2,\ 3,$$

где n — общее число элементов изображения. Оптимальное значение t^* находится по гистограмме яркостей путем выбора t^* в качестве p_0-ячейки, где величина p_0 определяется соотношением

$$p_0 = \frac{z - m_1}{\left(c_1^2 - 4c_0\right)^{1/2}}$$

и

$$c_0 = \frac{m_1 m_3 - m_2^2}{m_2 - m_1}, \quad c_1 = \frac{m_1 m_2 - m_3}{m_2 - m_1^2}, \quad z = \frac{1}{2}\left\{(c_1^2 - 4c_0)^{\frac{1}{2}} - c_1\right\}.$$

Метод минимальной ошибки

В методе минимальной ошибки [19] гистограмма яркостей рассматривается как оценка плотности распределения вероятности $p(g)$ смеси совокупности, состоящей из уровней яркости элементов изображения, принадлежащих объекту и фону. Обычно предполагается, что каждая из двух компонент $p(g|i)$ смеси нормально распределена со средним значением μ_i и среднеквадратичным отклонением σ_i и априорной вероятностью P_i, так что

$$p(g) = \sum_{i=1}^{2} P_i p(g|i),$$

где

$$p(g|i) = \frac{1}{\sqrt{2\pi}\,\sigma_i} \exp\left(-\frac{(g-\mu_i)}{2\sigma_i^2}\right).$$

В качестве значения порога можно выбрать корень квадратного уравнения

$$\frac{(g-\mu_1)^2}{\sigma_1^2} + \ln\sigma_1^2 - 2\ln P_1 = \frac{(g-\mu_2^2)}{\sigma_2^2} + \ln\sigma_2^2 - 2\ln P_2.$$

Однако параметры μ_i, σ_i^2 и P_i $(i=1, 2)$ плотности смеси $p(g)$, связанной с изображением, к которому применяется пороговая операция, как правило, неизвестны. С целью преодоления трудностей при оценке этих неизвестных параметров Киттлер и Иллингворт [19] ввели в качестве критерия функцию $J(t)$,

$$J(t) = 1 + 2\{P_1(t)\ln\sigma_1(t) + P_2(t)\ln\sigma_2(t)\} - 2\{P_1(t)\ln P_1(t) + P_2(t)\ln P_2(t)\},$$

где

$$P_1(t) = \sum_{g=0}^{t} h(g), \quad P_2(t) = \sum_{g=t+1}^{l-1} h(g),$$

$$\mu_1(t) = \frac{\left\{\sum_{g=0}^{t} h(g)\,g\right\}}{P_1(t)}, \quad \mu_2(t) = \frac{\left\{\sum_{g=t+1}^{l-1} h(g)\,g\right\}}{P_2(t)},$$

$$\sigma_1^2(t) = \frac{\left\{\sum_{g=0}^{t} (g-\mu_1(t))^2 h(g)\right\}}{P_1(t)}$$

и

$$\sigma_2^2(t) = \left\{ \sum_{g=t+1}^{l-1} (g - \mu_2(t))^2 h(g) \right\} \Big/ P_2(t).$$

Оптимальное значение порога находится путем минимизации $J(t)$, т. е.

$$t^* = \operatorname{Arg} \min_{t \in G} J(t).$$

3. ГЛОБАЛЬНЫЕ ПОРОГОВЫЕ ОПЕРАЦИИ: РЕГИОНАЛЬНО-ЗАВИСИМЫЕ МЕТОДЫ

Описываемые ниже методы не предназначены для прямого выбора значения порога. Вместо этого они осуществляют преобразование гистограммы яркостей изображения, с тем чтобы получить гистограмму с более глубокими впадинами и более резкими пиками. Затем к ней можно применить метод мод, описанный в разделе 2В, для определения значения порога. Общей чертой, присущей всем указанным методам, является то, что новая гистограмма строится взвешиванием элементов изображения в соответствии с их локальным свойством. Кроме того, в них предполагается, что каждое из рассматриваемых изображений состоит из объектов и фона, причем оба имеют унимодальное распределение яркостей.

В работе [23] Мейсон и другие предложили использовать краевой оператор (например, лапласиан, перекрестный оператор Робертса и т. д.) для взвешивания элементов изображения. Согласно их методу, значения краевого оператора малы для элементов изображения в однородных областях и этим элементам следует приписать больший вес. Однако для элементов изображения, расположенных в окрестности краевого элемента, значения краевого оператора велики и указанным элементам назначается меньший вес. Новая гистограмма яркостей, полученная в результате выполнения данного процесса взвешивания, будет иметь более резкие пики и более глубокие впадины.

В 1965 г. Катц [17] отметил, что ввиду более высоких краевых значений для элементов изображения, расположенных вблизи края, гистограмма яркостей для этих элементов должна иметь единственный пик для уровня яркости, промежуточного между уровнями яркости объекта и фона. Тем самым этот уровень яркости может считаться подходящим значением для порога. Данный вопрос также исследовался Уэшкой и Розенфельдом [46]. Некоторые варианты указанного метода были рассмотрены в [45], [43, 49]. Уэшка и Розенфельд [48] объединили их в терминах диаграммы разброса краевых значений по уровням яркостей.

Метод квадродеревьев является еще одним методом преобразования гистограмм, привлекающим наше внимание [51]. Этот метод основан на том факте, что среднеквадратичное отклонение в однородной области мало, а в неоднородной области велико. Таким образом, области с большим среднеквадратичным отклонением могут быть разбиты на меньшие по размерам и однородные области. Исходя из входного изображения, метод квадродеревьев производит его разбиение на кванты при условии, что среднеквадратичное отклонение его уровней яркости превосходит заданный порог. Этот процесс повторяется для каждого кванта. Тем самым строится разбиение исходного изображения на блоки с меньшими среднеквадратичными отклонениями с приемлемой степенью аппроксимации уровня яркости каждого блока его средним уровнем яркости. Результирующее изображение называется Q-изображением. Благодаря однородности каждого блока гистограмма яркостей Q-изображения будет иметь более резко выраженные пики и впадины.

В. *Методы, основанные на статистиках второго порядка уровней яркости*

Один из недостатков точечно-зависимых пороговых методов состоит в том, что они целиком зависят только от статистик первого порядка уровней яркости (т. е. от гистограммы) изображения. Описываемые нами методы будут основаны на статистиках второго порядка уровней яркости изображения.

(i) *Метод матриц совместной встречаемости*

Матрица совместной встречаемости была введена Хараликом для анализа текстур [14]. В общем случае матрица совместной встречаемости $M_{d,\varphi}$ определяется как матрица, элементы которой суть относительные частоты встречаемости двух соседних элементов изображения с уровнями яркости i и j, находящихся на расстоянии d друг от друга и имеющих ориентацию φ. Ясно, что число таких матриц может оказаться довольно большим в зависимости от выбора d и φ. В работе [12] Ахуджа и Розенфельд [2] следующим образом определили матрицу совместной встречаемости:

$$M = M_{(1,\,0)} + M_{(1,\,\pi/2)} + M_{(1,\,\pi)} + M_{(1,\,3\pi/2)},$$

т. е. (i, j)-й элемент матрицы M равен частоте встречаемости уровня яркости i в 4-окрестности уровня яркости j [1]).

[1]) Имеется в виду 4-окрестность элемента изображения с уровнем яркости j. — *Прим. перев.*

Ввиду однородности элементы изображения, принадлежащие внутренней части объектов или фону, должны в основном учитываться в элементах матрицы M, размещенных вблизи ее диагонали. Элементы изображения, расположенные вблизи краев, должны вносить свой вклад в недиагональные элементы матрицы M, так как в окрестности края уровни яркости меняются. Поэтому матрица M может использоваться для задания двух новых гистограмм:

a) гистограммы, основанной на почти-диагональных элементах матрицы. Эта гистограмма должна иметь глубокую впадину, отделяющую уровни яркостей объекта и фона;

b) гистограммы, основанной на недиагональных элементах матрицы. Она должна иметь резкий пик между уровнями яркости объекта и фона.

Значение порога можно выбрать в интервале уровней яркости, в котором впадина в гистограмме a) перекрывается с пиком в гистограмме b).

(ii) *Метод диаграммы разброса* (*яркость, локальная средняя яркость*)

Данный метод [18] довольно схож с методом матриц совместной встречаемости, рассмотренным в предыдущем разделе. При задании диаграммы разброса (яркость, локальная средняя яркость) начало координат выбирается как точка верхнего левого угла, и уровень яркости возрастает в направлении слева направо, а средняя яркость, вычисляемая по окну, возрастает в направлении сверху вниз. Интенсивность точки на этой диаграмме пропорциональна частоте встречаемости соответствующей пары (яркость, локальная средняя яркость). В работе [18] Кирби и Розенфельд предложили вычислять локальные средние яркости по квадратному окну размера 3×3.

Почти диагональные элементы диаграммы разброса представляют элементы изображения, для которых локальные средние яркости близки к их уровням яркости. Такие элементы должны, по-видимому, принадлежать внутренней части объектов или фону ввиду свойства их однородности. Таким образом, гистограмма уровня яркости этих элементов должна иметь глубокую впадину. С другой стороны, недиагональные элементы диаграммы разброса соответствуют элементам изображения вблизи края, так что гистограмма уровня яркостей этих элементов должна иметь резкий пик. Так же как в методе матриц совместной встречаемости, значение порога может быть определено по интервалу уровней яркости, в котором впадина и пик указанных гистограмм перекрывают друг друга.

С. *Метод Дерави и Пала*

В этом методе [9] используются матрицы перехода, аналогичные матрицам совместной встречаемости, которые были рассмотрены в разд. 3В(i), с целью определения двух «мер взаимодействия» для выбора значения порога. Оптимальное значение порога находится путем минимизации этих мер взаимодействия.

С помощью обозначений из раздела 3В(i) матрицы перехода, вводимые в [9], могут быть записаны в виде

$$T_h = M_{(1,\ 0)},$$
$$T_V = M_{(1,\ 3\pi/2)},$$
$$T_{hv} = T_v + T_h.$$

Любая из приведенных матриц перехода может использоваться при выборе значения порога, и она в последующем будет обозначаться через T. В [9] отмечается, что результаты, полученные по различным матрицам перехода, имеют несущественные отличия.

Пусть $T_{i,j}$ есть (i, j)-й элемент матрицы T и t — порог, по которому множество уровней яркости разбивается на два класса: $C_0 = \{0, 1, \ldots, t\}$ и $C_1 = \{t+1, t+2, \ldots, l-1\}$. Тогда T можно разбить на 4 части, определяемые по следующим параметрам:

$$a = \sum_{i=0}^{t} \sum_{j=0}^{t} T_{ij}, \qquad b = \sum_{i=t+1}^{l-1} \sum_{j=t+1}^{l-1} T_{ij},$$
$$c = \sum_{i=0}^{t} \sum_{j=t+1}^{l-1} T_{ij}, \qquad d = \sum_{i=t+1}^{l-1} \sum_{j=0}^{t} T_{ij}.$$

Таким образом, a, b, c и d представляют соответственно общее число переходов в C_0, общее число переходов в C_1, общее число переходов из C_0 в C_1 и общее число переходов из C_1 в C_0. Совместная и условная вероятности переходов между C_0 и C_1 могут оцениваться выражениями

$$P_j(t) = \frac{c+d}{a+b+c+d}$$

и

$$P_c(t) = \frac{1}{2}\left\{\frac{c}{a+c} + \frac{d}{b+d}\right\}.$$

Дерави и Пал [9] назвали величины $P_j(t)$ и $P_c(t)$ мерами взаимодействия. Они также отметили, что $P_j(t)$ имеет сходство с «мерой занятости», введенной в [47], и что $P_c(t)$ непосредственно не зависит от гистограммы. Оптимальное значение t^* находится путем максимизации любой из указанных мер взаимодействия.

D. *Релаксационные методы*

Идея релаксации была предложена Саузвеллом [38, 39] для ускорения сходимости рекурсивного решения системы линейных уравнений. При сегментации изображений релаксация применяется следующим образом. Сначала элементы изображения классифицируются по вероятностному правилу на «светлые» и «темные» на основе их уровней яркости. Затем вероятность отнесения элемента изображения к классам корректируется в соответствии с вероятностями соседних с ним элементов изображения. Процесс корректировки повторяется с тем, чтобы вероятности отнесения к «светлым» (соответственно к «темным») стали очень велики для элементов, принадлежащих светлым (соответственно темным) областям. Замечательной особенностью релаксационных методов является то, что они осуществляют параллельную обработку данных в противоположность последовательным методам, рассмотренным нами до сих пор.

(i) *Начальная классификация*

Розенфельд и Смит [36] предложили следующий метод для начальной классификации элементов изображения. Пусть d и l — наиболее темный и наиболее светлый уровни яркости и g_i — уровень яркости элемента изображения x_i. Тогда для x_i положим

$$p^0_{i,\ \text{темн.}} = \frac{l - g_i}{l - d}$$

и

$$p^0_{i,\ \text{светл.}} = \frac{g_i - d}{l - d}.$$

Несмотря на простоту этого метода, в тех случаях, когда уровни яркости для объекта и фона не относятся к различным половинам гистограммы яркостей, он становится непригодным. Для устранения этой трудности Розенфельд и Смит [36] предложили другую схему инициализации. Пусть m — средний уровень яркости. Тогда, если $g_i > m$, положим

$$p^0_{i,\ \text{светл,}} = \frac{1}{2} + \frac{1}{2} \cdot \frac{g_i - m}{l - m},$$

если же $g_i \leqslant m$, то положим

$$p^0_{i,\ \text{темн}} = \frac{1}{2} + \frac{1}{2} \frac{m - g_i}{m - d}.$$

Фекете и др. [11] предложили подход, в котором предполагается возможность разбиения гистограммы на две гауссовы совокупности, так что распределение уровней яркости может

быть представлено в виде суммы двух гауссовых распределений. Параметры этих гауссовых распределений определяются с помощью метода, предложенного в [7]. Они установили, что данный метод обеспечивает большую скорость сходимости процесса релаксации.

(ii) *Итеративная корректировка вероятностей*

Как это уже упоминалось, процесс корректировки состоит в изменении вероятностей для каждого элемента изображения исходя из вероятностей для соседних элементов изображения. Пусть Λ — множество меток классов (например, классов, состоящих из светлых и темных элементов изображения). Тогда для управления процессом корректировки меток классов вводится коэффициент совместимости $r_{ij}(\lambda, \lambda')$ элемента изображения x_i с меткой $\lambda \in \Lambda$ с другим элементом изображения x_j с меткой

$$r_{ij}(\lambda, \lambda') = \begin{cases} -1, & \text{если } \lambda \text{ и } \lambda' \textit{ несовместимы}, \\ 0, & \text{если } x_i \text{ и } x_j \textit{ независимы}, \\ 1, & \text{если } \lambda \text{ и } \lambda' \textit{ совместимы}. \end{cases}$$

Цукер и др. [52] предложили следующую формулу для корректировки вероятностей:

$$p_i^{k+1}(\lambda) = \frac{p_i^k(\lambda)\left[1 + q_i^k(\lambda)\right]}{\sum\limits_{\lambda \in \Lambda} p_i^k(\lambda)\left[1 + q_i^k(\lambda)\right]},$$

$$q_i^k(\lambda) = \frac{1}{8} \sum_{x_j \in N_i} \sum_{\lambda' \in \Lambda} r_{ij}(\lambda, \lambda')\, p_j^k(\lambda'),$$

где N_i есть 8-окрестность элемента x_i. Однако Павлидис [29] показал, что $p_i^k(\lambda)$ всегда изменяются при корректировке, что противоречит естественному допущению о неизменности меток в случае, когда соседние элементы изображения независимы. Этот теоретический недостаток устранил Пелег [30], который предложил следующую формулу для корректировки:

$$p_i^{k+1}(\lambda) = \frac{p_i^k(\lambda) \sum\limits_{x_j \in N_i} \sum\limits_{\lambda' \in \Lambda} r_{ij}(\lambda, \lambda')\, p_j^k(\lambda')}{\sum\limits_{\lambda \in \Lambda} p_i^k(\lambda) \sum\limits_{x_j \in N_i} \sum\limits_{\lambda' \in \Lambda} r_{ij}(\lambda, \lambda')\, p_i^k(\lambda')}.$$

E. *Методы градиентной релаксации*

При градиентной релаксации схема оптимальной разметки определяется путем максимизации функции-критерия методом градиентной оптимизации.

Пусть λ_1 и λ_2 — метки классов светлых и темных элементов изображения, $\{[p_i(\lambda_1), p_i(\lambda_2)]^T,\ i = 0,\ 1,\ \ldots,\ l-1\}$ — множество векторов вероятностей, соответствующих уровням яркости, $([q_i(\lambda_1), q_i(\lambda_2)]^T, i = 0, 1, \ldots, 1-1\}$ — множество векторов совместимости, где $q_i(\lambda_k) = \frac{1}{8} \sum_{x_i \in N_i} p_j(\lambda_k)$, N_i — это 8-окрестность элемента изображения x_i. В работе [4] Бхану и Фегерас указали, что схема оптимальной разметки для изображений с унимодальным распределением уровней яркости может задаваться следующим уравнением:

$$\text{максимизировать } C(p) := \sum_{i=0}^{l-1} p_i^T(\lambda)\, q_i(\lambda)$$

при

$$p_i(\lambda) \in K := \left\{ p(\lambda) = (p(\lambda_1),\ p(\lambda_2)) \mid p(\lambda_j) \geqslant 0,\quad \sum_{j=1}^{2} p(\lambda_j) = 1 \right\}.$$

Они отметили, что максимизация $C(p)$ равносильна уменьшению степени несовместимости и неоднозначности при разметке элементов изображения.

Мы предложим аналогичный метод исходя из другой функции-критерия, выводимой из теории информации. В данном методе схема оптимальной разметки определяется путем максимизации функции

$$\Psi(p) := \sum_{i=0}^{l-1} I(p_i,\ q_i) \quad \text{при } p_i \in K.$$

Величина $I(p_i, q_i)$ задается в виде

$$I(p_i,\ q_i) = \sum_{j=1}^{l} p_i(\lambda_j) \ln\left(\frac{p_i(\lambda_i)}{q_i(\lambda_j)}\right).$$

F. *Другие методы*

Пороговые методы для оптических систем распознавания печатных знаков также вызвали большой интерес. Поскольку в одном документе встречаются разные искажения, часто приходится использовать комбинацию нескольких пороговых операторов. Каждый из этих операторов обладает чувствительностью к определенному типу искажений. Например, в работе Бартца [3] строится комбинация четырех линейных пороговых операторов для формирования одного порога. Примером этих операторов является оператор $T = kv + c$, где v — средний контраст по множеству знаков, просмотренных на предыдущем шаге,

k и c — оптимизирующие параметры. Вольф [50] предложил двухэтапную процедуру для решения задачи штриховки печатных знаков. На первом этапе вычисляется средний уровень яркости для каждого элемента изображения в пределах окна 4×4. Элемент изображения X считается частью знака, если его уровень яркости темнее, чем средние уровни яркости двух элементов изображения, ориентации которых отличаются на 180°

o	n	n	n	o
n	n	o	n	n
n	o	X	o	n
n	n	o	n	n
o	n	n	n	o

Рис. 1. (5×5)-окно, используемое в работе Ульмана [41].

и находятся на расстоянии 8 друг от друга. Второй этап аналогичен первому и отличается тем, что в нем используется окно больших размеров.

В работе Ульмана [4] выбор значения порога для элемента изображения X производится на основе уровней яркостей в пределах окна W размером 5×5 с центром на элемент X. При подборе значения порога учитывается вклад лишь тех элементов изображения, которые помечены меткой n (рис. 1).

Ниже приводятся два экспериментальных правила, которые используются при определении значения порога для X. Пусть n_X — наибольший (самый светлый) уровень яркости в пределах W. Если $n_X \leqslant 40$, то применяется правило 1, если же $n_X > 40$, то правило 2. Вот эти два правила.

1) Пометить элемент X как точку объекта, если для некоторой точки n из W выполняется $g(X) - g(n) < \tau$, где $g(\cdot)$ обозначает уровень яркости точки $(\cdot)$ и τ — некоторый заданный порог; в противном случае элемент X помечается как точка фона.

2) Пометить элемент X как точку объекта, если по крайней мере для одной точки n из W имеем $g(X) < g(n)/\mu$, где μ — некоторая фиксированная константа, в противном случае элемент разметить как точку фона.

Моррин [25] использовал графики зависимости градиента от уровней яркости с целью преобразования полутоновых изображений с высокой разрешающей способностью и контрастностью для задания значения порога. Панда [28] предложил метод, в котором порог для данной точки выбирается в зависимости от уровня яркости и краевого значения для этой точки. Некоторые его результаты и выводы изложены в [28].

4. ЛОКАЛЬНЫЕ ПОРОГОВЫЕ МЕТОДЫ

В локальных пороговых операциях исходное изображение разбивается на подызображения меньших размеров и для каждого из подызображений определяется свой порог. Их применение вызывает разрывы уровней яркости на границах двух различных подызображений на результирующем изображении. Значение порога для области можно задать либо с помощью точечно-зависимого метода, либо метода с региональной зависимостью. Для устранения возникающих разрывов применяется метод сглаживания.

Чоу и Канеко [7, 8] предложили использовать окно размера 7×7 для локальной пороговой операции. В методе, разработанном ими, исходное изображение разбивается на окна размера 7×7 и для каждого из подызображений вычисляется соответствующий порог. Однако если подызображение имеет унимодальную гистограмму яркости, то для него порог не вычисляется. Для таких подызображений значения порога интерполируются по соседним подызображениям. Для изображений с бимодальной гистограммой яркости порог вычисляется следующим образом. Сначала гистограмма яркости для подызображения аппроксимируется суммой двух гауссовых распределений, а затем значение порога находится путем минимизации ошибки классификации относительно значения порога. В [26] рассмотрен ряд экспериментов по применению данного метода.

Фернандо и Монро [12] разработали метод локального определения значения порога для ангиограмм (рентгенограмм кровеносных сосудов). Было установлено, что глобальные пороговые методы неприменимы для этих изображений, которые, как правило, имеют унимодальные гистограммы с очень узким пиком. Согласно этому новому методу, исходное изображение разбивается на 16 неперекрывающихся подызображений и к каждому из подызображений применяется метод энтропийного определения значения порога, который изложен в работе Пуна [32]. Наконец, изображение, полученное после пороговой операции, полностью обрабатывается низкочастотным фильтром для устранения разрывов уровней яркости на границах между подызображениями.

5. МНОГОПОРОГОВЫЕ МЕТОДЫ

Многие глобальные пороговые методы (такие как методы Осту [27], Пуна [32, 33], Капура и др. [16], сохранения моментов [40], минимальной ошибки [19]) могут быть обобщены на случай многих порогов. В данном разделе мы обсудим три метода выбора значений нескольких порогов, которые не вошли в предыдущие разделы данной работы.

А. *Метод сегментации амплитуд*

Этот метод был предложен Боукхаробой и др. [5]. В нем учитываются характерные свойства функции распределения изображения, к которому применяется пороговая операция. В этом методе информация о значениях порогов извлекается из анализа кривизны функции распределения. Функция распределения $F(k)$ в точке k задается в виде

$$F(k) = \frac{\sum_{g=0}^{k} h(g)}{\sum_{g=0}^{l-1} h(g)}.$$

Кривизна функции F определяется по соотношению

$$C(x) = F''(x)\,[1 + (F'(x))^2]^{-3/2},$$

где F' и F'' — первая и вторая производные функции F соответственно. В [5] отмечается, что $C(x)$ имеет зашумленный вид и колебательное поведение, и поэтому для ее использования в пороговом методе необходимо осуществить сглаживание и аппроксимацию $C(x)$. Точки, в которых $F(k)$ обращается в нуль, определяют значения порогов, а также уровни яркости, приписываемые каждому классу.

В. *Метод Ванга и Харалика*

Данный метод [42] представляет собой рекурсивный метод для выбора значений нескольких порогов на цифровых изображениях. В нем сначала производится классификация элементов изображения на краевые и некраевые элементы. Затем дается классификация краевых элементов на относительно темные или относительно светлые на основании информации об их окрестностях. Для элементов изображения, оказавшихся краевыми и относительно темными, строится гистограмма яркости. Для элементов изображения, также являющихся краевыми, но относительно светлыми, строится другая гистограмма яркости. Значение порога выбирается исходя из уровня яркости, соответствующего одному из наивысших пиков этих двух гистограмм. Для получения значения нескольких порогов данная процедура повторяется рекурсивно применительно лишь к тем элементам изображения, яркости которых выше значения порога.

С. *Метод однородного контраста*

Это предложенный Кохлером [21] метод рекурсивного выбора значения порога. Он основан на следуюшей идее. Оптимальное значение порога для сегментации изображения должно выделять больше резкоконтрастных и меньше слабоконтрастных краев по сравнению с любыми другими значениями [21]. В этом методе для каждого возможного порога t строится гистограмма среднего контраста $\mu(t)$, и ее наивысший пик соответствует оптимальному значению порога. Средний контраст вычисляется по соотношению

$$\mu(t) = \frac{C(t)}{N(t)},$$

причем $\mu(t) = 0$ при $N(t) = 0$ и $C(t)$ — полный контраст, обнаруживаемый по порогу t, $N(t)$ — число краев, выделяемых по t. Для выбора значений нескольких порогов сначала выбирается произвольное начальное значение порога, а затем формируется новая гистограмма $\mu(t)$ посредством исключения вклада уже выделенных по начальному значению порога краев. Эта процедура повторяется до тех пор, пока максимальный средний контраст для любого порога не окажется меньше некоторого минимального среднего контрастного параметра $\theta > 1$.

6. ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ

Следует отметить, что не все из рассмотренных нами методов могут считаться автоматическими (т. е. не требующими

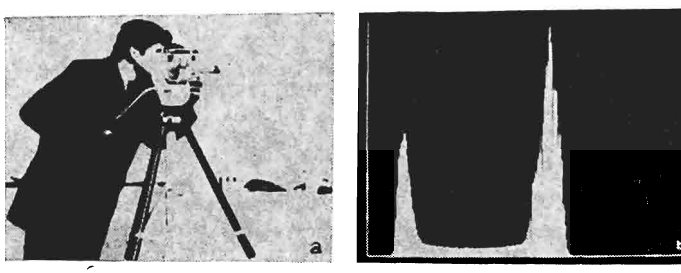


Рис. 2. *а*) оцифрованное изображение оператора, *b*) гистограмма изображения оператора.

вмешательства человека). Для некоторых необходима обратная связь с пользователем, поскольку однозначный выбор оптимального значения порога невозможен. При применении локальных пороговых методов изображение разбивается на подызображе-

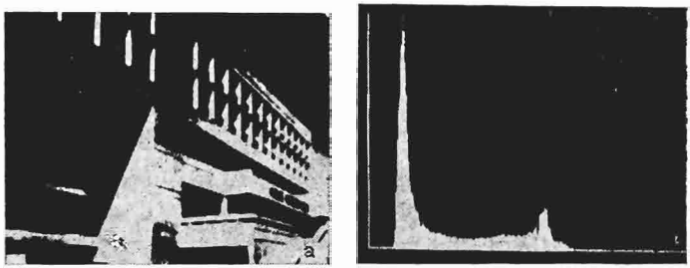


Рис. 3. *a*) оцифрованное изображение здания, *b*) гистограмма изображения здания.

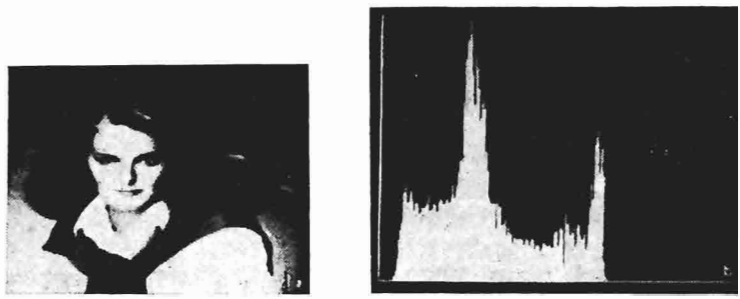


Рис. 4. *a*) оцифрованное изображение модели, *b*) гистограмма изображения модели.

Таблица 1. **Оптимальное значение порога для набора тестовых изображений**

Метод	Оптимальное значение порога		
	Оператор	Здание	Модель
матриц совместной встречаемости	63	63	63
Дерави и Пал	116		
анализа вогнутостей гистограмм	127	47	127
Иоханнсена и Билле	75	79	114
Капура и др.	123	69	81
минимальной ошибки	111	37	113
сохранения моментов	91	76	76
Осту	86	73	91
Пуна [32]	131	28	63

ния меньших размеров и для выбора оптимального значения порога для каждого подызображения применяется подходящий глобальный метод. По-видимому, глобальные методы часто используются для выбора значения порога. Поэтому мы намереваемся оценить некоторые из них, а именно методы Осту [27], Пуна [32], Йоханнсена и Билле [15], Капура и др. [16], метод матриц совместной встречаемости [2], метод анализа вогнутости гистограмм [35], метод, разработанный Дерави и Палом [9], метод сохранения моментов [40] и метод минимальной ошибки [19]. Указанные методы выбраны для сравнительного изучения, так как они являются автоматическими и в большинстве случаев не требуют субъективных суждений для нахождения наилучшего значения порога из множества оптимальных значений порога. Перечисленные методы были применены к трем снимкам: «оператор», «здание» и «модель». Снимок с оператором был оцифрован по его печатной копии в [32] с размером растра 415×395 и числом уровней яркости, равным 256. Оцифрованное

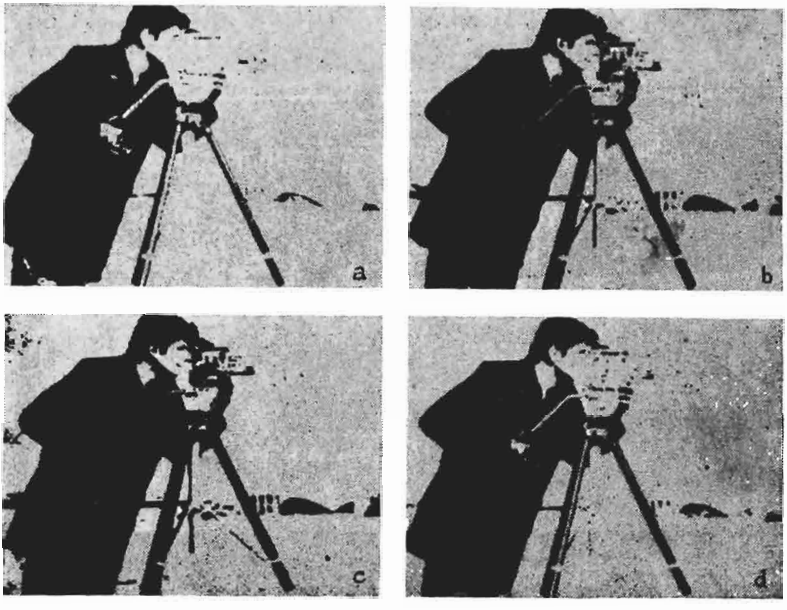


Рис. 5. Бинарные изображения оператора; *a*) метод матриц совместной встречаемости ($t^* = 63$), *b*) метод Дерави и Пала ($t^* = 116$), *c*) метод анализа вогнутостей гистограммы ($t^* = 127$), *d*) метод Йоханнсена и Билле ($t^* = 75$), *e*) метод Капура, Саху и Вонга ($t^* = 123$), *f*) метод минимальной ошибки ($t^* = 111$), *g*) метод сохранения моментов ($t^* = 91$), *h*) метод Осту ($t^* = 86$), *i*) метод Пуна ($t^* = 131$).

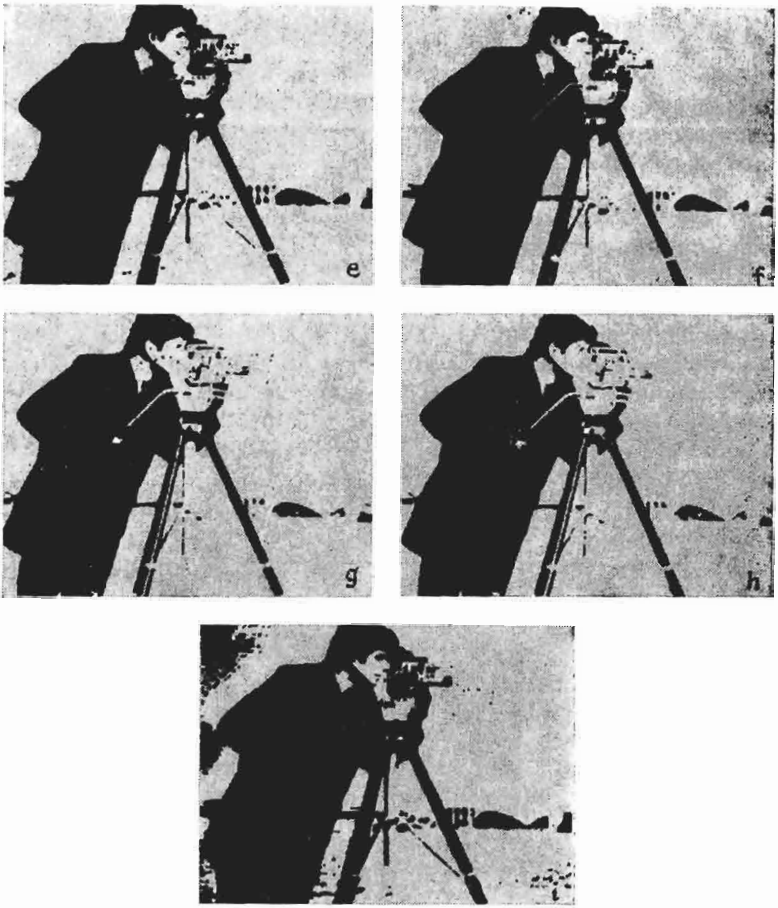


Рис. 5 (продолжение).

изображение и его гистограмма яркости показаны на рис. 2. Снимок здания также был оцифрован по его печатной копии из [32] с размером растра 411×403 и 256 уровнями яркости. Снимок модели оцифровывался с размером растра 321×314 и с 256 уровнями яркости. Изображения здания и модели приведены на рис. 3 и 4 вместе с их гистограммами. Значения порогов, полученных указанными методами, приведены в табл. 1.

Для метода матриц совместной встречаемости расстояние d между элементами изображения было выбрано равным 1, и 5 % элементов матрицы M, наиболее удаленные от ее диаго-

нали, были спроектированы на диагональ. В данном случае для всех трех изображений значение порога оказалось равным 63. Это значение не изменилось при использовании 10 % элементов вместо 5 %. Однако, когда величина d изменилась от 1 до 2 (соответственно до 3), значения порогов для изображений оператора и здания не изменились, но значение порога для изображения модели стало равным 30 (соответственно равным 33). Если принять процент элементов матрицы M, спроектированных на диагональ, равным 1 % и 2 % с сохранением значений d равными 2 и 3 соответственно, в обоих случаях были получены значения порога 63. Таким образом, при больших значениях d рекомендуется уменьшить процент элементов матрицы M, которые наиболее удалены от диагонали и отображаются на эту диагональ.

Метод Дерави и Пала [9] был реализован лишь применительно к изображению оператора. В их методе были выявлены два локальных минимума, один из которых в точке 45, другой — в точке 116. Поскольку пороговая операция при значении порога 45 не привела к лучшему изображению, в качестве оптимального значения порога было выбрано 116. На рис. 5—7 показаны бинарные изображения, полученные с помощью пороговых методов, рассмотренных в данном разделе.

7. МЕРЫ ДЛЯ ОЦЕНКИ ПОРОГОВЫХ МЕТОДОВ

Однородность и форма объектов на цифровых изображениях играют важную роль при отделении объектов от фона. Степень согласованности этих двух характеристик любого бинарного изображения с реальным изображением была оценена для трех тестовых изображений, рассмотренных в предыдущем разделе.

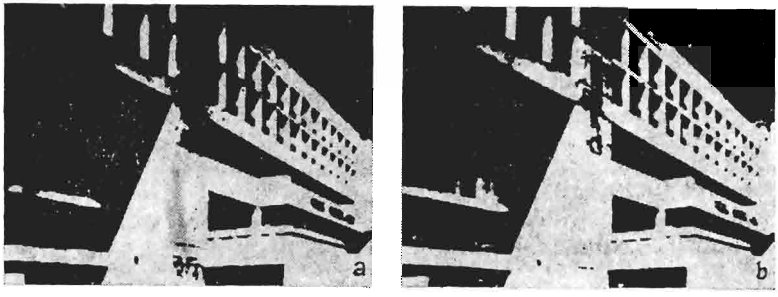


Рис. 6. Бинарные изображения здания: *a*) метод матриц совместной встречаемости ($t^* = 63$), *b*) метод анализа вогнутостей гистограмм ($t^* = 47$), *c*) метод Йоханнсена и Билле ($t^* = 79$), метод Капура, Саху и Вонга ($t^* = 69$), *e*) метод минимальной ошибки ($t^* = 37$), *f*) метод сохранения моментов ($t^* = 76$), *g*) метод Осту ($t^* = 73$), *h*) метод Пуна [32] ($t^* = 28$).

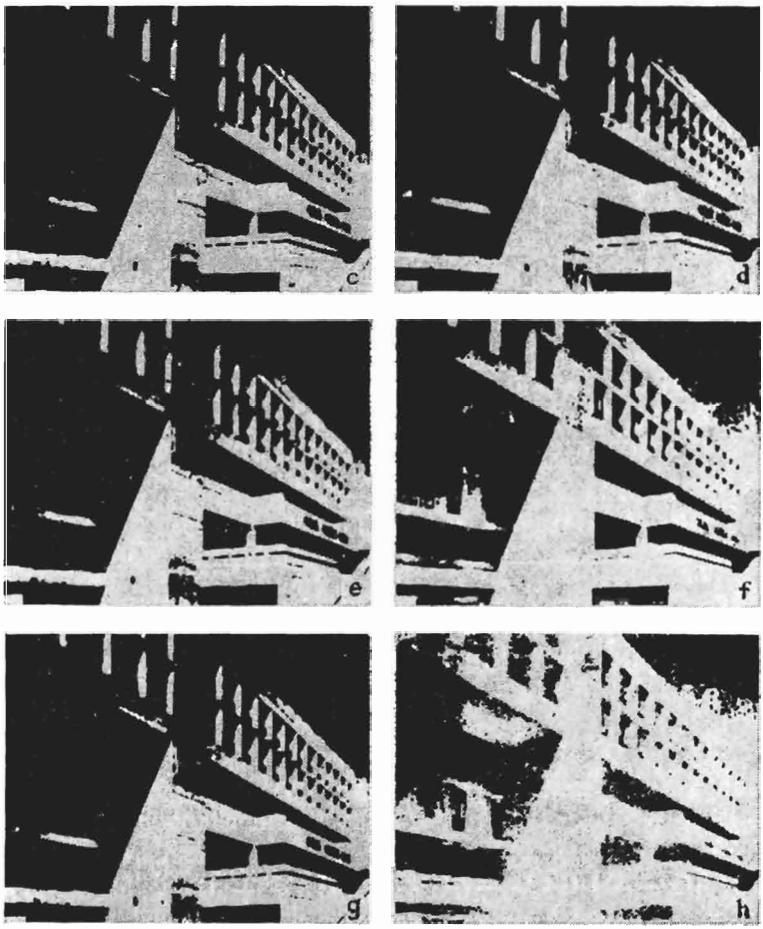


Рис. 6 (продолжение).

Мера однородности, выбранная нами, взята из работы Левина и Назифа [22]. *Мера однородности* $U(t)$ при заданном значении порога t определяется как

$$U(t) = 1 - \frac{\sigma_1^2 + \sigma_2^2}{C},$$

где

$$\sigma_i^2 = \sum_{(x,\, y) \in R_i} (f(x,\, y) - \mu_i)^2,$$

Рис. 7. Бинарные изображения модели: *a*) метод матриц совместной встречаемости ($t^* = 63$), *b*) метод анализа вогнутостей гистограмм ($t^* = 127$), *c*) метод Йоханнсена и Билле ($t^* = 114$), *d*) метод Капура, Саху и Вонга ($t^* = 81$), *e*) метод минимальной ошибки ($t^* = 113$), *f*) метод сохранения моментов ($t^* = 76$), *g*) метод Осту ($t^* = 91$).

Рис. 7 (продолжение).

R_i — сегментированная i-я область, $f(x, y)$ — уровень яркости элемента изображения (x, y), $\mu_i = \dfrac{\sum\limits_{(x, y) \in R_i} f(x, y)}{A_i}$, A_i — число элементов изображения в R_i, $i = 1, 2$, и C — нормирующий множитель. Для вычисления меры однородности U изображений, содержащих более двух областей, Левин и Назиф [22] включили в нее весовой множитель.

Мера формы S используется при количественной оценке формы объектов на тестовых изображениях. Мера S для данного изображения вычисляется следующим образом: (a) приписать каждому элементу изображения (x, y) обобщенное значение градиента $\Delta(x, y)$; (b) если уровень яркости элемента изображения (x, y) больше, чем средняя яркость соседних с ним элементов, то приписать знак «+» обобщенному значению градиента $\Delta(x, y)$, в противном случае приписать ему знак «—»; (c) вычислить меру формы S по формуле

$$S = \frac{\sum\limits_{(x, y)} \operatorname{sgn}\left(f(x, y) - \overline{f_{N(x, y)}}\right) \Delta(x, y) \operatorname{sgn}(f(x, y) - t)}{C},$$

где $\overline{f_{N(x, y)}}$ — средняя яркость в окрестности $N(x, y)$, t — значение порога для изображения, C — нормирующий множитель и

$$\operatorname{sgn}(x) = \begin{cases} 1, & \text{если } x \geqslant 0, \\ -1, & \text{если } x < 0. \end{cases}$$

Вычисление обобщенного значения градиента $\Delta(x, y)$ на элементе (x, y) проводится по формуле

$$\Delta(x, y) = \left[\sum_{k=1}^{4} D_k^2 + \sqrt{2} D_1 (D_3 + D_4) - \sqrt{2} D_2 (D_2 - D_4)\right]^{1/2},$$

где

$$D_1 = f(x + 1, y) - f(x - 1, y),$$
$$D_2 = f(x, y - 1) - f(x, y + 1),$$
$$D_3 = f(x + 1, y + 1) - f(x - 1, y - 1),$$
$$D_4 = f(x + 1, y - 1) - f(x - 1, y + 1).$$

Используя обе меры, можно получить оценки значения порога по каждому из перечисленных выше методов. В табл. 2a, 2b и 2c приведены результаты этих оценок. Заметим, что указанные значения перенормированы по наилучшему возможному значению порога для каждой меры (справа приведен ранг каждого метода).

Таблица 2a. **Результаты оценки для изображения оператора**

Метод	Порог	Однородность, U		Форма, S	
матриц совместной встречаемости	63	0.7121	5	0.7471	4
Дерави и Пала	116	0.6190	6	0.5790	6
анализа вогнутостей	127	0,3556	8	0.3387	8
Йоханнсена и Билле	75	0.9089	3	0.9452	3
Капура и др.	123	0.4775	7	0.4148	7
минимальной ошибки	111	0.7133	4	0.6951	5
сохранения моментов	91	0.9911	2	0.9839	2
Осту	86	0.9990	1	1.0000	1
Пуна	131	0.2205	9	0.2390	9

Таблица 2b. **Результаты оценки для изображения здания**

Метод	Порог	Однородность, U		Форма, S	
матриц совместной встречаемости	63	0.9098	5	0.8863	5
анализа вогнутостей гистограмм	47	0.5889	6	0.5597	6
Йоханнсена и Билле	79	0.9833	3	0.9990	**2**
Капура и др.	69	0.9792	4	0.9647	**4**
минимальной ошибки	37	0.3820	7	0.2987	7
сохранения моментов	76	0.9980	2	0.9995	1
Осту	73	0.9986	1	0.9931	3
Пуна [32]	28	0.1988	8	0.0131	8

Таблица 2c. **Результаты оценки для изображения модели**

Метод	Порог	Однородность, U		Форма, S	
матриц совместной встречаемости	63	0.5483	7	0.9597	1
анализа вогнутостей гистограмм	127	0.6246	6	0.2775	7
Йоханнсена и Билле	114	0.8334	5	0.4543	6
Капура и др.	81	0.9192	2	0.7636	3
минимальной ошибки	113	0.8465	3	0.4682	5
сохранения моментов	76	0.8445	**4**	0.7872	2
Осту	91	0.9986	1	0.7418	4
Пуна [32]	63	0.5483	7	0.9597	1

8. ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ ОЦЕНКИ

Анализируя табл. 2a, мы приходим к выводу, что методы Осту, сохранения моментов и Йоханнсена и Билле являются более предпочтительными методами выбора значения порога

для снимка оператора по обеим мерам: однородности области и формы. Для бимодальных изображений (см. табл. 3) оптимальные значения мер однородности и формы незначительно отличаются друг от друга. Таким образом, если для снимка оператора какой-то метод окажется приемлемым, то он будет таким же приемлемым по мере формы. Поэтому упомянутые методы

Таблица 3. **Оптимальное значение порога для мер однородности и формы**

Мера	Оператор	Здание	Модель
Однородность	87	74	93
Форма	86	78	57

в табл. 2a имеют одинаковый ранг как относительно меры однородности, так и относительно меры формы. В целом результаты оценки на этом изображении для указанных методов почти не отличаются.

Снимок здания не имеет четкой бимодальной гистограммы яркости. Таким образом, для него в отличие от снимка оператора метод Осту (см. табл. 2b) имеет ранг, равный трем, относительно меры формы и ранг, равный 1, относительно меры однородности. Метод сохранения моментов имеет ранг 2 относительно меры однородности и ранг 1 относительно меры формы. Методы Йоханнсена и Билле, Капура и др. уступают лишь методам Осту и сохранения моментов.

Отметим, что снимок модели не имеет бимодальной гистограммы. Для него (см. табл. 2c) метод Осту имеет ранг 1 относительно меры однородности и ранг 4 относительно меры формы. Метод матриц совместной встречаемости имеет ранг 1 по мере формы и ранг 7 по мере однородности. Метод Капура и др. оказался вторым по рангу относительно меры однородности и третьим по рангу относительно меры формы. Данный результат свидетельствует о том, что метод Капура и др. хорошо согласуется с обеими мерами для данного типа изображений. Метод матриц совместной встречаемости в этом смысле ориентирован на меру формы, а метод Осту — на меру однородности. Таким образом, для снимка модели, не имеющего бимодальной гистограммы, метод матриц совместной встречаемости можно считать приемлемым лишь для меры формы, в то время как метод Осту является хорошим лишь для меры однородности.

По результатам этой объективной оценки (табл. 2a, 2b и 2c) можно сделать вывод о том, что метод Осту, который основан

на дискриминантном анализе, можно считать одним из лучших пороговых методов, несмотря на многие его недостатки (см. [20, 34]). Поскольку наши меры оценки порогового метода основаны только на свойствах однородности и формы, вывод о преимуществе метода Осту не является неожиданным. Метод сохранения моментов оказывается сопоставимым с методом Осту. Характеристики таких методов, как метод Йоханнсена и Билле, Капура и др., почти не уступают характеристикам метода Осту и методов сохранения моментов. Бинарные изображения, полученные с помощью пороговых методов (перечисленных в табл. 1), являются источником ценной информации о пороговых методах. Мы учитываем эту зрительную информацию в качестве дополнительных данных. Для зрительного анализа представляется разумным рассматривать такие важные признаки, как черты лица и камеру на снимке оператора, края здания на правой части снимка здания, характерные черты лица и волос на изображении модели. Вместе с этими наблюдениями необходимо также принять во внимание степень искажения и потерю информации на изображении, полученном в результате применения к нему пороговой операции. При пристальном рассмотрении бинарных изображений оператора на рис. 5 можно прийти к выводу, что такие методы, как метод Пуна (рис. 5i), метод анализа вогнутостей гистограмм (рис. 5c) и метод совместной встречаемости (рис. 5a), не дают хороших значений порога для этого изображения. Этот результат можно усмотреть также из табл. 2a. Также отметим, что метод матриц совместной встречаемости приводит к бинарному изображению, на котором черты лица и камера оператора оказываются потерянными, в то время как метод Пуна и методы анализа вогнутостей гистограмм дают бинарные изображения с искажениями. И метод Осту, и метод сохранения моментов не сохраняют черты лица на изображении оператора. Эти детали и другая ценная информация сохраняются при применении метода Капура и др. и метода минимальной ошибки. Отметим, что метод Капура и др. имеет ранг 7 в табл. 2a относительно меры однородности, так же как и относительно меры формы, однако он позволяет получить бинарное изображение, на котором сохраняются детали лица оператора.

Аналогичный анализ изображений на рис. 6 показывает, что методы сохранения моментов (рис. 6f), матриц совместной встречаемости (рис. 6a), Осту (рис. 6h), Капура и др. (рис. 6d), Йоханнсена и Билле (рис. 6c) дают достаточно хорошие значения порога для снимка здания. Среди перечисленных методов методы Йоханнсена и Билле и сохранения моментов дают хорошие бинарные изображения. Применительно к бинарным

изображениям на рис. 7 отметим, что метод Капура и др. и метод сохранения моментов являются более предпочтительными при выборе значения порога. Для этого же изображения метод Осту также дает хорошее значение порога. Однако он уступает двум упомянутым методам.

Таким образом, как по визуальной оценке бинарных изображений, так и по их объективной оценке, основанной на мерах

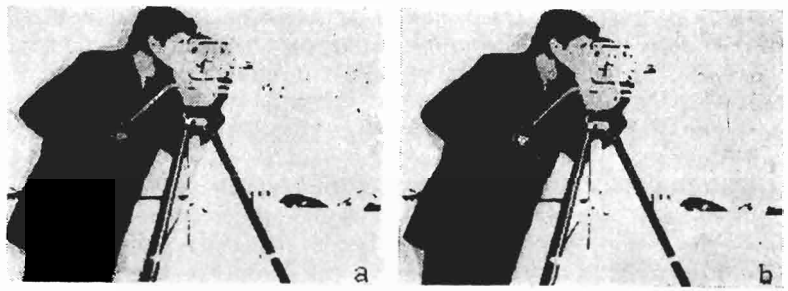


Рис. 8. Бинарные изображения оператора: *a*) оптимальная однородность ($t^* = 87$), *b*) оптимальная форма ($t^* = 86$).

однородности и формы, мы находим, что методы Йоханнсена и Билле, Капура и др., сохранения моментов, Осту являются приемлемыми пороговыми методами. В большинстве случаев методы Пуна, анализа вогнутости гистограмм и минимальной ошибки не отличаются хорошим качеством по сравнению с другими методами, рассмотренными в данном разделе. В одной из ранних работ [12] Фернандо и Монро указали на неудовлетворительное качество метода Пуна [32].

9. КОММЕНТАРИИ И ЗАКЛЮЧЕНИЕ

В данной работе при оценке автоматического порогового метода были использованы меры формы и однородности. В табл. 3 приведены значения уровней яркости, которые оптимизируют меры однородности и формы для тестовых изображений. Набор цифровых тестовых изображений, преобразованных к бинарному виду при уровнях яркости, доставляющих величинам S и U максимумы, показан на рис. 8, 9 и 10. По этим изображениям можно сделать вывод о пригодности указанных мер для получения бинарных изображений пороговым преобразованием.

В целом мы дали обзор различных пороговых методов и оценили качество некоторых глобальных методов, являющихся автоматическими по своему строению. Все описанные нами ме-

тоды оптимизируют некоторые функции-критерии и дают соответствующие обоснования для подобной оптимизации. Мы исследовали лишь связь оптимальных значений с соответствующими мерами однородности и формы по данному множеству

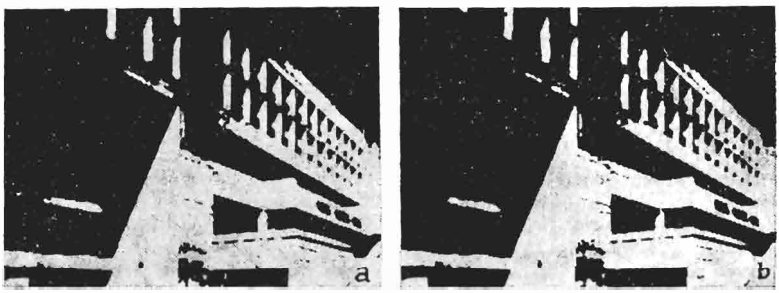


Рис. 9. Бинарные изображения здания: *a*) оптимальная однородность ($t^* = 74$), *b*) оптимальная форма ($t^* = 78$).

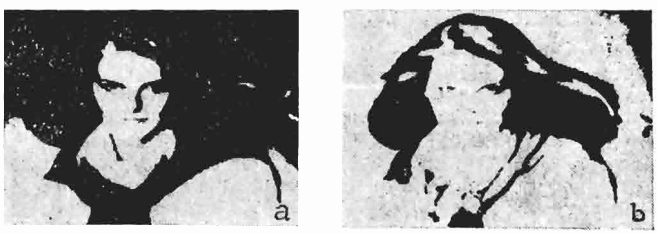


Рис. 10. Бинарные изображения модели: *a*) оптимальная однородность ($t^* = 93$), *b*) оптимальная форма ($t^* = 57$).

тестовых изображений. По результатам этого исследования мы пришли к заключению, что для указанного множества тестовых изображений методы Йоханнсена и Билле, Капура и др., метод Цэй сохранения моментов и метод Осту можно считать приемлемыми пороговыми методами, если к бинарному изображению предъявляются требования, выражаемые в терминах большей однородности и лучшей формы объекта на бинарном изображении. Однако остается открытым вопрос о качестве указанных методов, если рассматривать общий случай.

БЛАГОДАРНОСТЬ

Мы выражаем признательность рецензенту за помощь в улучшении вида статьи.

ЛИТЕРАТУРА

[1] N. Abramson, Information Theory and Coding, McGraw-Hill, New York, 1963.
[2] N. Ahuja and A. Rosenfeld, A note on the use of second-order gray-level statistics for threshold selection, IEEE Trans. Systems Man Cybernet. SMC-8, 1978, 895—899.
[3] M. R. Bartz, Optimizing a video processor for OCR, in Proceedings International Joint Conference on AI, 1969, pp. 79—90.
[4] B. Bhanu and O. Faugeras, Segmentation of images having inimodal distributions, IEEE Trans. Pallern Anal. Mach. Intell. PAMI-4, 1982, 408—419.
[5] S. Baukharouba, J. M. Rebordao, P. L. Wendel, An amplitude segmentation method based on the distribution function of an image, Comput. Vision Craphics Image Process, 29, 1985, 47—59.
[6] H. Bunke, H. Feistl, H. Heimann, G. Sagerer, F. Wolf, G. Zhou, Smoothing, thresholding, and contour extraction in images from gated blood pool studies, in First IEEE Symp. on Medical Imaging and Image Interpretation, Berlin, 1982.
[7] C. K. Chow and Kaneko, Boundary detection of radiographic images by a threshold method, in Proceedings, IFIP Congress 71, pp. 130—134.
[8] C. K. Chow and T. Kaneko, Automatic boundary detection of left ventricle from cineangiograms, comput. Biomed. Res. 5, 1972, 338—410.
[9] F. Deravi and S. K. Pal, Gray level thresholding using second-order statistics, Patten, Recognit. Lett. 1, 1983, 417—422.
[10] W. Doyle, Operation useful for similarity-invariant pattern recognition, J. Assoc. Comput. Mach. 9, 1962, 259—267.
[11] G. Fekete, J. O. Eklundh and A. Rosenfeld, Relaxation: Evaluation and applications, IEEE Trans. Pattern, Anal. Mach. Intell PAMI-3 1981, 460—469.
[12] S. M. X. Fernando and D. M. Monro, Variable thresholding applied to angiography, in Proceedings 6th International conference on Pattern Recognition, 1982.
[13] S. K. Fu and J. K. Mu, A survey on image segmentation, Pattern Recognit. 13, 1981, 3—16.
[14] R. M. Haralick, K. Shanmugam, and I. Dinstein, Texture features for image classification, IEEE Trans. Systems Man Cybernet. SMC-3, 1973, 610—621.
[15] G. Johannsen and J. Bille, A threshold selection method using information measures in Proceedings, 6th Int. Conf. Pattern Recognition, Munich, Germany, 1982, pp. 140—143.
[16] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Comput. ision Graphies Image Process, 29, 1985, 273—285.
[17] Y. H. Katz, Pattern recognition of meteorological sattelite cloud photography, Proceedings, Third Symp. on Remote Sensing of Environment, 1965, pp. 173—214.
[18] R. L. Kirby and A. Rosenfeld, A note on the use of (gray level local average gray level) space as on aid in thresholding selection, IEEE Trans. Systems Man Cybernet, SMC-9, 1979, 860—864.
[19] J. Kittler and J. Illingworth, Minimum error thresholding, Pattern Recognit, 19, 1986, 41—47.
[20] J. Kittler and J. Illingworth, On threshold selection using clustering criterion, IEEE Trans. Systems Man Cybernet, SMC-15, 625—653.
[21] R. Kohler, A segmentation system based on thresholding, Comput. Graphics Image Process. 15 1981 319—338.

[22] M. D. Levine and A. M. Nazif, Dynamic measurement of computer generated image segmentations, IEEE Trans. Pattern. Anal. Mach. Intell. PAMI-7, 1985, 155—164.
[23] D. Mason, I. J. Lauder, D. Rutoritz, and G. Spowart, Measurement of C-Bands in human chromosomes, comput. Biol. Med. 5, 1975, 179—201.
[24] D. L. Milgram, Region extraction using convergent evidence, Comput. Graphics Image Process 11, 1979, 1—12.
[25] T. H. Morrin, A black-white representation of a gray-salepicture, IEEE Trans. Comput. 23, 1974, 184—186.
[26] Y. Hakagawa and A. Rosenfeld, Some experiments on variable thresholding, Pattern Recognit. 11, 1979, 191—204.
[27] H. Ostu, A threshold selection method from gray-level histogram, IEEE Trans. Systems Man Cybernet. SMC-8, 1978, 62—66.
[28] D. P. Panda, Segmentation of FLIR Image by Pixel Classification, TR-508, University of Maryland Computer Science Center, 1977.
[29] Pavlidis, Structural Pattern Recognition, Springer-Verlag, New York, 1977.
[30] S. Peleg, A new probabilistic relaxation scheme, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2, 1980, 362—369.
[31] J. M. S. Prewitt and M. L. Mendelsohn, The analysis of cell: image, in Ann. New York, Acad. Sci. Vol. 128, pp. 1035—1053, New York Acad. Sci., New York, 1966.
[32] T. Pun, A new method for gray-level picture thresholding using the entropy of the histogram, Signal Process, 2, 1980, 223—237.
[33] T. Pun, Entropie thresholding: A new approach, comput. Vision Graphics Image Process, 16, 1981, 210—239.
[34] S. S. Reddi, S. F. Rudin, and H. R. Keshavan, An optimal multiple threshold scheme for image segmentation, IEEE Trans. Systems Man Cybernet. SMC-14, 1984, 661—665.
[35] A. Rosenfeld and P. De La Torre, Histogram concavity analysis as an aid in threshold selection, IEEE Trans. Systems Man Cybernet, SMC-13, 1983, 231—235.
[36] A. Rosenfeld and R. C. Smith, Thresholding using relaxation, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-3, 1981, 598—606.
[37] P. K. Sahoo, Theory and Applications of Some Measure of Uncertainties, Ph. D. thesis Department of Applied Mathematics, University of Waterloo, Waterloo, 1986.
[38] R. Southwell, Relaxation Methods in Engineering Science, A Treatise on Approximate computation, Oxford Univ. Press, London, 1940.
[39] R. Southwell, Relaxation Methods in Theoretical Physics, Oxford Univ. Press (Clarendon), London, 1946.
[40] W. Tsai, Moment-preserving thresholding: A new approach comput Vision Graphics Image Process. 29, 1985, 377—393.
[41] J. R. Ullman, Binarization using associative addressing, Pattern Recognit, 6, 1974, 127—135.
[42] S. Wang and R. M. Haralick, Automatie multithreshold selection, Comput. Vision Graphics Image Process, 25, 1984, 46—67.
[43] S. Watanabe, and the CYBEST group. An automated apparatus for cancer processing CYBEST, Comput. Vision Graphics Image Process, 3, 1974, 350—358.
[44] J. S. Weszka, A survey of threshold selection techniques, comput. Vision Graphics Image Process, 7, 1978, 259—265.
[45] J. S. Weszka, R. H. Hagel, and A. Rosenfeld, A threshold selection technique, IEEE Trans. Comput. C-23, 1974, 1322—1326.
[46] J. S. Weszka and A. Rosenfeld, Threshold Selection 4, TR-336, University of Maryland Computer Science Center, 1974.

[47] J. S. Weszka and A. Rosenfeld, Threshold evaluation techniques, IEEE Trans. Systems Man Cybernet. SMC-8, 1978, 622—629.
[48] J. S. Weszka and A. Rosenfeld, Histogram modification for threshold selection, IEEE Trans. Systems Man Cybernet, SMC-9, 1979, 38—51.
[49] J. S. Weszka, J. A. Vertuon and A. Rosenfeld, A Technique for Facilitating Threshold Selection for Objects Extraction from Digital Pictures, TR-243, University of Maryland Computer Science Center, 1973.
[50] R. H. Wolfe, A dynamic thresholding scheme for quantization of scanned image, in Proceedings, Automatic Pattern Recognition, 1969, pp. 143—162.
[51] A. Y. Wu and A. Rosenfeld, Threshold selection using quadtree, IEEE Trans. Pattern Anal. Mach. Intel. PAMI-4, 1982, 90—94.
[52] S. Zucker, R. Hummel and A. Rosenfeld, An Application of relaxation labelling to line and curve enhancement, IEEE Trans. Comput. C-26, 1977, 394—403.

Комбинаторные задачи статистической физики[1])

Ж. Вьенно[2])

Резюме. Введение предпочтительного направления в модели статистической механики изменяет критические показатели. Именно это объясняет происходящее в настоящее время возрастание интереса физиков к *ориентированным моделям* и, в частности, с 1982 г. — к *ориентированным фигурим*[3]) [8—25]. Именно они и составляют основной предмет этого обзора.

Точные результаты для размерности $d = 2$ (число фигур, средняя ширина и др.) получили Деррида, Хаким, Надаль и Ваннименю [17, 21], а для размерности $d = 2, 3$ — Дхар [12, 13]. Последний показал эквивалентность недавно решенной Бакстером модели *жестких шестиугольников* [30, 1, глава 14] и модели газа.

С другой стороны, автором в [26, 27] и автором совместно с Гуйю — Бошаном в [28] показано, что современные биективные методы перечислительной комбинаторики позволяют получать практически без каких-либо вычислений точные результаты для ориентированных фигур в рамках обычного подхода статистической механики.

Многие вопросы остались открытыми, в частности вопросы, касающиеся удивительного появления широко известных тождеств Роджерса — Рамануджана в модели жестких шестиугольников.

§ 1. ВВЕДЕНИЕ

Будут рассматриваться модели, используемые для объяснения явлений фазовых переходов: кипения воды, намагничивания ферромагнетиков и т. д. Предполагается заданной *решетка*,

---

[1]) Viennot G. Problémes combinatoires posés par la physique statistique. Seminaire N. Bourbaki, Astérisque N. 121—122 (1985), 225—246.

[2]) U. E. R. de Mathématiques et d' Informatique, Université de Bordeaux I, 351, Course de la Libération, 333405 Talence Cedex (France).

[3]) В оригинале — animaux (звери). — *Прим. ред.*

т. е. граф, вершины которого суть элементы $\mathbb{Z}^d$. Этот граф бесконечен и обладает достаточной «регулярностью». В этом обзоре мы рассмотрим *квадратные*, *треугольные* и *шестиугольные* решетки для $d = 2$ (см. соответственно рис. 1, 2 и 9), *кубические*

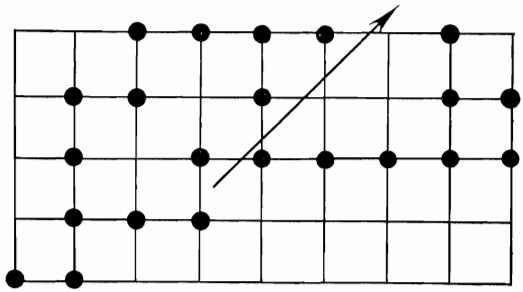


Рис. 1. Ориентированная фигура, имеющая единственный точечный источник.

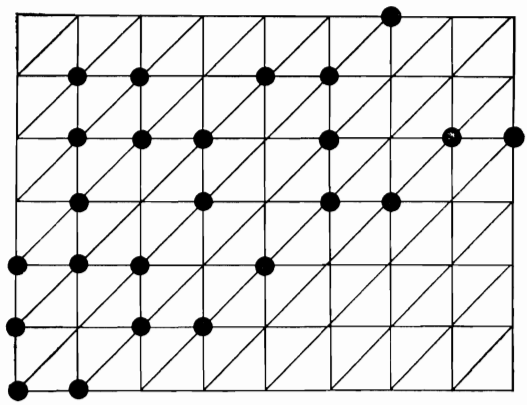


Рис. 2. Фигура на треугольной решетке.

и *кубоцентрические* — для $d = 3$ (см. § 8). Две вершины, связанные между собой ребром, называются *смежными*.

Каждой вершине i соответствует переменная σ_i, которая может принимать случайным образом конечное число q значений. Например, $\sigma_i = \pm 1$, если речь идет о спине, или, например, $\sigma_i = 1$ или $\sigma_i = 0$ в зависимости от того, «занята» вершина i (атомом) или «свободна». (Важнейшей функцией является статистическая сумма[1]).) Она выражается через энергию взаимодействия.

[1]) В оригинале — fonction de partition (функция разбиения).— *Прим. ред.*

Взаимодействие между вершинами i и j есть функция переменных σ_i и σ_j и температуры T. Иногда могут учитываться другие виды энергии (внешнее магнитное поле и т. д.). Дифференцируя статистическую сумму, получают обычно связываемые с системой физические величины (свободная энергия, плотность, восприимчивость, теплоемкость и т. д.).

Методология состоит в выборе подходящей идеализации для энергии взаимодействия, с тем чтобы сделать возможным точное вычисление статистической суммы. Обычно ограничиваются взаимодействием между ближайшими соседями. Наиболее известной является модель Изинга. Со времени решения этой модели Онзагером в 1944 г. (для $d = 2$ и при отсутствии внешнего магнитного поля) появилась серьезная литература о разрешимых моделях. Прекрасный обзор ее представляет собой недавно вышедшая книга Бакстера [1].

Несмотря на то что размерность этих моделей не превышает двух, они представляют интерес для понимания таких реальных процессов, как фазовые переходы. При фазовом переходе (т. е. при критической температуре T_C) физические величины имеют особенность. Поведение величины $F(T)$ в окрестности T_C определяется *критическим показателем* λ. Например, когда $T \to T_C$, может иметь место соотношение $F(T) \sim (T - T_C)^{-\lambda}$. (Обозначение $A \sim B$ имеет обычный для физики смысл, т. е. $\log A / \log B$ стремится к 1.) Вычислению критических показателей физики придают большое значение.

Пусть D — конечная связная область рассматриваемой решетки, содержащая N вершин. Пусть $Z_D(T)$ — статистическая сумма связанной с ней конечной системы. В действительности последняя есть полином, перечисляющий q^N возможных конфигураций определенных параметров. Пример такого полинома приведен в § 8 для модели жестких шестиугольников. Для модели Изинга задача состоит в перечислении 2^N конфигураций $\sigma_1, \ldots, \sigma_N = \pm 1$ на конечном прямоугольнике квадратной решетки при заданном числе пар смежных вершин i и j, таких что $\sigma_i \sigma_j = -1$. Важное значение имеет вычисление предела величины $Z_D^{1/N}$ при условии, что D становится (в определенном смысле) бесконечно большой. Этот предел $Z(T)$ называется *термодинамическим пределом* статистической суммы или приведенной статистической суммой, приходящейся на одну ячейку.

Связь между комбинаторикой и статистической механикой не нова. Например, давно известны комбинаторные решения модели Изинга [5]. Некоторые модели приводят вновь к известным объектам теории графов (хроматический полином, полином Татта, см. [6]).

Новым в комбинаторной постановке задачи об ориентированных фигурах является использование методов современной перечислительной комбинаторики, а именно упоминавшихся уже биективных методов. Начиная примерно с 70-х годов под влиянием различных школ, а именно Шютценберже во Франции и Рота в США, наступает расцвет того раздела комбинаторики, который в разделе 05A журнала Mathematical Reviews называется classical combinatorial problems (классические задачи комбинаторики).

Здесь речь идет не только о перечислении, но также, например, и об интерпретации некоторых известных в классических разделах математики тождеств путем построения взаимно однозначного соответствия между двумя конечными структурами, являющимися объектами изучения. Возникают многочисленные связи с приложениями (статистики, информатика и др.), а также с другими областями математики (например, симметрические функции и ортогональные многочлены). Более полное представление читатель может получить, прочитав статью Картье [58] или [54].

Постепенно основные идеи прояснялись, и появлялись комбинаторные «модели», позволяющие интерпретировать семейства тождеств в целом. Излагаемая ниже комбинаторика ориентированных фигур привела к трем следующим моделям: *коммутационный моноид*[1]) Картье и Фоата [38] в модернизированной (и более наглядной) версии — *укладок плиток* — Дюлюка, Вьенно [40]; *композиция разбиений* Фоата, Шютценберже [43, 46] (или *роды структур* Жуайяля [50]), связанная с экспоненциальными производящими рядами; *взвешенные пути Моцкина* в комбинаторной теории производящих ортогональных многочленов и исследованиях непрерывных дробей Флажоле [42] и Вьенно [55].

§ 2. ОРИЕНТИРОВАННЫЕ ФИГУРЫ

Обозначим $\Pi = \mathbb{Z} \times \mathbb{Z}$ множество вершин квадратной решетки. *Путь* на Π есть последовательность $\omega = (s_0, \ldots, s_n)$ вершин s_i из Π. Путь *ведет* из s_0 в s_n, его *длина* есть n, и (s_i, s_{i+1}) есть *элементарный шаг*. *Ориентированный путь* — это путь, элементарные шаги которого направлены либо на север, либо на восток, т. е. если $s_i = (x, y)$, то соответственно $s_{i+1} = (x, y+1)$ или $s_{i+1} = (x+1, y)$.

Ориентированная фигура (на квадратной решетке) есть конечное подмножество $A \subset \Pi$, содержащее непустое множество

[1]) В оригинале «monoide de commutation». Иногда используется название «частично коммутативный моноид» [1*]. — *Прим. ред.*

S точек, расположенных на прямой, перпендикулярной главной диагонали, и такое, что для всякой точки $\alpha \in A$ существует ориентированный путь, образованный вершинами фигуры и ведущий из некоторой точки S к α. Множество S, очевидно, единственно, и его элементы будем называть *точечными источниками*. Направление северо-восток называется физиками *предпочтительным направлением* (см. рис. 1, 3, 5).

Ширина (соответственно *длина*) ориентированной фигуры A есть минимум величины $|b-a|$ для всех целых a и b, таких что A располагается между прямыми, заданными уравнениями $y=x+a$, $y=x+b$ (соответственно $y=-x+a$, $y=-x+b$). При перечислении фигуры рассматриваются с точностью до сдвига. Пусть a_n — число ориентированных фигур размера (или мощности) $|A|=n$ с единственным точечным источником. Обозначим l_n (соответственно L_n) среднюю ширину (соответственно длину) фигур в предположении, что они распределены равномерно. Физики считают, что при $n\to\infty$ имеют место соотношения следующего типа:

$$(1)\ a_n \sim \mu^n n^{-\theta};\quad (2)\ l_n \sim n^{\nu_\perp};\quad (3)\ L_n \sim n^{\nu_\parallel}.$$

Задача состоит в вычислении показателей θ, $\nu_\perp$, $\nu_\parallel$ (и доказательстве их существования!). Вопрос, собственно говоря, не в описании термодинамической модели, но в том, чтобы представить модель как систему такого типа. Производящий ряд $f(t)=\sum_{n\geqslant 1} a_n t^n$ играет роль термодинамической функции с одной или несколькими вещественными критическими точками. Поведение $f(t)$ в окрестности одной из них, обозначенной t_C (наименьшей по абсолютной величине), позволяет (вообще говоря) вывести соотношение вида (1) и найти величины θ и $\mu=1/t_C$. Показатели θ, $\nu_\perp$ и $\nu_\parallel$ играют роль критических показателей. Они и соответствуют некоторым показателям термодинамических моделей. Кроме того, как будет видно в § 8, $-f(-t)$ есть в точности плотность в газовой модели Бакстера. Ориентированные фигуры описывают также модель, называемую *развлеченными полимерами* в движущихся растворах. Здесь имеется связь с ветвящимися марковскими процессами. Наконец, ориентированные фигуры близки к моделям *ориентированного просачивания* (см. [7]).

Заметим, что классическое просачивание соответствует обычным (неориентированным) *фигурам*. Для этих фигур понятие направленного пути заменяется понятием пути, который может иметь четыре типа элементарных шагов: на север, юг, запад, восток. Такие фигуры известны в комбинаторике под названием

полимино (каждая вершина замещается элементарным квадратом двоичной решетки). Однако пока еще мы слишком далеки от знания точных формул для таких изотропных фигур.

Введение предпочтительного направления изменяет *класс универсальности* модели (т. е. ее критические показатели). Более того, для ориентированного просачивания, как и для ориентированных фигур, два критических показателя $\nu_\perp$ и $\nu_\parallel$ заменяют показатель ν, характеризующий неориентированные модели.

Можно также определить ориентированные фигуры на треугольной решетке. Эту решетку можно рассматривать как квадратную решетку, которая дополнена диагоналями элементарных квадратов, параллельными главной диагонали. Тогда возможны ориентированные шаги трех типов: на север, на восток и шаг на северо-восток, т. е. $s_i = (x, y)$, $s_{i+1} = (x+1, y+1)$. Можно также обобщить понятие ориентированной фигуры на $\mathbb{Z}^d$. Относительно $d = 3$ отсылаем к § 8. Физики считают, что ориентированные фигуры на квадратных и треугольных решетках имеют одни и те же критические показатели: это положение известно как гипотеза универсальности. Класс универсальности зависит от размерности модели. Напротив, число μ в формуле (1), называемое *константой связи*, зависит от типа выбранной решетки.

§ 3. ФИЗИЧЕСКИЕ МЕТОДЫ ДЛЯ ОРИЕНТИРОВАННЫХ ФИГУР

Опишем коротко различные приближенные методы вычисления критических показателей ориентированных фигур, используемые физиками с 1982 г. Именно они представляют хороший набор сильных методов, обычно используемых в статистической физике, таких как, например, методы *ренормгруппы*. Заметим, что получающиеся результаты более или менее приближенные. Выписывание точной формулы для критических показателей еще не означает строгого доказательства. Например, для модели связей на плоской квадратной решетке значение 1/2 для порога просачивания было принято физиками в 1963 г., тогда как строгое доказательство (Кастена) получено не ранее 1980 г. Для ориентированных фигур известны два различных метода, дающие $\nu_\parallel = 0.8185 \pm 0.0010$ (Надаль и др. [21]) и $\nu_\parallel = 0.800 \pm \pm 0.001$ (Реднер и Янг [24]), тогда как строгого доказательства существования показателя $\nu_\parallel$ (хотя с тех пор Реднер и Янг улучшили это значение) все еще нет. Мы проводим различие между «доказать» и «показать».

Теория поля позволяет построить теорию *среднего поля* для ориентированных фигур, обосновываемую введением *предельной*

критической размерности d_C, *возмущения* критических показателей относительно $\varepsilon = d_C - d$, а также законов *скейлинга*. При $d \geqslant d_C$ критические показатели не зависят от d. Так, для ориентированных фигур без «*внутренних циклов*», т. е. таких фигур, в которых никакие два различных ориентированных пути фигуры не ведут в одну точку, Дэй и Любенский [11] дают $d_C = 7$ и $\varepsilon = 7 - d$. Эти фигуры оказываются объектами того же класса универсальности, что и (некоторые другие) ориентированные фигуры. Следовательно, $\nu_\perp = 1/4 + \varepsilon/36$, $\nu_\parallel = 1/2 + \varepsilon/24$ и $\theta = 3/2 - \varepsilon/12$.

Брейер и Янсен [9], а также Карди [10] показали эквивалентность модели ориентированных фигур размерности d и модели Ли — Янга размерности $d - 1$. Последняя представляет собою модель Изинга с мнимым магнитным полем, время рассматривается как предпочтительное направление. Он построил подмножество моделей, именуемых *критическими динамическими моделями*. Эта эквивалентность дает закон скейлинга $\theta = (d-1)\nu_\perp$ (см. также Фэмили [15]) и показывает, что для размерности два $\theta = \nu_\perp = 1/2$. Критический динамический показатель z характеризуется отношением $z = \nu_\parallel/\nu_\perp$. Его нахождение сопряжено с большими трудностями, теми же, что и для размерности 1. Отметим, что аналогичная эквивалентность, согласно Паризи и Сурла [36], имеет место между неориентированными фигурами размерности d и моделью Ли — Янга размерности $d - 1$. Это показывает, что критический показатель $\nu_\perp$ для ориентированных фигур размерности $d \geqslant 2$ тот же самый, что и показатель ν для неориентированных фигур размерности $d + 1$.

Указанная аппроксимация, называемая *аппроксимацией Флори*, была сначала предложена для *линейных полимеров* (т. е. несамопересекающихся путей). К ориентированному случаю она была приспособлена Любенским, Ваннименю [19] и Реднером, Конильо [23]. Ими найдены предельная критическая размерность $d_C = 7$ и для $d \geqslant d_C$ — показатели среднего поля, а именно $\nu_\perp = 1/4$ и $\nu_\parallel = 1/2$.

Перечисление фигур на *дереве Кели* (которое можно рассматривать как предел решетки $\mathbb{Z}^d$ при $d \to \infty$) снова приводит к тем же показателям среднего поля, см. Надаль [20].

Заметим, что фигуры на дихотомическом дереве являются ничем иным, как известными в комбинаторике *бинарными деревьями*. Их число есть классическое *число Каталана* $C_n = \frac{1}{n+1}\binom{2n}{n}$, имеющее производящий ряд $\frac{1}{2t}\left[1 - (1-4t)^{1/2}\right]$ и асимптотику $C_n \sim 4^n n^{-3/2}$.

Другие методы являются численными. В соответствии с первыми членами производящих рядов метод *аппроксимант Паде* позволяет давать оценки критических показателей, см. Стенли и др. [25], Реднер, Янг [24], Дхар и др. [14]. Прямые перечисления привели Дхара, Фани и Барма к предположению, что для числа ориентированных фигур на квадратной и треугольной решетке (размерности два) имеют место некоторые замечательные точные формулы.

§ 4. ФИГУРЫ НА ОГРАНИЧЕННОЙ ЛЕНТЕ (ПО ДЕРРИДА, ХАКИМУ, НАДАЛЮ, ВАННИМЕНЮ [17, 20, 21])

Для точного решения разрешимых моделей традиционно применяется метод *трансфер-матриц* (см., например, книгу Бакстера [1]). Он, в частности, хорошо приспособлен для перечисления ограниченных фигур *на ленте* ширины k, как показано на рис. 3. Фигура расположена между границами (зигзагообразными ломаными линиями), обозначенными на этом рисунке L_0 и L_{k-1}. Сужая фигуру на прямую D_m, заданную уравнением $y = -x - m$, получают точечную конфигурацию C_m. Существует 2^k возможных конфигураций. Определим матрицу $T_k(x)$, называемую трансфер-матрицей, строки и столбцы которой соответ-

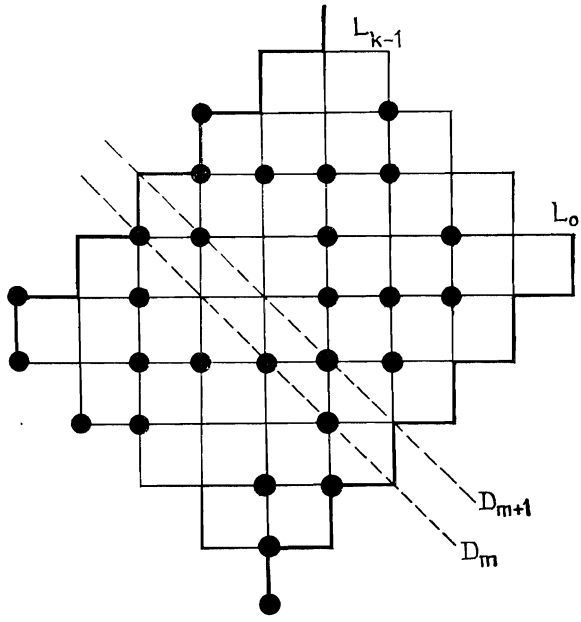


Рис. 3. Ориентированная фигура на ограниченной ленте.

ствуют этим 2^k конфигурациям. Элемент (C, C') есть $x^{|C|}$, если C и C' — два последовательных сечения C_m и C_{m+1} одной и той же ориентированной фигуры, и равен нулю в противном случае. Через $|C|$ обозначается число вершин, «занятых» данной конфигурацией. Отождествляя концы линий L_0 и L_{k-1}, можно определить фигуры на *циклической ленте* и соответствующую трансфер-матрицу $\bar{T}_k(x)$.

Обозначим $a_n^k(C)$ (соответственно $\bar{a}_n^k(C)$) число ориентированных фигур размера n с источником C, расположенных на ленте (соответственно на циклической ленте) ширины k. Производящий ряд $\sum_{n \geqslant 0} \bar{a}_n^k(C) t^n$ есть рациональная дробь, знаменателем которой является определитель $\det(I_k - \bar{T}_k(x))$, где I_k — единичная матрица размера $2^k \times 2^k$. Имеет место асимптотика

$$\bar{a}_n^k(C) \sim (\bar{\mu}_k)^n \bar{a}(C). \tag{4}$$

Вычисление $\bar{\mu}_k$ (как и аналога этой величины μ_k) сводится к вычислению наибольшего собственного значения трансфер-матрицы. Надаль и др. [21] получили замечательный результат

$$\bar{\mu}_k = 1 + 2\cos\frac{\pi}{2k}, \tag{5}$$

откуда следует, что $\mu = \lim_{k\to\infty} \mu_k = 3$. Они получили также точную формулу для $\bar{a}(C)$, который является ничем иным, как собственным вектором трансфер-матрицы, соответствующим собственному значению $\lambda_k(x) = 1$.

Методы *феноменологической перенормировки* позволяют дать точные оценки для $\nu_\perp$ и $\nu_\parallel$ и предположить, что $\nu_\parallel = 9/11$.

Для циклических источников обозначим через N_i число *дыр* ширины i, т. е. число последовательностей с i расположенными подряд «вакантными» позициями, окаймленными двумя позициями, «занятыми» источником.

Предложение 1 (Хаким, Надаль). *Пусть* $\alpha_p = (2p+1)\frac{\pi}{2k}$, *тогда*

$$\bar{a}_n^k(C) = \frac{1}{k}\sum_p (-1)^p \sin\alpha_p \prod_{i=1}^{k-1} \left(\frac{\sin[(i+1/2)\alpha_p]}{\sin(\alpha_p/2)}\right)^{N_i} (1 + 2\cos\alpha_p)^{n-1}. \tag{6}$$

Эту формулу, высказанную Надалем и др. [21] как предположение, доказали Хаким и Надаль [17], рассматривая трансфер-матрицу как оператор, действующий на пространстве спинов, и используя стандартную технику для интегрируемых

систем. Они дали формальный аналог, несколько более сложный, для $a_n^k(C)$ (свободные границы).

Из предложения 1 при $k \to \infty$ можно получить следующие замечательные точные формулы:

$$a_n = \sum_{0 \leqslant i \leqslant n} \binom{n-1}{i} \binom{i}{[i/2]}, \tag{7}$$

$$\sum_{n \geqslant 0} a_n t^n = \frac{1}{2}\left[\left(\frac{1+t}{1-3t}\right)^{1/2} - 1\right]. \tag{8}$$

Из них выводится асимптотическая формула вида (1) с $\mu = 3$ и $\theta = 1/2$. Формула (7) была угадана Дхаром и др. [14]. По

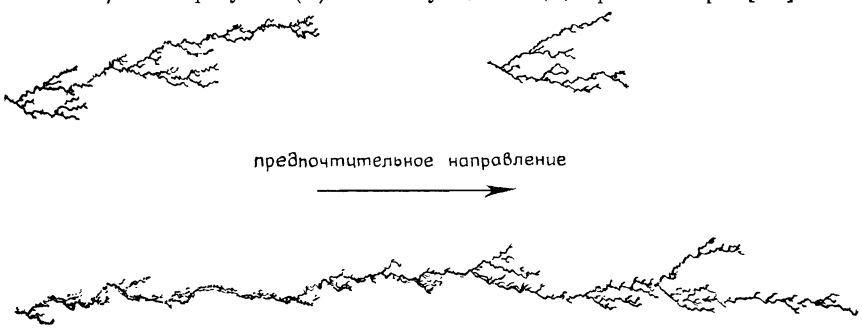


Рис. 4. Случайные ориентированные фигуры размера 500, 1000, 2000 (по Надалю, Деррида, Ваннименю).

аналогии с предложением 1 для свободных границ, Надаль [20] показал, что $\nu_\perp = 1/2$.

В § 8 можно найти другое доказательство формулы (8) (принадлежащее Дхару [12]), которое использует бакстеровскую модель газа. Точная формула (6) позволяет порождать определенным образом расположенные случайные фигуры, имеющие многочисленные сгустки точек. Каждая фигура размера n порождается с одной и той же вероятностью $1/a_n$. Как видно из рис. 4, имеются веские основания считать, что запрещение циклов в ориентированных фигурах не изменяет класса универсальности.

§ 5. ВЗАИМНО ОДНОЗНАЧНОЕ СООТВЕТСТВИЕ МЕЖДУ ОРИЕНТИРОВАННЫМИ ФИГУРАМИ РАЗМЕРНОСТИ ДВА (НА КВАДРАТНЫХ РЕШЕТКАХ) И НЕКОТОРЫМИ ПУТЯМИ РАЗМЕРНОСТИ ОДИН (ПО РАБОТЕ [28])

Пусть F_n — множество путей $\omega = (s_0 = 0, \ldots, s_n)$ на $\mathbb{Z}$, состоящих из элементарных шагов, удовлетворяющих неравенству

$|s_{i+1} - s_i| \leqslant 1$. В каждой вершине возможны три типа элементарных шагов. Обозначим через F_n^+ множество путей, суженное на N, т. е. таких, что $s_i \geqslant 0$.

Непосредственное вычисление показывает, что производящий ряд чисел $|F_{n-1}^+|$ именно тот, что фигурирует в (8). Путем установления элементарного взаимно однозначного соответствия можно также показать, что $|F_{n-1}^+|$ задается посредством (7).

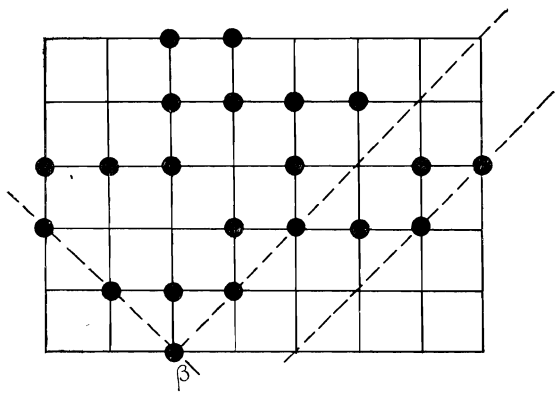


Рис. 5. Ориентированная фигура с компактным источником, $s(A) = 3$, $l_{\inf}(a) = 2$.

Возникает задача установления взаимно однозначного соответствия между множеством F_{n-1}^+ и множеством A_n ориентированных фигур на квадратной решетке, имеющих единственный точечный источник. Такое взаимно однозначное соответствие φ построено в [28].

Источник без дыр называется *компактным* (см. рис. 5). Фактически мы строим более общее взаимно однозначное соответствие ψ между F_{n-1} и ориентированными фигурами размера n, имеющими компактный источник.

Обозначения. Пусть A — некоторая ориентированная фигура. Число ее точечных источников обозначим $s(A)$. Пусть β — точечный источник, имеющий наименьшую ординату. Пусть b — целое число, такое что β располагается на прямой, заданной уравнением $y = x + b$. Нижняя полуширина фигуры A, обозначаемая $l_{\inf}(A)$, есть минимум величины $b - a$ на множестве целых чисел a, таких что рассматриваемая фигура целиком расположена выше прямой $y = x + a$. Пусть $\omega = (s_0, \ldots, s_n)$ — путь из F_n; обозначим через $m(\omega)$ наименьшее среди целых чисел s_i $(0 \leqslant i \leqslant n)$ и положим $d(\omega) = s_n - m(\omega)$.

Предложение 2 (Гуйю-Бошан, Вьенно). *Существует взаимно однозначное соответствие* ψ *между множеством путей* F_n *и множеством ориентированных фигур размера* $n+1$ *на квадратной решетке, имеющих компактный источник, которое удовлетворяет следующим соотношениям*:

$$m(\omega) = l_{\inf}(\psi(\omega)) \text{ и } 1 + d(\omega) = s(\psi(\omega)) \text{ для } \omega \in F_n. \qquad (9)$$

В частности, фигуры, имеющие единственный точечный источник, соответствуют путям из F_n, таким что $s_n = m(\omega)$. Обращением времени эти пути ставятся во взаимно однозначное соответствие с F_n^+. Таким образом, указанное взаимно однозначное соответствие φ установлено. Для всякого пути $\omega = (s_0, \ldots, s_n)$ из F_n^+ нижняя полуширина фигуры $A = \varphi(\omega)$ равна s_n. Теперь легко вывести (аналитически или посредством установления взаимно однозначного соответствия) следующую формулу, высказанную Дхаром в [12] как предположение.

Следствие 3.

$$l_n = \frac{2.3^{n-1}}{a_n} - 2.$$

Из соотношения (1) при $\mu = 3$, $\theta = 1/2$ следует, что $\nu_\perp = 1/2$. Обозначим $c_{n,p}$ число ориентированных фигур размера n, имеющих компактный источник мощности p. Из предложения 2 следует, что другое предположение Дхара, также доказанное Хакимом и Надалем, вытекает из (6), а именно:

$$c_{n+1,p} = c_{n,p-1} + c_{n,p} + c_{n,p+1}. \qquad (10)$$

Заметим также, что $a_n = c_{n,1}$ есть число ориентированных фигур размера n с компактным источником, нижняя полуширина которых равна нулю.

По-видимому, еще более любопытной является формула, приведенная в следствии 4.

Следствие 4. *Число ориентированных фигур размера* $n+1$ *на квадратной решетке, имеющих компактный источник, равно* 3^n.

Я не знаю никакого простого способа доказать это. Используемая биекция ψ получена комбинацией трех операторов U, V, W над ориентированными фигурами с компактным источником. Вместе эти операторы добавляют точку к фигуре A. Оператор U добавляет точечный источник, V оставляет $s(A)$ неизменным, и W уменьшает это число на единицу, кроме случая $s(A) = 1$, когда $s(W(A)) = 1$. Определение V требует особой аккуратности.

Некоторые свойства фигур на ленте допускают простой перевод на язык путей. Например, свойство «быть на ленте» сле-

дует из высказывания, что путь ω есть сужение на ограниченный отрезок $\mathbb{Z}$. Заметив, что ориентированные фигуры (вообще говоря) могут быть получены превращением точечных источников фигуры в компактный источник, можно было бы передоказать выражения Хакима, Надаля для $a_n^k(C)$. Но если взаимно однозначное отображение ψ единственно, это довольно тонкий результат. Лучше воспользоваться результатами §§ 7 и 8. Именно там приведена подлинная комбинаторная структура рассматриваемых ориентированных фигур.

§ 6. УКЛАДКИ [1])

В [26] автор показал, что биекция ψ может быть получена композицией трех взаимно однозначных отображений.

На каждом из этих трех этапов используется аппарат, построенный для решения ряда других задач комбинаторики.

Пусть X и P — два множества и π — отображение множества P в множество всех подмножеств X. *Помеченная укладка* (множества X плитками из P) есть (частично) упорядоченное множество $(E, \leqslant)$, снабженное отображением θ: $E \to P$, и удовлетворяющее двум следующим условиям:

(i) для всякого $x \in X$ множество $\{\alpha \in E,\ x \in \pi(\theta(\alpha))\}$ является цепью в E,

(ii) для любых α, $\beta \in E$, $\alpha \leqslant \beta$, существует цепь $\alpha = \alpha_1 \leqslant \alpha_2 \leqslant \ldots \leqslant \alpha_k = \beta$, такая что $\pi(\theta(\alpha_i)) \cap \pi(\theta(\alpha_{i+1})) \neq \varnothing$ для $1 \leqslant i \leqslant k-1$.

Множество X называется *базой*, P — множеством *плиток*. Отношение $\alpha \leqslant \beta$ есть «α предшествует β». Это определение формализует интуитивное понятие *укладки плиток*; которую можно представлять себе как объект, построенный из костяшек домино на шахматной доске и изображенный на рис. 6.

На этом рисунке $X = \mathbb{R} \times \mathbb{R}$, плитка — внутренность объединения двух элементарных прямоугольников с целыми координатами вершин. Отображение π есть тождественное отображение, θ ставит в соответствие каждой костяшке домино, входящей в укладку, ее «позицию» при проекции на шахматную доску. Вообще говоря, когда плитки — подмножества множества X, π — тождественное отображение. Мы ввели π в общее определение, потому что нам нужно строить укладки подмножеств множества X, имеющие определенную комбинаторную структуру (циклы и пути). В общем случае X и P будут бесконечными. Множество E будет всегда конечным.

[1]) В оригинале «empilements». — *Прим. ред.*

Легко определяется понятие изоморфизма между двумя укладками, имеющими одни и те же тройки (X, P, π). *Укладкой* множества X плитками из P (или типом укладки) назовем помеченную укладку множества X плитками из P, определенную с точностью до изоморфизма. Для более строгого определения нужно обратиться к «*родам структур*» Жуайяля [50]. Помеченные укладки образуют один род. Укладки суть *типы* этого рода. Грубо говоря, мы считаем, что элементы укладок суть плитки.

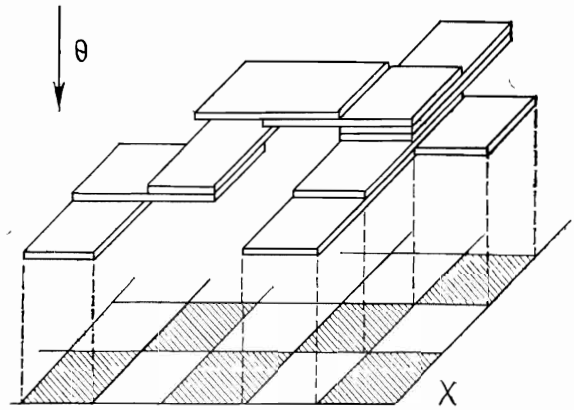


Рис. 6. Укладка плиток.

В задачах перечисления используют обычные степенные ряды для укладок и экспоненциальные ряды для помеченных укладок. Для этих последних всегда предполагают, что E есть отрезок $(1, \ldots, n)$.

Первый этап. Ориентированные фигуры $\Rightarrow$ укладки. Полагаем $X = \mathbb{Z}$. Назовем *домино* пару $(i, i+1)$, $i \in \mathbb{Z}$. Пусть P — множество таких домино, и пусть π — тождественное отображение на X. Как показано на рис. 7, ориентированные фигуры на треугольной решетке становятся укладками домино на $\mathbb{Z}$. Предпочтительное направление — север. Каждая точка фигуры замещается одним домино.

Лемма 5. *Описанное соответствие между ориентированными фигурами и укладками домино на* $\mathbb{Z}$ *взаимно однозначно.*

Точечные источники соответствуют минимальным плиткам (и только им). В частности, ориентированные фигуры, имеющие только один точечный источник, соответствуют *пирамидам*, т. е. укладкам, имеющим единственную минимальную плитку.

Укладка E называется *плотной* (strict), если выполнены следующие условия:

$$\alpha, \beta \in E, \quad \alpha < \beta, \quad \pi(\theta(\alpha)) = \pi(\theta(\beta)) \Rightarrow \exists \gamma \in E, \quad \alpha < \gamma < \beta. \quad (11)$$

При взаимно однозначном соответствии из леммы 5 фигурам на квадратной решетке соответствуют плотные укладки (домино на $\mathbb{Z}$).

Второй этап. Укладки $\Rightarrow$ пути. Здесь база X произвольна. Пусть u и v — два элемента X. Плитки теперь имеются двух

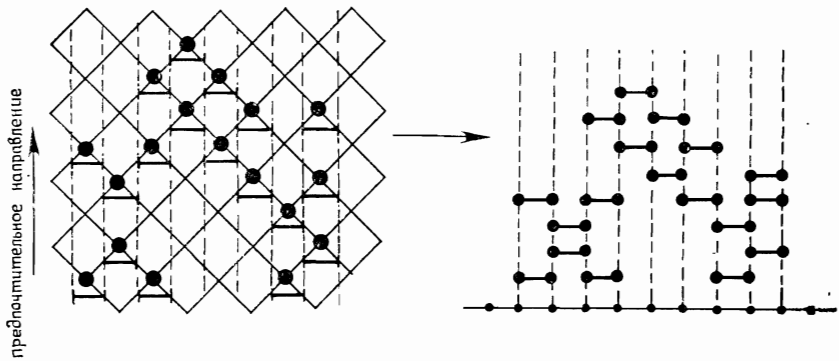


Рис. 7. Соответствие ориентированная фигура → укладка.

сортов: *циклы* (или циклические перестановки γ: $\Gamma \times \Gamma$ на подмножествах $\Gamma \subseteq X$) и *пути* $\omega = (s_0 = u, \ldots, s_n = v)$ — несамопересекающиеся пути, ведущие из u в v. По определению $\pi(\gamma) = \Gamma$ и $\pi(\omega) = \{s_0, \ldots, s_n\}$.

Лемма 6. *Существует взаимно однозначное соответствие между путями ω в X, ведущими из u в v, и пирамидами E на X, все плитки которых суть циклы, за исключением минимальной плитки, которая является путем (ведущим из u в v и несамопересекающимся). Более того, для всех вершин x и y из X число элементарных шагов (x, y) пути ω равно числу элементарных шагов (x, y) пути в пирамиде, умноженному на число циклов γ этой пирамиды, таких что $y = \gamma(x)$.*

В частности, для $X = \mathbb{Z}$ путь, ведущий из 0 в v, принадлежит F (определение см. в § 5) тогда и только тогда, когда соответствующая пирамида относится к следующему типу: путь, образованный минимальной плиткой, есть последовательность $(0, 1, 2, \ldots, v)$ при $v \geqslant 0$ (соответственно $(0, -1, -2, \ldots, v)$ при $v < 0$); циклы имеют длину один или два и образованы двумя последовательными точками. Эти циклы отождествляются

с плитками, именуемыми *мономино* $\{i\}$ и *домино* $\{i, i+1\}$. Если *стационарный шаг* $s_{i+1} = s_i$ запрещен для пути v, то укладки строятся только из домино.

Из этих замечаний в сочетании с леммами 5 и 6 получаем, что число b_n фигур (на треугольной решетке) размера n с единственным точечным источником таково:

$$b_n = \frac{1}{2}\binom{2n}{n}. \tag{12}$$

Эта формула, высказанная в виде предположения Дхаром и др. [14], позднее была получена Дхаром [12] с использованием газовой модели Бакстера (см. § 8). Асимптотически $b_n \sim 4^n n^{-1/2}$, и, таким образом, гипотеза универсальности подтверждается.

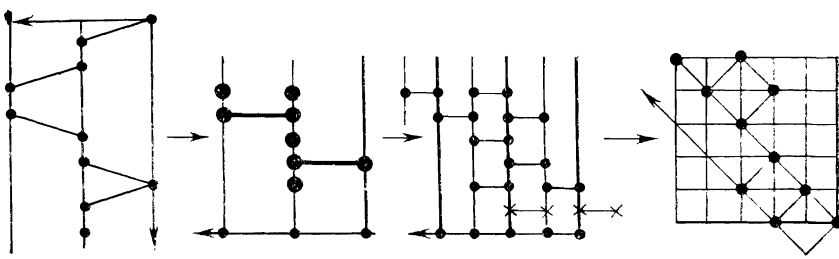


Рис. 8. Соответствие путь → ориентированная фигура при $\omega \in F_n^+$.

Третий этап. Укладки ⇒ плотные укладки. Это несколько техническое взаимно однозначное отображение, преобразующее плотную укладку домино на $\mathbb{N}$ в укладку мономино и домино на четных положительных числах (рис. 8). Оно порождено другим, более простым отображением, происходящим из комбинаторной модели для ортогональных полиномов [55] (о которой говорится в § 7).

Комбинируя эти три этапа, находим в точности конструкцию $\psi(\omega)$ из § 5 для случая, когда $\omega \in F_n^+$ (т. е. $l_{\inf}(\psi(n)) = 0$). Теперь остается сделать совсем немного для определения $\psi(n)$ в общем случае.

Укладки и коммутационные моноиды Картье и Фоата.

Картье и Фоата ввели в [38] *коммутационные моноиды* в связи со знаменитой master theorem [1]) Макмагона и свойствами

[1]) Этот термин переводится иногда как «главная теорема», иногда как «основная теорема». — *Прим. ред.*

перегруппировок последовательностей. Фоата [44, 45] показал, что этот аппарат позволяет приспособить комбинаторную модель для биективного (путем установления некоторого взаимно однозначного соответствия) доказательства классических теорем алгебры матриц (обращение матрицы, тождество Якоби и др.). В случае путей и циклов (лемма 6) Дюлюк и автор [40] ввели укладки с целью упрощения и унификации таких биективных рассуждений. Такая формализация применяется также в других разделах комбинаторики (ортогональные полиномы [55] и полиномы паросочетаний графа [39]).

Коммутационные моноиды определяются следующим образом. Пусть $\mathrm{Mo}(P)$ (обозначаемый также через P^*) — свободный моноид, порожденный P. Пусть C — подмножество $P \times P$. Коммутационный моноид, порожденный P и C, есть факторизация моноида $\mathrm{Mo}(P)$ по конгруэнции, порожденной отношением коммутативности $\alpha\beta \equiv \beta\alpha$ для $(\alpha, \beta) \in C$. Со всяким отображением π: $P \to X$ связывают множество пар (α, β), таких что $\pi(\alpha) \cap \pi(\beta) = \varnothing$. Тогда каждый элемент соответствующего коммутационного моноида находится во взаимно однозначном соответствии с укладкой множества X плитками из P. Каждое слово длины n из одного и того же класса эквивалентности находится во взаимно однозначном соответствии с некоторой помеченной укладкой $(E = \{1, \ldots, n\}, \prec)$, такой что $i \prec j$, когда плитка i предшествует j. Таким образом, лемма 6 есть просто измененная версия теоремы Картье, Фоата [38].

§ 7. ИНВЕРСИЯ

В этом параграфе предполагаем, что база X и множество P плиток укладки конечны. Пусть $\mathbb{K}$ — коммутативное кольцо с единицей, V — отображение $P \to \mathbb{K}$, именуемое *оценкой*. Практически $\mathbb{K}$ есть кольцо целых чисел или кольцо коммутативных полиномов от нескольких переменных с целыми коэффициентами. *Оценка укладки* (E, θ) есть произведение $\prod_{\alpha \in E} v(\theta(\alpha))$.

Укладка называется *тривиальной*, если элементы E попарно несравнимы. Это приводит к утверждению, что множества $\pi(\theta(\alpha))$ попарно не пересекаются. Обозначим $D(x)$ полином, заданный соотношением

$$D(x) = \sum_E (-1)^{|E|} v(E)\, x^{|E|}, \tag{13}$$

где суммирование производится по тривиальным укладкам множества X плитками из P.

Лемма 7. *Производящий ряд помеченных укладок множества X плитками из P есть*

$$\sum_E v(E)\, x^{|E|} = \frac{1}{D(x)}. \tag{14}$$

Эта лемма, в сущности, переложение теоремы 2.4 [38], дающее функцию Мёбиуса для коммутационных моноидов. Понятие функции Мёбиуса на (частично) упорядоченных множествах, обобщающее хорошо известную функцию Мёбиуса на целых числах, было популяризировано Рота [52]. Оно играет большую роль в комбинаторике.

Преимущество представленной здесь версии состоит в возможности сформулировать более сильную лемму для производящих рядов укладок, минимальные множества которых заранее фиксированы. В этом случае она принимает следующий вид:

$$\sum_F v(F)\, x^{|F|} = \frac{N(x)}{D(x)}, \tag{15}$$

где $N(x)$ есть некоторый полином из $\mathbb{K}[x]$.

Идея доказательства состоит в следующем. Пусть $N(x) = \sum_{(E,\,F)} (-1)^{|E|}\, v(E)\, v(F)\, x^{|E|+|F|}$, где E — тривиальная укладка, F — укладка, удовлетворяющая заданным условиям. Посредством «переноса» некоторой максимальной плитки одной укладки в другую определяют инволюцию, группирующую попарно члены с оценкой одинаковой величины и противоположными знаками. Остается конечное число членов, дающее числитель $N(x)$. Это соответствует парам (E, F), для которых такой «перенос» невозможен.

Отношение (15) применяют к укладкам домино на отрезке $X = (1, \ldots, k)$. Оценка домино выбирается равной 1. Обозначим $U_k(x)$ полином Чебышёва второго рода. Определенный соотношением $\sin(k+1)\xi = \sin\xi U_k(\cos\xi)$. Полином $D(x)$, определенный равенством (13), является тогда полиномом $F_k^*(x)$, сопряженным к ортогональному полиному $F_k(x) = U_k(x/2)$. Прямое применение лемм 5 и 7 показывает, что производящий ряд ориентированных ограниченных фигур на ленте ширины k и треугольной решетке есть $1/F_k(x)$. Если заданы точечные источники, то в этом случае выполнено соотношение (15). Числитель $N(x)$ есть произведение полиномов $F_i^*(x)$, индексы которых i соответствуют ширине дыр источника. Когда база X есть цикл C_k, имеющий k вершин, получаем фигуры на циклической ленте и, следовательно, полином $U_k(x)$ достаточно заменить полиномом Чебышёва первого рода $T_k(x)$, определяемым соотношением $\cos\xi = T_k(\cos\xi)$.

И опять случай ограниченных фигур на ленте (на квадратной решетке) получается комбинацией (15) с третьим этапом § 6. Образование пар мономино в укладках ведет к замене x на $x-1$. Никаких других вычислений для доказательства предложения 1 и его аналога в случае свободных границ не нужно.

Комбинаторная модель для ортогональных полиномов.

Предыдущие рассуждения позволяют построить часть комбинаторной модели для ортогональных производящих полиномов Флажоле [41] и Вьенно [55]. Эта модель, в терминах взвешенных путей, предложена Моцкином. Рассмотрим основную идею, переведя ее на язык укладок. Пусть $v(\{i\}) = b_i$ и $v(\{i-1, i\}) = \lambda_i$ суть соответственно оценки полимино и домино на $\mathbb{N}$. Полиномы $D_k(x)$, определенные равенством (13) при $X = \{1, \ldots, k-1\}$, являются сопряженными с полиномами $P_k(x)$, определяемыми линейными рекуррентными соотношениями

$$P_k(x) = (x - b_k) P_k(x) - \lambda_k P_{k-1}(x), \quad P_0 = 1, \quad P_1 = x - b_1. \quad (16)$$

Взаимно однозначное отображение (фактически то же самое, что и в (15)) показывает, что эти полиномы ортогональны относительно последовательности моментов $M_n = \sum_P v(P)$, где суммирование производится по пирамидам на $\mathbb{N}$, образованным n мономино или домино, и минимальное подмножество которых содержит 0. Производящий ряд этих пирамид допускает переход к непрерывным дробям типа Якоби — Стильтьеса. Ограничение укладок на отрезок $\{0, \ldots, k-1\}$ приводит к сокращению этих непрерывных дробей. Последние выбираются в виде (15). Описанный случай полиномов Чебышёва приводит к выбору $b_k = 0$, $\lambda_k = 1$. Для фигур на квадратной решетке имеем $b_k = 1$, $\lambda_k = 1$. В этом случае получается последовательность ортогональных полиномов, рассмотренная Надалем и др. [21] в их доказательстве соотношения (5) и вычислении собственного вектора $\bar{a}(C)$. Соответствующие моменты становятся числами укладок (эти числа указаны Каталаном и Моцкином). Отметим, что взаимно однозначное соответствие из третьего этапа § 6 порождено взаимно однозначным соответствием (гораздо более простым), объясняющим так называемые свойства свертки в непрерывных дробях.

Другие замечания. Соотношение (15) возникает также во многих других комбинаторных задачах. Если плитки суть циклы, оцениваемые элементами матрицы A, то $D(x) = \det(I - A)$, и лемма 7 представляет собой замаскированную master theorem Макмагона [51], тогда как соотношение (15) становится классической формулой обращения матрицы $(I - A)$.

Если плитки суть домино (оцененные единицей) на конечном графе (пара двух смежных вершин), то полином $D(x)$, определенный равенством (13), называют *полиномом паросочетания* графа. Вычисление этого полинома для плоской решетки (по крайней мере ее «термодинамического предела») — знаменитая задача, возникшая из статистической механики. Вычисление коэффициентов при более высоких степенях (число совершенных или полных паросочетаний) эквивалентно решению модели Изинга ($d = 2$, внешнее магнитное поле — нуль). Полиномы паросочетаний имеют вещественные корни. Биективное (почти полностью) доказательство дано де Сент-Катрин [39] как комбинация соотношения (15), леммы 6 и идеи Годсила [48].

§ 8. ОРИЕНТИРОВАННЫЕ ТРЕХМЕРНЫЕ ФИГУРЫ И МОДЕЛИ ЖЕСТКИХ ШЕСТИУГОЛЬНИКОВ

Пусть D — конечное связное подмножество треугольной решетки. Обозначим $h(n, d)$ число способов выбора n попарно не-

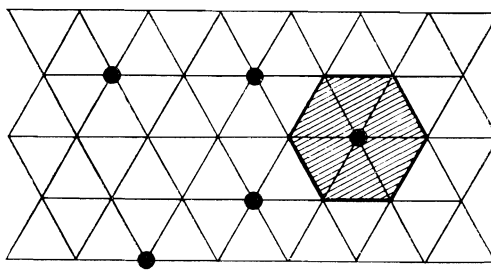


Рис. 9. Конфигурация точек, приводящих к $h(n, d)$.

смежных вершин (рис. 9). Статистическая сумма для этой модели, называемой *жесткими шестиугольниками*, есть

$$Z_D(t) = \sum_{n \geqslant 0} h(n, D)\, t^n. \qquad (17)$$

Напомним, что термодинамический предел при D, становящемся (в определенном смысле) большим, имеет вид

$$Z(t) = \lim_{D \to \infty} (Z_D(t))^{1/|D|}. \qquad (18)$$

Предположив существование этого предела, Бакстер решил эту модель ([30], [1, гл. 14] и Бакстер, Пирс [33]). В двух первых работах предполагаются более определенные свойства

аналитичности функции $\log Z(t)$. Модель допускает фазовые переходы при $t_C = \frac{1}{2}(11 + 5\sqrt{5}) \cong 11.09$.

Обозначим $R_{\mathrm{I}}(q)$ и $R_{\mathrm{II}}(q)$ общее значение двух членов тождеств Роджерса — Рамануджана

$$\sum_{n \geqslant 0} \frac{q^n}{(1-q) \ldots (1-q^n)} = \prod_{n \geqslant 0} \frac{1}{(1-q^{5n+1})(1-q^{5n+4})}, \tag{19}$$

$$\sum_{n \geqslant 0} \frac{q^{n^2+n}}{(1-q) \ldots (1-q^n)} = \prod_{n \geqslant 0} \frac{1}{(1-q^{5n+2})(1-q^{5n+3})}. \tag{20}$$

Для $0 < t < t_C$ функция $Z(t)$ получена посредством исключения q из двух следующих уравнений:

$$t = -q\left[\frac{R_{\mathrm{II}}(q)}{R_{\mathrm{I}}(q)}\right]^5, \tag{21}$$

$$Z = \prod_{n \geqslant 0} \frac{(1-q^{6n+2})(1-q^{6n+3})^2 (1-q^{6n+4})(1-q^{5n+1})^2 (1-q^{5n+4})^2 (1-q^{5n})^2}{(1-q^{6n+1})(1-q^{6n+5})(1-q^{6n})^2 (1-q^{5n+2})^3 (1-q^{5n+3})^3}. \tag{22}$$

Решение основывается на методе поквадрантных трансфер-матриц. Бакстер ввел более общую модель с двумя параметрами L и M на квадратной решетке с диагональными взаимодействиями. Переменная t, рассматриваемая как формальная переменная в указанных ниже производящих рядах, является здесь положительным числом, именуемым *активностью* (или фугасностью) газа. Бакстер определяет тогда соотношения между параметрами L, M и t, при которых трансфер-матрицы коммутируют. Далее получается биквадратичное соотношение, связывающее e^L и e^M, допускающее параметризацию с использованием эллиптической тэта-функции. Исходная модель жестких шестиугольников соответствует предельному случаю режима I Бакстера при $L = 0$, $M \to \infty$.

Дхар [12] показал, что производящий ряд ориентированных фигур (квадраты или треугольники, $d = 2$) можно заменой переменных свести к приведенной ниже модели обобщенных шестиугольников. Параметры не удовлетворяют соотношениям, которые делали бы модель разрешимой, но Бакстер [32] доказывает разрешимость, перейдя к величинам, не являющимся физически активными ($t < 0$) в разрешимой модели Ферхагена [37] (модель Изинга с магнитным полем на треугольном источнике с некоторыми соотношениями между параметрами). Для этого случая Дхар выводит производящий ряд (8) и ряд, соответствующий (12).

В другой статье [13] Дхар рассматривает производящий ряд $g(t)$ ориентированных фигур на $\mathbb{Z}^3$ (разрешен четвертый шаг по диагоналям для ориентированных путей в направлении (1, 1, 1)). Он доказал, воспользовавшись вероятностными методами (модель «роста кристаллов»), что $g(t)$ связана с плотностью газа $\rho(t) = t\frac{d}{dt}\log Z(t)$ в шестиугольной модели Бакстера соотношением

$$f(t) = -\rho(-t). \tag{23}$$

Чисто комбинаторное доказательство этого соотношения дано автором в [27]. В действительности полином (17) является полиномом типа (13). Плитки образованы шестью вершинами

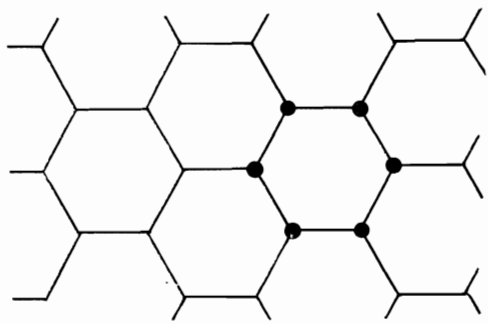


Рис. 10. Плитка для укладок шестиугольников.

элементарных шестиугольников шестиугольной решетки (см. рис. 10).

Лемма 7 интерпретирует $Z^{-1}(t)$ как укладку шестиугольников. Перейдем к экспоненциальным рядам с помеченными укладками. Здесь можно применить технику частичного смешивания Фоата, Шютценберже [43, 46] (или предложенную автором версию этой техники, модернизированную Жуайялем [50]) для интерпретации $\log Z^{-1}(t)$ как производящего ряда некоторых помеченных пирамид. Теперь легко понять переход к пределу (15). Рассмотрим «краевые эффекты» в ограниченных пирамидах на «цилиндре» множества D. Затем, применив оператор $t\frac{d}{dt}$, перейдем к «повышению» меток пирамид. Переходя к обычным степенным рядам, получаем, что $-\rho(-t)$ есть также производящий ряд для пирамид шестиугольников. Как и в двумерном случае, эти пирамиды находятся во взаимно однозначном соответствии с ориентированными фигурами на кубической решетке с диагональными шагами.

Дхар вывел из (23) и из решения Бакстера асимптотику для числа фигур. Она соответствует не имеющей физического смысла критической точке системы $t_{NP} = -1/t_c$. Отсюда следует, что $\mu = (11 + 5\sqrt{5})/2$ и $\theta = 5/6$. Заметим, что тождество

$$\frac{1}{2}(11 + 5\sqrt{5}) = \left(\frac{1+\sqrt{5}}{2}\right)^5$$

получено Роджерсом — Рамануджаном из непрерывной дроби ($q = 1$)

$$\frac{R_{II}(q)}{R_I(q)} = \cfrac{1}{1 + \cfrac{q}{1 + \cfrac{q^2}{\ddots}}}.$$

Замечание. Значительно более простое доказательство этого знаменитого тождества можно провести, установив взаимно однозначное соответствие с укладками домино на $\mathbb{N}$, если в качестве веса взять $v(\{i-1, i\}) = -q^i$. Фактически $R_I(q)$ и $R_{II}(q)$ — частные случаи полиномов $D(x)$ и $N(x)$, фигурирующих в (15).

Можно также рассмотреть вместе с Дхаром ориентированные фигуры на решетке, называемой «*кубоцентрической*», и комбинаторно доказать их эквивалентность модели, которая называется *жесткими квадратами*. Кроме того, эта модель есть предельный случай $L = M = 0$ модели Бакстера с двумя параметрами. Для этих значений модель неразрешима. Соответствующие пирамиды суть некоторые пирамиды квадратов.

Заметим, наконец, что модель жестких шестиугольников (соответственно жестких квадратов) сводима к *модели Поттса* для $q = 2$ (соответственно $q = 3$). С другой стороны, Эндрюс и Бакстер [29] пришли к пониманию решения шестиугольной модели с параметрами в случае, когда переменные $\sigma_i = 0$ или 1, приписанные каждой вершине треугольной решетки, могут принимать p различных значений. При этом доказаны мультилинейные аналоги тождеств типа Роджерса — Рамануджана. Доказана их эквивалентность некоторым *моделям вершин*.

§ 9. ЗАКЛЮЧИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

Было бы очень интересно иметь прямое комбинаторное доказательство того, что производящий ряд для пирамид шестиугольников задается соотношениями (21), (22), (23), и понять, в чем состоит их отличие от перечислений пирамид квадратов.

Вычисление плотности и «*параметра порядка*» для разных режимов шестиугольной модели (с параметрами) позволяет получить кроме тождеств Роджерса — Рамануджана двенадцать

других тождеств того же типа, шесть из которых являются новыми. Они изобретены Бакстером и доказаны Эндрюсом [57]. Удивительно, что они приводят к обычной классификации гипергеометрических функций. Было бы, в частности, интересно понять, почему тождества типа Роджерса — Рамануджана всегда появляются в решении шестиугольной модели (см. Бакстер [31]).

Тождества Роджерса — Рамануджана могут быть интерпретированы в терминах *разбиений целых чисел*, которые обычно представляются посредством некоторой ориентированной фигуры, именуемой *диаграммой Фере* (см. Эндрюс [56]). Поиск биективного доказательства заставляет специалистов по комбинаторике тратить много чернил, см. Гарсиа, Милн [47]. Эти тождества имеют интерпретацию в терминах алгебр Каца — Муди (см. обзор Макдональда на этом семинаре [59]). Роль этих алгебр существенна для понимания разрешимости модели. Выдвинуты основные идеи (отношение звезда — треугольник, отношение обращения). См., например, Бакстер [1] или Екель и Майяр [34, 35].

Бесконечные произведения членов вида $(1-q^i)$ возникают во многих разрешимых моделях. Знаменитая функция Δ возникла в модели, называемой *плоскими циклами*. Хотя комбинаторное изучение эллиптических функций Якоби начато (Дюмон [41], Флажоле [42], Вьенно [53]), тета-функции и модулярные формы пока еще не изучены с этой точки зрения. Разрешимые модели статистической механики представляют собой, быть может, хорошую отправную точку для такого рода исследований.

Наконец, физики интересуются *ориентированными растущими фигурами* (модель ориентированной агрегации, см., например, Надаль и др. [22]). Это ориентированные фигуры, помеченные целыми числами $1, 2, \ldots, n$ таким образом, что метки возрастают вдоль ориентированных путей. Знаменитые таблицы Юнга [58], соответствующие фигурам «диаграмм Фере», являются частным случаем такого рода объектов. Три типа формы ориентированных фигур приводят к формулам, именуемым в комбинаторике «*формулами крюка*» (formules d'équerre). Вот единственные типы таких формул, известные в настоящее время: таблицы Юнга, усеченные таблицы Юнга, растущие (бинарные) деревья, соответствующие фигурам без циклов. Существуют ли другие формулы крюка? Известно, что таблицы Юнга связаны с представлениями симметрических групп. Какая алгебраическая структура играет роль аналога растущих ориентированных фигур?

Благодарности. Я благодарю М. Е. Екеля, Ж. М. Майяра, Ж. П. Надаля и Ж. Ваннименю за живые дискуссии на темы статистической механики. Я благодарю также Д. Фоата и Ж. Л. Вердье за многочисленные замечания, а также А. Доди, который меня познакомил с Ж. Ваннименю, который в свою очередь познакомил меня с ориентированными фигурами.

ЛИТЕРАТУРА

A. Основные работы и статьи по синтезу

Обзор известных актуально разрешимых моделей

[1] R. Bakster, Exactly solved models in statistical mechanics, Academic Press, New York, 1982. [Имеется перевод: Бэкстер Р. Точно решаемые модели в статистической механике. — М.: Мир, 1985.]

Книги по статистической механике

[2] D. Ruelle, Statistical mechanics, rigorous results, Benjamin, New York, 1969. [Имеется перевод: Рюэль Д. Статистическая механика. Строгие результаты. — М.: Мир, 1971.]
[3] C. Domb et M. S. Green, eds, Phase transition and critical phenomena, Academic Press, New York, 1974.

Комбинаторные методы в статистической физике, комбинаторное решение модели Изинга и формулировка на языке графов

[4] J. K. Percus, Combinatorial methods, Applied mathematical sciences, vol. 4, Springer-Verlag, New York/Berlin, 1971.
[5] P. W. Kasteleyn, Graph theory and crystal physics, in «Graph theory and theoretical physics», F. Harary ed. Academic Press, New York, 1967, 43—110.
[6] F. Y. Wu. Graph theory in statistical physics, in «Studies in foundations and combinatorics», Advances in Math. supplementary studies, vol. 1, 1978.

Недавний обзор задач просачивания, ориентированного просачивания и фигур (с цитированными статьями по синтезу)

[7] M. Sahimi, Critical exponents and thresholds for percolation and conduction, in «Mathematics and physics of desordered media», Lecture Notes in Maths n° 1035, Springer-Verlag, New York/Berlin, 1983, 314—346.

B. Статьи об ориентированных фигурах

[8] N. Breuer, Corrections to scaling for directed branched polymers (lattice animals), preprint (1983), soumis à Z. Physik B.
[9] N. Breuer et H. K. Janssen, Critical behaviour of directed branched polymers and the dynamics at Yang — Lee edge singularity, Z. Physik B, 48 (1982) 347—350.
[10] J. L. Cardy, Directed lattice animals and the Lee — Yang edge singularity, J. Phys. A: Math. Gen., 15 (1982), 1593—1595.
[11] A. R. Day et T. C. Lubensky, ε expansion for directed animals, J. Phys. A: Math. Gen., 15 (1982), L 285—L 290.
[12] D. Dhar, Equivalence of the two-dimentional directed-site animal problem to Baxter's Hard-Square Lattice-Gas model, Phys. Rev. Lett., 49 (1982), 959—962.

[13] D. Dhar, Exact solution of a directed-site animals-enumeration problem in 3 dimensions, Phys. Rev. Lett., 59 (1983), 853—856.
[14] D. Dhar, M. K. Phani et M. Barma, Enumeration of directed site animals on two-dimensional lattices, J. Phys. A: Math. Gen., 15 (1982), L 279—L 284.
[15] F. Family, Relation between size and shape of isotropic and directed percolation clusters and lattice animals, J. Phys. A: Math. Gen., 15 (1982), L 583—L 592.
[16] J. E. Green et M. A. Moore, Application of directed lattice animal theory to river networks, J. Phys. A: Math. Gen., 15 (1982), L 597—L 599.
[17] V. Hakim et J. P. Nadal, Exact results for 2 D directed animals on a strip of finite width, J. Phys. A: Math. Gen., 16 (1983), L 213—L 218.
[18] H. Herrmann, F. Family et H. E. Stanley, Position-space renormalisation group for directed branced polymers, J. Phys. A: Mith. Gen., 16 (1983), L 375—L 379.
[19] T. C. Lubensky et J. Vannimenus, Flory approximation for directed branched polymers and directed percolation, J. Physique, 43 (1982), L 377—L 381.
[20] J. P. Nadal, Etude des systèmes dirigés en physique statistique, Thèse 3éme cycle, Univ. d'Orsay, 1983.
[21] J. P. Nadal, B. Derrida et J. Vannimenus, Directed lattice animals in 2 dimensions: numerical and exact results, J. Physique, 43 (1982), 1561.
[22] J. P. Nadal, B. Derrida et J. Vannimenus, Directed Diffusion-controlled Aggregation versus directed animals, Preprint (1983).
[23] S. Redner et A. Coniglio, Flory theory for directed lattice animals and directed percolation, J. Phys. A: Math. Gen., 15 (1982), L 273—L 278.
[24] S. Redner et Z. R. Yang, Size and shape directed lattice animals, J. Phys. A: Math. Gen., 15 (1982), L 177—L 187.
[25] H. E. Stanley, S. Redner et Z. R. Yang, Site and bond directed branched polymers for arbitrary dimensionality: evidence supporting a relation with the Lee — Yang edge arbitrary, J. Phys. A: Math. Gen., 15 (1982), L 569—L 573.
[26] G. Viennot, Combinatorial solution of the 2D directed lattice animals problem with heaps of dominos, en préparation.
[27] G. Viennot, Directed animals and combinatorial interpretation of the density of gaz, Advances in Applied Maths.
[28] G. Viennot et D, Gouyou — Beauchamps, The number of directed animals, Advances in Applied Maths.

C. Другие цитированные статьи по статистической механике

[29] G. E. Andrews, R. J. Baxter et P. J. Forresters, Eight vertex SOAS model and generalized Rogers — Ramanujan type identities, preprint (1984).
[30] R. J. Baxter, Hard hexagons: exact solution, J. Phys. A: Math. Gen., 13 (1980), L 61—L 70.
[31] R. J. Baxter, Rogers — Ramanujan identities in hard hexagon model, J. Stat. Physics, 26 (1981), 427—452.
[32] R. J. Baxter, письмо Дхару, цитированное в [12].
[33] R. J. Baxter et P. A. Pearce, Hard hexagones: interfacial tension and correlation length, J. Phys. A: Math. Gen., 15 (1982), L 897—L 910.
[34] M. T. Jaekel et J. M. Maillard, Modèles solubles en mécanique statistique, Proc. R. C. P. 25, vol. 29, Publ. IRMA, Strasbourg (1982), 93.
[35] J. M. Maillard, Equations fonctionnelles en mécanique statistique, Note CEA-N-2332, (1983).
[36] G. Parisi et N. Sourlas, Phys. Rev. Lett. 14 (1981), 871.
[37] A. M. W. Verhagen, J. Stat. Physics, 15 (1976), 219.

D. Цитированные статьи по комбинаторике

[38] P. Cartier et D. Foata, Problèmes combinatoires de commutation et réarrangements, Lecture Notes in Maths. n° 85, Springer-Verlag, New York/Berlin, 1969.
[39] M. de Sainte Catherine, Couplage ot Pfaffien en combinatoire et physique, Thèse 3ème cycle, Université de Bordeaux, I, 1983.
[40] S. Dulucq et G. Viennot, The Cartier — Foata commutation monoid revisited with heaps of pieces, manuscript, 1983.
[41] D. Dumont, Une approche combinatoire des fonctions elliptiques de Jacobi, Advances in Maths., 41 (1981), 1—39.
[42] P. Flajolet, Combinatorial aspects of continued fractions, Discrete Math., 32 (1980), 125—161.
[43] D. Foata, La série génératrice exponentielle dans les problèmés d'énumération, Presses de l'Univ. de Montréal, (1974).
[44] D. Foata, A non-commutative version of the matrix inversion formula, Advances in Maths., 31 (1979), 330—349.
[45] D. Foata, A combinatorial proof of Jacobi's identity, Annals of Discrete Maths. 6 (1980), 125—135.
[46] D. Foata, M. P. Schutzenberger, Théorie geométrique des polynomes Eulériens, Lecture Notes in Maths. n° 138, Springer-Verlag, New York/Berlin, 1970.
[47] A. Garsia, S. Milne, A Rogers—Ramanujan bijection, J. Combinatorial Th., A, 31 (1981), 289—339.
[48] C. D. Godsil, Matchings and walks in graphs, J. Graphs Th., 5 (1981), 285—291.
[49] S. N. Joni et G. C. Rota, Coalgebras and bialgebras in combinatorics, Studies in Applied Maths., 61 (1979), 93—139.
[50] A. Joyal, Une théorie combinatoire des séries formelles, Advances in Maths., 42 (1981), 1—82.
[51] P. A. MacMagon, Combinatory Analysis, Cambridge Univ. Press, London, 1915. Перепечатка: Chelsea, New York, (1960).
[52] G. C. Rota, On the foudations of combinatorial theory, I. Theory of Möbius functions, Z. Wahrsch. Band 2, Heft 4, (1964), 340—368.
[53] G. Viennot, Une interprétation combinatoire des coefficientes des développements en série entière des fonctions elliptiques de Jacobi, J. Combinatorial Th., A, 29 (1980), 121—133.
[54] G. Viennot, Théorie combinatoire des nombres d'Euler et Genocchi, Séminaire Théorie des Nombres 1981/82, Publication Université de Bordeaux I, 94 p.
[55] G. Viennot, Théorie combinatoire des polynômes ortogonaux généraux, Notes de seminaire de l'Univ. du Québec a Montréal, (1983), 165p.

E. Другие цитированные статьи и работы

[56] G. E. Andrews, The theory of partitions, Encyclopedia of Maths. and its applications, G. C. Rota ed., vol. 2, Addison-Wesley, Reading. (1976). [Имеется перевод: Эндрюс Г. Теория разбиений. — М.: Мир, 1979.]
[57] G. E. Andrews, The hard-hexagon model and Rogers — Ramanujan identities, Pros. Nat. Acad. Sci. U. S. A., 78, (1981), 5290—5292.
[58] P. Cartier, La théorie classique et moderne des fonctions symétriques, Séminaire Bourbaki, exposé n° 577, Astérisque n° 105—106, (1983), 1—23.
[59] I. G. MacDonald, Affine Lie algebras and modular forms, Séminaire Bourbaki, exposé n° 577, Lecture Notes in Maths. n° 901, Springer-Verlag, New York/Berlin, (1981), 258—275.

СОДЕРЖАНИЕ

Научное издание

КИБЕРНЕТИЧЕСКИЙ СБОРНИК

Новая серия

Выпуск

27

Заведующий редакцией чл.-корр. АН СССР В. И. Арнольд
Зам. зав. редакцией А. С. Попов
Научный редактор С. В. Чудов
Мл. научные редакторы Т. А. Дехтярева, Л. А. Королева
Художник Н. К. Сапожников
Художественный редактор В. И. Шаповалов
Технический редактор Л. П. Бирюкова
Корректор Л. И. Панова

ИБ № 7335

Сдано в набор 05.01.89. Подписано к печати 23.07.90. Формат $60\times90^1/_{16}$. Бумага типографская № 1. Печать высокая. Гарнитура литературная. Объем 6,25 бум. л. Усл. печ. л. 12,5. Усл. кр.-отт. 12,5. Уч.-изд. л. 11,56. Изд. № 1/7147. Тираж 2900 экз. Зак. 384. Цена 2 р. 50 к. В/О «Совэкспорткнига» государственного комитета СССР по печати. Издательство «Мир», 129820, ГСП, Москва, И-110, 1-й Рижский пер., 2.

Набрано в Ленинградской типографии № 2 головного предприятия ордена Трудового Красного Знамени Ленинградского объединения «Техническая книга» им. Евгении Соколовой Государственного комитета СССР по печати. 198052, г. Ленинград, Л-52, Измайловский проспект, 29. Отпечатано в Ленинградской типографии № 8 ордена Трудового Красного Знамени Ленинградского объединения «Техническая книга» им. Евгении Соколовой Государственного комитета СССР по печати. 190000, Ленинград, Прачечный переулок, 6.