

Новинка



НОВЫЕ РАЗДЕЛЫ КЛАССИЧЕСКОГО ТРУДА

КнУт

Искусство программирования

Том 4 Выпуск 4

т 4
в 4

Искусство программирования

ТОМ 4

Генерация всех деревьев

История
комбинаторной
генерации

выпуск

4



ДОНАЛЬД Э. КНУТ



AW

ИСКУССТВО ПРОГРАММИРОВАНИЯ

ТОМ 4, ВЫПУСК 4

THE ART OF COMPUTER PROGRAMMING

VOLUME 4, FASCICLE 4

Generating All Trees — History of Combinatorial Generation

DONALD E. KNUTH *Stanford University*



ADDISON-WESLEY

Upper Saddle River, NJ · Boston · Indianapolis · San Francisco
New York · Toronto · Montréal · London · Munich · Paris · Madrid
Capetown · Sydney · Tokyo · Singapore · Mexico City

ИСКУССТВО ПРОГРАММИРОВАНИЯ

ТОМ 4, ВЫПУСК 4

**Генерация всех деревьев.
История комбинаторной генерации**

ДОНАЛЬД Э. КНУТ *Станфордский университет*



Москва · Санкт-Петербург · Киев
2007

ББК 32.973.26-018.2.75

К53

УДК 681.142.2

Издательский дом “Вильямс”

Зав. редакцией С.Н. Тригуб

Перевод с английского и редакция канд. техн. наук И.В. Красикова

По общим вопросам обращайтесь в Издательский дом “Вильямс” по адресу:

info@williamspublishing.com, <http://www.williamspublishing.com>

115419, Москва, а/я 783; 03150, Киев, а/я 152

Кнут, Дональд Э.

К53 Искусство программирования, том 4, выпуск 4. Генерация всех деревьев. История комбинаторной генерации. : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2007. — 160 с. : ил. — Парал. тит. англ.

ISBN 978-5-8459-1158-2 (рус.)

Эта книга представляет собой один из выпусков очередных томов всемирно известной работы *Искусство программирования*, не нуждающейся ни в представлении, ни в рекламе. В данный выпуск вошли разделы четвертого тома, посвященные вопросам генерации всех деревьев, а также обзор истории генерации различных комбинаторных объектов. Материалы выпуска в будущем войдут в четвертый том серии, посвященный комбинаторным алгоритмам, — возможно, с определенными дополнениями и исправлениями на основе отзывов читателей данного выпуска.

ББК 32.973.26-018.2.75

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения издательства Addison-Wesley Publishing Company, Inc.

Authorized translation from the English language edition published by Addison-Wesley Publishing Company, Inc, Copyright © 2006 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Rights to this book were obtained by arrangement with Addison-Wesley Longman, Inc.

Russian language edition published by Williams Publishing House according to the Agreement with R&I Enterprises International, Copyright © 2007

ISBN 978-5-8459-1158-2 (рус.)
ISBN 0-321-33570-8 (англ.)

© Издательский дом “Вильямс”, 2007
© by Pearson Education, Inc., 2006

Оглавление

Предисловие	7
7 Комбинаторный поиск	11
7.2 Генерация всех возможных объектов	11
7.2.1 Генерация основных комбинаторных объектов	11
7.2.1.1 Генерация всех n -кортежей	11
7.2.1.2 Генерация всех перестановок	11
7.2.1.3 Генерация всех сочетаний	11
7.2.1.4 Генерация всех разбиений	12
7.2.1.5 Генерация всех разбиений множеств	12
7.2.1.6 Генерация всех деревьев	12
7.2.1.7 Исторические и иные сведения	67
Ответы к упражнениям	99
Предметный указатель	146

От издательства

Вы, читатель этой книги, и есть главный ее критик и комментатор. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересно услышать и любые другие замечания, которые вам хотелось бы высказать в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо, либо просто посетить наш Web-сервер и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится или нет вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию последующих книг. Наши координаты:

E-mail: info@williamspublishing.com
WWW: <http://www.williamspublishing.com>

Информация для писем:

из России: 115419, Москва, а/я 783
из Украины: 03150, Киев, а/я 152

Предисловие

*Я люблю работать в самых разных областях — это позволяет мне
“размазать” мои ошибки более тонким слоем...*

— Виктор Кли (*Victor Klee*)

Эта брошюра представляет собой четвертый выпуск книги *Искусство программирования, том 4. Комбинаторные алгоритмы*. Как пояснялось в выпуске 1 к тому 1, я издаю материалы в такой форме в связи с тем, что работа над четвертым томом займет еще много лет. Я не могу заставлять читателей ждать так долго; кроме того, мне нужна обратная связь от читателей.

В этом выпуске содержатся разделы 7.2.1.6 и 7.2.1.7 очень большой главы, посвященной комбинаторному поиску. В окончательном виде глава 7 должна будет состоять из трех томов (4A, 4B и 4C) — конечно, если это позволит мое здоровье. Глава начнется с краткого обзора теории графов, после которого в разделе 7.1 будут рассматриваться вопросы работы с битами и алгоритмы для работы с булевыми функциями. Раздел 7.2 посвящен генерации всех возможных объектов и начинается с подраздела 7.2.1 — генерации основных комбинаторных объектов. В подразделах с 7.2.1.1 по 7.2.1.5 детально рассматриваются вопросы о генерации всех n -элементных кортежей, перестановок, сочетаний и разбиений. Выпуск 4 содержит разделы 7.2.1.6, посвященный генерации различных древовидных структур, и 7.2.1.7, темой которого является история комбинаторной генерации. Раздел 7.2.2 будет посвящен перебору с возвратом в целом. Набросок содержания всей главы 7 можно найти на Web-сайте книги *Искусство программирования* (<http://www-cs-faculty.stanford.edu/~knuth/taocp.html>).

Написание этого материала доставило мне такое же удовольствие, как испытанное мною много лет назад при написании второго тома *Искусства программирования*. При работе над ним я к своему немалому удовольствию обнаружил, что основные принципы теории вероятности и теории чисел естественным образом проявляются при изучении алгоритмов для генерации случайных чисел и арифметики; при подготовке к написанию раздела 7.2.1 я нашел, что основные принципы элементарной комбинаторики точно так же естественным образом проявляются при изучении алгоритмов комбинаторной генерации. Я еще раз убедился, что красивые истории всегда имеют продолжение.

Я уже давно хотел написать о генерации деревьев, потому что древовидные структуры занимают особое место в сердце каждого программиста. Хотя в целом

сделано для www.infanata.org

я доволен изложенным в разделах 7.2.1.1–7.2.1.5 материалом о классических комбинаторных структурах, таких, как кортежи, перестановки, сочетания и разбиения, все же самый лакомый кусочек я приберег напоследок. И вот наступило время для этого десерта. С 1994 года я читаю ежегодную “Лекцию о рождественской елке”¹ в Стенфордском университете, рассказывая о замечательных фактах, касающихся деревьев, о которых я узнал в течение последнего года, и вот наконец я могу изложить все эти лекции в письменном виде. Как и любой десерт, эта тема очень “сытная” и приносит немало удовольствия. Теория деревьев, помимо прочего, связывает воедино ряд концепций из разных областей программирования.

Раздел 7.2.1.7, посвященный истории комбинаторной генерации, принес особое удовлетворение другой, “гуманистарной”, половине моего мозга, поскольку он наполнен поэзией, музыкой, религиозными представлениями, философией, логикой и интеллектуальными играми из разных культур со всего света. Корни комбинаторного мышления очень глубоки. Я не претендую на полноту, но думаю, что достаточно полно изучил эту тему, чтобы собрать все изученное вместе.

Первоначально я намеревался выделить этому материалу более скромное место, но когда увидел, насколько фундаментальны собранные здесь идеи, то понял, что одно лишь беглое упоминание о них вряд ли удовлетворит меня.

Я благодарю Франка Раски (Frank Ruskey) за смелое решение предложить ранние черновые варианты этого материала студентам и за то, что он поделился со мной результатами этого эксперимента. Мне помогли и многие другие читатели моих черновиков, особенно относящихся к разделу 7.2.1.7, где мне часто была нужна помощь из-за ограниченного знания других языков.

Я с удовольствием уплачу 2 доллара 56 центов тому, кто первым сообщит мне о любой замеченной в данном выпуске² ошибке, неважно какой — типографской, технической или исторической. Все существенные предложения по улучшению текста я оцениваю в 32 цента каждое. (Кроме того, если вы найдете лучшее решение упражнения, то сможете покрыть себя неувядающей славой — ваше имя будет опубликовано в окончательном издании книги. :-)

Перекрестные ссылки на пока что не написанный материал могут выглядеть в тексте как ‘00’; это заполнитель, который будет заменен реальным числом в будущем.

Приятного чтения!
Стэнфорд, Калифорния
Июнь 2005
Д. Э. К.

Об обозначениях. В начале главы 7 я определил некоторые операции над графами, для которых в настоящее время в литературе встречаются разные обозначения. В моей книге, если G — граф с вершинами $U = \{u_1, \dots, u_m\}$, а H — граф с вершинами $V = \{v_1, \dots, v_n\}$, то:

- $G + H$ является суммой, или соединением G и H : этот граф содержит $Pm + n$ вершин $U \cup V$ и ребра G и H ;

¹Определенная игра слов — в оригинале “Christmas tree lecture”, т.е. о Рождественском дереве. Название можно также перевести и как “Лекция у Рождественской елки”. — Примеч. пер.

²Конечно же, имеется в виду оригинальное издание. — Примеч. пер.

- $G \mp H$ является объединением (сосуммой (cosum)) G и H , т.е. дополнением соединения их дополнений (его ребрами являются ребра G и H плюс все ребра $u_j - v_k$);
- $G \times H$ представляет собой декартово произведение G и H : в нем содержится $m n$ вершин $U \times V$; а ребрами являются $(u, v) - (u', v)$, когда G имеет ребро $u - u'$, и $(u, v) - (u, v')$, когда ребро $v - v'$ содержится в H ;
- $G \diamond H$ — прямое произведение, или конъюнкция, G и H : его вершинами являются $U \times V$, а ребро $(u, v) - (u', v')$ содержится в нем тогда и только тогда, когда в G имеется ребро $u - u'$, а в H — ребро $v - v'$;
- $G \otimes H$ — сильное произведение G и H , объединяющее ребра $G \times H$ и $G \diamond H$;

по аналогии с сосуммой имеются различные виды сопроизведений.

Прочие использованные в этом выпуске обозначения можно найти в разделах “Основные обозначения” в конце томов 1, 2 и 3. Естественно, аналогичный раздел будет и в томе 4.

Глава 7

Комбинаторный поиск



Начальные разделы этой главы представлены в выпусках 0 и 1 тома 4, запланированных к выпуску в 2006 и 2007 году.

7.2 Генерация всех возможных объектов

7.2.1 Генерация основных комбинаторных объектов

В этом разделе рассматриваются методы обхода всех возможных объектов некоторого комбинаторного универсума, поскольку часто приходится сталкиваться с задачами, в которых необходим или желателен исчерпывающий перебор...

7.2.1.1 Генерация всех n -кортежей

Начнем с малого — рассмотрим, как получить все 2^n строк из n бинарных цифр...

7.2.1.2 Генерация всех перестановок

Следующей по важности после генерации кортежей из n элементов является задача генерации всех *перестановок* некоторого заданного множества или мульти-множества...



Полностью тексты разделов 7.2.1.1 и 7.2.1.2 можно найти в выпуске 2 тома 4.

7.2.1.3 Генерация всех сочетаний

Комбинаторика часто определяется как “изучение перестановок, сочетаний и т.п.”, так что теперь обратим наше внимание на сочетания...

7.2.1.4 Генерация всех разбиений

Великолепная книга Ричарда Стенли (Richard Stanley) *Перечислительная комбинаторика* (*Enumerative Combinatorics*, 1986) начинается с рассмотрения “двенадцатизадачия” — массива размером $2 \times 2 \times 3$ фундаментальных комбинаторных задач, часто возникающих на практике...

В предыдущих разделах данной главы мы рассмотрели n -кортежи, перестановки, сочетания и композиции... Таким образом, мы можем завершить наше изучение классической комбинаторной математики рассмотрением оставшихся пяти записей таблицы, которые включают *разбиения* (*partitions*)...

Разбиения целого числа представляют собой способы записи его в виде суммы положительных целых чисел, независимо от их порядка...

7.2.1.5 Генерация всех разбиений множеств

Перейдем теперь к рассмотрению другого вида разбиений. *Разбиения множества* представляют собой способы рассматривать множество как объединение непустых непересекающихся подмножеств...

 Полностью тексты разделов 7.2.1.3, 7.2.1.4 и 7.2.1.5 можно найти в выпуске 3 тома 4.

*Поясните важность такой последовательности:
un, dos, tres, quatre, cinc, sis, set, vuit, nou, deu...¹*

— Ричард П. Стенли (Richard P. Stanley), *Перечислительная комбинаторика* (*Enumerative Combinatorics*) (1999)

... в едином теле имеются пары членов с одним и тем же именем, но
различаемые как левые и правые...

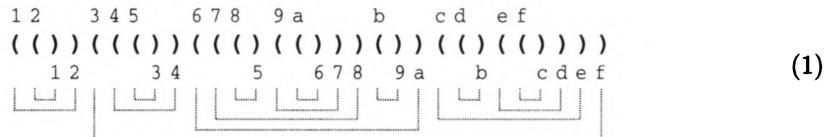
— Сократ (*Socrates*), *Phædrus* 266A (ок. 370 г. до н.э.)

7.2.1.6 Генерация всех деревьев

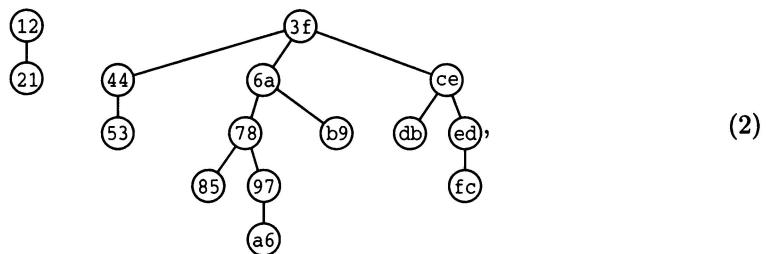
К этому моменту мы уже завершили изучение классических концепций комбинаторики: кортежей, перестановок, сочетаний и разбиений. Однако информатика внесла свой вклад, добавив еще один фундаментальный класс к традиционному репертуару, а именно иерархические конструкции, известные как деревья. Деревья встречаются в информатике повсюду, как видно из раздела 2.3 и практически каждого из последующих разделов *Искусства программирования*. Поэтому сейчас мы обратимся к изучению простых алгоритмов, с помощью которых можно исчерпывающе исследовать деревья различного вида.

¹Один, два, три... (фр.)

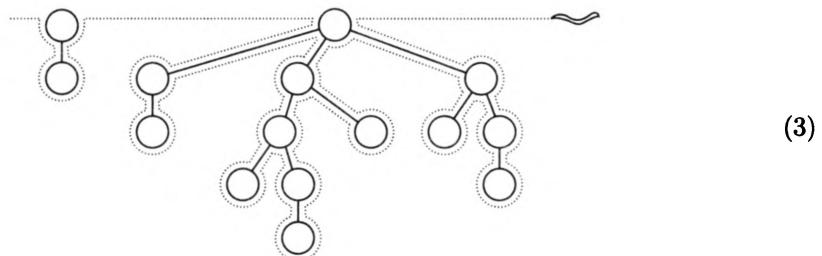
Сначала давайте рассмотрим связь между вложенными скобками и лесом деревьев. Например, строка



содержит 15 левых скобок ‘(’ с метками 1, 2, …, f и 15 правых скобок ‘)’, также помеченных от 1 до f. Серые линии под строками показывают, как соответствующие друг другу скобки образуют 15 пар 12, 21, 3f, 44, 53, 6a, 78, 85, 97, a6, b9, ce, db, ed и fc. Эта строка соответствует лесу

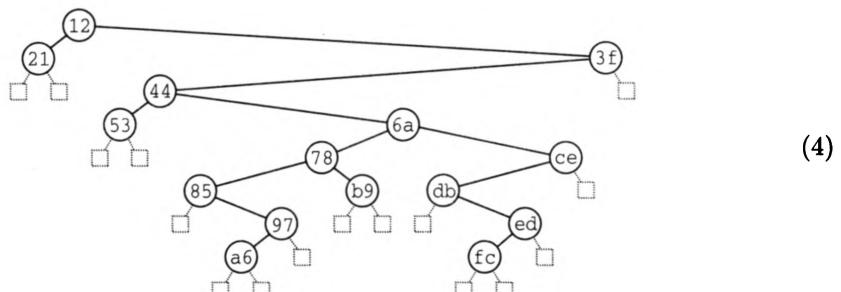


узлами которого являются (12) , (21) , $(3f)$, ..., (fc) в прямом порядке обхода (отсортированные по первой координате) и (21) , (12) , (53) , ..., $(3f)$ в обратном порядке обхода (отсортированные по второй координате). Если мы представим червяка, который



и видит, проползая по левой дуге узла, открывающую (левую) скобку ‘(’, а по правой — закрывающую (правую) скобку ‘)’, то его путь реконструирует исходную строку.

В свою очередь, лес (2) соответствует бинарному дереву



посредством “естественного соответствия”, рассмотренного в разделе 2.3.2. Здесь список узлов в *симметричном* порядке обхода — (21), (12), (53), …, (3f). Левое поддерево узла (x) в бинарном дереве представляет собой крайнего левого потомка (x) в лесу, или “внешний узел” \square , если (x) не имеет потомков. Правое поддерево (x) в бинарном дереве представляет собой правого “брата” в лесу, или “внешний узел” \square , если (x) — крайний справа потомок своего родителя. Корни деревьев в лесу рассматриваются как братья, и крайний слева корень в лесу является корнем бинарного дерева.

Строка скобок $a_1a_2\dots a_{2n}$ имеет корректную вложенность тогда и только тогда, когда содержит n символов ‘(’ и n символов ‘)’, где k -й символ ‘(’ предшествует k -му символу ‘)’ для $1 \leq k \leq n$. Простейший способ получить все строки вложенных скобок — посетить их в лексикографическом порядке. Далее приведен алгоритм, который рассматривает ‘)’ как лексикографически меньший символ по сравнению с ‘(’ и включает некоторые усовершенствования для повышения эффективности по сравнению с оригинальным алгоритмом И. Сембы (I. Semba) [Inf. Processing Letters, 12 (1981), 188–192].

Алгоритм Р (Вложенные скобки в лексикографическом порядке). Для данного целого $n \geq 2$ алгоритм генерирует все строки $a_1a_2\dots a_{2n}$ вложенных скобок.

- P1. [Инициализация.] Установить $a_{2k-1} \leftarrow ‘(’$ и $a_{2k} \leftarrow ‘)’$ для $1 \leq k \leq n$; установить также $a_0 \leftarrow ‘)’$ и $m \leftarrow 2n - 1$.
- P2. [Посещение.] Посетить строку вложенных скобок $a_1a_2\dots a_{2n}$. (В этот момент $a_m = ‘(’$ и $a_k = ‘)’$ для $m < k \leq 2n$.)
- P3. [Простой случай?] Установить $a_m \leftarrow ‘)’$. Затем, если $a_{m-1} = ‘)’$, установить $a_{m-1} \leftarrow ‘(’$, $m \leftarrow m - 1$ и вернуться к шагу P2.
- P4. [Поиск j .] Установить $j \leftarrow m - 1$ и $k \leftarrow 2n - 1$. Пока $a_j = ‘(’$, устанавливать $a_j \leftarrow ‘)’$, $a_k \leftarrow ‘(’$, $j \leftarrow j - 1$ и $k \leftarrow k - 2$.
- P5. [Увеличение a_j .] Завершить алгоритм, если $j = 0$. В противном случае установить $a_j \leftarrow ‘(’$, $m \leftarrow 2n - 1$ и вернуться к шагу P2. |

Позже мы увидим, что цикл на шаге P4 почти всегда короткий: операция $a_j \leftarrow ‘)’$ выполняется в среднем только около $\frac{1}{3}$ раза на одно посещение строки.

Почему работает алгоритм Р? Пусть A_{pq} — последовательность всех строк α , содержащих p левых скобок и $q \geq p$ правых (причем строка $(^{q-p}\alpha$ обладает корректной вложенностью), перечисленных в лексикографическом порядке. Тогда алгоритм Р предназначен для генерации A_{nn} , где, как легко видеть, A_{pq} подчиняются рекурсивным правилам

$$A_{pq} =) A_{p(q-1)}, (A_{(p-1)q} \text{ при } 0 \leq p \leq q \neq 0; \quad A_{00} = \varepsilon; \quad (5)$$

кроме того, A_{pq} пуста при $p < 0$ или $p > q$. Первым элементом A_{pq} является $)^{q-p}(\dots()$, где имеется p пар ‘()’; последний элемент равен $(^p)^q$. Таким образом, процесс лексикографической генерации состоит из сканирования справа в поисках завершающей строки вида $a_1\dots a_{2n} =)(^{p+1})^q$ и замене ее на $()^{q+1-p}(\dots()$. Шаги

Таблица 1. Вложенные скобки и сопутствующие объекты при $n = 4$

$a_1a_2\dots a_8$	Лес	Бинарное дерево	$d_1d_2d_3d_4$	$z_1z_2z_3z_4$	$p_1p_2p_3p_4$	$c_1c_2c_3c_4$	Соответствие
() () ()		1111	1357	1234	0000	
() () ()	.. :		1102	1356	1243	0001	
() () ()	· : ·		1021	1347	1324	0010	
() () ()	· : ·		1012	1346	1342	0011	
() () ()	· :		1003	1345	1432	0012	
() () ()	: ..		0211	1257	2134	0100	
() () ()	: :		0202	1256	2143	0101	
() () ()	: ·		0121	1247	2314	0110	
() () ()	· :		0112	1246	2341	0111	
() () ()	· :		0103	1245	2431	0112	
() () ()	: ·		0031	1237	3214	0120	
() () ()	: ·		0022	1236	3241	0121	
() () ()	· :		0013	1235	3421	0122	
() () ()	: :		0004	1234	4321	0123	

P4 и P5 эффективно выполняют данную задачу, а шаг P3 предназначен для обработки простого случая $p = 0$.

В табл. 1 проиллюстрирован выход алгоритма Р при $n = 4$ вместе с соответствующими лесами и бинарными деревьями, как в случаях (2) и (4). В таблице приведено и несколько других эквивалентных комбинаторных объектов. Например, строка вложенных скобок может быть закодирована как

$$()^{d_1} ()^{d_2} \dots ()^{d_n}, \quad (6)$$

где неотрицательные целые числа $d_1d_2\dots d_n$ характеризуются ограничениями

$$d_1 + d_2 + \dots + d_k \leq k \text{ для } 1 \leq k < n; \quad d_1 + d_2 + \dots + d_n = n. \quad (7)$$

Вложенные скобки могут быть также представлены последовательностью $z_1z_2\dots z_n$, которая определяет индексы появления левых скобок. По сути, $z_1z_2\dots z_n$ представляет собой одно из $\binom{2n}{n}$ сочетаний n элементов из множества $\{1, 2, \dots, 2n\}$, отвеча-

ющих определенным ограничениям

$$z_{k-1} < z_k < 2k \text{ для } 1 \leq k \leq n, \quad (8)$$

если считать, что $z_0 = 0$. Разумеется, все z связаны с d :

$$d_k = z_{k+1} - z_k - 1 \text{ для } 1 \leq k < n. \quad (9)$$

Алгоритм Р становится особенно простым, если переписать его для генерации сочетаний $z_1 z_2 \dots z_n$ вместо строк $a_1 a_2 \dots a_{2n}$ (см. упражнение 2).

Строка скобок может быть также представлена перестановкой $p_1 p_2 \dots p_n$, где k -я правая скобка соответствует p_k -й левой скобке; другими словами, k -й узел связанныго леса в обратном порядке обхода является p_k -м в прямом порядке обхода. (Согласно упражнению 2.3.2–20 узел j является потомком узла k в лесу тогда и только тогда, когда $j < k$ и $p_j > p_k$, если мы помечаем узлы в обратном порядке обхода.) Таблица инверсий $c_1 c_2 \dots c_n$ характеризует эту перестановку в соответствии с правилом, согласно которому справа от k имеется ровно c_k элементов, меньших k (см. упражнение 5.1.1–7); у допустимых таблиц инверсий $c_1 = 0$ и

$$0 \leq c_{k+1} \leq c_k + 1 \text{ для } 1 \leq k < n. \quad (10)$$

Кроме того, в упражнении 3 доказывается, что c_k представляет собой уровень, на котором в лесу находится k -й в прямом порядке обхода узел (глубина k -й левой скобки); этот факт эквивалентен формуле

$$c_k = 2k - 1 - z_k. \quad (11)$$

В табл. 1 и упражнении 6 проиллюстрирован специальный вид *соответствия* (matching), при котором $2n$ человек за круглым столом могут одновременно пожать руки, причем без перекрещивания.

Таким образом, алгоритм Р может оказаться полезным. Но если наша цель — генерация всех бинарных деревьев с n внутренними узлами, представленными левыми связями $l_1 l_2 \dots l_n$ и правыми связями $r_1 r_2 \dots r_n$, то лексикографическая последовательность в табл. 1 весьма неудобна; данные, которые требуются для перехода от одного дерева к следующему за ним, не будут легкодоступными. К счастью, существует остроумный альтернативный алгоритм для непосредственной генерации всех связанных бинарных деревьев.

Алгоритм В (Бинарные деревья). Для заданного $n \geq 1$ алгоритм генерирует все бинарные деревья с n внутренними узлами, представляя их с помощью левых связей $l_1 l_2 \dots l_n$ и правых связей $r_1 r_2 \dots r_n$, с узлами, помеченными в прямом порядке. (Таким образом, например, узел 1 — всегда корневой, а l_k всегда равно либо $k + 1$, либо 0; если $l_1 = 0$ и $n > 1$, то $r_1 = 2$.)

B1. [Инициализация.] Установить $l_k \leftarrow k + 1$ и $r_k \leftarrow 0$ для $1 \leq k < n$; установить также $l_n \leftarrow r_n \leftarrow 0$, и $l_{n+1} \leftarrow 1$ (для удобства на шаге B3).

B2. [Посещение.] Посетить бинарное дерево, представленное связями $l_1 l_2 \dots l_n$ и $r_1 r_2 \dots r_n$.

- B3.** [Поиск j .] Установить $j \leftarrow 1$. Пока $l_j = 0$, устанавливать $r_j \leftarrow 0$, $l_j \leftarrow j + 1$ и $j \leftarrow j + 1$. Затем прекратить выполнение алгоритма, если $j > n$.
- B4.** [Поиск k и y .] Установить $y \leftarrow l_j$ и $k \leftarrow 0$. Пока $r_y > 0$, устанавливать $k \leftarrow y$ и $y \leftarrow r_y$.
- B5.** [Продвижение y .] Если $k > 0$, установить $r_k \leftarrow 0$; в противном случае установить $l_j \leftarrow 0$. Затем установить $r_y \leftarrow r_j$, $r_j \leftarrow y$ и вернуться к шагу B2. ■

[См. W. Skarbek, *Theoretical Computer Science*, 57 (1988), 153–159; на шаге B3 используется идея Д. Корша (J. Korsh).] В упражнении 44 доказывается, что циклы на шагах B3 и B4 обычно достаточно коротки. Фактически в среднем требуется менее девяти обращений к памяти для преобразования связанного бинарного дерева в следующее за ним.

В табл. 2 показаны 14 бинарных деревьев, сгенерированных при $n = 4$, вместе с соответствующими лесами и двумя связанными последовательностями: $e_1e_2\dots e_n$ и $s_1s_2\dots s_n$, определяемыми тем свойством, что узел k в прямом порядке имеет e_k дочерних узлов и s_k потомков в связанном лесу. (Таким образом, s_k — размер k -го левого поддерева в бинарном дереве, а $s_k + 1$ — величина поля SCOPE в смысле 2.3.3–(5).) Следующий столбец повторяет 14 лесов из табл. 1 в лексикографическом порядке алгоритма P, но с зеркальным отражением по вертикали. Последний столбец показывает бинарные деревья, которые представляют лес в отраженном лексикографическом (солексном) порядке. Они также представляют лес из столбца 4, но с использованием связей с левым братом и правым потомком вместо левого потомка и правого брата. Последний столбец предоставляет интересную связь между вложенными скобками и бинарными деревьями, так что вносит определенный вклад в понимание того, почему алгоритм B корректно работает (см. упражнение 19).

***Коды Грея для деревьев.** Наш предыдущий опыт работы с другими комбинаторными объектами говорит нам, что, вероятно, мы можем генерировать строки скобок и деревья путем минимальных изменений между соседними экземплярами. И действительно, существует, как минимум, три очень красивых способа добиться этого.

Рассмотрим сначала случай вложенных скобок, которые могут быть представлены последовательностью $z_1z_2\dots z_n$, удовлетворяющей условию (8). “Близкий к идеальному” способ генерации всех таких сочетаний (см. раздел 7.2.1.3) представляет собой способ, при котором мы проходим по всем возможным сочетаниям так, что на каждом шаге изменяется только один компонент z_j , причем изменение равно ± 1 или ± 2 ; это означает, что из каждой строки скобок мы получаем следующую за ней путем простых изменений — либо (\leftrightarrow) , либо $(\leftrightarrow))$ (вблизи j -й левой скобки. Вот один из вариантов при $n = 4$:

1357, 1356, 1346, 1345, 1347, 1247, 1245, 1246, 1236, 1234, 1235, 1237, 1257, 1256.

Любое решение для $n - 1$ можно расширить для случая n , беря каждый шаблон $z_1z_2\dots z_{n-1}$ и позволяя z_n пройти по всем допустимым значениям с использованием *endo-порядка* или обратного к нему (см. 7.2.1.3–(45)), двигаясь вниз от $2n - 2$, а затем вверх до $2n - 1$ (или наоборот) и опуская все элементы, которые $\leq z_{n-1}$.

Таблица 2. Связанные бинарные деревья и сопутствующие объекты при $n = 4$

$l_1 l_2 l_3 l_4$	$r_1 r_2 r_3 r_4$	Бинарное дерево	Лес	$e_1 e_2 e_3 e_4$	$s_1 s_2 s_3 s_4$	Солексный лес	Левый брат/ правый потомок
2340	0000			1110	3210	
0340	2000			0110	0210	
2040	0300			2010	3010	.. : .	
2040	3000			1010	1010	: . .	
0040	2300			0010	0010	: . :	
2300	0040			1200	3200	.. : :	
0300	2040			0200	0200	: : :	
2300	0400			2100	3100	. : .	
2300	4000			1100	2100	: . .	
0300	2400			0100	0100	: . :	
2000	0340			3000	3000	. : .	
2000	4300			2000	2000	: . .	
2000	3040			1000	1000	: . .	
0000	2340			0000	0000	: . .	

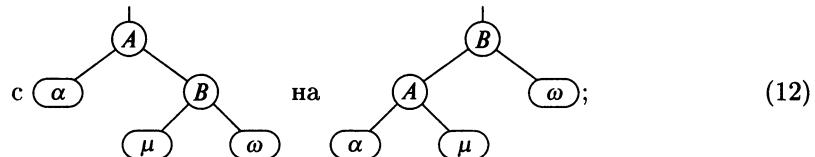
Алгоритм N (Вложенные скобки, близкие к идеальным). Алгоритм посещает все n -сочетания $z_1 z_2 \dots z_n$ элементов $\{1, \dots, 2n\}$, которые представляют собой индексы левых скобок в строке вложенных скобок, подчиняющиеся условию изменения индексов по одному. Процессом управляет вспомогательный массив $g_1 \dots g_n$, который представляет временные цели алгоритма.

- N1. [Инициализация.] Установить $z_j \leftarrow 2j - 1$ и $g_j \leftarrow 2j - 2$ для $1 \leq j \leq n$.
- N2. [Посещение.] Посетить n -сочетание $z_1 \dots z_n$. Затем установить $j \leftarrow n$.
- N3. [Поиск j .] Если $z_j = g_j$, установить $g_j \leftarrow g_j \oplus 1$ (тем самым обращая младший бит), $j \leftarrow j - 1$ и повторить этот шаг.
- N4. [Финал?] Если $g_j - z_j$ четно, установить $z_j \leftarrow z_j + 2$ и вернуться к шагу N2.

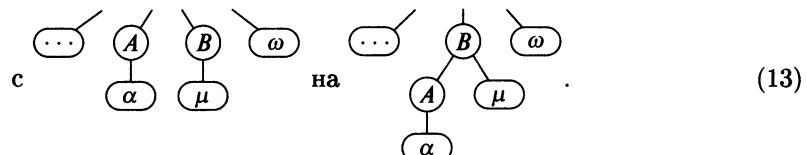
N5. [Уменьшение.] Установить $t \leftarrow z_j - 2$. Если $t < 0$, завершить работу алгоритма. В противном случае, если $t \leq z_{j-1}$, установить $t \leftarrow t + 2$ [$t < z_{j-1}] + 1$. Наконец, установить $z_j \leftarrow t$ и вернуться к шагу N2. |

[Похожий алгоритм был описан в работе D. Roelants van Baronaigien, *J. Algorithms*, **35** (2000), 100–107; см. также Xiang, Ushijima and Tang, *Inf. Proc. Letters*, **76** (2000), 169–174. Ф. Раски (F. Ruskey) и А. Проскуровски (A. Proskurowski) ранее в *J. Algorithms*, **11** (1990), 68–84, показали, как построить *идеальные* коды Грэя для всех таблиц $z_1 \dots z_n$ для четных $n \geq 4$ таким образом, чтобы на каждом шаге изменялось только одно значение z_j на ± 1 ; однако их построение весьма сложное, и неизвестна ни одна идеальная схема, достаточно простая для практического использования. В упражнении 48 показано, что идеальная схема невозможна для нечетных $n \geq 5$.]

Если наша цель состоит в генерации связанных древовидных структур, а не строк скобок, идеальности с точки зрения изменений z -индексов недостаточно, поскольку простые изменения наподобие $(\) \leftrightarrow (\)$ не обязательно соответствуют простым действиям со связями. Существенно лучший подход может быть основан на алгоритмах “поворотов”, с помощью которых мы поддерживали сбалансированность деревьев поиска в разделе 6.2.3. *Поворот влево* изменяет бинарное дерево



таким образом соответствующий лес изменяется



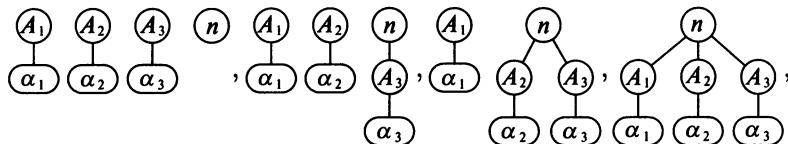
“Узел (A) становится крайним слева дочерним узлом своего правого брата”. *Поворот вправо*, разумеется, представляет противоположное преобразование: “крайний левый дочерний узел узла (B) становится его левым братом”. Вертикальная линия на рисунках (12) означает соединение с общим контекстом — левую или правую связь либо указатель на корень. Любое из поддеревьев α , μ или ω (или все они) может быть пустым. Троеточие ‘...’, в бинарном дереве (13), означающее дополнительных братьев в левой части семейства, содержащего (B) , также может быть пустым.

Приятная информация о поворотах заключается в том, что при этом изменяются только три связи: правая связь из (B) , левая из (B) и указатель сверху. Повороты сохраняют порядок симметричного обхода бинарного дерева и обратный порядок обхода леса. (Заметим также, что повороты бинарного дерева естественным образом соответствуют применению *ассоциативного закона*)

$$(\alpha\mu)\omega = \alpha(\mu\omega) \quad (14)$$

в алгебраической формуле.)

Для генерации всех бинарных деревьев или лесов посредством поворотов можно воспользоваться простой схемой, очень похожей на классический рефлексивный код Грэя для n -кортежей (алгоритм 7.2.1.1Н) и метод простых изменений для перестановок (алгоритм 7.2.1.2Р). Рассмотрим лес из $n - 1$ узлов, с k корнями (A_1, \dots, A_k) . В таком случае существует $k + 1$ лесов из n узлов, которые имеют одну и ту же последовательность первых $n - 1$ узлов в обратном порядке обхода с последней вершиной (n) ; например, при $k = 3$ это



полученные последовательными поворотами $(A_3), (A_2)$ и (A_1) влево. Кроме того, в крайних случаях, когда (n) находится справа или вверху, можно выполнять любой требующийся поворот над остальными $n - 1$ узлами, поскольку узел (n) не находится на указанном пути. Таким образом, как замечено в работе J.M. Lucas, D. Roelants van Baronaigien and F. Ruskey, *J. Algorithms*, 15 (1993), 343–366, можно расширить любой список $(n - 1)$ -узловых деревьев до списка n -узловых деревьев, просто позволив узлу (n) “прогуливаться” назад и вперед. Уделив внимание низкоуровневым деталям, можно выполнить указанную работу с замечательной эффективностью.

Алгоритм L (Связанные бинарные деревья путем поворотов). Алгоритм генерирует все пары массивов $l_0 l_1 \dots l_n$ и $r_1 \dots r_n$, представляющих собой левые и правые связи n -узловых бинарных деревьев, где l_0 — корень дерева, а связи (l_k, r_k) указывают соответственно левое и правое поддеревья k -го в симметричном порядке обхода узла. Каждое дерево получается из предшественника путем простого поворота. Два вспомогательных массива $k_1 \dots k_n$ и $o_0 o_1 \dots o_n$, представляющие обратные указатели и направления, используются для управления процессом.

- L1. [Инициализация.] Установить $l_j \leftarrow 0, r_j \leftarrow j + 1, k_j \leftarrow j - 1$ и $o_j \leftarrow -1$ для $1 \leq j < n$. Установить также $l_0 \leftarrow o_0 \leftarrow 1, l_n \leftarrow r_n \leftarrow 0, k_n \leftarrow n - 1$ и $o_n \leftarrow -1$.
- L2. [Посещение.] Посетить бинарное дерево или лес, представленные $l_0 l_1 \dots l_n$ и $r_1 \dots r_n$. Затем установить $j \leftarrow n$ и $p \leftarrow 0$.
- L3. [Поиск j .] Если $o_j > 0$, установить $m \leftarrow l_j$ и перейти к шагу L5, если $m \neq 0$. Если $o_j < 0$, установить $m \leftarrow k_j$; затем перейти к шагу L4, если $m \neq 0$; в противном случае установить $p \leftarrow j$. Если $m = 0$, в любом случае установить $o_j \leftarrow -o_j, j \leftarrow j - 1$ и повторить этот шаг.
- L4. [Поворот влево.] Установить $r_m \leftarrow l_j, l_j \leftarrow m, x \leftarrow k_m$ и $k_j \leftarrow x$. Если $x = 0$, установить $l_p \leftarrow j$, в противном случае установить $r_x \leftarrow j$. Вернуться к шагу L2.
- L5. [Поворот вправо.] Завершить работу программы, если $j = 0$. В противном случае установить $l_j \leftarrow r_m, r_m \leftarrow j, k_j \leftarrow m, x \leftarrow k_m$. Если $x = 0$, установить $l_p \leftarrow m$, в противном случае установить $r_x \leftarrow m$. Вернуться к шагу L2. |

В упражнении 38 доказывается, что алгоритму L требуется только около девяти обращений к памяти на генерацию одного дерева; таким образом, он почти такой же быстрый, как и алгоритм B. (Фактически можно сэкономить два обращения к памяти на шаг, если хранить количества деревьев o_n , l_n и k_n в регистрах. Но, конечно, так же может быть ускорен и алгоритм B.)

В табл. 3 показана последовательность бинарных деревьев и лесов, посещенных алгоритмом L при $n = 4$, с некоторыми вспомогательными таблицами, которые способствуют дальнейшему пониманию процесса. Перестановка $q_1q_2q_3q_4$ перечисляет узлы в прямом порядке обхода, в то время как они перечисляются в обратном порядке обхода в лесу (симметричном порядке в бинарном дереве); обратной к данной перестановке является перестановка $p_1p_2p_3p_4$ в табл. 1. “Со-лес” представляет собой сопряжение (отражение слева направо) леса; а числа $u_1u_2u_3u_4$ аналогичны $s_1s_2s_3s_4$ в табл. 2. В последнем столбце показан так называемый “дуальный лес”. Важность всех этих величин поясняется в упражнениях 11–13, 19, 24, 26 и 27.

Связи $l_0l_1\dots l_n$ и $r_1\dots r_n$ в алгоритме L и табл. 3 не сравнимы со связями $l_1\dots l_n$ и $r_1\dots r_n$ в алгоритме B и табл. 2, поскольку алгоритм L сохраняет симметрич-

Таблица 3. Бинарные деревья и леса, сгенерированные при помощи поворотов для $n = 4$

$l_0l_1l_2l_3l_4$	$r_1r_2r_3r_4$	$k_1k_2k_3k_4$	Бинарное дерево	Лес	$q_1q_2q_3q_4$	Со-лес	$u_1u_2u_3u_4$	Дуальный лес
10000	2340	0123		1234	0000	
10003	2400	0122		..;.	1243	;..	1000	
10002	4300	0121		.;..	1423	..;	2000	
40001	2300	0120		..;..	4123	..;..	3000	
40021	3000	0110		..;..	4132	..;..	3100	
10023	4000	0111		.;;.	1432	;..	2100	
10020	3040	0113		.;;.	1324	..;..	0100	
30010	2040	0103		..;..	3124	..;..	0200	
40013	2000	0100		..;..	4312	..;..	3200	
40123	0000	0000		;;..	4321	;;..	3210
30120	0040	0003		;;.	3214	..;..	0210	
20100	0340	0023		..;..	2134	..;..	0010	
20103	0400	0022		..;..	2143	..;..	1010	
40102	0300	0020		..;..	4213	..;..	3010	

ный/обратный порядок обхода, в то время как алгоритм В сохраняет прямой порядок обхода. Узел k в алгоритме L представляет собой k -й узел слева направо в бинарном дереве, так что для указания корня требуется значение l_0 ; в алгоритме В узел k представляет собой k -й узел в прямом порядке обхода, так что в этом случае корнем всегда является узел 1.

Алгоритм L обладает тем интересующим нас свойством, что на каждом шаге изменяются только три связи; но в действительности в этом отношении его можно улучшить, если придерживаться соглашения о прямом порядке обхода из алгоритма В. В упражнении 25 представлен алгоритм, который генерирует все связанные деревья или леса путем изменения только двух связей на каждом шаге, сохраняя прямой порядок обхода. Одна связь становится нулевой, в то время как вторая — ненулевой. Этот алгоритм “обрезки и прививки”, третий из обещанных красивых алгоритмов, имеет только один недостаток: его управляющий механизм существенно сложнее, чем у алгоритма L, так что ему требуется примерно на 40% больше времени на вычисления.

Количество деревьев. Существует простая формула для общего количества выводимых алгоритмами P, B, N и L деревьев, а именно:

$$C_n = \frac{1}{n+1} \binom{2n}{n} = \binom{2n}{n} - \binom{2n}{n-1}; \quad (15)$$

этот факт был доказан в формуле 2.3.4.4–(14). Первыми несколькими значениями являются

$$\begin{array}{ccccccccccccccccc} n & = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ C_n & = & 1 & 1 & 2 & 5 & 14 & 42 & 132 & 429 & 1430 & 4862 & 16796 & 58786 & 208012 & 742900 \end{array}$$

Они называются числами Каталана (Catalan numbers) — по имени Эжена Каталана (Eugène Catalan), который написал о них несколько важных статей [*Journal de math.*, 3 (1838), 508–516; 4 (1839), 95–99]. Приближение Стирлинга дает нам асимптотическое значение

$$C_n = \frac{4^n}{\sqrt{\pi n^{3/2}}} \left(1 - \frac{9}{8n} + \frac{145}{128n^2} - \frac{1155}{1024n^3} + \frac{36939}{32768n^4} + O(n^{-5}) \right); \quad (16)$$

в частности, мы можем сделать вывод, что

$$\frac{C_{n-k}}{C_n} = \frac{1}{4^k} \left(1 + \frac{3k}{2n} + O\left(\frac{k^2}{n^2}\right) \right) \text{ при } |k| \leq \frac{n}{2}. \quad (17)$$

(Само собой, C_{n-1}/C_n в точности равно $(n+1)/(4n-2)$ согласно формуле (15).) В разделе 2.3.4.4 была выведена также производящая функция

$$C(z) = C_0 + C_1 z + C_2 z^2 + C_3 z^3 + \dots = \frac{1 - \sqrt{1 - 4z}}{2z} \quad (18)$$

и доказана важная формула

$$[z^n] C(z)^r = \frac{r}{n+r} \binom{2n+r-1}{n} = \binom{2n+r-1}{n} - \binom{2n+r-1}{n-1}; \quad (19)$$

см. ответ к упражнению 2.3.4.4–33 и уравнение (5.70) в CMath.

Эти факты дают нам более чем достаточно информации для анализа алгоритма Р для лексикографической генерации вложенных скобок. Очевидно, что шаг Р2 выполняется C_n раз; затем шаг Р3 чаще всего выполняет простое изменение и возвращается к шагу Р2. Как часто происходит переход к шагу Р4? Все просто: это количество раз, когда на шаге Р2 $m = 2n - 1$. Величина m — это положение крайней правой скобки ‘(’, так что $m = 2n - 1$ ровно C_{n-1} раз. Таким образом, вероятность того, что на шаге Р3 устанавливается $m \leftarrow m - 1$ и происходит возврат к шагу Р2, равна $(C_n - C_{n-1})/C_n \approx 3/4$ согласно формуле (17). С другой стороны, пусть при переходе к шагу Р4 нам нужно установить $a_j \leftarrow ‘)’$ и $a_k \leftarrow ‘(’$ ровно $h - 1$ раз на каждом шаге. Количество случаев, когда $h > x$, равно количеству вложенных строк длиной $2n$, заканчивающихся тривиальными парами (\dots) в количестве x штук, а именно C_{n-x} . Таким образом, общее количество изменений a_j и a_k алгоритмом на шаге Р4 с учетом формулы (17) равно

$$\begin{aligned} C_{n-1} + C_{n-2} + \cdots + C_1 &= C_n \left(\frac{C_{n-1}}{C_n} + \frac{C_{n-2}}{C_n} + \cdots + \frac{C_1}{C_n} \right) = \\ &= \frac{1}{3} C_n \left(1 + \frac{2}{n} + O\left(\frac{1}{n^2}\right) \right); \end{aligned} \tag{20}$$

итак, сделанное ранее утверждение об эффективности данного алгоритма доказано.

Более глубокому пониманию способствует изучение рекурсивной структуры (5), лежащей в основе алгоритма Р. Последовательность A_{pq} в этой формуле имеет C_{pq} элементов, где

$$C_{pq} = C_{p(q-1)} + C_{(p-1)q} \text{ при } 0 \leq p \leq q \neq 0; \quad C_{00} = 1; \quad (21)$$

а при $p < 0$ или $p > q$ значение $C_{pq} = 0$. Так мы можем сформировать треугольный массив

$$\begin{array}{ccccccccc}
C_{00} & & & & & & 1 \\
C_{01} & C_{11} & & & & & 1 & 1 \\
C_{02} & C_{12} & C_{22} & & & & 1 & 2 & 2 \\
C_{03} & C_{13} & C_{23} & C_{33} & & = & 1 & 3 & 5 & 5 \\
C_{04} & C_{14} & C_{24} & C_{34} & C_{44} & & 1 & 4 & 9 & 14 & 14 \\
C_{05} & C_{15} & C_{25} & C_{35} & C_{45} & C_{55} & 1 & 5 & 14 & 28 & 42 & 42 \\
C_{06} & C_{16} & C_{26} & C_{36} & C_{46} & C_{56} & C_{66} & 1 & 6 & 20 & 48 & 90 & 132 & 132
\end{array} \tag{22}$$

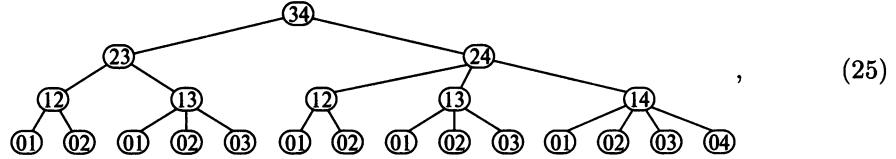
В нем каждый элемент представляет собой сумму ближайших соседей сверху и слева; на диагонали в нем находятся числа Каталана $C_n = C_{nn}$. Элементы этого треугольника имеют древнее происхождение, ведущее отсчет от работ Муавра в 1711 году, и называются “баллотировочными числами” (ballot numbers), поскольку представляют последовательности $p + q$ бюллетеней, для которых подсчет голосов ни в какой момент не покажет перевеса кандидата с p голосами над кандидатом с q голосами. Общую формулу

$$C_{pq} = \frac{q-p+1}{q+1} \binom{p+q}{p} = \binom{p+q}{p} - \binom{p+q}{p-1} \quad (23)$$

можно доказать как по индукции, так и многими другими способами (см. упражнение 39 и ответ к упражнению 2.2.1–4). Заметим, что согласно формуле (19) мы имеем

$$C_{pq} = [z^p] C(z)^{q-p+1}. \quad (24)$$

При $n = 4$ алгоритм Р, по сути, описывает дерево рекурсии



поскольку из спецификации (5) вытекает, что $A_{nn} = (A_{(n-1)n}$ и

$$A_{pq} =)^{q-p}(A_{(p-1)p},)^{q-p-1}(A_{(p-1)(p+1)},)^{q-p-2}(A_{(p-1)(p+2)}, \dots, (A_{(p-1)q}) \text{ при } 0 \leq p < q. \quad (26)$$

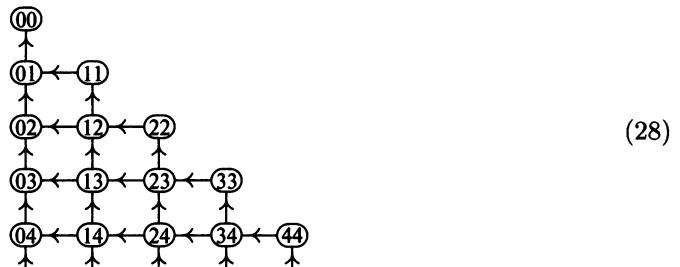
Количество листьев ниже узла (pq) в этом дереве рекурсии равно C_{pq} , а узел (pq) встречается ровно $C_{(n-q)(n-1-p)}$ раз на уровне $n - 1 - p$; таким образом, мы должны получить

$$\sum_q C_{(n-q)(n-1-p)} C_{pq} = C_n \text{ для } 0 \leq p < n. \quad (27)$$

Четырнадцать листьев (25) слева направо соответствуют четырнадцати строкам табл. 1 сверху вниз. Заметим, что элементы столбца $c_1 c_2 c_3 c_4$ в этой таблице назначают соответствующие числа 0000, 0001, 0010, ..., 0123 листьям дерева (25) в соответствии с “десятичной записью Дьюи” для узлов дерева (но с индексами, начинающимися с 0, а не с 1, и с дополнительным 0 в начале).

Червяк, который ползет от одного листа к следующему по низу дерева рекурсии, будет подниматься и опускаться на h уровней при изменении h координат $c_1 \dots c_n$, т.е. когда алгоритм Р сбрасывает значения h ‘(’ и h ‘)’. Это наблюдение облегчает понимание предыдущего вывода о том, что условие $h > x$ выполняется ровно C_{n-x} раз в процессе движения червяка.

Еще один способ понимания алгоритма Р возникает при рассмотрении бесконечного ориентированного графа, определяемого рекурсией (21):



Очевидно, что C_{pq} — количество путей от (pq) к (00) в этом ориентированном графе в силу (21). И действительно, каждая строка скобок A_{pq} непосредственно соответствует такому пути, где ‘(’ означает шаг влево, а ‘)’ — шаг вверх. Алгоритм Р

систематически исследует все такие пути, пытаясь сначала перейти вверх при продолжении частичного пути.

Таким образом, легко определить N -ю строку вложенных скобок, посещаемую алгоритмом Р, начиная с узла (nn) и выполняя следующие вычисления, находясь в узле (pq) : если $p = q = 0$, завершить работу; в противном случае, если $N \leq C_{p(q-1)}$, вывести ‘)’, установить $q \leftarrow q - 1$ и продолжить работу; в противном случае установить $N \leftarrow N - C_{p(q-1)}$, вывести ‘(’, установить $p \leftarrow p - 1$ и продолжить выполнение. Приведенный далее алгоритм [Frank Ruskey, Ph.D. thesis (University of California at San Diego, 1978), 16–24] избегает предвычисления треугольника Каталана путем вычисления C_{pq} “на ходу”.

Алгоритм U (Неранжированная строка вложенных скобок). Для заданных n и N , где $1 \leq N \leq C_n$, алгоритм вычисляет N -й вывод алгоритма Р.

- U1.** [Инициализация.] Установить $q \leftarrow n$ и $m \leftarrow p \leftarrow c \leftarrow 1$. Пока $p < n$, устанавливать $p \leftarrow p + 1$ и $c \leftarrow ((4p - 2)c)/(p + 1)$.
- U2.** [Выполнено?] Завершить работу алгоритма, если $q = 0$.
- U3.** [Вверх?] Установить $c' \leftarrow ((q + 1)(q - p)c)/((q + p)(q - p + 1))$. (В этот момент мы имеем $1 \leq N \leq c = C_{pq}$ и $c' = C_{p(q-1)}$.) Если $N \leq c'$, установить $q \leftarrow q - 1$, $c \leftarrow c'$, $a_m \leftarrow)$, $m \leftarrow m + 1$ и вернуться к шагу U2.
- U4.** [Влево?] Установить $p \leftarrow p - 1$, $c \leftarrow c - c'$, $N \leftarrow N - c'$, $a_m \leftarrow ($, $m \leftarrow m + 1$ и вернуться к шагу U3. |

Случайные деревья. Строку вложенных скобок $a_1a_2 \dots a_{2n}$ можно получить случайным образом, просто применения алгоритм U к случайному целому числу N между 1 и C_n . Однако эта идея не слишком хороша при n , превышающем 32 или около того, поскольку C_n может быть очень большим. Более простой способ, предложенный Д.Б. Арнольдом (D.B. Arnold) и М.Р. Слипом (M.R. Sleep) [ACM Trans. Prog. Languages and Systems, 2 (1980), 122–128] заключается в генерации случайного “путешествия червя”, начиная с узла (nn) в графе (28) и выполняя ветвления влево или вверх с соответствующими вероятностями. Полученный алгоритм практически такой же, как и алгоритм U, но работает только с неотрицательными положительными целыми числами, меньшими $n^2 + n + 1$.

Алгоритм W (Равномерно распределенные случайные строки вложенных скобок). Этот алгоритм генерирует случайную строку $a_1a_2 \dots a_{2n}$ корректно вложенных скобок.

- W1.** [Инициализация.] Установить $p \leftarrow q \leftarrow n$ и $m \leftarrow 1$.
- W2.** [Выполнено?] Завершить работу алгоритма, если $q = 0$.
- W3.** [Вверх?] Пусть X – случайное целое число в диапазоне $0 \leq X < (q + p)(q - p + 1)$. Если $X < (q + 1)(q - p)$, установить $q \leftarrow q - 1$, $a_m \leftarrow)$, $m \leftarrow m + 1$ и вернуться к шагу W2.
- W4.** [Влево.] Установить $p \leftarrow p - 1$, $a_m \leftarrow ($, $m \leftarrow m + 1$ и вернуться к шагу W3. |

Маршрут червя можно рассматривать как последовательность $w_0 w_1 \dots w_{2n}$, где w_m — текущая глубина червя после m шагов. Таким образом, $w_0 = 0$; $w_m = w_{m-1} + 1$, если $a_m = '('$, и $w_m = w_{m-1} - 1$, если $a_m = ')'$; мы также имеем $w_m \geq 0$, $w_{2n} = 0$. Последовательность $w_0 w_1 \dots w_{30}$, соответствующая (1) и (2), представляет собой 0121012321234345432321232343210. На шаге W3 алгоритма W мы имеем $q + p = 2n + 1 - m$ и $q - p = w_{m-1}$.

Назовем *контуром* (outline) леса путь, который проходит через точки $(m, -w_m)$ на плоскости $(0 \leq m \leq 2n)$, где $w_0 w_1 \dots w_{2n}$ — маршрут червя, соответствующий связанной строке $a_1 \dots a_{2n}$ вложенных скобок. На рис. 36 показано, что будет, если мы изобразим контуры всех 50-узловых лесов, причем степень затенения каждой точки соответствует количеству лесов, лежащих над ней. Например, w_1 всегда 1, так что треугольная область вверху слева на рис. 36 имеет максимально черный цвет; w_2 может принимать значения 0 либо 2, причем 0 встречается в $C_{49} \approx C_{50}/4$ случаях; соответственно ромбовидная область затенена на 75%. Таким образом, рис. 36 иллюстрирует форму случайного леса, аналогичную формам случайных разбиений, с которыми мы встречались на рис. 30, 31 и 35 в разделах 7.2.1.4 и 7.2.1.5.

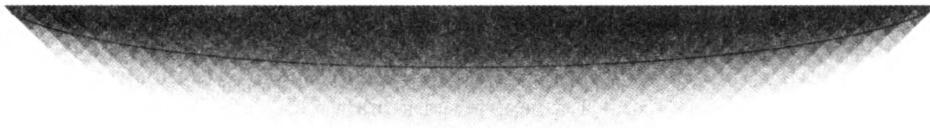


Рис. 36. Форма случайного 50-узлового леса

Конечно, в действительности мы не в состоянии изобразить контуры всех этих лесов, поскольку общее их количество равно $C_{50} = 1\ 978\ 261\ 657\ 756\ 160\ 653\ 623\ 774\ 456$. Но при помощи математики можно имитировать такое действие. Вероятность того, что $w_{2m} = 2k$ равна $C_{(m-k)(m+k)} C_{(n-m-k)(n-m+k)} / C_n$, поскольку имеется $C_{(m-k)(m+k)}$ способов начать с $m+k$ левых скобок и $m-k$ правых скобок и $C_{(n-m-k)(n-m+k)}$ способов закончить $n-(m+k)$ левыми скобками и $n-(m-k)$ правыми скобками. С использованием формулы (23) и приближения Стирлинга получаем, что эта вероятность равна

$$\begin{aligned} & \frac{(2k+1)^2 (n+1)}{(m+k+1)(n-m+k+1)} \binom{2m}{m-k} \binom{2n-2m}{n-m+k} \Big/ \binom{2n}{n} = \\ & = \frac{(2k+1)^2}{\sqrt{\pi} (\theta(1-\theta)n)^{3/2}} e^{-k^2/(\theta(1-\theta)n)} \left(1 + O\left(\frac{k+1}{n}\right) + O\left(\frac{k^3}{n^2}\right) \right), \quad (29) \end{aligned}$$

где $m = \theta n$ и $n \rightarrow \infty$, при $0 < \theta < 1$. Среднее значение w_{2m} выводится в упражнении 57; оно равно

$$\frac{(4m(n-m)+n) \binom{2m}{m} \binom{2n-2m}{n-m}}{n \binom{2n}{n}} - 1 = 4 \sqrt{\frac{\theta(1-\theta)n}{\pi}} - 1 + O\left(n^{-1/2}\right) \quad (30)$$

и показано на рис. 36 в виде кривой для $n = 50$.

Когда n велико, маршрут червя приближается к так называемому броуновскому движению, представляющему собой важную концепцию в теории вероятностей.

[См., например, Paul Lévy, *Processus Stochastiques et Mouvement Brownien* (1948), 225–237; Guy Louchard, *J. Applied Prob.*, **21** (1984), 479–499, и *BIT*, **26** (1986), 17–34; David Aldous, *Electronic Communications in Probability*, **3** (1998), 79–90; Jon Warren, *Electronic Communications in Probability*, **4** (1999), 25–29; J.-F. Marckert, *Random Structures and Algorithms*, **24** (2004), 118–132.]

Какой вид имеет случайное *бинарное дерево*? Этот вопрос был исследован Франком Раски (Frank Ruskey) [*SIAM J. Algebraic and Discrete Methods*, **1** (1980), 43–50], и ответ на него достаточно интересен. Предположим, что мы изобразили бинарное дерево как в (4), с m -м внутренним узлом в горизонтальной позиции m , где узлы пронумерованы в симметричном порядке. Если изобразить все 50-узловые бинарные деревья таким образом и наложить их одно на другое, то мы получим распределение положений узлов, показанное на рис. 37. Аналогично, если мы пронумеруем *внешние* узлы от 0 до n в симметричном порядке и разместим их в горизонтальных позициях .5, 1.5, …, $n + .5$, то “бахрома” от всех 50-узловых деревьев образует распределение, показанное на рис. 38. Обратите внимание, что корневой узел с наибольшей вероятностью имеет номер 1 или n , становясь крайним справа или слева; менее всего вероятно, что он находится посередине, в позициях $\lfloor(n+1)/2\rfloor$ или $\lceil(n+1)/2\rceil$.

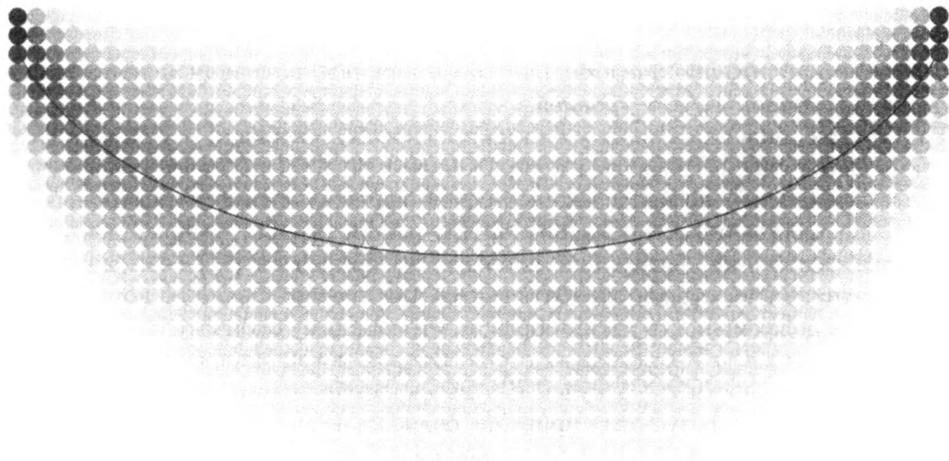


Рис. 37. Положения внутренних узлов в случайном 50-узловом бинарном дереве

Как и на рис. 36, гладкие кривые на рис. 37 и 38 показывают среднюю глубину узла; точные формулы выводятся в упражнениях 58 и 59. Асимптотически средняя глубина внешнего узла m равна

$$8\sqrt{\frac{\theta(1-\theta)n}{\pi}} - 1 + O\left(\frac{1}{\sqrt{n}}\right) \text{ при } m = \theta n \text{ и } n \rightarrow \infty \quad (31)$$

для всех фиксированных отношений $0 < \theta < 1$, что удивительно похоже на формулу (30); средняя глубина *внутреннего* узла m асимптотически та же, но ‘−1’ заменяется на ‘−3’. Таким образом мы можем сказать, что в *среднем* форма бинар-

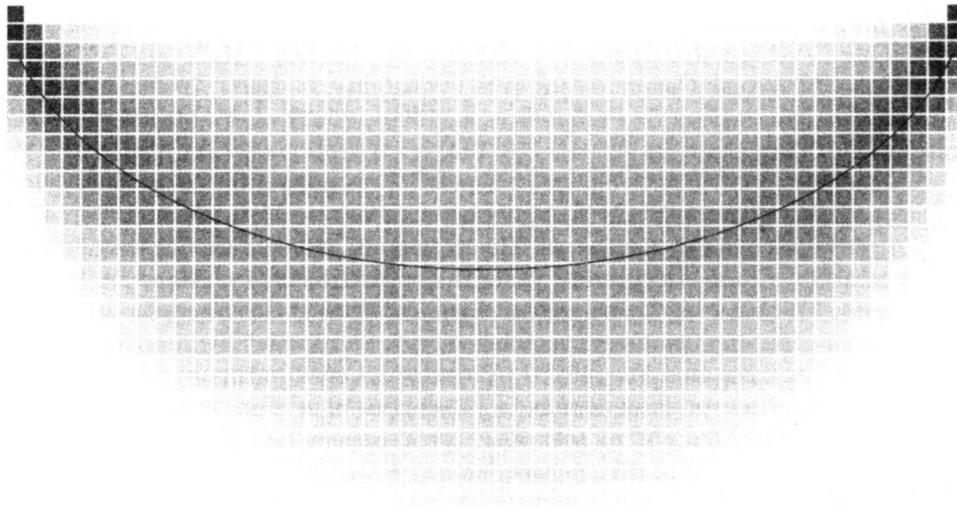


Рис. 38. Положения внешних узлов в случайном 50-узловом бинарном дереве

ного дерева приближенно представляет собой нижнюю половину эллипса шириной n единиц и глубиной $4\sqrt{n/\pi}$ уровней.

Три других заслуживающих упоминания способа генерации случайных закодированных лесов рассматриваются в упражнениях 60–62. Они менее непосредственны, чем алгоритм W, но представляют собой значительный комбинаторный интерес. Первый начинается с произвольной случайной строки, содержащей n левых и n правых скобок, не обязательно вложенных; каждая из $\binom{2n}{n}$ возможных строк равновероятна. Затем строка обрабатывается для преобразования в корректно вложенную последовательность таким образом, что каждому окончательному варианту соответствуют ровно $n + 1$ исходных строк. Второй метод аналогичен, но начинается с последовательности $n + 1$ нулей и n двоек, отображая их таким образом, что к каждому конечному результату приводят ровно $2n + 1$ исходных строк. Третий метод дает каждый конечный результат ровно из n битовых строк, содержащих $n - 1$ единиц и $n + 1$ нулей. Другими словами, эти три метода предоставляют комбинаторное доказательство того, что C_n одновременно равно $\binom{2n}{n} / (n + 1)$, $\binom{2n+1}{n} / (2n + 1)$ и $\binom{2n}{n-1} / n$. Например, при $n = 4$ мы получаем $14 = 70/5 = 126/9 = 56/4$.

Если мы хотим сгенерировать случайные бинарные деревья непосредственно в связанном виде, можно воспользоваться красивым методом, предложенным Ж.-Л. Реми (J.L. Rémy) [RAIRO Informatique Théorique, 19 (1985), 179–195]. Его подход особенно поучителен, поскольку показывает, как случайные деревья Каталана могут возникнуть “в природе”, с использованием удивительно простого механизма, основанного на классической идеи Олинде Родригеса (Olinde Rodrigues) [J. de Math., 3 (1838), 549]. Предположим, что наша цель — получение не только обычного n -узлового бинарного дерева, но и *декорированного* (decorated), т.е. расширенного бинарного дерева, в котором внешние узлы помечены числами от 0 до n в некотором порядке. Всего имеется $(n + 1)!$ способов декорировать любое данное бинарное дерево; таким образом, общее количество декорированных бинарных деревьев с n

внутренними узлами составляет

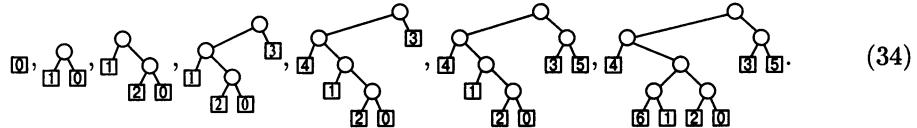
$$D_n = (n+1)! C_n = \frac{(2n)!}{n!} = (4n-2) D_{n-1}. \quad (32)$$

Реми обнаружил, что существует $4n-2$ простых пути построения декорированного дерева порядка n из данного декорированного дерева порядка $n-1$: мы просто выбираем любой узел из $2n-1$ узлов (внутренних или внешних) в данном дереве, например узел x , и заменяем его



таким образом вставляя новый внутренний узел и новый лист, перемещая x и его потомков (если таковые имеются) на один уровень вниз.

Например, вот один из вариантов построения декорированного дерева порядка 6:



Обратите внимание, что каждое декорированное дерево получается при помощи данного процесса единственным путем, поскольку предшественник каждого дерева должен быть деревом, которое получается путем вычеркивания листа с максимальным номером. Таким образом, построение Реми дает равномерно распределенные декорированные деревья; и если проигнорировать внешние узлы, то мы получим случайные бинарные деревья обычного, недекорированного вида.

Один привлекательный способ реализации процедуры Реми состоит в использовании таблицы связей $L_0 L_1 \dots L_{2n}$, в которой внешние узлы (листья) имеют четные номера, а внутренние (ветвления) — нечетные. Корневым узлом является узел L_0 ; левым и правым дочерними узлами внутреннего узла $2k-1$ являются соответственно L_{2k-1} и L_{2k} для $1 \leq k \leq n$. В этом случае программа получается короткой и приятной.

Алгоритм R (Растущее случайное бинарное дерево). Данный алгоритм конструирует представление с использованием связей $L_0 L_1 \dots L_{2N}$ (с указанными выше соглашениями) равномерно распределенного случайного бинарного дерева с N внутренними узлами.

R1. [Инициализация.] Установить $n \leftarrow 0$ и $L_0 \leftarrow 0$.

R2. [Выполнено?] (В этот момент связи $L_0 L_1 \dots L_{2n}$ представляют собой случайное n -узловое бинарное дерево.) Завершить работу алгоритма при $n = N$.

R3. [Продвижение.] Пусть X — случайное целое число между 0 и $4n+1$ включительно. Установить $n \leftarrow n+1$, $b \leftarrow X \bmod 2$, $k \leftarrow \lfloor X/2 \rfloor$, $L_{2n-b} \leftarrow 2n$, $L_{2n-1+b} \leftarrow L_k$, $L_k \leftarrow 2n-1$ и вернуться к шагу R2.

***Цепочки подмножеств.** Теперь, когда мы прочно усвоили получение деревьев и строк скобок, самое время перейти к рассмотрению *шаблона рождественской елки* (Christmas tree pattern)², который представляет собой замечательный способ размещения всех 2^n битовых строк длиной n в $\binom{n}{\lfloor n/2 \rfloor}$ строках и $n + 1$ столбцах, открытый де Брайном (de Bruijn), ван Эбенхорст Тенберген (van Ebbenhorst Tengbergen) и Круйсвейком (Kruyswijk) [Nieuw Archief voor Wiskunde, (2), **23** (1951), 191–193].

Шаблон рождественской елки порядка 1 представляет собой единственную строку ‘0 1’; шаблон порядка 2 имеет вид

$$\begin{array}{ccc} & 10 & \\ 00 & 01 & 11 \end{array} \quad (35)$$

В общем случае мы получаем шаблон рождественской елки порядка $n + 1$, беря каждую строку ‘ $\sigma_1 \sigma_2 \dots \sigma_s$ ’ шаблона порядка n и заменяя ее двумя строками.

$$\begin{array}{cccccc} \sigma_2 0 & \dots & \sigma_s 0 \\ \sigma_1 0 & \sigma_1 1 & \dots & \sigma_{s-1} 1 & \sigma_s 1 \end{array} \quad (36)$$

(При $s = 1$ первая из этих строк опускается.)

Действуя таким образом, мы получим пример шаблона порядка 8, приведенный в табл. 4. По индукции легко убедиться в следующем:

- i) каждая из 2^n битовых строк появляется в шаблоне ровно один раз;
- ii) битовые строки, содержащие k единиц, находятся в одном и том же столбце;
- iii) в пределах каждой строки каждая битовая строка получается из предыдущей путем единственного изменения 0 на 1.

Если рассматривать битовые строки как представляющие подмножества множества $\{1, \dots, n\}$, где единичные биты указывают члены множества, то свойство (iii) гласит, что каждая строка представляет *цепочку* (chain), в которой каждое подмножество покрывается следующим за ним. Используя символьную запись из раздела 7.1.3, можно записать: каждая строка $\sigma_1 \sigma_2 \dots \sigma_s$ обладает тем свойством, что $\sigma_j \subseteq \sigma_{j+1}$ и $\nu(\sigma_{j+1}) = \nu(\sigma_j) + 1$ при $1 \leq j < s$.

Свойства (i) и (ii) говорят о том, что если пронумеровать столбцы от 0 до n , то в столбце k имеется ровно $\binom{n}{k}$ элементов. С учетом того, что каждая строка отцентрирована по столбцам, это наблюдение доказывает, что общее количество строк, как и утверждалось, равно $\max_{0 \leq k \leq n} \binom{n}{k} = \binom{n}{\lfloor n/2 \rfloor}$. Обозначим эту величину как M_n .

Множество битовых строк C называется *клаттером* (clutter)³, или “антицепочной подмножеств”, если его битовые строки несравнимы в том смысле, что $\sigma \not\subseteq \tau$, где σ и τ — различные элементы C . Знаменитая теорема Эмануила Спернера (Emanuel Sperner) [Math. Zeitschrift, **27** (1928), 544–548] утверждает, что ни один клаттер на

²Это название выбрано из сентиментальных соображений, поскольку внешний вид шаблона напоминает елку, а кроме того, эта тема была предметом девятой ежегодной “лекции рождественской елки” автора в Стенфордском университете в декабре 2002 года.

³Дословно — помеха, шум. — Примеч. пер.

Таблица 4. Шаблон рождественской елки порядка 8

		10101010	10101001	10101011	
		10101000	10101100		
		10100100	10100101	10101101	
		10100010	10100110	10101110	
	10100000	10100001	10100011	10100111	10101111
		10110000	10110001	10110011	
		10110000	10110100		
		10010100	10010101	10110101	
		10010010	10010110	10110110	
	10010000	10010001	10010011	10010111	10110111
		10111000			
		10011000	10011001	10111001	
		10001010	10011010	10111010	
	10001000	10001001	10001011	10011011	10111011
		10001100	10011100	10111100	
		10000100	10000101	10001101	10111101
		10000010	10000110	10001110	10111110
10000000	10000001	10000011	10000111	10001111	10011111
		11001010			
		11001000	11001001	11001011	
		11000100	11000101	11001101	
		11000010	11000010	11000110	
	11000000	11000001	11000011	11000111	11001111
		11010010			
		11010000	11010001	11010011	
		11010000	11010100		
		01010100	01010101	11010101	
		01010010	01010110	11010110	
	01010000	01010001	01010011	01010111	11010111
		11011000			
		01011000	01011001	11011001	
		01001010	01011010	11011010	
	01001000	01001001	01001011	01011011	11011011
		01001100	01001100	11011100	
		01000100	01000101	01001101	01011101
		01000010	01000110	01001110	01011110
01000000	01000001	01000011	01000111	01001111	01011111
		11100010			
		11100000	11100001	11100011	
		11100000	11100100		
		01100100	01100101	11100101	
		01100010	01100110	11100110	
	01100000	01100001	01100011	01100111	11100111
		11101000			
		01101000	01101001	11101001	
		00101010	01101010	11101010	
	00101000	00101001	00101011	01101011	11101011
		00101100	00101100	11101100	
		00100100	00100101	00101101	01101101
		00100010	00100110	00101110	11101110
00100000	00100001	00100011	00100111	00101111	01101111
		11110000			
		01110000	01110001	11110001	
		00110010	01110010	11110010	
	00110000	00110001	00110011	01110011	11110011
		00110100	01110100	11110100	
		00010100	00010101	00110101	01110101
		00010010	00010110	00110110	01110110
00010000	00010001	00010011	00010111	00110111	01110111
		00111000	01111000	11111000	
		00011000	00011001	00111001	01111001
		00001010	00011010	00111010	11111010
	00001000	00001001	00001011	00111011	01111011
		00001100	00011100	00111100	11111100
		00000100	00000101	00001101	00111101
		00000010	00000110	00001110	00111110
00000000	00000001	00000011	00000111	00001111	00111111

множестве $\{1, \dots, n\}$ не может иметь больше M_n элементов; шаблон рождественской елки предоставляет простое доказательство этой теоремы, поскольку ни один кластер не может содержать более одного элемента из каждой строки.

На самом деле шаблоном рождественской елки можно воспользоваться для того, чтобы показать существенно большее. Сначала заметим, что имеется ровно $\binom{n}{k} - nk - 1$ строк длиной $n + 1 - 2k$ при $0 \leq k \leq n/2$, поскольку в столбце k ровно $\binom{n}{k}$ элементов. Например, в табл. 4 имеется одна строка длиной 9, а именно нижняя; в ней также $\binom{8}{1} - \binom{8}{0} = 7$ строк длиной 7, $\binom{8}{2} - \binom{8}{1} = 20$ строк длиной 5, $\binom{8}{3} - \binom{8}{2} = 28$ строк длиной 3 и $\binom{8}{4} - \binom{8}{3} = 14$ строк единичной длины. Кроме того, эти числа $\binom{n}{k} - \binom{n}{k-1}$ появляются в треугольнике Каталана (22), поскольку они равны $C_{k(n-k)}$ в соответствии с формулой (23).

Дальнейшее изучение показывает, что эта связь с числами Каталана — не простое совпадение; ключом к более глубокому пониманию шаблона рождественской елки являются вложенные скобки, потому что теория скобок говорит нам, где в массиве располагается произвольная битовая строка. Предположим, что вместо 0 и 1 мы используем символы (и) соответственно. Любая строка скобок, вложенных или нет, может быть записана единственным способом в виде

$$\alpha_0 \dots \alpha_{p-1} \alpha_p (\alpha_{p+1} \dots \alpha_q) \quad (37)$$

для некоторых p и q ($0 \leq p \leq q$), где подстроки $\alpha_0, \dots, \alpha_q$ корректно вложенные (возможно, пустые). Ровно p правых скобок и $q - p$ левых “свободны” в том смысле, что не имеют соответствующей пары. Например, у строки

$$)(())(())(())(((((())(())(())((), \quad (38)$$

$p = 5, q = 12, \alpha_0 = \varepsilon, \alpha_1 = ((())(), \alpha_2 = (), \alpha_3 = \varepsilon, \dots, \alpha_{12} = (()).$ В общем случае строка (37) представляет собой часть цепочки длиной $q + 1$,

$$\alpha_0 \dots \alpha_{q-1} \alpha_q, \alpha_0 \dots \alpha_{q-2} \alpha_{q-1} (\alpha_q, \dots, \alpha_0 (\alpha_1 \dots (\alpha_q, \quad (39)$$

которая начинается с q свободных правых скобок) и заменяет их по одной свободными левыми скобками (. Каждая строка шаблона рождественской елки получается точно таким способом, но с использованием 1 и 0 вместо (и); если цепочка $\sigma_1 \dots \sigma_s$ соответствует строкам вложенных цепочек $\alpha_0, \dots, \alpha_{s-1}$, то соответствующие цепочки (36) соответствуют строкам $\alpha_0, \dots, \alpha_{s-3}, \alpha_{s-2}(\alpha_{s-1})$ и $\alpha_0, \dots, \alpha_{s-3}, \alpha_{s-2}, \alpha_{s-1}, \varepsilon$. [См. Curtis Greene and Daniel J. Kleitman, *J. Combinatorial Theory*, **A20** (1976), 80–88.]

Обратите также внимание, что крайние справа элементы в каждой строке шаблона, такие, как 10101010, 10101011, 10101100, ..., 11111110, 11111111 в случае $n = 8$, находятся в лексикографическом порядке. Таким образом, например, 14 строк длиной 1 в табл. 4 в точности соответствуют 14 строкам вложенных скобок в табл. 1. Это наблюдение позволяет легко генерировать строки табл. 4 последовательно, снизу вверх, при помощи метода, аналогичного алгоритму Р (см. упражнение 77).

Пусть $f(x_1, \dots, x_n)$ — произвольная монотонная булева функция от n переменных. Если $\sigma = a_1 \dots a_n$ — произвольная битовая строка длиной n , для удобства можно записывать $f(\sigma) = f(a_1, \dots, a_n)$. Любая строка $\sigma_1 \dots \sigma_s$ шаблона рождественской елки образует цепочку, так что мы имеем

$$0 \leq f(\sigma_1) \leq \dots \leq f(\sigma_s) \leq 1. \quad (40)$$

Другими словами, имеется индекс t , такой, что $f(\sigma_j) = 0$ при $j < t$ и $f(\sigma_j) = 1$ при $j \geq t$; нам будут известны значения $f(\sigma)$ для всех 2^n битовых строк σ , если мы узнаем индексы t для каждой строки шаблона.

Джордж Хансель (Georges Hansel) [*Comptes Rendus Acad. Sci. (A)*, **262** (Paris, 1966), 1088–1090] заметил, что шаблон рождественской елки обладает еще одним важным свойством: если σ_{j-1} , σ_j и σ_{j+1} представляют собой три последовательных элемента любой строки, битовая строка

$$\sigma'_j = \sigma_{j-1} \oplus \sigma_j \oplus \sigma_{j+1} \quad (41)$$

находится в *предыдущей* строке. В действительности σ'_j находится в том же столбце, что и σ_j , и удовлетворяет условию

$$\sigma_{j-1} \subseteq \sigma'_j \subseteq \sigma_{j+1}; \quad (42)$$

это значение называется относительным дополнением (relative complement) σ_j в интервале $(\sigma_{j-1}.. \sigma_{j+1})$. Наблюдение Ханселя легко доказать по индукции при помощи рекурсивного правила (36), определяющего шаблон рождественской елки. Он воспользовался им для того, чтобы показать, что можно вывести значения $f(\sigma)$ для всех σ , вычисляя значения функции в относительно небольшом количестве мест; если известно значение $f(\sigma'_j)$, то в силу (42) известно либо $f(\sigma_{j-1})$, либо $f(\sigma_{j+1})$.

Алгоритм H (Исследование монотонной булевой функции). Пусть $f(x_1, \dots, x_n)$ — булева функция, неубывающая по каждой из булевых переменных (более о ней ничего не известно). Обозначим для данной битовой строки σ длиной n через $r(\sigma)$ номер строки, в которой σ находится в шаблоне рождественской елки; очевидно, что $1 \leq r(\sigma) \leq M_n$. Для $1 \leq m \leq M_n$ обозначим через $s(m)$ количество битовых строк в строке m ; а $\chi(m, k)$ будет означать битовую строку в столбце k этой строки при $(n + 1 - s(m))/2 \leq k \leq (n - 1 + s(m))/2$. Данный алгоритм определяет последовательность пороговых значений $t(1), t(2), \dots, t(M_n)$, таких, что

$$f(\sigma) = 1 \Leftrightarrow \nu(\sigma) \geq t(r(\sigma)), \quad (43)$$

путем вычисления f не более чем в двух точках на строку.

H1. [Цикл по m .] Выполнить шаги с H2 по H4 для $m = 1, \dots, M_n$; после этого завершить выполнение алгоритма.

H2. [Начальная строка m .] Установить $a \leftarrow (n + 1 - s(m))/2$ и $z \leftarrow (n - 1 + s(m))/2$.

H3. [Бинарный поиск.] Если $z \leq a + 1$, перейти к шагу H4. В противном случае установить $k \leftarrow \lfloor (a + z)/2 \rfloor$ и

$$\sigma \leftarrow \chi(m, k - 1) \oplus \chi(m, k) \oplus \chi(m, k + 1). \quad (44)$$

Если $k \geq t(r(\sigma))$, установить $z \leftarrow k$; в противном случае установить $a \leftarrow k$. Повторить шаг H3.

H4. [Вычисление.] Если $f(\chi(m, a)) = 1$, установить $t(m) \leftarrow a$; в противном случае, если $a = z$, установить $t(m) \leftarrow a + 1$; в противном случае установить $t(m) \leftarrow z + 1 - f(\chi(m, z))$. ■

Алгоритм Ханселя *оптимальен* в том смысле, что вычисляет f в наименьшем возможном количестве точек в наихудшем случае. Если f представляет собой пороговую функцию

$$f(\sigma) = [\nu(\sigma) > n/2], \quad (45)$$

любой корректный алгоритм, который исследует f на первых m строках шаблона рождественской елки, должен вычислять $f(\sigma)$ в столбце $\lfloor n/2 \rfloor$ каждой строки и в столбце $\lfloor n/2 \rfloor + 1$ каждой строки, размер которой больше 1. В противном случае невозможно отличить f от функции, которая не совпадает с ней только в неисследованных точках. [См. V.K. Korobkov, *Problemy Kibernetiki*, **13** (1965), 5–28, теорема 5.]

Ориентированные деревья и леса. Обратимся теперь к другому виду деревьев, в которых важны отношения “родитель–потомок”, но не порядок потомков в каждом семействе. *Ориентированный лес* из n узлов можно определить как последовательность указателей $p_1 \dots p_n$, где p_j — родитель узла j (если j — корень, $p_j = 0$); ориентированный граф с вершинами $\{0, 1, \dots, n\}$ и ребрами $\{j \rightarrow p_j | 1 \leq j \leq n\}$ не имеет ориентированных циклов. *Ориентированное дерево* представляет собой ориентированный лес с единственным корнем (см. раздел 2.3.4.2). Каждый ориентированный лес из n узлов эквивалентен $(n+1)$ -узловому ориентированному дереву, поскольку корень этого дерева можно рассматривать как родителя для всех корней леса. В разделе 2.3.4.4 мы видели, что имеется A_n ориентированных деревьев с n узлами. Вот несколько первых значений A_n :

$$\begin{array}{cccccccccccccc} n & = & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ A_n & = & 1 & 1 & 2 & 4 & 9 & 20 & 48 & 115 & 286 & 719 & 1842 & 4766 & 12486 & 32973 \end{array} \quad (46)$$

Асимптотически $A_n = c\alpha^n n^{-3/2} + O(\alpha^n n^{-5/2})$, где $\alpha \approx 2.9558$ и $c \approx 0.4399$. Так, например, только 9 из 14 лесов в табл. 1 различны, если игнорировать порядок по горизонтали и рассматривать только вертикальную ориентацию.

Каждый ориентированный лес соответствует единственному упорядоченному лесу, если мы соответствующим образом отсортируем члены каждого семейства с использованием упорядочения деревьев, предложенного Г.Я. Скоинсом (H.I. Scains) [*Machine Intelligence*, **3** (1968), 43–60]: вспомним из (11), что упорядоченные леса могут характеризоваться кодами их уровней $c_1 \dots c_n$, где c_j указывает уровень, на котором в лесу находится j -й в прямом порядке обхода узел. Упорядоченный лес называется *каноническим*, если последовательность кодов уровней поддеревьев в каждом семействе находится в невозрастающем лексикографическом порядке. Например, каноническими лесами в табл. 1 являются те, у которых коды уровней $c_1 c_2 c_3 c_4$ равны 0000, 0100, 0101, 0110, 0111, 0120, 0121, 0122 и 0123. Последовательность 0112 канонической не является, поскольку поддеревья корня имеют соответственно коды уровней 1 и 12; строка 1 лексикографически меньше 12. По индукции можно легко убедиться, что *канонические коды уровней лексикографически наибольшие среди всех способов переупорядочения поддеревьев данного ориентированного леса*.

Т. Бейер (T. Beyer) и С.М. Хедетниеми (S.M. Hedetniemi) [*SICOMP*, **9** (1980), 706–712] заметили, что существует удивительно простой способ генерации ориентированных лесов, если посещать их в *уменьшающемся* лексикографическом порядке канонических кодов уровней. Предположим, что $c_1 \dots c_n$ — канонический код, в котором $c_k > 0$ и $c_{k+1} = \dots = c_n = 0$. Следующая наименьшая последовательность

получается путем уменьшения c_k и увеличения $c_{k+1} \dots c_n$ до наибольших уровней, согласующихся с условием каноничности; эти уровни достаточно просто вычислить. Если $j = p_k$ — родительский по отношению к узлу k узел, получаем $c_j = c_k - 1 < c_l$ для $j < l \leq k$, а следовательно, уровни $c_j \dots c_k$ представляют поддерево, корнем которого является узел j . Для получения наибольшей последовательности уровней, меньших $c_1 \dots c_n$, мы заменяем $c_k \dots c_n$ первыми $n + 1 - k$ элементами бесконечной последовательности $(c_j \dots c_{k-1})^\infty = c_j \dots c_{k-1} c_j \dots c_{k-1} c_j \dots$. (Результатом будет удаление k из его текущей позиции крайнего справа потомка j и добавление новых поддеревьев, являющихся “братьями” j , путем клонирования j и его наследников настолько часто, насколько это возможно. Этот процесс клонирования может завершиться посередине последовательности $c_j \dots c_{k-1}$, но это не приведет к сложностям, поскольку каждый префикс канонической последовательности является каноническим.) Например, чтобы получить последующий элемент для любой последовательности канонических кодов, заканчивающейся на 234434330000000000, необходимо заменить окончание 3000000000 на 2344343234.

Алгоритм О (Ориентированные леса). Данный алгоритм генерирует все ориентированные леса из n узлов путем посещения всех канонических n -узловых лесов в уменьшающемся лексикографическом порядке их кодов уровней $c_1 \dots c_n$. Коды уровней, однако, явно не вычисляются; каждый канонический лес представляется непосредственно последовательностью родительских указателей $p_1 \dots p_n$ в порядке прямого обхода узлов. Для генерации всех ориентированных деревьев из $n + 1$ узлов можно представить, что корнем является узел 0.

- O1. [Инициализация.] Установить $p_k \leftarrow k - 1$ для $0 \leq k \leq n$. (В частности, этот шаг делает ненулевым p_0 для использования при проверке на завершение алгоритма; см. шаг O4.)
- O2. [Посещение.] Посетить лес, представленный родительскими указателями $p_1 \dots p_n$.
- O3. [Простой случай?] Если $p_n > 0$, установить $p_n \leftarrow p_{p_n}$ и вернуться к шагу O2.
- O4. [Поиск j и k .] Найти наибольшее $k < n$, такое, что $p_k \neq 0$. Завершить работу алгоритма, если $k = 0$; в противном случае установить $j \leftarrow p_k$ и $d \leftarrow k - j$.
- O5. [Клонирование]. Если $p_{k-d} = p_j$, установить $p_k \leftarrow p_j$; в противном случае установить $p_k \leftarrow p_{k-d} + d$. Вернуться к шагу O2, если $k = n$; в противном случае установить $k \leftarrow k + 1$ и повторить данный шаг. ■

Как и в других рассмотренных алгоритмах циклы на шагах O4 и O5 обычно достаточно коротки (см. упражнение 88). В упражнении 90 доказывается, что небольших изменений данного алгоритма достаточно для генерации всех размещений ребер, формирующих *свободные* деревья.

Остовные деревья. Теперь рассмотрим минимальные подграфы, которые “охватывают” данный граф. Если G — связный граф из n вершин, его *остовными деревьями* (spanning trees) являются подмножества из $n - 1$ ребер, не содержащие циклов; или, что эквивалентно, они представляют собой подмножества ребер, которые

образуют свободное дерево, соединяющее все вершины. Остовные деревья играют важную роль во многих приложениях, особенно при изучении сетей, так что задача генерации всех остовных деревьев рассматривалась многими авторами. Фактически систематические способы их перечисления разработаны в начале 20-го века В. Фосснером (W. Feussner) [*Annalen der Physik*, (4) 9 (1902), 1304–1329], задолго до того, как стали задумываться о генерации других типов деревьев.

В дальнейшем рассмотрении мы допускаем, что граф может содержать любое количество ребер между двумя вершинами; однако петли (ребра, выходящие и возвращающиеся в одну и ту же вершину) запрещены, поскольку петля не может быть частью дерева. Основная идея Фосснера очень проста и хорошо подходит для вычислений: если e — произвольное ребро графа G , то остовное дерево либо содержит его, либо нет. Предположим, что ребро e соединяет вершину u с вершиной v , и пусть оно является частью остовного дерева; тогда остальные $n - 2$ ребер этого дерева покрывают граф G/e , который мы получим, рассматривая вершины u и v как идентичные. Другими словами, остовные деревья, которые содержат e , по сути же, что и остовные деревья сжатого графа G/e , который получается в результате сжатия ребра e в точку. С другой стороны, остовные деревья, которые не содержат e , являются остовными деревьями уменьшенного графа $G \setminus e$, полученного в результате удаления ребра e . Таким образом, множество $S(G)$ всех остовных деревьев G удовлетворяет соотношению

$$S(G) = eS(G/e) \cup S(G \setminus e). \quad (47)$$

В своей диссертации в университете Виктории (1999) Малькольм Д. Смит (Malcolm J. Smith) разработал красивый способ использования рекурсии (47) для поиска всех остовных деревьев в порядке “кода Грэя двери-вертушки”: каждое дерево в его схеме получается из предшествующего путем простого удаления одного ребра и подстановки другого. Такой порядок деревьев найти нетрудно, проблема в том, чтобы сделать это эффективно.

Основная идея алгоритма Смита состоит в генерации $S(G)$ таким образом, что первое остовное дерево включает данное *близкое дерево* (near tree), состоящее из $n - 2$ ребер и не содержащее циклов. Эта задача тривиальна при $n = 2$; мы просто перечисляем все ребра. При $n > 2$ и данном близком дереве $\{e_1, \dots, e_{n-2}\}$ мы действуем следующим образом. Считаем, что граф G связный, в противном случае остовных деревьев не существует. Образуем G/e_1 и добавляем e_1 к каждому из его остовных деревьев, начиная с того, которое содержит $\{e_2, \dots, e_{n-2}\}$; заметим, что $\{e_2, \dots, e_{n-2}\}$ — близкое дерево для G/e_1 , так что эта рекурсия имеет смысл. Если последним найденным таким путем остовным деревом для G/e_1 является $f_1 \dots f_{n-2}$, завершим работу перечислением всех остовных деревьев для $G \setminus e_1$, начиная с того, которое содержит близкое дерево $\{f_1, \dots, f_{n-2}\}$.

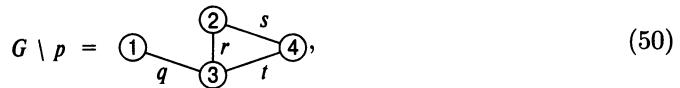
Предположим, например, что G — граф



с четырьмя вершинами и пятью ребрами $\{p, q, r, s, t\}$. Начиная с близкого дерева $\{p, q\}$, процедура Смита сначала образует сжатый граф



и перечисляет его оставные деревья, начиная с того, которое содержит q . Это может быть список qs, qt, ts, tr, rs ; таким образом, деревья pqs, pqt, pts, ptr и prs покрывают G . Остается только перечислить оставные деревья графа



начиная с того, которое содержит $\{r, s\}$; это деревья rsq , rqt , qts .

Детальная реализация алгоритма Смита весьма поучительна. Как обычно, мы представляем граф, используя для представления каждого ребра $u \rightarrow v$ двух дуг⁴ — $u \rightarrow v$ и $v \rightarrow u$, и поддерживая список “узлов дуг” для представления дуг, покидающих каждую вершину. Нам потребуется уменьшать и увеличивать количество ребер графа, так что эти списки лучше сделать дважды связанными. Если a указывает на узел дуги, представляющий $u \rightarrow v$, то

- $a \oplus 1$ указывает “напарника” a , который представляет дугу $v \rightarrow u$;
 - t_a является “верхушкой” a , т.е. v (следовательно, $t_{a \oplus 1} = u$);
 - i_a представляет собой необязательное имя, которое идентифицирует данное ребро (эквивалентно $i_{a \oplus 1}$);
 - n_a указывает на следующий элемент в списке дуг u ;
 - p_a указывает на предыдущий элемент в списке дуг u ;
 - l_a представляет собой связь, используемую для восстановления дуг, как поясняется ниже.

Вершины представлены целыми числами $\{1, \dots, n\}$; номер дуги $v - 1$ представляет собой головной узел дважды связанного списка для вершины v . Головной узел a распознается по тому факту, что его верхушка t_a (см. пояснение выше) равна 0. Обозначим степень вершины v как d_v . Так, например, граф (48) может быть представлен при помощи $(d_1, d_2, d_3, d_4) = (2, 3, 3, 2)$ и следующих 14 узлов дуг:

a	=	0	1	2	3	4	5	6	7	8	9	10	11	12	13
t_a	=	0	0	0	0	1	2	1	3	2	3	2	4	3	4
i_a	=					p	p	q	q	r	r	s	s	t	t
n_a	=	5	4	6	10	9	7	8	0	13	11	12	1	3	2
p_a	=	7	11	13	12	1	0	2	5	6	4	3	9	10	8

⁴Следуя автору, мы используем для обозначения направленного ребра термин “дуга”. — Примеч. пер.

Управлять неявной рекурсией алгоритма Смита удобно при помощи массива указателей на дуги $a_1 \dots a_{n-1}$. На уровне l процесса дуги $a_1 \dots a_{l-1}$ означают ребра, которые были включены в текущее оствовное дерево; a_l игнорируется, а дуги $a_{l+1} \dots a_{n-1}$ обозначают ребра близкого дерева сжатого графа $(\dots(G/a_1)\dots)/a_{l-1}$, которое должно быть частью следующего оствовного дерева.

Существует и другой массив указателей на дуги $s_1 \dots s_{n-2}$, представляющий стеки дуг, временно удаленных из текущего графа. Верхний элемент стека для уровня l — s_l , и каждая дуга a связывается со следующей за ней, l_a (которая равна 0 на дне стека).

Ребро, удаление которого разъединяет связный граф, называется *мостом* (bridge). Одним из ключевых моментов приведенного далее алгоритма является то, что мы хотим сохранять текущий граф связным; следовательно, мы не можем устанавливать $G \leftarrow G \setminus e$, когда e является мостом.

Алгоритм S (Все оствовные деревья). Данный алгоритм посещает все оствовные деревья заданного связного графа, представленного структурами данных, которые поясняются выше.

Для удаления и восстановления элементов в дважды связанных списках здесь используется метод “танцующих связей”, который будет всесторонне рассмотрен в разделе 7.2.2.1. Обозначение “ $\text{delete}(a)$ ” в шагах алгоритма представляет собой сокращенную запись пары операций

$$n_{p_a} \leftarrow n_a, \quad p_{n_a} \leftarrow p_a; \tag{51}$$

аналогично этому, “ $\text{undelete}(a)$ ” означает

$$p_{n_a} \leftarrow a, \quad n_{p_a} \leftarrow a. \tag{52}$$

S1. [Инициализация.] Установить $a_1 \dots a_{n-1}$ равным оствовному дереву графа (см. упражнение 94). Установить также $x \leftarrow 0$, $l \leftarrow 1$ и $s_1 \leftarrow 0$. Если $n = 2$, установить $v \leftarrow 1$, $e \leftarrow n_0$ и перейти к шагу S5.

S2. [Вход на уровень l .] Установить $e \leftarrow a_{l+1}$, $u \leftarrow t_e$ и $v \leftarrow t_{e \oplus 1}$. Если $d_u > d_v$, обменять $v \leftrightarrow u$ и установить $e \leftarrow e \oplus 1$.

S3. [Сжатие e .] (Сейчас u будет сделано идентичным v путем вставки списка смежности u в список v . Следует также удалить все бывшие ребра между u и v , включая ребро e , поскольку иначе такие ребра станут петлями. Удаленные ребра связываются вместе, так что мы сможем восстановить их на шаге S7.) Установить $k \leftarrow d_u + d_v$, $f \leftarrow n_{u-1}$ и $g \leftarrow 0$. Пока $t_f \neq 0$, выполняем следующее: если $t_f = v$, выполнить $\text{delete}(f)$, $\text{delete}(f \oplus 1)$ и установить $k \leftarrow k - 2$, $l_f \leftarrow g$, $g \leftarrow f$; в противном случае установить $t_{f \oplus 1} \leftarrow v$. Затем установить $f \leftarrow n_f$ и повторять эти действия до тех пор, пока не будет выполнено условие $t_f = 0$. Наконец, установить $l_e \leftarrow g$, $d_v \leftarrow k$, $g \leftarrow v - 1$, $n_{p_f} \leftarrow n_g$, $p_{n_g} \leftarrow p_f$, $p_{n_f} \leftarrow g$, $n_g \leftarrow n_f$ и $a_l \leftarrow e$.

S4. [Продвижение l .] Установить $l \leftarrow l + 1$. Если $l < n - 1$, установить $s_l \leftarrow 0$ и вернуться к шагу S2. В противном случае установить $e \leftarrow n_{v-1}$.

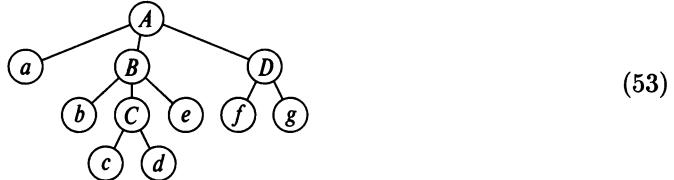
- S5.** [Посещение.] (Текущий граф в настоящий момент имеет только две вершины, одна из которых v .) Установить $a_{n-1} \leftarrow e$ и посетить оствное дерево $a_1 \dots a_{n-1}$. (Если $x = 0$, это первое посещаемое оствное дерево; в противном случае оно отличается от своего предшественника удалением x и вставкой e .) Установить $x \leftarrow e$ и $e \leftarrow n_e$. Повторить шаг S5, если $t_e \neq 0$.
- S6.** [Уменьшение l .] Установить $l \leftarrow l - 1$. Завершить работу алгоритма, если $l = 0$; в противном случае установить $e \leftarrow a_l$, $u \leftarrow t_e$ и $v \leftarrow t_{e \oplus 1}$.
- S7.** [Восстановление e .] Установить $f \leftarrow u - 1$, $g \leftarrow v - 1$, $n_g \leftarrow n_{p_f}$, $p_{n_g} \leftarrow g$, $n_{p_f} \leftarrow f$, $p_{n_f} \leftarrow f$ и $f \leftarrow p_f$. Пока $t_f \neq 0$, устанавливать $t_{f \oplus 1} \leftarrow u$ и $f \leftarrow p_f$. Затем установить $f \leftarrow l_e$, $k \leftarrow d_v$; пока $f \neq 0$, выполнять следующее: установить $k \leftarrow k + 2$, выполнить $\text{undelete}(f \oplus 1)$, $\text{undelete}(f)$ и установить $f \leftarrow l_f$. Наконец, установить $d_v \leftarrow k - d_u$.
- S8.** [Проверка моста.] Если e — мост, перейти к шагу S9. (См. один из способов выполнения данной проверки в упражнении 95.) В противном случае установить $x \leftarrow e$, $l_e \leftarrow s_l$, $s_l \leftarrow e$; выполнить $\text{delete}(e)$ и $\text{delete}(e \oplus 1)$. Установить $d_u \leftarrow d_u - 1$, $d_v \leftarrow d_v - 1$ и перейти к шагу S2.
- S9.** [Отмена удалений на уровне l .] Установить $e \leftarrow s_l$. Пока $e \neq 0$, выполнять следующие действия: установить $u \leftarrow t_e$, $v \leftarrow t_{e \oplus 1}$, $d_u \leftarrow d_u + 1$, $d_v \leftarrow d_v + 1$, выполнить $\text{undelete}(e \oplus 1)$ и $\text{undelete}(e)$, и установить $e \leftarrow l_e$. Вернуться к шагу S6. ■

Вы можете выполнить шаги алгоритма вручную на небольшом графе, таком, как (48). Обратите внимание на тонкий момент, возникающий на шагах S3 и S7, когда список смежности u становится пустым. Обратите также внимание на то, что возможны некоторые сокращения ценой усложнения алгоритма; такие улучшения алгоритма мы рассмотрим позже в этом разделе.

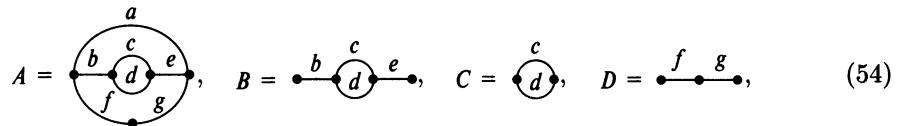
***Последовательно-параллельные графы.** Задача поиска всех оствных деревьев становится особенно простой, когда данный график имеет последовательное и/или параллельное разложение. *Последовательно-параллельный график* (*series-parallel graph*) между s и t представляет собой график G с двумя указанными вершинами s и t , ребра которого могут быть рекурсивно построены следующим образом: G либо состоит из единственного ребра $s-t$; либо представляет собой *последовательное сверхребро* (*serial superedge*), состоящее из $k \geq 2$ последовательно-параллельных подграфов G_j между s_j и t_j , соединенных в ряд, такой, что $s = s_1$ и $t_j = s_{j+1}$ для $1 \leq j < k$, и $t_k = t$; либо представляет собой *параллельное сверхребро* (*parallel superedge*), состоящее из $k \geq 2$ последовательно-параллельных подграфов G_j между s и t , соединенных параллельно. Такое разложение для данных s и t единственное, если потребовать, чтобы подграфы G_j последовательных сверхребер сами не являлись последовательными сверхребрами, а подграфы G_j параллельных сверхребер не являлись параллельными сверхребрами.

Любой последовательно-параллельный график удобно представить в виде дерева, у которого нет узлов степени 1. Листья дерева представляют ребра, а внутренние узлы — сверхребра, причем последовательные и параллельные сверхребра чередуются

от уровня к уровню. Например, дерево



соответствует последовательно-параллельным графам и подграфам



если верхний узел A рассматривать как параллельный. В (54) поименованы ребра, но не вершины, поскольку именно ребра играют главную роль в оставных деревьях.

Скажем, например, что *близкое дерево* последовательно-параллельного графа между s и t представляет собой множество из $n - 2$ ребер без циклов, которые не соединяют s с t . Оставные деревья и близкие деревья последовательно-параллельного графа легко описать рекурсивно следующим образом. 1. Оставное дерево последовательного сверхребра соответствует оставным деревьям всех его главных подграфов G_j ; близкое дерево соответствует оставным деревьям всех подграфов, кроме одного, у которого используется близкое дерево. 2. Близкое дерево параллельного сверхребра соответствует близким деревьям всех его главных подграфов G_j ; оставное дерево соответствует близким деревьям всех подграфов, кроме одного, у которого используется оставное дерево.

Правила 1 и 2 наводят на мысль об использовании следующих структур данных для перечисления оставных деревьев и/или близких деревьев последовательно-параллельных графов. Пусть p указывает на узел в дереве наподобие (53). Тогда определим

- $t_p = 1$ для последовательных сверхребер, 0 в противном случае (“тип” p);
- $v_p = 1$ в случае оставного дерева для p , и 0 в случае близкого дерева;
- l_p — указатель на крайнего левого потомка p , или 0, если p — лист;
- r_p — указатель на правого “брата” p (по циклу);
- d_p — указатель на выбранного потомка p , или 0, если p — лист.

Если q указывает на крайнего правого потомка p , то его “правый брат” r_q равен l_p . И если q указывает на произвольного потомка p , правила 1 и 2 гласят, что

$$v_q = \begin{cases} v_p, & \text{если } q = d_p; \\ t_p, & \text{если } q \neq d_p. \end{cases} \quad (55)$$

(Например, если p — внутренний узел, представляющий последовательное сверхребро, то для всех, кроме одного, потомков p получаем $v_q = 1$; единственным исключением является выбранный потомок d_p . Таким образом, у нас должно быть

остовное дерево для всех последовательно соединенных подграфов, образующих p , за исключением одного выбранного подграфа, и в этом случае мы имеем для p близкое дерево.)

Для заданных выбранных указателей d_p и произвольного значения 0 или 1 для v_p в корне дерева формула (55) говорит о том, каким образом происходит распространение этих значений вниз по дереву вплоть до листьев. Например, если установить в дереве (53) $v_A \leftarrow 1$ и выбранным потомком каждого внутреннего узла считать крайний слева (т.е. $d_A = a$, $d_B = b$, $d_C = c$ и $d_D = f$), то можно последовательно найти значения:

$$v_a = 1, v_B = 0, v_b = 0, v_C = 1, v_c = 1, v_d = 0, v_e = 1, v_D = 0, v_f = 0, v_g = 1. \quad (56)$$

Лист q присутствует в оствомном дереве тогда и только тогда, когда $v_q = 1$; следовательно, (56) определяет оствомное дерево *асег* последовательно-параллельного графа A из (54).

Для удобства будем говорить, что *конфигурациями* (*configs*) p являются его оствомные деревья, если $v_p = 1$, и его близкие деревья, если $v_p = 0$. Мы бы хотели сгенерировать все конфигурации корня. Внутренний узел p называется “простым”, если $v_p = t_p$; т.е. последовательный узел прост, если его конфигурациями являются оствомные деревья, а параллельный узел прост, если его конфигурациями являются близкие деревья. Если внутренний узел p прост, его конфигурации представляют собой декартово произведение конфигураций его потомков, а именно все независимо изменяющиеся k -кортежи дочерних конфигураций; выбранный потомок d_p в этом простом случае несущественен. Однако, если p не является простым, его конфигурации представляют собой объединение таких декартовых k -кортежей, взятых по всем возможным выборам d_p .

Простые узлы относительно редки: простым может быть максимум один потомок узла, не являющегося простым (а именно выбранный потомок), и все потомки простого узла не являются простыми, если только они не листья.

Даже при этом представление последовательно-параллельного графа в виде дерева делает рекурсивную генерацию всех его оствомных и/или близких деревьев достаточно простой и эффективной. Операции алгоритма S — сжатие и восстановление, удаление и его отмена, проверка моста — не требуются при работе с последовательно-параллельными графиками. Более того, в упражнении 90 показано, что имеется красивый способ получить оствомные деревья или близкие деревья в порядке кода Грея двери-вертушки, с использованием фокусных указателей, как в некоторых ранее встречавшихся нам алгоритмах.

***Усовершенствования алгоритма S.** Хотя алгоритм S предоставляет нам простой и достаточно эффективный способ посещения всех оствомных деревьев графа в общем случае, его автор Мальcolm Смит (Malcolm Smith) выяснил, что свойства последовательно-параллельных графов позволяют улучшить его. Например, если график имеет два или более ребер, проходящих между одними и теми же вершинами u и v , их можно скомбинировать в сверхребро; оствомные деревья исходного графа в таком случае могут быть получены из этого более простого графа. А если график имеет вершину v степени 2, так что с ней связаны только ребра $u-v$ и $v-w$, то можно удалить вершину v и заменить указанные ребра одним сверхребром между

u и *w*. Кроме того, можно удалить любую вершину степени 1 вместе с ее ребром, просто включая его в каждое оставное дерево.

Применив описанные приведения к данному графу G , мы получим приведенный граф \hat{G} , не имеющий параллельных ребер и вершин степени 1 и 2, и множество $m \geq 0$ последовательно-параллельных графов S_1, \dots, S_m , представляющих ребра (или сверхребра), которые должны быть включены во все оставные деревья G . Каждое оставшееся ребро $u-v$ графа \hat{G} фактически соответствует последовательно-параллельному графу S_{uv} между вершинами *u* и *v*. Оставные деревья графа G могут быть получены как объединение, взятое по всем оставным деревьям T графа \hat{G} , декартова произведения оставных деревьев графов S_1, \dots, S_m и оставных деревьев всех графов S_{uv} для ребер $u-v$ в T , вместе с близкими деревьями всех графов S_{uv} для ребер $u-v$, которые входят в \hat{G} , но не в T . Все оставные деревья T графа \hat{G} можно получить с использованием стратегии из алгоритма S .

Фактически при таком применении алгоритма S его операции по замене текущего графа G графом G/e или $G\backslash e$ обычно влекут за собой дальнейшие приведения, если появляются новые параллельные ребра или степень некоторых вершин графа падает ниже 3. Таким образом, это приводит к тому, что “останавливающее состояние” неявной рекурсии алгоритма S , а именно ситуация, когда остаются только две вершины (шаг $S5$), в действительности никогда не достигается: приведенный график \hat{G} либо состоит из единственной вершины без ребер, либо имеет, как минимум, четыре вершины и шесть ребер.

Производительность алгоритма S и его улучшенной версии — алгоритма S' — можно оценить, рассматривая количество обращений к памяти, выполняемых этими алгоритмами при генерации всех оставных деревьев типичных графов, как показано в табл. 5. Нижняя строка таблицы соответствует графу *plane_miles* (16, 0, 0, 1, 0, 0, 0) из Stanford GraphBase, который служит “естественным” противоядием для чисто математических примеров в предыдущих строках. Случайный мультиграф в предпоследней строке, также взятый из Stanford GraphBase, более точно можно описать его официальным именем *random_graph* (16, 37, 1, 0, 0, 0, 0, 0, 0). Хотя тор 4×4 изоморфен четырехмерному кубу (см. упражнение 7.2.1.1–17), эти изоморфные графы дают немного отличающееся время работы из-за того, что их вершины и ребра алгоритм обходит в разном порядке.

В общем случае алгоритм S не так уж плох для небольших примеров, за исключением разреженных графов; однако алгоритм S' превосходит его при наличии большого количества оставных деревьев. Когда алгоритм S' входит в полную силу, на каждое новое дерево он тратит примерно 18–19 обращений к памяти.

В табл. 5 также указывается, что математически определенные графы часто имеют на удивление “круглое” количество оставных деревьев. Например, Д.М. Цветкович (D.M. Cvetković) [Srpska Akademija Nauka, Matematičeski Institut, 11 (Belgrade, 1971), 135–141] открыл, помимо прочего, что n -мерный куб имеет ровно

$$2^{2^n-n-1} 1^{\binom{n}{1}} 2^{\binom{n}{2}} \dots n^{\binom{n}{n}} \quad (57)$$

оставных деревьев. В упражнениях 104–109 рассматриваются некоторые из причин этого.

Таблица 5. Рабочее время в обращениях к памяти, необходимое для генерации всех остовных деревьев

	<i>m</i>	<i>n</i>	<i>N</i>	Алгоритм S	Алгоритм S'	μ на одно дерево
Путь P_{10}	9	10	1	794 μ	473 μ	794.0
Путь P_{100}	99	100	1	9 974 μ	5 063 μ	9 974.0
Цикл C_{10}	10	10	10	3 480 μ	998 μ	348.0
Цикл C_{100}	100	100	100	355 605 μ	10 538 μ	3 556.1
Полный граф K_4	6	4	16	1 213 μ	1 336 μ	75.8
Полный граф K_{10}	45	10	100 000 000	3 759.58 M μ	1 860.95 M μ	37.6
Полный диграф $K_{5,5}$	25	10	390 625	23.43 M μ	8.88 M μ	60.0
4×4 решетка $P_4 \times P_4$	24	16	100 352	12.01 M μ	1.87 M μ	119.7
5×5 решетка $P_5 \times P_5$	40	25	557 568 000	54.68 G μ	10.20 G μ	98.1
4×4 цилиндр $P_4 \times C_4$	28	16	2 558 976	230.96 M μ	49.09 K μ	90.3
5×5 цилиндр $P_5 \times C_5$	45	25	38 720 000 000	3 165.31 G μ	711.69 G μ	81.7
4×4 тор $C_4 \times C_4$	32	16	42 467 328	3 168.15 M μ	823.08 M μ	74.6
Четырехмерный куб $P_2 \times P_2 \times P_2 \times P_2$	32	16	42 467 328	3 168.16 M μ	823.38 M μ	74.7
Случайный мультиграф	37	16	59 933 756	3 818.19 M μ	995.91 M μ	63.7
16 городов	37	16	179 678 881	11 772.11 M μ	3 267.43 M μ	65.5
						18.2

Обобщенный квазикод Грэя. Завершим данный раздел обсуждением еще одного вопроса, совершенно отличного от предыдущих, тем не менее связанного с деревьями. Рассмотрим гибридные варианты двух стандартных способов обхода непустого леса.

Обход в прямо-обратном порядке (prepostorder)

Посетить корень первого дерева

Обойти поддеревья первого дерева в обратно-прямом порядке

Обойти оставшиеся деревья в прямо-обратном порядке

Обход в обратно-прямом порядке (postpreorder)

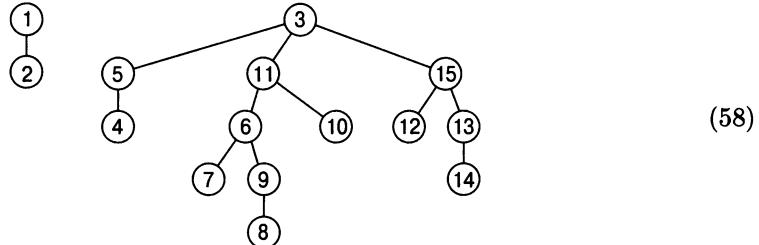
Обойти поддеревья первого дерева в прямо-обратном порядке

Посетить корень первого дерева

Обойти оставшиеся деревья в обратно-прямом порядке

В первом случае каждое дерево леса обходится в прямо-обратном порядке, причем первым посещается корень дерева; однако поддеревья этих корней обходятся в обратно-прямом порядке, причем последним посещается корень. Второй вариант аналогичен, но в нем “прямой” и “обратный” заменены друг на друга. В общем случае

прямо-обратный порядок посещает корни на каждом четном уровне леса первыми, а на нечетном — последними. Например, лес (2) превращается в



при нумерации его узлов в прямо-обратном порядке.

Прямо-обратный и обратно-прямой порядки не просто курьез — они приносят реальную пользу. Это связано с тем, что соседние узлы при любом из этих порядков всегда находятся близко друг к другу в лесу. Например, узлы k и $k+1$ являются смежными в (58) при $k = 1, 4, 6, 8, 10, 13$; они разделены только одним узлом при $k = 2, 5, 7, 9, 11$ (если представить невидимый сверхродительский узел на вершине леса). Минутное размыщение позволяет доказать по индукции, что между соседями как в прямо-обратном, так и в обратно-прямом порядке может располагаться не более двух узлов, поскольку обратно-прямой порядок всегда начинается с корня первого дерева или его крайнего слева потомка, а прямо-обратный порядок всегда заканчивается корнем последнего дерева или его крайним справа потомком.

Предположим, что мы хотим сгенерировать все комбинаторные шаблоны некоторого вида и посетить их в порядке наподобие кода Грея, так чтобы последовательные шаблоны всегда были “близки” друг к другу. Можно сформировать, по крайней мере концептуально, граф всех возможных шаблонов p с ребрами $p-q$ для всех пар шаблонов, близких друг к другу. Следующая теорема, открытая Миланом Секаниной (Milan Sekanina) [Spisy Přírodrovědecké Fakulty University v Brně, No. 412 (1960), 137–140], доказывает, что всегда возможен хороший код Грея, обеспечивающий получение одного шаблона из другого при помощи последовательности коротких шагов.

Теорема S. Вершины любого связного графа могут быть перечислены в циклическом порядке $(v_0, v_1, \dots, v_{n-1})$, так что расстояние между v_k и $v_{(k+1) \bmod n}$ не превышает 3 для $0 \leq k < n$.

Доказательство. Необходимо найти оствовное дерево графа и обойти его в прямо-обратном порядке. ■

Ученые в области теории графов традиционно говорят, что k -й степенью графа G является граф G^k , вершины которого те же, что и у графа G , а ребро $u?v$ существует в графе G^k тогда и только тогда, когда имеется путь длиной не более k между вершинами u и v в графе G . Это позволяет сформулировать теорему S более кратко при $n > 2$: куб связного графа гамильтонов.

Прямо-обратный обход полезен также в случае, когда мы хотим без циклов посетить узлы дерева при помощи ограниченного количества шагов.

Алгоритм Q (Прямо-обратный преемник в трижды связанным лесу). Если Р указывает на узел в лесу, представленном связями PARENT, CHILD и SIB, соответствующими родителю каждого узла, его крайнему слева потомку и правому

брату, то данный алгоритм вычисляет узел Q , следующий за P в прямо-обратном порядке. Предполагается, что известен уровень L , на котором в лесу находится узел P ; значение L обновляется так, что указывает уровень узла Q . Если P — последний узел в прямо-обратном порядке, алгоритм устанавливает $Q \leftarrow \Lambda$ и $L \leftarrow -1$.

- Q1.** [Прямой или обратный?] Если L четно, перейти к шагу Q4.
- Q2.** [Продолжение обратно-прямое.] Установить $Q \leftarrow SIB(P)$. Перейти к шагу Q6, если $Q \neq \Lambda$.
- Q3.** [Перемещение вверх.] Установить $P \leftarrow PARENT(P)$ и $L \leftarrow L - 1$. Перейти к шагу Q7.
- Q4.** [Продолжение прямо-обратное.] Если $CHILD(P) = \Lambda$, перейти к шагу Q7.
- Q5.** [Перемещение вниз.] Установить $Q \leftarrow CHILD(P)$ и $L \leftarrow L + 1$.
- Q6.** [Перемещение вниз при наличии возможности.] Если $CHILD(Q) \neq \Lambda$, установить $Q \leftarrow CHILD(Q)$ и $L \leftarrow L + 1$. Завершить работу алгоритма.
- Q7.** [Перемещение вправо или вверх.] Если $SIB(P) \neq \Lambda$, установить $Q \leftarrow SIB(P)$; в противном случае установить $Q \leftarrow PARENT(P)$ и $L \leftarrow L - 1$. Завершить работу алгоритма. |

Заметим, что, как и в алгоритме 2.4C, связь $PARENT(P)$ проверяется, только если $SIB(P) = \Lambda$. Полный обход представляет собой путешествие червя вокруг леса, наподобие (3): червь “видит” узлы на четных уровнях при проходе слева, а на нечетных — при проходе справа.

Упражнения

1. [15] Каким образом реконструировать строку скобок (1) при проползании червя по бинарному дереву (4)?
2. [20] (Ш. Закс (S. Zaks), 1980.) Модифицируйте алгоритм Р таким образом, чтобы он выдавал сочетания $z_1 z_2 \dots z_n$ из (8) вместо строк скобок $a_1 a_2 \dots a_{2n}$.
- ▶ 3. [23] Докажите, что (11) преобразует $z_1 z_2 \dots z_n$ в таблицу инверсий $c_1 c_2 \dots c_n$.
4. [20] Истинно или ложно утверждение: если строки $a_1 \dots a_{2n}$ сгенерированы в лексикографическом порядке, то в том же порядке находятся строки $d_1 \dots d_n$, $z_1 \dots z_n$, $p_1 \dots p_n$ и $c_1 \dots c_n$.
5. [15] Какие таблицы $d_1 \dots d_n$, $z_1 \dots z_n$, $p_1 \dots p_n$ и $c_1 \dots c_n$ соответствуют строке вложенных скобок (1)?
- ▶ 6. [20] Какое *соответствие* (см. последний столбец табл. 1) сопоставляется строке (1)?
7. [16]
 - а) В каком состоянии находится строка $a_1 a_2 \dots a_{2n}$ при завершении работы алгоритма Р?

b) Что содержится в массивах $l_1 l_2 \dots l_n$ и $r_1 r_2 \dots r_n$ при завершении работы алгоритма В?

8. [15] Как выглядят таблицы $l_1 \dots l_n$, $r_1 \dots r_n$, $e_1 \dots e_n$ и $s_1 \dots s_n$, соответствующие лесу (2)?

9. [M20] Покажите, что таблицы $c_1 \dots c_n$ и $s_1 \dots s_n$ связаны соотношением

$$c_k = [s_1 \geq k - 1] + [s_2 \geq k - 2] + \dots + [s_{k-1} \geq 1].$$

10. [M20] (*Обход червя.*) Для заданной строки $a_1 a_2 \dots a_{2n}$ обозначим через w_j превышение количества левых скобок над правыми в подстроке $a_1 a_2 \dots a_j$, $0 \leq j \leq 2n$. Докажите, что $w_0 + w_1 + \dots + w_{2n} = 2(c_1 + \dots + c_n) + n$.

11. [11] Если F — некоторый лес, *сопряженный* лес F^R получается путем зеркального отражения слева направо. Например, для 14 лесов из табл. 1

••••, ••!•, •!•!, •Λ, •!{, !••, !!•, Λ•, Λ!, {••, {!•, {Λ, {!Λ, {!

сопряженными являются

••••, !•••, •!•!, Λ•, {••, ••!•, !!•, •Λ, Λ!, {••, {!•, {Λ, {!Λ, {!

(солексные леса из табл. 2). Если F соответствует некоторой строке вложенных скобок $a_1 a_2 \dots a_{2n}$, то какой строке соответствует F^R ?

12. [15] Если F — некоторый лес, *транспонированный* лес F^T получается путем обмена в каждом бинарном дереве леса F левых и правых связей. Например, для 14 лесов из табл. 1 транспонированными являются

!, Λ, Λ!, {!, {Λ, {!Λ, {!

Что дает транспонирование леса (2)?

13. [20] Продолжая упражнения 11 и 12, как соотносятся прямой и обратный порядок обхода помеченного леса F и прямой и обратный порядок обхода (a) F^R ; (b) F^T ?

► 14. [21] Найдите все помеченные леса F , такие, что $F^{RT} = F^{TR}$.

15. [20] Предположим, что B — бинарное дерево, полученное из леса F путем связи каждого узла со своим левым братом и крайним справа потомком, как в упражнении 2.3.2–5 и последнем столбце табл. 2. Пусть F' — лес, естественным образом соответствующий B , посредством связей с левым потомком и правым братом. Докажите, что $F' = F^{RT}$ (с использованием обозначений из упражнений 11 и 12).

16. [20] Пусть F и G — леса, а FG означает лес, полученный путем размещения деревьев F слева от деревьев G ; кроме того, определим $F | G = (G^T F^T)^T$. Приведите интуитивное объяснение оператора $|$ и докажите, что он ассоциативен.

17. [M46] Охарактеризуйте все *непомеченные* леса F , такие, что $F^{RT} = F^{TR}$ (см. упражнение 14).

18. [30] Два леса называются *родственными* (cognate), если один из них может быть получен из другого при помощи некоторого количества операций получения сопряженного леса и/или транспонирования. Примеры в упражнениях 11 и 12 показывают, что каждый лес из четырех узлов принадлежит одному из трех классов родственности:

$$\cdots \asymp \{ ; \lambda \asymp \cdots ; \asymp ; \cdots \asymp \{ \cdot \asymp \cdot \{ \asymp \lambda ; \\ ; ; \asymp \lambda \cdot \asymp \cdot \lambda \asymp \lambda \cdot \asymp \lambda \asymp \cdot ; \cdot .$$

Изучите множество всех лесов из 15 узлов. Сколько классов эквивалентности родственных лесов они образуют? Какой класс наибольший? Какой класс наименьший? Чему равен размер класса, содержащего (2)?

19. [28] Пусть F_1, F_2, \dots, F_N — последовательность непомеченных лесов, соответствующих вложенным скобкам, сгенерированным алгоритмом Р, и пусть G_1, G_2, \dots, G_N — последовательность непомеченных лесов, соответствующих бинарным деревьям, генерируемым алгоритмом В. Докажите, что $G_k = F_k^{RTTR}$ (с использованием обозначений из упражнений 11 и 12). (Лес F^{RTTR} называется *дуальным* к лесу F и далее в ряде упражнений обозначается как F^D .)

20. [25] Вспомним из раздела 2.3, что *степенью* узла в дереве называется количество его дочерних узлов и что расширенное (extended) бинарное дерево характеризуется тем свойством, что каждый его узел имеет степень либо 0, либо 2. В расширенном бинарном дереве (4) последовательность степеней узлов в прямом порядке обхода представляет собой 2200222002220220002002202200000; эта строка нулей и двоек идентична последовательности скобок (1), с тем отличием, что каждая скобка ‘(’ заменяется двойкой, каждая скобка ‘)’ — нулем, а также добавляется дополнительный ноль.

a) Докажите, что последовательность неотрицательных целых чисел $b_1 b_2 \dots b_N$ представляет собой последовательность степеней леса в прямом порядке обхода тогда и только тогда, когда она удовлетворяет следующему свойству для $1 \leq k \leq N$:

$$b_1 + b_2 + \dots + b_k + f > k \text{ тогда и только тогда, когда } k < N.$$

Здесь $f = N - b_1 - b_2 - \dots - b_N$ — количество деревьев в лесу.

b) Вспомним из упражнения 2.3.4.5–6: *расширенное тернарное дерево* характеризуется тем свойством, что каждый его узел имеет степень либо 0, либо 3; расширенное тернарное дерево с n внутренними узлами имеет $2n + 1$ внешних узлов, т.е. всего $N = 3n + 1$ узлов. Разработайте алгоритм для генерации всех тернарных деревьев с n внутренними узлами путем генерации соответствующих последовательностей $b_1 b_2 \dots b_N$ в лексикографическом порядке.

- 21. [26] (С. Закс (S. Zaks) и Д. Ричардс (D. Richards), 1979.) Продолжая выполнение упражнения 20, поясните, как сгенерировать последовательности степеней в прямом порядке обхода для всех лесов с $N = n_0 + \dots + n_t$ узлами, среди которых степень j имеют ровно n_j узлов. *Пример:* при $t = 3$, $n_0 = 4$ и $n_1 = n_2 = n_3 = 1$ корректными последовательностями $b_1 b_2 b_3 b_4 b_5 b_6 b_7$ являются

1203000, 1230000, 1300200, 1302000, 1320000, 2013000, 2030010, 2030100, 2031000, 2103000, 2130000, 2300010, 2300100, 2301000, 2310000, 3001200, 3002010, 3002100, 3010200, 3012000, 3020010, 3020100, 3021000, 3100200, 3102000, 3120000, 3200010, 3200100, 3201000, 3210000.

- 22. [30] (Д. Корш (J. Korsh), 2004.) В качестве альтернативы алгоритму В покажите, что бинарные деревья могут быть также эффективно сгенерированы непосредственно в связном виде, если генерировать их в *согласованном* порядке чисел $d_1 \dots d_{n-1}$, определенных в (9). (Реальные значения $d_1 \dots d_{n-1}$ не должны вычисляться явно; должна выполняться работа со связями $l_1 \dots l_n$ и $r_1 \dots r_n$ таким образом, чтобы получались бинарные деревья, соответствующие значениям $d_1 d_2 \dots d_{n-1}$, равным $000\dots 0, 100\dots 0, 010\dots 0, 110\dots 0, 020\dots 0, 001\dots 0, \dots, 000\dots (n-1)$.)

- 23. [25]

- Какова последняя строка, посещаемая алгоритмом N?
- Каково последнее бинарное дерево, посещаемое алгоритмом L?

Указание: см. упражнение 40.

24. [22] Какие последовательности $l_0 l_1 \dots l_{15}, r_1 \dots r_{15}, k_1 \dots k_{15}, q_1 \dots q_{15}$ и $u_1 \dots u_{15}$ соответствуют бинарному дереву (4) и лесу (2) при использовании обозначений из табл. 3?

- 25. [30] (*Обрезка и прививка.*) Представляя бинарные деревья способом, использующимся в алгоритме В, разработайте алгоритм, который посещает все таблицы связей $l_1 \dots l_n$ и $r_1 \dots r_n$ таким образом, что между визитами только одна связь изменяется с j на 0 и одна — с 0 на j для некоторого индекса j . (Другими словами, каждый шаг удаляет некоторое поддерево j из бинарного дерева и помещает его в другое место, сохраняя прямой порядок обхода.)

26. [M21] (*Решетка Кревераса.*) Пусть F и F' — n -узловые леса, узлы которых пронумерованы от 1 до n в прямом порядке обхода. Мы записываем $F < F'$ (“ F срастается с F' ”), если из того, что j и k являются братьями в F' , вытекает, что они братья и в F ; $1 \leq j < k \leq n$. На рис. 39 приведено такое частичное упорядочение для случая $n = 4$; каждый лес кодируется последовательностью $c_1 \dots c_n$ из (9) и (10), которая указывает глубину каждого узла. (При таком кодировании j и k являются братьями тогда и только тогда, когда $c_j = c_k \leq c_{j+1}, \dots, c_{k-1}$.)

- Пусть Π — разбиение $\{1, \dots, n\}$. Покажите, что существует лес F с узлами, помеченными $(1, \dots, n)$ в прямом порядке обхода, такой, что

$$j \equiv k \pmod{\Pi} \Leftrightarrow j \text{ брат } k \text{ в } F$$

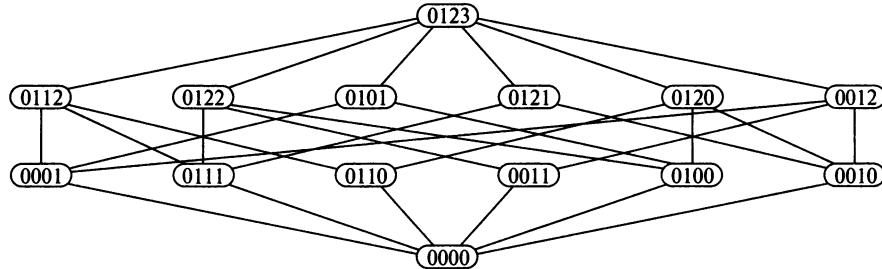


Рис. 39. Решетка Кревераса четвертого порядка. Каждый лес представлен последовательностью глубин узлов $c_1 c_2 c_3 c_4$ в прямом порядке обхода (см. упражнения 26–28)

тогда и только тогда, когда Π обладает свойством *непересекаемости* (noncrossing):

из $i < j < k < l$ и из $i \equiv k$ и $j \equiv l$ (modulo Π) вытекает $i \equiv j \equiv k \equiv l$ (modulo Π).

- b) Поясните, как для двух заданных n -узловых лесов F и F' вычислить их наименьшую верхнюю границу $F \vee F'$, которая представляет собой элемент, такой, что $F \lessdot G$ и $F' \lessdot G$ тогда и только тогда, когда $F \vee F' \lessdot G$.
 - c) Когда F' покрывает F по отношению к \lessdot (см. упражнение 7.2.1.4–55)?
 - d) Покажите, что если F' покрывает F , то он меньше F ровно на один лист.
 - e) Сколько лесов покрывают F так, что узел k имеет e_k дочерних узлов ($1 \leq k \leq n$)?
 - f) Какой лес является дуальным по отношению к лесу (2) при использовании определения дуальности из упражнения 19?
 - g) Докажите, что $F \lessdot F'$ выполняется тогда и только тогда, когда $F'^D \lessdot F^D$. (Из-за этого свойства дуальные элементы располагаются симметрично относительно центра рис. 39.)
 - h) Объясните, как для двух заданных n -узловых лесов F и F' вычислить их наибольшую нижнюю границу $F \wedge F'$, т.е. элемент, такой, что $G \lessdot F$ и $G \lessdot F'$ тогда и только тогда, когда $G \lessdot F \wedge F'$.
 - i) Удовлетворяет ли эта решетка полумодулярному закону, аналогичному закону из упражнения 7.2.1.5–12 (f)?
- 27. [M33] (Решетка Тамари.) Продолжая выполнение упражнения 26, запишем $F \dashv F'$, если j -й узел в прямом порядке обхода для всех j имеет, как минимум, столько же потомков в F' , как и в F . Другими словами, если F и F' характеризуются своими последовательностями $s_1 \dots s_n$ и $s'_1 \dots s'_n$ (см. табл. 2), то $F \dashv F'$ тогда и только тогда, когда $s_j \leq s'_j$ при $1 \leq j \leq n$ (рис. 40).

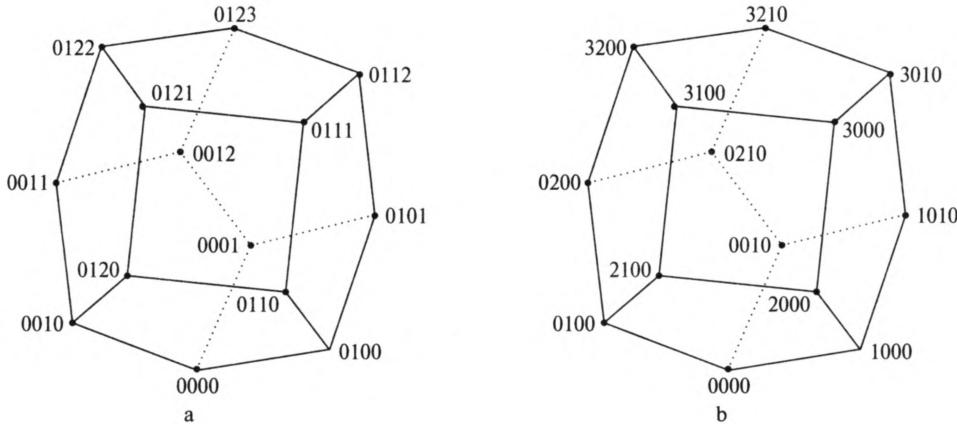


Рис. 40. Решетка Тамари четвертого порядка. Каждый лес представлен: а — последовательностью глубин узлов; б — количеством потомков в прямом порядке обхода (см. упражнения 26–28)

- Покажите, что координаты $\min(s_1, s'_1) \min(s_2, s'_2) \dots \min(s_n, s'_n)$ определяют лес, который является наибольшей нижней границей F и F' (обозначим ее как $F \perp F'$). *Указание:* докажите, что $s_1 \dots s_n$ соответствует лесу тогда и только тогда, когда из $0 \leq k \leq s_j$ вытекает $s_{j+k} + k \leq s_j$, для $0 \leq j \leq n$, если положить $s_0 = n$.
- Когда при таком частичном упорядочении F' покрывает F ?
- Докажите, что $F \dashv F'$ тогда и только тогда, когда $F'^D \dashv F^D$. (Сравните с упражнением 26 (g).)
- Поясните, как вычислить наименьшую верхнюю границу $F \top F'$ для данных F и F' .
- Докажите, что из $F < F'$ в решетке Кревераса вытекает $F \dashv F'$ в решетке Тамари.
- Истинно или ложно: $F \wedge F' \dashv F \perp F'$?
- Истинно или ложно: $F \vee F' < F \top F'$?
- Каковы длиннейший и кратчайший пути от верха решетки Тамари до ее низа, такие, что на пути каждый предшествующий лес покрывает последующий? (Такие пути называются *максимальными цепочками* решетки; сравните с упражнением 7.2.1.4–55 (h).)

28. [M26] (Решетка Стенли.) Продолжая упражнения 26 и 27, определим еще одно частичное упорядочение на n -узловых деревьях, говоря, что $F \subseteq F'$, если координаты глубины $c_1 \dots c_n$ и $c'_1 \dots c'_n$ удовлетворяют соотношениям $c_j \leq c'_j$ при $1 \leq j \leq n$ (рис. 41).

- Докажите, что это частичное упорядочение представляет собой решетку, пояснив, каким образом вычислить наибольшую нижнюю границу $F \cap F'$ и наименьшую верхнюю границу $F \cup F'$ любых двух заданных лесов.

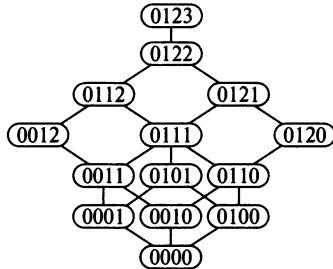


Рис. 41. Решетка Стенли четвертого порядка. Каждый лес представлен последовательностью глубин узлов в прямом порядке обхода (см. упражнения 26–28)

b) Покажите, что решетка Стенли удовлетворяет дистрибутивным законам

$$F \cap (G \cup H) = (F \cap G) \cup (F \cap H), \quad F \cup (G \cap H) = (F \cup G) \cap (F \cup H).$$

c) Когда в данной решетке F' покрывает F ?

d) Истинно или ложно утверждение: $F \subseteq G$ тогда и только тогда, когда $F^R \subseteq G^R$?

e) Докажите, что $F \subseteq F'$ в решетке Стенли, когда $F \dashv F'$ в решетке Тамари.

29. [HM31] Покрывающий граф для решетки Тамари иногда называют “ассоциаэдром” (associahedron) из-за его связи с ассоциативным законом (14), доказанной в упражнении 27 (b). Ассоциаэдр четвертого порядка, показанный на рис. 40, имеет три квадратные грани и шесть пятиугольных граней. (Сравните с рис. 23 из упражнения 7.2.1.2–60, на котором показан “пермутаэдр” четвертого порядка, известное Архимедово тело.) Почему фигура на рис. 40 не входит в классический список однородных многогранников?

30. [M26] Следом (footprint) леса называется битовая строка $f_1 \dots f_n$, определяемая следующим образом:

$$f_j = [\text{узел } j \text{ в прямом порядке обхода не является листом}].$$

a) Если F имеет след $f_1 \dots f_n$, какой след имеет F^D (см. упражнение 27)?

b) Сколько лесов имеют след $1010110111110000101010001011000$?

c) Докажите, что $f_j = [d_j = 0]$, $1 \leq j < n$, с использованием обозначений из (6).

d) Два элемента решетки называются *комplementарными* (complementary), если их наибольшая нижняя граница представляет собой нижний элемент, а наименьшая верхняя граница — верхний элемент. Покажите, что F и F' комплементарны в решетке Тамари тогда и только тогда, когда их следы комплементарны в том смысле, что $f'_1 \dots f'_{n-1} = \bar{f}_1 \dots \bar{f}_{n-1}$.

► 31. [M28] Бинарное дерево с n внутренними узлами называется *вырожденным* (degenerate), если его высота равна $n - 1$.

- a) Сколько n -узловых бинарных деревьев вырождено?
- b) В табл. 1, 2 и 3 мы видели, что бинарные деревья и леса могут быть закодированы при помощи различных n -кортежей чисел. Поясните для каждой из кодировок $c_1 \dots c_n, d_1 \dots d_n, e_1 \dots e_n, k_1 \dots k_n, p_1 \dots p_n, s_1 \dots s_n, u_1 \dots u_n$ и $z_1 \dots z_n$, как можно при беглом взгляде на кодировку сказать, вырождено ли соответствующее бинарное дерево или нет.
- c) Истинно или ложно утверждение: если лес F вырожден, то вырожден и F^D ?
- d) Докажите, что если и F и F' вырождены, то вырождены $F \wedge F' = F \perp F'$ и $F \vee F' = F \top F'$.

► 32. [M30] Докажите, что если $F \dashv F'$, то существует лес F'' , такой, что для всех G

$$F' \perp G = F \text{ тогда и только тогда, когда } F \dashv G \dashv F''.$$

Следовательно, в решетке Тамари выполняются *полудистрибутивные законы*:

$$\begin{aligned} \text{из } F \perp G = F \perp H \text{ вытекает } F \perp (G \top H) = F \perp G; \\ \text{из } F \top G = F \top H \text{ вытекает } F \top (G \perp H) = F \top G. \end{aligned}$$

► 33. [M27] (*Представление деревьев при помощи перестановок.*) Пусть σ — цикл $(1 \ 2 \ \dots \ n)$.

- a) Докажите, что для заданного бинарного дерева, узлы которого пронумерованы от 1 до n в симметричном порядке обхода, существует единственная перестановка λ множества $\{1, \dots, n\}$, такая, что для $1 \leq k \leq n$

$$\begin{aligned} \text{LLINK}[k] &= \begin{cases} k\lambda, & \text{если } k\lambda < k; \\ 0 & \text{в противном случае;} \end{cases} \\ \text{RLINK}[k] &= \begin{cases} k\sigma\lambda, & \text{если } k\sigma\lambda > k; \\ 0 & \text{в противном случае.} \end{cases} \end{aligned}$$

Таким образом, λ упаковывает $2n$ полей связей в единый n -элементный массив.

- b) Покажите, что эту перестановку λ наиболее легко описать в циклическом виде, когда бинарное дерево является представлением левый брат/правый потомок леса F . Как выглядит циклический вид $\lambda(F)$ для леса (2) ?
- c) Найдите простое соотношение между $\lambda(F)$ и дуальной перестановкой $\lambda(F^D)$.
- d) Докажите, что в упражнении 26 F' покрывает F тогда и только тогда, когда $\lambda(F') = (j \ k) \lambda(F)$, где j и k — братья в F .
- e) Следовательно, количество максимальных цепочек в решетке Кревераса порядка n равно количеству способов разложения n -цикла в виде произведения $n - 1$ перестановок. Вычислите это значение. *Указание:* см. формулу 1.2.6–(16).

34. [M25] (Р.П. Стенли (R.P. Stanley).) Покажите, что количество максимальных цепочек в решетке Стенли n -го порядка равно

$$(n(n-1)/2)! / \left(1^{n-1} 3^{n-2} \dots (2n-5)^2 (2n-3)^1 \right).$$

35. [HM37] (Д.Б. Тайлер (D.B. Tyler) и Д.Р. Хикерсон (D.R. Hickerson).) Объясните, почему все знаменатели в асимптотической формуле (16) являются степенями 2.

► **36.** [M25] Проанализируйте алгоритм генерации тернарных деревьев из упражнения 20 (б). Указание: в соответствии с упражнением 2.3.4.4–11 имеется $(2n+1)^{-1} \binom{3n}{n}$ тернарных деревьев с n внутренними узлами.

► **37.** [M40] Проанализируйте алгоритм Закса–Ричардса для генерации всех деревьев с данным распределением $n_0, n_1, n_2, \dots, n_t$ степеней (см. упражнение 21). Указание: см. упражнение 2.3.4.4–32.

38. [M22] Чему равно общее количество обращений к памяти, выполняемое алгоритмом L, выраженное как функция от n ?

39. [22] Докажите формулу (23), показав, что элементы A_{pq} в (5) соответствуют диаграмме Юнга с двумя строками.

40. [M22] (а) Докажите, что C_{pq} нечетно тогда и только тогда, когда $p \& (q+1) = 0$, т.е. когда бинарное представление p и $q+1$ не имеют общих битов. (б) Следовательно, C_n нечетно тогда и только тогда, когда $n+1$ представляет собой степень 2.

41. [M21] Покажите, что баллотировочные числа имеют простую производящую функцию $\sum C_{pq} w^p z^q$.

► **42.** [M22] Сколько непомеченных лесов с n узлами являются (а) самосопряженными; (б) самотранспонированными; (с) самодуальными? (См. упражнения 11, 12, 19 и 26.)

43. [M21] Выразите C_{pq} через числа Каталана $\langle C_0, C_1, C_2, \dots \rangle$. Целью является получение простой формулы для малых величин $q-p$. (Например, $C_{(q-2)q} = C_q - C_{q-1}$.)

► **44.** [M27] Докажите, что алгоритм В делает $8\frac{2}{3} + O(n^{-1})$ обращений к памяти на одно посещенное бинарное дерево.

45. [M26] Проанализируйте обращения к памяти, выполняемые алгоритмом из упражнения 22. Сравните полученное значение со значением для алгоритма В.

46. [M30] (Обобщенные числа Каталана.) Обобщим соотношения (21), определив

$$C_{pq}(x) = C_{p(q-1)}(x) + x^{q-p} C_{(p-1)q}(x), \text{ если } 0 \leq p \leq q \neq 0; \quad C_{00}(x) = 1;$$

и

$$C_{pq}(x) = 0, \text{ если } p < 0 \text{ или } p > q;$$

таким образом, $C_{pq} = C_{pq}(1)$. Пусть также $C_n(x) = C_{nn}(x)$, так что

$$\begin{aligned} \langle C_0(x), C_1(x), \dots \rangle &= \\ &= \langle 1, 1, 1+x, 1+2x+x^2+x^3, 1+3x+3x^2+3x^3+2x^4+x^5+x^6, \dots \rangle. \end{aligned}$$

- a) Покажите, что $[x^k] C_{pq}(x)$ — количество путей от (pq) до (00) в графе (28), которые имеют площадь k , где “площадь” пути представляет собой количество прямоугольных ячеек над ним. (Таким образом, L-образный путь имеет максимально возможную площадь, равную $p(q-p) + \binom{p}{2}$.)
- b) Докажите, что $C_n(x) = \sum_F x^{c_1+\dots+c_n} = \sum_F x^{\text{длина внутреннего пути}(F)}$, где сумма берется по всем n -узловым лесам F .
- c) Покажите, что если $C(x, z) = \sum_{n=0}^{\infty} C_n(x) z^n$, то $C(x, z) = 1 + zC(x, z) C(x, xz)$.
- d) Кроме того, $C(x, z) C(x, xz) \dots C(x, x^r z) = \sum_{p=0}^{\infty} C_{p(p+r)}(x) z^p$.

47. [M27] Продолжая предыдущее упражнение, обобщите тождество (27).

48. [M28] (Ф. Раски (F. Ruskey) и А. Проскуровски (A. Proskurowski).) Вычислите $C_{pq}(x)$ при $x = -1$ и используйте полученный результат для того, чтобы показать, что “идеальный” код Грея для вложенных скобок невозможен при нечетном $n \geq 5$.

49. [17] Какова миллионная в лексикографическом порядке строка, состоящая из 15 вложенных пар скобок?

50. [20] Разработайте алгоритм, обратный алгоритму U: для данной строки вложенных скобок $a_1 \dots a_{2n}$ он должен определять ее ранг $N - 1$ в лексикографическом порядке. Чему равен ранг (1)?

51. [M22] Пусть $\bar{z}_1 \bar{z}_2 \dots \bar{z}_n$ — дополнение $z_1 z_2 \dots z_n$ до $2n$; другими словами, $\bar{z}_j = 2n - z_j$, где z_j определено в формуле (8). Покажите, что если $\bar{z}_1 \bar{z}_2 \dots \bar{z}_n$ — $(N+1)$ -е n -сочетание $\{0, 1, \dots, 2n-1\}$, сгенерированное алгоритмом 7.2.1.3L, то $z_1 z_2 \dots z_n$ является $(N - \kappa_n N + 1)$ -м n -сочетанием $\{1, 2, \dots, 2n\}$, сгенерированным алгоритмом из упражнения 2. (Здесь κ_n означает n -ю функцию Крускала, определенную в 7.2.1.3-(60).)

52. [M23] Найдите среднее и дисперсию величины d_n в табл. 1 при случайном выборе строки вложенных скобок $a_1 \dots a_{2n}$.

53. [M28] Пусть X — расстояние от корня расширенного бинарного дерева до крайнего слева внешнего узла.

- a) Каково математическое ожидание значения X , если все бинарные деревья с n узлами равновероятны?
- b) Каково математическое ожидание значения X в случайном *бинарном дереве поиска*, построенном с использованием алгоритма 6.2.2T на основе случайной перестановки $K_1 \dots K_n$?
- c) Каково математическое ожидание значения X в случайном *вырожденном* (в смысле упражнения 31) бинарном дереве?
- d) Каково математическое ожидание значения 2^X во всех трех случаях?

54. [HM29] Чему равно математическое ожидание и дисперсия $c_1 + \dots + c_n$ (см. упражнение 46)?

55. [HM33] Вычислите $C'_{pq}(1)$ — общую площадь всех путей в упражнении 46 (а).

56. [M23] (Ренцо Спруньюли (Renzo Sprugnoli), 1990.) Докажите формулу суммирования

$$\sum_{k=0}^{m-1} C_k C_{n-1-k} = \frac{1}{2} C_n + \frac{2m-n}{2n(n+1)} \binom{2m}{m} \binom{2n-2m}{n-m} \text{ для } m \leq n.$$

57. [M28] Выразите суммы $S_p(a, b) = \sum_{k \geq 0} \binom{2a}{a-k} \binom{2b}{b-k} k^p$ в аналитическом виде для $p = 0, 1, 2, 3$ и используйте полученные формулы для доказательства формулы (30).

58. [HM34] Обозначим через t_{lmn} количество n -узловых бинарных деревьев, в которых внешний узел m (при нумерации внешних узлов от 0 до n в симметричном порядке обхода) находится на уровне l . Пусть также $t_{mn} = \sum_{l=1}^n l t_{lmn}$, так что t_{mn}/C_n — средний уровень внешнего узла m ; и пусть $t(w, z)$ — сверхпроизводящая функция

$$\sum_{m,n} t_{mn} w^m z^n = (1+w)z + (3+4w+3w^2)z^2 + (9+13w+13w^2+9w^3)z^3 + \dots$$

Докажите, что

$$t(w, z) = (C(z) - wC(wz))/(1-w) - 1 + zC(z)t(w, z) + wzC(wz)t(w, z),$$

и выведите простую формулу для чисел t_{mn} .

59. [HM29] Аналогично: пусть T_{lmn} — количество всех n -узловых бинарных деревьев, в которых *внутренний* узел m находится на уровне l . Найдите простую формулу для $T_{mn} = \sum_{l=1}^n l T_{lmn}$.

► **60.** [M26] (*Сбалансированные строки*.) Стока вложенных скобок α является *атомарной* (atomic), если она имеет вид (α') , где α' — строка вложенных скобок. Каждая строка вложенных скобок может быть единственным образом представлена как произведение атомов $\alpha_1 \dots \alpha_r$. Стока с одинаковым количеством левых и правых скобок называется *сбалансированной* (balanced). Каждая сбалансированная строка может быть единственным образом представлена в виде $\beta_1 \dots \beta_r$, где каждая подстрока β_j является либо атомом, либо *соатомом* (co-atom) (обращением атома). *Дефектом* (defect) сбалансированной строки является половина длины ее соатомов. Например, сбалансированная строка

$$(()) ((()))) (() ((()) ((()) (()$$

имеет разложение $\beta_1 \beta_2 \beta_3 \beta_4 \beta_5 \beta_6 \beta_7 \beta_8 = \alpha_1 \alpha_2^R \alpha_3 \alpha_4^R \alpha_5^R \alpha_6 \alpha_7^R \alpha_8$, с четырьмя атомами и четырьмя соатомами; ее дефект равен $|\alpha_2 \alpha_4 \alpha_5 \alpha_7|/2 = 9$.

a) Докажите, что дефект сбалансированной строки равен количеству индексов k , для которых k -я правая скобка *предшествует* k -й левой скобке.

- b) Если строка $\beta_1 \dots \beta_r$ сбалансирована, ее можно отобразить на строку вложенных скобок простым обращением соатомов. Однако описанное далее отображение более интересно, поскольку производит несмещенные (случайные равномерно распределенные) строки вложенных скобок из несмешанных сбалансированных строк. Пусть имеется s соатомов $\beta_{i_1} = \alpha_{i_1}^R, \dots, \beta_{i_s} = \alpha_{i_s}^R$. Заменим каждый соатом на '('; затем добавим строку $)\alpha'_{i_s} \dots)\alpha'_{i_1}$, где $\alpha_j = (\alpha'_j)$. Например, строка, приведенная выше, отображается на строку $\alpha_1(\alpha_3((\alpha_6(\alpha_8)\alpha'_7)\alpha'_5)\alpha'_4)\alpha'_2$, которая представляет собой строку (1) из начала данного раздела.
- c) Разработайте алгоритм, который применяет описанное отображение к заданной сбалансированной строке $b_1 \dots b_{2n}$.
- d) Разработайте также алгоритм для обратного отображения: для данной строки вложенных скобок $\alpha = a_1 \dots a_{2n}$ и целого числа $0 \leq l \leq n$ он должен находить сбалансированную строку $\beta = b_1 \dots b_{2n}$, дефект которой равен l и которая отображается на строку α . Какая сбалансированная строка с дефектом 11 отображается на строку (1)?

► 61. [M26] (Лемма Рани (Raney) о цикле.) Пусть $b_1b_2 \dots b_N$ — строка неотрицательных целых чисел, такая, что $f = N - b_1 - b_2 - \dots - b_N > 0$.

- a) Докажите, что свойству последовательности степеней леса в прямом порядке обхода из упражнения 20 удовлетворяют ровно f циклических сдвигов $b_{j+1} \dots b_N b_1 \dots b_j$, $1 \leq j \leq N$.
- b) Разработайте эффективный алгоритм для определения всех таких j для данной строки $b_1b_2 \dots b_N$.
- c) Поясните, как сгенерировать случайный лес с $N = n_0 + \dots + n_t$ узлами, в котором степень j имеют ровно n_j узлов. (Например, в качестве частного случая этой процедуры при $N = tn + 1$, $n_0 = (t - 1)n + 1$, $n_1 = \dots = n_{t-1} = 0$ и $n_t = n$ мы получаем случайное n -узловое t -арное дерево.)

62. [22] Бинарное дерево может быть представлено также битовыми строками $(l_1 \dots l_n, r_1 \dots r_n)$, где l_j и r_j указывают, пусты или нет левое и правое поддеревья узла j в прямом порядке обхода (см. теорему 2.3.1А). Докажите, что если $l_1 \dots l_n$ и $r_1 \dots r_n$ — произвольные битовые строки, такие, что $l_1 + \dots + l_n + r_1 + \dots + r_n = n - 1$, то только один циклический сдвиг $(l_{j+1} \dots l_n l_1 \dots l_j, r_{j+1} \dots r_n r_1 \dots r_j)$ дает корректное представление бинарного дерева, и поясните, как его найти.



63. [16] Если первые две итерации алгоритма Реми дают

то какие декорированные бинарные деревья возможны после очередной итерации?

64. [20] Какая последовательность значений X в алгоритме R соответствует декорированным деревьям из (34) и каковы конечные значения $L_0 L_1 \dots L_{12}$?

65. [38] Обобщите алгоритм Реми (алгоритм R) для t -арных деревьев.

66. [21] Деревом Шрёдера (Schröder) называется бинарное дерево, в котором каждая ненулевая правая связь окрашена либо в белый, либо в черный цвет. Вот количества S_n деревьев Шрёдера для небольших n :

$n = 0$	1	2	3	4	5	6	7	8	9	10	11	12
$S_n = 1$	1	3	11	45	197	903	4279	20793	103049	518859	2646723	13648869

Например, $S_3 = 11$, как видно из приведенных вариантов:



(Белые связи “пустотельные”; на рисунке показаны также внешние узлы.)

- a) Найдите простое соответствие между деревьями Шрёдера с n внутренними узлами и обычными деревьями с $n + 1$ листьями и без узлов со степенью, равной 1.
- b) Разработайте код Грея для деревьев Шрёдера.

67. [M22] Какова производящая функция $S(z) = \sum_n S_n z^n$ для чисел Шрёдера?

68. [10] Какой вид имеет шаблон рождественской елки порядка 0?

69. [20] Содержатся ли в табл. 4 шаблоны рождественской елки порядка 6 и 7, возможно, в замаскированном виде?

► **70.** [20] Найдите простое правило, которое определяет для каждой битовой строки σ другую битовую строку σ' , которая называется *напарником* (mate) первой строки и обладает следующими свойствами: (i) $\sigma'' = \sigma$; (ii) $|\sigma'| = |\sigma|$; (iii) либо $\sigma \subseteq \sigma'$, либо $\sigma' \subseteq \sigma$; (iv) $\nu(\sigma) + \nu(\sigma') = |\sigma|$.

71. [M21] Пусть M_{tn} — размер наибольшего возможного множества S n -битовых строк, обладающих тем свойством, что если σ и τ — члены S , такие, что $\sigma \subseteq \tau$, то $\nu(\sigma) < \nu(\tau) + t$. (Таким образом, например, $M_{1n} = M_n$ согласно теореме Спернера.) Найдите формулу для M_{tn} .

► **72.** [M28] Если начать с единственной строки $\sigma_1 \sigma_2 \dots \sigma_s$ длиной s и применить к ней правило роста (36) n раз подряд, то сколько строк мы получим?

73. [15] Каковы первый и последний элементы строки шаблона рождественского дерева порядка 30, содержащей битовую строку 01100100100001111101101011100?

74. [M26] Продолжая предыдущее упражнение, ответьте, сколько строк предшествует указанной в нем?

► **75.** [HM23] Пусть $(r_1^{(n)}, r_2^{(n)}, \dots, r_{n-1}^{(n)})$ — номера строк, в которых шаблон рождественской елки порядка n содержит $n - 1$ элементов; например, из табл. 4 видно, что $(r_1^{(8)}, \dots, r_7^{(8)}) = (20, 40, 54, 62, 66, 68, 69)$. Найдите формулы для $r_{j+1}^{(n)} - r_j^{(n)}$ и для $\lim_{n \rightarrow \infty} r_j^{(n)} / M_n$.

76. [HM46] Рассмотрите предельный вид шаблона рождественской елки при $n \rightarrow \infty$. Имеет ли он, например, фрактальную размерность при выборе некоторого подходящего масштабирования?

77. [21] Разработайте алгоритм для генерации последовательности крайних справа элементов $a_1 \dots a_n$ строк шаблона рождественской елки для данного n . Указание: эти битовые строки характеризуются тем свойством, что $a_1 + \dots + a_k \geq k/2$ при $0 \leq k \leq n$.

78. [20] Истинно или ложно утверждение: если $\sigma_1 \dots \sigma_s$ — строка шаблона рождественской елки, то ею же является и строка $\bar{\sigma}_s^R \dots \bar{\sigma}_1^R$ (обращенная последовательность обращенных дополнений)?

79. [M26] Количество перестановок $p_1 \dots p_n$, у которых имеется ровно один “спуск” $p_k > p_{k+1}$, равно, согласно 5.1.3–(12), числу Эйлера $\langle \begin{smallmatrix} n \\ 1 \end{smallmatrix} \rangle = 2^n - n - 1$. Количество элементов в шаблоне рождественской елки выше нижней строки такое же.

- a) Найдите комбинаторное пояснение этого совпадения, приведя взаимнооднозначное соответствие между перестановками с одним спуском и неотсортированными битовыми строками.
- b) Покажите, что две неотсортированные битовые строки принадлежат одной и той же строке шаблона рождественской елки тогда и только тогда, когда они соответствуют перестановкам, которые определяют одну и ту же диаграмму P при соответствии Робинсона–Шенстеда (теорема 5.1.4A).

80. [30] Будем говорить, что две битовые строки *совместимы* (concordant), если одну из них можно получить из другой путем преобразования подстрок $010 \leftrightarrow 100$ или $101 \leftrightarrow 110$. Например, строки

$$\begin{array}{ccccccccc} 011100 & \leftrightarrow & 011010 & \leftrightarrow & 010110 & \leftrightarrow & 010101 & \leftrightarrow & 011001 \\ & & \downarrow & & \downarrow & & & & \\ & & 100110 & \leftrightarrow & 100101 & \leftrightarrow & 101001 & \leftrightarrow & 110001 \end{array}$$

взаимно совместимы, но ни одна другая строка не совместима ни с одной из приведенных.

Докажите, что строки взаимно совместимы тогда и только тогда, когда они принадлежат одному и тому же столбцу шаблона рождественской елки и строкам одинаковой длины в этом шаблоне.

81. [M30] *Биклэттером* (biclutter) порядка (n, n') называется семейство S пар битовых строк (σ, σ') , где $|\sigma| = n$ и $|\sigma'| = n'$, обладающих тем свойством, что для различных членов (σ, σ') и (τ, τ') семейства S условия $\sigma \subseteq \tau$ и $\sigma' \subseteq \tau'$ выполняются, только если $\sigma \neq \tau$ и $\sigma' \neq \tau'$.

Воспользуйтесь шаблонами рождественской елки, чтобы доказать, что S содержит не более $M_{n+n'}$ пар строк.

► **82.** [M26] Пусть $E(f)$ — количество вычислений функции f алгоритмом Н.

- a) Покажите, что $M_n \leq E(f) \leq M_{n+1}$, причем равенство достигается, когда f — константа.

b) Какая из всех функций f , таких, что $E(f) = M_n$, минимизирует $\sum_{\sigma} f(\sigma)$?

c) Какая из всех функций f , таких, что $E(f) = M_{n+1}$, минимизирует $\sum_{\sigma} f(\sigma)$?

83. [M20] (Д. Хансель (G. Hansel).) Покажите, что существует не более 3^{M_n} монотонных булевых функций $f(x_1, \dots, x_n)$ от n булевых переменных.

► **84.** [HM27] (Д. Кляйтман (D. Kleitman).) Пусть A — матрица действительных чисел размером $m \times n$, в которой каждый столбец v имеет длину $\|v\| \geq 1$, и пусть b — m -мерный вектор-столбец. Докажите, что не более M_n векторов-столбцов $x = (a_1, \dots, a_n)^T$ с компонентами $a_j = 0$ или 1 , удовлетворяют условию $\|Ax - b\| \leq \leq 1/2$. Указание: воспользуйтесь конструкцией, аналогичной шаблону рождественской елки.

85. [HM35] (Филипп Голль (Philippe Golle).) Пусть V — произвольное векторное пространство, содержащееся во множестве всех действительных n -мерных векторов, но не содержащее ни одного из единичных векторов $(1, 0, \dots, 0)$, $(0, 1, 0, \dots, 0)$, \dots , $(0, \dots, 0, 1)$. Докажите, что V содержит не более M_n векторов, все компоненты которых равны 0 или 1 ; более того, верхняя граница M_n достижима.

86. [15] Если (2) рассматривать как *ориентированный лес*, а не как упорядоченный, то какой канонический лес ему соответствует? Укажите этот лес как с использованием кодов уровней $c_1 \dots c_{15}$, так и при помощи указателей на родительские узлы $p_1 \dots p_{15}$.

87. [M20] Пусть F — упорядоченный лес, в котором k -й в прямом порядке обхода узел находится на уровне c_k , а его родительский узел — p_k , причем $p_k = 0$, если узел является корнем.

a) Сколько лесов удовлетворяет условию $c_k = p_k$ для $1 \leq k \leq n$?

b) Предположим, что F и F' имеют коды уровней $c_1 \dots c_n$ и $c'_1 \dots c'_n$ и родительские связи $p_1 \dots p_n$ и $p'_1 \dots p'_n$ соответственно. Докажите, что лексикографически $c_1 \dots c_n \leq c'_1 \dots c'_n$ тогда и только тогда, когда $p_1 \dots p_n \leq p'_1 \dots p'_n$.

88. [M20] Проанализируйте алгоритм О: как часто выполняется шаг О4? Чему равно общее количество изменений p_k на шаге О5?

89. [M46] Как часто на шаге О5 выполняется присваивание $p_k \leftarrow p_j$?

► **90.** [M27] Если $p_1 \dots p_n$ — каноническая последовательность указателей на родительские узлы для ориентированного леса, то граф с вершинами $\{0, 1, \dots, n\}$ и ребрами $\{k-p_k \mid 1 \leq k \leq n\}$ является *свободным деревом*, т.е. связным графом без циклов (см. теорему 2.3.4.1А). И наоборот: каждое свободное дерево соответствует таким способом, как минимум, одному ориентированному лесу. Однако указатели на родительские узлы 011 и 000 соответствуют одному и тому же свободному дереву ; аналогично: последовательности 012 и 010 также дают одно дерево .

Цель этого упражнения — наложить на последовательности $p_1 \dots p_n$ ограничения с тем, чтобы каждое свободное дерево получалось только один раз. В 2.3.4.4–(9) мы доказали, что количество структурно различных свободных деревьев с $n + 1$ узлами имеет очень простую производящую функцию, показав, что свободное дерево всегда имеет, как минимум, один *центроид* (centroid).

- a) Покажите, что канонический n -узловой лес соответствует свободному дереву с одним центроидом тогда и только тогда, когда в лесу ни одно дерево не имеет больше $\lfloor n/2 \rfloor$ узлов.
- b) Модифицируйте алгоритм О таким образом, чтобы он генерировал все последовательности $p_1 \dots p_n$, удовлетворяющие пункту (a).
- c) Поясните, как найти все $p_1 \dots p_n$ для свободных деревьев, имеющих два центрида.

91. [M37] (А. Ньенхuis (A. Nijenhuis) и Г. Вильф (H. Wilf).) Покажите, что случайное ориентированное дерево может быть сгенерировано при помощи процедуры, аналогичной алгоритму случайного разбиения из упражнения 7.2.1.4–47.

92. [15] Являются ли первое и последнее оствовные деревья, посещенные алгоритмом S, смежными, в том смысле что они имеют $n - 2$ общих ребер?

93. [20] Восстановливает ли алгоритм S граф в первоначальное состояние по завершении работы?

94. [22] Алгоритм S требует “прокрутить стартер”, т.е. найти начальное оствовное дерево на шаге S1. Объясните, как это сделать.

95. [26] Завершите алгоритм S, реализовав проверку моста на шаге S8.

► **96.** [28] Проанализируйте приближенное время работы алгоритма S, когда данный граф представляет собой (a) путь P_n длиной $n - 1$; (b) цикл C_n длиной n .

97. [15] Является ли граф (48) последовательно-параллельным?

98. [16] Какой последовательно-параллельный граф соответствует графу (53), если узел A — последовательный?

► **99.** [30] Рассмотрим последовательно-параллельный граф, представленный деревом, как в (53), со значениями узлов, удовлетворяющими условию (55). Эти значения определяют оствовное дерево или близкое дерево, в соответствии с тем, равно ли значение v_p корня p единице или нулю. Покажите, что описанный далее метод генерирует все прочие конфигурации корня.

- i) В начале все непростые узлы являются активными, а все остальные — пассивными.
- ii) Выбираем крайний справа активный узел p в прямом порядке обхода; если все узлы пассивны, завершаем работу.
- iii) Изменяем $d_p \leftarrow r_{d_p}$, обновляем все значения в дереве и посещаем новую конфигурацию.
- iv) Активизируем все непростые узлы справа от p .
- v) Если d_p прошло по всем дочерним узлам p , поскольку p — последний ставший активным узел, делаем его пассивным. Возвращаемся к шагу (ii).

Поясните также, как эффективно выполнить описанные шаги. *Указание:* для реализации шага (v) введите указатель z_p ; делайте узел p пассивным, когда d_p становится равным z_p , и в этот момент сбрасывайте значение z_p , делая его равным предыдущему значению d_p . Для реализации шагов (ii) и (iv) воспользуйтесь указателем f_p , аналогичным используемому в алгоритмах 7.2.1.1L и 7.2.1.1K.

100. [40] Реализуйте “алгоритм S/ двери-вертушки” для генерации всех остовных деревьев путем комбинации алгоритма S и идей из упражнения 99.

101. [46] Имеется ли простой способ перечисления в порядке “двери-вертушки” всех n^{n-2} остовных деревьев полного графа K_n ? (Порядок, получаемый с использованием алгоритма S, слишком сложен.)

102. [46] *Ориентированным остовным деревом* (oriented spanning tree) ориентированного графа D , состоящего из n вершин, известным также под названием “остовной древовидности” (spanning arborescence), является ориентированное поддерево D , содержащее $n - 1$ дуг. Теорема о матрице, соответствующей дереву (упражнение 2.3.4.2–19) гласит, что ориентированные поддеревья, имеющие данный корень, могут быть легко подсчитаны путем вычисления детерминанта размером $(n - 1) \times (n - 1)$.

Можно ли перечислить эти поддеревья в порядке “двери-вертушки”, при котором на каждом шаге происходит удаление одной дуги и замещение ее другой?

► **103.** [HM39] (*Барханы.*) Рассмотрим произвольный ориентированный граф D , состоящий из вершин V_0, V_1, \dots, V_n с дугами e_{ij} от V_i к V_j , где $e_{ii} = 0$. Будем считать, что D имеет, как минимум, одно ориентированное остовное дерево с корнем V_0 ; это предположение означает, что если мы соответствующим образом перенумеруем вершины, то $e_{i0} + \dots + e_{i(i-1)} > 0$ для $1 \leq i \leq n$. Пусть $d_i = e_{i0} + \dots + e_{in}$ — общая исходящая степень вершины V_i . Поместим x_i песчинок на вершину V_i для $0 \leq i \leq n$ и сыграем в следующую игру. Если $x_i \geq d_i$ для любого $i \geq 1$, уменьшим x_i на d_i и установим $x_j \leftarrow x_j + e_{ij}$ для всех $j \neq i$ (другими словами, если это возможно, перешлем по одной песчинке из V_i по всем выходящим из нее дугам, за исключением вершины $i = 0$). Назовем эту операцию “рассыпанием” V_i , а последовательность рассыпаний — “лавиной”. Вершина V_0 особая; вместо рассыпания она собирает песчинки, которые, по сути, покидают систему). Продолжаем до тех пор, пока не будет выполнено условие $x_i < d_i$ для $1 \leq i \leq n$. Такое состояние $x = (x_1, \dots, x_n)$ назовем *устойчивым*.

- Докажите, что каждая лавина завершается устойчивым состоянием после конечного числа рассыпаний. Более того, конечное состояние зависит только от начального состояния, но не от порядка выполнения рассыпаний.
- Пусть $\sigma(x)$ — устойчивое состояние, являющееся результатом начального состояния x . Назовем устойчивое состояние *рекуррентным*, если это $\sigma(x)$ для некоторого x с $x_i \geq d_i$ для $1 \leq i \leq n$. (Рекуррентные состояния соответствуют “барханам”, которые образовались за длинный промежуток времени, после многократного случайного введения новых песчинок.) Найдите рекуррентные состояния для частного случая $n = 4$ и дуг D

$$V_1 \rightarrow V_0, V_1 \rightarrow V_2, V_2 \rightarrow V_0, V_2 \rightarrow V_1, V_3 \rightarrow V_0, V_3 \rightarrow V_4, V_4 \rightarrow V_0, V_4 \rightarrow V_3.$$

- c) Пусть $d = (d_1, \dots, d_n)$. Докажите, что x рекуррентно тогда и только тогда, когда $x = \sigma(x + t)$, где t — вектор $d - \sigma(d)$.
- d) Пусть a_i — вектор $(-e_{i1}, \dots, -e_{i(i-1)}, d_i, -e_{i(i+1)}, \dots, -e_{in})$ для $1 \leq i \leq n$; таким образом, рассыпание V_i соответствует изменению вектора состояния $x = (x_1, \dots, x_n)$, который становится равен $x - a_i$. Будем говорить, что два состояния x и x' *конгруэнтны* (записываем как $x \equiv x'$), если $x - x' = m_1 a_1 + \dots + m_n a_n$ для некоторых целых чисел m_1, \dots, m_n . Докажите, что существует ровно столько же классов эквивалентности конгруэнтных состояний, сколько и ориентированных остовных деревьев в D с корнем V_0 . *Указание:* см. теорему о матрице, соответствующей дереву (упражнение 2.3.4.2-19).
- e) Докажите, что если $x \equiv x'$, а также и x , и x' рекуррентны, то $x = x'$.
- f) Докажите, что каждый класс конгруэнтности содержит единственное рекуррентное состояние.
- g) Пусть D — *сбалансированный* граф в том смысле, что входящая степень каждой вершины равна ее исходящей степени. Докажите, что в таком случае x является рекуррентным состоянием тогда и только тогда, когда $x = \sigma(x + a)$, где $a = (e_{01}, \dots, e_{0n})$.
- h) Проиллюстрируйте эту концепцию для случая, когда D представляет собой “колесо” с n спицами, когда всего имеется $3n$ дуг, $V_j \rightarrow V_0$ и $V_j \leftrightarrow V_{j+1}$ для $1 \leq j \leq n$, где вершина V_{n+1} рассматривается как идентичная V_1 . Найдите взаимно однозначное соответствие между ориентированными остовными деревьями этого ориентированного графа и рекуррентными состояниями его барханов.
- i) Аналогично проанализируйте рекуррентные барханы, когда D представляет собой *полный* граф из $n + 1$ вершин, а именно когда $e_{ij} = [i \neq j]$ для $0 \leq i, j \leq n$. *Указание:* см. упражнение 6.4-31.

► 104. [HM21] Пусть G — граф с n вершинами $\{V_1, \dots, V_n\}$, с ребрами e_{ij} между вершинами V_i и V_j . Обозначим через $C(G)$ матрицу с элементами $c_{ij} = -e_{ij} + \delta_{ij} d_i$, где $d_i = e_{i1} + \dots + e_{in}$ — степень вершины V_i . Будем говорить, что *сторонами* (aspects) G являются собственные значения $C(G)$, т.е. корни $\alpha_0, \dots, \alpha_{n-1}$ уравнения $\det(\alpha I - C(G)) = 0$. Поскольку $C(G)$ — симметричная матрица, ее собственные значения являются действительными числами, и мы можем считать, что $\alpha_0 \leq \alpha_1 \leq \dots \leq \alpha_{n-1}$.

- a) Докажите, что $\alpha_0 = 0$.
- b) Докажите, что G имеет ровно $c(G) = \alpha_1 \dots \alpha_{n-1}/n$ остовных деревьев.
- c) Чему равны стороны полного графа K_n ?

105. [HM37] Продолжая выполнение упражнения 104, мы хотим доказать, что часто есть простой путь определения сторон G , когда граф G построен из других графов, стороны которых известны. Предположим, что G' имеет стороны $\alpha'_0, \dots, \alpha'_{n'-1}$, а граф G'' — стороны $\alpha''_0, \dots, \alpha''_{n''-1}$. Какими будут стороны графа G в следующих случаях?

- a) $G = \overline{G}'$ представляет собой дополнение G' . (Считаем, что в нашем случае $e'_{ij} \leq i \neq j$.)
- b) $G = G' + G''$ представляет собой сумму (соединение) G' и G'' .
- c) $G = G' \dot{+} G''$ представляет собой сосумму (объединение) G' и G'' .
- d) $G = G' \times G''$ представляет собой декартово произведение G' и G'' .
- e) $G = L(G')$ представляет собой линейный граф G' , где G' является однородным графом степени d' (все вершины G' имеют ровно по d' соседей, и в графе нет петель).
- f) $G = G' \diamond G''$ представляет собой прямое произведение (конъюнкцию) G' и G'' , причем G' — однородный граф степени d' , а G'' — однородный граф степени d'' .
- g) $G = G' \otimes G''$ представляет собой сильное произведение однородных графов G' и G'' .

► 106. [HM37] Найдите общее количество остовых деревьев (a) в решетке $P_m \times P_n$ размером $m \times n$; (b) в цилиндре $P_m \times C_n$ размером $m \times n$; (c) в торе $C_m \times C_n$ размером $m \times n$. Почему эти числа имеют тенденцию состоять только из малых простых сомножителей? Указание: покажите, что стороны P_n и C_n можно выразить через числа $\sigma_{kn} = 4 \sin^2 \frac{k\pi}{2n}$.

107. [M24] Определите стороны всех связных графов, содержащих $n \leq 5$ вершин и не имеющих ни петель, ни параллельных ребер.

108. [HM40] Распространите результаты упражнений 104–106 на ориентированные графы.

109. [M46] Найдите комбинаторное пояснение тому факту, что выражение (57) представляет собой количество остовых деревьев в n -мерном кубе.

► 110. [M27] Докажите, что если G — произвольный связный мультиграф без петель, то он имеет $c(G) > \sqrt{(d_1 - 1) \dots (d_n - 1)}$ остовых деревьев, где d_j — степень вершины j .

111. [05] Перечислите узлы дерева (58) в обратно-прямом порядке обхода.

112. [15] Если узел p в лесу предшествует узлу q в прямо-обратном порядке и следует за ним в обратно-прямом, то что можно сказать об узлах p и q ?

► 113. [20] Как прямо-обратный и обратно-прямой порядки обхода леса F соотносятся с прямо-обратным и обратно-прямым порядками обхода сопряженного леса F^R (см. упражнение 13)?

114. [15] Если вы хотите обойти весь лес в прямо-обратном порядке с применением алгоритма Q , то с чего вы должны начать процесс?

115. [20] Проанализируйте алгоритм Q : как часто выполняется каждый из шагов алгоритма в процессе полного обхода леса?

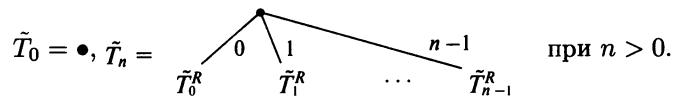
► 116. [28] Если узлы леса F помечены от 1 до n в прямо-обратном порядке, то будем называть узел k ($1 \leq k \leq n$) *везучим* (lucky), если он смежен с узлом $k+1$ в F ; *невезучим* (unlucky), если он удален на три шага; *обычным* (ordinary) в противном случае. При таком определении узел $n+1$ представляет собой воображаемый суперкорень, рассматриваемый как родительский для каждого корня.

- Докажите, что везучие узлы располагаются только на четных уровнях, а невезучие — только на нечетных.
- Покажите, что количество везучих узлов ровно на один больше количества невезучих узлов, кроме случая $n = 0$.

117. [21] Сколько n -узловых лесов не содержат невезучих узлов?

118. [M28] Сколько везучих узлов имеется (a) в полном t -арном дереве с $(t^k - 1)/(t - 1)$ внутренними узлами; (b) в дереве Фибоначчи порядка k , с $F_{k+1} - 1$ внутренними узлами? (См. 2.3.4.5–(6) и рис. 8 в разделе 6.2.1.)

119. [21] Скрученное биномиальное дерево \tilde{T}_n порядка n определяется рекурсивно правилами



(Сравните с 7.2.1.3–(21); мы обращаемся к порядку дочерних узлов на чередующихся уровнях.) Покажите, что прямо-обратный обход \tilde{T}_n имеет простую связь с бинарным кодом Грея.

120. [22] Истинно или ложно утверждение: квадрат графа гамильтонов, если график связный и не имеет мостов?

121. [M34] (Ф. Нойман (F. Neuman), 1964.) Производной графа G называется граф $G^{(1)}$, полученный путем удаления всех вершин степени 1 и ребер, которые с ними соединены. Докажите, что если T — свободное дерево, то его квадрат T^2 содержит гамильтонов путь тогда и только тогда, когда его производная не имеет вершин степени, большей 4, и выполняются два следующих условия:

- все вершины степени 3 или 4 в $T^{(1)}$ лежат на единственном пути;
- между любыми двумя вершинами степени 4 в $T^{(1)}$ есть, как минимум, одна вершина, степень которой в T равна 2.

► 122. [31] (Головоломка Дьюдени (Dudeney) о представлении сотни.) Имеется много интересных способов представить число 100 путем вставки арифметических операторов (и, возможно, скобок) в последовательность 123456789, например:

$$\begin{aligned} 100 &= 1 + 2 \times 3 + 4 \times 5 - 6 + 7 + 8 \times 9 = \\ &= (1 + 2 - 3 - 4) \times (5 - 6 - 7 - 8 - 9) = \\ &= ((1 / ((2 + 3) / 4 - 5 + 6)) \times 7 + 8) \times 9. \end{aligned}$$

- a) Сколько имеется таких представлений числа 100? Чтобы сделать этот вопрос точным, с учетом закона ассоциативности и других алгебраических свойств будем считать, что выражения записываются в каноническом виде в соответствии с приведенным далее синтаксисом.

$$\begin{aligned}\langle \text{expression} \rangle &\rightarrow \langle \text{number} \rangle \mid \langle \text{sum} \rangle \mid \langle \text{product} \rangle \mid \langle \text{quotient} \rangle \\ \langle \text{sum} \rangle &\rightarrow \langle \text{term} \rangle + \langle \text{term} \rangle \mid \langle \text{term} \rangle - \langle \text{term} \rangle \mid \langle \text{sum} \rangle + \langle \text{term} \rangle \mid \langle \text{sum} \rangle - \langle \text{term} \rangle \\ \langle \text{term} \rangle &\rightarrow \langle \text{number} \rangle \mid \langle \text{product} \rangle \mid \langle \text{quotient} \rangle \\ \langle \text{product} \rangle &\rightarrow \langle \text{factor} \rangle \times \langle \text{factor} \rangle \mid \langle \text{product} \rangle \times \langle \text{factor} \rangle \mid (\langle \text{quotient} \rangle) \times \langle \text{factor} \rangle \\ \langle \text{quotient} \rangle &\rightarrow \langle \text{factor} \rangle / \langle \text{factor} \rangle \mid \langle \text{product} \rangle / \langle \text{factor} \rangle \mid (\langle \text{quotient} \rangle) / \langle \text{factor} \rangle \\ \langle \text{factor} \rangle &\rightarrow \langle \text{number} \rangle \mid (\langle \text{sum} \rangle) \\ \langle \text{number} \rangle &\rightarrow \langle \text{digit} \rangle\end{aligned}$$

Используемые цифры — от 1 до 9 в указанном порядке.

- b) Расширим задачу (a), допуская использование чисел, состоящих из нескольких цифр:

$$\langle \text{number} \rangle \rightarrow \langle \text{digit} \rangle \mid \langle \text{number} \rangle \langle \text{digit} \rangle$$

Например, $100 = (1/(2 - 3 + 4)) \times 567 - 89$. Какое из таких представлений наиболее короткое и какое наиболее длинное?

- c) Расширим задачу (b), допуская использование десятичной точки:

$$\begin{aligned}\langle \text{number} \rangle &\rightarrow \langle \text{digit string} \rangle \mid \langle \text{digit string} \rangle . \langle \text{digit string} \rangle \\ \langle \text{digit string} \rangle &\rightarrow \langle \text{digit} \rangle \mid \langle \text{digit string} \rangle \langle \text{digit} \rangle\end{aligned}$$

Например, $100 = (.1 - 2 - 34 \times .5) / (.6 - .789)$.

- 123.** [21] Продолжая предыдущее упражнение, ответьте, какие наименьшие положительные целые числа, которые *не могут* быть представлены с использованием соглашений (a), (b) и (c)?

- **124.** [40] Эксперимент с методами черчения расширенных бинарных деревьев, вызванный простыми природными моделями. Например, мы можем назначить каждому узлу x значение $v(x)$, именуемое *числом Хортон–Страхлера* (Horton–Strahler number), следующим образом. Каждый внешний узел (лист) имеет значение $v(x) = 0$; внутренний узел с дочерними узлами (l, r) имеет значение $v(x) = \max(v(l), v(r)) + [v(l) = v(r)]$. Ребро от внутреннего узла x до его родителя можно изобразить как прямоугольник высотой $h(v(x))$ и шириной $w(v(x))$, а прямоугольники ребер к дочерним узлам (l, r) могут быть повернуты на углы $\theta(v(l(x)), v(r(x)))$ и $-\theta(v(r(x)), v(l(x)))$ для некоторых функций h , w и θ . На рис. 42 показаны типичные результаты при выборе $w(k) = 3 + k$, $h(k) = 18k$, $\theta(k, k) = 30^\circ$, $\theta(j, k) = ((k+1)/j) \times 20^\circ$ для $0 \leq k < j$ и $\theta(j, k) = ((k-j)/k) \times 30^\circ$ для $0 \leq j < k$; корни находятся внизу. На рис. 42,а показано бинарное дерево (4); на рис. 42,б — случайное 100-узловое дерево, сгенерированное алгоритмом R; на рис. 42,в — дерево Фибоначчи 11-го порядка со 143 узлами; а на рис. 42,г представлено случайное 100-узловое бинарное дерево поиска. (Очевидно, что деревья б–г относятся к различным видам.)



Рис. 42. “Органические” бинарные деревья

[Эта тема] связана почти со всеми полезными знаниями, которые только может воспринять человеческий разум.

— Якоб Бернулли (*James Bernoulli*).
Ars Conjectandi (*Искусство догадки*) (1713)

7.2.1.7 Исторические и иные сведения

Ранние работы по генерации комбинаторных шаблонов начались вместе с зарождением цивилизации. Это очень интересная история, и, как мы увидим, она охватывает многие части мира и многие области человеческой деятельности, включая поэзию, музыку и религию. Здесь будут рассмотрены только некоторые основные вопросы, но такой краткий экскурс в историю не может не стимулировать интерес читателя и не подвигнуть его на более глубокое изучение данной темы.

Списки всех кортежей из n элементов прослеживаются тысячи лет назад в древних Китае, Индии и Греции. Наиболее примечательный источник (в силу того, что эта книга в современном переводе стала бестселлером) — это китайская *I Ching*, или *Yijing*, что означает “Книга изменений”. В этой книге, одной из пяти классических книг конфуцианства, содержится ровно $2^6 = 64$ главы, каждая глава символизирует гексаграмму, образованную из шести линий, каждой из которых имеет вид либо — (‘инь’), либо — (‘ян’). Например, гексаграмма 1 состоит из линий ян ☰, гексаграмма 2 — из линий инь ☷, а гексаграмма 64 представляет собой чередующиеся ян и инь: ☱. Вот полный список гексаграмм:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64

(1)

Такое размещение 64 возможных вариантов называется упорядочением Кинг Веня, так как авторство книги *I Ching* приписывается Кинг Веню (King Wen) (около 1100 года до н. э.), легендарному основателю династии Чу. К сожалению, древние тексты трудно надежно датировать, так что современные историки не видят оснований считать, что такой список гексаграмм был составлен ранее III века до н. э.

Обратите внимание, что в перечне (1) гексаграммы располагаются парами: за гексаграммой с нечетным номером идет гексаграмма, представляющая собой зеркальное отражение относительно горизонтальной оси, кроме случаев, когда гексаграмма симметрична, — в этих случаях гексаграмма спарена со своим дополнением ($1 = \bar{2}$, $27 = \bar{28}$, $29 = \bar{30}$, $61 = \bar{62}$). Гексаграммы, состоящие из двух триграмм, представляющих четыре основных элемента — воздух (☰), землю (☷), огонь (☲) и воду (☵), также размещаются специальным образом. Остальные гексаграммы размещаются по сути случайному образом. Определенные шаблоны, существующие в расположении пар, не более чем совпадения, подобные совпадениям в записи числа π (см. 3.3–(1)).

Инь и ян представляют собой взаимодополняющие стороны элементарных сил природы, всегда находящиеся в противоборстве и переходящие друг в друга. В опре-

деленном смысле *I Ching* представляет собой тезаурус, в котором гексаграммы служат указателями к накопленным знаниям о фундаментальных концепциях наподобие следующих: давать (䷀), или получать (䷁), скромность (䷂), радость (䷃), товарищество (䷄), изъятие (䷅), мир (䷆), война (䷇), организация (䷈), коррупция (䷉), незрелость (䷊), изящество (䷋) и т.д. Можно выбрать пару гексаграмм случайным образом, получая вторую из первой, например независимо заменяя каждый инь на ян и наоборот с вероятностью 1/4; такой метод дает 4 096 способов размышлений об экзистенциалистских загадках, например: не может ли соответствующий марковский процесс приоткрыть тайну смысла жизни?

Строгий логический способ упорядочения гексаграмм был дан около 1060 года н. э. Шао Юнгом (Shao Yung). Его лексикографическое упорядочение от ䷀ к ䷁ к ䷂ к ䷃ к ䷄ к ䷅ к ䷆ к ䷇ к ䷈ к ䷉ к ䷊ (каждую гексаграмму надо читать снизу вверх) существенно более понятное, чем порядок Кинг Веня. Когда Г.В. Лейбниц (G.W. Leibniz) изучал эту последовательность гексаграмм в 1702 году, он пришел к ложному выводу о том, что китайские математики были знакомы с бинарной арифметикой. [См. Frank Swetz, *Mathematics Magazine*, 76 (2003), 276–291. Дополнительную информацию об *I Ching* можно найти, например, в Joseph Needham *Science and Civilization in China* 2 (Cambridge University Press, 1956), 304–345; R.J. Lynn, *The Classic of Changes* (New York : Columbia University Press, 1994).]

Другой древнекитайский философ, Янг Цинг (Yang Hsiung), предложил систему, основанную на 81 тернарной тетраграмме, вместо 64 гексаграмм. Его Канон высшего таинства (*Canon of Supreme Mystery*), написанный около 2 года до н. э., не так давно был переведен на английский язык Майклом Найланом (Michael Nylan) (New York : Albany, 1993). Янг описывает структуру полного иерархического тернарного дерева, в которой имеется три края, по три провинции в каждом крае, по три департамента в каждой провинции, по три семьи в каждом департаменте, и по девять коротких стихов на семью — итого 729 стихов, т.е. почти в точности два стиха на каждый день года. Его тетраграммы располагаются в строгом лексикографическом порядке при чтении сверху вниз: ䷀, ䷁, ䷂, ䷃, ䷄, ䷅, ䷆, ䷇, ..., ䷊. В действительности, как поясняется в книге Найлана (с. 28), Янг разработал простой способ для вычисления ранга каждой тетраграммы, как если бы он пользовался троичной системой счисления. Таким образом, нас не должно удивлять упорядочение бинарных гексаграмм Шао Юнгом, хотя он и жил более чем на 1000 лет раньше.

Индийская поэзия. Бинарные кортежи из n элементов изучались в совершенно ином контексте мудрецами в древней Индии, посвятившими себя священным ведическим песнопениям. Слоги в санскрите либо короткие (।), либо длинные (᳚), а изучение шаблонов слогов называется просодией (prosody). Современные писатели вместо символов । и ᳚ используют символы — и —. Типичная ведическая строфа состоит из четырех строк с n слогами в строке, для некоторого $n \geq 8$. Таким образом, нужен способ классифицировать все 2^n возможных шаблонов. Классическая работа Пингала (Pingala) Чандасутра (Chandasutra), написанная до 400 года н. э. (вероятно, существенно ранее — точная датировка неизвестна), описывает процедуру поиска индекса k любого заданного шаблона, состоящего из — и —, а также поиск шаблона для заданного значения k . Другими словами, Пингала поясняет, как найти ранг данного шаблона и как найти шаблон по заданному рангу. Таким об-

разом, он зашел в своих исследованиях дальше Янг Цинга (Yang Hsiung), который рассматривал только задачу поиска ранга, но не шаблона по заданному рангу. Методы Пингалы также связаны с возведением в степень, как мы видели ранее в связи с алгоритмом 4.6.3А.

Следующий важный шаг был сделан просодистом Кедарой (Kedāra) в его книге *Vṛttaratnākara*, которая, как считают, была написана в VIII веке. Кедара пошагово описал процедуру для перечисления всех n кортежей от $\sim\cdots\sim$ к $\sim\cdots\sim\sim\cdots\sim$, т.е., по сути, алгоритм 7.2.1.1М для основания системы счисления, равного 2. Его метод можно считать первым явным алгоритмом для генерации комбинаторной последовательности. [См. B. van Nooten, *J. Indian Philos.*, **21** (1993), 31–50.]

Стихотворные измерения можно также рассматривать как ритмы, с одним тактом для каждого \sim и с двумя для каждого $\sim\sim$. Хотя n -тактовый шаблон может включать от n до $2n$ тактов, музыкальные ритмы для маршей или танцев обычно основаны на фиксированном количестве тактов. Следовательно, естественно рассматривать множество всех последовательностей \sim и $\sim\sim$, содержащих фиксированное значение m тактов. Такие последовательности называются последовательностями кода Морзе (Morse) длины m , и из упражнения 4.5.3–32 известно, что их ровно F_{m+1} . Например, 21 последовательность для $m = 7$ имеет вид

$$\begin{aligned} &\sim\sim\sim\sim\sim\sim\sim, \sim\sim\sim\sim\sim\sim\sim, \sim\sim\sim\sim\sim\sim\sim, \\ &\sim\sim\sim\sim\sim\sim\sim, \sim\sim\sim\sim\sim\sim\sim, \sim\sim\sim\sim\sim\sim\sim, \sim\sim\sim\sim\sim\sim\sim. \end{aligned} \tag{2}$$

Таким образом, индийские просодисты пришли к открытию последовательности Фибоначчи, как уже отмечалось в разделе 1.2.8.

Кроме того, анонимный автор *Prākṛta Paiṅgala* (около 1320 года) разработал элегантный алгоритм для определения ранга ритма и восстановления ритма по рангу для m -тактовых ритмов. Для поиска k -го шаблона начинаем с записи m символов \sim , затем выражаем разность $d = F_{m+1} - k$ как сумму чисел Фибоначчи $F_{j_1} + \dots + F_{j_t}$; здесь F_{j_1} — наибольшее число Фибоначчи, не превышающее d , F_{j_2} — наибольшее число Фибоначчи, не превышающее $d - F_{j_1}$, и так до тех пор, пока не будет получен нулевой остаток. Тогда такты $j - 1$ и j заменяются с $\sim\sim$ на \sim для $j = j_1, \dots, j_t$. Например, для получения пятого элемента последовательности (2) мы вычисляем $21 - 5 = 16 = 13 + 3 = F_7 + F_4$; ответом является $\sim\sim\sim\sim$.

Несколько годами позже Нараяна Пандита (*Nārāyaṇa Paṇḍita*) рассматривал более общую задачу поиска всех композиций m , части которых не превышают q , где q — любое данное положительное целое число. Как следствие, он открыл последовательности Фибоначчи q -го порядка 5.4.2–(4), которые пригодились через 600 лет после этого в многофазной сортировке; он также открыл соответствующие алгоритмы определения ранга и элемента по заданному рангу. [См. Parmanand Singh, *Historia Mathematica*, **12** (1985), 229–244, а также упражнение 16.]

Пингала дает специальные имена всем трехсложным измерениям:

$$\begin{array}{ll} \text{---} = \text{म (m)}, & \text{---} = \text{त (t)}, \\ \text{---} = \text{य (y)}, & \text{---} = \text{ज (j)}, \\ \text{---} = \text{र (r)}, & \text{---} = \text{भ (bh)}, \\ \text{---} = \text{स (s)}, & \text{---} = \text{न (n)}. \end{array} \quad (3)$$

И с тех пор все изучающие санскрит должны заучивать их наизусть. Давным-давно кто-то разработал способ запоминания этих кодов, придумав несуществующее слово *yamātārājabhānasalagām* (यमाताराजभानसलगाम्). Смысл в том, что десять слогов этого слова можно записать как

ya mā tā rā ja bhā na sa la gām, (4)

после чего каждый трехсложный шаблон начинается с его имени. Происхождение слова *yamātarājabhānasalagām* неизвестно; Сабхаш Как (Subhash Kak) [*Indian J. History of Science*, 35 (2000), 123–127] проследил его до книги С.Р. Брауна *Sanskrit Prosody* (1869), с. 28; таким образом, его можно считать наиболее ранним известным “циклом де Брейна” для кодирования бинарных n -кортежей.

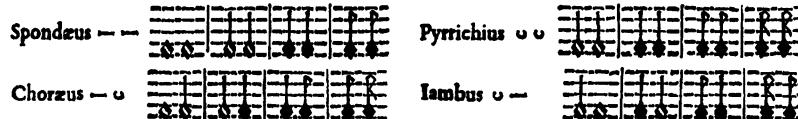
Тем временем в Европе. Классическая греческая поэзия также основывалась на группах коротких и/или длинных слогов, называемых метрической стопой (metrical feet), аналогично тактам в музыке. Например, два коротких слога ‘—’ назывались *типтихий*, а два длинных ‘— —’ — *спондей*, поскольку эти ритмы использовались соответственно в песнях войны (*типтихη*) и мира (*спονδαι*). Греческие названия метрических стоп ассимилировались латынью и в конечном счете современными языками, в том числе английским:

— арсис (arsis)	———— гиперпирихий (прокелевсаматик) (procelesmatic)
— тезис (thesis)	
— пирихий (pyrrhic)	———— четвертый пеон (fourth paeon)
— ямб (iambus)	———— третий пеон (third paeon)
— трохей (хорей) (trochee)	———— ионик меньший (нисходящий) (minor ionic)
— спондей (spondee)	———— второй пеон (second paeon)
— трибрахий (tribrach)	———— диямб (diambus)
— анапест (anapest)	———— антиспаст (antispast)
— амфибрахий (amphibrach)	———— первый эпитрит (first epitrite) (5)
— бакхий (bacchius)	———— первый пеон (first paeon)
— дактиль (dactyl)	———— хориямб (choriambus)
— амфимакр (amphimacer)	———— дитрохей, дихорей (ditrochée)
— антибакхий (palimbacchius)	———— второй эпитетрит (second epitrite)
— молосс (molossus)	———— ионик больший (восходящий) (major ionic)
	———— третий эпитетрит (third epitrite)
	———— четвертый эпитетрит (fourth epitrite)
	———— диспондей (dispondee)

Зачастую используются альтернативные имена, например хорей (*choree*) вместо трохей (*trochee*), кретик (*cretic*) вместо амфимакр (*amphimacer*). Кроме того, во времена, когда Диомед (*Diomedes*) писал свою латинскую грамматику (около 375 года), как минимум по одному имени было у каждой из 32 пятисложных стоп.

Диомед также указал на отношения между дополняющими шаблонами; он привел пример трибрахия и молосса как *противоположностей* (*contrarius*), подобно амфибрахию и амфимакру. Однако он рассматривал как противоположности дактиль и анапест, а также бакхий и антибакхий, хотя сам термин *palimbacchius* обозначает не что иное, как “обратный *bacchius*”. У греков не имелось стандартного порядка перечисления стоп, а их названия никак не связаны с бинарными числами, которые они представляют. [См. H. Keil, *Grammatici Latini*, 1 (1857), 474–482; W. von Christ, *Metrik der Griechen und Römer* (1879), 78–79.]

Дошедшие до нас фрагменты работы Аристоксена (*Aristoxenus*) Элементы ритмов (около 325 года до н. э.) показывают, что та же терминология была применима и к музыке. Те же традиции пережили эпоху возрождения; например, в книге Athanasius Kircher *Musurgia Universalis*, 2 (Rome, 1650) на стр. 32 можно встретить следующий фрагмент:



Кирхер описал все трех- и четырехнотные ритмы, соответствующие (5).

Ранние списки перестановок. Мы уже проследили историю формулы для подсчета перестановок в разделе 5.1.2; однако нетривиальные списки перестановок не публиковались еще сотни лет после того, как формула $n!$ была открыта. Первая из известных ныне таблиц была создана итальянским врачом Шаббетай Донноло (*Shabbetai Donnolo*) в его комментариях к каббалистической книге *Sefer Yetzirah*, написанной в 946 году. В табл. 1 показан его список для $n = 5$ в том виде, в каком он был опубликован в Варшаве в 1884 году. (Буквы иврита в этой таблице используют средневековый шрифт, традиционно применявшийся в комментариях; обратите внимание, что буква **ו** заменяется буквой **ו** на левом конце слова.) Донноло перечислил все 120 перестановок шестибуквенного слова **בָּנְיַתְהָ**, начинающиеся с **ב**; затем он заметил, что другие 120 перестановок можно получить при использовании другой буквы в качестве первой, т.е. всего 720 перестановок. Его список включает группирование шести перестановок, но оно выполнено случайным образом, что и привело его к ошибке (см. упражнение 4). Хотя он знал общее количество перестановок и количество перестановок, начинающихся с данной буквы, алгоритма для их генерации у Донноло не было.

Полный список всех 720 перестановок множества $\{a, b, c, d, e, f\}$ имеется в книге Jeremias Drexel *Orbis Phaethon* (Munich, 1629), на с. 668–671; в Кельнском издании 1631 года этот список находится на с. 526–531.) Он привел его в качестве доказательства того, что хозяин может размещать шестерых гостей за завтраком и обедом разными способами в течение года — всего 360 дней; пять дней в году он отвел для поста на Страстной неделе. Немного позже Marin Mersenne (Marin Mersenne) привел все 720 перестановок шести нот $\{\text{до, ре, ми, фа, соль, ля}\}$ в своей книге *Traitez*

Таблица 1. Средневековый список перестановок

de la Voix et des Chants (Том 2. *Harmonie Universelle*, 1636) на с. 111–115; далее на с. 117–128 он предоставил ту же информацию в нотной записи:



Таблица Дрекселя организована лексикографически по столбцам; таблица Мерсенна лексикографически упорядочена в соответствии с отношением до $<$ ре $<$ ми $<$ фа $<$ соль $<$ ля; она начинается с перестановки “до, ре, ми, фа, соль, ля” и заканчивается “ля, соль, фа, ми, ре, до”. Мерсенн также подготовил огромную рукопись, в которой перечислил все 40 320 перестановок *восьми* нот на 672 страницах размером ин-фолио, после чего привел алгоритмы для вычисления ранга перестановки и перестановки по рангу [Bibliothèque nationale de France, Fonds Français, no. 24 256].

Из раздела 7.2.1.2 мы знаем, что важные идеи алгоритма 7.2.1.2Р были разработаны в Англии несколькими годами позже.

Методы перечисления всех перестановок мультимножества с *повторяющимися* элементами часто неверно понимались ранними авторами. Например, когда Бхаскара (Bhāskara) представил перестановки множества $\{4, 5, 5, 5, 8\}$ в разделе 271 своей книги *Līlāvatī* (около 1150 года), он привел их в следующем порядке:

Мерсенн использовал более практичный, но не совсем систематический порядок при перечислении 60 анаграмм латинского имени IESVS на с. 131 своей книги. Когда Атанасиус Кирхер (Athanasius Kircher) хотел проиллюстрировать 30 перестановок пятинотной мелодии на с. 10–11 книги *Musurgia Universalis*, 2 (1650), такое отсутствие системы привело к неприятностям (см. упражнение 5).



Джон Уоллис (John Wallis) лучше справился со своей задачей. На с. 117 своей книги *Discourse of Combinations* (1685) он корректно перечислил 60 анаграмм слова “messes” в лексикографическом порядке, если считать $m < e < s$; а также рекомендовал соответствующий алфавитный порядок (с. 126).

Позже мы увидим, что индийские ученые Сарнгадева Śāringadeva и Нараяна Nārāyaṇa разработали теорию генерации перестановок в XIII и XIV веках, но эта работа опередила свое время и осталась незамеченной.

Список Секи. Такаказу Секи (Takakazu Seki) (1642–1708) был харизматичным учителем и ученым, революционизировавшим изучение математики в Японии в XVII веке. При изучении исключения переменных из систем однородных уравнений он пришел к выражениям $a_1b_2 - a_2b_1$ и $a_1b_2c_3 - a_1b_3c_2 + a_2b_3c_1 - a_2b_1c_3 + a_3b_1c_2 - a_3b_2c_1$, которые сегодня известны как *определители* (*детерминанты*). В 1683 году он опубликовал брошюру с этим открытием, в которую включил остроумную схему для перечисления всех перестановок таким образом, чтобы половина из них была “живая” (четна), а половина — “мертвая” (нечетна). Начиная со случая $n = 2$, когда ‘12’ — живо, а ‘21’ — мертвое, он сформулировал следующие правила для $n > 2$.

- 1) Возьми каждую живую перестановку для $n - 1$, увеличь все ее элементы на 1 и вставь впереди 1. Это правило дает $(n - 1)!/2$ “базовых перестановок” $\{1, \dots, n\}$.
- 2) Из каждой базовой перестановки образуй $2n$ других путем поворотов и отражений:

$$a_1a_2 \dots a_{n-1}a_n, a_2 \dots a_{n-1}a_na_1, \dots, a_na_1a_2 \dots a_{n-1}; \quad (8)$$

$$a_na_{n-1} \dots a_2a_1, a_1a_na_{n-1} \dots a_2, \dots, a_{n-1} \dots a_2a_1a_n. \quad (9)$$

Если n нечетно, перестановки в первой строке живые, а во второй — мертвые.

Если n четно, перестановки в каждой строке чередуются — живая, мертвая, …, живая, мертвая.

Например, при $n = 3$ единственной базовой перестановкой является 123. Таким образом, 123, 231, 312 — живые, в то время как 321, 132 и 213 — мертвые, и мы успешно генерируем шесть членов определителя 3×3 . Базовыми перестановками при $n = 4$ являются 1234, 1342, 1423; скажем, из 1342 мы получаем множество из восьми элементов, а именно

$$+1342 - 3421 + 4213 - 2134 + 2431 - 1243 + 3124 - 4312, \quad (10)$$

с чередующимися живыми (+) и мертвыми (−) перестановками. Определитель 4×4 , таким образом, включает члены $a_1b_3c_4d_2 - a_3b_4c_2d_1 + \dots - a_4b_3c_1d_2$, а также 16 других.

Правило Секи для генерации перестановок достаточно привлекательно, но, к сожалению, оно не работает при $n > 4$. Похоже, эта ошибка оставалась незамеченной столетиями. [См. Y. Mikami, *The Development of Mathematics in China and Japan* (1913), 191–199; *Takakazu Seki's Collected Works* (Osaka, 1974), 18–20.]

Списки сочетаний. Наиболее ранний исчерпывающий список *сочетаний*, выдержавший разрушительное действие времени, находится в главе 63 последней книги Сусруты (*Suśruta*), представляющей собой трактат о медицине и написанной до 600 года (вероятно, намного ранее). Заметив, что лекарства могут быть сладкими, кислыми, солеными, острыми, горькими и/или вяжущими, Сусрута старательно перечислил все (15, 20, 15, 6, 1, 6) случаев, когда одновременно встречаются два, три, четыре, пять, шесть и одно из этих качеств.

Бхаскара (*Bhāskara*) повторил этот пример в разделах 110–114 своей книги *Līlāvatī* и заметил, что те же самые рассуждения применимы и к шестисложным стихотворным шаблонам с заданным количеством длинных слов. Однако он просто упомянул конечные их количества (6, 15, 20, 15, 6, 1), без перечисления самих сочетаний. В разделах 274 и 275 он заметил, что числа $(n)(n-1)\dots(n-k+1)/(k(k-1)\dots(1))$ указывают наряду с количеством сочетаний количество *композиций* (compositions), т.е. упорядоченных разбиений, однако это замечание также сделано без приведения полного списка.

*Чтобы избежать многословия,
рассмотрим этот вопрос вкратце,
поскольку наука о вычислениях — это океан без границ.
— Бхаскара (*Bhāskara*) (около 1150)*

Интересный список сочетаний имеется в знаменитой работе по алгебре *Al-Bāhir fi'l-hisāb* (Блистательная книга о вычислениях), написанной аль-Самавалем (*al-Samaw'al*) из Багдада в 1144 году, когда ему было всего 19 лет. В заключительной части работы он представил список из $\binom{10}{6} = 210$ линейных уравнений с 10 неизвестными:

٦٥	٧٥٤٣٢١	١		(1) $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 65$
٧٠	٧٥٤٣٢١	ـ		(2) $x_1 + x_2 + x_3 + x_4 + x_5 + x_7 = 70$
٨٥	٨٥٤٣٢١	ــ		(3) $x_1 + x_2 + x_3 + x_4 + x_5 + x_8 = 75$
⋮				
٩١	١٠٩٨٧٦٤	ـــ		(209) $x_4 + x_6 + x_7 + x_8 + x_9 + x_{10} = 91$
١٠٠	١٠٩٨٧٦٥	ــــ		(210) $x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} = 100$

(11)

Каждое сочетание шести из десяти элементов дает одно из уравнений. Целью аль-Самавала, несомненно, была демонстрация переопределенной системы линейных уравнений, которая, тем не менее, имеет единственное решение; для приведенной системы оно имеет вид $(x_1, x_2, \dots, x_{10}) = (1, 4, 9, 16, 25, 10, 15, 20, 25, 5)$. [Salah Ahmad and Roshdi Rashed, *Al-Bāhir en Algèbre d'As-Samaw'al* (Damascus, 1972), 77–82, ٢٤٨–٢٢١.]

Игральные кости. Определенный проблеск в области элементарной комбинаторики был и в средневековой Европе, особенно в связи с вопросом о всех возможных сочетаниях выпадения трех костей. Разумеется, существует всего $\binom{8}{3} = 56$ способов выбрать три элемента из шести при разрешенных повторениях. Азартные игры были официально запрещены, но это не помешало упомянутым 56 способам стать общеизвестными. Около 965 года епископ Вибольд (Wibold) из Камбраи на севере Франции, разработал игру под названием *Ludus Clericalis*, дабы ею могли наслаждаться смиренные пастыри, оставаясь при этом достаточно набожными. Его идея состояла в том, чтобы связать с каждым набором очков одну из 56 добродетелей в соответствии с приведенной таблицей.

любовь	великодушие	молитва
вера	мудрость	привязанность
надежда	сожаление	суд Божий
справедливость	радость	бдительность
благоразумие	воздержанность	покорность
умеренность	удовлетворенность	невинность
храбрость	безмятежность	раскаяние
дружелюбие	мастерство	исповедь
целомудрие	простота	зрелость
милосердие	гостеприимство	забота
послушание	бережливость	постоянство
страх Божий	терпение	ум
предусмотрительность	усердие	томление
осторожность	бедность	плач
настойчивость	мягкость	веселье
доброжелательность	девственность	сочувствие
скромность	уважение	самообладание
смирение	благочестие	повинование
доброта	снисхождение	

Игроки бросают кости, и первый, кто выбросил сочетание очков, соответствующее некоторой добродетели, получает ее. После того как все сочетания окажутся выброшенными, победителем становится наиболее добродетельный из игроков. Вибольд заметил, что любовь — наилучшая добродетель из всех. Он разработал сложную систему подсчета очков, по которой две добродетели могли комбинироваться, если сумма очков на всех шести костях для этих добродетелей равнялась 21. Например, так можно скомбинировать любовь и повинование или целомудрие и ум, и такие комбинации ценились выше любых отдельных добродетелей. Вибольд рассматривал и более сложные варианты игры, в которых вместо точек на костях использовались гласные буквы.

Таблица Вибольда была представлена в лексикографическом порядке, как это сделано выше, при первом описании игры Балдериком (Balderic) в его *Chronicon Cameracense* около 150 лет спустя. [Patrologia Latina, 134 (Paris, 1884), 1007–1016.]

Однако в другом средневековом манускрипте возможные результаты бросания kostей представлены в совершенно ином порядке:

В этом случае автор знал, как следует поступить с повторяющимися значениями, но использовал очень сложный, разработанный для данной конкретной задачи способ обработки ситуаций, когда очки на всех трех костях различны. [См. D.R. Bellhouse, *International Statistical Review*, **68** (2000), 123–136.]

Забавное стихотворение “Chaunce of the Dyse”, приписываемое Джону Лидгейту (John Lydgate), было написано в начале 1400-х годов для вечеринок. Его начальные строфы приглашают каждого человека бросить три игральные кости; в остальных строфах, индексированных в обратном лексикографическом порядке от ☐☐☐ до ☐☐☒ до ... до ☐☐☐, приводится 56 веселых описаний характера бросающего. [Полный текст опубликован Э.П. Хаммонд (E.P. Hammond) в *Englische Studien*, 59 (1925), 1–16; перевод на современный английский был бы очень кстати.]

*I pray to god that euery wight may caste
Vpon three dyse ryght as is in hys herte
Whether he be rechelesse or stedfaste
So moote he lawghen outher elles smerte
He that is guilty his lyfe to converte
They in trouthe haue suffred many a throwe
Moote ther chaunce fal as they moote be knowe.*

Раймунд Луллий. Значительный вклад в комбинаторные концепции внесен энергичным донкихотствующим каталонским поэтом, писателем-романистом, энциклопедистом, преподавателем, мистиком и миссионером по имени Раймон Луллий (Ramon Llull) (около 1232–1316). Подход Луллия к знаниям состоял, по сути, в определении базовых принципов с последующим рассмотрением всех возможных их сочетаний.

Например, одна из глав его *Ars Compendiosa Inveniendi Veritatem* (около 1274) начинается с перечисления шестнадцати свойств, присущих Богу: доброта, величие, бессмертие, могущество, мудрость, любовь, добродетель, истина, слава, совершенство, справедливость, великодушие, милосердие, скромность, владычество и терпение. Затем Луллий пишет $\binom{16}{2} = 120$ кратких эссе (около 80 слов каждое), рассматривающих доброту Бога в связи с величием, доброту в связи с бессмертием и т.д., завершая рассмотрением владычества Бога в связи с терпением. В другой главе

он рассматривает семь добродетелей (веру, надежду, милосердие, справедливость, благородство, силу духа и умеренность) и семь пороков (чревоугодие, похоть, жадность, леность, гордыня, зависть и гнев), написав $\binom{14}{2} = 91$ подраздел, в каждом из которых попарно рассматриваются добродетель и порок. Другие главы так же систематически разделены на $\binom{8}{2} = 28$, $\binom{15}{2} = 105$, $\binom{4}{2} = 6$ и $\binom{16}{2} = 120$ подразделов. (Интересно, что было бы, если бы Луллий был знаком со списком Вибольда из 56 добродетелей? Написал бы он комментарии к каждой из $\binom{56}{2} = 1540$ пар?)

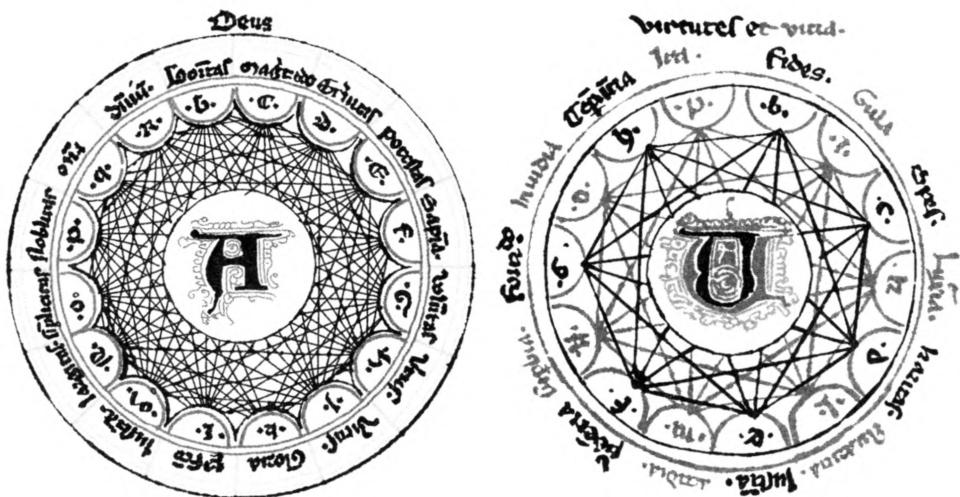


Рис. 44. Иллюстрация из манускрипта, подаренного Раймундом Луллием венецианскому дожу в 1280 году [Из его *Ars Demonstrativa*, Biblioteca Marciana, VI 200, folio 3^V]

Луллий иллюстрировал свою методологию, изображая круговые диаграммы, наподобие приведенных на рис. 44. Фигура слева, *Deus*, именует 16 божественных свойств — те же, что перечислены выше, за исключением того, что любовь (*amor*) здесь названа волей (*voluntas*), а последними четырьмя являются соответственно простота, ранг, милосердие и владычество. Каждому свойству назначена буква, и диаграмма иллюстрирует их взаимоотношения в виде полного графа K_{16} на множестве вершин ($B, C, D, E, F, G, H, I, K, L, M, N, O, P, Q, R$). Фигура справа, *virtutes et vicia*, показывает семь добродетелей (b, c, d, e, f, g, h), чередующихся с семью пороками (i, k, l, m, n, o, p); в оригинале рукописи добродетели изображены синими чернилами, а пороки — красными. Обратите внимание, что в этом случае на иллюстрации содержатся два независимых полных графа K_7 разных цветов. (Луллий больше не пытается сравнивать каждую отдельную добродетель с каждым отдельным пороком, поскольку и так понятно, что каждая добродетель лучше каждого порока.)

Луллий использовал тот же подход и когда писал о медицине: вместо сопоставления теологических концепций его книга *Liber Principiorum Medicinæ* (около 1275) рассматривает сочетания симптомов и лечения. Луллий также писал книги о философии, логике, юриспруденции, астрологии, зоологии, геометрии, риторике и рыцарстве — всего более 200 работ. Следует отметить, однако, высокую повторяемость его книг. Современные методы сжатия данных, вероятно, привели бы к тому, что

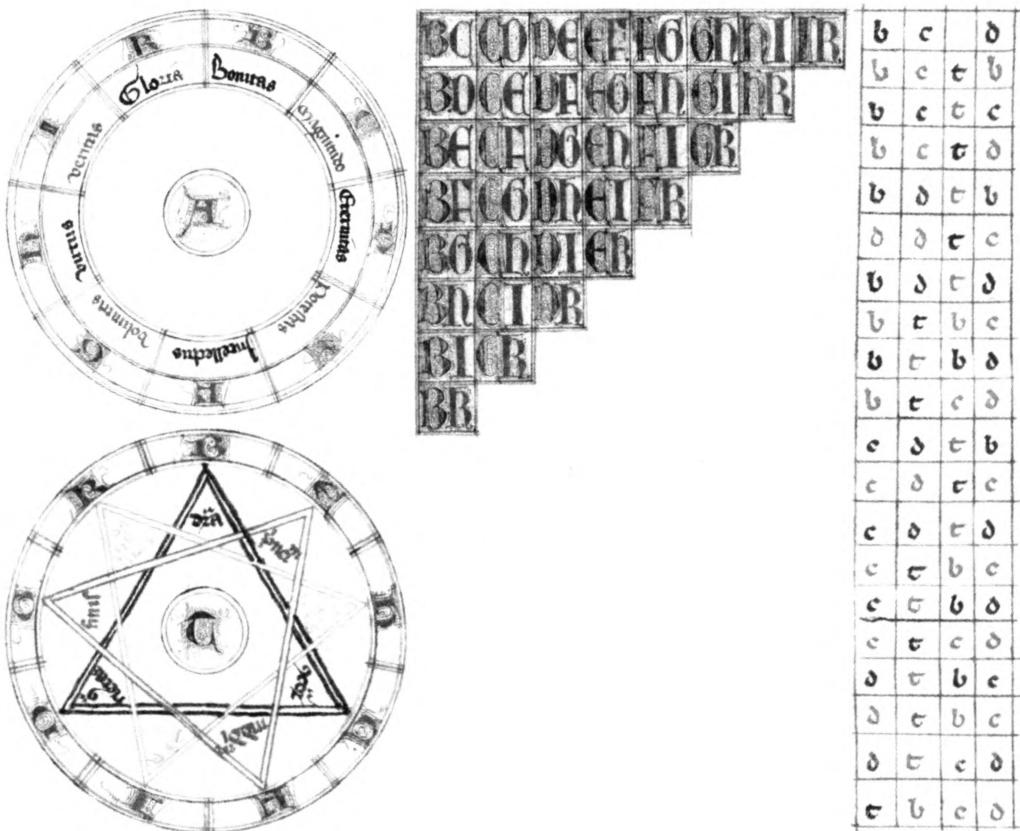


Рис. 45. Иллюстрации из манускрипта Луллия, подаренного королеве Франции (около 1325 года) [Badische Landesbibliothek Karlsruhe, Codex St. Peter perg. 92, folios 28^V and 39^V]

труды Луллия после сжатия были бы существенно меньшего размера, чем, скажем, Аристотеля.

В конечном счете Луллий решил упростить свою систему, работая в основном с группами из девяти элементов. Взгляните, например, на рис. 45, где в окружности, изображенной слева вверху, перечислены только первые девять из всех свойств, присущих Богу ($B, C, D, E, F, G, H, I, K$). В ступенчатой диаграмме, расположенной рядом с окружностью, приведены $\binom{9}{2} = 36$ пар свойств (BC, BD, \dots, IK). Добавляя к семи две добродетели — терпение и сострадание, и к семи два порока — ложь и непостоянство, Луллий рассматривает пары добродетелей и пороков при помощи той же диаграммы. Он предложил также использовать эту диаграмму для проведения выборов при девяти кандидатах [см. I. McLean and J. London, *Studia Lulliana*, 32 (1992), 21–37].

Вписанные в окружность треугольники (рис. 45, слева внизу) иллюстрируют еще один ключевой аспект подхода Луллия. Треугольник (B, C, D) означает (различность, совместимость, противоположность), треугольник (E, F, G) — (начало, средина, конец), а треугольник (H, I, K) — (больше, равно, меньше). Эти три вложенных

графа K_3 представляют три вида трехзначной логики. Луллий ранее экспериментировал с другими такими тройками, в особенности тройкой (истинно, неизвестно, ложно). Представление о том, как он использовал эти треугольники, можно получить, познакомившись с анализом Луллием четырех основных элементов (земля, воздух, огонь, вода). Все четыре элемента различны. Земля совместима с огнем, который совместим с воздухом, который совместим с водой, которая совместима с землей. Земля является противоположностью воздуху, а огонь — воде. Этим завершается анализ с использованием треугольника (B, C, D) . Переходя к треугольнику (E, F, G) , Луллий замечает, что разные процессы в природе начинаются при доминировании одного элемента над другим; затем выполняется переход, или среднее состояние, после чего достигается конечное состояние, например воздух становится горячим. По поводу треугольника (H, I, K) Луллий говорит, что в общем случае огонь $>$ воздух $>$ вода $>$ земля в плане их “сфер”, “скоростей” и “знатности”; тем не менее мы также имеем, например, воздух $>$ огонь по отношению к поддержанию жизни, а при совместной работе воздух и огонь равны.

На рис. 45 справа приведена представленная Луллием вертикальная таблица, смысл которой станет понятен из упражнения 11. Он также разработал подвижные концентрические колеса, помеченные буквами $(B, C, D, E, F, G, H, I, K)$ и другими именами, что позволило ему работать со многими разнообразными типами элементов одновременно. Работая таким способом, верный последователь Луллия мог быть уверен, что рассматривает все возможные случаи. [Луллий мог видеть подобные колеса в ближайшей еврейской общине; см. M. Idel, *J. Warburg and Courtauld Institutes*, 51 (1988), 170–174, а также вклейки 16–17.]

Несколько столетиями позже Атанасиус Кирхер (Athanasius Kircher) опубликовал развитие системы Луллия как часть большого тома, озаглавленного *Ars Magna Sciendi sive Combinatoria* (Amsterdam, 1669), в котором на с. 173 имелось пять подвижных колес. Кирхер также расширил множество используемых Луллием полных графов K_n , приведя иллюстрации полных двудольных графов $K_{m,n}$; например, рис. 46 взят со с. 171 книги Кирхера; на с. 170 той же книги показан граф $K_{18,18}$.

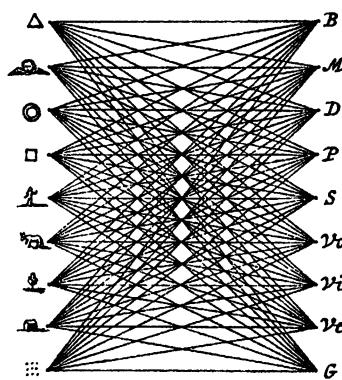


Рис. 46. Граф $K_{9,9}$, опубликованный Кирхером в 1669 году

Это искусство исследовать и изобретать.

*Когда идеи комбинируются всеми возможными способами,
новые комбинации открывают новые пути для размышлений
и приводят к открытию новых истин и аргументов.*

— Мартин Гарднер (*Martin Gardner*),
Логические машины и диаграммы (Logic Machines and Diagrams) (1958)

Наиболее далеко идущим современным развитием методов в духе Луллия, вероятно, является работа Иосифа Шиллингера (*Joseph Schillinger*) *The Schillinger System of Musical Composition* (New York : Carl Fischer, 1946), замечательный двухтомник, рассматривающий теории ритма, мелодии, гармонии, контрапунктов, композиции, оркестровки и прочего с точки зрения комбинаторики. Например, на с. 56 Шиллингер перечисляет 24 перестановки $\{a, b, c, d\}$ в порядке кода Грея (алгоритм 7.2.1.2Р); на с. 57 он применяет их не к мелодии, а к ритму, к длительности нот; на с. 364 он приводит симметричный цикл

$$(2, 0, 3, 4, 2, 5, 6, 4, 0, 1, 6, 2, 3, 1, 4, 5, 3, 6, 0, 5, 1), \quad (13)$$

универсальный цикл 2-сочетаний для семи объектов $\{0, 1, 2, 3, 4, 5, 6\}$; другими словами, (13) представляет собой эйлеров путь в K_7 : все $\binom{7}{2} = 21$ пар цифр встречаются ровно по одному разу. Такие шаблоны — настоящая золотоносная жила композитора, но мы должны быть благодарны, что лучшие ученики Шиллингера (наподобие Джорджа Гershвина (*George Gershwin*)) все же не ограничились строго математическим восприятием эстетики.

Тако, ван Шутен и Изкуэрдо. В 1650-х годах были опубликованы еще три книги, связанные с рассматриваемой нами темой. Андре Тако (*André Tacquet*) написал общедоступную книгу *Arithmeticae Theoria et Praxis* (Louvain, 1656), которая перепечатывалась и исправлялась в течение следующих 50 лет. В конце книги (с. 376 и 377) он привел процедуру для перечисления сочетаний по два, по три и т.д.

Книга Франца ван Шутена (*Frans van Schooten*) *Exercitationes Mathematicæ* (Leiden, 1657) была более серьезной. На с. 373 в ней перечислены все сочетания в виде привлекательной схемы

$$\begin{array}{c} a \\ \overline{b. ab} \\ \hline \overline{c. ac. bc. abc} \\ \hline \overline{d. ad. bd. abd. cd. acd. bcd. abcd} \end{array}, \quad (14)$$

и следующие несколько страниц посвящены расширению данного шаблона при введении букв e, f, g, h, i, k “и так до бесконечности”. На с. 376 автор замечает, что в выражении (14) можно заменить (a, b, c, d) на $(2, 3, 5, 7)$ и получить делители 210, превышающие единицу.

$$\begin{array}{r} 2 \\ \overline{3 \ 6} \\ \hline \overline{5 \ 10 \ 15 \ 30} \\ \hline \overline{7 \ 14 \ 21 \ 42 \ 35 \ 70 \ 105 \ 210} \end{array} \quad (15)$$

На следующей странице начальная идея распространяется на

$$\begin{array}{c} a \\ \overline{a. \ aa} \\ \overline{b. \ ab. \ aab} \\ \hline c. \ ac. \ aac. \ bc. \ abc. \ aabc \end{array}, \quad (16)$$

таким образом позволяя иметь две буквы a . Тем не менее в действительности автор не понял это расширение; его следующий пример

$$\begin{array}{c} a \\ \overline{a. \ aa} \\ \overline{a. \ aaa} \\ \overline{b. \ ab. \ aab. \ aaab} \\ \hline b. \ bb. \ abb. \ aabb. \ aaabb \end{array} \quad (17)$$

выполнен неверно, указывая границы знаний того времени (см. упражнение 13).

На с. 411 ван Шутен заметил, что веса $(a, b, c, d) = (1, 2, 4, 8)$ в (14) при использовании сложения дают

$$\begin{array}{r} 1 \\ \overline{2 \ 3} \\ \overline{4 \ 5 \ 6 \ 7} \\ \hline 8 \ 9 \ 10 \ 11 \ 12 \ 13 \ 14 \ 15 \end{array}. \quad (18)$$

Однако он не увидел здесь связи с двоичными числами.

Двухтомник Себастьяна Изкуэрдо (Sebastián Izquierdo) *Pharus Scientiarum* (Lyon, 1659) — “Маяк науки” — включает в себя хорошо организованное обсуждение комбинаторики под заглавием *Disputatio 29, De Combinatione*. Автор приводит детальное обсуждение четырех ключевых частей “двенадцатизадачия” Стенли, а именно n -кортежей, n -размещений, n -мульти сочетаний и n -сочетаний из m объектов, которые находятся в первых двух строках и двух столбцах табл. 7.2.1.4-1.

В разделах 81–84 *De Combinatione* автор перечисляет (в лексикографическом порядке) все сочетания из m букв по n , для $2 \leq n \leq 5$ и $n \leq m \leq 9$; он также протабулировал их для $m = 10$ и 20 , при $n = 2$ и 3 . Однако при перечислении m^n размещений m элементов по n он использовал более сложный порядок (см. упражнение 14).

Изкуэрдо был первым, кто открыл формулу $\binom{m+n-1}{n}$ для сочетаний m элементов по n с неограниченными повторениями; это правило встречается в § 48–51 его книги. Однако в § 105, когда он пытается перечислить все такие сочетания при $n = 3$, он не знает, что существует простой способ сделать это. Фактически его перечисление 56 случаев для $m = 6$ больше напоминает старый запутанный порядок (12).

Сочетания с повторениями не были хорошо поняты до тех пор, пока в 1713 году не вышла книга Якоба Бернулли (James Bernoulli) *Ars Conjectandi* (“Искусство догадки”). В части 2, главе 5 Бернулли просто перечислил все возможности в лексикографическом порядке и показал, что формула $\binom{m+n-1}{n}$ является простым следствием применения индукции. [Никколо Тарталья (Niccolò Tartaglia), кстати, близко подошел к получению этой формулы в своей книге *General trattato di numeri, et misure*, 2 (Venice, 1556), 17^r и 69^v; то же справедливо и в отношении магрибского математика Ибн Мунима (Ibn Mun'im) и его книги *Fiqh al-Hisāb*, написанной в XIII веке.]

Нулевой случай. Перед тем как завершить наше рассмотрение ранних работ по сочетаниям, следует не забыть небольшой, но важный шаг, сделанный Джоном Уоллисом (John Wallis) в книге *Discourse of Combinations* (1685), где на с. 110 он, в частности, рассматривает сочетание 0 объектов из m элементов: “очевидно, что, когда мы *выбираем Ничто*, то мы *оставляем Все*. Это можно сделать только одним способом, сколько бы объектов у нас ни было”. Кроме того, на с. 113 он утверждает, что $\binom{0}{0} = 1$: “в этом случае выбрать все или оставить все — это одно и то же”.

Однако, приводя таблицу факториалов $n!$ для $n \leq 24$, он нешел так далеко, чтобы указать, что $0! = 1$ или что имеется ровно одна перестановка пустого множества.

Работа Нааяны. Замечательная монография *Gapita Kāutudī* (“Наслаждение лотосом вычислений”), написанная Нааяной Пандита (*Nāgāyaṇa Paṇḍita*) в 1356 году, стала не так давно известна за пределами Индии благодаря переводу на английский язык Парманандом Сингхом (Parmanand Singh) [*Gapita Bhāratī*, 20 (1998), 25–82; 21 (1999), 10–73; 22 (2000), 19–85; 23 (2001) 18–82; 24 (2002) 35–98]; см. также диссертацию Таканори Кусуба (Takanori Kusuba) из Brown University (1993). Глава 13 книги Нааяны, озаглавленная *Anka Pāśa* (“Соединение чисел”), посвящена комбинаторной генерации. Хотя 97 “сутр” этой главы достаточно загадочны, они содержат полную теорию, опередившую мировую науку в этой области на несколько столетий.

Например, Нааяна описывает генерацию перестановок в сутрах 49–55а, где приводит алгоритм для перечисления всех перестановок множества в уменьшающемся солексном порядке, приводя при этом способ вычисления ранга данной перестановки и восстановления перестановки по рангу. Эти алгоритмы появились более чем за столетие до этого в известной работе Сарнгадевы (*Śāringadeva*) *Saṅgitaratnākara* (“Добыча жемчуга музыки”), § 1.4.60–71. Таким образом, Сарнгадева, по сути, открыл факториальное представление положительных чисел. В сутрах 57–60 Нааяна развел алгоритмы Сарнгадевы так, что можно было легко переставлять мульти множества общего вида; например, он перечисляет перестановки мульти множества $\{1, 1, 2, 4\}$ в обратном солексном порядке:

$$1124, 1214, 2114, 1142, 1412, 4112, 1241, 2141, 1421, 4121, 2411, 4211.$$

В сутрах 88–92 Нааяна занимается систематической генерацией сочетаний. Помимо иллюстрации 3-сочетаний множества $\{1, \dots, 8\}$, а именно

$$(678, 587, 478, \dots, 134, 124, 123),$$

он также рассматривает представление этих сочетаний в виде битовых строк в обратном порядке (*возрасташущий* солексный порядок), тем самым развив метод Бхаттотпалы (*Bhaṭṭotpala*), разработанный в X веке:

$$(11100000, 11010000, 10110000, \dots, 00010011, 00001011, 00000111).$$

Он почти — но не до конца — открыл теорему 7.2.1.3L.

Перестановочная поэзия. Обратимся теперь к одному занимательному вопросу, привлекавшему внимание ряда видных математиков в XVII веке; это прольет свет

на состояние комбинаторных знаний в Европе того времени. Иезуитский священник Бернард Баухуис (Bernard Bauhuis) сочинил одностороннюю хвалу Деве Марии в виде латинского гекзаметра:

$$\text{Tot tibi sunt dotes, Virgo, quot sidera cælo.} \quad (19)$$

[“У тебя столько добродетелей, Дево, как звезд на небе”; см. *Epigrammatum Libri V* (Cologne, 1615), 49.] Его стих побудил Эрикуса Путеануса (Erycius Puteanus), профессора университета в Лувене (University of Louvain), написать книгу *Pietatis Thau-mata* (Antwerp, 1617), в которой он представил 1022 перестановки слов Баухуиса. Например, у Путеануса имеются следующие варианты:

107	Tot dotes tibi, quot cælo sunt sidera, Virgo.	
270	Dotes tot, cælo sunt sidera quot, tibi Virgo.	
329	Dotes, cælo sunt quot sidera, Virgo tibi tot.	
384	Sidera quot cælo, tot sunt Virgo tibi dotes.	(20)
725	Quot cælo sunt sidera, tot Virgo tibi dotes.	
949	Sunt dotes Virgo, quot sidera, tot tibi cælo.	
1022	Sunt cælo tot Virgo tibi, quot sidera, dotes.	

Он остановился на 1022 перестановках, потому что 1022 — количество видимых звезд, перечисленных в знаменитом каталоге Птолемея (Ptolemy).

Идея такой перестановки слов была хорошо известна в то время; эту игру слов в своей книге *Poetices Libri Septem* (Lyon, 1561), Book 2, Chapter 30, Юлиус Скалигер (Julius Scaliger) назвал “Стихами-хамелеонами” (Proteus verses). Латинский язык приспособлен для перестановок наподобие (20), поскольку окончания латинских слов зачастую выражают грамматическое значение существительного, делая относительный порядок слов существенно менее важным для смысла предложения, чем, например, в английском языке. Путеанус указал, однако, что он специально избегал неподходящих перестановок, например

$$\text{Sidera tot cælo, Virgo, quot sunt tibi dotes,} \quad (21)$$

поскольку они указывают не нижнюю, а *верхнюю* границу количества достоинств Девы Марии. [См. с. 12 и 103 книги Путеануса.]

Существует всего $8! = 40\,320$ способов перестановки слов в строке (19). Но цель состоит не в том, чтобы получить их все — большинство из них “нечитаемо”. Каждый же из 1022 стихов Путеануса соответствует строгим правилам классического гекзаметра, которым следовали греческие и латинские поэты со времен Гомера и Вергилия:

- i) каждое слово состоит из длинных (—) или коротких (˘) слогов;
- ii) слоги каждой строки принадлежат одному из 32 шаблонов:

$$\left\{ \begin{array}{l} -\sim \\ -- \end{array} \right\} \left\{ \begin{array}{l} -\sim \\ -- \end{array} \right\} \left\{ \begin{array}{l} -\sim \\ -- \end{array} \right\} \left\{ \begin{array}{l} -\sim \\ -- \end{array} \right\} -\sim \left\{ \begin{array}{l} -\sim \\ -- \end{array} \right\} \quad (22)$$

Другими словами, имеется шесть метрических стоп, причем каждая из первых четырех может быть либо дактилем, либо спондеем в терминологии (5); пятая стопа должна быть дактилем, а последняя — трохеем или спондеем.

Правила для определения длинных и коротких слогов в латинской поэзии в общем случае достаточно запутаны, но восемь слов Баухуиса соответствуют следующим шаблонам:

$$\begin{aligned} \text{tot} = -, \text{tibi} = \left\{ \begin{array}{l} \sim \sim \\ \sim - \end{array} \right\}, \text{sunt} = -, \text{dotes} = --, \\ \text{Virgo} = \left\{ \begin{array}{l} - \sim \\ -- \end{array} \right\}, \text{quot} = -, \text{sidera} = - \sim \sim, \text{cælo} = --. \end{aligned} \quad (23)$$

Обратите внимание, что у поэтов есть по два варианта при использовании слов ‘tibi’ и ‘Virgo’. Таким образом, например, строка (19) соответствует шаблону гекзаметра

$$\begin{array}{cccccccccc} - & \sim & \sim & - & - & - & - & - & \sim & - & - \\ \text{Tot} & \text{ti} & \text{-bi} & \text{sunt} & \text{do} & \text{-tes, Vir-} & \text{go, quot} & \text{si-de-} & \text{ra} & \text{cæ-lo.} & \end{array} \quad (24)$$

(Дактиль, спондей, спондей, спондей, дактиль, спондей; “там-тата там-там там-там там-там там-тата там-там”. Запятые представляют небольшие паузы при чтении стиха, именуемые цезурами (cæsura); здесь они на рассматриваются, хотя Путеанус аккуратно вставил их в каждую из 1022 перестановок.)

Возникает естественный вопрос: если переставить слова Баухуиса случайным образом, каковы шансы на то, что они будут гекзаметром? Иными словами, сколько перестановок подчиняются правилам (i) и (ii) с учетом шаблонов из (23)? Г.В. Лейбниц (G.W. Leibniz) рассматривал этот вопрос среди других в своей работе *Dissertatio de Arte Combinatoria* (1666), опубликованной, когда он претендовал на место в университете Лейпцига (University of Leipzig). В это время Лейбничу было только 19 лет, и по большей части он был самоучкой, так что его понимание комбинаторики было весьма ограниченным; например, он считал, что имеется 600 перестановок множества {до, до, ре, ми, фа, соль} и 480 — множества {до, до, ре, ми, фа}, и даже утверждал, что (22) представляет 76 вариантов, а не 32 [см. §§ 5 и 8 в его задаче 6].

Однако Лейбниц понимал, что следовало бы разработать общие методы для подсчета всех “подходящих” перестановок в ситуациях, когда многие подстановки таковыми не являются. Он рассмотрел несколько примеров стихов-хамелеонов, корректно проведя подсчеты для более простых и наделав массу ошибок там, где слова оказывались сложнее. Хотя Лейбниц и упомянул работу Путеануса, он не пытался подсчитать количество гекзаметров среди перестановок строки (19).

Существенно более успешный подход был предложен несколько лет спустя Жаном Престетом (Jean Prestet) в работе *Éléments des Mathématiques* (Paris, 1675), 342–438. Престет привел рассмотрение данного вопроса, из которого вытекало наличие ровно 2196 перестановок слов хвалы Баухуиса, являющихся гекзаметрами. Однако вскоре он обнаружил, что упустил некоторое количество случаев, в частности случаи 270, 384, 725 из (20). Поэтому он полностью переписал данный материал при переиздании книги *Nouveaux Éléments des Mathématiques* в 1689 году. На с. 127–133 новой книги Престет показывает, что правильное количество перестановок-гекзаметров равно 3276, что почти на 50% превышает ранее указанное им число.

Тем временем Джон Уоллис (John Wallis) рассмотрел данную задачу в работе *Discourse of Combinations* (London, 1685), 118–119, опубликованной в качестве дополнения к его книге *Treatise of Algebra*. Объяснив, почему он считает, что корректное количество равно 3096, Уоллис допустил предположение, что он мог упустить некоторые варианты и/или сосчитать другие более чем по одному разу; “однако в настоящее время я не заметил за собой ни того, ни другого”.

Анонимный критик работы Уоллиса указал, что верное количество перестановок составляет 2580, но не дал никакого доказательства этому факту [*Acta Eruditorum*, 5 (1686), 289]. Этим критиком почти наверняка был Лейбниц, хотя никакого ключа к пониманию числа 2580 среди его многочисленных неопубликованных заметок найдено не было.

Наконец, на сцене появляется Я. Бернулли, в инаугурационной лекции которого при вступлении в должность декана факультета философии университета Базеля (University of Basel) в 1692 году была упомянута данная задача и сказано, что аккуратный анализ дал корректный ответ, равный 3312(!) перестановкам. Доказательство Бернулли было опубликовано посмертно в первом издании *Ars Conjectandi* (1713), 79–81. [Эти страницы, кстати, были опущены в более поздних изданиях этой знаменитой книги и в сборниках работ Бернулли, поскольку изначально он не предназначал их для публикации; редактор внес их в книгу по ошибке. См. *Die Werke von Jakob Bernoulli*, 3 (Basel : Birkhauser, 1975), 78, 98–106, 108, 154–155.]

Так кто же был прав? Сколько же гекзаметров среди перестановок — 2196, 3276, 3096, 2580 или 3312? В.А. Уитворт (W.A. Whitworth) и В.Э. Хартли (W.E. Hartley) заново рассмотрели этот вопрос в *The Mathematical Gazette*, 2 (1902), 227–228, где каждый из них представил элегантные доказательства и сделал выводы, что ни одно из предложенных чисел не является корректным ответом. Их общим решением было 2880, и это был первый случай, когда два математика независимо получили одно и то же решение данной задачи. Но из упражнений 21 и 22 вы узнаете истину: прав был только Бернулли, все остальные ошибались. Кроме того, изучение трехстраничного вывода Бернулли указывает, что он был успешен в основном потому, что автор строго придерживался метода, который сейчас называется *методом с возвратом* (backtrack method). Мы тщательно изучим этот метод в разделе 7.2.2, где также узнаем, что рассматриваемая здесь задача легко решается как частный случай *задачи точного покрытия* (exact cover problem).

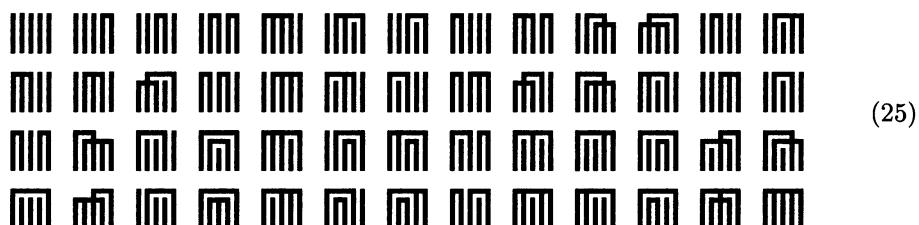
*Даже мудрейшие и осторожнейшие люди часто страдают от того,
что логики называют неполным перечислением возможностей.*
— Якоб Бернулли (James Bernoulli) (1692)

Разбиения множества. Похоже, впервые разбиения множеств изучались в Японии, где в конце XV — начале XVI веков среди высших классов стала популярна игра *генджи-ко* (genji-ko). Ведущий должен скрытно выбрать пять пакетов фимиама, причем некоторые из них могут быть одинаковы, и по очереди воскурить их. Игроки должны попытаться определить, какие запахи были одинаковы, а какие различны — словом, угадать, какое из $\varpi_5 = 52$ разбиений множества $\{1, 2, 3, 4, 5\}$ выбрано ведущим.



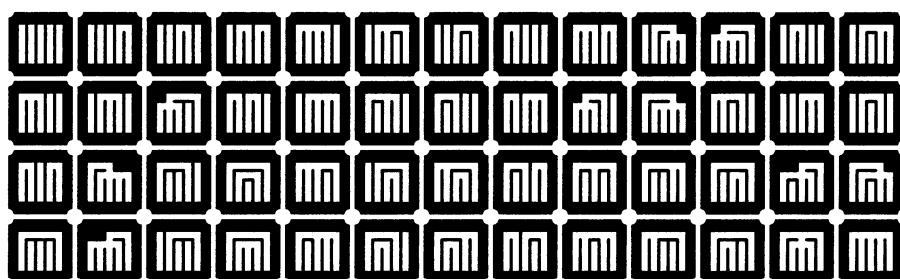
Рис. 47. Диаграммы, использовавшиеся для представления разбиений множества в XVI веке в Японии [копия из коллекции Тамаки Яно (Tamaki Yano) из университета Сайтамы]

Вскоре стало привычным изображать 52 возможных варианта с помощью диаграмм, подобных приведенным на рис. 47. Например, верхняя диаграмма при чтении справа налево указывает, что одинаковы два первых запаха и три последних, т.е. это разбиение $12 | 345$. Две другие диаграммы представляют соответственно разбиения $124 | 35$ и $1 | 24 | 35$. Чтобы помочь запоминанию, каждый из 52 шаблонов получил свое название благодаря знаменитой книге XI века г-жи Мурасаки (Murasaki) *Рассказ о генджи*, в соответствии с приведенной ниже последовательностью [Encyclopedia Japonicae (Tokyo : Sanseido, 1910), 1299].



(Как и во множестве других примеров, варианты здесь перечислены без какого-то логичного порядка.)

Привлекательная природа шаблонов генджи-ко привела к тому, что многие семьи использовали их в качестве геральдических символов. Например, ниже приведены стилизованные версии (25), которые были найдены в стандартном каталоге узоров кимоно начала XX века:



[См. Fumie Adachi, *Japanese Design Motifs* (New York : Dover, 1972), 150–153.]

В начале XVIII века Такаказу Секи (Takakazu Seki) со своими учениками приступил к изучению количества разбиений множества ϖ_n для произвольного n , вдохновленный известным результатом $\varpi_5 = 52$. Ёшицуке Мацунага (Yoshisuke Matsunaga) нашел формулы для количества разбиений множества, когда имеется k_j подмножеств размером n_j для $1 \leq j \leq t$, и $k_1 n_1 + \cdots + k_t n_t = n$ (см. ответ к упражнению 1.2.5–21). Он также открыл базовое рекуррентное соотношение 7.2.1.5–(14), а именно

$$\varpi_{n+1} = \binom{n}{0} \varpi_n + \binom{n}{1} \varpi_{n-1} + \binom{n}{2} \varpi_{n-2} + \cdots + \binom{n}{n} \varpi_0, \quad (26)$$

при помощи которого легко вычислить значения ϖ_n .

Открытия Мацунаги остались неопубликованными до выхода в 1769 году книги Ёриюки Аrimы (Yoriyuki Arima) *Shūki Sanpō*. Задача 56 из этой книги состоит в решении уравнения “ $\varpi_n = 678\,570$ ” относительно n . Детальное решение Аrimы (с надлежащими благодарностями Мацунаге) дает ответ $n = 11$.

Вскоре после этого Масанобу Сака (Masanobu Saka) изучал количество $\{ \binom{n}{k} \}$ способов разбиения множества из n элементов на k подмножеств в своей работе *Sanpo-Gakkai* (1782). Он открыл рекуррентную формулу

$$\left\{ \begin{matrix} n+1 \\ k \end{matrix} \right\} = k \left\{ \begin{matrix} n \\ k \end{matrix} \right\} + \left\{ \begin{matrix} n \\ k-1 \end{matrix} \right\} \quad (27)$$

и протабулировал результаты для $n \leq 11$. Джеймс Стирлинг (James Stirling) в своей работе *Methodus Differentialis* (1730) открыл числа $\{ \binom{n}{k} \}$ в чисто алгебраическом контексте; таким образом, Сака был первым, кто осознал их комбинаторную важность.

Интересный алгоритм для перечисления всех разбиений множества был впоследствии открыт Тошиаки Хондой (Toshiaki Honda) (см. упражнение 24). Более детальную информацию о генджи-ко и ее связи с историей математики можно найти в статьях Tamaki Yano, *Sugaku Seminar*, 34, 11 (Nov. 1995), 58–61; 34, 12 (Dec. 1995), 56–60.

Разбиения множеств оставались практически не известными в Европе до гораздо более позднего времени, за исключением трех не связанных между собой случаев. Во-первых, в 1589 году Георг и/или Ричард Паттенхам (Georg and/or Richard Puttenham) опубликовал книгу *The Arte of English Poesie*, на с. 70–72 которой содержатся диаграммы, похожие на диаграммы генджи-ко. Например, приведенные ниже семь диаграмм использованы для иллюстрации возможных схем рифм в пятистрочных стихах, “тде некоторые из них грубее и неприятнее для уха, чем иные”. Но этот визуально привлекательный список был неполон (см. упражнение 25).



Во-вторых, неопубликованная рукопись Г.В. Лейбница (G.W. Leibniz) конца XVII века свидетельствует, что он пытался вычислить количество способов разбиения множества $\{1, \dots, n\}$ на три или четыре подмножества, но практически безуспешно. Он посчитал $\{ \binom{n}{2} \}$ очень громоздким методом, который не позволил ему легко заметить, что $\{ \binom{n}{2} \} = 2^{n-1} - 1$. Лейбниц попытался вычислить $\{ \binom{n}{3} \}$ и $\{ \binom{n}{4} \}$

только для $n \leq 5$ и сделал несколько числовых ошибок, что привело его к неверному ответу. [См. E. Knobloch, *Studia Leibnitiana Supplementa*, 11 (1973), 229–233; 16 (1976), 316–321.]

Третий европейский случай разбиений множества носит совершенно иной характер. Джон Уоллис (John Wallis) посвятил третью главу своей книги *Discourse of Combinations* (1685) вопросу о “кратных частях”, истинных делителях чисел, в частности он изучал множество всех способов разложения данного числа на множители. Этот вопрос эквивалентен задаче о разбиениях *мультимножества*; например, разложение p^3q^2r по сути то же, что и разбиения мультимножества $\{p, p, p, q, q, r\}$, где p , q и r — простые числа. Уоллис разработал превосходный алгоритм для перечисления всех разложений данного целого числа n , по сути предвосхитив алгоритм 7.2.1.5M (см. упражнение 28). Однако он не исследовал важные частные случаи, когда n представляет собой степень простого числа (эквивалентно разбиениям целого числа) или когда n не содержит квадратов (эквивалентно разбиениям множества). Таким образом, хотя Уоллис и мог решить более общую задачу, ее сложность парадоксальным образом увела его с пути открытия разбиений чисел, чисел Белла, количества подмножеств Стирлинга и разработки простых алгоритмов генерации разбиений целых чисел или разбиений множеств.

Разбиения целых чисел. Разбиения целых чисел выходили на сцену комбинаторики еще медленнее. Епископ Вибольд (Wibold) (около 965 года) знал о разбиениях целых чисел на три части, не превышающие 6. О том же знал и Галилей (Galileo), который написал о них небольшую работу (около 1627 года) и который изучал частоты выпадения очков при бросании трех игральных костей. [*Sopra le scoperte de i dadi*, в его работе (*Opere*, Vol. 8, 591–594) Галилей перечислил разбиения в уменьшающемся лексикографическом порядке.]

Мерсенн (Mersenne) перечислил разбиения 9 на любое количество частей на с. 130 своей книги *Traitez de la Voix et des Chants* (1636). Для каждого разбиения $9 = a_1 + \dots + a_k$ он, кроме того, вычислил мультиномиальный коэффициент $9!/(a_1! \dots a_k!)$; как мы видели ранее, его интересовало количество различных мелодий, например он знал, что имеется $9!/(3!3!3!) = 1680$ мелодий из девяти нот $\{a, a, a, b, b, b, c, c, c\}$. Однако он не упомянул случаи $8 + 1$ и $3 + 2 + 1 + 1 + 1 + 1$, вероятно потому, что не перечислял возможности каким-либо систематическим путем.

Лейбниц (Leibniz) рассматривал разбиения на две части в задаче 3 в работе *Dissertatio de Arte Combinatoria* (1666), а его неопубликованные заметки указывают, что впоследствии он потратил немало времени, пытаясь перечислить разбиения, состоящие из трех и более слагаемых. Он называл их (разумеется, на латыни) “discerpitions” или реже “divulsions”, а иногда — разделами (“sections”), рассеянием (“dispersions”) или даже разбиениями (“partitions”). Они интересовали его в первую очередь из-за их связи с мономиальными симметричными функциями $\sum x_{i_1}^{a_1} x_{i_2}^{a_2} \dots$. Однако множество попыток Лейбница почти всегда приводили к неудаче, за исключением случая трех слагаемых, когда он почти (но не совсем) открыл формулу для $| \frac{n}{3} |$ (см. упражнение 7.2.1.4–31). Например, он аккуратно сосчитал только 21 разбиение 8, упустив случай $2 + 2 + 2 + 1 + 1$, а для $p(9)$ он получил только 26, потеряв $3 + 2 + 2 + 2$, $3 + 2 + 2 + 1 + 1$, $2 + 2 + 2 + 1 + 1 + 1$ и $2 + 2 + 1 + 1 + 1 + 1 + 1$, несмотря на то что он пытался перечислять разбиения систематически в убывающем лексико-

графическом порядке. [См. E. Knobloch, *Studia Leibnitiana Supplementa*, **11** (1973), 91–258; **16** (1976), 255–337; *Historia Mathematica*, **1** (1974), 409–430.]

Де Муавр (de Moivre) был первым, реально достигшим успеха при изучении разбиений в своей статье “Метод возведения бесконечного многочлена в любую заданную степень, или извлечение из него заданного корня” [*Philosophical Transactions*, **19** (1697), 619–625 и рис. 5]. Он доказал, что коэффициент при z^{m+n} в $(az + bz^2 + cz^3 + \dots)^m$ имеет по одному члену для каждого разбиения n ; например, коэффициент при z^{m+6} равен

$$\begin{aligned} & \binom{m}{6} a^{m-6} b^6 + 5 \binom{m}{5} a^{m-5} b^4 c + 4 \binom{m}{4} a^{m-4} b^3 d + 6 \binom{m}{4} a^{m-4} b^2 c^2 + \\ & + 3 \binom{m}{3} a^{m-3} b^2 e + 6 \binom{m}{3} a^{m-3} bcd + 2 \binom{m}{2} a^{m-2} bf + \\ & + \binom{m}{3} a^{m-3} c^3 + 2 \binom{m}{2} a^{m-2} ce + \binom{m}{2} a^{m-2} d^2 + \binom{m}{1} a^{m-1} g. \end{aligned} \quad (29)$$

Если мы положим $a = 1$, то член со степенями $b^i c^j d^k e^l \dots$ соответствует разбиению с i единицами, j двойками, k тройками и т.д. Так, например, при $n = 6$ он получил разбиения в порядке

$$111111, 11112, 1113, 1122, 114, 123, 15, 222, 24, 33, 6. \quad (30)$$

Он пояснил, как перечислить разбиения рекурсивно (хотя и другим языком, связанным с его собственными обозначениями): для $k = 1, 2, \dots, n$ начинаем с k и добавляем (ранее перечисленные) разбиения $n - k$, наименьшая часть которых не меньше k .

*[Мое решение] упорядочено перед публикацией,
не столько для красоты,
сколько в процессе некоторых размышлений,
не достойных внимания любителей истины.
Абрахам де Муавр (Abraham de Moivre) (1717)*

П.Р. де Монморт (P.R. de Montmort) протабулировал все разбиения чисел ≤ 9 на количество частей ≤ 6 в своем труде *Essay d'Analyse sur les Jeux de Hazard* (1708) в связи с задачей об игральных костях. Его разбиения перечислены в ином, чем в (30), порядке, например:

$$111111, 21111, 2211, 222, 3111, 321, 33, 411, 42, 51, 6. \quad (31)$$

По-видимому, Монморт не был знаком с работой де Муавра.

До сих пор практически никто из рассмотренных нами авторов не описал используя им процедуру генерации комбинаторных шаблонов. Мы можем только догадываться об их методах (или их отсутствии), изучая опубликованные ими списки. Более того, в таких редких случаях, как статья де Муавра, в которой имеется явно описанный метод, автор полагает, что существуют списки всех шаблонов для случаев $1, 2, \dots, n - 1$, перед тем как приступить к созданию списка для случая n . За исключением Кедары (Kedara) и Нараяны (Nāgāyaṇa), ни один из авторов не привел метода генерации шаблонов “на лету”, когда очередной шаблон получается из предшествующего непосредственно, без просмотра вспомогательных таблиц.

Естественно, что современные программисты предпочитают более прямые методы, требующие меньшего количества памяти.

Р.Й. Башкович (R.J. Boscovich) опубликовал первый прямой алгоритм для генерации разбиений в *Giornale de' Letterati* (Rome, 1747), с. 393–404, вместе с двумя таблицами (с. 404). Его метод, который для $n = 6$ дает выход

$$111111, 11112, 1122, 222, 1113, 123, 33, 114, 24, 15, 6, \quad (32)$$

генерирует разбиения в точности в обратном порядке по отношению к порядку, получаемому при работе алгоритма 7.2.1.4Р. Метод Башковича, по сути, описан в разделе 7.2.1.4, с тем исключением, что обратный порядок приводит к несколько более простому и быстрому алгоритму, чем порядок, выбранный Башковичем.

Башкович опубликовал продолжение своей работы в *Giornale de' Letterati* (Rome, 1748, с. 12–27 и 84–99), развив свой алгоритм в двух направлениях. Во-первых, он рассмотрел генерацию только тех разбиений, части которых принадлежат заданному множеству S , с тем чтобы возводить в t -ю степень символные полиномы с разреженными коэффициентами. (Он утверждал, что наибольший общий делитель всех элементов S должен быть равен 1; в действительности, однако, его метод не работает, если $1 \notin S$.) Во-вторых, он разработал алгоритм для генерации разбиений n на t частей для заданных n и t . И опять ему не повезло: для решения этой задачи впоследствии был разработан более удачный алгоритм 7.2.1.4Н, что лишило Башковича шансов на славу.

Ода Гинденбургу. Изобретателем алгоритма 7.2.1.4Н был Карл Фридрих Гинденбург (Carl Friedrich Hindenburg), который “переоткрыл” алгоритм Нараяны 7.2.1.2L — способ генерации перестановок мульти множества. К сожалению, этот небольшой успех привел его к вере в то, что он осуществил революционный прорыв в математике (хотя он и снизошел до того, что упомянул других исследователей, в частности де Муавра, Эйлера и Ламберта, которые близко подошли к аналогичным открытиям).

Гинденбург был в числе основателей первых математических журналов Германии (издававшихся в 1786–1789 и 1794–1800 годах) и автором статей в них. Он неоднократно занимал должность декана в университете Лейпцига (University of Leipzig), а в 1792 году был его ректором. Если бы он имел немного больше способностей к математике по сравнению с тем, чем был наделен на самом деле, германская математика процветала бы именно в Лейпциге, а не в Берлине и Геттингене. Однако первая же математическая работа Гинденбурга, *Beschreibung einer ganz neuen Art, nach einem bekannten Gesetze fortgehende Zahlen durch Abzählen oder Abmessen bequem und sicher zu finden* (Leipzig, 1776), достаточно ярко показала, чего следует ожидать: его “ganz neue” (совершенно новая) идея заключалась в придании комбинаторного смысла цифрам десятичной записи чисел. Невероятно, но Гинденбург завершил свою монографию большими таблицами — таблицей чисел от 0000 до 9999, после которой следовали таблицы четных и нечетных чисел в отдельности (!).

Гинденбург публиковал письма людей, хваливших его работу, и приглашал их писать статьи в его журналы. В 1796 году он редактировал *Sammlung combinatorisch-analytischer Abhandlungen*, где в подзаголовке было указано, что теорема о полиномах де Муавра “является наиболее важной во всем математическом анализе”. Около десятка людей объединили свои усилия и образовали то, что впоследствии

стало известно как “школа комбинаторики Гинденбурга”, и опубликовали тысячи страниц, заполненных эзотерическим символизмом, поражавшим воображение множества людей, не имеющих отношения к математике.

С точки зрения информатики работа этой школы не может считаться совершен но тривиальной. Например, лучший студент Гинденбурга, Г.А. Роте (H.A. Rothe), заметил, что имеется простой способ перейти от последовательности кода Морзе к следующей за ней в лексикографическом порядке или к предшествующей. Другой ученик, И.К. Буркхардт (J.C. Burkhardt), заметил, что последовательности кодов Морзе длиной n могут быть легко сгенерированы, если сначала рассмотреть коды без тире, затем с одним тире, далее с двумя и т.д. Их целью была не табуляция стихотворных форм с n тактами, как это делалось в Индии, а перечисление членов континуантов $K(x_1, x_2, \dots, x_n)$, формула 4.5.3–(4). [См. *Archiv für reine und angewandte Mathematik*, 1 (1794), 154–194.] Кроме того, на с. 53 упоминавшегося выше *Sammlung* Г.С. Клюгель (G.S. Klugel) привел способ перечисления всех перестановок, который впоследствии получил название алгоритма Орд-Смита (Ord-Smith); см. формулы (23)–(26) в разделе 7.2.1.2.

Гинденбург верил, что его методы заслуживают достойного места в учебных курсах алгебры, геометрии и вычислений. Однако и он, и его ученики ограничивались в своих работах составлением списков комбинаторных объектов. Закопавшись в своих формулах и формализме, им редко удавалось открыть что-то действительно интересное с точки зрения математики. Эжен Нетто (Eugen Netto) замечательно охарактеризовал их работу в *Geschichte der Mathematik* М. Кантора (M. Cantor), 4 (1908), 201–219: “Какое-то время они контролировали германский рынок, но большинство из того, что они откопали, тут же засыпало песком забвения”.

Грустным итогом их исследований стало то, что в результате комбинаторика в целом стала восприниматься как лженаука. Геста Миттаг-Леффлер (Gosta Mittag-Leffler), собравший великолепную библиотеку математической литературы примерно через 100 лет после смерти Гинденбурга, решил поместить все такие работы на отдельной полке, помеченной “Декаденс”. Эта категория осталась в шведском институте Миттаг-Леффлера и сегодня, несмотря на то что этот институт привлекает со всего мира математиков, работающих в области комбинаторики, труды которых можно назвать как угодно, но никак не упадничеством.

Во всем есть хорошая сторона, и мы можем отметить, как минимум, одну хорошую книгу, появившуюся в результате всей этой деятельности. Это *Die combinatorische Analysis* (Vienna, 1826) Андреаса фон Эттингсхаузена (Andreas von Ettinghausen), заслуживающая внимания как первая книга, в которой методы генерации комбинаторных объектов рассматриваются ясно и последовательно. Эттингсхаузен рассмотрел общие принципы лексикографической генерации в § 8 и применил их для построения метода перечисления всех перестановок (§ 11), сочетаний (§ 30) и разбиений (§ 41–44).

А где же деревья? Итак, мы уже видели, что на протяжении истории человечества неоднократно создавались списки кортежей, перестановок, сочетаний и разбиений. Таким образом, мы рассмотрели эволюцию тем в разделах 7.2.1.1–7.2.1.5, но наш рассказ будет неполон, если мы не проследим происхождение генерации деревьев — тему раздела 7.2.1.6.

Однако исторические сведения по этой теме до прихода компьютеров практически отсутствуют, если не считать нескольких статей, опубликованных в XIX веке Артуром Кейли (Arthur Cayley). Основная работа Кейли по деревьям была опубликована в 1875 году и перепечатана в его *Collected Mathematical Papers* (Vol. 4, 427–460); она содержала большую иллюстрацию со всеми свободными деревьями не более чем с девятью непомеченными вершинами. Ранее в этой статье он также привел девять *ориентированных* деревьев с пятью вершинами. Методы, использовавшиеся Кейли для получения этих списков, были весьма сложными, совершенно отличавшимися от алгоритма 7.2.1.6О и упр. 7.2.1.6–90. Все свободные деревья не более чем с десятью вершинами были перечислены много лет спустя Ф. Харари (F. Harary) и Д. Принсем (G. Prins) [Acta Math., **101** (1958), 158–162], которые дошли до $n = 12$ для случаев свободных деревьев, не имеющих узлов степени 2 и не обладающих симметрией.

Деревья, наиболее любимые кибернетиками, — бинарные деревья, или эквивалентные упорядоченные леса, или вложенные скобки — как ни странно, в литературе отсутствуют. В разделе 2.3.4.5 мы видели, что многие математики на протяжении 1700–1800-х годов изучали количество бинарных деревьев; мы также знаем, что числа Каталана C_n подсчитывают десятки различных комбинаторных объектов. Однако до 1950 года, похоже, никто не публиковал список из $C_4 = 14$ объектов порядка 4 *любого* вида, и еще меньше авторов публиковали список из $C_5 = 42$ объектов порядка 5. (За косвенным исключением — 42 диаграммы генджи-ко из (25), не имеющие пересекающихся линий, оказываются эквивалентны 5-узловым бинарным деревьям и лесам. Однако этот факт не изучался до XX века.)

Имеется несколько отдельных примеров, когда в былые дни авторы составляли списки из $C_3 = 5$ объектов, связанных с числами Каталана. Здесь также первым был Кейли; он привел бинарные деревья с тремя внутренними узлами и четырьмя листьями [Philosophical Magazine, **18** (1859), 374–378].



(В той же статье проиллюстрированы другие разновидности деревьев, эквивалентные так называемым слабым упорядочениям (weak orderings).) В 1901 году Нетто перечислил пять способов расстановки скобок в выражении ' $a + b + c + d$:

$$(a+b)+(c+d), [(a+b)+c]+d, [a+(b+c)]+d, a+[(b+c)+d], a+[b+(c+d)] \quad (34)$$

[Lehrbuch der Combinatorik, § 122.] Пять перестановок множества $\{+1, +1, +1, -1, -1\}$, частичные суммы которых неотрицательны, перечислены П. Эрдешем (P. Erdős) и Ирвингом Каплански (Irving Kaplansky) [Scripta Math., **12** (1946), 73–75]:

$$\begin{aligned} 1 + 1 + 1 - 1 - 1, & 1 + 1 - 1 + 1 - 1 - 1, 1 + 1 - 1 - 1 + 1 - 1, \\ 1 - 1 + 1 + 1 - 1 - 1, & 1 - 1 + 1 - 1 + 1 - 1. \end{aligned} \quad (35)$$

Несмотря на то что в приведенных примерах используется только пять объектов, мы видим, что упорядочение в (33) и (34) выполнено “как получится”; и только упорядочение (35), соответствующее алгоритму 7.2.1.6Р, систематическое и лексикографическое.

Мы также должны вкратце отметить работу Вальтера фон Дика (Walther von Dyck), поскольку многие современные статьи используют термин “Слова Дика” (“Dyck words”), говоря о строках вложенных скобок. Дик был педагогом, известным, помимо прочего, как сооснователь Немецкого музея в Мюнхене (Deutsches Museum). Он также написал две пионерские статьи по теории свободных групп [*Math. Annalen*, **20** (1882), 1–44; **22** (1883), 70–108]. Так называемые слова Дика имеют очень слабое отношение к его реальным исследованиям. Он изучал слова на алфавите $\{x_1, x_1^{-1}, \dots, x_k, x_k^{-1}\}$, которые приводятся к пустой строке после многократного удаления пар соседних букв вида $x_i x_i^{-1}$ или $x_i^{-1} x_i$; связь со скобками и деревьями возникает, только если ограничиться первым случаем, $x_i x_i^{-1}$.

Итак, мы можем заключить, что, несмотря на огромный рост интереса к бинарным деревьям и их “родственникам” после 1950 года, такие деревья — единственный предмет нашего рассмотрения, исторические корни которого крайне неглубоки.

После 1950. Разумеется, выход на сцену электронных вычислительных машин резко все изменил. Первой ориентированной на применение компьютеров публикацией о методах комбинаторной генерации стала статья Ч.Б. Томпкинса (C.B. Tompkins) “Machine attacks on problems whose variables are permutations” (Машинное решение задач, переменными которых являются перестановки) [*Proc. Symp. Applied Math.*, **6** (1956), 202–205]. За ней не замедлили последовать тысячи других.

Ряд статей Д.Г. Лемера (D.H. Lehmer), особенно “Teaching combinatorial tricks to a computer” (Обучение компьютера комбинаторным трюкам) [*Proc. Symp. Applied Math.*, **10** (1960), 179–193], оказали исключительно большое влияние на исследования в то время. [См. также *Proc. 1957 Canadian Math. Congress* (1959), 160–173; *Proc. IBM Scientific Computing Symposium on Combinatorial Problems* (1964), 23–30; а также главу 1 книги *Applied Combinatorial Mathematics* под ред. E.F. Beckenbach (Wiley, 1964), 5–31.] Лемер оказался важным связующим звеном с предыдущими поколениями. Так, например, записи в Стенфордской библиотеке свидетельствуют, что в январе 1932 года он изучал *Lehrbuch der Combinatorik* Э. Нетто.

Основные публикации, посвященные рассматривавшимся нами конкретным алгоритмам, упоминались в предыдущих разделах, так что повторять их здесь нет необходимости. Однако следует упомянуть три книги, которые заслуживают особого внимания, поскольку именно в них были заложены общие принципы.

- *Elements of Combinatorial Computing* Марка Б. Уэллса (Mark B. Wells) (Pergamon Press, 1971), особенно глава 5.
- *Combinatorial Algorithms* Альберта Ньенхuisа (Albert Nijenhuis) и Герберта С. Вильфа (Herbert S. Wilf) (Academic Press, 1975). Второе издание книги вышло в 1978 году и содержало дополнительный материал; впоследствии Вильф написал *Combinatorial Algorithms: An Update* (Philadelphia : SIAM, 1989).
- *Combinatorial Algorithms: Theory and Practice* Эдварда М. Рейнгольда (Edward M. Reingold), Юрга Нивергельта (Jurg Nievergelt) и Нарсингха Део (Narsingh Deo) (Prentice-Hall, 1977), особенно материал главы 5.

Роберт Седжвик (Robert Sedgewick) опубликовал первый серьезный обзор по методам генерации перестановок в *Computing Surveys*, **9** (1977), 137–164, 314. Второй

вехой стал обзор Карлы Сэведж (Carla Savage), посвященный кодам Грея [SIAM Review, **39** (1997), 605–629].

Ранее мы уже отмечали, что алгоритмы для генерации объектов, подсчитывающих при помощи чисел Каталана, не были созданы до тех пор, пока разработчики программного обеспечения не ощутили в них потребность. Первые опубликованные алгоритмы не цитировались в разделе 7.2.1.6, поскольку были вытеснены более эффективными методами, однако здесь самое подходящее место упомянуть о них. Во-первых, Г.Я. Скоинс (H.I. Scoins) привел два рекурсивных алгоритма для генерации упорядоченных деревьев в той же статье, которую мы уже цитировали, говоря о генерации *ориентированных* деревьев [Machine Intelligence, **3** (1968), 43–60]. Его алгоритмы работают с представлением бинарных деревьев в виде битовых строк, что, по сути, эквивалентно польской префиксной записи или вложенным скобкам. Затем Марк Уэллс (Mark Wells) в разделе 5.5.4 упоминавшейся выше книги генерировал бинарные деревья, представляя их в виде непересекающихся разбиений множеств. Наконец, Гари Кнотт (Gary Knott) [CACM, **20** (1977), 113–115] разработал рекурсивные алгоритмы для определения ранга бинарного дерева и поиска бинарного дерева по заданному рангу, представляя деревья при помощи перестановок $q_1 \dots q_n$ из табл. 7.2.1.6–3.

Алгоритмы для генерации остовых деревьев заданного графа были опубликованы рядом авторов еще в 1950-е годы; толчок к таким алгоритмам дало изучение электрических сетей. Вот некоторые из самых ранних работ в этом направлении: N. Nakagawa, IRE Trans., CT-5 (1958), 122–127; W. Mayeda, IRE Trans., CT-6 (1959), 136–137, 394; H. Watanabe, IRE Trans., CT-7 (1960), 296–302; S. Hakimi, J. Franklin Institute, **272** (1961), 347–359.

В главах 2 и 3 книги Donald L. Kreher and Douglas R. Stinson *Combinatorial Algorithms: Generation, Enumeration, and Search* (CRC Press, 1999), можно найти полезную информацию по всей рассматриваемой теме.

Франк Раски (Frank Ruskey) подготовил книгу *Combinatorial Generation*, которая будет содержать всестороннее рассмотрение комбинаторных вопросов и исчерпывающую библиографию по данной теме. Черновики некоторых глав из этой книги можно найти в Интернете.

Упражнения

Во многих из предлагаемых упражнений от современного читателя требуется найти и/или исправить ошибки в литературе давно минувших дней. И цель отнюдь не в том, чтобы, злорадно ухмыляясь, показать, насколько мы в XXI веке умнее, а в том, чтобы понять, что даже пионеры в некоторой области могут остаться. Полное понимание того, что множество идей не столь просты, как может показаться сегодняшнему программисту и математику, придет к вам лишь тогда, когда вы обнаружите, как много сил вынуждены были тратить ученые мирового уровня на то, чтобы разобраться с этими новыми концепциями.

1. [15] Существует ли понятие “вычислительная техника” в *I Ching*?
- 2. [M30] (Генетический код.) Молекулы ДНК представляют собой строки “нуклеотидов” из четырехбуквенного алфавита {T, C, A, G}, а большинство молекул белков представляют собой строки “аминокислот” из 20-буквенного алфавита {A, C, D, E, F,

$G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$. Три последовательных нуклеотида образуют базовый блок — “кодон”, а цепочка $x_1y_1z_1x_2y_2z_2\dots$ ДНК определяет белок $f(x_1y_1z_1)f(x_2y_2z_2)\dots$, где $f(x, y, z)$ — элемент в строке z и столбце y матрицы x в массиве

$$\begin{pmatrix} F & S & Y & C \\ F & S & Y & C \\ L & S & - & - \\ L & S & - & W \end{pmatrix} \quad \begin{pmatrix} L & P & H & R \\ L & P & H & R \\ L & P & Q & R \\ L & P & Q & R \end{pmatrix} \quad \begin{pmatrix} I & T & N & S \\ I & T & N & S \\ I & T & K & R \\ M & T & K & R \end{pmatrix} \quad \begin{pmatrix} V & A & D & G \\ V & A & D & G \\ V & A & E & G \\ V & A & E & G \end{pmatrix}.$$

(Здесь $(T, C, A, G) = (1, 2, 3, 4)$; например, $f(\text{CAT})$ представляет собой элемент в строке 1 и столбце 3 матрицы 2, т.е. Н.) Кодирование выполняется до тех пор, пока кодон не приведет к останавливающему элементу ‘-’.

a) Покажите, что существует простой способ сопоставить каждый кодон гексаграмме из *I Ching*, причем 21 возможный выход $\{A, C, D, \dots, W, Y, -\}$ соответствует 21 последовательной гексаграмме в упорядочении (1) Кинг Веня (King Wen).

b) Является ли это открытие сенсационным?

3. [20] Какой вид имеет миллионная строфа из 30 тактов в солексном порядке, аналогичном (2)? Каков ранг $\overbrace{\dots}^{30}$?

4. [19] Проанализируйте недочеты списка перестановок Донноло (Donnolo) в табл. 1

5. [16] Где ошибка в списке перестановок пяти нот Кирхера (Kircher) в (7)?

6. [25] В своей книге *Traitez de la Voix et des Chants* (1636) Мерсенн (Mersenne) опубликовал таблицу первых 64 факториалов на с. 108–110. Его значение для $64!$ приближенно равно $\approx 2.2 \times 10^{89}$, хотя на самом деле оно должно быть равно $\approx 1.3 \times 10^{89}$. Найдите копию этой книги и выясните, где он ошибся.

7. [20] Какие перестановки множества $\{1, 2, 3, 4, 5\}$ “живы”, а какие “мертвы” в соответствии с правилами Секи (Seki) (8) и (9)?

► 8. [M27] Придумайте исправление, которое нужно внести в правило (9), чтобы процедура Секи стала корректной.

9. [15] Получите из (11) арабский способ написания арабских цифр $(0, 1, \dots, 9)$.

► 10. [HM27] Каково математическое ожидание количества бросков костей в игре *Ludus Clericalis* для получения всех возможных добродетелей?

11. [21] Расшифруйте вертикальную таблицу Луллия на рис. 45 справа. Какие 20 комбинаторных объектов она представляет? Указание: не позволяйте опечаткам ввести вас в заблуждение.

12. [M20] Свяжите универсальный цикл Шиллингера (Schillinger) (13) с универсальным циклом Пуансо (Poinset) из упражнения 7.2.1.3–106.

13. [21] Что ван Шутен (van Schooten) должен был написать вместо (17)? Приведите также соответствующую таблицу для сочетаний мультимножества $\{a, a, a, b, b, c\}$.

► **14.** [20] Завершите последовательность из § 95 *De Combinatione* Изкуэрдо (Izquierdo):

ABC ABD ABE ACD ACE ACB ADE ADB ADC AEB

15. [15] Если все n -сочетания с повторениями $x_1 \dots x_n$ множества $\{1, \dots, m\}$ перечислены в лексикографическом порядке, причем $x_1 \leq \dots \leq x_n$, то сколько из них начинается с числа j ?

16. [20] (Нараяна Пандита (*Nārāyaṇa Paṇḍita*), 1356.) Разработайте алгоритм для генерации всех композиций числа n из частей, не превышающих q , т.е. всех упорядоченных разбиений $n = a_1 + \dots + a_t$, где $1 \leq a_j \leq q$ при $1 \leq j \leq t$ и произвольном значении t . Проиллюстрируйте ваш метод для $n = 7$ и $q = 3$.

17. [HM27] Проанализируйте алгоритм из упражнения 16.

18. [10] Шуточный вопрос: Лейбниц опубликовал свою *Dissertatio de Arte Combinatoria* в 1666 году. Что особенного в этом году с точки зрения перестановок?

19. [17] В какой из строф Путеануса (Puteanus) (20) ‘tibi’ трактуется как —, а не как ——?

20. [M25] К визиту трех титулованных особ в Дрезден в 1617 году поэт опубликовал 1617 перестановок гекзаметра

Dant tria jam Dresdæ, ceu sol dat, lumina lucem.

“Тroe даны ныне Дрездену, как Солнце дает луч за лучом”. [Gregor Kleppis, *Proteus Poeticus* (Leipzig: 1617).] Сколько перестановок этих слов являются гекзаметрами? Указание: строфа имеет дактили в первой и пятой стопах, в остальных стопах — спондеи.

21. [HM30] Пусть $f(p, q, r; s, t)$ — количество способов получить (o^p, o^q, o^r) путем конкатенации строк $\{s \cdot o, t \cdot oo\}$, где $p + q + r = s + 2t$. Например, $f(2, 3, 2; 3, 2) = 5$, поскольку этими пятью способами являются

$$(o|o, o|oo, oo), \quad (o|o, oo|o, oo), \quad (oo, o|o|o, oo), \quad (oo, o|oo, o|o), \quad (oo, oo|o, o|o).$$

a) Покажите, что

$$f(p, q, r; s, t) = \frac{[u^p v^q w^r z^s]1}{(1 - zu - u^2)(1 - zv - v^2)(1 - zw - w^2)}.$$

- b) Воспользуйтесь функцией f для подсчета перестановок (19), являющихся гекзаметрами, при дополнительном условии, что пятая стопа не начинается посреди слова.
- c) Подсчитайте остальные случаи.

► 22. [M40] Познакомьтесь с решениями задачи о количестве гекзаметров среди перестановок (19), опубликованными Престетом, Уоллисом, Уитвортом и Хартли. Какие ошибки они допустили?

23. [20] Какой порядок 52 диаграмм генджи-ко соответствует алгоритму 7.2.1.5Н?

► 24. [23] В начале 1800-х годов Тошиаки Хонда (Toshiaki Honda) предложил рекурсивное правило для генерации всех разбиений $\{1, \dots, n\}$. При $n = 4$ его алгоритм давал такую последовательность разбиений:



Можете ли вы сказать, какой будет последовательность при $n = 5$? Указание: см. формулу (26).

25. [15] Автора изданной в XVI веке книги *The Arte of English Poesie* интересовали только “полные” в смысле упражнения 7.2.1.5–35 схемы рифм; другими словами, каждая строка должна рифмоваться, как минимум, с одной какой-то другой. Кроме того, схемы должны быть “неприводимы” в смысле упражнения 7.2.1.2–100: разбиение наподобие 12 | 345 можно разделить на двухстрочный стих, за которым следует трехстрочный. Наконец, схема не должна тривиально состоять из строк, которые рифмуются каждая с каждой. Является ли при этих условиях (28) полным списком пятистрочных схем рифм?

► 26. [HM25] Сколько n -строчных схем рифм удовлетворяют ограничениям упражнения 25?

► 27. [NM31] Разбиение множества 14|25|36 может быть представлено диаграммой генджи-ко наподобие ; однако каждая такая диаграмма для данного разбиения должна иметь, как минимум, три места пересечения линий, а такие пересечения иногда рассматриваются как нежелательные. Сколько разбиений множества $\{1, \dots, n\}$ имеют диаграммы генджи-ко, в которых линии пересекаются не более одного раза?

► 28. [25] Пусть a , b и c — простые числа. Джон Уоллис (John Wallis) перечислил все возможные разложения a^3b^2c следующим образом: $cbaaa, cbbaa \cdot a, bcaa \cdot b, bbaaa \cdot c, cbba \cdot aa, cbba \cdot a \cdot a, cbaa \cdot ba, cbaa \cdot b \cdot a, bbaa \cdot ca, bbaa \cdot c \cdot a, caaa \cdot bb, caaa \cdot b \cdot b, baaa \cdot cb, baaa \cdot c \cdot b, cbb \cdot aaa, cbb \cdot aa \cdot a, cbb \cdot a \cdot a \cdot a, cba \cdot baa, cba \cdot ba \cdot a, cba \cdot aa \cdot b, cba \cdot b \cdot a \cdot a, bba \cdot caa, bba \cdot ca \cdot a, bba \cdot aa \cdot c, bba \cdot c \cdot a \cdot a, caaa \cdot bb \cdot a, caaa \cdot ba \cdot b, caaa \cdot b \cdot b \cdot a, baa \cdot cb \cdot a, baa \cdot ca \cdot b, baa \cdot b \cdot a, baa \cdot c \cdot b \cdot a, aaa \cdot cb \cdot b, aaa \cdot bb \cdot c, aaa \cdot c \cdot b \cdot b, cb \cdot ba \cdot aa, cb \cdot ba \cdot a \cdot a, cb \cdot aa \cdot b \cdot a, cb \cdot b \cdot a \cdot a, bb \cdot ca \cdot aa, bb \cdot ca \cdot a \cdot a, bb \cdot aa \cdot c \cdot a, bb \cdot c \cdot a \cdot a \cdot a, ca \cdot ba \cdot ba, ca \cdot ba \cdot b \cdot a, ca \cdot aa \cdot b \cdot b, ca \cdot b \cdot b \cdot a \cdot a, ba \cdot ba \cdot c \cdot a, ba \cdot aa \cdot c \cdot b, ba \cdot c \cdot b \cdot a \cdot a, aa \cdot c \cdot b \cdot b \cdot a \cdot a, c \cdot b \cdot b \cdot a \cdot a \cdot a$. Какой алгоритм использовал Уоллис для генерации разложения в данном порядке?

► 29. [24] В каком порядке Уоллис (Wallis) сгенерировал бы все разложения числа $abcde = 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17$? Ваш ответ должен представлять собой последовательность диаграмм генджи-ко.

30. [M20] Чему равен коэффициент $a_1^{i_1}a_2^{i_2} \dots z^{m+n}$ в $(a_0z + a_1z^2 + a_2z^3 + \dots)^m$? (См. формулу (29).)

31. [20] Сравните упорядочения разбиений (30) де Муавра (de Moivre) и (31) де Монморта (de Montmort) с алгоритмом 7.2.1.4Р.

32. [21] (Р.Й. Башкович (R.J. Boscovich), 1748.) Перечислите все разбиения 20, у которых все части представляют собой 1, 7 или 10. Разработайте алгоритм для перечисления всех таких разбиений заданного целого числа $n > 0$.

Ответы к упражнениям

Раздел 7.2.1.6

1. Он может “видеть” левую скобку слева от каждого внутреннего узла, а правую — снизу от каждого внутреннего узла. Можно также связать правые скобки со встречающимися *внешними* узлами, кроме самого последнего \square (см. упражнение 20).

2. Z1. [Инициализация.] Установить $z_k \leftarrow 2k - 1$ для $0 \leq k \leq n$ (считаем, что $n \geq 2$).

Z2. [Посещение.] Посетить сочетание $z_1 z_2 \dots z_n$.

Z3. [Простой случай?] Если $z_{n-1} < z_n - 1$, установить $z_n \leftarrow z_n - 1$ и вернуться к Z2.

Z4. [Поиск j .] Установить $j \leftarrow n - 1$ и $z_n \leftarrow 2n - 1$. Пока $z_{j-1} = z_j - 1$, устанавливать $z_j \leftarrow 2j - 1$ и $j \leftarrow j - 1$.

Z5. [Уменьшение z_j .] Завершить работу алгоритма, если $j = 1$. В противном случае установить $z_j \leftarrow z_j - 1$ и вернуться к шагу Z2. ■

3. Пометим узлы дерева в прямом порядке обхода. Первые $z_k - 1$ элементов $a_1 \dots a_{2n}$ содержат $k - 1$ левых скобок и $z_k - k$ правых скобок. Таким образом, имеется избыток $2k - 1 - z_k$ левых скобок по сравнению с правыми, когда червь впервые достигает узла k , а $2k - 1 - z_k$ представляет собой уровень (или глубину) этого узла.

Пусть $q_1 \dots q_n$ — инверсия $p_1 \dots p_n$, так что узел k представляет собой q_k -й узел в обратном порядке обхода. Поскольку k находится слева от j в $p_1 \dots p_n$ тогда и только тогда, когда $q_k < q_j$, мы видим, что c_k — количество узлов j , предшествующих k в прямом порядке обхода, но следующих за ним в обратном порядке, т.е. количество предков k ; это вновь дает нам уровень k .

Альтернативное доказательство. Можно также показать, что обе последовательности — $z_1 \dots z_n$ и $c_1 \dots c_n$ — обладают, по сути, одной и той же рекурсивной структурой (5): $Z_{pq} = (Z_{p(q-1)} + 1^p), 1 (Z_{(p-1)q} + 1^{p-1})$ при $0 \leq p \leq q$, а $C_{pq} = C_{p(q-1)}, (q-p) C_{(p-1)q}$. (Рассмотрите пары к последней, предпоследней и т.д. левым скобкам.)

Кстати, формула ' $c_{k+1} + d_k = c_k + 1$ ' эквивалентна (11).

4. Почти истинно; просто строки $d_1 \dots d_n$ и $z_1 \dots z_n$ находятся в *убывающем* порядке, в то время как строки $p_1 \dots p_n$ и $c_1 \dots c_n$ — в *возрастающем*. (Это лексикографическое свойство для последовательности перестановок $p_1 \dots p_n$ не наследуется автоматически из лексикографического порядка соответствующих таблиц инверсий $c_1 \dots c_n$; этот результат выполняется для данного конкретного класса $p_1 \dots p_n$.)

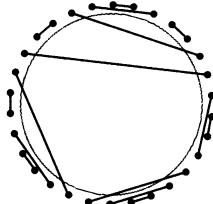
5. $d_1 \dots d_{15} = 0 \ 2 \ 0 \ 0 \ 2 \ 0 \ 0 \ 1 \ 0 \ 3 \ 2 \ 0 \ 1 \ 0 \ 4;$

$z_1 \dots z_{15} = 1 \ 2 \ 5 \ 6 \ 7 \ 10 \ 11 \ 12 \ 14 \ 15 \ 19 \ 22 \ 23 \ 25 \ 26;$

$p_1 \dots p_{15} = 2 \ 1 \ 5 \ 4 \ 8 \ 10 \ 9 \ 7 \ 11 \ 6 \ 13 \ 15 \ 14 \ 12 \ 3;$

$c_1 \dots c_{15} = 0 \ 1 \ 0 \ 1 \ 2 \ 1 \ 2 \ 3 \ 3 \ 4 \ 2 \ 1 \ 2 \ 2 \ 3.$

6. Скобки связаны друг с другом, как обычно; мы просто искривляем строку и замыкаем ее в кольцо так, что a_{2n} становится соседним с a_1 , и замечаем, что разница между левыми и правыми скобками может быть восстановлена из контекста. Если a_1 соответствует низу окружности, как в табл. 1, мы получим показанную диаграмму.



[А. Эррера (A. Errera), *Mémoires de la Classe Sci. 8°, Acad. Royale de Belgique*, (2) **11**, 6 (1931), 26 pp.]

7. (а) Она равна))()...(); присваивание $a_1 \leftarrow '('$ восстановит начальное состояние строки.

(б) Будет восстановлено исходное бинарное дерево (из шага В1), за исключением того, что $l_n = n + 1$.

8. $l_1 \dots l_{15} = 2 \ 0 \ 4 \ 5 \ 0 \ 7 \ 8 \ 0 \ 10 \ 0 \ 0 \ 13 \ 0 \ 15 \ 0;$

$r_1 \dots r_{15} = 3 \ 0 \ 0 \ 6 \ 0 \ 12 \ 11 \ 9 \ 0 \ 0 \ 0 \ 0 \ 14 \ 0 \ 0;$

$e_1 \dots e_{15} = 1 \ 0 \ 3 \ 1 \ 0 \ 2 \ 2 \ 0 \ 1 \ 0 \ 0 \ 2 \ 0 \ 1 \ 0;$

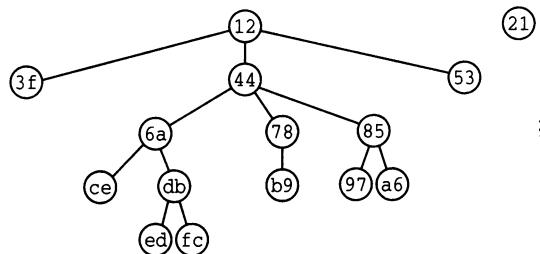
$s_1 \dots s_{15} = 1 \ 0 \ 12 \ 1 \ 0 \ 5 \ 3 \ 0 \ 1 \ 0 \ 0 \ 3 \ 0 \ 1 \ 0.$

9. Узел j является предком узла k тогда и только тогда, когда $s_j + j \geq k$. (Как следствие, получаем $c_1 + \dots + c_n = s_1 + \dots + s_n$.)

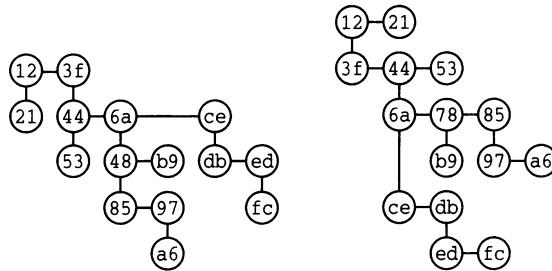
10. Если j — индекс z_k k -й левой скобки, то $w_j = c_k + 1$ и $w'_{j'} = c_k$, где j' — индекс соответствующей правой скобки.

11. Обменяйте левые и правые скобки в $a_{2n} \dots a_1$ для получения зеркального изображения $a_1 \dots a_{2n}$.

12. Зеркальное отображение (4) соответствует лесу



однако суть транспонирования станет более очевидна, если изобразить связи с правым братом и левым потомком горизонтально и вертикально, а затем выполнить транспонирование получившейся матрицы.



13. (а) По индукции по количеству узлов можно получить⁵

$$\text{preorder}(F^R) = \text{postorder}(F)^R \text{ и } \text{postorder}(F^R) = \text{preorder}(F)^R.$$

(б) Пусть F соответствует бинарному дереву B . Тогда, как упоминалось после 2.3.2–(6), $\text{preorder}(F) = \text{preorder}(B)$ и $\text{postorder}(F) = \text{inorder}(B)$. Поэтому $\text{preorder}(F^T) = \text{preorder}(B^R) = \text{postorder}(B)^R$ не имеет простой связи с $\text{preorder}(F)$ или $\text{postorder}(F)$. Но

$$\text{postorder}(F^T) = \text{inorder}(B^R) = \text{inorder}(B)^R = \text{postorder}(F)^R.$$

14. Согласно ответу к упражнению 13, $\text{postorder}(F^{RT}) = \text{preorder}(F) = \text{preorder}(B)$ где F естественным путем соответствует B ; в свою очередь, $\text{postorder}(F^{TR}) = \text{preorder}(F^T)^R = \text{postorder}(B)$. Таким образом, $F^{RT} = F^{TR}$ тогда и только тогда, когда F имеет не более одного узла.

15. Если F^R естественным образом соответствует бинарному дереву B' , корнем B' является корень крайнего справа дерева F . Левая связь узла x в B' представляет собой связь с крайним слева потомком x в F^R , который является крайним справа потомком x в F ; аналогично: правая связь представляет собой связь с левым братом в F .

Примечание: поскольку B естественным путем соответствует F^{RT} , ответ к упражнению 13 гласит, что

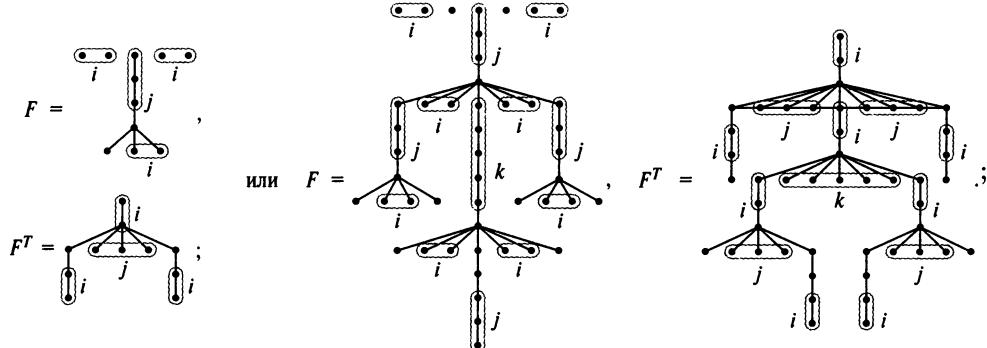
$$\text{inorder}(B) = \text{postorder}(F^{RT}) = \text{postorder}(F^R)^R = \text{preorder}(F).$$

16. Лес $F \mid G$ получается путем размещения деревьев F под первым в обратном порядке обхода узлом G . Ассоциативность этого оператора доказывается следующим образом: $F \mid (G|H) = (H^T G^T F^T)^T = (F \mid G) \mid H$. Заметим, кстати, что $\text{postorder}(F \mid G) = \text{postorder}(F) \text{postorder}(G)$ и что $F \mid (GH) = (F \mid G) H$, если G — непустой лес.

17. Любой непустой лес можно записать как $F = (G \mid \cdot) H$, где \cdot означает одноузловой лес; тогда $F^R = H^R (G^R \mid \cdot)$ и $F^T = (H^T \mid \cdot) G^T$. В частности, равенство $F^R = F^T$ невозможно, если только H не является пустым лесом Λ , поскольку первое дерево H^R не может быть $H^T \mid \cdot$; G также должно быть Λ . Кроме того, $F = F^T$ тогда и только тогда, когда $G = H^T$. В этом случае невозможно также выполнение $F^R = F^{RT}$, если только не выполняется условие $G = \Lambda$; в противном случае первое дерево G^{TR} должно иметь больше узлов, чем само дерево G .

Похоже на то, что условие $F^{RT} = F^{TR}$ не выполняется, если только не выполняется условие $F = F^R$. При этом предположении $F^{RT} = F^{TR}$ тогда и только тогда, когда и F и F^T являются самосопряженными. Дэвид Каллан (David Callan) открыл два бесконечных семейства таких лесов с параметрами $i, j, k \geq 0$.

⁵Напомним, что preorder — прямой порядок обхода, postorder — обратный порядок, а inorder — симметричный порядок обхода. — Примеч. пер.



(В этих примерах $i = 2$, $j = 3$ и $k = 5$.) Существуют ли другие возможности?

18. Всего имеется $C_{15} = 9\,694\,845$ лесов, разбитых на 20 982 класса. Наибольший класс представляет собой цикл длиной 58 968, одним из элементов которого является $((((((())))))((((()))))$. Самыми короткими являются шесть двухэлементных классов (в соответствии с упражнением 17), состоящих из строк

$$\begin{array}{ll}
 ()()()()()()()(), & ()()()((())()()()()), \\
 ()()((())((())((())(), & ()()(((((())))((())(), \\
 ()((())((())((())(), & ()(((((())))((())((())()
 \end{array}$$

и их транспозиций.

Строки $((((((())))))((())()()()()$, $((())()()()((((())))))$ и $((((())((())((())((())$) имеют клинообразные бинарные деревья и образуют единый класс размером 3. Путь, проходящий от $((((((())))))((())((())((())$) до $((((((())))))((())((())((())$), содержит 3 120 элементов, одним из которых является (2). В соответствии с гипотезой из ответа к упражнению 19, кратчайший возможный цикл имеет длину 6; при $n = 15$ имеется 66 таких циклов. (Следующий в порядке возрастания размера цикл имеет длину 10 и включает элемент $((((((())))))((())((())$; цикл с такой длиной только один.)

19. Преобразование F_j в F_{j+1} в алгоритме Р можно перефразировать следующим образом: “найти последний узел в прямом порядке обхода, скажем, x , который имеет левого брата, скажем, y . Удалить x из его семейства и сделать его новым крайним справа дочерним узлом y . И если $x < n$, заменить всех потомков x ($x + 1, \dots, n$) тривиальными одноузловыми деревьями”.

Таким образом, преобразование F_j^R в F_{j+1}^R можно описать следующим образом, если вспомнить, что k -й узел F_j в прямом порядке обхода представляет собой k -й с конца узел F_j^R в обратном порядке обхода: “найти первый узел в прямом порядке обхода, скажем, x , у которого есть правый брат, скажем, y . Удалить x из его семейства и сделать его новым крайним слева дочерним узлом y . И если $x < n$, заменить всех потомков x ($x + 1, \dots, n$) тривиальными одноузловыми деревьями”.

Аналогично можно перефразировать преобразование G_j в G_{j+1} , выполняемое алгоритмом В: “найти j , корень крайнего слева нетривиального дерева; затем найти k , его крайний справа дочерний узел. Удалить k и его потомков из семейства j и вставить их между j и правым братом j . Наконец, если $j > 1$, сделать j и его братьев дочерними по отношению к $j - 1$, а $j - 1$ — дочерним узлом $j - 2$ и т.д.”.

Когда это преобразование изменяет представление с левым братом и правым сыном с G_j^{RT} на G_{j+1}^{RT} (см. упражнение 15), оно оказывается идентичным преобразованию F_j^R в F_{j+1}^R в представлении с левым сыном и правым братом. Следовательно, $G_j^{RT} = F_j^R$, поскольку при $j = 1$ выполнение этого тождества очевидно.

(Отсюда вытекает, что последовательность таблиц $e_1 \dots e_{n-1}$ для бинарных деревьев, сгенерированная алгоритмом B, представляет собой последовательность таблиц $d_{n-1} \dots d_1$ для строк скобок, сгенерированную алгоритмом P; это явление продемонстрировано в табл. 1 и 2.)

Лес F^{RT^R} называется *дуальным* лесу F ; см. упражнение 26 (f). Ряд симметрий списков лесов был изучен М.Ч. Эром (M.C. Er) [Comp. J., 32 (1989), 76–85].

- 20.** (a) Это утверждение, обобщающее лемму 2.3.1Р, легко доказать по индукции.
 (b) Приведенная ниже процедура практически идентична алгоритму P.

T1. [Инициализация.] Установить $b_{3k-2} \leftarrow 3$ и $b_{3k-1} \leftarrow b_{3k} \leftarrow 0$ для $1 \leq k \leq n$;
 установить также $b_0 \leftarrow b_N \leftarrow 0$ и $m \leftarrow N - 3$, где $N = 3n + 1$.

T2. [Посещение.] Посетить последовательность $b_1 \dots b_N$. (Сейчас $b_m = 3$ и $b_{m+1} \dots b_N = 0 \dots 0$.)

T3. [Простой случай?] Установить $b_m \leftarrow 0$. Если $b_{m-1} = 0$, установить $b_{m-1} \leftarrow 3$,
 $m \leftarrow m - 1$ и перейти к шагу T2.

T4. [Поиск j .] Установить $j \leftarrow m - 1$ и $k \leftarrow N - 3$. Пока $b_j = 3$, устанавливать
 $b_j \leftarrow 0$, $b_k \leftarrow 3$, $j \leftarrow j - 1$ и $k \leftarrow k - 3$.

T5. [Увеличение b_j .] Завершить работу алгоритма, если $j = 0$. В противном случае
 установить $b_j \leftarrow 3$, $m \leftarrow N - 3$ и вернуться к шагу T2. ■

[См. S. Zaks, *Theoretical Comp. Sci.*, 10 (1980), 63–82. В этой статье Закс указывает, что еще проще сгенерировать последовательность $z_1 \dots z_n$ индексов j , таких, что $b_j = 3$, с использованием алгоритма, практически идентичного алгоритму из ответа к упражнению 2, поскольку комбинация $z_1 \dots z_n$ для корректного тернарного дерева характеризуется неравенствами $z_{k-1} < z_k \leq 3k - 2$.]

21. Для решения этой задачи можно по сути скомбинировать алгоритмы P и 7.2.1.2L. Для удобства будем считать, что $n_t > 0$ и $n_1 + \dots + n_t > 1$.

G1. [Инициализация.] Установить $l \leftarrow N$. Затем для $j = t, \dots, 2, 1$ (в указанном порядке) выполнить n_j раз следующие операции: установить $b_{l-j} \leftarrow j$, $b_{l-j+1} \leftarrow \dots \leftarrow b_{l-1} \leftarrow 0$ и $l \leftarrow l - j$. Наконец, установить $b_0 \leftarrow b_N \leftarrow c_0 \leftarrow 0$ и $m \leftarrow N - t$.

G2. [Посещение.] Посетить $b_1 \dots b_N$. (В этот момент $b_m > 0$ и $b_{m+1} = \dots = b_N = 0$.)

G3. [Простой случай?] Если $b_{m-1} = 0$, установить $b_{m-1} \leftarrow b_m$, $b_m \leftarrow 0$, $m \leftarrow m - 1$ и вернуться к шагу G2.

G4. [Поиск j .] Установить $c_1 \leftarrow b_m$, $b_m \leftarrow 0$, $j \leftarrow m - 1$ и $k \leftarrow 1$. Пока $b_j \geq c_k$, устанавливать $k \leftarrow k + 1$, $c_k \leftarrow b_j$, $b_j \leftarrow 0$ и $j \leftarrow j - 1$.

G5. [Увеличение b_j .] Если $b_j > 0$, найти наименьшее $l \geq 1$, такое, что $b_j < c_l$, и выполнить обмен $b_j \leftrightarrow c_l$. Иначе если $j > 0$, установить $b_j \leftarrow c_1$ и $c_1 \leftarrow 0$. В противном случае завершить работу алгоритма.

G6. [Обращение и развертывание.] Установить $j \leftarrow k$ и $l \leftarrow N$. Пока $c_j > 0$, устанавливать $b_{l-c_j} \leftarrow c_j$, $l \leftarrow l - c_j$ и $j \leftarrow j - 1$. Затем установить $m \leftarrow N - c_k$ и вернуться к шагу G2. ■

В этом алгоритме предполагается, что $N > n_1 + 2n_2 + \dots + tn_t$. [См. SICOMP, 8 (1979), 73–81.]

22. Вначале заметим, что значение d_1 может увеличиваться тогда и только тогда, когда в связанным представлении $r_1 = 0$. В противном случае следующий за $d_1 \dots d_{n-1}$ элемент получается путем поиска наименьшего j , такого, что $d_j > 0$, и установкой $d_j \leftarrow 0$, $d_{j+1} \leftarrow d_{j+1} + 1$. Можно также считать, что $n > 2$.

K1. [Инициализация.] Установить $l_k \leftarrow k + 1$ и $r_k \leftarrow 0$ для $1 \leq k < n$; установить также $l_n \leftarrow r_n \leftarrow 0$.

K2. [Посещение.] Посетить бинарное дерево, представленное связями $l_1l_2 \dots l_n$ и $r_1r_2 \dots r_n$.

K3. [Простые случаи?] Установить $y \leftarrow r_1$. Если $y = 0$, установить $r_1 \leftarrow 2$, $l_1 \leftarrow 0$ и вернуться к шагу K2. Иначе если $l_1 = 0$, установить $l_1 \leftarrow 2$, $r_1 \leftarrow r_2$, $r_2 \leftarrow l_2$, $l_2 \leftarrow 0$ и вернуться к шагу K2. В противном случае установить $j \leftarrow 2$ и $k \leftarrow 1$.

K4. [Поиск j и k .] Если $r_j > 0$, установить $k \leftarrow j$ и $y \leftarrow r_j$. Затем, если $j \neq y - 1$, установить $j \leftarrow j + 1$ и повторить данный шаг.

K5. [Перетасовка поддеревьев.] Установить $l_j \leftarrow y$, $r_j \leftarrow r_y$, $r_y \leftarrow l_y$ и $l_y \leftarrow 0$. Если $j = k$, перейти к шагу K2.

K6. [Сдвиг поддеревьев.] Завершить работу алгоритма, если $y = n$. В противном случае, пока $k > 1$, устанавливать $k \leftarrow k - 1$, $j \leftarrow j - 1$ и $r_j \leftarrow r_k$. Затем, пока $j > 1$, устанавливать $j \leftarrow j - 1$ и $r_j \leftarrow 0$. Вернуться к шагу K2. ■

(См. анализ алгоритма в упражнении 45. Корш [Comp. J., 48 (2005), 488–497; 49 (2006), 351–357] показал, что данный алгоритм можно расширить для t -арных деревьев, а также нашел эффективное обобщение алгоритма В для t -арных деревьев.)

23. (а) Поскольку переменная z_n начинается с $2n - 1$ и проходит назад и вперед C_{n-1} раз, в конце ее значение оказывается равным $2n - 1 - (C_{n-1} \bmod 2)$ при $n > 1$. Кроме того, последнее значение z_j представляет собой константу при всех $n \geq j$. Таким образом, последняя строка $z_1z_2 \dots$ имеет вид 1 2 5 6 9 11 13 14 17 19 ... и содержит все нечетные числа, меньшие $2n$, за исключением 3, 7, 15, 31, ...

(б) Аналогично: перестановка в прямом порядке обхода, характеризующая последнее дерево, представляет собой $2^k \ 2^{k-1} \ \dots \ 1 \ 3 \ 5 \ 6 \ 7 \ 9 \ 10 \ \dots$, где $k = \lfloor \lg n \rfloor$. В лесу узел 2^j является родительским по отношению к 2^{j-1} узлам $\{2^{j-1}, 2^{j-1} + 1, \dots, 2^j - 1\}$, $1 < j \leq k$, а деревья $\{2^k + 1, \dots, n\}$ тривиальны.

Примечание: если алгоритм N после его окончания перезапустить с шага N2, он генерирует ту же последовательность, но в обратном порядке. Алгоритм L обладает тем же свойством.

24. $l_0 l_1 \dots l_{15} = 2 \ 0 \ 1 \ 0 \ 3 \ 0 \ 0 \ 6 \ 5 \ 0 \ 8 \ 0 \ 0 \ 12 \ 11 \ 4; r_1 \dots r_{15} = 0 \ 15 \ 0 \ 10 \ 7 \ 0 \ 0 \ 9 \ 0 \ 14 \ 13 \ 0 \ 0 \ 0 \ 0; k_1 \dots k_{15} = 0 \ 0 \ 2 \ 2 \ 4 \ 5 \ 5 \ 4 \ 8 \ 4 \ 10 \ 11 \ 11 \ 10 \ 2; q_1 \dots q_{15} = 2 \ 1 \ 15 \ 4 \ 3 \ 10 \ 8 \ 5 \ 7 \ 6 \ 9 \ 14 \ 11 \ 13 \ 12 \text{ и } u_1 \dots u_{15} = 12 \ 3 \ 1 \ 0 \ 0 \ 5 \ 0 \ 3 \ 1 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0.$ (Если узлы леса F пронумерованы в обратном порядке обхода, k_j является левым братом j ; или если j — крайний слева дочерний узел p , то $k_j = k_p$. Формулируя иначе, k_j — родитель j в лесу F^{TR} . И k_j равно также $j - 1 - u_{n+1-j}$, количеству элементов, меньших j , находящихся в строке $q_1 \dots q_n$ слева от j .)

25. Используя подсказки из алгоритмов N и R, мы хотим расширить каждое $(n - 1)$ -узловое дерево в список из двух или большего количества n -узловых деревьев. Идея в этом случае заключается в том, чтобы сделать n дочерним узлом $n - 1$ в бинарном дереве в начале и конце каждого такого списка. Приведенный далее алгоритм использует дополнительные поля связей p_j и s_j , где p_j указывает на родительский по отношению к j узел в лесу, а s_j указывает на левого брата j или на крайнего справа брата j , если j — крайний слева в своем семействе. (Эти указатели p_j и s_j , конечно же, не являются перестановками $p_1 \dots p_n$ в табл. 1 или полями $s_1 \dots s_n$ в табл. 2. Фактически $s_1 \dots s_n$ представляют собой перестановки λ из упражнения 33.)

M1. [Инициализация.] Установить $l_j \leftarrow j + 1, r_j \leftarrow 0, s_j \leftarrow j, p_j \leftarrow j - 1$ и $o_j \leftarrow -1$ для $1 \leq j \leq n$, за исключением $l_n \leftarrow 0$.

M2. [Посещение.] Посетить $l_1 \dots l_n$ и $r_1 \dots r_n$. Затем установить $j \leftarrow n$.

M3. [Поиск j .] Если $o_j > 0$, установить $k \leftarrow p_j$ и перейти к шагу M5, если $k \neq j - 1$. Если $o_j < 0$, установить $k \leftarrow s_j$ и перейти к шагу M4, если $k \neq j - 1$. Если $k = j - 1$ в любом случае, установить $o_j \leftarrow -o_j, j \leftarrow j - 1$ и повторить данный шаг.

M4. [Передача вниз.] (В этот момент k представляет собой левого брата j или крайний справа член семейства j .) Если $k \geq j$, завершить работу алгоритма, если $j = 1$; в противном случае установить $x \leftarrow p_j, l_x \leftarrow 0, z \leftarrow k$ и $k \leftarrow 0$ (тем самым отсоединив узел j от родительского узла и выведя его на верхний уровень). Но если $k < j$, установить $x \leftarrow p_j + 1, z \leftarrow s_x, r_k \leftarrow 0$ и $s_x \leftarrow k$ (тем самым отсоединив j от k и перенеся его на уровень вниз). Затем установить $x \leftarrow k + 1, y \leftarrow s_x, s_x \leftarrow z, s_j \leftarrow y, r_y \leftarrow j$ и $x \leftarrow j$. Пока $x \neq 0$, устанавливать $p_x \leftarrow k$ и $x \leftarrow r_x$. Вернуться к шагу M2.

M5. [Передача вверх.] (В этот момент k является родителем j .) Установить $x \leftarrow k + 1, y \leftarrow s_j, z \leftarrow s_x, s_x \leftarrow y$ и $r_y \leftarrow 0$. Если $k \neq 0$, установить $y \leftarrow p_k, r_k \leftarrow j, s_j \leftarrow k, s_{y+1} \leftarrow z$ и $x \leftarrow j$; в противном случае установить $y \leftarrow j - 1, l_y \leftarrow j, s_j \leftarrow z$ и $x \leftarrow j$. Пока $x \neq 0$, устанавливать $p_x \leftarrow y$ и $x \leftarrow r_x$. Вернуться к шагу M2. |

Примечание о времени работы алгоритма: можно доказать, как в упражнении 44, что стоимость шага M3 составляет $2C_n + 3(C_{n-1} + \dots + C_1)$ обращений к памяти и что шаги M4 и M5 вместе стоят $8C_n - 2(C_{n-1} + \dots + C_1)$ плюс удвоенное количество присваиваний $x \leftarrow r_x$. Последнюю величину трудно точно проанализировать; например, при $n = 15$ и $j = 6$ алгоритм устанавливает $x \leftarrow r_x$ (1, 2, 3, 4, 5, 6) раз в (45, 23, 7, 9, 2, 4) случаях соответственно. Однако эвристически среднее количество присваиваний $x \leftarrow r_x$ должно быть примерно $2 - 2^{j-n}$ для данного j , т.е. всего около $(2C_n - (C_n - C_{n-1}) - (C_{n-1} - C_{n-2})/2 - (C_{n-2} - C_{n-3})/4 - \dots)/C_n \approx 8/7$.

Эмпирические тесты подтверждают предсказанное поведение, показывая, что общая стоимость в пересчете на одно дерево стремится к $265/21 \approx 12.6$ обращений к памяти при $n \rightarrow \infty$.

26. (a) Необходимость условия очевидна. А если оно выполняется, можно однозначно построить F : узел 1 и его братья являются корнями леса, а их потомки индуктивно определяются непересекающимися разбиениями. (Фактически можно вычислить координаты глубины $c_1 \dots c_n$ непосредственно из ограниченно растущей последовательности $\Pi = a_1 \dots a_n$: установить $c_1 \leftarrow 0$ и $i_0 \leftarrow 0$. Для $2 \leq j \leq n$, если $a_j > \max(a_1 \dots a_{j-1})$, установить $c_j \leftarrow c_{j-1} + 1$ и $i_{a_j} \leftarrow c_j$, в противном случае установить $c_i \leftarrow i_{a_j}$.)

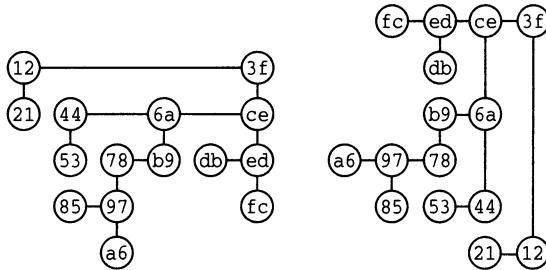
(b) Если Π и Π' удовлетворяют условию непересекаемости, так что их наибольшим общим уточнением⁶ является $\Pi \vee \Pi'$, и можно поступить так, как в упражнении 7.2.1.5–12 (a).

(c) Пусть x_1, \dots, x_m — дочерние узлы некоторого узла F и $1 \leq j < k \leq m$. Образуем F' путем удаления x_{j+1}, \dots, x_k из их семейства и присоединим их в качестве дочерних к узлу $x_{j+1} - 1$, крайнему правому потомку x_j .

(d) Очевидно из ответа (c). Таким образом, леса ранжируются снизу вверх по количеству содержащихся в них внутренних узлов (которое на единицу меньше количества блоков в Π).

(e) Ровно $\sum_{k=0}^n e_k (e_k - 1)/2$, где $e_0 = n - e_1 - \dots - e_n$ — количество корней.

(f) Дуализация подобна операции транспонирования в упражнении 12, но вместо левого сына и правого брата мы используем левого брата и правого сына и выполняем транспонирование относительно *младшей* диагонали.



(“Правые” связи теперь идут вниз. Обратите внимание, что j является крайним справа дочерним узлом k в F тогда и только тогда, когда j является левым братом k в F^D . Прямой порядок обхода F^D представляет собой обращение прямого порядка обхода F , так же, как обратный порядок обхода F^T представляет собой обращение обратного порядка обхода F .)

(g) Из ответа (f) видно, что F' покрывает F тогда и только тогда, когда F^D покрывает F'^D . (Следовательно, F^D имеет $n + 1 - k$ листьев, если у F их k .)

(h) $F \wedge F' = (F^D \vee F'^D)^D$.

⁶См. упражнение 7.2.1.5–12. — Примеч. ред.

(i) Нет. Если бы это было так, то в соответствии с дуальностью должно было бы выполнятся равенство. Но, например, $0101 \wedge 0121 = 0000$ и $0101 \vee 0121 = 0123$, в то время как $\text{leaves}(0101) + \text{leaves}(0121) \neq \text{leaves}(0000) + \text{leaves}(0123)$ ⁷.

[Непересекающиеся разбиения впервые были рассмотрены Г.В. Беккером (H.W. Becker) в *Math. Mag.*, **22** (1948), 23–26. В 1971 году Г. Креверас (G. Kreweras) доказал, что они образуют решетку; см. ссылки в ответе к упражнению 2.3.4.6–3.]

27. (a) Это утверждение эквивалентно упражнению 2.3.3–19.

(b) Если мы представим лес с использованием связей с правым сыном и левым братом, то прямой порядок обхода будет соответствовать симметричному обходу бинарного дерева (см. упражнение 2.3.2–5), а s_j будет равно размеру правого поддерева узла j . Поворот влево вокруг любого внутреннего узла этого бинарного дерева уменьшает ровно одну из координат s , и величина уменьшения настолько мала, насколько это возможно при условии корректности таблицы $s_1 \dots s_n$. Следовательно, F' покрывает F тогда и только тогда, когда F получается из F' путем такого поворота. (Поворот в представлении с левым сыном и правым братом аналогичен, но по отношению к обратному порядку обхода.)

(c) Дуализация сохраняет отношение покрытия, но заменяет “лево” на “право” и наоборот.

(d) $F \top F' = (F^D \perp F'^D)^D$. Таким же образом, как указывается в упражнении 6.2.3–32, можно независимо минимизировать размеры левых поддеревьев.

(e) Покрывающее преобразование в ответе 26 (c) очевидным образом делает $s_j \leq s'_j$ для всех j .

(f) Истинно, потому что $F \wedge F' \subset F \dashv F \perp F'$ и $F \wedge F' \subset F' \dashv F \perp F'$.

(g) Ложно; например, $0121 \vee 0122 = 0123$ и $0121 \wedge 0122 = 0122$. (Но, применяя дуализацию к выражению из ответа (f), получим $F \top F' \dashv F \vee F'$.)

(h) Длиннейший путь (длиной $\binom{n}{2}$) многократно уменьшает крайнее справа ненулевое значение s_j на 1. Кратчайший путь длиной $n - 1$ просто поочередно устанавливает крайние слева ненулевые значения s_j равными 0.

Ответ к упражнению 6.2.3–32 содержит ряд ссылок на литературу о решетках Тамари.

28. (a) Нужно просто вычислить $\min(c_1, c'_1) \dots \min(c_n, c'_n)$ и $\max(c_1, c'_1) \dots \max(c_n, c'_n)$, поскольку $c_1 \dots c_n$ является корректной последовательностью тогда и только тогда, когда $c_1 = 0$ и $c_j \leq c_{j-1} + 1$ для $1 < j \leq n$.

(b) Это очевидно вытекает из ответа (a). *Примечание:* элементы произвольной дистрибутивной решетки могут быть представлены как порядковые идеалы некоторого частичного упорядочения. Для случая, приведенного на рис. 41, частичное упорядочение имеет вид , а аналогичная треугольная сетка со сторонами длиной $n - 2$ дает решетку Стенли порядка n .

(c) Возьмем узел k леса F , у которого имеется левый брат j . Удалим k из его семейства и поместим его в качестве нового правого дочернего узла j , за которым разместим его бывшие дочерние узлы в качестве дочерних узлов j ; бывшие дочерние узлы k сохраняют своих потомков. (Эта операция соответствует замене ‘()’ на ‘(())’ в строке вложенных скобок. Таким образом, “идеальный” код Грэя для скобок соответствует Гамильтонову пути в покрывающем графе решетки Стенли. Для $n = 4$

⁷ $\text{leaves}()$ означает количество листьев у аргумента этой функции. — Примеч. пер.

имеется 38 таких путей, а именно $(8, 6, 6, 8, 4, 6)$ путей от 0123 до соответственно $(1001, 0010, 0012, 0100, 0111, 0120)$.)

(d) Истинно, поскольку отношение покрытия из ответа (c) обладает лево-правой симметрией. ($F \subseteq F'$ тогда и только тогда, когда $w_j \leq w'_j$ для $0 \leq j \leq 2n$, где w_j определено в упражнении 10. Если $w_0 \dots w_{2n}$ представляет собой обход червем леса F , то обратная последовательность $w_{2n} \dots w_0$ представляет собой обход леса F^R . Обратите внимание, что отношение покрытия изменяет только одну координату w_j . Вычислить $F \cap F'$ и $F \cup F'$ можно, если искать минимумы и максимумы для w , а не для c .)

(e) См. упражнение 9. (Таким образом, $F \perp F' \subseteq F \cap F'$ и т.д., как в упражнении 27 (f).)

Примечание: Стенли разработал свою решетку в *Fibonacci Quarterly*, **13** (1975), 222–223. Поскольку три важные решетки определены на одних и тех же элементах, требуются три обозначения для различных упорядочений; здесь для этого использованы символы \leftarrow , \dashv и \subseteq , напоминающие первые буквы фамилий Креверас, Тамари и Стенли⁸.

29. Если мы “склеим” шесть правильных пятиугольников, то получим 14 вершин, координаты которых после соответствующих поворотов и масштабирования будут следующими:

$$\begin{aligned} p_{1010} &= p_{0000}^- = p_{3000}^* = p_{2100}^{*-} = \left(-1, \sqrt{3}, 2/\phi\right); \\ p_{0010} &= p_{3100}^* = \left(\phi^{-2}, \sqrt{3}\phi, 0\right); \quad p_{3010} = p_{0100}^- = (0, 0, 2); \quad p_{3210} = p_{0200}^- = (2, 0, 2/\phi); \\ p_{0210} &= p_{3200}^* = \left(\sqrt{5}, \sqrt{3}, 0\right); \quad p_{1000} = p_{2000}^* = \left(-\phi^2, \sqrt{3}/\phi, 0\right), \end{aligned}$$

где $(x, y, z)^*$ означает $(x, -y, z)$, а $(x, y, z)^-$ означает $(x, y, -z)$. Но тогда три четырехугольные “границы” не являются квадратами; более того, их вершины даже не лежат в одной плоскости.

(Однако можно получить похожее тело с правильными квадратами и неправильными пятиугольниками, соединяя два подходящих тетраэдра и обрезая три “склеенных” угла. Альтернативные множества координат для ассоциаэдра, представляющие определенный математический интерес, но не имеющие никакой внешней привлекательности для глаз, рассматриваются Гюнтером Циглером (Günter Ziegler) в *Lectures on Polytopes* (New York : Springer, 1995), пример 9.11.)

30. (a) $\bar{f}_{n-1} \dots \bar{f}_1 0$, поскольку внутренний узел j в симметричном порядке обхода имеет непустое правое поддерево тогда и только тогда, когда внутренний узел $j+1$ в симметричном порядке обхода имеет пустое левое поддерево.

(b) В общем случае, если след имеет вид $1^{p_1} 0^{q_1+1} 1^{p_2+1} 0^{q_2+1} \dots 1^{p_k+1} 0^{q_k+1}$, нам нужно подсчитать все бинарные деревья, узлы которых в симметричном порядке обхода имеют спецификацию $R^{p_1} NL^{q_1} BR^{p_2} NL^{q_2} B \dots R^{p_k} NL^{q_k}$, где B означает “оба поддерева не пусты”, R – “правое поддерево не пустое, левое – пустое”, L – “левое поддерево не пустое, правое – пустое”, а N – “оба поддерева пусты”. Это количество

⁸ В оригинале указано, что имеется в виду написание фамилии Стенли по-русски. — Примеч. *nep.*

в общем случае равно

$$\binom{p_1 + q_1}{p_1} \binom{p_2 + q_2}{p_2} \cdots \binom{p_k + q_k}{p_k} C_{k-1}$$

и в конкретном указанном в условии упражнения случае составляет

$$\begin{aligned} \binom{1+0}{1} \binom{0+0}{0} \binom{1+0}{1} \binom{5+3}{5} \binom{0+0}{0} \binom{0+0}{0} \binom{0+2}{0} \binom{0+0}{0} \binom{1+2}{1} C_8 = \\ = 240\,240. \end{aligned}$$

- (c) $d_j = 0$ тогда и только тогда, когда $c_{j+1} > c_j$, в соответствии с упражнением 3.
 (d) В общем случае след $F \perp F'$ равен $f_1 \dots f_n \wedge f'_1 \dots f'_n$, в соответствии с упражнением 27 (а); след $F \top F'$ равен $f_1 \dots f_n \vee f'_1 \dots f'_n$, согласно ответу (а) и упражнению 27 (д).

[Тот факт, что в решетке Тамари всегда имеется комплементарный элемент, доказан Г. Лаксером (H. Lakser); см. G. Grätzer, *General Lattice Theory* (1878), упражнение I.6.30.]

31. (a) 2^{n-1} (см. упражнение 6.2.2–5).

(b) $c_1 \leq \dots \leq c_n$; $d_1, \dots, d_n \leq 1$; из $e_j > 0$ вытекает $e_j + \dots + e_n = n - j$; $k_{j+1} \leq k_j + 1$; $p_1 \leq \dots \leq p_j \geq \dots \geq p_n$ для некоторого j ; из $s_j > 0$ вытекает $s_j = n - j$; $u_1 \geq \dots \geq u_n$; $z_{j+1} \leq z_j + 2$. (Прочие ограничения, применимые в общем случае, снижают количество возможных кортежей с данными свойствами до 2^{n-1} в каждом конкретном случае.)

(c) Истинно только в n случаях из 2^{n-1} . (Но F^T является вырожденным.)

(d) Вырожденный лес со следом $f_1 \dots f_n$ обладает тем свойством, что $c_{j+1} = c_j + f_j$. Элементы $j < k$ являются братьями тогда и только тогда, когда $f_j = f_{j+1} = \dots = f_{k-1} = 0$. Таким образом, если F'' является вырожденным лесом со следом $f_1 \dots f_n \wedge f'_1 \dots f'_n$, то $F'' \subset F$ и $F'' \subset F'$; следовательно, $F'' \subset F \wedge F' \dashv F \perp F'$. Согласно ответу (b), мы также имеем $F \perp F' \dashv F''$. Аналогично доказывается и то, что $F \vee F' = F \top F'$ является вырожденным лесом со следом $f_1 \dots f_n \vee f'_1 \dots f'_n$.

Таким образом, когда решетки Кревераса и Тамари ограничены вырожденными лесами, они становятся идентичными булевой решетке подмножеств $\{1, \dots, n-1\}$. [Этот результат в случае решетки Тамари открыт Д. Марковски (G. Markowsky), *Order*, 9 (1992), 265–290, статья которого содержит также описание многих других свойств решеток Тамари.]

32. Пусть F и F' имеют s -координаты соответственно $s_1 \dots s_n$ и $s'_1 \dots s'_n$. Назовем индекс j замороженным (frozen), если $s_j < s'_j$ или $j = 0$. Мы хотим определить значения замороженных координат и максимизировать остальные. Пусть $s_0 = n$ и пусть для $0 \leq k \leq n$

$$s''_k = s_j - k + j, \text{ где } j = \max \{i | 0 \leq i \leq k, i \text{ заморожено, а } i + s_j \geq k\}.$$

Поскольку $s_k \leq s_j - (k - j)$ при $0 \leq k - j \leq s_j$, мы имеем $s''_k \geq s_k$, причем равенство достигается, когда k — замороженный индекс.

Значения $s''_0 s''_1 \dots s''_n$ соответствуют корректному лесу в соответствии с условием из упражнения 27 (а). Если $k \geq 0$ и $0 \leq l \leq s''_k = s_j - k + j$ и $s''_{k+l} = s'_j - k - l + j'$, то

мы имеем $s''_{k+l} + l \leq s''_k$ для $0 \leq j' - j \leq s_j$, потому что в этом случае $s'_j + j' - j \leq s_j$. Поскольку $j + s_j \geq k + l \geq j'$, не может быть $j > j'$ или $j' > j + s_j$.

Пусть F''' — лес с координатами, удовлетворяющими условию $s_k \leq s''' \leq s''_k$. Тогда $\min(s'_k, s'''_k) = s_k$, поскольку $s_k = s''_k$ при замороженном k , и $s_k = s'_k$ в противном случае.

И наоборот, если F''' — лес, такой, что $F' \perp F''' = F$, должно выполняться $s_k \leq s'''_k \leq s''_k$. Условие $s'''_k < s_k$ влечет за собой $s'''_k < s'_k$. А если k — минимальное значение, для которого $s'''_k > s''_k$, то мы имеем $s''_k = s_j - k + j$ для некоторого замороженного j , где $0 \leq j \leq k$ и $j + s_j \geq k$. Тогда из $s'''_k \geq s_j$ вытекает $k - j \leq s'''_k$, следовательно, $s'''_k + k - j \leq s'''_j$. Если $j < k$, мы имеем $s'''_j \leq s''_j = s_j$, т.е. получаем противоречие. Но из $j = k$ вытекает, что $\min(s'''_k, s'_k) > s_k$.

Чтобы получить первый полудистрибутивный закон, применим этот принцип, заменив F на $F \perp G$, а F' — на F . Тогда из гипотез $F \dashv G \dashv F''$ и $F \dashv H \dashv F''$ следует, что $F \dashv G \wedge H \dashv F''$. Второй полудистрибутивный закон получается путем применения дуальности к первому.

(Ральф Фрис (Ralph Freese) предложил называть F'' *псевдодополнением* (pseudo-complement) F' над F .)

33. (a) Пусть $k\lambda$ равно $\text{LLINK}[k]$, если $\text{LLINK}[k] \neq 0$; в противном случае пусть оно принимает значение $\text{RLINK}[k-1]$, если $k \neq 1$, а если это не так, то пусть $k\lambda$ равно корю бинарного дерева. Это правило определяет перестановку, поскольку $k\lambda = j$ тогда и только тогда, когда либо $k = \text{parent}(j) + [j$ является правым дочерним узлом], либо $k = 1$ и j является корнем. Кроме того, $k\lambda \geq k$, когда $\text{LLINK}[k] = 0$, и $k\sigma\lambda \leq k$, когда $\text{RLINK}[k] = 0$. [Обобщение на t -арные деревья можно найти в работе P.H. Edelman, *Discrete Math.*, **40** (1982), 171–179.]

(b) Используя представление (2) из ответа к упражнению 26 (f), мы видим, что в этом случае $\lambda(F)$ равно $(3 \ 1) (2) (12 \ 6 \ 4) (5) (11 \ 7) (14 \ 13) (9 \ 8) (15) (10)$. В общем случае циклы представляют собой семейства лесов, расположенные в уменьшающемся порядке в пределах каждого цикла; узлы пронумерованы в прямом порядке обхода. [См. Dershowitz and Zaks, *Discrete Math.*, **62** (1986), 215–218.]

(c) $\lambda(F^D) = \rho\sigma\lambda\rho$, где ρ — “зеркальная” перестановка $(1 \ n) (2 \ n-1) \dots$, поскольку в дуальном лесу выполняется обмен $\text{LLINK} \leftrightarrow \text{RLINK}$ и отражение нумерации в прямом порядке обхода.

(d) Разбиение цикла $(x_j x_k)(x_1 \dots x_m) = (x_1 \dots x_j x_{k+1} \dots x_m)(x_{j+1} \dots x_k)$ соответствует ответу 26 (c).

(e) Согласно ответу (d), каждый покрывающий путь соответствует разложению $(n \dots 2 \ 1)$. Пусть q_n — количество таких разложений. Тогда мы получаем рекуррентное соотношение $q_1 = 1$ и $q_n = \sum_{l=1}^{n-1} (n-l) \binom{n-2}{l-1} q_l q_{n-l}$, поскольку имеется $n-l$ выборов, где $k-j=l$, при которых первая перестановка разбивает цикл на части размерами l и $n-l$, после чего имеется $\binom{n-2}{l-1}$ способов чередования последующих разложений. Решением рекуррентного соотношения является $q_n = n^{n-2}$, поскольку

$$\begin{aligned} \sum_{l=1}^{n-1} \binom{n-1}{l} l^{l-1} (y-l)^{n-1-l} &= \lim_{x \rightarrow 0} \sum_{l=1}^{n-1} \binom{n-1}{l} (x+l)^{l-1} (y-l)^{n-1-l} = \\ &= \lim_{x \rightarrow 0} \frac{(x+y)^{n-1} - y^{n-1}}{x} = (n-1) y^{n-1}. \end{aligned}$$

[См. J. Denes, *Magyar Tudományos Akadémia Matematikai Kutató Intézetének Közleményei*, 4 (1959), 63–70. Естественным представляется поиск соответствия между факторизацией и помеченными свободными деревьями, так как количество и тех и других равно n^{n-2} . Вероятно, простейшее из них для данного $(1 \ 2 \dots n) = (x_1 \ y_1) \dots (x_{n-1} \ y_{n-1})$, где $x_j < y_j$, следующее. Предположим, что цикл, содержащий x_j и y_j в $(x_j \ y_j) \dots (x_{n-1} \ y_{n-1})$, представляет собой $(z_1 \dots z_m)$, где $z_1 < \dots < z_m$. Если $y_j = z_m$, пусть $a_j = z_1$; в противном случае пусть $a_j = \min \{z_i | z_i > x_j\}$. Можно показать, что $a_1 \dots a_{n-1}$ — последовательность парковки $n - 1$ машин из упражнения 6.4–29, а в упражнении 6.4–31 показана ее связь со свободными деревьями.]

34. Каждый покрывающий путь снизу вверх эквивалентен диаграмме Юнга формы $(n - 1, n - 2, \dots, 1)$, так что мы можем воспользоваться теоремой 5.1.4Н (см. упражнение 5.3.4–38).

[Подсчет таких путей в решетке Тамари остается загадкой: соответствующая последовательность имеет вид 1, 1, 2, 9, 98, 2981, 340549, ...]

35. Умножьте на $n + 1$ и обратитесь к АММ, 97 (1990), 626–630.

36. Путем очевидных поправок к шагам Т1–Т5 можно обобщить результат для произвольных t -арных деревьев. Пусть $C_n^{(t)}$ — количество t -арных деревьев с n внутренними узлами; таким образом, $C_n = C_n^{(2)}$ и $C_n^{(t)} = ((t - 1)n + 1)^{-1} \binom{tn}{n}$. Если между посещениями изменяется h степеней b_j , то $h \geq x$ в $C_{n-x}^{(t)}$ случаях. Так что простой случай осуществляется с вероятностью $1 - C_{n-1}^{(t)} / C_n^{(t)} \approx 1 - (t - 1)^{t-1} / t^t$, и среднее количество установок $b_j \leftarrow 0$ на шаге Т4 составляет $\left(C_{n-1}^{(t)} + \dots + C_1^{(t)} \right) / C_n^{(t)} \approx (t - 1)^{t-1} / (t^t - (t - 1)^{t-1})$, или $4/23$ при $t = 3$.

Можно также изучить t -арную рекурсивную структуру $A_{pq}^{(t)} = 0A_{p(q-1)}^{(t)}, tA_{(p-1)q}^{(t)}$, где $0 \leq (t - 1)p \leq q$, обобщающую (5). Количество таких последовательностей степеней, $C_{pq}^{(t)}$, удовлетворяет рекуррентному соотношению (21) с тем исключением, что $C_{pq}^{(t)} = 0$ при $p < 0$ или $(t - 1)p > q$. Общее решение имеет вид

$$C_{pq}^{(t)} = \frac{q - (t - 1)p + 1}{q + 1} \binom{p + q}{p} = \binom{p + q}{p} - (t - 1) \binom{p + q}{p - 1},$$

и $C_n^{(t)} = C_{n((t-1)n)}^{(t)}$. Треугольник для $t = 3$ начинается следующим образом:

1
1
1 1
1 2
1 3 3
1 4 7
1 5 12 12
1 6 18 30
1 7 25 55 55
1 8 33 88 143

37. Базовое лексикографическое рекуррентное соотношение для всех таких лесов имеет вид

$$A(n_0, n_1, \dots, n_t) = 0A(n_0 - 1, n_1, \dots, n_t), \\ 1A(n_0, n_1 - 1, \dots, n_t), \dots, tA(n_0, n_1, \dots, n_t - 1),$$

где $n_0 > n_2 + 2n_3 + \dots + (t-1)n_t$ и $n_1, \dots, n_t \geq 0$; в противном случае $A(n_0, n_1, \dots, n_t)$ — пустая последовательность, за исключением последовательности $A(0, \dots, 0) = \varepsilon$, состоящей из одной пустой строки. Шаг G1 вычисляет первый элемент $A(n_0, \dots, n_t)$. Мы хотим проанализировать пять величин:

- C — количество выполнений шага G2 (общее количество лесов);
- E — количество переходов от шага G3 к шагу G2 (количество простых случаев);
- K — количество перемещений b_j в список c на шаге G4;
- L — количество сравнений b_j с некоторым c_i на шаге G5;
- Z — количество установок $c_1 \leftarrow 0$ на шаге G5.

Тогда присваивание $b_{l-c_j} \leftarrow c_j$ в цикле на шаге G6 выполняется $K - Z - n_1 - \dots - n_t$ раз.

Пусть n — вектор (n_0, n_1, \dots, n_t) , и пусть e_j — единичный вектор с 1 в позиции j . Пусть $|n| = n_0 + n_1 + \dots + n_t$ и $\|n\| = n_1 + 2n_2 + \dots + tn_t$. Используя эти обозначения, мы можем переписать приведенное выше базовое рекуррентное соотношение в более удобном виде:

$$A(n) = 0A(n - e_0), 1A(n - e_1), \dots, tA(n - e_t) \text{ при } |n| > \|n\|.$$

Рассмотрим общее рекуррентное соотношение

$$F(n) = f(n) + \left(\sum_{j=0}^t F(n - e_j) \right) [|n| > \|n\|],$$

где $F(n) = 0$, если вектор n имеет отрицательный компонент. Если $f(n) = [|n| = 0]$, то $F(n) = C(n)$ — общее количество лесов. Согласно ответу к упражнению 2.3.4.4–32

$$C(n) = \frac{(|n| - 1)! (|n| - \|n\|)}{n_0! n_1! \dots n_t!} = \sum_{j=0}^t (1-j) \binom{|n| - 1}{n_0, \dots, n_{j-1}, n_j - 1, n_{j+1}, \dots, n_t},$$

что является обобщением формулы $C_{pq}^{(t)}$ из упражнения 26 (которая представляет собой случай $n_0 = (t-1)q + 1$ и $n_t = p$). Аналогично: при выборе других функций ядра $f(n)$ мы получаем рекуррентные соотношения для остальных величин $E(n)$, $K(n)$, $L(n)$ и $Z(n)$, необходимые для нашего анализа:

$$\begin{aligned} f(n) = [|n| = n_0 + 1 \text{ и } n_0 > \|n\|] &\quad \text{дает } F(n) = E(n); \\ f(n) = [|n| > n_0] &\quad \text{дает } F(n) = E(n) + K(n); \\ f(n) = [|n| = \|n\| + 1] &\quad \text{дает } F(n) = C(n) + K(n) - Z(n); \\ f(n) = \sum_{1 \leq j < k \leq t} n_j [n_k > 0] &\quad \text{дает } F(n) = L(n). \end{aligned}$$

Символьные методы из упражнения 2.3.4.4–32, похоже, не в состоянии привести к быстрому решению этих более общих рекуррентных соотношений, но можно легко установить значение $C - E$, заметив, что на шаге G2 $b_m + m < N$ тогда и только тогда, когда предыдущим шагом был G3; следовательно,

$$C(n) - E(n) = \sum_{j=1}^t C(n - f_j), \text{ где } f_j = e_j - (j-1)e_0;$$

эта сумма перечисляет все подлеса, в которых $n_1 + \dots + n_t$ — количество внутренних узлов (не являющихся листьями) — уменьшено на 1. Аналогично можно обозначить через

$$C^{(x)}(n) = \sum \{C(n - i_1 f_1 - \dots - i_t f_t) \mid i_1 + \dots + i_t = x\}$$

количество подлесов, имеющих $n_1 + \dots + n_t - x$ внутренних узлов. Тогда мы имеем

$$K(n) - Z(n) = \sum_{x=1}^{|n|} C^{(x)}(n).$$

Эта формула аналогична (20), поскольку $k - [b_j = 0] \geq x \geq 1$ на шаге G5 тогда и только тогда, когда $b_{m-x} > 0$ и $b_{m-x+1} \geq \dots \geq b_m$. Такие строки степеней в прямом порядке обхода взаимно однозначно соответствуют лесам из $C^{(x)}(n)$, если удалить из строки $b_1 \dots b_N$ подстроку $b_{m-x+1} \dots b_m$ и соответствующее количество завершающих нулей.

Из этих формул можно заключить, что алгоритм Закса–Ричардса при $n_1 = n_2 + \dots + n_t + O(1)$ требует только $O(1)$ операций на одно посещение дерева, поскольку $C(n - f_j)/C(n) = n_j n_0^{j-1}/(|n| - 1)^j \leq 1/4 + O(|n|^{-1})$ при $j > 1$. Действительно, значение K достаточно мало почти во всех случаях, представляющих практический интерес. Однако алгоритм может оказаться медленным при больших значениях n_1 . Например, если $t = 1$, $n_0 = m+r+1$ и $n_1 = m$, алгоритм, по сути, вычисляет все r -сочетания $m+r$ элементов; тогда $C(n) = \binom{m+r}{r}$ и $K(n) - Z(n) = \binom{m+r}{r+1} = \Omega(mC(n))$ при фиксированном r . [Для обеспечения эффективности во всех случаях можно отслеживать завершающие единицы; см. Ruskey and Roelants van Baronaigien, *Congressus Numerantium*, 41 (1984), 53–62.]

Точные формулы для K , Z и (в особенности) L , похоже, слишком сложны, но эти величины можно вычислить следующим образом. Назовем “активным блоком” леса крайнюю справа подстроку ненулевых степеней; например, активным блоком 302102021230000000 является 2123. Все перестановки активного блока встречаются одинаково часто. Действительно, обозначим через $D(n)$ сумму величин “Завершающие_нули(β) – 1”, взятую по всем строкам степеней в прямом порядке обхода β для лесов, определяемых n . Тогда блок с n'_j вхождениями j ($1 \leq j \leq t$) является активным ровно в $D(n - n'_1 f_1 - \dots - n'_t f_t) + [n'_1 + \dots + n'_t = n_1 + \dots + n_t]$ случаях. Например, можно в трех местах вставить 21230000 в заданную строку 3021020000 для получения леса с активным блоком 2123. Вклады в K и L при выравнивании активного блока влево (когда перед ним нет ни одного нуля) могут быть вычислены

как в упражнении 7.2.1.2–6, а именно:

$$k(n) = w(e_{n_1}(z) \dots e_{n_t}(z)),$$

$$l(n) = w\left(e_{n_1}(z) \dots e_{n_t}(z) \sum_{1 \leq i < j \leq t} (n_i - zr_i(z)) r_j(z)\right),$$

с использованием обозначений из упомянутого упражнения. Аналогичные вклады получаются и в общем случае; следовательно,

$$K(n) = k(n) + \sum D(n - n') k(n'),$$

$$L(n) = l(n) + \sum D(n - n') l(n'),$$

$$Z(n) = \sum D(n - n'),$$

где суммирование выполняется по всем векторам n' , таким, что $n'_j \leq n_j$ для $1 \leq j \leq t$, и $|n'| - \|n'\| = |n| - \|n\|$ и $n'_1 + \dots + n'_t \leq n_1 + \dots + n_t - 2$.

Остается определить $D(n)$. Пусть $C(n; j)$ означает количество лесов со спецификацией $n = (n_0, \dots, n_t)$, в которых последний внутренний узел в прямом порядке обхода имеет степень j . Тогда

$$C(n) = \sum_{j=1}^t C(n; j) \text{ и } C(n + e_1; 1) = C(n + e_2; 2) = \dots = C(n + e_t; t) = C(n) + D(n).$$

Из этой бесконечной системы линейных уравнений можно вывести, что $C(n) + D(n)$ равно

$$\sum_{i_2=0}^{n_2} \dots \sum_{i_t=0}^{n_t} (-1)^{i_2+\dots+i_t} \binom{i_2 + \dots + i_t}{i_2, \dots, i_t} C(n + (1 + i_2 + \dots + i_t) e_1 - i_2 f_2 - \dots - i_t f_t).$$

Разумеется, хотелось бы получить более простое выражение, если, конечно, оно существует.

38. Очевидно, что шаг L1 выполняет $4n+2$ обращений к памяти. Шаг L3 переходит к шагам L4 или L5 $C_j - C_{j-1}$ раз для конкретного значения j ; следовательно, его стоимость составляет

$$2C_n + 3 \sum_{j=0}^n (n-j)(C_j - C_{j-1}) = 2C_n + 3(C_{n-1} + \dots + C_1 + C_0)$$

обращений к памяти. Шаги L4 и L5 имеют суммарную стоимость $6C_n - 6$ обращений к памяти. Таким образом, весь процесс требует $9 + O(n^{-1/2})$ обращений к памяти на одно посещение.

39. Диаграмма Юнга формы (p, q) и записи y_{ij} соответствуют элементу A_{pq} , который имеют левые скобки в позициях $p + q - 1 - y_{21}, \dots, p + q + 1 - y_{2p}$, и правые скобки в позициях $p + q + 1 - y_{11}, \dots, p + q + 1 - y_{1q}$. Длины уголков равны $\{q + 1, q, \dots, 1, p, p - 1, \dots, 1\} \setminus \{q - p + 1\}$, так что $C_{pq} = (p + q)!(q - p + 1)/((p!(q + 1)!)$ согласно теореме 5.1.4Н.

40. (a) $C_{pq} = \binom{p+q}{p} - \binom{p+q}{p-1} \equiv \binom{p+q}{p} + \binom{p+q}{p-1} = \binom{p+q+1}{p}$ (modulo 2); теперь воспользуйтесь упражнением 1.2.6–11. (b) Из 7.1.3–(36) мы знаем, что $\nu(n \& (n+1)) = \nu(n+1) - 1$.

41. Она равна $C(wz)/(1 - zC(wz)) = 1/(1 - z - wzC(wz)) = (1 - wC(wz))/(1 - w - z)$, где $C(z)$ — производящая функция для чисел Каталана (18). Легко видеть, что первая из этих формул, $C(wz) + zC(wz)^2 + z^2C(wz)^3 + \dots$, эквивалентна (24). [См. P.A. MacMahon, *Combinatory Analysis*, 1 (Cambridge Univ. Press, 1915), 128–130.]

42. (a) Элементы $a_1 \dots a_n$ определяют полностью самосопряженную строку вложенных скобок $a_1 \dots a_{2n}$, причем имеется $C_{q(n-q)}$ вариантов $a_1 \dots a_n$ с q правыми скобками. Таким образом, окончательный ответ:

$$\sum_{q=0}^{\lfloor n/2 \rfloor} C_{q(n-q)} = \sum_{q=0}^{\lfloor n/2 \rfloor} \left(\binom{n}{q} - \binom{n}{q-1} \right) = \binom{n}{\lfloor n/2 \rfloor}.$$

(b) Ровно $C_{(n-1)/2}$ [n нечетно], поскольку самотранспонированное бинарное дерево определяется своим левым поддеревом. Пункт (c) имеет тот же ответ, поскольку лес F является самодуальным тогда и только тогда, когда лес F^R — самотранспонированный.

43. По индукции по $q - p$ получаем

$$C_{pq} = C_q - \binom{q-p-1}{1} C_{q-1} + \dots = \sum_{r=0}^{q-p} (-1)^r \binom{q-p-r}{r} C_{q-r}.$$

44. Количество обращений к памяти между посещениями составляет $3j - 2$ на шаге B3, $h + 1$ на шаге B4 и 4 на шаге B5, где h — количество присваиваний $y \leftarrow r_y$. Количество бинарных деревьев, у которых $h \geq x$, для заданных j и x равно $[z^{n-j-x-1}] C(z)^{x+3}$ при $j < n$, поскольку такие деревья получаются путем присоединения $x + 3$ поддеревьев ниже $j + x + 1$ внутренних узлов. Присваивая $x = 0$ и используя формулу (24) и упражнение 43, мы находим, что данное значение j встречается $C_{(n-j-1)(n-j+1)} = C_{n+1-j} - C_{n-j}$ раз. Таким образом, $\sum j$ по всем бинарным деревьям равна $n + \sum_{j=1}^n (C_{n+1-j} - C_{n-j}) j = C_n + C_{n-1} + \dots + C_1$. Аналогично этому, $\sum (h + 1)$ равна

$$\begin{aligned} \sum_{j=1}^{n-1} \sum_{x=0}^{n-j-1} C_{(n-j-x-1)(n-j+1)} &= \sum_{j=1}^{n-1} C_{(n-j-1)(n-j+2)} = \sum_{j=1}^n (C_{n-j+2} - 2C_{n-j+1}) = \\ &= C_{n+1} - (C_n + C_{n-1} + \dots + C_0). \end{aligned}$$

Таким образом, общая стоимость алгоритма составляет

$$C_{n+1} + 4C_n + 2(C_{n-1} + \dots + C_1) + O(n) = (26/3 - 10/(3n) + O(n^{-2})) C_n$$

обращений к памяти.

45. Каждый из простых случаев на шаге K3 встречается C_{n-1} раз, так что общая стоимость этого шага составляет $3C_{n-1} + 8C_{n-1} + 2(C_n - 2C_{n-1})$ обращений

к памяти. Выборка r_i на шаге K4 выполняется $[z^{n-i-1}] C(z)^{i+2} = C_{(n-i-1)n}$ раз; суммирование для $i \geq 2$ дает общее количество обращений к памяти в этом цикле, равное $C_{(n-3)(n+1)} = C_{n+1} - 3C_n + C_{n-1}$. Стоимость шага K5 — $6C_n - 12C_{n-1}$. Шаг K6 немного сложнее, но можно показать, что операция $r_j \leftarrow r_k$ выполняется $C_n - 3C_{n-1} + 1$ раз при $n > 2$, а операция $r_j \leftarrow 0$ выполняется $C_{n-1} - n + 1$ раз. Таким образом, общее количество обращений к памяти — $C_{n+1} + 7C_n - 9C_{n-1} + n + 3 = (8.75 - 9.375/n + O(n^{-2})) C_n$.

Хотя это общее количество асимптотически хуже, чем у алгоритма B в ответе к упражнению 44, большой отрицательный коэффициент при n^{-1} означает, что в действительности алгоритм B выигрывает только при $n \geq 58$; такие большие значения n на практике не встречаются.

46. (a) Движение влево от (pq) увеличивает площадь на $q - p$.

(b) Шаги влево на пути от (nn) до (00) соответствуют левым скобкам в $a_1 \dots a_{2n}$, и мы имеем на таком k -м шаге $q - p = c_k$.

(c) Эквивалентно, $C_{n+1}(x) = \sum_{k=0}^n x^k C_k(x) C_{n-k}(x)$. Это рекуррентное соотношение выполняется, поскольку $(n+1)$ -узловой лес F состоит из корня крайнего слева дерева с k -узловым лесом F_l (потомками этого корня) и $(n-k)$ -узловым лесом F_r (остальные деревья) и поскольку

$$\begin{aligned} \text{длина внутреннего пути } (F) &= k + \text{длина внутреннего пути } (F_l) + \\ &\quad + \text{длина внутреннего пути } (F_r). \end{aligned}$$

(d) Строки в $A_{p(p+r)}$ имеют вид $\alpha_0(\alpha_1) \dots (\alpha_{r-1})\alpha_r$, где каждое α_j представляет собой подстроку корректно вложенных скобок. Площадь такой строки равна сумме по всем j площадей α_j плюс $r - j$, умноженное на количество левых скобок в α_j .

Примечание: полиномы $C_{pq}(x)$ были представлены Л. Карлицем (L. Carlitz) и Д. Риорданом (J. Riordan) в *Duke Math. J.*, **31** (1964), 371–388; тождество в условии (d) эквивалентно их формуле (10.12). Они также доказали, что

$$C_{pq}(x) = \sum_r (-1)^r x^{r(r-1)-\binom{q-p}{2}} \binom{q-p-r}{r}_x C_{q-r}(x);$$

Это обобщает результат упражнения 43. Из (c) мы получаем цепную дробь $C(x, z) = 1/(1 - z/(1 - xz/(1 - x^2z/(1 - \dots))))$, для которой Д.Н. Ватсон (G.N. Watson) доказал ее равенство $F(x, z)/F(x, z/x)$, где

$$F(x, z) = \sum_{n=0}^{\infty} \frac{(-1)^n x^{n^2} z^n}{(1-x)(1-x^2)\dots(1-x^n)}$$

[см. *J. London Math. Soc.*, **4** (1929), 39–48]. Мы уже встречались с этой производящей функцией в несколько измененном виде в упражнении 5.2.1–15.

Длина внутреннего пути леса представляет собой “длину левого пути” соответствующего бинарного дерева, а именно сумму количества левых ветвлений на пути от корня по всем внутренним узлам. Более общий полином

$$C_n(x, y) = \sum x^{\text{длина левого пути}(T)} y^{\text{длина правого пути}(T)},$$

где сумма берется по всем n -узловым бинарным деревьям T , похоже, не имеет простого аддитивного рекуррентного соотношения наподобие соотношения для

$C_{nn}(x) = C_n(x, 1)$, рассматривавшегося в данном упражнении; однако мы имеем $C_{n+1}(x, y) = \sum_k x^k C_k(x, y) y^{n-k} C_{n-k}(x, y)$. Таким образом, сверхпроизводящая функция $C(x, y, z) = \sum_n C_n(x, y) z^n$ удовлетворяет функциональному уравнению $C(x, y, z) = 1 + zC(x, y, xz)C(x, y, yz)$. (Случай $x = y$ рассматривался в упражнении 2.3.4.5–5.)

47. $C_n(x) = \sum_q x^{\binom{q-p}{2}} C_{pq}(x) C_{(n-q)(n-1-p)}(x)$ для $0 \leq p < n$.

48. Пусть $\bar{C}(z) = C(-1, z)$ с использованием обозначений из упражнения 46 и пусть $\bar{C}(z)\bar{C}(-z) = F(z^2)$. Тогда $\bar{C}(z) = 1 + zF(z^2)$ и $\bar{C}(-z) = 1 - zF(z^2)$; так что $F(z) = 1 - zF(z^2)$ и $F(z) = C(-z)$. Отсюда следует, что

$$C_{pq}(-1) = [z^p] C(-z^2)^{\lceil (q-p)/2 \rceil} (1 + zC(-z^2))^{\lfloor q-p \text{ четно} \rfloor};$$

при четном q это выражение равно $(-1)^{(p/2)} C_{(p/2)(q/2-1)}$ [p четно], а при нечетном — $(-1)^{\lfloor p/2 \rfloor} C_{\lfloor p/2 \rfloor \lfloor q/2 \rfloor}$. Идеальный код Грэя для строк A_{pq} может существовать, только если $|C_{pq}(-1)| \leq 1$, поскольку соответствующий граф двудольный (см. рис. 41); $|C_{pq}(-1)|$ представляет собой разность между размерами частей, поскольку каждый идеальный переход изменяет $c_1 + \dots + c_n$ на ± 1 .

49. Согласно алгоритму U при $n = 15$ и $N = 10^6$ это строка

$((((())(((())(((((())(()))))))$.

50. Внесите следующие изменения в алгоритм U. К шагу U1 добавьте присваивание $r \leftarrow 0$. На шаге U3 проверку $N \leq c'$ замените проверкой $a_m =)$. На шаге U4 вместо $N \leftarrow N - c'$ выполните установку $r \leftarrow r + c'$. Кроме того, опустите присваивание a_m на шагах U3 и U4.

Строка в (1) имеет ранг 3 141 592 (кто бы мог подумать?).

51. По теореме 7.2.1.3L $N = \binom{\bar{z}_1}{n} + \binom{\bar{z}_2}{n-1} + \dots + \binom{\bar{z}_n}{1}$; следовательно, $\kappa_n N = \binom{\bar{z}_1}{n-1} + \binom{\bar{z}_2}{n-2} + \dots + \binom{\bar{z}_n}{0}$, поскольку $\bar{z}_n \geq 1$. Теперь заметим, что $N - \kappa_n N$ представляет собой ранг $z_1 z_2 \dots z_n$, согласно формуле (23) и упражнению 50. (Пусть, например, $z_1 \dots z_4 = 1256$ с рангом 6 в табл. 1. Тогда $\bar{z}_1 \dots \bar{z}_4 = 7632$, $N = 60$ и $\kappa_4 60 = 54$. Заметим, что N достаточно велико, поскольку $\bar{z}_1 = 2n - 1$; из рис. 27 видно, что $\kappa_n N$ обычно превышает N при малых N .)

52. Количество завершающих правых скобок имеет то же распределение, что и количество ведущих левых скобок, и последовательность строк вложенных скобок, которая начинается с $\binom{k}{k} - \binom{k}{k} A_{(n-k)(n-1)}$. Следовательно, вероятность того, что $d_n = k$, равна $C_{(n-k)(n-1)}/C_n$. Используя 1.2.6–(25), можно найти, что

$$\begin{aligned} \sum_{k=0}^n \binom{k}{t} C_{(n-k)(n-1)} &= \sum_{k=0}^n \left(\binom{2n-1-k}{n-1} - \binom{2n-1-k}{n} \right) \binom{k}{t} = \\ &= \binom{2n}{n+t} - \binom{2n}{n+t+1} = C_{(n-t)(n+t)}, \end{aligned}$$

отсюда следует, что среднее значение и дисперсия равны соответственно $3n/(n+2) = 3 - 6/(n+2)$ и $2n(2n^2 - n - 1)/((n+2)^2(n+3)) = 4 + O(n^{-1})$. [Моменты этого распределения были впервые вычислены Р. Кемпом (R. Kemp) в *Acta Informatica*, 35 (1998), 17–89, теорема 9. Заметим, что $c_n = d_n - 1$ имеет, по сути, то же поведение.]

53. (a) Согласно упражнению 52 — $3n/(n+2)$.

(b) В соответствии с упражнением 6.2.2–7 математическое ожидание равно H_n .

(c) По индукции можно найти, что математическое ожидание равно $2 - 2^{-n}$.

(d) Любая (фиксированная) последовательность левых и правых ветвлений имеет одно и то же распределение шагов перед тем, как будет встречен лист (другими словами, вероятность встретить узел с бинарным обозначением Дьюи 01101 равна вероятности встретить узел 00000). Таким образом, если $X = k$ с вероятностью p_k , каждый из 2^k потенциальных узлов на уровне k является внешним с вероятностью p_k . Математическое ожидание $\sum_k 2^k p_k$, таким образом, представляет собой математическое ожидание количества внешних узлов, а именно $n+1$ во всех трех случаях. (Этот результат можно проверить непосредственно, с учетом того, что $p_k = C_{(n-k)(n-1)}/C_n$ в случае (a); $p_k = [n]/n!$ в случае (b); $p_k = 2^{-k+[k=n]}$ в случае (c)).

Примечание: средняя длина пути в указанных трех случаях равна $\Theta(\sqrt{n})$, $\Theta(\log n)$ и $\Theta(n)$ соответственно; таким образом, этот путь оказывается более длинным при более коротком среднем пути к листу. Это поясняется тем, что вездесущие “дыры” возле корня приводят к удлинению других путей. Случай (a) имеет интересное обобщение для t -арных деревьев, для которых $p_k = C_{(n-k)((t-1)n-1)}^{(t)}/C_n^{(t)}$ с использованием обозначений из упражнения 36. Тогда среднее расстояние до листа равно $(t+1)n/((t-1)n+2)$, и очень поучительно доказать с использованием телескопических рядов, что

$$\sum_k t^k C_{(n-k)((t-1)n-1)}^{(t)} = \binom{tn}{n}.$$

54. Дифференцируя по x , получим

$$C'(x, z) = zC'(x, z)C(x, xz) + zC(x, z)(C'(x, xz) + zC'(x, xz)),$$

где $C'(x, z)$ обозначает производную $C(x, z)$ по z . Таким образом, $C'(1, z) = 2zC'(1, z)C(z) + z^2C(z)C'(z)$; а поскольку $C'(z) = C(z)^2 + 2zC(z)C'(z)$, можно найти решение для $C'(1, z)$, получая $z^2C(z)^3/(1 - 2zC(z))^2$. Следовательно, $\sum(c_1 + \dots + c_n) = [z^n]C'(1, z) = 2^{2n-1} - \frac{1}{2}(3n+1)C_n$, что согласуется с упражнением 2.3.4.5–5. Аналогично находим

$$\sum(c_1 + \dots + c_n)^2 = [z^n]C''(1, z) = \left(\frac{5n^2 + 19n + 6}{6}\right)\binom{2n}{n} - \left(1 + \frac{3n}{2}\right)4^n.$$

Таким образом, математическое ожидание и дисперсия равны соответственно $\frac{1}{2}\sqrt{\pi}n^{3/2} + O(n)$ и $\left(\frac{5}{6} - \frac{\pi}{4}\right)n^{3/2} + O(n)$.

55. Дифференцируя, как в упражнении 54, и используя формулы из упражнения 46 (d) и 5.2.1–14 вместе с $[z^n] C(z)^r / (1 - 4z) = 2^{2n+r} - \sum_{j=1}^r 2^{r-j} \binom{2n+j}{n}$, получаем

$$\begin{aligned} C'_{p(p+r)}(1) &= [z^p] \left((r+1) \frac{z^2 C(z)^{r+3}}{1-4z} + \binom{r+1}{2} \frac{z C(z)^{r+2}}{\sqrt{1-4z}} \right) = \\ &= [z^p] \left((r+1) \frac{C(z)^{r+1} - 2C(z)^r + C(z)^{r-1}}{1-4z} + \binom{r+1}{2} \frac{C(z)^{r+1} - C(z)^r}{\sqrt{1-4z}} \right) = \\ &= (r+1) \left(2^{2p+r-1} - \binom{2p+r+1}{p} - \sum_{j=1}^{r-1} 2^{r-1-j} \binom{2p+j}{p} \right) + \binom{r+1}{2} \binom{2p+r}{p-1}. \end{aligned}$$

56. Используйте упражнение 1.2.6–53 (b). [См. *BIT*, **30** (1990), 67–68.]

57. $2S_0(a, b) = \binom{2a}{a} \binom{2b}{b} + \binom{2a+2b}{a+b}$, согласно 1.2.6–(21). Упражнение 1.2.6–53 гласит, что

$$\sum_{k=a-m}^a \binom{2a}{a-k} \binom{2b}{b-k} k = (m+1)(a+b-m) \binom{2a}{m+1} \binom{2b}{a+b-m};$$

следовательно, $2S_1(a, b) = \binom{2a}{a} \binom{2b}{b} \frac{ab}{a+b}$. Поскольку $b^2 S_p(a, b) - S_{p+2}(a, b) = S_p(a, b-1)$, находим, что $2S_2(a, b) = \binom{2a+2b}{a+b} \frac{ab}{2a+2b-1}$; $2S_3(a, b) = \binom{2a}{a} \binom{2b}{b} a^2 b^2 / (a+b)^2$. Формула (30) получается путем подстановки $a = m$, $b = n - m$ и $C_{(x-k)(x+k)} = \binom{2x}{x-k} - \binom{2x}{x-k-1}$.

Аналогично: среднее значение w_{2m-1} равно

$$\sum_{k \geq 0} (2k-1) C_{(m-k)(m+k-1)} C_{(n-m-k+1)(n-m+k)},$$

деленному на C_n , или

$$\begin{aligned} \frac{2S_3(m, n+1-m) - S_2(m, n+1-m)}{m(n+1-m)C_n} &= \\ &= \frac{m(n+1-m)}{n} \binom{2m}{m} \binom{2n+2-2m}{n+1-m} \Big/ \binom{2n}{n} - 1. \end{aligned}$$

[R. Kemp, *BIT*, **20** (1980), 157–163; H. Prodinger, *Soochow J. Math.*, **9** (1983), 193–196.]

58. Суммируя по всем случаям, когда левое поддерево содержит k внутренних узлов, получаем

$$t_{lmn} = [l = m = n = 0] + \sum_{k=0}^{m-1} C_k t_{(l-1)(m-k-1)(n-k-1)} + \sum_{k=m}^{n-1} C_{n-1-k} t_{(l-1)mk}.$$

Таким образом, тройная производящая функция $t(v, w, z) = \sum_{l,m,n} t_{lmn} v^l w^m z^n$ удовлетворяет уравнению

$$t(v, w, z) = 1 + vwzC(wz)t(v, w, z) + vzC(z)t(v, w, z).$$

Мы получаем также аналогичное линейное соотношение для $t(w, z) = \partial t(v, w, z) / \partial v|_{v=1}$, поскольку $t(1, w, z) = \sum_{n=0}^{\infty} \sum_{m=0}^n C_n w^m z^n = (C(z) - wC(wz)) / (1 - w)$ и $zC(z)^2 = C(z) - 1$. Алгебраические преобразования дают нам

$$t(w, z) = \frac{C(z) + wC(wz) - (1 + w)}{(1 - w)^2 z} - \frac{2wC(z)C(wz)}{(1 - w)^2} - \frac{C(z) - wC(wz)}{1 - w},$$

и мы получаем формулу $t_{mn} = (m + 1)C_{n+1} - 2 \sum_{k=0}^m (m - k)C_k C_{n-k} - C_n$. Теперь доказать, что

$$\sum_{k=0}^{m-1} (k + 1)C_k C_{n-1-k} = \frac{m}{2n} \binom{2m}{m} \binom{2n - 2m}{n - m},$$

можно так же, как в упражнении 56; отсюда следует, что

$$t_{mn} = 2 \binom{2m}{m} \binom{2n - 2m}{n - m} \frac{(2m + 1)(2n - 2m + 1)}{(n + 1)(n + 2)} - C_n \text{ для } 0 \leq m \leq n.$$

[P. Kirschenhofer, *J. Combinatorics, Information and System Sciences*, **8** (1983), 44–60. Информация о более высоких моментах и обобщениях может быть найдена в работах W.J. Gutjahr, *Random Structures and Algorithms*, **3** (1992), 361–374; A. Panholzer and H. Prodinger, *J. Statistical Planning and Inference*, **101** (2002), 267–279. Обратите внимание, что производящая функция $t(v, w, z)$ дает

$$t_{lmn} = \sum_k \binom{l}{k} C_{(m-k)(m-1)} C_{(n-m-l+k)(n-m-1)}.$$

Используя тот факт, что $\sum_k \binom{k}{r} C_{(n-k)(m-1)} = C_{(n-r)(m+r)}$ при $m \geq 1$, получаем формулу $t_{mn} + C_n = \sum_k (k + 1)C_{(m-k)(m-1)}C_{(n-m)(n-m+k+1)}$, сумма в которой, следовательно, может быть записана в аналитическом виде.]

59.

$$\begin{aligned} T(w, z) &= \frac{w(C(z) - C(wz))}{(1 - w)} - wzC(z)C(wz) + zC(z)T(w, z) + wzC(wz)T(wz) = \\ &= \frac{w((C(z) + C(wz) - 2)/z - (1 + w)C(z)C(wz) - (1 - w)(C(z) - C(wz)))}{(1 - w)^2}. \end{aligned}$$

Следовательно, $T_{mn} = t_{mn} - \sum_{k=m}^n C_k C_{n-k}$. [Является ли данное доказательство комбинаторным?] Итак,

$$T_{mn} = \binom{2m}{m} \binom{2n + 2 - 2m}{n + 1 - m} \frac{4m(n + 1 - m) + n + 1}{2(n + 1)(n + 2)} - \frac{1}{2}C_{n+1} - C_n, \text{ при } 1 \leq m \leq n.$$

60. (a) Это количество правых скобок в соатомах. (Следовательно, это также количество k , для которых $w_{2k-1} < 0$ в соответствующем “обходе червя”.)

(b) Для удобства положим $d('') = +1$ и $d('') = -1$.

A1. [Инициализация.] Установить $i \leftarrow j \leftarrow 1$ и $k \leftarrow 2n$.

A2. [Выполнено?] Завершить работу алгоритма, если $j > k$. В противном случае установить $a_j \leftarrow '(', j \leftarrow j + 1$.

A3. [Атом?] Если $b_i = ')'$, установить $s \leftarrow -1, i \leftarrow i + 1$ и перейти к шагу A4. В противном случае установить $s \leftarrow 1, i \leftarrow i + 1$ и, пока $s > 0$, устанавливать $a_j \leftarrow b_i, j \leftarrow j + 1, s \leftarrow s + d(b_i), i \leftarrow i + 1$. Вернуться к шагу A2.

A4. [Соатом.] Установить $s \leftarrow s + d(b_i)$. Затем, если $s < 0$, установить $a_k \leftarrow b_i, k \leftarrow k - 1, i \leftarrow i + 1$ и повторить шаг A4. В противном случае установить $a_k \leftarrow ')', k \leftarrow k - 1, i \leftarrow i + 1$ и вернуться к шагу A2. ■

(с) Сбалансированная строка с дефектом 11, отображающаяся на строку (1) — ((())(((())(()))(((((())((()(((). В общем случае мы находим такую строку, определяя индекс m непосредственно перед l -й с конца правой скобкой и индексы $(u_0, v_0), \dots, (u_{s-1}, v_{s-1})$ соответствующих друг другу скобок, такие, что $u_j \leq m < v_j$.

I1. [Инициализация.] Установить $c \leftarrow j \leftarrow s \leftarrow 0, k \leftarrow m \leftarrow 2n$ и $u_0 \leftarrow 2n + 1$.

I2. [Сканирование справа налево.] Если $a_k = ')'$, перейти к шагу I3; если $a_k = '('$, перейти к шагу I4; если $k = 0$, перейти к шагу I5.

I3. [Обработка ''].] Установить $r_j \leftarrow k, j \leftarrow j + 1, c \leftarrow c + 1$. Если $c = l$, установить $m \leftarrow k - 1, s \leftarrow j$ и $u_s \leftarrow k$. Затем уменьшить k на 1 и вернуться к шагу I2.

I4. [Обработка ''].] (В этот момент левая скобка a_k соответствует правой скобке $a_{r_{j-1}}$.) Установить $j \leftarrow j - 1$. Если $r_j > m$, установить $u_j \leftarrow k$ и $v_j \leftarrow r_j$. Затем уменьшить k на 1 и вернуться к шагу I2.

I5. [Подготовка к перестановке.] Установить $i \leftarrow j \leftarrow 1, k \leftarrow 2n$ и $c \leftarrow 0$.

I6. [Перестановка.] Пока $j \neq u_c$, устанавливать $b_i \leftarrow a_j, i \leftarrow i + 1, j \leftarrow j + 1$. Затем, если $c = s$, завершить работу алгоритма. В противном случае установить $b_i \leftarrow ')', i \leftarrow i + 1, j \leftarrow j + 1$. Пока $k \neq v_c$, устанавливать $b_i \leftarrow a_k, i \leftarrow i + 1$ и $k \leftarrow k - 1$. Затем установить $b_i \leftarrow '(', i \leftarrow i + 1, k \leftarrow k - 1, c \leftarrow c + 1$ и повторить шаг I6. ■

Примечание: тот факт, что дефект l ($0 \leq l \leq n$) имеют ровно C_n сбалансированных строк длиной $2n$, был открыт П.А. Мак-Мэганом (P.A. MacMahon) [*Philosophical Transactions*, **209** (1909), 153–175, §20], а затем повторно открыт К.Л. Чангом (K.L. Chung) и В. Феллером (W. Feller) [*Proc. Nat. Acad. Sci.*, **35** (1949), 605–608] с применением производящих функций. Простое комбинаторное пояснение было найдено позже Д.Л. Ходжесом (J.L. Hodges) [*Biometrika*, **42** (1955), 261–262], который заметил, что если $\beta_1 \dots \beta_r$ имеет дефект $l > 0$ и если $\beta_k = \alpha_k^R$ — ее крайний справа соатом, то сбалансированная строка $\beta_1 \dots \beta_{k-1} (\beta_{k+1} \dots \beta_r) \alpha_k'^R$ имеет дефект $l - 1$ (и это преобразование обратимо). Эффективное отображение в данном упражнении похоже на конструкцию М.Д. Аткинсона (M.D. Atkinson) и Ё.-Р. Сака (J.-R. Sack) [*Information Processing Letters*, **41** (1992), 21–23].

61. (а) Пусть $c_j = 1 - b_j$; тогда $c_j \leq 1$, $c_1 + \dots + c_N = f$ и мы должны доказать, что утверждение

$$c_1 + c_2 + \dots + c_k < f \text{ тогда и только тогда, когда } k < N,$$

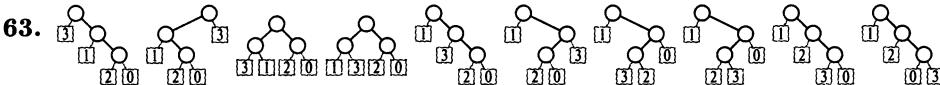
справедливо ровно при f циклических сдвигах. Можно определить c_j для всех целых j , полагая $c_{j+N} = c_j$. Определим также Σ_j для всех j , полагая $\Sigma_0 = 0$ и $\Sigma_j = \Sigma_{j-1} + c_j$; тогда $\Sigma_{j+Nt} = \Sigma_j + ft$ и $\Sigma_{j+1} \leq \Sigma_j + 1$. Отсюда следует, что для каждого целого x существует наименьшее целое $j = j(x)$, такое, что $\Sigma_j = x$. Кроме того, $j(x) < j(x+1)$ и $j(x+f) = j(x) + N$. Таким образом, интересующее нас условие выполняется тогда и только тогда, когда мы выполняем сдвиг на $j(x) \bmod N$ (где $x = 1, 2, \dots$), или на f . (История этой важной леммы рассматривается в ответе к упражнению 2.3.4.4–32.)

(б) Начнем с $l \leftarrow m \leftarrow s \leftarrow 0$. Затем для $k = 1, 2, \dots, N$ (в указанном порядке) выполним следующие действия: установим $s \leftarrow s + 1 - b_k$ и, если $s > m$, установим $m \leftarrow s$, $j_l \leftarrow k$ и $l \leftarrow (l+1) \bmod f$. Ответом, согласно доказательству из ответа (а), будет j_0, \dots, j_{f-1} .

(с) Начнем с произвольной строки $b_1 b_2 \dots b_N$, содержащей n_j значений j , $0 \leq j \leq t$. Применим случайную перестановку к этой строке, после чего применим алгоритм из ответа (б). Затем случайным образом выберем значение из (j_0, \dots, j_{f-1}) и используем результат циклического сдвига в качестве последовательности в прямом порядке обхода, определяющей лес.

[См. L. Alonso, J.L. Rémy and R. Schott, *Algorithmica*, **17** (1997), 162–182, где приведен еще более общий алгоритм.]

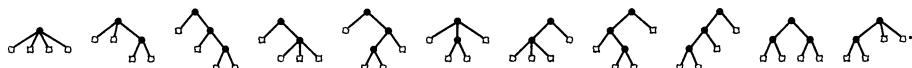
62. Битовые строки $(l_1 \dots l_n, r_1 \dots r_n)$ корректны тогда и только тогда, когда строка $b_1 \dots b_n$, где $b_j = l_j + r_j$, корректна в упражнении 20. Следовательно, можно воспользоваться упражнением 61. [См. J.F., *Information Processing Letters*, **45** (1993), 291–294.]



64. $X = 2k + b$, где $(k, b) = (0, 1), (2, 1), (0, 0), (5, 1), (6, 0), (1, 1)$; в конце $L_0 L_1 \dots L_{12} = 5 \ 11 \ 3 \ 4 \ 0 \ 7 \ 9 \ 8 \ 1 \ 6 \ 10 \ 12 \ 2$.

65. См. A. Panholzer and H. Prodinger, *Discrete Mathematics*, **250** (2002), 181–195; M. Luczak and P. Winkler, *Random Structures and Algorithms*, **24** (2004), 420–443.

66. (а) “Сожмем” белые ребра, объединяя соединяемые ими узлы. Например, одиннадцать изображенным деревьям Шрёдера для $n = 3$ будут соответствовать обычные деревья



При таком соответствии левая связь означает “это дочерний узел”, белая правая связь — “есть и другие дочерние узлы”, а черная правая связь — “это последний дочерний узел”.

(б) Имитируем алгоритм L, но между поворотами используем обычный код Грея для прохода по всем цветовым шаблонам имеющихся правых связей (в этом упражнении данная последовательность проиллюстрирована для случая $n = 3$).

Заметим, что деревья Шрёдера также соответствуют последовательно-параллельным графам, как в дереве из (53). Однако при этом налагается определенный порядок на ребра и/или сверхребра, соединенные параллельно; так что более точно говорить о соответствии последовательно-параллельным графикам, *уложенными на плоскости* (с непомеченными ребрами и вершинами, за исключением s и t).

67. $S(z) = 1 + zS(z)(1 + 2(S(z) - 1))$, поскольку $1 + 2(S(z) - 1)$ подсчитывает правые поддеревья; таким образом, искомая функция $S(z) = (1 + z - \sqrt{1 - 6z + z^2})/(4z)$.

Примечание: мы уже встречались с числами Шрёдера в упражнении 2.3.4.4–31, где $G(z) = zS(z)$; а также в упражнении 2.2.1–11, где $b_n = 2S_{n-1}$ при $n \geq 2$ и где было найдено рекуррентное соотношение $(n-1)S_n = (6n-3)S_{n-1} - (n-2)S_{n-2}$. Асимптотический рост чисел Шрёдера исследован в упражнении 2.2.1–12. Треугольный массив чисел S_{pq} , аналогичный (22), может использоваться для генерации случайных чисел Шрёдера. Эти числа удовлетворяют соотношению

$$\begin{aligned} S_{pq} &= S_{p(q-1)} + S_{(p-1)q} + S_{(p-2)q} + \cdots + S_{0q} = S_{p(q-1)} + 2S_{(p-1)q} - S_{(p-1)(q-1)} = \\ &= \frac{q-p+1}{q+1} \sum_{k=0}^p \binom{q+1}{p-k} \binom{p-1}{k} 2^k = \\ &= \sum_{k=0}^p \left(\binom{q}{p-k} \binom{p-1}{k} - \binom{q}{p-k-1} \binom{p-1}{k-1} \right) 2^k = \\ &= [w^p z^q] S(wz)/(1 - zS(wz)); \end{aligned}$$

двойная производящая функция в последней строке открыта Э. Дойчем (E. Deutsch). Многие другие свойства деревьев Шрёдера рассматриваются в книге Р. Стенли (R. Stanley) [*Enumerative Combinatorics*, 2 (1999), упражнение 6.39].

68. Единственная строка, содержащая пустую битовую строку ε . (Общее правило (36) для перехода от порядка $n-1$ к порядку n преобразует эту строку в '0 1', шаблон порядка 1.)

69. Первые $\binom{6}{3} = 20$ строк представляют собой шаблон рождественской елки порядка 6, если проигнорировать '10' в начале каждой строки. Шаблон порядка 7 увидеть немного сложнее, однако существует $\binom{7}{3} = 35$ строк, в которых крайняя слева запись начинается с 0. Игнорируйте во всех таких строках крайнюю справа битовую строку и 0 в начале каждой остающейся битовой строки. (Есть и другие ответы на данный вопрос.)

70. Если σ находится в столбце k шаблона рождественской елки, то σ' находится в столбце $n-k$ в той же строке. (Если вместо битов рассматривать скобки, то это правило дает зеркальное отображение свободных скобок в смысле ответа к упражнению 11, согласно (39).)

71. M_{tn} представляет собой сумму t наибольших биномиальных коэффициентов $\binom{n}{k}$, поскольку каждая строка шаблона рождественской елки может содержать не более t элементов S и поскольку мы получаем такое множество S выбором всех строк σ , у которых $(n-t)/2 \leq \nu(\sigma) \leq (n+t-1)/2$. (Формула

$$M_{tn} = \sum_{n-t \leq 2k \leq n+t-1} \binom{n}{k}$$

проста, насколько это возможно; однако для малых t имеются более простые формулы, наподобие $M_{(2)n} = M_{n+1}$, а кроме того, $M_{tn} = 2^n$ для $t > n$.)

72. Мы получим M_{sn} , то же число, что и в предыдущем упражнении. В действительности по индукции можно доказать, что существует ровно $\binom{n}{n-k} - \binom{n}{k-s}$ строк длиной $s + n - 2k \geq 0$.

73. 011001001000000000100101001100, 11100101101111111101101011100; см. (38).

74. Согласно лексикографическому порядку, нужно подсчитать количество строк, крайние правые элементы которых имеют вид 0^{*29} , 10^{*28} , 110^{*27} , 111000^{*24} , 11100100^{*22} , 111001010^{*21} , 11100101100^{*19} , 111001011010^{*18} , 1110010110110^{*17} , ..., т.е. всех 30-битовых строк, предшествующих $\tau = 1110010110111111101101011100$.

Если θ имеет на p больше единиц, чем нулей, то количество строк шаблона рождественской елки, завершающихся битовой строкой θ^{*n} , равно количеству строк, завершающихся $1^p * n$, и в соответствии с упражнением 71 равно $M_{(p+1)n}$, так как все такие строки являются потомками начальной строки ' $0^p 0^{p-1} 1 \dots 1^p$ ' на n -м шаге.

Следовательно, ответ на поставленный вопрос $M_{0(29)} + M_{1(28)} + M_{2(27)} + M_{1(24)} + \dots + M_{(12)3} + M_{(13)2} = \sum_{k=1}^{21} M_{(2k-1-z_k)(n-z_k)} = 0 + \binom{28}{14} + \binom{27}{14} + \binom{27}{13} + \binom{24}{12} + \dots + 8 + 4 = 84\,867\,708$, где $(z_1, \dots, z_{21}) = (1, 2, 3, 6, \dots, 27, 28)$ — последовательность позиций, в которых в τ встречается 1.

75. $r_1^{(n)} = M_{n-2}$, поскольку строка $r_1^{(n)}$ представляет собой нижнего потомка первой строки в (33). Кроме того, $r_{j+1}^{(n)} - r_j^{(n)} = M_{j(n-1-j)} - M_{(j-1)(n-2-j)} = M_{(j+1)(n-2-j)}$ в соответствии с формулой из ответа к упражнению 74, поскольку соответствующая последовательность $z_1 \dots z_{n-1}$ для строки $r_j^{(n)} - 1^j 0^{n-1-j}$. Следовательно, поскольку $M_{jn}/M_n \rightarrow j$ для фиксированного значения j при $n \rightarrow \infty$, мы получаем

$$\lim_{n \rightarrow \infty} \frac{r_j^{(n)}}{M_n} = \sum_{k=1}^j \frac{k}{2^{k+1}} = 1 - \frac{j+2}{2^{j+2}}.$$

Мы также неявно доказали, что $\sum_{k=0}^n M_{k(n-k)} = M_{n+1} - 1$.

76. Первые $\binom{2n}{n}$ элементов бесконечной последовательности

$$Q = 1313351313351335355713133513133513353557131335133535571335355735575779\dots$$

представляют собой размеры строк в шаблоне порядка $2n$; эта последовательность $Q = q_1 q_2 q_3 \dots$ является единственной фиксированной точкой преобразования, отображающего $1 \mapsto 13$ и $n \mapsto (n-2)nn(n+2)$ для нечетных $n > 1$, представляющего собой два шага из (36).

Пусть $f(x) = \limsup_{n \rightarrow \infty} s(\lceil xM_n \rceil)/n$ при $0 < x \leq 1$. Эта функция, по-видимому, почти везде стремится к нулю; однако она равна 1, когда x имеет вид $(q_1 + \dots + q_j)/2^n$, в соответствии с ответом к упражнению 72. С другой стороны, если мы определим $g(x) = \lim_{n \rightarrow \infty} s(\lceil xM_n \rceil)/\sqrt{n}$, то функция $g(x)$ является измеримой, с $\int_0^1 g(x) dx = \sqrt{\pi}$, хотя $g(x)$ бесконечна, когда $f(x) > 0$. (Строгое доказательство или опровержение этих гипотез пока отсутствует.)

77. Указание следует из (39) при рассмотрении обхода червя; так что мы можем действовать следующим образом.

- X1.** [Инициализация.] Установить $a_j \leftarrow 0$ для $0 \leq j \leq n$; установить также $x \leftarrow 1$.
 (На последующих шагах $x = 1 + 2(a_1 + \dots + a_n)$.)
- X2.** [Коррекция хвоста.] Пока $x \leq n$, устанавливать $a_x \leftarrow 1$ и $x \leftarrow x + 2$.
- X3.** [Посещение.] Посетить битовую строку $a_1 \dots a_n$.
- X4.** [Простой случай?] Если $a_n = 0$, установить $a_n \leftarrow 1$, $x \leftarrow x + 2$ и вернуться к шагу X3.
- X5.** [Поиск и продвижение a_j .] Установить $a_n \leftarrow 0$ и $j \leftarrow n - 1$. Затем, пока $a_j = 1$, устанавливать $a_j \leftarrow 0$, $x \leftarrow x - 2$ и $j \leftarrow j - 1$. Завершить работу, если $j = 0$; в противном случае установить $a_j \leftarrow 1$ и вернуться к шагу X2. ■

78. Истинно согласно (39) и упражнению 11.

79. (а) Перечислите индексы нулей, затем индексы единиц; например, битовая строка из упражнения 73 соответствует перестановке

$$1 \ 4 \ 5 \ 7 \ 8 \ 10 \ 11 \ 12 \ 13 \ 20 \ 23 \ 25 \ 29 \ 30 \ 2 \ 3 \ 6 \ 9 \ 14 \ 15 \ 16 \ 17 \ 18 \ 19 \ 21 \ 22 \ 24 \ 26 \ 27 \ 28.$$

(б) Диаграмма P содержит в верхней строке индексы левых скобок и свободных скобок (пользуясь обозначениями из (39)); остальные индексы находятся во второй строке. Таким образом, из (38)

$$P = \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|} \hline 1 & 2 & 3 & 6 & 8 & 9 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 21 & 22 & 24 & 26 & 27 & 28 \\ \hline 4 & 5 & 7 & 10 & 20 & 23 & 25 & 29 & 30 & & & & & & & & & & & & \\ \hline \end{array}.$$

[См. в K.-P. Vo, *SIAM J. Algebraic and Discrete Methods*, **2** (1981), 324–332, обобщение для цепочек подмультимножеств.]

80. Этот любопытный факт является следствием упражнения 79 совместно с теоремой 6 из статьи автора о диаграммах [см. *Pacific J. Math.*, **34** (1970), 709–727].

81. Предположим, что σ и σ' принадлежат соответственно цепочкам длиной s и s' в шаблонах рождественской елки порядка n и n' . В биклаттере могут находиться на более $\min(s, s')$ из ss' возможных пар строк из этих цепочек. Кроме того, с учетом (39), эти ss' пар строк в действительности при конкатенации образуют ровно $\min(s, s')$ цепочек в шаблоне рождественской елки порядка $n + n'$. Таким образом, сумма $\min(s, s')$ по всем парам цепочек равна $M_{n+n'}$, что и дает искомый результат. Кстати, при этом мы доказали неочевидное тождество

$$\sum_{j,k} \min(m+1-2j, n+1-2k) C_{j(m-j)} C_{k(n-k)} = M_{m+n}.$$

Примечание: это расширение теоремы Спернера было доказано независимо Г. Катоной (G. Katona) [*Studia Sci. Math. Hungar.*, **1** (1966), 59–63] и Д. Кляйтманом (D. Kleitman) [*Math. Zeitschrift*, **90** (1965), 251–259]. Доказательство из этого упражнения и некоторые дополнительные результаты приведены в Greene and Kleitman, *J. Combinatorial Theory*, **A20** (1976), 80–88.

82. (a) Существует, как минимум, одно вычисление в каждой строке m ; их два тогда и только тогда, когда $s(m) > 1$ и первое вычисление дает 0. Таким образом, если f тождественно 1, мы получаем минимум, M_n ; если f тождественно 0, мы получаем максимум, $M_n + \sum_m [s(m) > 1] = M_{n+1}$.

(b) Пусть $f(\chi(m, n/2)) = 0$ в $C_{n/2}$ случаях, когда $s(m) = 1$; в противном случае пусть $f(\chi(m, a)) = 1$, где a определяется алгоритмом. Когда n нечетно, из этого правила вытекает, что $f(\sigma)$ всегда равно 1; но когда n четно, $f(\sigma) = 0$ тогда и только тогда, когда σ — первая битовая строка в своей строке. (Чтобы понять, почему это так, воспользуйтесь тем фактом, что строка, содержащая σ'_j в формуле (41), всегда имеет размер $s - 2$.) Эта функция f в действительности монотонна; если $\sigma \leq \tau$ и если σ имеет свободные левые скобки, то же справедливо и для τ . Например, в случае $n = 8$ мы имеем

$$f(x_1, \dots, x_8) = x_8 \vee x_6 x_7 \vee x_4 x_5 (x_6 \vee x_7) \vee x_2 x_3 (x_4 (x_5 \vee x_6 \vee x_7) \vee x_5 (x_6 \vee x_7)).$$

(c) В этих обстоятельствах функция (45) является решением для всех n .

83. На шаге H4 возможно не более 3 исходов, на самом деле не более 2 при $s(m) = 1$. [См. более строгие границы в упражнении 5.3.4–31; в обозначениях из этого упражнения имеется ровно $\delta_n + 2$ монотонных булевых функций от n булевых переменных.]

84. Разобьем 2^n битовых строк вместо цепочек на M_n блоков, причем строки $\{\sigma_1, \dots, \sigma_s\}$ каждого блока удовлетворяют условию $\|A\sigma_i^T - A\sigma_j^T\| \geq 1$ при $i \neq j$; тогда условию $\|A\sigma^T - v\| \leq 1/2$ могут удовлетворять не более одной битовой строки в каждом блоке.

Пусть A' обозначает первые $n - 1$ столбцов A , а v — n -й столбец. Предположим, что $\{\sigma_1, \dots, \sigma_s\}$ — блок A' , при этом количество индексов таково, что $v^T A' \sigma_1^T$ минимально среди всех $v^T A' \sigma_j^T$. Тогда правило (36) определяет соответствующие блоки для A , поскольку

$$\|A(\sigma_i 0)^T - A(\sigma_j 0)^T\| = \|A(\sigma_i 1)^T - A(\sigma_j 1)^T\| = \|A' \sigma_i^T - A' \sigma_j^T\|$$

и

$$\begin{aligned} \|A(\sigma_j 1)^T - A(\sigma_1 0)^T\|^2 &= \|A' \sigma_j^T + v - A' \sigma_1^T\|^2 = \\ &= \|A' (\sigma_j - \sigma_1)^T\|^2 + \|v\|^2 + 2v^T A' (\sigma_j - \sigma_1)^T \geq \|v\|^2 \geq 1. \end{aligned}$$

[Справедливо и многое другое; см. *Advances in Math.*, 5 (1970), 155–157. Этот результат является расширением теоремы Д.Э. Литлвуда (J.E. Littlewood) и А.С. Оффорда (A.C. Offord), *Mat. Sbornik*, 54 (1943), 277–285, которые рассмотрели случай $m = 2$.]

85. Если V имеет размерность $n - m$, можно перенумеровать координаты так, что

$$\begin{aligned} (1, & 0, \dots, 0, & x_{11}, \dots, & x_{1m}) \\ (0, & 1, \dots, 0, & x_{21}, \dots, & x_{2m}) \\ \vdots & \vdots \ddots \vdots & \vdots & \vdots \\ (0, & 0, \dots, 1, & x_{(n-m)1}, \dots, & x_{(n-m)m}) \end{aligned}$$

представляет собой базис, в котором ни один вектор-строка $v_j = (x_{j1}, \dots, x_{jm})$ не состоит из нулей. Пусть $v_{n-m+1} = (-1, 0, \dots, 0), \dots, v_n = (0, 0, \dots, -1)$. Тогда количество $0 - 1$ векторов в V равно количеству $0 - 1$ решений $Ax = 0$, где A — матрица размером $m \times n$ со столбцами v_1, \dots, v_n . Но эта величина не превосходит количество решений $\|Ax\| < \frac{1}{2} \min(\|v_1\|, \dots, \|v_n\|)$, которое, согласно упражнению 84, не превосходит M_n .

И обратно: базис с $m = 1$ и $x_{j1} = (-1)^{j-1}$ дает M_n решений. [Этот результат имеет применение в электронном голосовании; см. диссертацию Голля на соискание ученой степени Ph.D. (Stanford, 2004).]

86. Сначала переупорядочим 4-узловые поддеревья так, чтобы их коды уровней представляли собой 0121 (плюс константа); затем будем сортировать все большие и большие поддеревья, пока все они не станут каноническими. В результате мы получим коды уровней 0 1 2 3 4 3 2 1 2 3 2 1 2 0 1, а соответствующими указателями на родительские узлы являются 0 1 2 3 4 3 2 1 8 9 8 1 12 0 14.

87. (а) Условие выполняется тогда и только тогда, когда $c_1 < \dots < c_k \geq c_{k+1} \geq \dots \geq c_n$ для некоторого k , так что общее количество таких ситуаций равно $\sum_k \binom{n-1}{n-k} = 2^{n-1}$.

(б) Заметим, что $c_1 \dots c_k = c'_1 \dots c'_k$ тогда и только тогда, когда $p_1 \dots p_k = p'_1 \dots p'_k$; в таком случае $c_{k+1} < c'_{k+1}$ тогда и только тогда, когда $p_{k+1} < p'_{k+1}$.

88. Посещается ровно A_{n+1} лесов, и у A_k из них $p_k = \dots = p_n = 0$. Следовательно, шаг О4 выполняется A_n раз; p_k на шаге О5 изменяется $A_{k+1} - 1$ раз при $1 \leq k < n$. Шаг О5 также изменяет p_n всего $A_n - 1$ раз. Среднее количество обращений к памяти на одно посещение, таким образом, составляет $2 + 3/(\alpha - 1) + O(1/n) \approx 3.534$, если хранить p_n в регистре. [См. E. Kubicka, *Combinatorics, Probability and Computing*, 5 (1996), 403–417.]

89. Если на шаге О5 присваивание $p_n \leftarrow p_j$ выполняется ровно Q_n раз, то присваивание $p_k \leftarrow p_j$ выполняется $Q_k + A_{k+1} - A_k$ раз для $1 < k < n$, поскольку каждый префикс канонического $p_1 \dots p_n$ является каноническим. Таким образом, получаем $(Q_1, Q_2, \dots) = (0, 0, 1, 2, 5, 9, 22, 48, 118, 288, \dots)$; можно показать, что $Q_n = \sum_{d \geq 1} \sum_{1 \leq c < n/d-1} a_{(n-cd)(n-cd-d)}$, где a_{nk} — количество канонических родительских последовательностей $p_1 \dots p_n$ с $p_n = k$. Однако сами значения a_{nk} пока что остаются загадкой.

90. (а) Это свойство эквивалентно 2.3.4.4–(7); вершина 0 является центроидом.

(б) Пусть $m = \lfloor n/2 \rfloor$. Требуется внести следующие изменения. В конце шага О1 установить $p_{m+1} \leftarrow 0$, а также $p_{2m+1} \leftarrow 0$, если n нечетно. В конце шага О4 установить $i \leftarrow j$ и, пока $p_i \neq 0$, устанавливать $i \leftarrow p_i$. (Тогда i является корнем дерева, содержащего j и k .) В начале шага О5, если $k = i + m$ и $i < j$, установить $j \leftarrow i$ и $d \leftarrow m$.

(с) Если n четно, бицентроидных деревьев с $n + 1$ вершинами не существует. В противном случае найдем все пары $(p'_1 \dots p'_m, p''_1 \dots p''_m)$ канонических лесов из $m = \lfloor n/2 \rfloor$ узлов, где $p'_1 \dots p'_m \geq p''_1 \dots p''_m$; пусть $p_1 = 0$, $p_{j+1} = p'_j + 1$ и $p_{m+j+1} = (p''_j + m + 1)$ [$p''_j > 0$] для $1 \leq j \leq m$. (Все такие последовательности будут сгенерированы двумя реализациями алгоритма О. Такой алгоритм для свободных деревьев разработан Ф. Раски (F. Ruskey) и Г. Ли (G. Li) [см. SODA, 10 (1999), S939–S940].)

91. Воспользуемся следующей рекурсивной процедурой $W(n)$. Если $n \leq 2$, процедура возвращает единственное n -узловое ориентированное дерево. В противном случае выбираются положительные целые числа j и d так, что данная пара (j, d) получается с вероятностью $dA_d A_{n-jd}/((n-1)A_n)$. Вычисляются случайные ориентированные деревья $T' \leftarrow W(n-jd)$ и $T'' \leftarrow W(d)$. Возвращается дерево T , полученное путем связи j копий T'' с корнем T' . [Combinatorial Algorithms (Academic Press, 1975), глава 25.]

92. Не всегда. [Р.Л. Камминс (R.L. Cummins) доказал в *IEEE Trans. CT-13* (1966), 82–90, что граф $S(G)$ всегда содержит цикл; см. также С.А. Holzmann and F. Harary, *SIAM J. Applied Math.*, **22** (1972), 187–193. Однако их построение непригодно для эффективного вычисления, поскольку требует предварительной информации о четности размеров промежуточных результатов.]

93. Да. Шаг S7 аннулирует шаг S3, а шаг S9 аннулирует удаление на шаге S8.

94. Например, можно воспользоваться поиском в глубину с использованием вспомогательной таблицы $b_1 \dots b_n$.

- i) Установить $b_1 \dots b_n \leftarrow 0 \dots 0$, затем установить $v \leftarrow 1$, $w \leftarrow 1$, $b_1 \leftarrow 1$ и $k \leftarrow n-1$.
- ii) Установить $e \leftarrow n_{v-1}$. Пока $t_e \neq 0$, выполнять следующие действия:
 - a) установить $u \leftarrow t_e$. Если $b_u \neq 0$, перейти к подшагу (c);
 - b) установить $b_u \leftarrow w$, $w \leftarrow u$, $a_k \leftarrow e$, $k \leftarrow k-1$. Завершить работу алгоритма, если $k = 0$;
 - c) установить $e \leftarrow n_e$.
- iii) Если $w \neq 1$, установить $v \leftarrow w$, $w \leftarrow b_w$ и вернуться к шагу (ii). В противном случае сообщить об ошибке: данный граф не является связным.

В действительности работу алгоритма можно завершить, как только подшаг (b) сделает k равным 1, поскольку алгоритм S никогда не смотрит на начальное значение a_1 . Однако мы можем продолжить работу и выполнить проверку связности графа.

95. Описанные далее шаги выполняют поиск в ширину из u , проверяя, достигима ли вершина v без использования ребра e . Используется вспомогательный массив $b_1 \dots b_n$ указателей на дуги, который должен быть инициализирован нулевыми значениями в конце шага S1; мы сновабросим его значения к $0 \dots 0$.

- i) Установить $w \leftarrow u$ и $b_w \leftarrow v$.
- ii) Установить $f \leftarrow n_{u-1}$. Пока $t_f \neq 0$, выполнять следующие действия:
 - a) установить $v' \leftarrow t_f$. Если $b'_v \neq 0$, перейти к подшагу (d);
 - b) если $v' \neq v$, установить $b'_v \leftarrow v$, $b_w \leftarrow v'$, $w \leftarrow v'$ и перейти к подшагу (d);
 - c) если $f \neq e \oplus 1$, перейти к шагу (v);
 - d) установить $f \leftarrow n_f$.
- iii) Установить $u \leftarrow b_u$. Если $u \neq v$, вернуться к шагу (ii).

- iv) Установить $u \leftarrow t_e$. Пока $u \neq v$, устанавливать $w \leftarrow b_u$, $b_u \leftarrow 0$, $u \leftarrow w$. Перейти к шагу S9 (e — мост).
- v) Установить $u \leftarrow t_e$. Пока $u \neq v$, устанавливать $w \leftarrow b_u$, $b_u \leftarrow 0$, $u \leftarrow w$. Затем вновь установить $u \leftarrow t_e$ и продолжить выполнение шага S8 (e — не мост).

Перед началом описанных вычислений можно воспользоваться двумя быстрыми эвристиками. Если $d_u = 1$, то очевидно, что e является мостом; если $l_{t_e} \neq 0$, то очевидно, что e мостом не является (поскольку существует другое ребро между u и v). Эти частные случаи быстро обнаруживаются путем поиска в ширину; эксперименты автора показывают, что их использование определенно стбит затрачиваемых усилий. Например, проверка l_{t_e} обычно позволяет сэкономить около 3% общего времени вычислений.

96. (a) Пусть e_k — дуга $k - 1 \rightarrow k$. Шаги, описанные в упражнении 94, устанавливают $a_k \leftarrow e_{n+1-k}$ для $n > k \geq 1$. Тогда на уровне k мы сжимаем e_{n-k} при $1 \leq k < n - 1$. После посещения (единственного) оставшегося дерева $e_{n-1} \dots e_2 e_n$, мы восстанавливаем e_{n-k} и быстро обнаруживаем, что это мост, для $n - 1 > k \geq 1$. Таким образом, время работы линейно зависит от n ; авторская реализация выполняет ровно $102n - 226$ обращений к памяти при $n \geq 3$.

Однако этот результат весьма существенно зависит от порядка ребер в начальном оставшемся дереве. Если на шаге S1 получается порядок “органных труб” наподобие

$$e_{n/2+1} e_{n/2} e_{n/2+2} e_{n/2-1} \dots e_{n-1} e_2$$

в позициях $a_2 \dots a_{n-1}$ при четном n , то время работы должно быть $\Omega(n^2)$, поскольку $\Omega(n)$ проверок мостов требуют времени $\Omega(n)$ каждая.

(b) Здесь a_k изначально представляет собой e_{n-k} для $n > k \geq 1$, где e_1 — дуга $n \rightarrow 1$. Посещенные оставшиеся деревья при $n \geq 4$ представляют собой соответственно $e_{n-2} \dots e_1 e_n$, $e_{n-2} \dots e_1 e_{n-1}$, $e_{n-2} \dots e_2 e_{n-1} e_n$, $e_{n-2} \dots e_3 e_{n-1} e_n e_1$, \dots , $e_{n-1} e_n e_1 \dots e_{n-3}$. Вслед за деревом $e_{n-2} \dots e_{k+2} e_{n-1} e_n e_1 \dots e_k$ вычисления опускаются на уровень $n - k - 3$ и вновь поднимаются для $0 \leq k \leq n - 4$; все проверки мостов эффективны. Таким образом, общее время работы квадратично (в авторской версии — $35.5n^2 + 7.5n - 145$ обращений к памяти при $n \geq 5$).

Кстати, в обозначениях Stanford GraphBase P_n представляет собой $board(n, 0, 0, 0, 1, 0, 0)$, а C_n — $board(n, 0, 0, 0, 1, 1, 0)$; вершины в Stanford GraphBase нумеруются от 0 до $n - 1$.

97. Да, когда $\{s, t\}$ равно $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, $\{2, 4\}$ или $\{3, 4\}$, но не $\{1, 4\}$.

98. $A' =$; это так называемый “дуальный планарный граф” планарного графа A . (Близкие деревья A' комплементарны оставшимся деревьям A и наоборот.)

99. Этот метод работает (в чем можно убедиться по индукции по размеру дерева), по сути, по тем же причинам, по которым он работает для n -кортежей в разделе 7.2.1.1, но с дополнительным условием, как именно мы должны обозначать каждый дочерний узел непростого узла.

Листья всегда пассивны, и они не являются ни простыми, ни непростыми; поэтому мы считаем, что внутренние узлы пронумерованы от 1 до m в прямом порядке обхода. Пусть $f_p = p$ для всех внутренних узлов, за исключением тех случаев, когда p — пассивный непростой узел, у которого ближайший непростой узел справа является активным; в этом случае f_p должен указывать на ближайший активный непростой узел слева. (Для такого определения мы представляем наличие искусственных узлов 0 и $m + 1$ слева и справа; оба эти узла непростые и активные.)

- F1.** [Инициализация.] Установить $f_p \leftarrow p$ для $0 \leq p \leq m$; установить также $t_0 \leftarrow 1$, $v_0 \leftarrow 0$ и установить все z_p так, чтобы $r_{z_p} = d_p$.
- F2.** [Выбор узла p .] Установить $q \leftarrow m$; затем, пока $t_q = v_q$, устанавливать $q \leftarrow q - 1$. Установить $p \leftarrow f_q$ и $f_q \leftarrow q$; завершить работу алгоритма, если $p = 0$.
- F3.** [Изменение d_p .] Установить $s \leftarrow d_p$, $s' \leftarrow r_s$, $k \leftarrow v_p$ и $d_p \leftarrow s'$. (Сейчас $k = v_s \neq v'_s$.)
- F4.** [Обновление значений.] Установить $q \leftarrow s$ и $v_q \leftarrow k \oplus 1$. Пока $d_q \neq 0$, устанавливать $q \leftarrow d_q$ и $v_q \leftarrow k \oplus 1$. (Теперь q является листом, который входит в конфигурацию, если $k = 0$, и покидает ее, если $k = 1$.) Аналогично установить $q \leftarrow s'$ и $v_q \leftarrow k$. Пока $d_q \neq 0$, устанавливать $q \leftarrow d_q$ и $v_q \leftarrow k \oplus 1$. (Теперь q является листом, который покидает конфигурацию, если $k = 0$, и входит в нее, если $k = 1$.)
- F5.** [Посещение.] Посетить текущую конфигурацию, представленную всеми значениями листьев.
- F6.** [Пассивирование p ?] (Все непростые узлы справа от p в настоящий момент активны.) Если $d_p \neq z_p$, вернуться к шагу F2. В противном случае установить $z_p \leftarrow s$, $q \leftarrow p - 1$; пока $t_q = v_q$, устанавливать $q \leftarrow q - 1$. (Теперь q представляет собой первый непростой узел слева от p ; мы сделаем p пассивным.) Установить $f_p \leftarrow f_q$, $f_q \leftarrow q$ и вернуться к шагу F2. ■

Хотя шаг F4 может преобразовывать непростые узлы в простые и наоборот, обновлять указатели f не требуется, поскольку они остаются корректно установленными.

100. Завершенную программу под названием GRAYSPSPAN можно найти на сайте автора. Доказательство ее асимптотической эффективности использует результат, полученный в упражнении 110.

102. Если это так, то обычные оставные деревья могут быть перечислены в *строгом порядке двери-вертушки*, при котором ребра, на каждом шаге входящие в оставное дерево и покидающие его, являются смежными.

Интересные алгоритмы для генерации всех ориентированных оставных деревьев с заданным корнем приведены в работах Harold N. Gabow and Eugene W. Myers, *SICOMP*, 7 (1978), 280–287 и S. Kapoor and H. Ramesh, *Algorithmica*, 27 (2002), 120–130.

103. (а) Рассыпание лексикографически увеличивает (x_0, x_1, \dots, x_n) , но не изменяет значение $x_0 + \dots + x_n$. В каком бы порядке мы ни рассыпали V_i и V_j , результат будет один и тот же.

(b) Добавление песчинки изменяет 16 устойчивых состояний следующим образом:

Дано: 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
 $+0001 0001 0010 0011 0001 0101 0110 0111 0101 1001 1010 1011 1001 1101 1110 1111 1101$
 $+0010 0010 0011 0001 0010 0110 0111 0101 0110 1010 1011 1001 1010 1110 1111 1101 1110$
 $+0100 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 0100 0101 0110 0111$
 $+1000 1000 1001 1010 1011 1100 1101 1110 1111 0100 0101 0110 0111 1000 1001 1010 1011$

Рекуррентными состояниями являются девять случаев с $x_1 + x_2 > 0$ и $x_3 + x_4 > 0$. Заметим, что многократное добавление 0001 приводит к бесконечному циклу $0000 \rightarrow 0001 \rightarrow 0010 \rightarrow 0011 \rightarrow 0001 \rightarrow 0010 \rightarrow \dots$, однако состояния 0001, 0010 и 0011 не являются рекуррентными.

(c) Если $x = \sigma(x + t)$, то также для всех $k \geq 0$ справедливо $x = \sigma(x + kt)$. Все компоненты t положительны; таким образом, $x = \sigma(x + \max(d_1, \dots, d_n))$ — рекуррентное состояние. И наоборот, предположим, что $x = \sigma(d + y)$, где все $y_i \geq 0$; тогда $d + y + t$ рассыпается до состояния $x + t$, а оно рассыпается до $\sigma(d) + y + t = d + y$. Следовательно, $\sigma(x + t) = \sigma(d + y) = x$.

(d) Имеется $N = \det(a_{ij})$ классов, поскольку элементарные операции со строками (упр. 4.6.1–19) приводят матрицу к треугольному виду, сохраняя конгруэнтность.

(e) Имеются неотрицательные целые числа $m_1, \dots, m_n, m'_1, \dots, m'_n$, такие, что

$$x + m_1 a_1 + \dots + m_n a_n = x' + m'_1 a_1 + \dots + m'_n a_n \text{ и равно, скажем, } y.$$

Для достаточно большого k вектор $y + kt$ рассыпается за $m_1 + \dots + m_n$ шагов до $x + kt$, а за $m'_1 + \dots + m'_n$ шагов — до $x' + kt$. Следовательно, $x = \sigma(x + kt) = \sigma(x' + kt) = x'$.

(f) Приведение к треугольному виду в ответе (d) показывает, что $x \equiv x + Ny$ для произвольных векторов y . Рассыпание сохраняет конгруэнтность; следовательно, каждый класс содержит рекуррентное состояние.

(g) Поскольку в сбалансированном ориентированном графе $a = a_1 + \dots + a_n$, мы получаем $x \equiv x + a$. Если x — рекуррентное состояние, мы видим, что фактически каждая вершина рассыпается только один раз, если $x + a$ сокращается до x , так как векторы $\{a_1, \dots, a_n\}$ линейно независимы.

И наоборот: если $\sigma(x + a) = x$, мы должны доказать, что состояние x рекуррентно. Пусть $z_m = \sigma(ma)$; должны существовать некоторые положительные k и m , для которых $z_{k+m} = z_m$. Тогда каждая вершина рассыпается k раз, пока $z_m + ka$ сокращается до z_m . Следовательно, существуют векторы $y_j = (y_{j1}, \dots, y_{jn})$ с $y_{jj} \geq d_j$, такие, что $(m + k)a$ рассыпается до y_j . Отсюда следует, что $x + n(m + k)a$ рассыпается до $x + y_1 + \dots + y_n$ и $\sigma(x + y_1 + \dots + y_n) = \sigma(x + n(m + k)a) = x$.

(h) Рассматривая индексы циклически, остовные деревья с дугами $V_j \rightarrow V_0$ для $j = i_1, \dots, i_k$ имеют $n - k$ других дуг: $V_j \rightarrow V_{j-1}$ для $i_l < j \leq i_l + q_l$ и $V_j \rightarrow V_{j+1}$ для $i_l + q_l < j < i_{l+1}$. Аналогично: рекуррентные состояния имеют $x_j = 2$ для $j = i_1, \dots, i_k$ и $x_j = 1$ для $i_l < j < i_{l+1}$, за исключением $x_j = 0$ при $j = i_l + q_l$ и $q_l > 0$.

(i) В этом случае состояние $x = (x_1, \dots, x_n)$ рекуррентно тогда и только тогда, когда $(n - x_1, \dots, n - x_n)$ является решением задачи о парковке, приведенной в указании, поскольку $t = (1, \dots, 1)$, а последовательность незапарковавшихся машин оставляет “дыру”, которая останавливает рассыпание $x + t$ в x .

Примечание: эта модель, разработанная Дипаком Дхаром (Deepak Dhar) [Phys. Review Letters, **64** (1990), 1613–1616], привела к появлению массы статей в физических журналах. Дхар заметил, что если внести случайным образом M песчинок, то при $M \rightarrow \infty$ все рекуррентные состояния становятся равновероятными. Данное упражнение основано на материалах работы R. Cori and D. Rossin, European J. Combinatorics, **21** (2000), 447–459.

Теория барханов доказывает, что каждый ориентированный граф D порождает абелеву группу, элементы которой некоторым образом соответствуют ориентированным оствовым деревьям D с корнем V_0 . В частности, это истинно в случае, когда D — обычный граф с дугами $u \rightarrow v$ и $v \rightarrow u$ для смежных узлов u и v . Таким образом, например, можно “сложить” два оствовых дерева, а некоторое оствовое дерево может рассматриваться как “нуль”. Элегантное соответствие между оствовыми деревьями и рекуррентными состояниями в частном случае, когда D представляет собой обычный граф, найдено Р. Кори (R. Cori) и И. Ле Борне (Y. Le Borgne) [Advances in Applied Math., **30** (2003), 44–52]. Однако для ориентированного графа D в общем случае простое соответствие неизвестно. Предположим, например, что $n = 2$ и $(e_{10}, e_{12}, e_{20}, e_{21}) = (p, q, r, s)$; тогда имеется $pr + ps + qr$ ориентированных деревьев и рекуррентные состояния соответствуют обобщенным двухмерным торам, как в упражнении 7-00. Но даже в “сбалансированном” случае, когда $p+q \geq s$ и $r+s \geq q$, по-видимому, нет простого отображения оствовых деревьев на рекуррентные состояния.

104. (a) Если $\det(\alpha I - C) = 0$, существует вектор $x = (x_1, \dots, x_n)^T$, такой, что $Cx = \alpha x$ и $\max(x_1, \dots, x_n) = x_m = 1$ для некоторого m . Тогда $\alpha = \alpha x_m = c_{mm} - \sum_{j \neq m} e_{mj} x_j \geq c_{mm} - \sum_{j \neq m} e_{mj} = 0$. (Кстати, действительная симметричная матрица, собственные значения которой неотрицательны, называется *положительно полуопределенной* (positive semidefinite). Наше доказательство устанавливает хорошо известный факт, что этим свойством обладает любая симметричная матрица, у которой $c_{mm} \geq |\sum_{j \neq m} c_{mj}|$ для $1 \leq m \leq n$.) Таким образом, $\alpha_0 \geq 0$; а поскольку $C(1, \dots, 1)^T = (0, \dots, 0)^T$, $\alpha_0 = 0$.

(b) $\det(xI - C(G)) = x(x - \alpha_1) \dots (x - \alpha_{n-1})$; а согласно теореме о матрице, соответствующей дереву, коэффициент при x равен $(-1)^{n-1} n$, умноженному на количество оствовых деревьев.

(c) $\det(\alpha I - C(K_n)) = \det((\alpha - n) I + J) = (\alpha - n)^{n-1} \alpha$ в соответствии с упражнением 1.2.3–36; здесь J — матрица, все элементы которой равны 1. Таким образом, сторонами полного графа являются $0, n, \dots, n$.

105. (a) Если $e_{ij} = a + b e'_{ij}$, то $C(G) = naI - aJ + bC(G')$. Если C — произвольная матрица, суммы элементов строк которой равны нулю, можно доказать тождество

$$\det(xI + yJ - zC) = \frac{x + ny}{x} z^n \det((x/z)I - C),$$

прибавляя столбцы от 2-го до n -го к первому, вынося за скобки $(x + ny)/z$, вычитая из столбцов со второго по n -й первый столбец, умноженный на y/x , а затем вычитая столбцы со второго по n -й из столбца 1. Следовательно, принимая $x = \alpha - na$, $y = a$, $z = b$, $a = 1$ и $b = -1$, мы находим, что G имеет стороны $0, n - \alpha_{n-1}, \dots, n - \alpha_1$. (В частности, этот результат согласуется с упражнением 104 (c), если G' — пустой граф $\overline{K_n}$.)

(b) Отсортируйте $\{\alpha'_0, \dots, \alpha'_{n'-1}, \alpha''_0, \dots, \alpha''_{n''-1}\}$. (Для разнообразия — очень простой случай.)

(c) Здесь $\overline{G} = \overline{G}' + \overline{G}''$, так что в соответствии с частями (a) и (b) сторонами G являются $\{0, n' + n'', n'' + \alpha'_1, \dots, n'' + \alpha'_{n'-1}, n' + \alpha''_1, \dots, n' + \alpha''_{n''-1}\}$. (В частности, G представляет собой $K_{m,n}$, если $G' = \overline{K_m}$ и $G'' = \overline{K_n}$, следовательно, сторонами $K_{m,n}$ являются $\{0, (n-1) \cdot m, (m-1) \cdot n, m+n\}$.)

(d) $C(G) = I'_n \otimes C(G'') + C(G') \otimes I''_n$, где I_n обозначает единичную матрицу размером $n \times n$, а символ \otimes — произведение Кронекера двух матриц. Сторонами $C(G)$ являются $\{\alpha'_j + \alpha''_k \mid 0 \leq j < n', 0 \leq k < n''\}$; если A и B — произвольные матрицы, собственными значениями которых являются соответственно $\{\lambda_1, \dots, \lambda_m\}$ и $\{\mu_1, \dots, \mu_n\}$, то собственными значениями $A \otimes I_n + I_m \otimes B$ будут $m n$ сумм $\lambda_j + \mu_k$. *Доказательство:* выберем S и T так, что $S^{-1}AS$ и $T^{-1}BT$ — треугольны. Затем воспользуемся матричным тождеством $(A \otimes B)(C \otimes D) = AC \otimes BD$, чтобы показать, что

$$(S \otimes T)^{-1} (A \otimes I_n + I_m \otimes B) (S \otimes T) = (S^{-1}AS) \otimes I_n + I_m \otimes (T^{-1}BT).$$

(В частности, многократное применение этой формулы показывает, что сторонами n -мерного куба являются $\{\binom{n}{0} \cdot 0, \binom{n}{1} \cdot 2, \dots, \binom{n}{n} \cdot 2n\}$, а формула (57) следует из упражнения 104 (b).)

(e) Если G — однородный граф степени d' , его сторонами являются $a_j = d' - \lambda_{j+1}$, где $\lambda_1 \geq \dots \geq \lambda_n$ — собственные значения матрицы смежности $A = (e_{ij})$. Матрицей смежности G' является $A' = B^T B - d'I'_n$, где $B = (b_{ij})$ — матрица инциденций размером $n \times n'$ с элементами $b_{ij} = [\text{Ребро } i \text{ касается вершины } j]$, а $n = n'd'/2$ — количество ребер. Матрицей смежности G является $A = BB^T - 2I_n$. Мы имеем

$$x^n \det(xI'_n - B^T B) = x'^n \det(xI_n - BB^T);$$

это тождество следует из того факта, что коэффициенты $\det(xI - A)$ можно выразить через $\text{trace}(A^k)$ для $k = 1, 2, \dots$, с использованием тождеств Ньютона (см. упражнение 1.2.9–10). Так что сторонами G являются стороны G' , а также $n - n'$ сторон, равных $2d'$. [Этот результат получен в работе Е.В. Vakhovsky, *Sibirskii Mat. Zhurnal*, **6** (1965), 44–49; см. также Н. Sachs, *Wissenschaftliche Zeitschrift der Technischen Hochschule Ilmenau*, **13** (1967), 405–412.]

(f) $A = A' \otimes A''$, так что сторонами являются $\{d''\alpha'_j + d'\alpha''_k - \alpha'_j\alpha''_k \mid 0 \leq j < n', 0 \leq k < n''\}$.

(g) $A(G) = I'_n \otimes A'' + A' \otimes I''_n + A' \otimes A'' = (I'_n + A') \otimes (I''_n + A'') - I_n$ дает стороны, равные $\{(d'' + 1)\alpha'_j + (d' + 1)\alpha''_k - \alpha'_j\alpha''_k \mid 0 \leq j < n', 0 \leq k < n''\}$.

106. (a) Если α — сторона пути P_n , то имеется ненулевое решение $(x_0, x_1, \dots, x_{n+1})$ уравнений $\alpha x_k = 2x_k - x_{k-1} - x_{k+1}$ для $1 \leq k \leq n$, причем $x_0 = x_1$ и $x_n = x_{n+1}$. Если положить $x_k = \cos(2k-1)\theta$, то мы получим $x_0 = x_1$ и $2x_k - x_{k-1} - x_{k+1} = 2x_k - (2\cos 2\theta)x_k$; следовательно, $2 - 2\cos 2\theta = 4\sin^2 \theta$ будет стороной, если выбрать θ так, чтобы $x_n = x_{n+1}$, и так, чтобы все x не были нулями. Таким образом, сторонами P_n являются $\sigma_{0n}, \dots, \sigma_{(n-1)n}$.

Поскольку $c(P_n) = 1$, из упражнения 104 (b) должно получаться $\alpha_1 \dots \alpha_{n-1} = n$; следовательно,

$$c(P_m \times P_n) = \prod_{j=1}^{m-1} \prod_{k=1}^{n-1} (\sigma_{jm} + \sigma_{kn}).$$

(b, c) Аналогично: если α является стороной цикла C_n , существует ненулевое решение указанных уравнений, такое, что $x_n = x_0$. В этом случае мы испытываем $x_k = \cos 2k\theta$ и находим решения при $\theta = j\pi/n$, где $0 \leq j < \lceil n/2 \rceil$. В то же время $x_k = \sin k\theta$ дает остальные линейно независимые решения для $\lceil n/2 \rceil \leq j < n$. Следовательно, сторонами C_n являются $\sigma_{0n}, \sigma_{2n}, \dots, \sigma_{(2n-2)n}$, и мы получаем

$$c(P_m \times C_n) = n \prod_{j=1}^{m-1} \prod_{k=1}^{n-1} (\sigma_{jm} + \sigma_{(2k)n}), \quad c(C_m \times C_n) = mn \prod_{j=1}^{m-1} \prod_{k=1}^{n-1} (\sigma_{(2j)m} + \sigma_{(2k)n}).$$

Пусть $f_n(x) = (x + \sigma_{1n}) \dots (x + \sigma_{(n-1)n})$ и $g_n(x) = (x + \sigma_{2n}) \dots (x + \sigma_{(2n-2)n})$. Эти полиномы имеют целые коэффициенты; действительно, $f_n(x) = U_{n-1}(x/2 + 1)$ и $g_n(x) = 2(T_n(x/2 + 1) - 1)/x$, где $T_n(x)$ и $U_n(x)$ — полиномы Чебышева, определяемые как $T_n(\cos \theta) = \cos n\theta$ и $U_n(\cos \theta) = (\sin(n+1)\theta)/\sin \theta$. Вычисление $c(P_m \times P_n)$ можно свести к вычислению определителя размером $m \times m$, поскольку он является результантом $f_m(x)$ и $f_n(-x)$ (см. упражнение 4.6.1-12). Аналогично: $\frac{1}{n}c(P_m \times C_n)$ и $\frac{1}{mn}c(C_m \times C_n)$ представляют собой результанты $f_m(x)$ и $g_n(-x)$, а также $g_m(x)$ и $g_n(-x)$ соответственно.

Пусть $\alpha_n(x) = \prod_{d \mid n} f_d(x)^{\mu(n/d)}$; таким образом, $\alpha_1(x) = 1$, $\alpha_2(x) = x + 2$, $\alpha_3(x) = (x + 3)(x + 1)$, $\alpha_4(x) = x^2 + 4x + 2$, $\alpha_5(x) = (x^2 + 5x + 5)(x^2 + 3x + 1)$, $\alpha_6(x) = x^2 + 4x + 1$ и т.д. Рассматривая так называемые характеристические полиномы, можно показать, что $\alpha_n(x)$ является неприводимым над целыми числами при четном n , а в остальных случаях представляет собой произведение двух неприводимых полиномов одной и той же степени. Аналогично: если $\beta_n(x) = \prod_{d \mid n} g_d(x)^{\mu(n/d)}$, то $\beta_n(x)$ при $n \geq 3$ является квадратом неприводимого полинома. Эти факты поясняют наличие достаточно малых простых множителей в результате. Например, наибольший простой множитель в $c(P_m \times P_n)$ при $m \leq n \leq 10$ равен 1 009; он встречается только в результанте $\alpha_6(x)$ и $\alpha_9(-x)$, который равен $662\,913 = 3^2 \cdot 73 \cdot 1\,009$.

107. Для $n = (1, \dots, 5)$ имеется $(1, 1, 2, 6, 21)$ неизоморфных графов; однако нам нужно рассмотреть только случаи с $\leq \frac{1}{2} \binom{n}{2}$ ребрами в силу упражнения 105 (а). Оставшиеся случаи при $n = 4$ представляют собой свободные деревья: звезда, являющаяся дополнением $K_1 + K_3$ со сторонами 0, 1, 1, 4, и P_4 со сторонами $0, 2 - \sqrt{2}, 2, 2 + \sqrt{2}$ согласно упражнению 106. При $n = 5$ существует три свободных дерева: звезда со сторонами 0, 1, 1, 1, 5; P_5 со сторонами $0, 2 - \phi, 3 - \phi, 1 + \phi, 2 + \phi$; а также $\text{---} \bullet \bullet$ со сторонами $0, r_1, 1, r_2, r_3$, где $(r_1, r_2, r_3) \approx (0.52, 2.31, 4.17)$ — корни уравнения $x^3 - 7x^2 + 13x - 5 = 0$.

И наконец, имеется пять случаев с одним циклом: $\text{---} \times$ представляет собой $K_1 \mp (K_2 + \overline{K_2})$, так что его сторонами являются 0, 1, 1, 3, 5; C_5 имеет стороны $0, 3 - \phi, 3 - \phi, 2 + \phi, 2 + \phi$; сторонами $\text{---} \bullet \bullet$ являются $0, r_1, r_2, 3, r_3$; у его дополнения $\text{---} \diamond \diamond$ стороны представляют собой $0, 5 - r_3, 2, 5 - r_2, 5 - r_1$; у $\text{---} \Delta$ стороны — $0, (5 - \sqrt{13})/2, 3 - \phi, 2 + \phi, (5 + \sqrt{13})/2$.

108. Пусть дан ориентированный граф D с вершинами $\{V_1, \dots, V_n\}$, и пусть e_{ij} — количество дуг от V_i к V_j . Определим $C(D)$ и его стороны, как и ранее. Поскольку матрица $C(D)$ не обязательно симметрична, действительность сторон больше не гарантирована. Но если α представляет собой сторону, то стороной является и комплексно сопряженное значение $\bar{\alpha}$; если мы упорядочим стороны в соответствии с их действительными частями, то, как и ранее, получим $\alpha_0 = 0$. Формула

$c(D) = \alpha_1 \dots \alpha_{n-1} / n$ остается верной, если рассматривать $c(D)$ как *среднее количество ориентированных* оствовых деревьев, взятое по всем возможным корням V_j . Очевидно, что сторонами транзитивного турнира T_n , дугами которого являются $V_i \rightarrow V_j$ при $1 \leq i < j \leq n$, являются $0, 1, \dots, n-1$; очевидны так же и стороны его подграфов.

Выходы в частях (а)–(д) ответа к упражнению 105 остаются без изменений. Например, рассмотрим $K_1 \mp T_3$, сторонами которого являются $0, 2, 3, 4$. Этот ориентированный граф D имеет $(2, 4, 6, 12)$ оствовых деревьев с четырьмя возможными корнями, и $c(D)$ действительно равно $2 \cdot 3 \cdot 4 / 4$. Обратите также внимание, что ориентированный граф  является своим собственным дополнением и имеет те же стороны, что и T_3 .

Ориентированные графы допускают также другое семейство интересных операций: если D' и D'' являются ориентированными графами на непересекающихся множествах вершин V' и V'' , рассмотрим добавление a дуг $v' \rightarrow v''$ и b дуг $v'' \rightarrow v'$, где $v' \in V'$ и $v'' \in V''$. Манипулируя определителями так же, как и в упражнении 105 (а), можно показать, что получающийся в результате ориентированный граф имеет стороны $\{0, an'' + bn', an'' + \alpha'_1, \dots, an'' + \alpha'_{n'-1}, bn' + \alpha''_1, \dots, bn' + \alpha''_{n''-1}\}$. В частном случае $a = 1$ и $b = 0$ можно ввести удобное обозначение для получающегося графа: $D' \rightarrow D''$; так, например, $T_n = K_1 \rightarrow T_{n-1}$. Ориентированный граф $K_{n_1} \rightarrow K_{n_2} \rightarrow \dots \rightarrow K_{n_m}$ с $n_1 + n_2 + \dots + n_m$ вершинами имеет стороны $\{0, n_m \cdot s_m, \dots, n_2 \cdot s_2, (n_1 - 1) \cdot s_1\}$, где $s_k = n_k + \dots + n_m$.

Очевидно, что сторонами ориентированного пути Q_n от V_1 до V_n являются $0, 1, \dots, 1$. Ориентированный цикл O_n имеет стороны $\{0, 1 - \omega, \dots, 1 - \omega^{n-1}\}$, где $\omega = e^{2\pi i/n}$.

Также есть красивый результат для ориентированного графа, в котором дуги соответствуют вершинам исходного графа: стороны графа D^* получаются из сторон графа D простым добавлением $\tau_k - 1$ копий числа σ_k ($1 \leq k \leq n$), где τ_k — входящая степень вершины V_k , а σ_k — ее исходящая степень. (Если $\tau_k = 0$, мы *удаляем* одну сторону, равную σ_k .) Соответствующее доказательство похоже (хотя и проще) на доказательство из упражнения 2.3.4.2–21.

Историческое примечание: результаты упражнений 104 (б) и 105 (а) основаны на работах А.К. Kelmans, *Avtomatika i Telemekhanika*, **26** (1965), 2194–2204; **27**, 2 (February 1966), 56–65 (перевод на английский язык в *Automation and Remote Control*, **26** (1965), 2118–2129; **27** (1966), 233–241. Мирославу Фидлеру (Miroslav Fiedler) [*Czech. Math. J.*, **23** (1973), 298–305] принадлежит авторство упражнения 105 (д); им же доказаны интересные результаты о стороне α_1 , которую он назвал “алгебраической связностью” графа G. Герман Креверас (Germain Kreweras) в *J. Combinatorial Theory, B***24** (1978), 202–212, пересчитал оствовые деревья решеток, цилиндров и торов, а также ориентированные оствовые деревья ориентированных торов, таких, как $O_m \times O_n$. Отличный обзор сторон графов опубликован Бояном Мохаром (Bojan Mohar) в *Graph Theory, Combinatorics and Applications* (Wiley, 1991), 871–898; *Discrete Math.*, **109** (1992), 171–183. Исчерпывающее обсуждение важных семейств собственных значений графов и их свойств, включая полную библиографию, можно найти в книге D.M. Cvetković, M. Doob and H. Sachs, *Spectra of Graphs*, third ed. (1995).

109. Вероятно, имеется также пояснение, связанное с барханами из упражнения 103.

110. Доказательство по индукции. Предположим, что имеется $k \geq 1$ параллельных ребер между u и v . Тогда $c(G) = kc(G_1) + c(G_2)$, где G_1 — это граф G со склеенными вершинами u и v , а G_2 — граф G , из которого удалены эти k ребер. Пусть $d_u = k + a$ и $d_v = k + b$.

Случай 1. G_2 — связный граф. Тогда $ab > 0$, так что можно записать $a = x + 1$ и $b = y + 1$. Мы имеем $c(G_1) > \alpha\sqrt{x+y+1}$ и $c(G_2) > \alpha\sqrt{xy}$, где α — произведение по всем остальным $n - 2$ вершинам; легко убедиться, что

$$k\sqrt{x+y+1} + \sqrt{xy} \geq \sqrt{(x+k)(y+k)}.$$

Случай 2. Не существует таких вершин u и v , для которых G_2 — связный граф. Тогда каждое мультиребро G представляет собой мост; другими словами, G — свободное дерево, за исключением наличия параллельных ребер. В этом случае результат тривиален, если существует вершина степени 1. В противном случае предположим, что u — конечная точка, и ее степень равна $d_u = k$ ребер $u-v$. Если $d_v > k+1$, то $c(G) = kc(G_1) > \alpha k\sqrt{x}$, где $d_v = k+1+x$, и легко проверить, что при $x > 0$ справедливо неравенство $k\sqrt{x} > \sqrt{(k-1)(k+x)}$. Если $d_v = k$, мы получаем $c(G) = k > \sqrt{(k-1)^2}$. Наконец, если $d_v = k+1$, примем $v_0 = u$, $v_1 = v$ и рассмотрим единственный путь $v_1-v_2-\dots-v_r$, где $r > 1$ и v_r имеет степень, большую 2; каждая пара вершин v_j и v_{j+1} ($1 \leq j < r$) соединена одним ребром. Гипотеза индукции выполняется.

[Другие нижние границы количества оставных деревьев получены в работе A.V. Kostochka, *Random Structures and Algorithms*, 6 (1995), 269–274.]

111. 2 1 5 4 11 7 9 8 6 10 15 12 14 13 3.

112. Либо p находится на четном уровне и является предком q , либо q находится на нечетном уровне и является предком p .

113. $\text{prepostorder}(F^R) = \text{postpreorder}(F)^R$ и $\text{postpreorder}(F^R) = \text{prepostorder}(F)^R$.⁹

114. Наиболее элегантный подход, учитывающий, что лес может быть пустым, заключается в такой инициализации, что $\text{CHILD}(\Lambda)$ указывает на корень крайнего слева дерева, если таковое имеется. Затем первое посещение обеспечивается установками $Q \leftarrow \Lambda$ и $L \leftarrow -1$, после чего выполняется переход к шагу Q6.

115. Предположим, что имеется n_e узлов на четных уровнях и n_o на нечетных и что n'_e узлов на четных уровнях не являются листьями. Тогда шаги Q1, …, Q7 выполняются соответственно $(n_e + n_o, n_o, n'_e, n_e, n'_e, n_o + 1, n_e)$ раз, включая одно выполнение шага Q6 в соответствии с ответом к упражнению 114.

116. (a) Этот результат следует из алгоритма Q.

(b) В действительности все узлы, не являющиеся обычными, строго чередуются между везучими и невезучими, начиная и заканчивая везучими узлами. *Доказательство:* рассмотрите лес F' , получающийся из F путем удаления крайнего слева листа, и воспользуйтесь индукцией по n .

117. Такие леса — это те, которые в представлении с левым сыном и правым братом дают вырожденное бинарное дерево (см. упражнение 31). Таким образом, окончательный ответ — 2^{n-1} .

⁹prepostorder означает прямо-обратный порядок обхода, а postpreorder — обратно-прямой. — Примеч. пер.

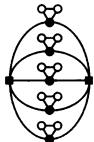
118. (a) t^{k-2} , $k > 1$; везение встречается только возле крайних листьев.

(b) Получающееся интересное рекуррентное соотношение приводит к решению $(F_k + 1 - (k + 1) \bmod 3)/2$.

119. Пометим каждый узел x значением $v(x) = \sum\{2^k \mid k \text{ — метка дуги на пути от корня к } x\}$. Тогда значения узлов в прямо-обратном порядке обхода представляют собой бинарный код Грея Γ_n , поскольку в упражнении 113 показано, что они удовлетворяют рекуррентному соотношению 7.2.1.1–(5).

(Если мы применим тот же способ пометок к обычному биномиальному дереву T_n и обойдем его в прямом порядке, то получим просто целые числа $0, 1, \dots, 2^n - 1$.)

120. Можно. Только четыре из “полых” вершин на приведенной иллюстрации могут следовать за “квадратными” вершинами в гамильтоновом цикле; следовательно, одной пустой паре не повезло. [См. H. Fleischner and H.V. Kronk, *Monatshefte für Mathematik*, **76** (1972), 112–117.]



121. Более того, гамильтонов путь от u к v в T^2 существует тогда и только тогда, когда выполняются подобные условия, описанные далее. Мы оставляем u и/или v в $T^{(i)}$, если они имеют степень 1, и требуем, чтобы путь в условии (i) содержался в пути от u к v (исключая сами u и v). Условие (ii) также усиливается посредством замены ‘вершины степени 4’ на ‘опасные вершины’, где вершина $T^{(i)}$ называется опасной, если она либо имеет степень 4, либо имеет степень 2 и равна u или v . Наименьший невозможный случай — $T = P_4$, где u и v выбраны так, чтобы они не являлись конечными точками. [*Časopis pro Pěstování Matematiky*, **89** (1964), 323–338.]

Следовательно, T^2 содержит гамильтонов цикл тогда и только тогда, когда T является *гусеницей* (caterpillar), т.е. свободным деревом, производная которого представляет собой путь. [См. Frank Harary and A.J. Schwenk, *Mathematika*, **18** (1971), 138–140.]

122. (a) Можно представить выражение в виде бинарного дерева, в котором операторы представлены внутренними узлами, а цифры — внешними. Если бинарное дерево реализовано так, как в алгоритме B, то главное ограничение, налагаемое грамматикой, состоит в том, что если $r_j = k > 0$, то оператором в узле j будет + или – тогда и только тогда, когда оператором в узле k будет × или /. Следовательно, общее количество возможных деревьев с n листьями равно $2^n S_{n-1}$, где S_n — число Шрёдера; при $n = 9$ это значение равно 10 646 016. (См. упражнение 66, только замените в нем ‘лево’ на ‘право’ и наоборот.) Мы можем быстро сгенерировать их все и убедиться, что всего имеется 1 640 решений. Без умножений обходится только одно решение, $1 + 2 / ((3 - 4) / (5 + 6) - (7 - 8) / 9)$; пять пар скобок требуют 20 решений, таких, как $((1 - 2) / ((3/4) \times 5 - 6)) \times 7 + 8) \times 9$; совсем без скобок обходятся только 15 решений.

(b) В этом случае существует $1 + \sum_{k=1}^8 \binom{8}{2} 2^{k+1} S_k = 23\,463\,169$ возможных деревьев и 3 365 решений. Кратчайшее решение (длина которого равна 12) было найдено Дьюдени [*The Weekly Dispatch* (18 June 1899)] и имеет вид 123 – 45 – 67 + 89; однако

сам Дьюдени в то время не был уверен, что оно наилучшее. Самое длинное решение имеет длину 27; как упоминалось выше, таких решений 20.

(c) При этом количество возможных случаев резко возрастает — до $2 + \sum_{k=1}^8 \binom{8}{k} 4^{k+1} S_k = 8\ 157\ 017\ 474$; количество решений равно 96 504. Наиболее длинное решение единственное, оно имеет длину 40: $((((.1 / (.2 + .3)) / .4) / .5) / (.6 - .7)) / (.8 - .9)$. Есть пять занимательных примеров наподобие $.1 + (2 + 3 + 4 + 5) \times 6 + 7 + 8 + .9$ с семью плюсами, а также 10 решений наподобие $(1 - .2 - .3 - 4 - .5 - 6) \times (7 - 8 - 9)$ с семью минусами.

*Есть только небольшое правило,
и нет ни одного точного способа показать,
что мы получили наилучшее возможное решение.
— Генри Э. Дьюдени (Henry E. Dudeney)*

Примечания. В первом издании *Illustriertes Spielbuch für Mädchen* Мари Леске (Marie Leske) (1864) содержится первое известное упоминание этой задачи; в 11-м издании (1899) приведено решение $100 = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 \times 9$. См. также ссылку из упражнения 7.2.1.1–111.

Ричард Беллман (Richard Bellman) [AMM, 69 (1962), 640–643] пояснил, как решается частный случай (a), когда операторы ограничены сложением и умножением и нельзя использовать скобки. Его метод динамического программирования можно также использовать и при решении общей задачи для снижения количества рассматриваемых случаев. Идея состоит в определении действительных чисел, получаемых для каждого подинтервала цифр $\{1, \dots, n\}$ для данного оператора в качестве корня дерева. Можно также сэкономить на вычислениях, отбрасывая случаи, которые для подинтервалов $\{1, \dots, 8\}$ и $\{2, \dots, 9\}$ не могут привести к целым решениям. Таким образом количество существенно различных деревьев, которые требуется рассмотреть, снижается (a) до 2 747 275; (b) до 6 834 708; (c) до 741 167 401.

Арифметика с плавающей точкой для решения данной задачи ненадежна, но можно использовать подпрограммы для работы с рациональными числами из раздела 4.5.1; при этом никогда не приходится работать с целыми числами, превышающими по абсолютному значению 10^9 .

123. (a) 2 284, но $2\ 284 = (1 + 2 \times 3) \times (4 + 5 \times 67) - 89$; (b) 6 964, но $6\ 964 = (1/.2) \times \times 34 + 5 + 6789$; (c) 14 786, но $14\ 786 = -1 + 2 \times (.3 + 4 + 5) \times (6 + 789)$. [Если мы допустим использование знака “минус” слева от выражения, как это делал Дьюдени, то получим 1 361, 2 758 и 85 054 дополнительных решений упражнения 122 (a), (b) и (c), включая 19 более длинных выражений в случае (a), таких, как $-(1 - 2) \times \times ((3 + 4) \times (5 - (6 - 7) \times 8) + 9)$. При таком расширении соглашений наименьшими непредставимыми числами являются: (a) 3 802; (b) 8 312; (c) 17 722.] Общее количество представимых чисел (положительных, отрицательных и нуля) равно: (a) 27 666; (b) 136 607; (c) 200 765.

124. Числа Хортона–Страхлера возникли при изучении течения рек в работах R.E. Horton, Bull. Geol. Soc. Amer., 56 (1945), 275–370; A.N. Strahler, Bull. Geol. Soc. Amer., 63 (1952), 1117–1142. Многие идеи по рисованию деревьев можно найти в классической статье Viennot, Eyrrolles, Janey and Arques, Computer Graphics, 23, 3 (July 1989), 31–40.

Раздел 7.2.1.7

1. Вероятно, оно связано с гексаграммой 21 (“выполнение вычислений” (“crunching”)) (䷂); однако древние комментаторы в большей степени связывают эту гексаграмму с обеспечением законности, чем со взаимодействием электронов¹⁰.

2. (а) Пусть первый нуклеотид в кодоне (T, C, A, G) будет представлен соответственно элементами (☱, ☰, ☲, ☱); второй нуклеотид аналогично представлен элементами (☲, ☷, ☵, ☶), а третий — (☵, ☸, ☹, ☺). Наложим друг на друга все три представления. Таким образом, например, гексаграмма 34 представляет собой ☷ = ☰ + ☷ + ☺ и тем самым является кодоном ТТС, который отображается на аминокислоту F. При использовании такого соответствия гексаграммы с 34 по 54 включительно отображаются соответственно на значения ($F, G, L, Q, W, D, S, -, P, Y, K, A, I, T, N, H, M, R, V, E, C$). Кроме того, имеются три гексаграммы, отображающиеся на ‘-’ — это гексаграммы 1, 9 и 41, а именно ☰, ☲ и ☱, которые в Ching имеют названия соответственно “созидание”, “укрощение” и “устранение излишнего”; все они на удивление подходят для обозначения завершения белка.

(б) Рассмотрим $\binom{64}{6,6,6,4,4,4,4,3,3,2,2,2,2,2,2,2,1,1} \approx 2.3 \times 10^{69}$ способов перестановки элементов массива генетического кода размером $4 \times 4 \times 4$. Ровно

$$2402880402175789790003993681964551328451668718750185553920000000 \approx 2.4 \times 10^{63}$$

из них содержат, как минимум, один отрезок из последовательности 21 различного элемента. [Использование принципа включения и исключения позволяет показать, что любое мультимножество $\{(n_1 + 1) \cdot x_1, \dots, (n_r + 1) \cdot x_r\}$ с r различными элементами и $n_r = 0$ имеет ровно

$$(n+1) \binom{n}{n_1, \dots, n_r} r! - \sum_{k=1}^r (n+1-k) k! (r-k)! a_k \sum_{\substack{0 \leq d_1, \dots, d_r \leq 1 \\ d_1 + \dots + d_r = k}} \binom{n-k}{n_1 - d_1, \dots, n_r - d_r}$$

таких перестановок, где $n = n_1 + \dots + n_r$, а a_k — количество неприводимых перестановок с k элементами (упражнение 7.2.1.2–100).] Таким образом, только одна из каждого миллиона перестановок может быть указана корректно.

Однако имеется $4!^3 \binom{6}{2,2,2} = 1\,244\,160$ способов представления кодонов так, как это сделано в части (а), и большинство из них соответствуют различным перестановкам аминокислот (за исключением обмена представления Т и С в третьей позиции).

И действительно, эмпирически около 31% всех 64 гексаграмм дают соответствующее отображение кодонов. Таким образом, конструкция в части (а) не дает оснований подозревать авторов *I Ching* в каком-либо участии в разработке генетического кода.

3. Поскольку $F_{31} - 10^6 = F_{28} + F_{22} + F_{20} + F_{18} + F_{16} + F_{14} + F_9$, миллионной строфой является ☰— ☱— ☷— ☺— ☲— ☰. Ответ на второй вопрос еще проще: $F_{31} - (F_5 + F_8 + F_{10} + F_{16} + F_{18} + F_{27} + F_{30}) = 314\,159$.

4. Одно из двух появлений ☷ в строке 4 должно быть ☷; это может быть простой типографской опечаткой. Аналогично, ☷ в строке 8 должно иметь вид

¹⁰По всей видимости, автор подразумевает еще одно значение слова “crunching” — раздавливать с хрустом. — Примеч. пер.

ל'מג. Однако Донноло несет ответственность за шесть случаев в строках 3 и 4, когда крайние справа буквы **ל** появляются дважды, в то время как перестановки с крайними справа буквами **ל** отсутствуют.

5. Последняя перестановка должна быть , а не .

6. В списке Мерсенна n -е значение m_n соответствует $n!$ только для $1 \leq n \leq 13$ и $15 \leq n \leq 38$. Мерсенн знал, что $14! = 87178291200 \neq m_{14} = 8778291200$, поскольку вставил '1' в личную копию книги (в настоящее время являющейся собственностью Bibliothèque Nationale; ее факсимиле было опубликовано в 1975 году). Однако остальные ошибки в его таблице не просто типографские, поскольку они распространяются на последующие значения, за исключением случая m_{50} , а именно: $m_{39} = 39! + 10^{26} - 10^{10}$; $m_{40} = 40m_{39}$; $m_{41} = 41m_{40} - 4 \cdot 10^{25} - 14 \cdot 10^{11}$; $m_n = nm_{n-1}$ для $n = 42, 43, 44, 46, 47, 48, 49, 55, 60$ и 62 ; $m_{50} = 50m_{49} + 10^{66}$; $m_{51} = 51 \cdot 50 \cdot m_{49}$. При вычислении $m_{45} = 9 \cdot 45 \cdot m_{44} - 10^{40} + 10^{29}$ он, похоже, хотел упростить вычисления, поскольку легче выполнить умножения на 5 и 9, чем на 45. Однако он ошибся и выполнил умножение на 9 *дважды*. Большинство ошибок указывают на ненадежность метода умножения, который может зависеть от того, какие счеты использованы для вычислений: $m_{52} = 52m_{51} + 5 \cdot 10^{56} - 2 \cdot 10^{47} + 10^{34}$; $m_{53} = 53m_{52} - 4 \cdot 10^{29}$; $m_{54} = 54m_{53} + 10^{16}$; $m_{57} = 57m_{56} + 10^{33} + 10^{24}$; $m_{58} = 58m_{57} + 10^{67} - 10^{35} + 10^{32} + 11 \cdot 10^{26}$; $m_{59} = 59m_{58} + 10^{66} + 10^{49} - 10^{28}$; $m_{61} = 61m_{60} - 5 \cdot 10^{81}$; $m_{63} = 63m_{62} + 10^{82} - 10^{74}$; $m_{64} = 64m_{63} + 3 \cdot 10^{81} + 10^{67} + 2 \cdot 10^{38} - 2 \cdot 10^{33} - 10^{23}$.

Оставшийся случай, $m_{56} \approx 10.912m_{55}$, непостижим. Это значение $\equiv 56m_{55} \pmod{10^{17}}$, но эти цифры тоже ничего не говорят ни уму, ни сердцу.

Примечание: Атанасиус Кирхер (Athanasius Kircher) явно копировал результаты Мерсенна, когда табулировал $n!$ для $1 \leq n \leq 50$ на с. 157 своей книги *Ars Magna Scientiæ* (1669), поскольку он повторил все ошибки Мерсенна. Однако Кирхер использовал значения $10m_{14}$, $m_{45}/10$ и $10m_{49}$ вместо m_{14} , m_{45} и m_{49} ; вероятно, он пытался сделать последовательность растущей более равномерно. Неясно, кто первым вычислил точное значение $39!$; в упражнении 1.2.5–4 описана история числа $1000!$.

7. Базовыми перестановками являются 12345, 13254, 14523, 15432, 12453, 14235, 15324, 13542, 12534, 15243, 13425, 14352. Но после этого мы находим, что все 60 из четных перестановок одновременно и живы и мертвы, поскольку (9) отличается от (8) на четную перестановку. (Более того, если бы можно было как-то исправить ситуацию при $n = 5$, то при $n = 6$ половина живых перестановок стали бы нечетными.)

8. Например, можно заменить (9) на

$$a_n a_3 \dots a_{n-1} a_2 a_1, a_1 a_4 \dots a_n a_3 a_2, \dots, a_{n-1} a_2 \dots a_{n-2} a_1 a_n,$$

где переходы осуществляются путем обмена местами двух конечных элементов слева и справа и циклического сдвига последовательности влево на один элемент. Такая модификация работает правильно, поскольку все перестановки имеют корректную четность, а живые и мертвые перестановки содержат a_1 во всех возможных позициях (по сути, мы получили дуальную таблицу Симса (Sims) для чередующейся группы, как в 7.2.1.2–(32); однако вместо $(0, 1, \dots, n - 1)$ наши элементы именованы как $(n, n - 1, \dots, 1)$).

Более простой способ генерации перестановок с корректными знаками был опубликован Э. Безу (*É. Bézout*) [*Memoires Acad. Royale des Sciences* (Paris, 1764), 292]: каждая перестановка $\pm a_1 \dots a_{n-1}$ множества $\{1, \dots, n-1\}$ дает n других, $\pm a_1 \dots \dots a_{n-1} a_n \mp a_1 \dots a_{n-2} a_n a_{n-1} \pm \dots$.

9. $(\cdot, 1, 2, 3, \xi, \circ, \natural, \vee, \wedge, \emptyset)$; или, пожалуй, мы должны сказать $(\emptyset, \wedge, \vee, \natural, \circ, \xi, \tau, \gamma, 1, \cdot)$. *Примечание:* для порядковых номеров в уравнениях использовалась иная система; например, ‘3’ означало 200. Кроме того, следует заметить, что (11) на самом деле представляет перевод работы аль-Самаваля (*al-Samaw'al*) на *современный* арабский. Ахмад и Раshed основывались на копии XIV века, в которой использовалось похожее, но более старое начертание цифр: $(\circ, 1, 2, 3, \natural, \emptyset, \wedge, \vee, \wedge, \emptyset)$. Сам аль-Самаваль мог использовать еще более древнее начертание.

10. Если все 56 вариантов равновероятны, ответ равен $56H_{56} \approx 258.2$, как в задаче о сортировании купонов (3.3.2–8). Однако (6, 30, 20) случаев имеют вероятности соответственно $(1/216, 1/72, 1/36)$. Таким образом, верным будет ответ

$$\int_0^{\infty} \left(1 - \left(1 - e^{-t/216} \right)^6 \left(1 - e^{-t/72} \right)^{30} \left(1 - e^{-t/36} \right)^{20} \right) dt \approx 546.6,$$

что составляет около 42% от верхней границы $216H_{216}$. [См. P. Flajolet, D. Gardy, and L. Thimonier, *Discrete Applied Math.*, **39** (1992), 207–229.]

11. В таблице приведены $\binom{6}{3} = 20$ сочетаний (b, c, d, B, C, D) по три; кроме того, они находятся в лексикографическом порядке, если считать $b < c < d < B < C < D$. Буква t (t) означает “переход от нижнего регистра к верхнему”. [См. A. Bonner, *Selected Works of Ramon Llull* (Princeton, 1985), 596–597.] На рисунке есть две опечатки: в начале строки 6 ‘d’ должно быть ‘b’, а в конце строки 18 ‘c’ должно быть ‘d’. Стока 1 была бы более согласованной с остальными, если бы имела вид

b	c	d	t
---	---	---	---

но в этой строке переключение регистра, конечно же, не требуется.

12. Умножьте цикл Пуансо на $5 \pmod{7}$.

13. При наличии n различных букв лучше писать только n строк.

$$\begin{array}{cccccc} a. & aa. & aaa \\ \hline b. & ab. & aab. & aaab. & bb. & abb. & aabb. \end{array}$$

Тогда присваивание весов $(a, b) = (1, 4)$ дает числа от 1 до 11, как в (18). (Первая строка (16) также должна быть опущена.) Аналогично: для $\{a, a, a, b, b, c\}$ мы можем неявно присвоить c вес 12 и добавить дополнительную строку

$$c. ac. aac. aaac. bc. abc. aabc. aaabc. bbc. abb. aabb. aaabb.$$

[Я Бернулли (*J. Bernoulli*) почти сделал это в *Ars Conjectandi*, часть 2, глава 6.]

14. ABC ABD ABE ACD ACE ACB ADE ADB ADC AEB AEC AED BCD BCE BCA BDE BDA BDC BEA BEC BED BAC BAD BAE CDE CDA CDB CEA CEB CED CAB CAD CAE CBD CBE CBA DEA DEB DEC DAB DAC DAE DBC DBE DBA

DCE DCA DCB EAB EAC EAD EBC EBD EBA ECD ECA ECB EDA EDB EDC. Это облекное упорядочение (см. алгоритм 7.2.1.3R), циклически проходящее по еще не использованным буквам. [Аналогичное упорядочение использовалось при построении всех 120 перестановок пяти букв в каббалистической работе *Sha'ari Tzedeq*, приписываемой Натану бен Саадьяху Харару (Natan ben Sa'adyah Har'ar) из Мессины, Сицилия, жившему в XIII веке; см. *Le Porte della Giustizia* (Milan : Adelphi, 2001).]

15. После j мы помещаем $(n - 1)$ -сочетания с повторениями множества $\{j, \dots, m\}$, так что ответ — $\binom{(m+1-j)+(n-1)-1}{n-1} = \binom{m+n-j-1}{n-1}$. [Жан Боррель (Jean Borrel), известный также как Бутеонис (Buteonis), указал этот результат в своей книге *Logistica* (Lyon, 1560) на с. 305–309. Он протабулировал все выбрасывания n костей для $1 \leq n \leq 4$, а затем использовал суммирование по j , чтобы вывести, что есть всего $56 + 35 + 20 + 10 + 4 + 1 = 252$ различных результата выбрасывания для $n = 5$ и 462 для $n = 6$.]

16. N1. [Инициализация.] Установить $r \leftarrow n$, $t \leftarrow 0$ и $a_0 \leftarrow 0$.

N2. [Продвижение.] Пока $r \geq q$, устанавливать $t \leftarrow t + 1$, $a_t \leftarrow q$ и $r \leftarrow r - q$. Затем, если $r > 0$, установить $t \leftarrow t + 1$ и $a_t \leftarrow r$.

N3. [Посещение.] Посетить композицию $a_1 \dots a_t$.

N4. [Поиск j .] Устанавливать $j \leftarrow t$, $t - 1$, … до тех пор, пока не будет выполнено условие $a_j \neq 1$. Завершить работу алгоритма, если $j = 0$.

N5. [Уменьшение a_j .] Установить $a_j \leftarrow a_j - 1$, $r \leftarrow t - j + 1$, $t \leftarrow j$; вернуться к шагу N2.

Например, разбиениями для $n = 7$ и $q = 3$ являются 331, 322, 3211, 313, 3121, 3112, 31111, 232, 2311, 223, 2221, 2212, 22111, 2131, 2122, 21211, 2113, 21121, 21112, 211111, 133, 1321, 1312, 13111, 1231, 1222, 12211, 1213, 12121, 12112, 121111, 1132, 11311, 1123, 11221, 11212, 112111, 11131, 11122, 111211, 11113, 111121, 111112, 1111111.

Сутры 79 и 80 Нааяны, по сути, описывают приведенную процедуру, но с обращенными строками (133, 223, 1123, …), поскольку он предпочитал уменьшающийся солексный порядок. [Ранее Сарнгадева (*Sāṅgadeva*) в *Saṅgitaratnākara* (§ 5, 316–375) детально разработал теорию для множества всех разбиений, которые могут быть образованы базовыми частями {1, 2, 4, 6}.]

17. Количество V_n посещений равно $F_{n+q-1}^{(q)} = \Theta(\alpha_q^{nm})$ (см. упражнение 5.4.2–7). Количество X_n выполнений проверки $a_j = 1$ на шаге N4 удовлетворяет уравнению $X_n = X_{n-1} + \dots + X_{n-q} + 1$, и мы находим, что $X_n = V_0 + \dots + V_n = = (qV_n + (q - 1)V_{n-1} + \dots + V_{n-q+1} - 1)/(q - 1) = \Theta(V_n)$. Количество Y_n установок $a_t \leftarrow q$ на шаге N2 удовлетворяет тому же рекуррентному соотношению, и мы находим, что $Y_n = X_{n-q}$. И наконец, условие $r = 0$ на шаге N2 выполняется V_{n-q} раз.

18. Если воспользоваться римскими цифрами, то 1666 записывается как MDCLXVI, где M > D > C > L > X > V > I.

19. Это строки 329 и 1022 (всего среди 1022 строф Путеануса данным свойством обладают 139).

20. Если ‘tria’ предшествует ‘lumina’, существует $5! \times 2! \times (11, 12, 12, 16)$ способов получить дактиль в (1, 2, 3, 4)’ й стопе соответственно. Если ‘lumina’ предшествует

‘tria’, таких способов $5! \times 2! \times (16, 12, 12, 11)$. Таким образом, общее количество перестановок — 24 480. [Лейбниц рассматривал эту задачу в конце своей *Dissertatio de Arte Combinatoria* и получил ответ 45 870; однако его аргументация переполнена ошибками.]

21. (a) Ясно, что производящая функция $1/((1 - zu - yu^2)(1 - zv - yv^2)(1 - zw - yw^2))$ равна $\sum_{p,q,r,s,t \geq 0} f(p, q, r; s, t) u^p v^q w^r z^s y^t$.

(b) Если ‘tibi’ соответствует $\sim\sim$, а ‘Virgo’ представляет собой $\sim\sim$, то общее количество равно $3!3!$, умноженному на $\sum_{k=0}^3 (f(2k+1, 6-2k, 2; 3, 3) + f(2k, 6-2k, 2; 2, 3))$, т.е. $36((7+7)+(9+5)+(10+5)+(14+7)) = 2304$. В противном случае ‘tibi’ соответствует $\sim\sim$, ‘Virgo’ представляет собой $\sim\sim$, а общее количество равно $2!3!$, умноженному на $\sum_{k=0}^3 (f(2k, 5-2k, 2; 3, 2) + f(2k, 6-2k, 1; 3, 2))$, т.е. $12((7+6)+(5+4)+(4+4)+(0+6)) = 432$.

(c) Пятая стопа начинается со второго слога ‘cælo’, ‘dotes’ или ‘Virgo’. Следовательно, дополнительное количество равно $3!3! \sum_{k=0}^2 f(2k, 5-2k, 2; 3, 2) = 36(7+5+4) = 576$, и общее количество гекзаметров среди перестановок составляет $2304 + 432 + 576 = 3312$.

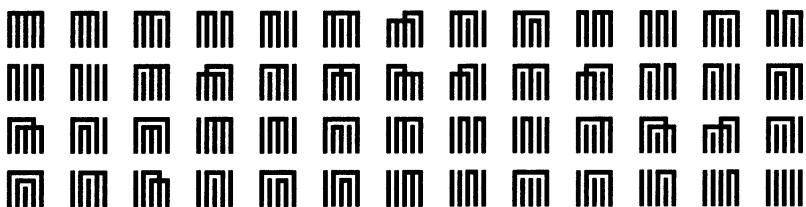
22. Пусть $\alpha \in \{\text{quot, sunt, tot}\}$, $\beta \in \{\text{cælo, dotes, Virgo}\}$, $\sigma = \text{sidera}$ и $\tau = \text{tibi}$. Анализ Престета, по сути, эквивалентен анализу Бернулли, но он забыл включить 36 случаев $\alpha\alpha\sigma\tau\beta\beta\sigma\beta$. (В его защиту можно сказать, что эти случаи бесплодны с точки зрения поэзии; Путеанус не нашел им применения.) В издании 1675 года книги Престета опущены также все перестановки, заканчивающиеся на $\tau\beta$.

Уоллис разделил все возможности на 23 типа: $T_1 \cup T_2 \cup \dots \cup T_{23}$. Он заявил, что типы 6 и 7 каждый дают по 324 строфы, однако в действительности $|T_6| = |T_7| = 252$, поскольку его переменная i должна быть равна 7, а не 9. Кроме того, многие решения посчитаны им дважды: $|T_3 \cap T_5| = 252$, $|T_2 \cap T_7| = |T_5 \cap T_7| = |T_3 \cap T_6| = |T_6 \cap T_{10}| = 36$ и $|T_{11} \cap T_{12}| = |T_{12} \cap T_{13}| = |T_{14} \cap T_{15}| = 12$. Он пропустил 36 вариантов $\alpha\beta\beta\alpha\sigma\sigma\tau\beta$ (19 из которых использовал Путеанус). Он также пропустил все перестановки из упражнения 21 (c); Путеанус использовал 250 из их общего количества 576. В латинском издании книги Уоллиса, опубликованном в 1693 году, был исправлен ряд типографских ошибок, но не ошибки математические.

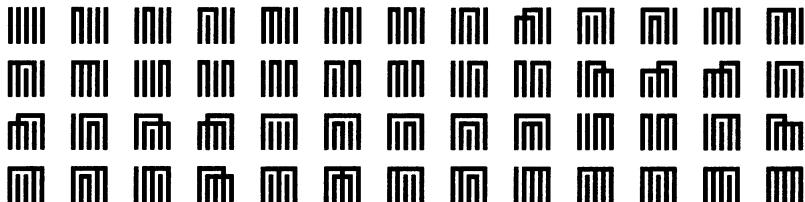
Уитворт и Хартли пропустили все случаи, когда ‘tibi’ = $\sim\sim$ (см. упражнение 19), возможно потому, что количество людей, знающих классический гекзаметр, постоянно уменьшается.

[Кстати, об ошибках: на самом деле Путеанус опубликовал только 1020, а не 1022 различных перестановок, поскольку строки 592 и 593 в его списке идентичны строкам 601 и 602. Однако у него не должно было возникнуть проблем с поиском еще двух гекзаметров — их можно получить, например, заменой ‘tot sunt’ на ‘sunt tot’ в строках 252, 345, 511, 548, 659, 663, 678, 693 или 797.]

23. Читая каждую диаграмму слева направо, так что $12 \mid 345 \leftrightarrow \text{III}$, получаем:



24. Вот его правило: для $k = 0, 1, \dots, n - 1$ и для каждого сочетания $0 < j_1 < \dots < j_k < n$ из $n - 1$ элементов по k посещаем все разбиения $\{1, \dots, n - 1\} \setminus \{j_1, \dots, j_k\}$ вместе с блоком $\{j_1, \dots, j_k, n\}$. Для $n = 5$ получается следующий порядок:



Однако, строго говоря, ответ к данному упражнению — “Нет”, поскольку правило Хонды неполное, пока не указан порядок сочетаний. Он генерировал сочетания в *лексикографическом* порядке (лексикографическом порядке $j_t \dots j_1$). Лексикографический порядок $j_1 \dots j_t$ также согласуется с приведенным списком для $n = 4$, но при этом располагается перед . Источник: T. Hayashi, *Tohoku Math. J.*, **33** (1931), 332–337.

25. Нет, в (28) отсутствует схема $14 | 235$ (представляющая собой зеркальное отражение схемы 2 относительно горизонтальной оси).

26. Пусть a_n — количество неприводимых разбиений $\{1, \dots, n\}$, и пусть a'_n — количество одновременно неприводимых и полных разбиений. Эти последовательности начинаются с $\langle a_1, a_2, \dots \rangle = \langle 1, 1, 2, 6, 22, 92, 426, \dots \rangle$ и $\langle a'_1, a'_2, \dots \rangle = \langle 0, 1, 1, 3, 9, 33, 135, \dots \rangle$. Ответом к данному упражнению является $a'_n - 1$ для $n \geq 2$. Также оказывается, что a_n является количеством симметричных полиномов степени n от некоммутирующих переменных [см. M.C. Wolf, *Duke Math. J.*, **2** (1936), 626–637, где также протабулированы неприводимые разбиения на k частей.]

Если $A(z) = \sum_n a_n z^n$ и $B(z) = \sum_n \varpi_n z^n$ — неэкспоненциальная генерирующая функция для чисел Белла, то $A(z)B(z) = B(z) - 1$; следовательно, $A(z) = 1 - 1/B(z)$. Из результата упражнения 7.2.1.5–35 вытекает, что $\sum_n a'_n z^n = zA(z)/(1 + z - A(z)) = z(B(z) - 1)/(1 + zB(z))$. К сожалению, $B(z)$ не имеет красивого аналитического вида, хотя и удовлетворяет интересному функциональному соотношению $1 + zB(z) = B(z/(1 + z))$. Заметим, что неприводимые разбиения множества с $n > 1$ соответствуют “циклам колеблющейся диаграммы”, в которых не имеется трех последовательных λ , равных нулю (см. упражнение 7.2.1.5–27).

27. Задача поставлена неоднозначно, поскольку нет точного определения диаграмм генджи-ко. Потребуем, чтобы все вертикальные линии одного блока имели одинаковую длину; при таком ограничении, например, разбиение $145|236$ не имеет диаграммы генджи-ко с одним пересечением, поскольку диаграмма не разрешена.

Количество разбиений без пересечений равно C_n (см. упражнение 7.2.1.6–26). Для наличия одного пересечения элементы пересекающихся блоков должны находиться в ограниченном растущей последовательности $x^i y x^j y^k$, $x^i y^{j+1} x y^k$ или $x^i y^j x y^k x^l$ где $i, j, k, l > 0$.

Предположим, что мы рассматриваем шаблон $x^i y x^j y^k$. Имеется $t = i + j + k + 2$ “слотов” между $i + 1 + j + k$ элементами данного шаблона, и количество способов заполнения их непересекающимися разбиениями равно $\sum_{i_1 + \dots + i_t = n - i - j - k - 1} C_{i_1} \dots C_{i_t}$.

Это число можно выразить как

$$[z^{n-i-j-k-1}] C(z)^{i+j+k+2} = C_{(n-i-j-k-1)n}$$

согласно 7.2.1.6–(24). Суммирование по k дает $C_{(n-i-j-2)(n+1)}$; последующее суммирование по j и i дает $C_{(n-4)(n-3)}$.

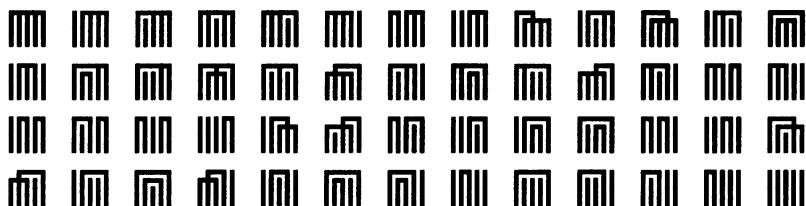
Аналогично: два других шаблона дают вклад $C_{(n-5)(n+3)}$ и $C_{(n-5)(n+4)}$. Таким образом, общее количество разбиений с одним пересечением равно $C_{(n-5)(n+3)} + C_{(n-4)(n+4)}$.

28. Упорядочим делители $cbbaaa$ по количеству их простых сомножителей, а затем в солексикографическом порядке: $1 \prec a \prec b \prec c \prec aa \prec ba \prec ca \prec bb \prec cb \prec aaa \prec baa \prec caa \prec bba \prec cba \prec cbb \prec baaa \prec caaa \prec bbaa \prec cbba \prec bbba \prec cbaaa \prec cbaaa \prec cbbaa \prec cbbaaa$.

Для каждого такого делителя d в уменьшающемся порядке считаем первым сомножителем d и рекурсивно продолжаем разложение числа $cbbaaa/d$, первый множитель которого $\preccurlyeq d$.

Если делители упорядочены лексикографически (т.е. $1 < a < aa < aaa < b < ba < \dots < cbbaa < cbaaa$), алгоритм Уоллиса становится эквивалентен алгоритму 7.2.1.5М с $(n_1, n_2, n_3) = (1, 2, 3)$. Вероятно, этот более сложный порядок делителей выбран в связи с тем, что он более близок к обычному порядку чисел при $a \approx b \approx c$; например, при $(a, b, c) = (7, 11, 13)$ получается точный числовой порядок. Генерируя делители в соответствии с этой более сложной схемой, Уоллис, по сути, генерировал сочетания мульти множества, которые, как отмечалось в разделе 7.2.1.3, эквивалентны ограниченным композициям. [Источник: *Discourse of Combinations* (1685), 126–128, с двумя исправленными опечатками.]

29. Разложения $edcba$, $edcb \cdot a$, $edca \cdot b$, \dots , $e \cdot d \cdot c \cdot b \cdot a$ соответствуют



30. Коэффициент равен 0, кроме случаев, когда $i_1 + 2i_2 + \dots = n$; тогда он равен $\binom{m}{k} a_0^{m-k} \binom{k}{i_1, i_2, \dots}$, где $k = i_1 + i_2 + \dots$. (Рассмотрите $(a_0 z)^m$, умноженное на $(1 + (a_1/a_0) z + (a_2/a_0) z^2 + \dots)^m$.)

31. Порядок, генерируемый этим алгоритмом — уменьшающийся лексикографический, обратный разбиению (31), если считать, что у разбиений $a_1 \dots a_k$ $a_1 \geq \dots \geq a_k$; порядок де Муавра — возрастающий *солексикографический*.

32. $20 \cdot 1 = 7 + 13 \cdot 1 = 2 \cdot 7 + 6 \cdot 1 = 10 + 10 \cdot 1 = 10 + 7 + 3 \cdot 1 = 2 \cdot 10$. В общем случае Бонкович предложил начинать с $n \cdot 1$ и вычислять следующий элемент $a \cdot 10 + b \cdot 7 + c \cdot 1$ таким образом: если $c \geq 7$, следующий элемент представляет собой $a \cdot 10 + (b + 1) \cdot 7 + (c - 7) \cdot 1$; иначе если $c + 7b \geq 10$, следующий элемент — $(a + 1) \cdot 10 + (c + 7b - 10) \cdot 1$; в противном случае завершаем работу.

Предметный указатель

- Adachi, Fumie, 86
Ahmad, Salah, 74; 141
al-Samaw' al, ibn Yahyā ibn Yahūda
 al-Maghribī, 74; 141
Aldous, David John, 27
Alonso, Laurent, 122
Arima, Yoriyuki, 87
Aristotle of Stagira, son of Nicomachus
 (Αριστοτέλης Νικομάχου ὁ
 Σταγείριτης), 78
Aristoxenus (Ἀριστόξενος), 71
Arnold, David Bryan, 25
Arques, Didier, 138
Atkinson, Michael David, 121

Baldéric, 75
Bauhuis, Bernard, 83
Beckenbach, Edwin Ford, 93
Becker, Harold W., 107
Bell, Eric Temple, 88; 144
Bellhouse, David Richard, 76
Bellman, Richard Ernest, 138
Bernoulli, Jacques (Jakob, James), 67; 81; 85;
 141; 143
Beyer, Wendell Terry, 34
Bézout, Étienne, 141
Bhattotpala, 82
Bhāskara II, Ācārya, son of Māheśvara, 72; 74
Bonner, Anthony Edmonde, 141
Borgne, Yvan Françoise Andre, Le, 132
Borrel, Jean (=Buteonis, Ioannes), 142
Bošković, Ruđer Josip (Бошковић, Руђер
 Јосип), 90; 98; 145
Brown, Charles Philip, 70
Bruijn, Nicolaas Govert de, 30; 70
Burkhardt, Johann Carl, 91
Buteonis, Ioannes (=Borrel, Jean), 142

Callan, Columcille David, 101
Cantor, Moritz Benedikt, 91
Carlitz, Leonard, 116

Catalan, Eugène Charles, 22; 23; 32; 92
Cayley, Arthur, 92
Christ, Wilhelm von, 71
Chung, Kai Lai, 121
Cori, Robert, 132
Cummins, Richard Lee, 128
Cvetković, Dragos Mladen (Цветковић,
 Драгош Младен), 42; 135

Dénes, József, 111
Deo, Narsingh, 93
Dershowitz, Nachum, 110
Deutsch, Emeric, 123
Dewey, Melvil, 24
Dhar, Deepak, 132
Diomedes (Διομήδης), 71
Donnolo, Shabbetai ben Avraham, 71; 95
Doob, Michael, 135
Drexel, Jeremias, 71
Dudeney, Henry Ernest, 64; 137
Dyck, Walther Franz Anton von, 93

Edelman, Paul Henry, 110
Er, Meng Chiau, 103
Erdős, Pal, 92
Errera, Alfred, 100
Ettingshausen, Andreas von, 91
Euler, Leonhard, 58; 90
Eyrolles, Georges, 138

Feller, Willibald, 121
Feussner, Wilhelm, 36
Fibonacci, Leonardo, of Pisa; Leonardo filio
 Bonacci Pisano, 69
Fiedler, Miroslav, 135
Flajolet, Philippe Patrick Michel, 141
Fleischner, Herbert, 137
Freese, Ralph Stanley, 110

Gabow, Harold Neil, 130
Galilei, Galileo, 88

- Gardner, Martin, 80
 Gardy, Daniele, 141
 Gershwin, George, 80
 Golle, Philippe, 59; 127
 Grätzer, George, 109
 Greene, Curtis, 32; 125
 Gutjahr, Walter Josef, 120
 Hakimi, Seifollah Louis, 94
 Hammond, Eleanor Prescott, 76
 Hansel, Georges, 33; 59
 Har'ar, Natan ben Sa'adyah, 142
 Harary, Frank, 92; 128; 137
 Hartley, William Ernest, 85; 97; 143
 Hayashi, Tsuruichi, 144
 Hedetniemi, Sarah Lee Mitchell, 34
 Hickerson, Dean Robert, 53
 Hindenburg, Carl Friedrich, 90
 Hodges, Joseph Lawson Jr., 121
 Holzmann Poisson, Carlos Alfonso, 128
 Homer ("Ομηρος"), 83
 Honda, Toshiaki, 87; 97; 144
 Horton, Robert Elmer, 65; 138
 Ibn Mun'im, 81
 Idel, Moshe, 79
 Izquierdo, Sebastián, 81; 96
 Janey, Nicolas, 138
 Kak, Subhash Chandra, 70
 Kaplansky, Irving, 92
 Kapoor, Sanjiv, 130
 Katona, Gyula (Optimalis Halmaz), 125
 Kedāra, Bhaṭṭa, 69; 89
 Keil, Heinrich, 71
 Kemp, Rainer, 117; 119
 Kircher, Athanasius, 71; 72; 79; 95; 140
 Kirschenhofer, Peter, 120
 Klee, Victor La Rue, Jr., 7
 Kleitman, Daniel J. (Isaiah Solomon), 32; 59;
 125
 Kleppis, Gregor (=Kleppisius, Gregorius), 96
 Klugel, Georg Simon, 91
 Knobloch, Eberhard Heinrich, 88; 89
 Knott, Gary Don, 94
 Knuth, Donald Ervin, 8
 Korsh, James F., 17; 48; 104; 122
 Kreher, Donald Lawson, 94
 Kreweras, Germain, 48; 50; 52; 107; 109; 135
 Kronecker, Leopold, 133
 Kronk, Hudson Van Etten, 137
 Kruskal, Joseph Bernard, Jr., 54
 Kruyswijk, Dirk, 30
 Kubicka, Ewa, 127
 Kusuba, Takanori, 82
 Lakser, Harry, 109
 Lambert, Johann Heinrich, 90
 Lehmer, Derrick Henry, 93
 Leibniz, Gottfried Wilhelm, Freiherr von, 68;
 84; 85; 87; 88; 96; 143
 Leske, Marie, 138
 Lévy, Paul, 27
 Li, Gang, 127
 Littlewood, John Edensor, 126
 Llull, Ramon (=Lullus, Raimundus), 76; 95
 London, John, 78
 Louche, Guy, 27
 Lucas, Joan Marie, 20
 Luczak (=Luczak), Małwina Joanna, 122
 Ludus Clericalis, 75; 95
 Lydgate, John, 76
 Lynn, Richard John, 68
 MacMahon, Percy Alexander, 115; 121
 Marckert, Jean-Francois, 27
 Markowsky, George, 109
 Matsunaga, Yoshisuke, 87
 Mayeda, Wataru, 94
 McLean, Iain Sinclair, 78
 Mersenne, Marin, 71; 72; 88; 95; 140
 Mikami, Yoshio, 74
 Mittag-Leffler, Magnus Gosta, 91
 Mohar, Bojan, 135
 Moivre, Abraham de, 23; 89; 90; 98; 145
 Montmort, Pierre Remond de, 89; 98
 Morse, Samuel Finley Breese, 69; 91
 Murasaki Shikibu, 86
 Myers, Eugene Wimberly, Jr., 130
 Nakagawa, Noriyuki, 94
 Nārāyaṇa Pañḍita, son of Nṛsimha, 69; 73;
 82; 89; 90; 96
 Needham, Joseph, 68
 Netto, Otto Erwin Johannes Eugen, 91–93
 Neuman, Frantisek, 64
 Nievergelt, Jurg, 93
 Nijenhuis, Albert, 60; 93
 Nooten, Barend Adrian Anske Johannes van,
 69
 Nylan, Michael, 68

- Offord, Albert Cyril, 126
 Ord-Smith, Richard Albert James, 91
 Panholzer, Alois, 120; 122
 Piṅgala, Ācārya, 68; 70
 Poinsot, Louis, 95
 Prestet, Jean, 84; 97; 143
 Prins, Geert Caleb Ernst, 92
 Prodinger, Helmut, 119; 120; 122
 Proskurowski, Andrzej, 19; 54
 Ptolemy, Claudius, of Alexandria (Πτολεμαῖος Κλαύδιος ὁ Ἀλεξανδρινός), 83
 Puteanus, Erycius, 83; 96; 142; 143
 Puttenham, Georg and/or Richard, 87
 Ramesh, Hariharan, 130
 Raney, George Neal, 56
 Rashed, Roshdi (=Rashid, Rushdi), 74; 141
 Reingold, Edward Martin, 93
 Rémy, Jean-Luc, 28; 122
 Richards, Dana Scott, 48
 Riordan, John, 116
 Robinson, Gilbert de Beauregard, 58
 Rodrigues, Benjamin Olinde, 28
 Roelants van Baronaigien, Dominique, 19; 20; 113
 Rossin, Dominique Gilles, 132
 Rothe, Heinrich August, 91
 Ruskey, Frank, 8; 19; 20; 25; 27; 54; 94; 113; 127
 Sachs, Horst, 133; 135
 Sack, Jorg-Rudiger Wolfgang, 121
 Saka, Masanobu, 87
 Śāringadeva, son of Soḍhaladeva, 73; 82; 142
 Savage, Carla Diane, 94
 Scaliger, Giulio (=Scaliger, Julius Caesar), 83
 Schensted, Craige Eugene, 58
 Schillinger, Joseph Moiseyevich, 80; 95
 Schooten, Frans van, 80; 96
 Schott, Rene Pierre, 122
 Schröder, Ernst, 57; 137
 Schwenk, Allen John, 137
 Scoins, Hubert Ian, 34; 94
 Sedgewick, Robert, 93
 Sekanina, Milan, 44
 Seki, Takakazu, 73; 87; 95
 Sembra, Ichiro, 14
 Shao Yung, 68
 Sims, Charles Coffin, 140
 Singh, Parmanand, 69; 82
 Skarbek, Władysław Kazimierz, 17
 Sleep, Michael Ronan, 25
 Smith, Malcolm James, 36; 41
 Sperner, Emanuel, 30; 57; 125
 Sprugnoli, Renzo, 55
 Stanley, Richard Peter, 12; 50; 53; 81; 123
 Stinson, Douglas Robert, 94
 Stirling, James, 87; 88
 Strahler, Arthur Newell, 65; 138
 Suśruta, 74
 Swetz, Frank Joseph, 68
 Tacquet, Andre, 80
 Tamari, Dov, 49; 51; 52; 107; 109
 Tang, Changjie, 19
 Tartaglia, Niccolò Fontana, 81
 Tengbergen, Cornelia van Ebbenhorst, 30
 Thimonier, Loys, 141
 Tompkins, Charles Brown, 93
 Tyler, Douglas Blaine, 53
 Ushijima, Kazuo, 19
 Vergil (=Publius Vergilius Maro), 83
 Viennot, Gerard Michel Francois Xavier, 138
 Vo, Kiem-Phong, 125
 Wallis, John, 73; 82; 85; 88; 97; 143; 145
 Warren, Jon, 27
 Watanabe, Hitoshi, 94
 Watson, George Neville, 116
 Wells, Mark Brimhall, 93; 94
 Wen, King of Chou, 67; 68; 95
 Whitworth, William Allen, 85; 97; 143
 Wibold (=Wiboldus, Cameracensis episcopus), 75; 88
 Wilf, Herbert Saul, 60; 93
 Winkler, Peter, 122
 Wolf, Margarete Caroline, 144
 Xiang, Limin, 19
 Yang Hsiung, 68; 69
 Yano, Tamaki, 86; 87
 Yuong, Alfred, 111
 Zaks, Shmuel, 45; 48; 103; 110
 Ziegler, Günter Matthias, 108
 Адачи, Фуми (Adachi, Fumie), 86

- Алгоритм**
- Вложенные скобки**
 - близкие к идеальным, 18
 - в лексикографическом порядке, 14
 - в представлении с сочетаниями, 99
 - Все оставные деревья, 38
 - усовершенствования, 41
 - Генерации
 - бинарных деревьев, 16
 - всех тернарных деревьев, 103
 - композиций, 142
 - случайного ориентированного дерева, 127
 - Генерация
 - бинарных деревьев в солексном порядке, 104
 - крайних справа элементов шаблона рождественской елки, 124
 - последовательностей степеней лесов с N узлами, 103
 - Закса-Ричардса, 53
 - Неранжированная строка вложенных скобок, 25
 - Обрезки и прививки, 22; 105
 - Определение ранга строки вложенных скобок, 117
 - Поиск начального оставного дерева, 128
 - Проверки моста, 128
 - Прямо-обратный преемник в трижды связанном лесу, 44
 - Равномерно распределенные случайные строки вложенных скобок, 25
 - Растущее случайное бинарное дерево, 29
 - Связанные бинарные деревья путем поворотов, 20
 - Алонсо, Лорент (Alonso, Laurent), 122
 - Аль-Самаваль, ибн Яхъя ибн Яхуда аль-Магриби (al-Samaw'al, ibn Yaḥyā ibn Yahūda al-Maghribī), 74; 141
 - Альдус, Дэвид Джон (Aldous, David John), 27
 - Амфибрахий, 70
 - Амфимакр, 70; 71
 - Анапест, 70
 - Антибакхий, 70
 - Антиспаст, 70
 - Арабские цифры, 95; 141
 - Аrima, Ёриюки (Arima, Yoriyuki), 87
 - Аристоксен (Aristoxenus ('Αριστόξενος)), 71
 - Аристотель из Стагиры, сын Никомаха (Aristotle of Stagira, son of Nicomachus (Αριστοτέλης Νικομάχου ὁ Σταγιρίτης)), 78
 - Аркю, Дидье (Arques, Didier), 138
 - Арнольд, Дэвид Брайан (Arnold, David Bryan), 25
 - Арсис, 70
 - Ассоциаэрд, 51
 - Аткинсон, Майкл Дэвид (Atkinson, Michael David), 121
 - Атомарная строка, 55
 - Ахмад, Салах (Ahmad, Salah), 74; 141
 - Бакхий, 70
 - Балдерик (Baldéric), 75
 - Баллотировочные числа, 23; 53
 - Баухуис, Бернард (Bauhuis, Bernard), 83
 - Безу, Этьен (Bézout, Étienne), 141
 - Бейер, Венделл Терри (Beyer, Wendell Terry), 34
 - Бекенбах, Эрвин Форд (Beckenbach, Edwin Ford), 93
 - Беккер, Гарольд В. (Becker, Harold W.), 107
 - Белл, Эрик Темпль (Bell, Eric Temple), 88; 144
 - Белла числа, 88; 144
 - Беллман, Ричард Эрнест (Bellman, Richard Ernest), 138
 - Беллхаус, Дэвид Ричард (Bellhouse, David Richard), 76
 - Бернулли, Якоб (Bernoulli, Jacques (Jakob, James)), 67; 81; 85; 141; 143
 - Биклаттер, 58
 - Бинарное дерево поиска, 54
 - Бинарное случайное дерево, 27
 - Битовая строка, 30
 - Близкое дерево, 36; 40
 - Боннер, Энтони Эдмонд (Bonner, Anthony Edmonde), 141
 - Борне, Иван Франсуа Андре, Ле (Borgne, Yvan Françoise Andre, Le), 132
 - Боррель, Жан (Borrel, Jean (=Buteonis, Ioannes)), 142
 - Бошкович, Ругер Јосип (Bošković, Ruđer Bošković, Ruđer Josip), 90; 98; 145
 - Браун, Чарльз Филип (Brown, Charles Philip), 70

- Брейн, Николаас Говерт де (Bruijn, Nicolaas Govert de), 30; 70
- Брейна де цикл, 70
- Буркхардт, Иоганн Карл (Burkhardt, Johann Carl), 91
- Бутеонис, Иоанн (Buteonis, Ioannes (=Borrel, Jean)), 142
- Бхаскара II, Акарья, сын Махесвары (Bhāskara II, Ācārya, son of Māheśvara), 72; 74
- Бхаттопала (Bhaṭṭotpala), 82
- Ватанабе, Хитоши (Watanabe, Hitoshi), 94
- Watson, Джордж Невилл (Watson, George Neville), 116
- Ваховский, Евгений Борисович, 133
- Ведическая строфа, 68
- Вень, Кинг из Чу (Wen, King of Chou), 67; 68; 95
- Вергилий, Публий Марий (Vergil (=Publius Vergilius Maro)), 83
- Вибольд, епископ Камбрейский (Wibold (=Wiboldus, Cameracensis episcopus)), 75; 88
- Вильф, Герберт Саул (Wilf, Herbert Saul), 60; 93
- Вложенные скобки, 13
- Корректность, 14
 - Представления, 15
 - Случайные, 25
- Vo, Ким-Фонг (Vo, Kiem-Phong), 125
- Вольф, Маргарет Керолайн (Wolf, Margarete Caroline), 144
- Вырожденное бинарное дерево, 52; 54
- Виенно, Жерар Мишель Франсуа Ксавье (Viennot, Gerard Michel Francois Xavier), 138
- Габов, Гарольд Нейл (Gabow, Harold Neil), 130
- Галилей, Галилео (Galilei, Galileo), 88
- Гарди, Даниэль (Gardy, Daniele), 141
- Гарднер, Мартин (Gardner, Martin), 80
- Гекзаметр, 83; 96
- Гексаграмма, 67; 139
- Генджи-ко, 85
- Генерация деревьев, 91
- Генетический код, 94
- Гershwin, Джордж (Gershwin, George), 80
- Гинденбург, Карл Фридрих (Hindenburg, Carl Friedrich), 90
- Гиперпиррихий, 70
- Голль, Филипп (Golle, Philippe), 59; 127
- Головоломка Дьюдени, 64
- Гомер (Homer ('Одиссея')), 83
- Граф
- Двудольный, 79
 - Дуальный планарный, 129
 - Последовательно-параллельный, 39
 - Производная, 64
- Греческая поэзия, 70
- Грин, Кертис (Greene, Curtis), 32; 125
- Грятцер, Георг (Grätzer, George), 109
- Гусеница, 137
- Гутъяр, Уолтер Джозеф (Gutjahr, Walter Josef), 120
- Дактиль, 70
- Двенадцатизадачие, 81
- Двудольный граф, 79
- Декорированное бинарное дерево, 28
- Денеш, Йозеф (Dénés, József), 111
- Део, Нарсингх (Deo, Narsingh), 93
- Дерево, 12
- Бинарное вырожденное, 52; 54
 - Бинарное декорированное, 28; 56
 - Бинарное случайное, 27
 - Близкое, 36; 40
 - Естественное соответствие, 14
 - Каталана случайное, 28
 - Коды Грея, 17
 - идеальные, 19 - Количество деревьев, 22
 - Ориентированное, 34
 - Ориентированное оствное, 61
 - Оствное, 35; 38
 - Поиска бинарное, 54
 - Порядок обхода
 - обратно-прямой, 43
 - прямо-обратный, 43
 - симметричный, 14 - Расширенное бинарное, 47
 - Расширенное тернарное, 47
 - Скрученное биномиальное, 64
 - Случайные деревья, 25
 - Шрёдера, 57
- Дершвиц, Наум (Dershovitz, Nachum), 110
- Десятичная запись Дьюи, 24
- Дефект сбалансированной строки, 55
- Диаграмма
- Юнга, 53; 111; 114

- Дик, Вальтер Франц Антон фон (Dyck, Walther Franz Anton von), 93
- Дика слова, 93
- Диомед (Diomedes ($\Deltaιομήδης$)), 71
- Диспондей, 70
- Дитрохей, 70
- Дихорей, 70
- Диямб, 70
- Добродетели, 75; 77; 78
- Дойч, Эмерик (Deutsch, Emeric), 123
- Донноло, Шаббетаи ибн Авраам (Donnolo, Shabbetai ben Avraham), 71; 95
- Дрексель, Иеремия (Drexel, Jeremias), 71
- Дуальный лес, 47; 103
- Дуб, Майкл (Doob, Michael), 135
- Дхар, Дипак (Dhar, Deepak), 132
- Дьюдени, Генри Эрнст (Dudeney, Henry Ernest), 64; 137
- Дьюдени головоломка, 64
- Дьюи десятичная запись, 24
- Дьюи, Мелвил (Dewey, Melvil), 24
- Жане, Николя (Janey, Nicolas), 138
- Задача точного покрытия, 85
- Закс, Шмуль (Zaks, Shmuel), 45; 48; 103; 110
- Закса–Ричардса алгоритм, 53
- Иbn Муним (Ibn Mun‘im), 81
- Игровые кости, 74
- Идель, Моше (Idel, Moshe), 79
- Изкуэрдо, Себастьян (Izquierdo, Sebastián), 81; 96
- Индийская поэзия, 68
- Ионик, 70
- Исследование монотонной булевой функции, 33
- Кайл, Генрих (Keil, Heinrich), 71
- Как, Сабхаш Чандра (Kak, Subhash Chandra), 70
- Каллан, Колумсиль Дэвид (Callan, Columcille David), 101
- Камминс, Ричард Ли (Cummins, Richard Lee), 128
- Канонический лес, 34
- Кантор, Мориц Бенедикт (Cantor, Moritz Benedikt), 91
- Каплански, Ирвинг (Kaplansky, Irving), 92
- Капур, Санджив (Kapoor, Sanjiv), 130
- Карлиц, Леонард (Carlitz, Leonard), 116
- Каталан, Эжен Шарль (Catalan, Eugène Charles), 22; 23; 32; 92
- Каталана треугольник, 23; 25; 32
- Каталана числа, 22; 23; 32; 53; 92; 115
- Обобщенные, 53
- Катона, Гюла (Katona, Gyula (Optimalis Halmaz)), 125
- Кедара, Бхатта (Kedāra, Bhaṭṭa), 69; 89
- Кейли, Артур (Cayley, Arthur), 92
- Кельманс, Александр Кольманович, 135
- Кемп, Райнер (Kemp, Rainer), 117; 119
- Кирхер, Атанасиус (Kircher, Athanasius), 71; 72; 79; 95; 140
- Киршенхофер, Петер (Kirschenhofer, Peter), 120
- Клаттер, 30
- Клеппис, Грегор (Kleppis, Gregor (=Kleppisius, Gregorius)), 96
- Кли, Виктор ла Руе, мл. (Klee, Victor La Rue, Jr.), 7
- Клюгель, Георг Симон (Klugel, Georg Simon), 91
- Кляйтман, Даниэль (Kleitman, Daniel J. (Isaiah Solomon)), 32; 59; 125
- Кноблох, Эбергард Генрих (Knobloch, Eberhard Heinrich), 88; 89
- Кнотт, Гари Дон (Knott, Gary Don), 94
- Кнут, Дональд Эрвин (Knuth, Donald Ervin), 8
- Код
- Генетический, 94
 - Морзе, 69
- Количества деревьев, 22
- Композиция, 69; 74; 96; 145
- Контур леса, 26
- Кори, Роберт (Cori, Robert), 132
- Коробков, Виталий Константинович, 34
- Корш, Джеймс Ф. (Korsh, James F.), 17; 48; 104; 122
- Косточка, Александр Васильевич, 136
- Креверас, Герман (Kreweras, Germain), 48; 50; 52; 107; 109; 135
- Кревераса решетка, 48; 50; 52; 109
- Кретик, 71
- Крехер, Дональд Лоусон (Kreher, Donald Lawson), 94
- Крист, Вильгельм фон (Christ, Wilhelm von), 71
- Кронекер, Леопольд (Kronecker, Leopold), 133
- Кронекера произведение, 133

- Кронк, Хадсон Ван Эттен (Kronk, Hudson Van Etten), 137
- Круйсвейк, Дирк (Kruyswijk, Dirk), 30
- Крускал, Иосиф Бернард мл. (Kruskal, Joseph Bernard, Jr.), 54
- Крускала функция, 54
- Ксянг, Лимин (Xiang, Limin), 19
- Кубицка, Ева (Kubicka, Ewa), 127
- Кусуба, Таканори (Kusuba, Takanori), 82
- Лаксер, Гарри (Lakser, Harry), 109
- Ламберт, Иоганн Генрих (Lambert, Johann Heinrich), 90
- Леви, Поль (Lévy, Paul), 27
- Лейбниц, Готтфрид Вильгельм, Фрайхерр фон (Leibniz, Gottfried Wilhelm, Freiherr von), 68; 84; 85; 87; 88; 96; 143
- Лемер, Деррик Генри (Lehmer, Derrick Henry), 93
- Лемма
Рани, 56
- Лес
Дуальный, 47; 103
Канонический, 34; 59
Непомеченый, 47
Ориентированный, 34; 59
Родственный, 47
След, 51
Сопряженный, 46
Транспонированный, 46
Упорядоченный, 34
- Леске, Мари (Leske, Marie), 138
- Ли, Ганг (Li, Gang), 127
- Лидгейт, Джон (Lydgate, John), 76
- Линн, Ричард Джон (Lynn, Richard John), 68
- Литлвуд, Джон Эденсор (Littlewood, John Edensor), 126
- Лондон, Джон (London, John), 78
- Лукас, Джоан Мари (Lucas, Joan Marie), 20
- Луллий, Раймон (Lull, Ramon (=Lullus, Raimundus)), 76; 95
- Лучак, Мальвина Джоанна (Luczak (=Luczak), Malwina Joanna), 122
- Лучард, Гай (Louchard, Guy), 27
- Майдэда, Ватару (Mayeda, Wataru), 94
- Майерс, Юджин Вимберли мл. (Myers, Eugene Wimberly, Jr.), 130
- Мак-Мэган, Перси Александр (MacMahon, Percy Alexander), 115; 121
- Маклин, Ян Синклер (McLean, Iain Sinclair), 78
- Максимальная цепочка решетки, 50
- Маркерт, Жан-Франсуа (Marckert, Jean-Francois), 27
- Марковски, Джордж (Markowsky, George), 109
- Мацунага, Ёшикуе (Matsunaga, Yoshisuke), 87
- Мерсенне, Марин (Mersenne, Marin), 71; 72; 88; 95; 140
- Метод
С возвратом, 85
Танующих связей, 38
- Метрическая стопа, 70
- Миками, Ёшио (Mikami, Yoshio), 74
- Миттаг-Леффлер, Магнус Геста (Mittag-Leffler, Magnus Gosta), 91
- Молосс, 70
- Монморт, Пьер Ремонд де (Montmort, Pierre Remond de), 89; 98
- Морзе, Сэмюэль Финли Бриз (Morse, Samuel Finley Breese), 69; 91
- Морзе, код, 91
- Мост, 38
- Мохар, Боян (Mohar, Bojan), 135
- Муавр, Абрахам де (Moivre, Abraham de), 23; 89; 90; 98; 145
- Мультиномиальный коэффициент, 88
- Мупасаки Шикибу (Murasaki Shikibu), 86
- Найлан, Майкл (Nylan, Michael), 68
- Накагава, Нориюки (Nakagawa, Noriyuki), 94
- Наравана Пандита, сын Нрсимха (Nāravāna Paññita, son of Nṛsiṁha), 69; 73; 82; 89; 90; 96
- Непересекающееся разбиение, 107
- Непомеченный лес, 47
- Нетто, Отто Эрвин Иоганнес Эжен (Netto, Otto Erwin Johannes Eugen), 91–93
- Нивергельт, Юрг (Nievergelt, Jurg), 93
- Нидхам, Иозеф (Needham, Joseph), 68
- Нойман, Франтишек (Neuman, František), 64
- Нутен, Баренд Адриан Анске Йоханнес ван (Nooten, Barend Adrian Anske Johannes van), 69
- Нъенхуис, Альберт (Nijenhuis, Albert), 60; 93

- Обобщенные числа Каталана, 53
 Орд-Смит, Ричард Альберт Джеймс
 (Ord-Smith, Richard Albert James),
 91
 Ориентированное дерево, 34
 Ориентированное оствовое дерево, 61
 Ориентированные леса, 34; 35
 Остовная древовидность, 61
 Оствовое дерево, 35; 38
 Относительное дополнение битовой строки,
 33
 Оффорд, Альберт Сирил (Offord, Albert
 Cyril), 126
 Панхольцер, Алоиз (Panholzer, Alois), 120;
 122
 Параллельное сверхребро, 39
 Паттенхам, Георг и/или Ричард
 (Puttenham, Georg and/or
 Richard), 87
 Пеон, 70
 Перестановка, 71
 Перестановка мульти множества, 72
 Пермуздр, 51
 Пингала, Акарья (Pingala, Ācārya), 68; 70
 Пиррихий, 70
 Полином Чебышева, 134
 Положительно полуопределенная матрица,
 132
 Пороки, 77; 78
 Последовательно-параллельный граф, 39
 Последовательное сверхребро, 39
 Престет, Жан (Prestet, Jean), 84; 97; 143
 Принс, Джильт Калеб Эрнст (Prins, Geert
 Caleb Ernst), 92
 Продингер, Гельмут (Prodinger, Helmut),
 119; 120; 122
 Произведение Кронекера, 133
 Производная графа, 64
 Прокелевсаматик, 70
 Проскуровски, Анджей (Proskurowski,
 Andrzej), 19; 54
 Просодия, 68
 Птолемей, Клавдий из Александрии
 Ptolemy, Claudius, of Alexandria
 (Πτολεμαῖος Κλαύδιος ὁ
 Ἀλεξανδρινός), 83
 Пуансо, Луи (Poinsot, Louis), 95
 Пуансо цикл, 95
 Путеанус, Эрикус (Puteanus, Erycius), 83;
 96; 142; 143
- Разбиение
 Множества, 85
 Мульти множества, 88
 Непересекающееся, 107
 Целых чисел, 88
 Рамеш, Харихаран (Ramesh, Hariharan),
 130
 Ранг, 68
 Рани, Джордж Нил (Raney, George Neal),
 56
 Раски, Франк (Ruskey, Frank), 8; 19; 20; 25;
 27; 54; 94; 113; 127
 Расширенное бинарное дерево, 47
 Расширенное тернарное дерево, 47
 Рашед, Роши (Rashed, Roshdi (=Rashid,
 Rushdi)), 74; 141
 Результант, 134
 Рейнгольд, Эдвард Мартин (Reingold,
 Edward Martin), 93
 Релантс ван Баронайген, Доминик
 (Roelants van Baronaigien,
 Dominique), 19; 20; 113
 Реми, Жан-Люк (Rémy, Jean-Luc), 28; 122
 Решетка
 Кревераса, 48; 50; 52; 109
 Максимальная цепочка, 50
 Стенли, 50; 53
 Тамари, 49; 51; 52; 107; 109; 111
 Риордан, Джон (Riordan, John), 116
 Ричардс, Дана Скотт (Richards, Dana
 Scott), 48
 Робинсон, Жильбер де Борегард (Robinson,
 Gilbert de Beauregard), 58
 Родригес, Бенджамин Олинде (Rodrigues,
 Benjamin Olinde), 28
 Родственный лес, 47
 Россин, Доминик Жиль (Rossin, Dominique
 Gilles), 132
 Роте, Генрих Август (Rothe, Heinrich
 August), 91
 Сак, Ёрг-Рюдигер Вольфганг (Sack,
 Jorg-Rudiger Wolfgang), 121
 Сака, Масанобу (Saka, Masanobu), 87
 Сарнгадева, сын Содхаладевы (Śāṅgadeva,
 son of Sodhaladeva), 73; 82; 142
 Сакс, Хорст (Sachs, Horst), 133; 135
 Сбалансированные строки, 55

- Свойство**
 непересекаемости, 49
 Ханселя, 33
- Седжвик, Роберт (Sedgewick, Robert), 93
- Секанина, Милан (Sekanina, Milan), 44
- Секи, Такаказу (Seki, Takakazu), 73; 87; 95
- Семба, Ичиро (Semba, Ichiro), 14
- Симметричный порядок обхода дерева, 14
- Симс, Чарльз Коффин (Sims, Charles Coffin), 140
- Симса таблица, 140
- Сингх, Пармананд (Singh, Parmanand), 69; 82
- Скалигер, Юлиус (Scaliger, Giulio (=Scaliger, Julius Caesar)), 83
- Скарбек, Владислав Казимир (Skarbek, Wladyslaw Kazimierz), 17
- Скоинс, Губерт Ян (Scoins, Hubert Ian), 34; 94
- Скрученное биномиальное дерево, 64
- След леса, 51
- Слип, Майкл Ронан (Sleep, Michael Ronan), 25
- Слова Дика, 93
- Случайные деревья Каталана, 28
- Смит, Малькольм Джеймс (Smith, Malcolm James), 36; 41
- Солекский порядок, 17
- Сопряженный лес, 46
- Сочетания, 74
- Спернер, Эмануил (Sperner, Emanuel), 30; 57; 125
- Спернера теорема, 30; 57; 125
- Спондей, 70
- Спрунголи, Ренцо (Sprugnoli, Renzo), 55
- Стенли, Ричард Питер (Stanley, Richard Peter), 12; 50; 53; 81; 123
- Стенли решетка, 50; 53
- Степень узла, 47
- Стинсон, Дуглас Роберт (Stinson, Douglas Robert), 94
- Стirling, Джеймс (Stirling, James), 87; 88
- Стirlingа числа, 88
- Стихи-хамелеоны, 83
- Страхлер, Артур Ньювелл (Strahler, Arthur Newell), 65; 138
- Строфа
 Ведическая, 68
- Сусрута (Suśruta), 74
- Сэвеж, Карла Диана (Savage, Carla Diane), 94
- Таблица Симса, 140
- Тайлер, Дуглас Блейн (Tyler, Douglas Blaine), 53
- Тако, Андре (Tacquet, André), 80
- Тамари, Дов (Tamari, Dov), 49; 51; 52; 107; 109
- Тамари решетка, 49; 51; 52; 107; 109; 111
- Танг, Чанджи (Tang, Changjie), 19
- Тарталья, Николо Фонтана (Tartaglia, Niccolò Fontana), 81
- Тезис, 70
- Тенберген, Корнелия ван Эбенхорст (Tengbergen, Cornelia van Ebbenhorst), 30
- Теорема Спернера, 30; 57; 125
- Тимонье, Лойс (Thimonier, Loys), 141
- Тождество Ньютона, 133
- Томпkins, Чарльз Браун (Tompkins, Charles Brown), 93
- Транспонированный лес, 46
- Треугольник Каталана, 23; 25; 32
- Трибрахий, 70
- Трохей, 70
- Уинклер, Питер (Winkler, Peter), 122
- Уитворт, Вильям Аллен (Whitworth, William Allen), 85; 97; 143
- Уоллис, Джон (Wallis, John), 73; 82; 85; 88; 97; 143; 145
- Уоррен, Йон (Warren, Jon), 27
- Упорядоченный лес, 34
- Ушиима, Кацуо (Ushijima, Kazuo), 19
- Уэллс, Марк Бримхолл (Wells, Mark Brimhall), 93; 94
- Феллер, Вилибальд (Feller, Willibald), 121
- Фибоначчи, Леонардо (Fibonacci, Leonardo, of Pisa Leonardo filio Bonacci Pisano), 69
- Фибоначчи числа, 69
- Фидлер, Мирослав (Fiedler, Miroslav), 135
- Флажоле, Филипп патрик Мишель (Flajolet, Philippe Patrick Michel), 141
- Фляйшнер, Герберт (Fleischner, Herbert), 137
- Фосснер, Вильгельм (Feussner, Wilhelm), 36
- Фрис, Ральф Стенли (Freese, Ralph Stanley), 110
- Функция Крускала, 54

- Хакими, Сейфолла Луи (Hakimi, Seifollah Louis), 94
 Хаммонд, Элеонор Прескотт (Hammond, Eleanor Prescott), 76
 Хансель, Джордж (Hansel, Georges), 33; 59
 Харар, Натан бен Саадьях (Har'ar, Natan ben Sa'adyah), 142
 Харари, Франк (Harary, Frank), 92; 128; 137
 Харти, Вильям Эрнест (Hartley, William Ernest), 85; 97; 143
 Хаяши, Цуруичи (Hayashi, Tsuruichi), 144
 Хедетниеми, Сара Ли Митчелл (Hedetniemi, Sarah Lee Mitchell), 34
 Хикерсон, Дин Роберт (Hickerson, Dean Robert), 53
 Ходжес, Джозеф Лоусон мл. (Hodges, Joseph Lawson Jr.), 121
 Хольцман Пуассон, Карлос Альфонсо (Holzmann Poisson, Carlos Alfonso), 128
 Хонда, Тошиаки (Honda, Toshiaki), 87; 97; 144
 Хорей, 70
 Хориямб, 70
 Хортон, Роберт Элмер (Horton, Robert Elmer), 65; 138
 Хортонса-Страхлера числа, 65; 138
 Цветкович, Драгош Младен (Cvetković, Dragoš Mladen (Цветковин, Драгом Младен)), 42; 135
 Цезура, 84
 Центроид, 59
 Цепочка подмножеств, 30
 Циглер, Гюнтер Маттиас (Ziegler, Günter Matthias), 108
 Цикл
 де Брейна, 70
 Колеблющиеся диаграммы, 144
 Пуансо, 95
 Шиллингера, 95
 Чанг, Кай Лай (Chung, Kai Lai), 121
 Чебышев, Пафнутий Львович, 134
 Чебышева полином, 134
 Числа
 Баллотировочные, 23; 53
 Белла, 88; 144
 Каталана, 22; 23; 32; 53; 92; 115
 обобщенные, 53
 Стирлинга, 88
 Фибоначчи, 69
 Хортонса-Страхлера, 65; 138
 Шрёдера, 57; 123; 137
 Эйлера, 58
 Шаблон рождественской елки, 30; 57; 58
 Шао Юнг (Shao Yung), 68
 Швенк, Аллен Джон (Schwenk, Allen John), 137
 Швец, Франк Йозеф (Swetz, Frank Joseph), 68
 Шенстед, Крейг Юджин (Schensted, Craige Eugene), 58
 Шиллингер, Иосиф Моисеевич (Schillinger, Joseph Moiseyevich), 80; 95
 Шиллингера цикл, 95
 Шотт, Рене Пьер (Schott, René Pierre), 122
 Шрёдер, Эрнст (Schröder, Ernst), 57; 137
 Шрёдера дерево, 57
 Шрёдера числа, 57; 123; 137
 Шутен, Франц ван (Schooten, Frans van), 80; 96
 Эдельман, Пол Генри (Edelman, Paul Henry), 110
 Эйлер, Леонард (Euler, Leonhard (Эйлеръ, Леонардъ)), 58; 90
 Эйлера числа, 58
 Эйроллс, Жорж (Eyrolles, Georges), 138
 Эпирит, 70
 Эр, Мэн Чяу (Er, Meng Chiau), 103
 Эрдеш, Пал (Erdős, Pal), 92
 Эррера, Альфред (Errera, Alfred), 100
 Эттингхаузен, Андреас фон (Ettingshausen, Andreas von), 91
 Юнг, Альфред (Yuong, Alfred), 111
 Юнга диаграмма, 53; 111; 114
 Ямб, 70
 Янг Цинг (Yang Hsiung), 68; 69
 Яно, Тамаки (Yano, Tamaki), 86; 87

Научно-популярное издание

Дональд Э. Кнут

**Искусство программирования
Том 4, выпуск 4
Генерация всех деревьев.
История комбинаторной генерации**

Литературный редактор *Т.П. Кайгородова*
Берстка *А.Н. Полинчик*
Художественный редактор *С.А. Чернокозинский*
Корректор *Л.А. Гордиенко*

Издательский дом “Вильямс”
127055, г. Москва, ул. Лесная, д. 43, стр. 1

Подписано в печать 21.05.2007. Формат 70×100/16.
Гарнитура Times. Печать офсетная.
Усл. печ. л. 12,9. Уч.-изд. л. 11,8.
Тираж 3000 экз. Заказ № 4015.

Отпечатано по технологии СтР
в ОАО “Печатный двор” им. А. М. Горького
197110, Санкт-Петербург, Чкаловский пр., 15.

ASCII СИМВОЛЫ

	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	#a	#b	#c	#d	#e	#f	
*2x		!	"	#	\$	%	&	'	()	*	+	,	-	.	/	*2x
*3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?	*3x
*4x	Q	A	B	C	D	E	F	G	H	I	J	K	L	M	N	0	*4x
*5x	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_	*5x
*6x	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	*6x
*7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	¤	*7x
	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	#a	#b	#c	#d	#e	#f	

MMIX КОДЫ ОПЕРАЦИЙ (ОПКОДЫ)

	*0	*1	*2	*3	*4	*5	*6	*7	
*0x	TRAP $5v$	FCMP v	FUN v	FEQL v	FADD $4v$	FIX $4v$	FSUB $4v$	FIXU $4v$	*0x
	FLOT[I] $4v$		FLOTU[I] $4v$		SFLOT[I] $4v$		SFLOTU[I] $4v$		
*1x	FMUL $4v$	FCMPE $4v$	FUNE v	FEQLE $4v$	FDIV $40v$	FSQRT $40v$	FREM $4v$	FINT $4v$	*1x
	MUL[I] $10v$		MULU[I] $10v$		DIV[I] $60v$		DIVU[I] $60v$		
*2x	ADD[I] v		ADDU[I] v		SUB[I] v		SUBU[I] v		*2x
	2ADDU[I] v		4ADDU[I] v		8ADDU[I] v		16ADDU[I] v		
*3x	CMP[I] v		CMPU[I] v		NEG[I] v		NEGU[I] v		*3x
	SL[I] v		SLU[I] v		SR[I] v		SRU[I] v		
*4x	BN[B] $v+\pi$		BZ[B] $v+\pi$		BP[B] $v+\pi$		BOD[B] $v+\pi$		*4x
	BNN[B] $v+\pi$		BNZ[B] $v+\pi$		BNP[B] $v+\pi$		BEV[B] $v+\pi$		
*5x	PBN[B] $3v-\pi$		PBZ[B] $3v-\pi$		PBP[B] $3v-\pi$		PBOD[B] $3v-\pi$		*5x
	PBNN[B] $3v-\pi$		PBNZ[B] $3v-\pi$		PBNP[B] $3v-\pi$		PBEV[B] $3v-\pi$		
*6x	CSN[I] v		CSZ[I] v		CSP[I] v		CSOD[I] v		*6x
	CSNN[I] v		CSNZ[I] v		CSNP[I] v		CSEV[I] v		
*7x	ZSN[I] v		ZSZ[I] v		ZSP[I] v		ZSOD[I] v		*7x
	ZSNN[I] v		ZSNZ[I] v		ZSNP[I] v		ZSEV[I] v		
*8x	LDB[I] $\mu+v$		LDBU[I] $\mu+v$		LDW[I] $\mu+v$		LDWU[I] $\mu+v$		*8x
	LDT[I] $\mu+v$		LDTU[I] $\mu+v$		LDO[I] $\mu+v$		LDOU[I] $\mu+v$		
*9x	LDSF[I] $\mu+v$		LDHT[I] $\mu+v$		CSWAP[I] $2\mu+2v$		LDUNC[I] $\mu+v$		*9x
	LDVTS[I] v		PRELD[I] v		PREGO[I] v		GO[I] $3v$		
*Ax	STB[I] $\mu+v$		STBU[I] $\mu+v$		STW[I] $\mu+v$		STWU[I] $\mu+v$		*Ax
	STT[I] $\mu+v$		STTU[I] $\mu+v$		STO[I] $\mu+v$		STOU[I] $\mu+v$		
*Bx	STSF[I] $\mu+v$		STHT[I] $\mu+v$		STCO[I] $\mu+v$		STUNC[I] $\mu+v$		*Bx
	SYNCD[I] v		PREST[I] v		SYNCID[I] v		PUSHGO[I] $3v$		
*Cx	OR[I] v		ORN[I] v		NOR[I] v		XOR[I] v		*Cx
	AND[I] v		ANDN[I] v		NAND[I] v		NXOR[I] v		
*Dx	Bdif[I] v		WDIF[I] v		TDIF[I] v		ODIF[I] v		*Dx
	MUX[I] v		SADD[I] v		MOR[I] v		MXOR[I] v		
*Ex	SETH v	SETMH v	SETML v	SETL v	INCH v	INCMH v	INCML v	INCL v	*Ex
	ORH v	ORMH v	ORML v	ORL v	ANDNH v	ANDNMH v	ANDNML v	ANDNL v	
*Fx	JMP[B] v		PUSHJ[B] v		GETA[B] v		PUT[I] v		*Fx
	POP $3v$	RESUME $5v$	[UN]SAVE $20\mu+v$		SYNC v	SWYM v	GET v	TRIP $5v$	
	*8	*9	*A	*B	*C	*D	*E	*F	

$\pi = 2v$, если условный переход выполняется; $\pi = 0$, если условный переход не выполняется.

Компьютерные науки/Программирование

Искусство программирования

ДОНАЛЬД Э. КНУТ

Эта многотомная работа по анализу алгоритмов давно считается исчерпывающим описанием классической информатики. Три завершенных тома, вышедших в свет к этому времени, представляют собой неоценимый источник информации для теории и практики программирования. Многочисленные читатели этого труда говорят о его огромном влиянии на них и их профессионализм. Ученых потрясает красота и элегантность анализа Кнута, программисты-практики используют эти книги как справочник для решения возникающих перед ними проблем. И все восхищены обширностью, ясностью, точностью и юмором *Искусства программирования*.

В настоящее время, работая над четвертым и последующими томами и над обновлением уже вышедших, Кнут создал серию небольших брошюр, называемых выпусками, которые планирует регулярно издавать. Каждый выпуск содержит один или несколько разделов с новым или обновленным материалом. В конечном счете материалы выпусков войдут в окончательные версии каждого тома, и начатая в 1962 году гигантская работа будет завершена.

Том 4, выпуск 4

Данный выпуск посвящен генерации всех деревьев — фундаментальной теме, которая на удивление тесно связана с первыми тремя томами *Искусства программирования*. Всестороннее рассмотрение этого широко известного вопроса, подкрепленное 124 новыми упражнениями, продолжает построение прочного фундамента для программирования, начатое в первых томах. В этом же выпуске содержится и обзор истории комбинаторной генерации. Путешествуя по разным континентам и разным векам, Кнут рассказывает читателям завораживающую историю, интересную любому программисту, особенно если учесть, что многие части этой истории никогда прежде никем не были рассказаны. Самое интригующее в этой истории — две задачи, которые пока что никто так и не смог решить.

Дональд Э. Кнут известен всему миру своей пионерской работой по алгоритмам и методам программирования, разработками издательских систем T_EX и METAFONT, а также научными работами. Заслуженный профессор Искусства программирования Станфордского университета в настоящее время посвящает все свое время работе над завершением данных выпусков и всех семи томов своей великой работы.



Издательский дом “ВИЛЬЯМС”
www.williamspublishing.com



ADDISON-WESLEY
Pearson Education
www.awprofessional.com

ISBN 978-5-8459-1158-2

0 6 2 4 4

9 785845 911582