3.4. ДРУГИЕ ВИДЫ СЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

В ПРЕДЫДУЩИХ РАЗДЕЛАХ мы обсуждали, как генерировать на компьютере последовательность чисел $U_0, U_1, U_2, \ldots$, которые ведут себя так, как если бы каждое число выбиралось независимо и случайно между 0 и 1 с равномерным распределением. Однако при использовании случайных чисел часто требуются другие виды распределений. Например, чтобы сделать случайный выбор из k альтернатив, нужны случайные *целые* числа, лежащие между 1 и k. Если необходимо моделировать случайное время ожидания между появлениями независимых событий, желательно получить случайные числа с *показательным распределением*. Иногда в случайных *числах* нет необходимости, но нужны случайные *перестановки* (случайное размещение n объектов) или случайное *сочетание* (случайный выбор k объектов из совокупности, содержащей n объектов).

В принципе, любая из этих случайных величин может быть получена из равномерно распределенных случайных величин $U_0, U_1, U_2, \ldots$. Значительное число "случайных трюков" было придумано для эффективного преобразования равномерно распределенных случайных чисел. Изучив эти методы, получим возможность правильно использовать случайные числа при любом применении метода Монте-Карло.

Вероятно, что кто-нибудь когда-нибудь придумает генератор случайных чисел, который будет вырабатывать одну из этих случайных величин *непосредственно*, а не косвенно через равномерное распределение. Но прямые методы, как доказано, не практичны, за исключением генератора "случайный двоичный разряд", описанного в разделе 3.2.2. (См. также упр. 3.4.1–31; в нем равномерное распределение используется, главным образом, для инициализации, после которой метод является почти полностью прямым.)

В следующем разделе предполагается наличие случайной последовательности равномерно распределенных между 0 и 1 независимых действительных чисел. Равномерно распределенная случайная величина U генерируется всякий раз, когда в ней возникает необходимость. Эти числа обычно представлены в компьютере словом с десятичной точкой слева.

3.4.1. Численные распределения

В этом разделе объединены наиболее известные методы получения случайных чисел для различных важных распределений. Многие из методов первоначально были предложены Джоном фон Нейманом (John von Neumann) в начале 50-х. Постепенно они усовершенствовались другими математиками, особенно Джорджем Марсалья (George Marsaglia), И. Г. Аренсом (J. H. Ahrens) и У. Дитером (U. Dieter).

A. Случайный выбор из ограниченного множества. Самые простые и наиболее общие типы распределений, используемых в приложениях, — это распределения случайных *целых* чисел. Целые числа между 0 и 7 могут быть извлечены из трех двоичных разрядов U на бинарном компьютере; поэтому эти три двоичных разряда можно извлечь из *старшей значащей* (слева) части компьютерного слова, поскольку самые младшие двоичные разряды, производимые многими генераторами случайных чисел, недостаточно случайны (см. раздел 3.2.1.1).

В общем случае случайные целые числа X, которые лежат между 0 и $k-1$, можно получить, *умножив* U на k и положив $X = \lfloor kU \rfloor$. На MIX можно записать

```
LDA  U
MUL  K
```
(1)

После выполнения этих двух операций требуемое целое число появится в регистре A. Чтобы получить случайное целое число, лежащее между 1 и k, следует добавить единицу к этому результату. (Операция "INCA 1" последует за (1).)

С помощью данного метода каждое целое число можно получить с приблизительно равной вероятностью. Существует незначительная ошибка, так как длина слова компьютера конечна (см. упр. 2), но эта ошибка совершенно незначительна, если k мало́, например $k/m < 1/10000$.

В более общем случае можно получить, если необходимо, различные веса для различных целых чисел. Предположим, что значение $X = x_1$ должно быть получено с вероятностью p_1, $X = x_2$ — с вероятностью p_2, ... и $X = x_k$ — с вероятностью p_k. Генерируем равномерное число U и положим

$$X = \begin{cases} x_1, & \text{если } 0 \le U < p_1; \\ x_2, & \text{если } p_1 \le U < p_1 + p_2; \\ \vdots & \\ x_k, & \text{если } p_1 + p_2 + \cdots + p_{k-1} \le U < 1. \end{cases} \tag{2}$$

(Заметим, что $p_1 + p_2 + \cdots + p_k = 1$.)

Существует "наилучший возможный" способ сравнения U с различными значениями $p_1 + p_2 + \cdots + p_s$, как подразумевается в (2) (см. раздел 2.3.4.5). В частных случаях можно обойтись более эффективными методами; например, для того, чтобы получить одно из одиннадцати чисел 2, 3, ..., 12 с соответствующими "игре в кости" вероятностями $\frac{1}{36}, \frac{2}{36}, \ldots, \frac{6}{36}, \ldots, \frac{2}{36}, \frac{1}{36}$, можно вычислить два независимых случайных целых числа между 1 и 6 и сложить их.

Тем не менее существует действительно более быстрый способ выбора $x_1, \ldots, x_k$ с произвольно заданной вероятностью, основанный на остроумном подходе, который был введен в употребление А. Дж. Уолкером (см. A. J. Walker, *Electronics Letters* **10**, 8 (1974), 127–128; *ACM Trans. Math. Software* **3** (1977), 253–256). Предположим, что мы образуем kU и рассматриваем целую часть $K = \lfloor kU \rfloor$ и дробную часть $V = (kU) \bmod 1$ раздельно, например после выполнения операций (1) получим K в регистре A и V — в регистр. Затем всегда можно получить желаемое распределение, выполнив операции

$$\text{если } \quad V < P_K, \quad \text{то } \quad X \leftarrow x_{K+1}, \quad \text{иначе } \quad X \leftarrow Y_K \tag{3}$$

для некоторых подходящих таблиц $(P_0, \ldots, P_{k-1})$ и $(Y_0, \ldots, Y_{k-1})$. В упр. 7 показано, как вообще можно вычислить такие таблицы. Метод Уолкера иногда называют методом псевдонимов .

На бинарном компьютере обычно полезно предполагать, что k является степенью 2, потому что умножение может быть заменено сдвигом. Это можно делать без потери общности, введя дополнительные x_j, которые появляются с вероятностью 0. Например, снова рассмотрим игру в кости. Предположим, что равенство $X = j$

должно произойти со следующими 16 вероятностями.

$$\begin{array}{rcccccccccccccccc} j = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ p_j = & 0 & 0 & \frac{1}{36} & \frac{2}{36} & \frac{3}{36} & \frac{4}{36} & \frac{5}{36} & \frac{6}{36} & \frac{5}{36} & \frac{4}{36} & \frac{3}{36} & \frac{2}{36} & \frac{1}{36} & 0 & 0 & 0 \end{array}$$

Это можно осуществить, используя (3), если $k = 16$ и $x_{j+1} = j$ при $0 \le j < 16$ и если таблицы для P и Y имеют следующий вид.

$$\begin{array}{rcccccccccccccccc} j = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ P_j = & 0 & 0 & \frac{4}{9} & \frac{8}{9} & 1 & \frac{7}{9} & 1 & 1 & 1 & \frac{7}{9} & \frac{7}{9} & \frac{8}{9} & \frac{4}{9} & 0 & 0 & 0 \\ Y_j = & 5 & 9 & 7 & 4 & * & 6 & * & * & * & 8 & 4 & 7 & 10 & 6 & 7 & 8 \end{array}$$

(Когда $P_j = 1$, Y_j не используются.) Например, значение 7 встречается с вероятностью $\frac{1}{16} \cdot \big((1 - P_2) + P_7 + (1 - P_{11}) + (1 - P_{14})\big) = \frac{6}{36}$, как и требуется. Это необычный способ бросания игральных костей, но результаты получаются такие же, как и в реальной ситуации.

Вероятности p_j, безусловно, могут быть представлены неотрицательными весами $w_1, w_2, \ldots, w_k$; если обозначить сумму весов через W, то $p_j = w_j/W$. В разных применениях отдельные веса весьма изменчивы. Матиас, Виттер и Ни (см. работу Matias, Vitter, and Ni, *SODA* **4** (1993), 361–370) показали, как изменять веса и генерировать X с постоянным средним временем.

B. Общие методы для непрерывных распределений. В общем случае распределение действительных чисел может быть выражено в терминах "функции распределения" $F(x)$, которая точно определяет вероятность того, что случайная величина X не превысит значение x:

$$F(x) = \Pr(X \le x). \tag{4}$$

Эта функция всегда монотонно возрастает от 0 до 1, т. е.

$$F(x_1) \le F(x_2), \quad \text{если } x_1 \le x_2; \qquad F(-\infty) = 0, \qquad F(+\infty) = 1. \tag{5}$$

Примеры функций распределения приведены в разделе 3.3.1 (см. рис. 3). Если $F(x)$ непрерывна и строго возрастающая (так что $F(x_1) < F(x_2)$, когда $x_1 < x_2$), то она принимает все значения между 0 и 1 и существует *обратная функция* $F^{[-1]}(y)$, такая, что для $0 < y < 1$

$$y = F(x) \qquad \text{тогда и только тогда, когда} \qquad x = F^{[-1]}(y). \tag{6}$$

В большинстве случаев, когда $F(x)$ непрерывна и строго возрастающая, можно вычислить случайную величину X с распределением $F(x)$, полагая

$$X = F^{[-1]}(U), \tag{7}$$

где U — равномерно распределенная случайная величина. Действительно, вероятность того, что $X \le x$, равна вероятности, что $F^{[-1]}(U) \le x$, а именно — вероятности того, что $U \le F(x)$, т. е. $F(x)$.

Теперь проблема сводится к решению задачи численного анализа — к нахождению хороших методов вычисления $F^{[-1]}(U)$ с требуемой точностью. Численный

анализ в этой книге о получисленных алгоритмах не рассматривается, однако существует ряд важных методов, способных улучшить общий подход (7), и здесь они будут рассмотрены.

Заметим, что если X_1 — случайная величина, имеющая функцию распределения $F_1(x)$, и если X_2 — независимая от X_1 случайная величина с функцией распределения $F_2(x)$, то

$$\begin{aligned} &\max(X_1, X_2) \quad \text{имеет распределение} \quad F_1(x)F_2(x), \\ &\min(X_1, X_2) \quad \text{имеет распределение} \quad F_1(x) + F_2(x) - F_1(x)F_2(x). \end{aligned} \tag{8}$$

(См. упр. 4.) Например, равномерно распределенная случайная величина U имеет распределение $F(x) = x$ для $0 \le x \le 1$; если U_1, U_2, ..., U_t — независимые равномерно распределенные случайные величины, то $\max(U_1, U_2, \ldots, U_t)$ имеет функцию распределения $F(x) = x^t$ при $0 < x \le 1$. Эта формула является основой критерия "максимум-t", описанного в разделе 3.3.2. Обратная функция равна $F^{[-1]}(y) = \sqrt[t]{y}$. В частном случае при $t = 2$ получаем, следовательно, что формулы

$$X = \sqrt{U} \qquad \text{и} \qquad X = \max(U_1, U_2) \tag{9}$$

дадут одинаковое распределение случайной величины X, хотя, на первый взгляд, это не очевидно. Нет необходимости извлекать квадратный корень из равномерно распределенной случайной величины.

Количество подобных хитростей бесконечно: *любой* алгоритм, использующий случайные числа на входе, дает на выходе случайные величины с *некоторым* распределением. Задача состоит в нахождении общих методов составления алгоритма, обеспечивающего заданную функцию распределения на выходе. Вместо того чтобы рассматривать подобные методы в исключительно абстрактных терминах, изучим, как они могут применяться в важных случаях.

C. Нормальное распределение. Возможно, наиболее значительным неравномерным, непрерывным распределением является *нормальное распределение с нулевым средним значением и среднеквадратичным отклонением, равным единице:*

$$F(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt. \tag{10}$$

Значительность данного распределения показана в разделе 1.2.10. В нашем случае обратную функцию $F^{[-1]}$ не так легко вычислить; но, как мы увидим, существует несколько технических приемов моделирования этого распределения.

1) *Метод полярных координат*, предложенный Дж. Э. П. Боксом, М. Э. Мюллером и Дж. Марсальей [см. G. E. P. Box, M. E. Muller, and G. Marsaglia, *Annals Math. Stat.* **29** (1958), 610–611, и Boeing Scientific Res. Lab. report D1-82-0203 (1962)].

Алгоритм P (*Метод полярных координат для нормальных случайных величин*). Этот алгоритм вычисляет две независимые нормально распределенные случайные величины: X_1 и X_2.

P1. [Получение равномерно распределенных случайных величин.] Генерируем две независимые случайные величины U_1 и U_2, равномерно распределенные между 0

и 1. Присвоить $V_1 \leftarrow 2U_1 - 1$, $V_2 \leftarrow 2U_2 - 1$. (Здесь V_1 и V_2 равномерно распределены между $-$ё1 и $+1$. На большинстве компьютеров предпочтительнее представление V_1 и V_2 в виде чисел с плавающей точкой.)

P2. [Вычисление S.] Присвоить $S \leftarrow V_1^2 + V_2^2$.

P3. [Проверить $S \geq 1$?] Если $S \geq 1$, возврат к шагу P1. (Шаги P1–P3 выполняются в среднем 1.27 раз со среднеквадратичным отклонением, равным 0.587; см. упр. 6.)

P4. [Вычисление X_1, X_2.] Присвоить X_1 и X_2 следующие значения:

$$X_1 \leftarrow V_1\sqrt{\frac{-2\ln S}{S}}, \qquad X_2 \leftarrow V_2\sqrt{\frac{-2\ln S}{S}}. \tag{11}$$

Это требуемые нормально распределенные случайные величины. ▮

Для доказательства законности данного метода используем элементарную аналитическую геометрию и вычисления: если на шаге P3 $S < 1$, точка плоскости с декартовыми координатами (V_1, V_2) *является случайной точкой, равномерно распределенной внутри единичного круга*. Перейдя к полярным координатам $V_1 = R\cos\Theta$, $V_2 = R\sin\Theta$, получим

$$S = R^2, \quad X_1 = \sqrt{-2\ln S}\cos\Theta, \quad X_2 = \sqrt{-2\ln S}\sin\Theta.$$

Используя также полярные координаты $X_1 = R'\cos\Theta'$ и $X_2 = R'\sin\Theta'$, получим $\Theta' = \Theta$ и $R' = \sqrt{-2\ln S}$. Ясно, что R' и Θ' независимы, поскольку R и Θ независимы в единичном круге. К тому же Θ' равномерно распределено между 0 и 2π, и вероятность того, что $R' \leq r$, равна вероятности, что $-2\ln S \leq r^2$, т. е. вероятности, что $S \geq e^{-r^2/2}$. Эта вероятность равна $1 - e^{-r^2/2}$, так как $S = R^2$ равномерно распределено между 0 и 1. Вероятность того, что R' лежит между r и $r+dr$, поэтому равна дифференциалу от $1 - e^{-r^2/2}$, т. е. $re^{-r^2/2}\,dr$. Аналогично вероятность того, что Θ' лежит между θ и $\theta + d\theta$, равна $(1/2\pi)\,d\theta$. Совместная вероятность того, что $X_1 \leq x_1$ и $X_2 \leq x_2$, равняется

$$\begin{aligned}\int_{\{(r,\theta)\,|\,r\cos\theta\leq x_1,\ r\sin\theta\leq x_2\}} &\frac{1}{2\pi}e^{-r^2/2}r\,dr\,d\theta \\ &= \frac{1}{2\pi}\int_{\{(x,y)\,|\,x\leq x_1,\ y\leq x_2\}} e^{-(x^2+y^2)/2}\,dx\,dy \\ &= \left(\sqrt{\frac{1}{2\pi}}\int_{-\infty}^{x_1} e^{-x^2/2}\,dx\right)\left(\sqrt{\frac{1}{2\pi}}\int_{-\infty}^{x_2} e^{-y^2/2}\,dy\right).\end{aligned}$$

Это и доказывает, что X_1 и X_2 независимы и нормально распределены.

2) *Метод прямоугольника-клина-хвоста* предложен Дж. Марсальей. Здесь используется функция

$$F(x) = \operatorname{erf}(x/\sqrt{2}\,) = \sqrt{\frac{2}{\pi}}\int_0^x e^{-t^2/2}\,dt, \qquad x \geq 0, \tag{12}$$

которая является функцией распределения *абсолютного значения* нормальной случайной величины. Затем X вычисляется в соответствии с распределением (12).

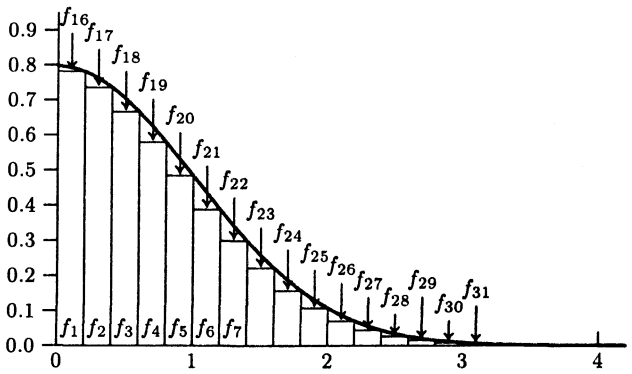

Рис. 9. Площадь под графиком плотности распределения разделена на 31 часть. Площадь каждой части равна среднему числу вычислений случайной величины с такой плотностью.

Припишем случайный знак ее значению, и это сделает ее действительно нормальной случайной величиной.

Метод прямоугольника-клина-хвоста основан на важных общих технических приемах, которые будут рассмотрены ниже по мере построения алгоритма. Первая ключевая идея — рассматривать $F(x)$ как *смесь* нескольких других функций, т. е. записать

$$F(x) = p_1F_1(x) + p_2F_2(x) + \cdots + p_nF_n(x), \tag{13}$$

где $F_1, F_2, \ldots, F_n$ — подходящие распределения и $p_1, p_2, \ldots, p_n$ — неотрицательные вероятности, сумма которых равна 1. Если генерировать случайную переменную X, выбирая распределение F_j с вероятностью p_j, то легко видеть, что X точно будет иметь F-распределение. С некоторыми распределениями $F_j(x)$, пожалуй, трудно иметь дело, даже труднее, чем с F, но мы обычно устраиваем так, что вероятности p_j в этом случае очень малы. Большинство распределений $F_j(x)$ будут довольно хорошо устроены, поскольку они будут простой модификацией равномерного распределения. Изложение завершается чрезвычайно эффективной программой, поскольку *среднее* время счета этой программы очень мало́.

Рассма́триваемый здесь метод легче понять, если работать с *производными* распределений, а не с самими распределениями. Пусть

$$f(x) = F'(x), \qquad f_j(x) = F_j{}'(x)$$

будут *плотностями распределений*. Тогда равенство (13) можно записать как

$$f(x) = p_1f_1(x) + p_2f_2(x) + \cdots + p_nf_n(x). \tag{14}$$

Каждая $f_j(x)$ есть ≥ 0, и общая площадь под графиком $f_j(x)$ равна 1. Поэтому существует подходящий графический метод отображения зависимости (14): площадь под $f(x)$ разделена на n частей и части, соответствующие $f_j(x)$, имеют площадь p_j. На рис. 9 иллюстрируется интересующий нас случай при $f(x) = F'(x) = \sqrt{2/\pi}\,e^{-x^2/2}$; площадь под этой кривой разделена на $n = 31$ часть. Существует 15 прямоугольников, представляющих $p_1f_1(x), \ldots, p_{15}f_{15}(x)$, 15 клинообразных частей, представляющих $p_{16}f_{16}(x), \ldots, p_{30}f_{30}(x)$, и оставшаяся часть $p_{31}f_{31}(x)$ — "хвост", т. е. график $f(x)$ при $x \geq 3$.

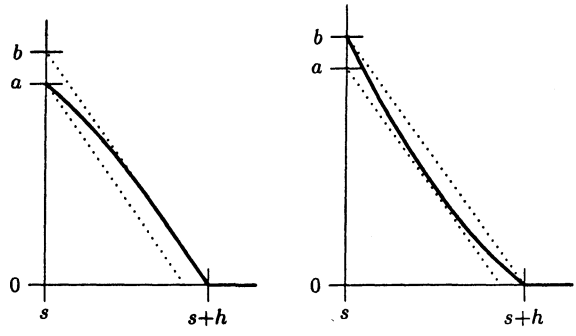

Рис. 10. Плотность распределения, для которой алгоритм L может использоваться при генерировании случайных чисел.

Прямоугольные части $f_1(x)$, ..., $f_{15}(x)$ представляют *равномерное распределение*. Например, $f_3(x)$ представляет случайную равномерно распределенную величину, лежащую между $\frac{2}{5}$ и $\frac{3}{5}$. Высота $p_j f_j(x)$ равна $f(j/5)$; следовательно, площадь j-го прямоугольника равна

$$p_j = \frac{1}{5} f(j/5) = \sqrt{\frac{2}{25\pi}}\, e^{-j^2/50} \qquad \text{для } 1 \le j \le 15. \tag{15}$$

Чтобы генерировать распределение, соответствующее таким прямоугольным частям, просто вычислим

$$X = \tfrac{1}{5}U + S, \tag{16}$$

где U равномерно и S принимает значение $(j-1)/5$ с вероятностью p_j и не зависит от U. Так как $p_1 + \cdots + p_{15} = .9183$, можно просто использовать равномерно распределенные случайные величины, подобные этим приблизительно в 92% случаев.

В оставшихся 8% случаев будем обычно генерировать одно из клинообразных распределений F_{16}, ..., F_{30}. Типичный пример, который показывает, что необходимо делать, представлен на рис. 10. Когда $x < 1$, часть кривой вогнутая, а когда $x > 1$, она выпуклая, но в каждом случае часть кривой достаточно близка к прямой линии и, как показано, может быть заключена между двумя параллельными линиями.

Чтобы перебрать эти клиновидные распределения, будем использовать другой общий технический прием: *метод отбраковки* фон Неймана получения сложной плотности из другой плотности, которая ее "заключает". Описанный выше метод полярных координат является простым примером такого подхода: шаги P1–P3 получают случайную точку внутри единичного круга, генерируя ее в большем круге, отбраковывая ее и начиная снова, если точка была вне круга.

Общий метод отбраковки является даже более сильным, чем этот. Пусть нужно генерировать случайную величину X с плотностью f и пусть g — другая плотность распределения, такая, что

$$f(t) \le cg(t) \tag{17}$$

для всех t, где c — константа. Тогда генерируем случайную величину X с плотностью g, а также независимую равномерно распределенную случайную величину U. Если $U \ge f(X)/cg(X)$, отбрасываем X и начинаем снова с другими X и U. Когда

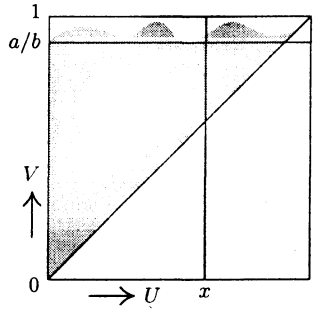

Рис. 11. Область "принятия гипотезы" алгоритма L.

условие $U < f(X)/cg(X)$ в конце концов выполняется, на выходе X будет иметь требуемую плотность f. [*Доказательство.* $X \le x$ произойдет с вероятностью $p(x) = \int_{-\infty}^{x} \big(g(t)\,dt \cdot f(t)/cg(t)\big) + qp(x)$, где величина $q = \int_{-\infty}^{\infty} \big(g(t)\,dt \cdot (1 - f(t)/cg(t))\big) = 1 - 1/c$ равна вероятности отбраковки; следовательно, $p(x) = \int_{-\infty}^{x} f(t)\,dt$.]

Техника отбраковки наиболее эффективна, когда c мало́, так как должно быть в среднем c итераций, прежде чем значение будет принято (см. упр. 6). В одних случаях $f(x)/cg(x)$ всегда равно 0 или 1 и нет необходимости генерировать U. В других случаях, если $f(x)/cg(x)$ трудно вычислить, следует постараться "втиснуть" его между двумя более простыми граничными функциями

$$r(x) \le f(x)/cg(x) \le s(x) \tag{18}$$

и точное значение $f(x)/cg(x)$ не нужно вычислять, если не выполняется неравенство $r(x) \le U < s(x)$. Следующий алгоритм разрешает проблему клина, совершенствуя метод отбраковки.

Алгоритм L (*Плотности, близкие к линейным*). Этот алгоритм можно использовать для генерирования случайной величины X с любым распределением, плотность $f(x)$ которого удовлетворяет следующим условиям (см. рис. 10):

$$\begin{aligned} f(x) &= 0 \qquad\qquad \text{для } x < s \text{ и для } x > s + h; \\ a - b(x-s)/h \le f(x) &\le b - b(x-s)/h \qquad \text{для } s \le x \le s + h. \end{aligned} \tag{19}$$

L1. [Получить $U \le V$.] Генерировать две независимые случайные величины U и V, равномерно распределенные между 0 и 1. Если $U > V$, заменить $U \leftrightarrow V$.

L2. [Простой случай?] Если $V \le a/b$, перейти к шагу L4.

L3. [Попытаемся снова?] Если $V > U + (1/b)f(s + hU)$, возвратиться к шагу L1. (Если a/b близко к 1, данный шаг алгоритма нужен не очень часто).

L4. [Вычисление X.] Присвоить $X \leftarrow s + hU$. ▮

Когда достигнут шаг L4, точка (U, V) является случайной точкой на площади, заштрихованной на рис. 11, а именно — $0 \le U \le V \le U + (1/b)f(s + hU)$. Условия (19) гарантируют, что

$$\frac{a}{b} \le U + \frac{1}{b} f(s + hU) \le 1.$$

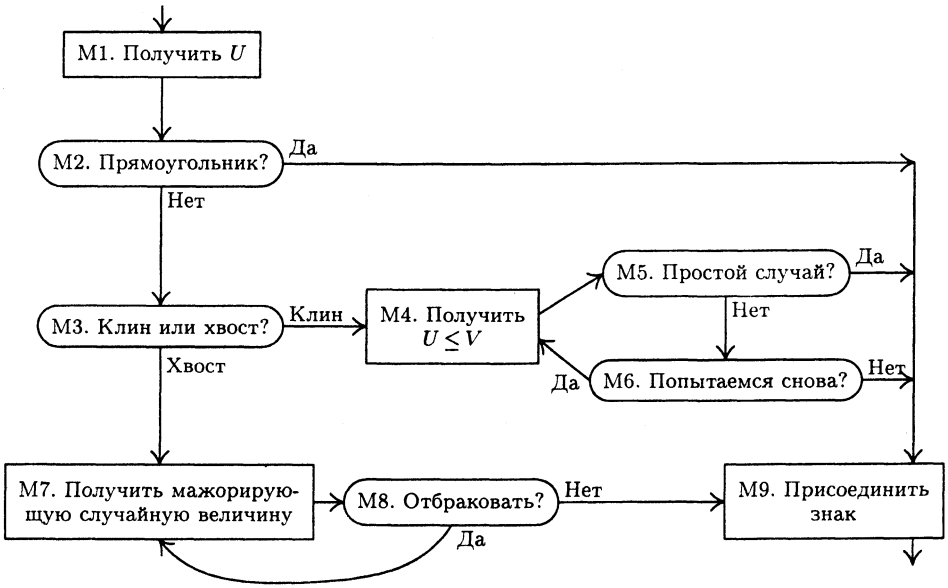

Рис. 12. Алгоритм прямоугольника-клина-хвоста для генерирования нормальных случайных величин.

Сейчас вероятность того, что $X \le s + hx$ для $0 \le x \le 1$, — это площадь, лежащая слева от вертикальной линии $U = x$ (см. рис. 11) и деленная на общую площадь, т. е.

$$\int_0^x \frac{1}{b} f(s+hu)\,du \Big/ \int_0^1 \frac{1}{b} f(s+hu)\,du = \int_s^{s+hx} f(v)\,dv.$$

Следовательно, X имеет нужное распределение.

С подходящими константами a_j, b_j и s_j алгоритм L можно применять к клинообразным плотностям f_{j+15} (см. рис. 9) для $1 \le j \le 15$. Последнее распределение F_{31} нуждается в обработке приблизительно один раз из 370; оно используется тогда, когда получаем результат $X \ge 3$. В упр. 11 показано, что стандартная схема отбраковки может использоваться для этого "хвоста". Рассмотрим процедуру в целом.

Алгоритм M (*Метод прямоугольника-клина-хвоста для нормально распределенных случайных величин*). Для этого алгоритма (рис. 12) используем вспомогательные таблицы $(P_0, \ldots, P_{31})$, $(Q_1, \ldots, Q_{15})$, $(Y_0, \ldots, Y_{31})$, $(Z_0, \ldots, Z_{31})$, $(S_1, \ldots, S_{16})$, $(D_{16}, \ldots, D_{30})$, $(E_{16}, \ldots, E_{30})$, построенные так, как в упр. 10; примеры выбираются из табл. 1. Предполагается, что используется бинарный компьютер (подобные процедуры могут быть построены для десятичных машин).

M1. [Получить U.] Генерируем равномерно распределенное случайное число $U = (.b_0 b_1 b_2 \ldots b_t)_2$. (Здесь b_j — двоичные разряды в бинарном представлении U. Для приемлемой точности t должно быть по крайней мере равно 24.) Присвоить $\psi \leftarrow b_0$. (Позже ψ будет использовано для определения знака результата.)

M2. [Прямоугольник?] Присвоить $j \leftarrow (b_1 b_2 b_3 b_4 b_5)_2$ (двоичное число определяется старшими двоичными разрядами U) и присвоить $f \leftarrow (.b_6 b_7 \ldots b_t)_2$ (дробная часть определяется оставшимися двоичными разрядами). Если $f \geq P_j$, присвоить $X \leftarrow Y_j + fZ_j$ и перейти к шагу M9. Иначе, если $j \leq 15$ (т. е. $b_1 = 0$), присвоить $X \leftarrow S_j + fQ_j$ и перейти к шагу M9. (Это использование метода псевдонимов Уолкера (Walker) (3).)

M3. [Клин или хвост?] (Сейчас $15 \leq j \leq 31$ и каждое определенное значение j появляется с вероятностью p_j.) Если $j = 31$, перейти к шагу M7.

M4. [Получить $U \leq V$.] Генерировать две новые независимые равномерно распределенные случайные величины U и V; если $U > V$, заменить $U \leftrightarrow V$. (Сейчас выполняется алгоритм L.) Присвоить $X \leftarrow S_{j-15} + \frac{1}{5}U$.

M5. [Простой случай?] Если $V \leq D_j$, перейти к шагу M9.

M6. [Попытаемся снова?] Если $V > U + E_j(e^{(S_{j-14}^2 - X^2)/2} - 1)$, вернуться к шагу M4; иначе — перейти к шагу M9. (Вероятность, что этот шаг нужно будет выполнять, мала.)

M7. [Получить мажорирующую случайную величину.] Генерировать две новые независимые равномерно распределенные случайные величины U и V и присвоить $X \leftarrow \sqrt{9 - 2\ln V}$.

M8. [Отбраковать?] Если $UX \geq 3$, возвратиться к шагу M7. (Это будет случаться приблизительно в одном из двенадцати случаев после достижения шага M8.)

M9. [Присоединить знак.] Если $\psi = 1$, присвоить $X \leftarrow -X$. ▮

Этот алгоритм является очень миленьким примером тесного переплетения математической теории с изобретательностью программирования — прекрасная иллюстрация искусства программирования! На выполнение шагов M1, M2 и M9 уходит почти все время; остальные шаги осуществляются намного быстрее. Впервые метод прямоугольника-клина-хвоста опубликовали Дж. Марсалья, Дж. Марсалья, М. Д. Мак-Ларен и Т. А. Брей (см. работы G. Marsaglia, *Annals Math. Stat.* **32** (1961), 894–899; G. Marsaglia, M. D. MacLaren, and T. A. Bray, *CACM* **7** (1964), 4–10). В дальнейшем алгоритм M усовершенствовали Дж. Марсалья, К. Ананханараяпан и Н. Дж. Поль (см. работу G. Marsaglia, K. Ananthanarayanan, and N. J. Paul, *Inf. Proc. Letters* **5** (1976), 27–30).

3) *Метод "чет-нечет"* принадлежит Дж. Э. Форсайту (G. E. Forsythe). Поразительно простая техника генерирования случайных величин с обычной экспоненциальной плотностью вида

$$f(x) = Ce^{-h(x)}\,[a \leq x < b], \tag{20}$$

где

$$0 \leq h(x) \leq 1 \qquad \text{для } a \leq x < b, \tag{21}$$

была открыта Джоном фон Нейманом и Дж. Е. Форсайтом приблизительно в 1950 году. Идея основана на методе отбраковки, описанном ранее. Предполагается, что $g(x)$ — это равномерное распределение на интервале $[a\,..\,b)$. Присвоим $X \leftarrow a + (b - a)U$, где U — равномерно распределенная случайная величина, и примем X с вероятностью $e^{-h(X)}$. Последняя операция может быть осуществлена путем сравнения $e^{-h(X)}$ с V либо $h(X)$ с $-\ln V$, когда V — другая равномерно

Таблица 1

ПРИМЕРЫ ТАБЛИЦ, ИСПОЛЬЗУЕМЫХ С АЛГОРИТМОМ M*

j	P_j	P_{j+16}	Q_j	Y_j	Y_{j+16}	Z_j	Z_{j+16}	S_{j+1}	D_{j+15}	E_{j+15}
0	.000	.067		0.00	0.59	0.20	0.21	0.0		
1	.849	.161	.236	− 0.92	0.96	1.32	0.24	0.2	.505	25.00
2	.970	.236	.206	− 5.86	−0.06	6.66	0.26	0.4	.773	12.50
3	.855	.285	.234	− 0.58	0.12	1.38	0.28	0.6	.876	8.33
4	.994	.308	.201	−33.13	1.31	34.93	0.29	0.8	.939	6.25
5	.995	.304	.201	−39.55	0.31	41.35	0.29	1.0	.986	5.00
6	.933	.280	.214	− 2.57	1.12	2.97	0.28	1.2	.995	4.06
7	.923	.241	.217	− 1.61	0.54	2.61	0.26	1.4	.987	3.37
8	.727	.197	.275	0.67	0.75	0.73	0.25	1.6	.979	2.86
9	1.000	.152	.200	0.00	0.56	0.00	0.24	1.8	.972	2.47
10	.691	.112	.289	0.35	0.17	0.65	0.23	2.0	.966	2.16
11	.454	.079	.440	− 0.17	0.38	0.37	0.22	2.2	.960	1.92
12	.287	.052	.698	0.92	−0.01	0.28	0.21	2.4	.954	1.71
13	.174	.033	1.150	0.36	0.39	0.24	0.21	2.6	.948	1.54
14	.101	.020	1.974	− 0.02	0.20	0.22	0.20	2.8	.942	1.40
15	.057	.086	3.526	0.19	0.78	0.21	0.22	3.0	.936	1.27

*Практически эти данные могут быть приведены с намного большей точностью; в таблице приведено только достаточное количество цифр, чтобы интересующиеся читатели могли более точно проверить собственные алгоритмы вычисления значений.

распределенная случайная величина, но работу можно проделать без применения каких-либо трансцендентных функций следующим интересным способом. Присвоим $V_0 \leftarrow h(X)$ и будем генерировать обычные равномерно распределенные случайные величины V_1, V_2, ..., пока не найдутся $K \geq 1$ с $V_{K-1} < V_K$. Для фиксированных X и k вероятность того, что $h(X) \geq V_1 \geq \cdots \geq V_k$, равна $1/k!$, умноженному на вероятность того, что $\max(V_1, \ldots, V_k) \leq h(X)$, т. е. $h(X)^k/k!$. Следовательно, вероятность того, что $K = k$, равна $h(X)^{k-1}/(k-1)! - h(X)^k/k!$, а вероятность того, что K — нечетное число, равна

$$\sum_{k \text{ нечетное}, k \geq 1} \left(\frac{h(X)^{k-1}}{(k-1)!} - \frac{h(X)^k}{k!} \right) = e^{-h(X)}. \tag{22}$$

Следовательно, мы отбраковываем X и начинаем снова, если K — четное число. Принимаем X как случайную величину с плотностью распределения (20), если K — нечетное число. Обычно мы не генерируем много значений V_j, чтобы определить K, поскольку среднее значение K (при заданном X) равно $\sum_{k \geq 0} \Pr(K > k) = \sum_{k \geq 0} h(X)^k/k! = e^{h(X)} \leq e$.

Несколькими годами позже Форсайт показал, что этот подход приводит к получению эффективного метода вычисления нормальных случайных величин без использования вспомогательных программ для вычисления квадратных корней или логарифмов, как в алгоритмах P и M. Его процедуру с улучшенным выбором интервалов $[a\,..\,b)$, осуществленную И. Г. Аренсом (J. H. Ahrens) и У. Дитером (U. Dieter), можно представить в виде алгоритма.

Алгоритм F (*Метод "чет-нечет" для нормальных случайных величин*). Этот алгоритм генерирует нормальные случайные величины на бинарном компьютере; предполагается, что имеется приблизительно $t+1$ двоичных разрядов точности. Он нуждается в таблице значений $d_j = a_j - a_{j-1}$ для $1 \leq j \leq t+1$, где a_j определяется

отношением

$$\sqrt{\frac{2}{\pi}}\int_{a_j}^{\infty} e^{-x^2/2}\,dx = \frac{1}{2^j}. \tag{23}$$

F1. [Получить U.] Генерировать равномерно распределенное число $U=(.b_0b_1\ldots b_t)_2$, где b_0, b_1, $\ldots$, b_t означают двоичные разряды в бинарной записи. Присвоить $\psi \leftarrow b_0$, $j \leftarrow 1$ и $a \leftarrow 0$.

F2. [Найти первый нуль b_j.] Если $b_j = 1$, присвоить $a \leftarrow a+d_j$, $j \leftarrow j+1$ и повторить этот шаг. (Если $j = t+1$, трактовать b_j как нуль.)

F3. [Генерировать X.] (Сейчас $a = a_{j-1}$, и текущее значение j появится с вероятностью $\approx 2^{-j}$. Будем генерировать X в интервале $[a_{j-1}\,..\,a_j)$, используя метод отбраковки, описанный выше, с $h(x) = x^2/2 - a^2/2 = y^2/2 + ay$, где $y = x - a$. В упр. 12 доказывается, что $h(x) \le 1$, как требуется в (21).) Присвоить $Y \leftarrow d_j$, умноженное на $(.b_{j+1}\ldots b_t)_2$ и $V \leftarrow (\frac{1}{2}Y + a)Y$. (Так как среднее значение j равно 2, обычно достаточно старших двоичных разрядов в $(.b_{j+1}\ldots b_t)_2$ для обеспечения приличной точности. Вычисления без труда выполняются арифметикой с фиксированной точкой.)

F4. [Отбраковать?] Генерируем равномерно распределенную случайную величину U. Если $V < U$, перейти к шагу F5. Иначе — присвоить V новую равномерно распределенную случайную величину. Если сейчас $U < V$ (т. е. если K четное, как обсуждалось выше), возвратиться к шагу F3, иначе — повторить шаг F4.

F5. [Выход из программы по X.] Присвоить $X \leftarrow a + Y$. Если $\psi = 1$, присвоить $X \leftarrow -X$. ▮

Значения d_j для $1 \le j \le 47$ появились в статье Аренса и Дитера (Ahrens and Dieter, *Math. Comp.* **27** (1973), 927–937). В ней обсуждается усовершенствованный алгоритм, который повышает скорость его выполнения за счет использования больших таблиц. Алгоритм F привлекателен тем, что работает почти с такой же скоростью, как алгоритм M, и его легче выполнить. Среднее число равномерно распределенных случайных величин на каждую нормальную случайную величину равно 2.53947. Р. П. Брент (R. P. Brent, *CACM* **17** (1974), 704–705) показал, как можно сократить это число до 1.37446 с помощью двух вычитаний и одного деления на каждую хранимую равномерно распределенную случайную величину.

4) *Отношение равномерно распределенных случайных величин.* Существует еще один хороший метод генерирования нормальных случайных величин, открытый в 1976 году А. Дж. Киндерманом (A. J. Kinderman) и Дж. Ф. Монаханом (J. F. Monahan). Его суть — генерирование случайной точки (U, V) в области, определенной как

$$0 < u \le 1, \qquad -2u\sqrt{\ln(1/u)} \le v \le 2u\sqrt{\ln(1/u)}, \tag{24}$$

и получение отношения $X \leftarrow V/U$. Заштрихованная площадь на рис. 13 — это магическая область неравенства (24), осуществляющая эту работу. Прежде чем изучать теорию, приведем алгоритм, эффективность и простота которого очевидны.

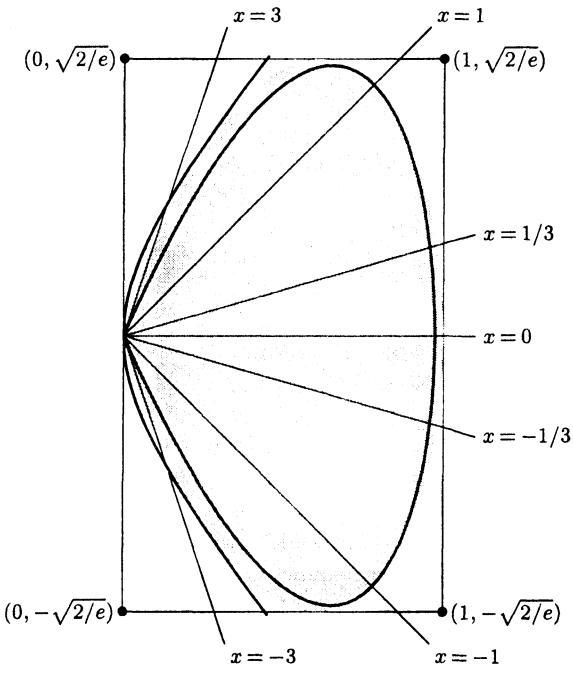

Рис. 13. Область "принятия" в методе отношения равномерных случайных величин для нормально распределенных случайных величин. Длины линий с отношением координат x распределены нормально.

Алгоритм R (*Метод отношений для нормальных случайных величин*). Этот алгоритм генерирует нормальные случайные величины X.

R1. [Получить U, V.] Генерировать две независимые равномерно распределенные случайные величины U и V, где $U \neq 0$, и присвоить $X \leftarrow \sqrt{8/e}\,(V - \frac{1}{2})\,/U$. (Сейчас X равно отношению координат $(U, \sqrt{8/e}\,(V - \frac{1}{2}))$ случайной точки в прямоугольнике, содержащем заштрихованную область на рис. 13. Принимаем X, если соответствующая точка в самом деле находится в заштрихованной области, иначе — начинаем сначала.)

R2. [Необязательная проверка верхней грани.] Если $X^2 \leq 5 - 4e^{1/4}U$, на выходе — X и завершение алгоритма. (Этот шаг можно опустить, если пожелаете; он так или иначе проверяет, находятся ли избранные точки во внешней области рис. 13, делая излишним вычисление логарифма.)

R3. [Необязательная проверка нижней грани.] Если $X^2 \geq 4e^{-1.35}/U + 1.4$, возвратиться к шагу R1. (Этот шаг также может быть опущен; он так или иначе проверяет, находятся ли выбранные точки за внешней областью рис. 13, делая ненужным вычисление логарифма).

R4. [Окончательная проверка.] Если $X^2 \leq -4 \ln U$, выход X и завершение алгоритма; иначе — возврат к шагу R1. ▮

В упр. 20 и 21 представлен временной анализ; анализируются четыре различных алгоритма, так как шаги R2 и R3 при желании могут быть включены или пропущены. В следующей таблице показано, сколько в среднем времени уходит на

выполнение каждого шага в зависимости от применения необязательной проверки.

Шаг	Ни одного	Только R2	Только R3	Оба
R1	1.369	1.369	1.369	1.369
R2	0	1.369	0	1.369
R3	0	0	1.369	0.467
R4	1.369	0.467	1.134	0.232

(25)

Таким образом, стоит отбросить необязательную проверку, если есть быстрые операции логарифмирования, но если подпрограмма вычисления логарифмов выполняется довольно медленно, то проверку стоит включить.

Но зачем делать эту работу? По одной единственной причине — чтобы вычислить вероятность того, что $X \le x$, которая оказывается точным значением (10). Но такое вычисление вряд ли будет очень простым, если не удастся найти хороший прием. Во всяком случае, в первую очередь, нужно понять, как может быть открыт такой алгоритм. Киндерман (Kinderman) и Монахам (Monahan) открыли его, разрабатывая следующую теорию, которая может использоваться с любой хорошо себя ведущей плотностью $f(x)$ [см. *ACM Trans. Math. Software* **3** (1977), 257–260].

Предположим, что точка (U, V) равномерно генерируется в области (u, v)-плоскости, определенной неравенствами

$$u > 0, \qquad u^2 \le g(v/u) \tag{26}$$

для некоторой неотрицательной интегрируемой функции g. Если присвоить $X \leftarrow V/U$, то вероятность того, что $X \le x$, можно вычислить путем интегрирования по $du\,dv$ по всей области, определенной двумя соотношениями в (26) и добавочным условием $v/u \le x$ с последующим делением на этот же интеграл, но без дополнительного условия. Допустим, $v = tu$, так что $dv = u\,dt$. Тогда интеграл имеет вид

$$\int_{-\infty}^{x} dt \int_{0}^{\sqrt{g(t)}} u\,du = \frac{1}{2}\int_{-\infty}^{x} g(t)\,dt.$$

Следовательно, вероятность, что $X \le x$, равна

$$\int_{-\infty}^{x} g(t)\,dt \Big/ \int_{-\infty}^{+\infty} g(t)\,dt. \tag{27}$$

Нормальное распределение возникает, когда $g(t) = e^{-t^2/2}$, а условие $u^2 \le g(v/u)$ упрощается в этом случае до $(v/u)^2 \le -4\ln u$. Легко видеть, что множество всех (u, v), удовлетворяющих этому соотношению, полностью содержится в прямоугольнике на рис. 13.

Грани шагов R2 и R3 определяют внутреннюю и внешнюю области простыми граничными уравнениями. Хорошо известное неравенство

$$e^x \ge 1 + x,$$

справедливое для всех действительных чисел x, можно использовать, чтобы показать, что неравенства

$$1 + \ln c - cu \;\le\; -\ln u \;\le\; 1/(cu) - 1 + \ln c \tag{28}$$

выполняются для любой константы $c > 0$. В упр. 21 доказывается, что $c = e^{1/4}$ является наилучшей возможной константой для использования на шаге R2. Более сложная ситуация складывается на шаге R3, и в этом случае, кажется, не существует простого выражения для оптимального c, но вычислительные эксперименты показывают, что наилучшее значение c для шага R3 приблизительно равно $e^{1.35}$. Аппроксимирующей кривой (28) является касательная к истинной границе, когда $u = 1/c$.

Существует возможность получения быстрого метода путем разделения области на подобласти, с большинством из которых иметь дело намного быстрее. Конечно, подразумевается, что нужны дополнительные таблицы, как в алгоритмах M и F. Интересная альтернатива, требующая меньшего числа вспомогательных таблиц, предложена Аренсом и Дитером в *CACM* **31** (1988), 1330–1337.

5) *Нормальная случайная величина из нормальной случайной величины.* В упр. 31 обсуждается интересный подход, который позволяет экономить время работы, так как сразу использует нормальные случайные величины вместо равномерно распределенных случайных величин. Этот метод, введенный в употребление в 1996 году К. С. Уэллесом (C. S. Wallace), в настоящее время не нашел достаточного теоретического обоснования, но удовлетворяет многим эмпирическим критериям.

6) *Преобразования нормального распределения.* До сих пор рассматривалось нормальное распределение со средним, равным нулю, и среднеквадратичным отклонением, равным единице. Если X имеет такое распределение, то

$$Y = \mu + \sigma X \tag{29}$$

имеет нормальное распределение со средним, равным μ, и среднеквадратичным отклонением, равным σ. Кроме того, если X_1 и X_2 — независимые нормальные случайные величины со средним, равным нулю, среднеквадратичным отклонением, равным единице, и

$$Y_1 = \mu_1 + \sigma_1 X_1, \qquad Y_2 = \mu_2 + \sigma_2\big(\rho X_1 + \sqrt{1-\rho^2}\, X_2\big), \tag{30}$$

то Y_1 и Y_2 — *зависимые* нормально распределенные случайные величины со средними, равными μ_1, μ_2, среднеквадратичными отклонениями, равными σ_1, σ_2, и коэффициентом корреляции ρ. (Для генерирования n переменных обратитесь к упр. 13.)

D. Показательное распределение. После равномерного и нормального распределений следующим наиболее важным распределением случайной величины является *показательное распределение.* Такие распределения появляются в ситуациях "время поступления". Например, если радиоактивное вещество излучает альфа-частицы так, что одна частица в среднем испускается каждые μ секунд, то время между двумя последовательными испусканиями имеет показательное распределение со средним, равным μ. Это распределение задается формулой

$$F(x) = 1 - e^{-x/\mu}, \qquad x \geq 0. \tag{31}$$

1) *Метод логарифма.* Очевидно, если $y = F(x) = 1 - e^{-x/\mu}$, то $x = F^{[-1]}(y) = -\mu \ln(1-y)$. Следовательно, $-\mu \ln(1-U)$ имеет экспоненциальное распределение

согласно (7). Поскольку $1 - U$ равномерно распределено, когда U имеет такое же распределение, можно сделать вывод, что

$$X = -\mu \ln U \tag{32}$$

имеет экспоненциальное распределение со средним, равным μ. (Случай, когда $U = 0$, должен трактоваться особо; его можно заменять любым подходящим значением ϵ, так как вероятность возникновения этого случая крайне мала.)

2) *Метод случайной минимизации.* Как было показано в алгоритме F, существует простой и быстрый способ альтернативного вычисления логарифма равномерной случайной величины. Следующий особенно эффективный подход разработали Дж. Марсалья, М. Сибая (M. Sibuya) и И. Г. Аренс (J. H. Ahrens) [см. *CACM* **15** (1972), 876–877].

Алгоритм S (*Экспоненциальное распределение со средним, равным* μ). Этот алгоритм вырабатывает экспоненциальные случайные величины на бинарном компьютере, используя равномерно распределенные случайные величины с точностью до $(t + 1)$ двоичных разрядов. Постоянные

$$Q[k] = \frac{\ln 2}{1!} + \frac{(\ln 2)^2}{2!} + \cdots + \frac{(\ln 2)^k}{k!}, \qquad k \geq 1, \tag{33}$$

могут вычисляться заранее до тех пор, пока они не превысят следующее значение: $Q[k] > 1 - 2^{-t}$.

S1. [Получить U и сдвинуть.] Генерировать $(t+1)$ двоичных разрядов равномерной случайной двоичной дроби $U = (.b_0 b_1 b_2 \ldots b_t)_2$; определить местоположение первого нулевого двоичного разряда b_j и избавиться от старших $j + 1$ двоичных разрядов с помощью присвоения $U \leftarrow (.b_{j+1} \ldots b_t)_2$. (Как и в алгоритме F, среднее число отбрасываемых двоичных разрядов равно 2.)

S2. [Немедленное принятие?] Если $U < \ln 2$, присвоить $X \leftarrow \mu(j \ln 2 + U)$ и завершить алгоритм. (Отметим, что $Q[1] = \ln 2$.)

S3. [Минимизация.] Найти наименьшее $k \geq 2$, такое, что $U < Q[k]$. Генерировать k новых равномерно распределенных случайных величин $U_1, \ldots, U_k$ и присвоить $V \leftarrow \min(U_1, \ldots, U_k)$.

S4. [Получение ответа.] Присвоить $X \leftarrow \mu(j + V) \ln 2$. ▮

Можно также использовать альтернативный метод генерирования случайных величин с показательным распределением (например, метод отношений равномерных случайных величин, как в алгоритме R).

E. Другие непрерывные распределения. Кратко рассмотрим, как обращаться с другими распределениями, которые достаточно часто возникают на практике.

1) *Гамма-распределение* порядка $a > 0$ определяется как

$$F(x) = \frac{1}{\Gamma(a)} \int_0^x t^{a-1} e^{-t}\, dt, \qquad x \geq 0. \tag{34}$$

При $a = 1$ оно является показательным распределением со средним, равным 1; при $a = \frac{1}{2}$ — это распределение случайной величины $\frac{1}{2}Z^2$, где Z — нормально

распределенная случайная величина (среднее равно 0, дисперсия — 1). Если X и Y — независимые случайные величины, имеющие гамма-распределение порядка a и b соответственно, то $X+Y$ имеет гамма-распределение порядка $a+b$. Так, например, сумма k независимых случайных величин, имеющих показательное распределение со средним, равным 1, имеет гамма-распределение порядка k. Если метод логарифма (32) использовался для генерирования таких случайных величин, имеющих показательное распределение, то здесь необходимо вычислить только один логарифм: $X \leftarrow -\ln(U_1 \ldots U_k)$, где U_1, ..., U_k — равномерно распределенные случайные величины, не равные нулю. Таким методом можно генерировать все случайные величины с гамма-распределением порядка a, где a — целое. Для завершения картины соответствующий метод для $0 < a < 1$ приведен в упр. 16.

Простой метод логарифма также очень медленный, когда a большое, поскольку он требует $\lfloor a \rfloor$ равномерно распределенных случайных величин. Более того, существует реальный риск, что произведение $U_1 \ldots U_{\lfloor a \rfloor}$ приведет к переполнению (потере плавающей точки). Для большого a следующий алгоритм, предложенный И. Г. Аренсом, является достаточно эффективным и легко записывается в терминах стандартных программ. [См. *Ann. Inst. Stat. Math.* **13** (1962), 231–237.]

Алгоритм A (*Гамма-распределение порядка* $a > 1$).

A1. [Генерирование кандидата.] Присвоить $Y \leftarrow \tan(\pi U)$, где U — равномерно распределенная случайная величина, и присвоить $X \leftarrow \sqrt{2a-1}\,Y + a - 1$. (Вместо $\tan(\pi U)$ можно использовать метод полярных координат, вычисляя отношение V_2/V_1, как на шаге P4 алгоритма P.)

A2. [Принимать?] Если $X \le 0$, возврат к шагу A1. Иначе — генерировать равномерно распределенную случайную величину V и возвратиться к шагу A1, если $V > (1+Y^2)\exp\bigl((a-1)\ln(X/(a-1)) - \sqrt{2a-1}\,Y\bigr)$. Иначе — принять X. ▮

Среднее время выполнения шага A1 < 1.902, когда $a \ge 3$.

Существует также заманчивый подход для больших a, основанный на таком замечательном факте, что гамма-распределение случайных величин приблизительно равно aX^3, когда X — нормально распределенная величина со средним значением $1 - 1/(9a)$ и среднеквадратичным отклонением $1/\sqrt{9a}$ [см. работы Э. Б. Уилсон (E. B. Wilson) и М. М. Гилферти (M. M. Hilferty), *Proc. Nat. Acad. Sci.* **17** (1931), 684–688, а также Дж. Марсалья (G. Marsaglia, *Computers and Math.* **3** (1977), 321–325)]*.

В некоторой степени усложненный, но значительно более быстрый алгоритм, который генерирует случайные величины, имеющие гамма-распределение, за приблизительно вдвое большее время, чем случайные величины, имеющие нормальное распределение, приведен в статье И. Г. Аренса и У. Дитера, *CACM* **25** (1982), 47–54, в которой содержится полезное описание принципов, используемых при построении алгоритма.

2) *Бета-распределение* с положительными параметрами a и b определяется как

$$F(x) = \frac{\Gamma(a+b)}{\Gamma(a)\,\Gamma(b)} \int_0^x t^{a-1}(1-t)^{b-1}\,dt, \qquad 0 \le x \le 1. \tag{35}$$

---

* Заменить "$+(3a-1)$" на "$-(3a-1)$" на шаге 3 алгоритма на с. 323.

Пусть X_1 и X_2 — независимые случайные величины, имеющие гамма-распределение соответственно порядка a и b. Присвоить $X \leftarrow X_1/(X_1+X_2)$. Другим полезным для малых a и b методом является использование присвоений

$$Y_1 \leftarrow U_1^{1/a} \qquad \text{и} \qquad Y_2 \leftarrow U_2^{1/b},$$

повторяемых до тех пор, пока не получится $Y_1+Y_2 \le 1$; тогда $X \leftarrow Y_1/(Y_1+Y_2)$. [См. работу М. Д. Йонка (M. D. Jöhnk, *Metrika* **8** (1964), 5–15).] Еще одним подходом, если a и b — целые числа, которые не настолько велики, является присвоение X b-й самой большой по величине из $a+b-1$ независимых равномерно распределенных случайных величин (см. упр. 9 в начале гл. 5). [См. также более прямой метод, описанный Р. Ч. С. Ченгом (R. C. H. Cheng, *CACM* **21** (1978), 317–322).]

3) *χ^2-распределение* с ν степенями свободы (3.3.1–(22)) можно получить, если положить, что $X \leftarrow 2Y$, где Y — случайная величина, имеющая гамма-распределение порядка $\nu/2$.

4) *F-распределение* (распределение отношения дисперсий) со степенями свободы ν_1 и ν_2 определено следующим образом:

$$F(x) = \frac{\nu_1^{\nu_1/2}\,\nu_2^{\nu_2/2}\,\Gamma\big((\nu_1+\nu_2)/2\big)}{\Gamma(\nu_1/2)\,\Gamma(\nu_2/2)} \int_0^x t^{\nu_1/2-1}(\nu_2+\nu_1 t)^{-\nu_1/2-\nu_2/2}\,dt, \tag{36}$$

где $x \ge 0$. Пусть Y_1 и Y_2 — независимые случайные величины, имеющие χ^2-распределение со степенями свободы ν_1 и ν_2 соответственно. Положим $X \leftarrow Y_1\nu_2/Y_2\nu_1$ или $X \leftarrow \nu_2 Y/\nu_1(1-Y)$, где Y_k, $k=1,2$, — случайные величины, имеющие бета-распределение с параметрами $\nu_1/2$ и $\nu_2/2$.

5) *t-распределение* с ν степенями свободы определено следующим образом:

$$F(x) = \frac{\Gamma\big((\nu+1)/2\big)}{\sqrt{\pi\nu}\,\Gamma(\nu/2)} \int_{-\infty}^x (1+t^2/\nu)^{-(\nu+1)/2}\,dt. \tag{37}$$

Пусть Y_1 — нормально распределенная случайная величина (среднее — 0, дисперсия — 1) и Y_2 — случайная независимая от Y_1 величина, имеющая χ^2-распределение с ν степенями свободы. Положим $X \leftarrow Y_1/\sqrt{Y_2/\nu}$. С другой стороны, когда $\nu > 2$, можно поступить следующим образом. Пусть Y_1 — нормально распределенная случайная величина (среднее — 0, дисперсия — 1) и Y_2 — независимая от нее случайная величина, имеющая показательное распределение со средним $2/(\nu-2)$. Положим $Z \leftarrow Y_1^2/(\nu-2)$ и отбросим (Y_1, Y_2), если $e^{-Y_2-Z} \ge 1-Z$. Иначе положим

$$X \leftarrow Y_1/\sqrt{(1-2/\nu)(1-Z)}.$$

Последний метод предложен Джорджем Марсальей, *Math. Comp.* **34** (1980), 235–236. [См. также работу А. Дж. Киндермана, Дж. Ф. Монахама и Дж. Дж. Рамаджа (A. J. Kinderman, J. F. Monahan, and J. G. Ramage, *Math. Comp.* **31** (1977), 1009–1018).]

6) *Случайные точки на n-мерной сфере единичного радиуса.* Пусть X_1, X_2, ..., X_n — независимые нормально распределенные случайные величины (среднее — 0,

дисперсия — 1); требуемая точка выбирается на единичной сфере следующим образом:

$$(X_1/r, X_2/r, \ldots, X_n/r), \qquad \text{где } r = \sqrt{X_1^2 + X_2^2 + \cdots + X_n^2}. \tag{38}$$

Если мы вычисляем X_k, используя метод полярных координат (алгоритм P), необходимо каждый раз вычислять две независимые случайные величины X_k, $k = 1, 2$, и в обозначениях алгоритма выполняется равенство $X_1^2 + X_2^2 = -2 \ln S$. Так можно сэкономить немного времени, требуемого для вычисления r. Справедливость (38) вытекает из того факта, что функция распределения точки $(X_1, \ldots, X_n)$ имеет плотность, зависящую только от расстояния от начала координат, поэтому при проецировании на единичную сферу она имеет равномерное распределение. Данный метод впервые был предложен Дж. В. Брауном (G. W. Brown, *Modern Mathematics for the Engineer*, First series, edited by E. F. Beckenbach (Бекенбах Э. Ф.), New York: McGraw-Hill, 1956, 302). Чтобы получить случайную точку *внутри* n-мерной сферы, Р. П. Брент (R. P. Brent) предложил взять точку на поверхности этой сферы и умножить ее координаты на $U^{1/n}$.

Для размерности 3 можно использовать значительно более простой метод, так как каждая координата этой точки равномерно распределена между -1 и 1. Если найти V_1, V_2 и S, используя шаги P1–P3 алгоритма P, искомая точка на поверхности шара будет иметь вид $(\alpha V_1, \alpha V_2, 2S - 1)$, где $\alpha = 2\sqrt{1 - S}$. [Robert E. Knop, *CACM* **13** (1970), 326.]

F. Важные целочисленные распределения. Вероятностные распределения на множестве целых чисел (случайные величины, принимающие только целые значения. — *Прим. ред.*) могут быть получены, в сущности, с помощью технических приемов, описанных в начале раздела, но некоторые из этих распределений настолько важны, что заслуживают специального упоминания.

1) *Геометрическое распределение.* Если некоторое событие происходит с вероятностью p, то число N независимых испытаний, проведенных до появления события (или до момента, когда событие происходит впервые), имеет геометрическое распределение, т. е. $N = 1$ с вероятностью p, $N = 2$ с вероятностью $(1 - p)p$, ..., $N = n$ с вероятностью $(1-p)^{n-1}p$. Это, по существу, ситуация, которая уже рассматривалась в разделе 3.3.2. Данное распределение непосредственно связано с числом циклов алгоритмов из настоящего раздела, как и циклы на шагах P1–P3 метода полярных координат.

Удобный метод генерирования случайной величины с таким распределением состоит в следующем:

$$N \leftarrow \lceil \ln U / \ln(1 - p) \rceil. \tag{39}$$

Чтобы проверить эту формулу, заметим, что $\lceil \ln U / \ln(1 - p) \rceil = n$ тогда и только тогда, когда $n - 1 < \ln U / \ln(1 - p) \leq n$, т. е. $(1 - p)^{n-1} > U \geq (1 - p)^n$ и это происходит с требуемой вероятностью $(1 - p)^{n-1}p$. Величину $\ln U$ можно также заменить величиной $-Y$, где Y имеет показательное распределение со средним 1.

Частный случай, когда $p = \frac{1}{2}$, совсем просто реализуется на бинарном компьютере, так как формула (39) сводится к присвоению $N \leftarrow \lceil -\lg U \rceil$, т. е. N становится на единицу больше числа старших нулевых разрядов в двоичном представлении числа U.

2) *Биномиальное распределение* (t,p). Если некоторое событие происходит с вероятностью p и проводится t независимых испытаний, то общее число N появлений этого события равно n с вероятностью $\binom{t}{n} p^n(1-p)^{t-n}$ (см. раздел 1.2.10). Другими словами, если генерируется U_1, ..., U_t, достаточно подсчитать, сколько из них не превосходят p. Для малых t с помощью этого метода легко можно получить точное значение N.

Для больших t можно генерировать случайную величину X, имеющую бета-распределение с целыми параметрами a и b, где $a+b-1=t$. При этом эффективно получаем b-й наибольший из t элементов, не беспокоясь о генерировании остальных элементов. Сейчас, если $X \geq p$, положим $N \leftarrow N_1$, где N_1 имеет биномиальное распределение $(a-1, p/X)$, так как оно показывает, сколько из $a-1$ случайных величин в области $[0..X)$ меньше, чем p. Если $X < p$, положим $N \leftarrow a+N_1$, где N_1 имеет биномиальное распределение $\big(b-1, (p-X)/(1-X)\big)$, так как N_1 показывает нам, сколько из $b-1$ случайных чисел в области $[X..1)$ меньше p. Выбирая $a = 1 + \lfloor t/2 \rfloor$, параметр t можно свести к разумной величине после примерно $\lg t$ преобразований такого рода. (Этот подход предложен И. Г. Аренсом, который также предложил альтернативный метод для значений параметра t средней величины; см. упр. 27.)

3) *Пуассоновское распределение* со средним μ. Пуассоновское распределение соотносится с показательным распределением, как биномиальное распределение с геометрическим: если некоторое событие может произойти в любой момент времени, то пуассоновское распределение — это распределение случайной величины, которая равна числу событий, происшедших в единицу времени (при некоторых других ограничениях. — *Прим. ред.*). Например, число альфа-частиц, которые испускаются радиоактивным веществом за одну секунду, имеет распределение Пуассона.

В соответствии с этим принципом можно получить пуассоновскую случайную величину N, генерируя независимые показательные случайные величины $X_1, X_2, \ldots$ со средним $1/\mu$ и останавливаясь, как только будет получено $X_1 + \cdots + X_m \geq 1$, где $N \leftarrow m-1$. Вероятность того, что $X_1 + \cdots + X_m \geq 1$, равна вероятности, что случайная величина, имеющая гамма-распределение порядка m, будет $\geq \mu$, а это равно $\int_\mu^\infty t^{m-1}e^{-t}\,dt/(m-1)!$. Следовательно, вероятность, что $N=n$, равна

$$\frac{1}{n!}\int_\mu^\infty t^n e^{-t}\,dt - \frac{1}{(n-1)!}\int_\mu^\infty t^{n-1}e^{-t}\,dt = e^{-\mu}\frac{\mu^n}{n!}, \qquad n \geq 0. \tag{40}$$

При генерировании показательной случайной величины методом логарифма, описанным выше, нужно остановиться, когда $-(\ln U_1 + \cdots + \ln U_m)/\mu \geq 1$. Упрощая это выражение, получим, что требуемую пуассоновскую величину можно получить, вычислив $e^{-\mu}$ до определенной точности и генерируя одну или более равномерно распределенных случайных величин U_1, U_2, ... до тех пор, пока их произведение не будет удовлетворять неравенству $U_1 \ldots U_m \leq e^{-\mu}$. Окончательно полагаем, что $N \leftarrow m-1$. В среднем нужно генерировать $\mu+1$ равномерно распределенную величину, поэтому рассматриваемый подход очень полезен, когда μ не слишком велико.

Когда μ большое, можно получить метод порядка $\log\mu$, используя сведения о том, как соотносятся гамма- и биномиальное распределения высокого порядка.

Сначала генерируем случайную величину X, имеющую гамма-распределение порядка $m = \lfloor \alpha\mu \rfloor$, где α — подходящая постоянная. (Так как X эквивалентно $-\ln(U_1 \dots U_m)$, по существу, производим m шагов предыдущего метода.) Если $X < \mu$, полагаем $N \leftarrow m + N_1$, где N_1 — пуассоновская случайная величина со средним $\mu - X$, а если $X \geq \mu$, то полагаем $N \leftarrow N_1$, где N_1 имеет биномиальное распределение $(m-1, \mu/X)$. Этот метод предложен И. Г. Аренсом и У. Дитером, которые предположили, что $\frac{7}{8}$ — хорошее значение для α.

Справедливость сформулированного утверждения, когда $X \geq \mu$, является следствием такого важного принципа: "Пусть $X_1, \dots, X_m$ — независимые показательные случайные величины с одинаковым средним и пусть $S_j = X_1 + \cdots + X_j$ и $V_j = S_j/S_m$ для $1 \leq j \leq m$. Тогда распределение $V_1, V_2, \dots, V_{m-1}$ такое же, как распределение $m-1$ независимых случайных величин, расположенных в порядке возрастания". Чтобы формально доказать этот принцип, подсчитаем вероятность того, что $V_1 \leq v_1, \dots, V_{m-1} \leq v_{m-1}$, если $S_m = s$ для произвольных значений $0 \leq v_1 \leq \cdots \leq v_{m-1} \leq 1$. Пусть $f(v_1, v_2, \dots, v_{m-1})$ — $(m-1)$-кратный интеграл

$$\int_0^{v_1 s} \mu e^{-t_1/\mu}\, dt_1 \int_0^{v_2 s - t_1} \mu e^{-t_2/\mu}\, dt_2 \dots \times \int_0^{v_{m-1}s - t_1 - \cdots - t_{m-2}} \mu e^{-t_{m-1}/\mu}\, dt_{m-1} \cdot \mu e^{-(s - t_1 - \cdots - t_{m-1})/\mu};$$

тогда

$$\frac{f(v_1, v_2, \dots, v_{m-1})}{f(1, 1, \dots, 1)} = \frac{\int_0^{v_1} du_1 \int_{u_1}^{v_2} du_2 \dots \int_{u_{m-2}}^{v_{m-1}} du_{m-1}}{\int_0^1 du_1 \int_{u_1}^1 du_2 \dots \int_{u_{m-2}}^1 du_{m-1}},$$

если произвести замену переменных $t_1 = su_1$, $t_1 + t_2 = su_2, \dots, t_1 + \cdots + t_{m-1} = su_{m-1}$. Последнее отношение соответствует вероятности того, что равномерно распределенные случайные величины $U_1, \dots, U_{m-1}$ удовлетворяют неравенствам $U_1 \leq v_1, \dots, U_{m-1} \leq v_{m-1}$, если задано, что они также удовлетворяют неравенствам $U_1 \leq \cdots \leq U_{m-1}$.

В некоторой степени более эффективным, но более сложным будет метод для биномиальной и пуассоновской случайных величин, предложенный в упр. 22.

G. Для дальнейшего чтения. Факсимиле письма фон Неймана (von Neumann), датированного 21 мая 1947 года, в котором метод отбраковки впервые увидел свет, появилось в *Stanisław Ulam 1909–1984*, в специальном выпуске *Los Alamos Science* (Los Alamos National Lab., 1987), 135–136. В книге *Non-Uniform Random Variate Generation* Л. Девроя (L. Devroye) (Springer, 1986) обсуждается намного больше алгоритмов генерирования случайных величин с неравномерным распределением; там же подробно рассматривается эффективность каждого метода для типичных компьютеров.

В. Херманн и Г. Дерфлингер [W. Hörmann and G. Derflinger, *ACM Trans. Math. Software* **19** (1993), 489–495] указали, что может быть опасно использовать метод отбраковки для линейных конгруэнтных генераторов с малыми множителями $a \approx \sqrt{m}$.

С теоретической точки зрения интересно рассмотреть *оптимальный* метод генерирования случайных величин с заданным распределением. Оптимальность здесь означает, что этот метод генерирует требуемую случайную величину, используя минимальное возможное число случайных разрядов. Начала этой теории можно

найти в книге Д. Э. Кнута и Э. Ч. Яо (D. E. Knuth and A. C. Yao, *Algorithms and Complexity*, edited by J. F. Traub (New York: Academic Press, 1976), 357–428).

Упр. 16 рекомендуется как обзор различных методов, приведенных в этом разделе.

УПРАЖНЕНИЯ

1. [*10*] Пусть α и β — действительные числа, $\alpha < \beta$. Как можно генерировать случайные действительные числа, равномерно распределенные между α и β?

2. [*M16*] Пусть mU — случайное целое число, принимающее значения между 0 и $m-1$ с одинаковыми вероятностями. Чему *точно* равна вероятность того, что $\lfloor kU \rfloor = r$, если $0 \le r < k$? Сравните с требуемой вероятностью $1/k$.

▶ **3.** [*14*] Как можно рассмотреть U в качестве целого числа и *разделить* его на k, чтобы получить случайное целое число между 0 и $k-1$, вместо того чтобы выполнять умножение, как предложено в разделе. Тогда (1) было бы заменено на

```
ENTA 0;  LDX U;  DIV K
```

с результатом в регистре X. Хороший ли это метод?

4. [*M20*] Докажите два соотношения в (8).

5. [*21*] Предложите эффективный метод генерирования случайной величины с функцией распределения $F(x) = px + qx^2 + rx^3$ ($0 \le x \le 1$. — *Прим. ред.*), где $p \ge 0$, $q \ge 0$, $r \ge 0$ и $p + q + r = 1$.

6. [*HM21*] Величина X получена следующим образом.

Шаг 1. Генерировать две независимые равномерно распределенные случайные величины U и V.

Шаг 2. Если $U^2 + V^2 \ge 1$, возврат к шагу 1; иначе — присвоить $X \leftarrow U$.

Какова функция распределения X? Сколько времени выполняется шаг 1? (Найдите среднее и среднеквадратичное отклонения).

7. [*20*] (А. Дж. Уолкер (A. J. Walker).) Предположим, что имеется набор кубиков k различных цветов, скажем n_j кубиков цвета C_j при $1 \le j \le k$, и k коробок $\{B_1, \ldots, B_k\}$, в каждую из которых можно поместить n кубиков. Кроме того, $n_1 + \cdots + n_k = kn$, так что кубики будут полностью заполнять коробки. Докажите (конструктивно), что всегда можно разместить кубики в коробках так, чтобы в каждой из них содержались кубики самое большее двух различных цветов. Действительно, можно устроить так, чтобы всякий раз в коробке B_j содержалось два цвета, один из которых — цвет C_j. Покажите, как использовать это утверждение, чтобы подсчитать P и Y, требуемые в (3), при заданном вероятностном распределении $(p_1, \ldots, p_k)$.

8. [*M15*] Покажите, что операцию (3) можно заменить операцией

$$\text{если} \quad U < P_K, \quad \text{то} \quad X \leftarrow x_{K+1}; \quad \text{иначе} \quad X \leftarrow Y_K.$$

(Таким образом, вместо V используется истинное значение U.) Будет ли это более удобным или подходящим для изменения P_0, P_1, ..., P_{k-1}.

9. [*HM10*] Почему кривая $f(x)$ на рис. 9 выпукла для Выпуклая кривая Вогнутая кривая $x < 1$ и вогнута для $x > 1$?

▶ **10.** [*HM24*] Объясните, как вычислить вспомогательные постоянные P_j, Q_j, Y_j, Z_j, S_j, D_j, E_j, чтобы алгоритм M давал ответ с правильным распределением.

▸ **11.** [*HM27*] Докажите, что шаги M7 и M8 алгоритма M порождают случайную величину с хвостом, близким к нормальному распределению, т. е. вероятность того, что $X \le x$ должна быть точно равна

$$\int_3^x e^{-t^2/2}\,dt \Big/ \int_3^\infty e^{-t^2/2}\,dt, \qquad x \ge 3.$$

[*Указание.* Покажите, что это частный случай метода отбраковки с $g(t) = Cte^{-t^2/2}$ для некоторого C.]

12. [*HM23*] (Р. П. Брент (R. P. Brent).) Докажите, что числа a_j, определенные в (23), удовлетворяют соотношению

$$a_j^2 - a_{j-1}^2 < 2\ln 2 \qquad \text{для всех } j \ge 1.$$

[*Указание.* Если $f(x) = e^{x^2/2}\int_x^\infty e^{-t^2/2}\,dt$, покажите, что $f(x) < f(y)$ для $0 \le x < y$.]

13. [*HM25*] Дано множество n независимых нормальных случайных величин X_1, X_2, ..., X_n со средним 0 и дисперсией 1. Покажите, как найти константы b_j и a_{ij}, $1 \le j \le i \le n$, такие, что если

$$Y_1 = b_1 + a_{11}X_1, \quad Y_2 = b_2 + a_{21}X_1 + a_{22}X_2, \quad \ldots, \quad Y_n = b_n + a_{n1}X_1 + \cdots + a_{nn}X_n,$$

то $Y_1, Y_2, \ldots, Y_n$ — зависимые нормально распределенные случайные величины, Y_j имеют среднее μ_j и заданную ковариационную матрицу (c_{ij}). (Ковариации c_{ij} между Y_i и Y_j определяются как среднее значение случайной величины $(Y_i - \mu_i)(Y_j - \mu_j)$. В частности, c_{jj} — это дисперсия Y_j, квадрат их среднеквадратичных отклонений. Не все матрицы (c_{ij}) могут быть ковариационными, и ваше построение, конечно, будет работать всякий раз, когда данное условие будет выполняться.)

14. [*M21*] Найдите распределение cX, если X — случайная величина с непрерывной функцией распределения $F(x)$ и если c — постоянная (возможно, отрицательная).

15. [*HM21*] Если X_1 и X_2 — независимые случайные величины с соответствующими функциями распределения $F_1(x)$ и $F_2(x)$ и плотностями $f_1(x) = F_1'(x)$, $f_2(x) = F_2'(x)$, какова функция распределения и плотность случайной величины $X_1 + X_2$?

▸ **16.** [*HM22*] (И. Г. Аренс.) Предложите алгоритм для генерирования случайной величины, имеющей гамма-распределение порядка a, когда $0 < a \le 1$, с помощью метода отбраковки с $cg(t) = t^{a-1}/\Gamma(a)$ для $0 < t < 1$ и с $cg(t) = e^{-t}/\Gamma(a)$ для $t \ge 1$.

▸ **17.** [*M24*] Чему равна *функция распределения* $F(x)$ случайной величины, имеющей геометрическое распределение с параметром p? Чему равна *производящая функция* $G(z)$? Чему равны среднее и дисперсия этого распределения?

18. [*M24*] Предложите метод вычисления случайной целочисленной величины N, такой, что N принимает значение n с вероятностью $np^2(1-p)^{n-1}$, $n \ge 0$. (Когда p сравнительно мало́, этот случай представляет особый интерес.)

19. [*22*] Случайная величина N с *отрицательным биномиальным распределением* с параметрами (t, p) принимает значения $N = n$ с вероятностью $\binom{t-1+n}{n}p^t(1-p)^n$. (В отличие от обычного биномиального распределения t не обязано быть целым, так как эта величина неотрицательна для всех n, каково бы ни было $t > 0$.) Обобщая упр. 18, объясните как генерировать целые случайные числа N с этим распределением, когда t — малое положительное целое число. Какой вы предложите метод для $t = p = \frac{1}{2}$?

20. [*M20*] Пусть A — площадь заштрихованной области на рис. 13, R — площадь содержащего ее прямоугольника, I — площадь внутренней области, определенной на шаге R2, и E — площадь между внутренней областью, отбрасываемой на шаге R3, и другой частью

прямоугольника. Определите длительность каждого шага алгоритма R для каждого из четырех вариантов в (25) в терминах A, R, I и E.

21. [*HM29*] Найдите формулы для величин A, R, I и E, определенных в упр. 20. (Для I и особенно для E можно использовать интерактивную алгебраическую систему.) Покажите, что $c = e^{1/4}$ — наилучшая возможная постоянная на шаге R2 для проверки "$X^2 \leq 4(1 + \ln c) - 4cU$".

22. [*HM40*] Можно ли получить точное пуассоновское распределение для больших μ, генерируя приближенно нормально распределенную величину, превращая ее в целочисленную некоторым подходящим способом и применяя (возможно, сложную) коррекцию в небольшом числе случаев?

23. [*HM23*] (Дж. фон Нейман.) Будут ли следующие два способа генерирования случайной величины X эквивалентны (т. е. будут ли случайные величины X при этом одинаково распределены)?

Метод 1. Присвоить $X \leftarrow \sin((\pi/2)U)$, где U — равномерно распределенная случайная величина.

Метод 2. Генерировать две равномерно распределенные случайные величины U и V; если $U^2 + V^2 \geq 1$, то повторять генерирование до тех пор, пока $U^2 + V^2 < 1$. Затем присвоить $X \leftarrow |U^2 - V^2|/(U^2 + V^2)$.

24. [*HM40*] (С. Улам (S. Ulam) и Дж. фон Нейман.) Пусть V_0 — случайно выбранное действительное число между 0 и 1. Определим последовательность $\langle V_n \rangle$ с помощью правила $V_{n+1} = 4V_n(1 - V_n)$. Если произвести вычисление с хорошей точностью, то в результате получится такая же последовательность, как случайная последовательность $\sin^2 \pi U_n$, где U_n — равномерно распределенные случайные величины, т. е. с функцией распределения члена последовательности, равной $F(x) = \int_0^x dx/\sqrt{2\pi x(1-x)}$. Если записать $V_n = \sin^2 \pi U_n$, то получим, что $U_{n+1} = (2U_n) \bmod 1$. Так как почти все действительные числа имеют случайное двоичное представление (см. раздел 3.5), полученная последовательность U_n будет равнораспределенной. Однако если вычисление V_n выполняется только с ограниченной точностью, то наши аргументы становятся несостоятельными, поскольку будет мешать ошибка округления. [См. работу фон Неймана *Collected Works* **5**, 768–770.]

Проанализируйте последовательность $\langle V_n \rangle$, определенную в разделе 3.3.4, когда вычисления производятся только с ограниченной точностью. Выполните эмпирический и теоретический анализ (для различных V_0). Будет ли последовательность иметь распределение, подобное ожидаемому?

25. [*M25*] Пусть X_1, X_2, ..., X_5 — двоичные слова, каждый двоичный разряд которых является независимой случайной величиной, принимающей значение 0 или 1 с вероятностью $\frac{1}{2}$. Чему равна вероятность того, что расположение данных двоичных разрядов $X_1 \vee (X_2 \wedge (X_3 \vee (X_4 \wedge X_5)))$ содержит 1? Обобщите результат.

26. [*M18*] Пусть N_1 и N_2 — независимые пуассоновские случайные величины со средними μ_1 и μ_2, где $\mu_1 > \mu_2 \geq 0$. Докажите или опровергните следующие утверждения: (a) $N_1 + N_2$ имеет распределение Пуассона со средним $\mu_1 + \mu_2$; (b) $N_1 - N_2$ имеет распределение Пуассона со средним $\mu_1 - \mu_2$.

27. [*22*] (И. Г. Аренс.) В большинстве бинарных компьютеров существует эффективный способ подсчета единиц в бинарном слове (см. раздел 7.1). Следовательно, существует хороший путь получения биномиального распределения (t, p), когда $p = \frac{1}{2}$, путем генерирования t случайных двоичных разрядов и подсчета числа единиц.

Предложите алгоритм, который генерирует случайные числа с биномиальным распределением (t, p) для *произвольного* p, используя в качестве источника случайных чисел только программу для частного случая, когда $p = \frac{1}{2}$. [*Указание.* Моделируйте сначала

процесс, который выглядит, как выбор самого старшего двоичного разряда равномерно распределенной случайной величины t, затем — как выбор второго двоичного разряда этой случайной величины, если старшего двоичного разряда недостаточно, чтобы определить, будет ли случайная величина $< p$, и т. д.]

28. [*HM35*] (Р. П. Брент.) Разработайте метод генерирования случайной точки на поверхности эллипсоида, определенного следующим образом: $\sum a_k x_k^2 = 1$, где $a_1 \geq \cdots \geq a_n > 0$.

29. [*M20*] (Дж. Л. Бентли (J. L. Bentley) и Дж. Б. Сакс (J. B. Saxe).) Найдите простой метод генерирования n равномерно распределенных между 0 и 1 чисел $X_1, \ldots, X_n$, требуя, чтобы они были упорядочены: $X_1 \leq \cdots \leq X_n$. Алгоритм должен состоять только из $O(n)$ шагов.

30. [*M30*] Объясните, как генерировать множество случайных точек (X_j, Y_j), таких, что если R — некоторый прямоугольник площадью α, содержащийся в единичной сфере, числа (X_j, Y_j), лежащие в R, имеют пуассоновское распределение со средним $\alpha\mu$.

31. [*HM39*] (*Непосредственное генерирование нормальных случайных величин.*)

a) Докажите, что если $a_1^2 + \cdots + a_k^2 = 1$ и $X_1, \ldots, X_k$ — независимые нормально распределенные случайные величины со средним 0 и дисперсией 1, то $a_1X_1 + \cdots + a_kX_k$ — нормальные случайные числа со средним 0 и дисперсией 1.

b) В доказательстве (a) предполагается, что можно генерировать новые нормальные случайные величины, используя старые точно так, как получают новые равномерно распределенные случайные величины из старых. Например, можно использовать идею 3.2.2–(7), но с рекуррентным соотношением, подобным

$$X_n = (X_{n-24} + X_{n-55})/\sqrt{2} \qquad \text{или} \qquad X_n = \tfrac{3}{5}X_{n-24} + \tfrac{4}{5}X_{n-55},$$

после получения множества нормальных случайных величин $X_0, \ldots, X_{54}$. Объясните, почему это *нехорошая* идея.

c) Покажите, тем не менее, что *существует* более быстрый по сравнению с другими метод генерирования нормальных случайных величин, использующий усовершенствованные идеи (a) и (b). [*Указание.* Если X и Y — независимые нормальные случайные величины, то такими же будут $X' = X\cos\theta + Y\sin\theta$ и $Y' = -X\sin\theta + Y\cos\theta$ для любых углов θ.]

32. [*HM30*] (К. С. Уоллес (C. S. Wallace).) Пусть X и Y — независимые случайные величины с показательным распределением со средним 1. Докажите, что X' и Y' также являются независимыми случайными величинами, имеющими показательное распределение со средним 1, если получить их из X и Y любым из следующих способов.

a) Задана $0 < \lambda < 1$,

$$X' = (1-\lambda)X - \lambda Y + (X+Y)[(1-\lambda)X < \lambda Y], \qquad Y' = X + Y - X'.$$

b) $(X', Y') = \begin{cases} (2X, Y-X), & \text{если } X \leq Y; \\ (2Y, X-Y), & \text{если } X > Y. \end{cases}$

c) Если X и Y имеют двоичное представление $X = (\ldots x_2x_1x_0.x_{-1}x_{-2}x_{-3}\ldots)_2$ и $Y = (\ldots y_2y_1y_0.y_{-1}y_{-2}y_{-3}\ldots)_2$, то X' и Y' имеют "перемешанные" значения

$$X' = (\ldots x_2y_1x_0.y_{-1}x_{-2}y_{-3}\ldots)_2, \qquad Y' = (\ldots y_2x_1y_0.x_{-1}y_{-2}x_{-3}\ldots)_2.$$

33. [*20*] Алгоритмы P, M, F и R генерируют нормальные случайные величины, поглощая неизвестное число равномерно распределенных , случайных величин U_1, U_2, ... Как их можно преобразовать, чтобы на выходе получить функцию только от одной случайной величины U?

3.4.2. Случайные выборки и перемешивания

В многочисленных случаях при обработке данных для беспристрастного выбора n случайных записей приходится обращаться к файлу, содержащему N записей. Такая проблема возникает, например, при контроле качества или других статистических вычислениях, где используются выборки. Обычно N очень велико, поэтому невозможно хранить сразу все данные в памяти, да и отдельные записи часто настолько большие, что в памяти нельзя содержать даже n записей. Поэтому необходима эффективная процедура для выбора n записей, которая определяла бы одно из двух — принимать или отбрасывать каждую запись при ее появлении, записывая принятую запись в выходной файл.

Для решения данной проблемы разработано несколько методов. Наиболее очевидным подходом является выбор каждой записи с вероятностью n/N. Возможно, иногда такой подход оправдан, но он дает в *среднем* только n записей в выборке. Среднеквадратичное отклонение равно $\sqrt{n(1-n/N)}$, и выборка может оказаться либо слишком большой либо слишком малой, чтобы дать необходимые результаты.

К счастью, простое преобразование "очевидной" процедуры позволяет получить то, что нужно: $(t+1)$-я запись будет выбрана с вероятностью $(n-m)/(N-t)$, если m записей уже выбраны. Эта вероятность обоснована, поскольку среди всех возможных способов такого выбора n записей из N, чтобы точно m записей было выбрано из первых t, доля способов, когда при этом выбирается также и $(t+1)$-я запись, равна

$$\binom{N-t-1}{n-m-1} \Big/ \binom{N-t}{n-m} = \frac{n-m}{N-t}. \tag{1}$$

(Иначе говоря, это вероятность выбора $(t+1)$-й записи при условии, что n записей из N выбирались так, что среди первых t было выбрано ровно m записей. — *Прим. ред.*)

Идея, развитая в предыдущем разделе, немедленно приводит к следующему алгоритму.

Алгоритм S (*Техника подбора выборки*). Выбрать n записей случайным образом из множества N, где $0 < n \le N$.

S1. [Инициализация.] Присвоить $t \leftarrow 0$, $m \leftarrow 0$. (В этом алгоритме m является числом записей, выбранных ранее, а t — общим числом введенных записей, с которыми мы работаем.)

S2. [Генерировать U.] Генерировать случайное число U, равномерно распределенное между 0 и 1.

S3. [Проверка.] Если $(N-t)U \ge n-m$, перейти к шагу S5.

S4. [Выбор.] Включить следующую запись в выборку и увеличить m и t на 1. Если $m < n$, перейти к шагу S2. Иначе выборка является полной и алгоритм завершен.

S5. [Пропустить.] Пропустить следующую запись (не включать ее в выборку), увеличить t на 1 и вернуться к шагу S2. ▮

На первый взгляд, алгоритм может показаться ненадежным и на самом деле неправильным, но тщательный анализ (см. упражнение, приведенное ниже) показывает, что он вполне надежен. Нетрудно проверить следующие факты.

a) Будет введено максимум N записей (мы никогда не достигнем конца файла, пока не выберем n элементов).

b) Выборка полностью беспристрастна. В частности, вероятность того, что любой заданный элемент выбран, как, например, последний элемент файла, равна n/N.

Утверждение (b) справедливо несмотря на то, что выбран $(t+1)$-й элемент *не* с вероятностью n/N, а с вероятностью, заданной в (1)! Это послужило причиной некоторой неразберихи в опубликованной литературе. Может ли читатель объяснить это кажущееся противоречие?

(*Замечание.* Используя алгоритм S, следует позаботиться о наличии различных источников случайных чисел U при каждом выполнении программы во избежание зависимости между выборками, полученными в разные дни. Это можно сделать, например, выбирая каждый раз различные значения X_0 в линейном конгруэнтном методе. Начальному значению X_0 может быть присвоено число месяца в день выполнения программы или последнее случайное число X, которое генерировалось при предыдущем прогоне программы.)

Обычно мы не проходим все N записей. В самом деле, так как указанное выше утверждение (b) гласит, что последняя запись выбирается с вероятностью n/N, вероятность того, что работа алгоритма завершится *до* выбора последней записи, точно равна $(1-n/N)$. Среднее число рассмотренных записей, когда $n=2$, приблизительно равно $\frac{2}{3}N$; общие формулы даны в упр. 5 и 6.

Алгоритм S и другие подобные методы обсуждаются в работе Ч. Т. Фана, Мервина Э. Мюллера и Ивана Резуха (C. T. Fan, Mervin E. Muller, and Ivan Rezucha, *J. Amer. Stat. Assoc.* **57** (1962), 387–402). Независимо этот метод был открыт Т. Г. Джонсом (T. G. Jones, *CACM* **5** (1962), 343).

Если значение N заранее неизвестно, то возникают трудности, поскольку знание точного значения N является решающим для выполнения алгоритма S. Допустим, необходимо случайно выбрать n записей из файла, если точно неизвестно, сколько записей в настоящее время содержится в этом файле. Можно сначала пересчитать все записи, а затем перейти к выбору n записей, но лучше так выбрать $m \geq n$ начальных записей на первом проходе, чтобы m было намного меньше N. Тогда на втором проходе нужно будет рассмотреть только m записей. Искусство состоит, конечно, в том, чтобы при подобном выборе в результате получить истинную случайную выборку из исходного файла.

Так как заранее неизвестно, когда завершится ввод, "модель" случайной выборки из записей следует хранить на входе при первом просмотре, чтобы всегда быть готовым к его окончанию. При чтении файла мы строим "резервуар", содержащий только записи, которые образуют предварительные выборки. Первые n записей всегда будут входить в резервуар. Когда $(t+1)$-я запись введена для $t \geq n$, в памяти формируется таблица n индексов записей, которые были выбраны среди первых t записей. Проблемой является сохранение этого состояния с t, увеличенным на единицу, т. е. поиск новой случайной выборки среди $t+1$ имеющихся известных сейчас записей. Нетрудно видеть, что новую запись следует включить в выборку с вероятностью $n/(t+1)$; при этом она заменит случайный элемент предыдущей выборки.

Итак, описанную работу выполняет следующий алгоритм.

Алгоритм R (*Резервуар выбора*). Выбрать случайно n записей из файла неизвестного размера $\geq n$, $n > 0$ задано. Дополнительный файл назовем резервуаром, содержащим все записи, которые являются кандидатами для окончательной выборки. Алгоритм использует таблицу различных индексов $I[j]$ для $1 \leq j \leq n$, каждый из которых указывает на одну из записей в резервуаре.

R1. [Инициализация.] Введем первые n записей и скопируем их в резервуар. Присвоим $I[j] \leftarrow j$ для $1 \leq j \leq n$ и присвоим $t \leftarrow m \leftarrow n$. (Если файл будет выборкой, имеющей меньше n записей, то, конечно, необходимо будет прервать выполнение алгоритма и сообщить о неблагоприятном исходе. В алгоритме индексы $I[1], \ldots, I[n]$ указывают записи в текущей выборке; m является размерностью резервуара, а t — номером входных записей, рассмотренных до сих пор.)

R2. [Конец файла?] Если записей на ввод больше нет, то перейти к шагу R6.

R3. [Генерирование и проверка.] Увеличить t на 1, затем генерировать случайное целое число M между 1 и t (включительно). Если $M > n$, перейти к шагу R5.

R4. [Увеличить резервуар.] Копировать следующую запись входного файла в резервуар, увеличить m на 1 и присвоить $I[M] \leftarrow m$. (Запись, предварительно указанная как $I[M]$, заменяется в выборке новой записью.) Возврат к шагу R2.

R5. [Пропустить.] Пропустить следующую запись входного файла (не включать ее в резервуар) и возвратиться к шагу R2.

R6. [Следующий просмотр.] Упорядочить индексы I таблицы входов так, чтобы $I[1] < \cdots < I[n]$, затем тщательно разобрать резервуар, копируя записи с этими индексами в выходной файл, который фиксирует окончательный выбор. ▮

Алгоритм R предложен Аланом Дж. Вотерманом (Alan G. Waterman). Читатель при желании может выполнить упр. 9, используя эти операции.

Если записи достаточно короткие, конечно, излишне помещать их в резервуар. n записей текущего файла можно постоянно хранить в памяти, и алгоритм станет более простым (см. упр. 10).

Относительно алгоритма R возникает естественный вопрос: "Какова предполагаемая размерность резервуара?". В упр. 11 показано, что среднее значение m точно равно $n(1 + H_N - H_n)$, т. е. приблизительно $n\big(1 + \ln(N/n)\big)$. Так, если $N/n = 1000$, в резервуаре будет содержаться лишь около $1/125$ целых записей исходного файла.

Заметим, что алгоритмы S и R можно применять, чтобы получать выборки для различных независимых классов одновременно. Например, если есть большой файл фамилий и адресов постоянных жителей США, можно случайным образом сделать выборку точно 10 жителей из каждого из 50 штатов, не просматривая 50 раз файл и не выполняя первую сортировку файла по штатам.

Возможно значительное улучшение алгоритмов S и R, когда n/N мало́. Скажите, сколько записей можно пропустить, вместо того чтобы решать, пропускать ли каждую запись, если генерируется единственная случайная величина? (См. упр. 8.)

Проблема выборки может быть рассмотрена как вычисление случайного *сочетания* в соответствии с обычным определением сочетания N по n (см. раздел 1.2.6). Рассмотрим задачу вычисления случайной *перестановки* t объектов. Назовем ее

задачей *перемешивания* (*тасования*), так как, например, тасование колоды карт является ничем иным, как ее случайной перестановкой.

Достаточно легко убедить любого игрока в карты, что традиционная процедура тасования карт ужасно несовершенна. Нет надежды получить таким методом каждую из $t!$ перестановок с примерно равными вероятностями. Рассказывают, что знаток игры в бридж использует это обстоятельство, когда принимает решение, стоит ли ему блефовать. По крайней мере семь "перемешиваний" колоды из 52 карт происходит с вероятностью, примерно равной 10% от вероятности получить эти перемешивания при равномерном распределении, в то время как 14 случайных перемешиваний имеют вероятность появления, примерно равную вероятности их получения при равномерном распределении [см. Aldous and Diaconis, *AMM* **93** (1986), 333–348].

Если t мало́, можно получить случайную перестановку очень быстро, генерируя случайное целое число между 1 и $t!$. Например, когда $t = 4$, случайного числа между 1 и 24 достаточно, чтобы выбрать случайную перестановку из таблицы всех возможностей. Но для больших t необходимо быть более осторожным, если требуется, чтобы каждая перестановка была равновероятной, так как $t!$ намного больше точности отдельного случайного числа.

Подходящая процедура перемешивания может быть получена с помощью уже упоминаемого алгоритма 3.3.2P, который дает простое соответствие между каждой из $t!$ возможных перестановок и последовательностью чисел $(c_1, c_2, \ldots, c_{t-1})$ с $0 \le c_j \le j$. Можно легко получить такое множество случайных чисел, и это соответствие можно использовать для получения случайных перестановок.

Алгоритм P (*Перемешивание*). Пусть X_1, X_2, ..., X_t — множество t чисел для перемешивания.

P1. [Инициализация.] Присвоить $j \leftarrow t$.

P2. [Генерирование U.] Генерировать случайное число U, равномерно распределенное между 0 и 1.

P3. [Замена.] Присвоить $k \leftarrow \lfloor jU \rfloor + 1$. (Сейчас k — случайное целое число между 1 и j. В упр. 3.4.1–3 объясняется, что деление на j *не* следует использовать для вычисления k.) Заменим $X_k \leftrightarrow X_j$.

P4. [Уменьшение j.] Уменьшить j на 1. Если $j > 1$, возвратиться к шагу P2. ▮

Впервые этот алгоритм опубликовали Р. А. Фишер и Ф. Ятс (R. A. Fisher and F. Yates, *Statistical Tables* (London, 1938), Example 12) на обычном языке, а Р. Дурстенфелд — на языке компьютерном (R. Durstenfeld, *CACM* **7** (1964), 420). Чтобы просто генерировать случайную перестановку $\{1, \ldots, t\}$ вместо перемешивания заданной последовательности $(X_1, \ldots, X_t)$, можно избежать операции замены $X_k \leftrightarrow X_j$, выполнив увеличение j от 1 до t и присвоив $X_j \leftarrow X_k$, $X_k \leftarrow j$ [см. D. E. Knuth, *The Stanford GraphBase* (New York: ACM Press, 1994), 104].

Р. Салфи (R. Salfi, *COMPSTAT 1974* (Vienna, 1974), 28–35) указал, что алгоритм P не имеет возможности генерировать более m различных перестановок, когда мы получаем равномерно распределенные U_j из линейной конгруэнтной последовательности по модулю m или используем рекуррентное соотношение $U_{n+1} = f(U_n)$,

для которого U_n может дать только m различных значений, потому что окончательная перестановка в таком случае полностью определяется первыми генерированными значениями U. Так, например, если $m = 2^{32}$, определенные перестановки из 13 элементов никогда не появятся, поскольку $13! \approx 1.45 \times 2^{32}$. В большинстве случаев на самом деле нет *необходимости* получать все 13! перестановок, но огорчает, что исключение данных перестановок определяется фактически простым математическим правилом, таким как решетчатая структура (см. раздел 3.3.4).

Эта проблема не возникает, когда используется генератор Фибоначчи с запаздыванием, подобный 3.2.2–(7), с достаточно длинным периодом. Но даже таким методом невозможно получать все перестановки постоянно, если нельзя точно задать по крайней мере $t!$ различных начальных значений для инициализации генератора. Другими словами, не существует возможности получить $\lg t!$ истинно случайных двоичных разрядов на выходе, если не получить $\lg t!$ истинно случайных двоичных разрядов на входе. Как говорится в разделе 3.5, не стоит из-за этого огорчаться.

Алгоритм S можно легко преобразовать для получения случайной перестановки случайных сочетаний (см. упр. 15). Случайные объекты комбинаторики других видов (например, разбиения) рассматриваются в разделе 7.2 и в книге *Combinatorial Algorithms* by Nijenhuis and Wilf (New York: Academic Press, 1975).

УПРАЖНЕНИЯ

1. [*M12*] Объясните (1).

2. [*20*] Докажите , что алгоритм S никогда не пытается прочесть более N записей своего входного файла.

▶ **3.** [*22*] $(t+1)$-я запись в алгоритме S выбирается с вероятностью $(n-m)/(N-t)$, а не n/N, однако алгоритм требует, чтобы выборка была беспристрастной, поэтому каждая запись должна выбираться с *одинаковой* вероятностью. Почему оба эти утверждения правильны?

4. [*M23*] Пусть $p(m,t)$ — вероятность того, что выбрано точно m записей среди первых t в методе отбора выборок. Покажите непосредственно из алгоритма S, что

$$p(m,t) = \binom{t}{m}\binom{N-t}{n-m}\Big/\binom{N}{n} \qquad \text{для } 0 \le t \le N.$$

5. [*M24*] Чему равно среднее значение t по завершении алгоритма S? (Другими словами, сколько из N записей будет просмотрено в среднем, прежде чем выборка станет полной?)

6. [*M24*] Чему равно среднеквадратичное отклонение значения, вычисленного в упр. 5?

7. [*M25*] Докажите, что любой *заданный* выбор n записей из множества N записей может быть получен алгоритмом S с вероятностью $1/\binom{N}{n}$. Следовательно, выборка является совершенно беспристрастной.

▶ **8.** [*M39*] (Д. С. Виттер (J. S. Vitter).) Алгоритм S производит одну равномерно распределенную случайную величину для каждой входной записи. Цель этого упражнения — рассмотреть более эффективный подход, при котором можно быстрее вычислить точное число X входных записей, которые следует пропустить, прежде чем сделать первый выбор.

a) Чему равна вероятность того, что $X \ge k$, k задано?

b) Покажите, что результат (a) позволяет выполнить вычисление X, генерируя только одну равномерно распределенную случайную величину U, и производит $O(X)$ других вычислений.

c) Покажите, что можно также присвоить $X \leftarrow \min(Y_N, Y_{N-1}, \ldots, Y_{N-n+1})$, где Y_t независимы и каждое Y_t является случайным целым числом в области $0 \le Y_t < t$.

d) Найдите максимальную скорость, т. е. покажите, что X также может быть вычислено в среднем за $O(1)$ шагов, если использовать "метод втискивания", подобный 3.4.1–(18).

9. [*12*] Пусть $n = 3$. Если алгоритм R применяется к файлу, содержащему 20 записей, которые пронумерованы от 1 до 20, и если случайные числа, генерированные на шаге R3, равны соответственно

$$4, 1, 6, 7, 5, 3, 5, 11, 11, 3, 7, 9, 3, 11, 4, 5, 4,$$

какая запись попадет в резервуар? Какая из них окажется в окончательной выборке?

10. [*15*] Преобразуйте алгоритм R так, чтобы обойтись без резервуара, предполагая, что n записей текущей выборки можно хранить в памяти.

▶ **11.** [*M25*] Пусть p_m — вероятность того, что точно m элементов занесены в резервуар в течение первого прохода алгоритма R. Определите производящую функцию $G(z) = \sum_m p_m z^m$ и найдите среднее значение и среднеквадратичное отклонение. (Используйте идеи из раздела 1.2.10.)

12. [*M26*] Суть алгоритма P состоит в том, что любую перестановку π можно однозначно записать как результат транспонирования в виде $\pi = (a_t t)\ldots(a_3 3)(a_2 2)$, где $1 \le a_j \le j$ для $t \ge j > 1$. Докажите, что существует также единственное представление вида $\pi = (b_2 2)(b_3 3)\ldots(b_t t)$, где $1 \le b_j \le j$ для $1 < j \le t$, и составьте алгоритм для вычисления b_k через a_k за $O(t)$ шагов.

13. [*M23*] (С. В. Голомб (S. W. Golomb).) Один из наиболее простых способов тасования игральных карт — разделение колоды карт на две максимально равные части и их тасование вместе. (В правилах Хойла карточных игр при обсуждении профессиональной этики игроков в карты сказано: "Тасование такого вида можно выполнить приблизительно трехкратным тщательным перемешиванием игральных карт".) Рассмотрим колоду из $2n-1$ игральных карт $X_1, X_2, \ldots, X_{2n-1}$. "Идеальное перемешивание" — это разделение s раз этой колоды на части $X_1, X_2, \ldots, X_n$ и $X_{n+1}, \ldots, X_{2n-1}$. Чередуя их, получим X_1, X_{n+1}, X_2, X_{n+2}, $\ldots$, X_{2n-1}, X_n. Операция "разрезания" c^j меняет $X_1, X_2, \ldots, X_{2n-1}$ на $X_{j+1}, \ldots, X_{2n-1}, X_1, \ldots, X_j$. Покажите, что, комбинируя идеальное перемешивание и разрезание, можно получить не более $(2n-1)(2n-2)$ различных упорядочений компоновок игральных карт, если $n > 1$.

14. [*22*] Разрезав и перетасовав перестановку $a_0 a_1 \ldots a_{n-1}$, можно заменить ее последовательностью, содержащей подпоследовательности

$$a_x\, a_{(x+1) \bmod n} \cdots a_{(y-1) \bmod n} \qquad \text{и} \qquad a_y\, a_{(y+1) \bmod n} \cdots a_{(x-1) \bmod n},$$

которые перемешаны определенным способом для некоторых x и y. Таким образом, последовательность 3890145267 является разрезанием и перетасовкой 0123456789 с $x = 3$ и $y = 8$.

a) Начав с расположенных обычным образом 52 игральных карт

2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A,
♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠

мистер Д. Г. Квик (J. H. Quick) ("Студент") сделал случайное разрезание и перетасовку, затем убрал крайнюю слева карту и вставил ее в случайное место. Получилась последовательность

9 10 K J Q A K A 2 Q 3 2 3 4 5 6 7 4 8 9 5 10 6 J 7 Q 8 K 9 10 J Q A K 2 3 A 4 2 3 4 5 6 5 6 7 8 7 9 10 J 8.
♣ ♣ ♡ ♣ ♡ ♣ ♣ ♢ ♡ ♢ ♠ ♠ ♠ ♠ ♠ ♢ ♠ ♠ ♢ ♠ ♢ ♠ ♢ ♠ ♢ ♠ ♢ ♢ ♢ ♢ ♠ ♢ ♣ ♣ ♢ ♣ ♡ ♡ ♡ ♣ ♡ ♡ ♡ ♡ ♣ ♡ ♡ ♡ ♣

Какую карту он убрал из крайней слева позиции?

b) Начав снова с колоды в ее обычном порядке, Квик сделал *три* разрезания и перетасовки, прежде чем сдвинул крайнюю слева карту на новое место:

Какую карту он сдвинул на этот раз?

▸ **15.** [*30*] (Оле-Йохан Дахл (Ole-Johan Dahl).) Если $X_k = k$ для $1 \le k \le t$ в начале алгоритма P и если остановить алгоритм, когда j достигнет значения $t-n$, последовательность $X_{t-n+1}, \ldots, X_t$ станет случайной перестановкой случайных комбинаций из n элементов. Покажите, как эффективно смоделировать эту процедуру, используя только $O(n)$ ячеек памяти.

▸ **16.** [*M25*] Придумайте метод вычисления случайной выборки по n записей из N при заданных N и n, основываясь на идее рандомизации (раздел 6.4). Можете использовать в нем $O(n)$ ячеек для хранения и в среднем $O(n)$ единиц времени. Метод должен представлять выборку как упорядоченное множество целых чисел $1 \le X_1 < X_2 < \cdots < X_n \le N$.

17. [*M22*] (P. B. Флойд (R. W. Floyd).) Докажите, что следующий алгоритм генерирует случайную выборку S из n целых чисел из $\{1, \ldots, N\}$. Присвойте сначала $S \leftarrow \emptyset$, а затем для $j \leftarrow N-n+1, N-n+2, \ldots, N$ (в таком порядке) присвойте $k \leftarrow \lfloor jU \rfloor + 1$ и

$$S \leftarrow \begin{cases} S \cup \{k\}, & \text{если } k \notin S; \\ S \cup \{j\}, & \text{если } k \in S. \end{cases}$$