Генерирование случайных чисел

Бочарова Ирина Викторовна
МАОУ «СОШ№7», г. Екатеринбург,
преподаватель
Кислова Мария Андреевна
МАОУ «СОШ№7», г. Екатеринбург,
обучающийся

АННОТАЦИЯ. Рассмотрены исторический экскурс и некоторые аспекты разработки применения датчиков случайных чисел. В качестве примера приведена программа формирования датчика случайных чисел с использование линейного конгруэнтного метода.

КЛЮЧЕВЫЕ СЛОВА. Случайные числа, датчик случайных чисел, алгоритм, программа.

«Генерация случайных чисел слишком важна, чтобы
оставлять её на волю случая»
Роберт Р. Кавью

«Всякий, кто питает слабость к арифметическим
методам получения случайных чисел,
грешен вне всяких сомнений»
Джон фон Нейман

Случайные числа имеют большую область применения. Некоторые области, на наш взгляд, наиболее зависящие от применения случайных чисел, приведены ниже.

В связи с компьютеризацией почти всех сфер нашей жизнедеятельности, компьютерная безопасность становится все более злободневной проблемой. Наши пароли, трафик, покупки, сбережения и интеллектуальная собственность – все это представляет собой драгоценную информацию для злоумышленников, более того, ради столь заманчивого способа разбогатеть в больших масштабах на какие только ухищрения они не идут. Поэтому этим данным необходимо обеспечить надежную защиту, которая, к тому же, будет постоянно развиваться и совершенствоваться.

Всем системам компьютерной безопасности, в которых применяется криптография, необходимы случайные числа. Их, как правило, используют как ключ к шифрованию – пока другой человек не знает, как выглядит ваш ключ, ваши данные в безопасности.

Случайные числа имеют немалый вес во многих научных сферах. Особенно полезны они оказываются в статистике, моделировании и постановке экспериментов. Например, случайная выборка при сборе данных позволяет за более короткое время получить ту же информацию, что и при полной. В моделировании с помощью случайных чисел можно воспроизводить физические явления, являющиеся по своей природе стохастическими, а в математическом моделировании случайные числа выступают как один из инструментов численного анализа.

Ну и, наконец, случайные числа позволяют проводить тестирование эффективности компьютерных алгоритмов и проводить отладку программы, кроме того, рандомизированные алгоритмы оказываются намного надежнее детерминированных (возвращающих те же выходные значения при тех же входных значениях).

В связи с этим возникает вопрос: каким образом можно раздобыть истинную случайность? Какое-то время эту задачу выполняли простейшие генераторы случайных чисел, такие как кости или лототроны. Но, разумеется, такие приспособления были малоэффективны. Например, ученым для постановки экспериментов требовалось большое количество стохастических данных и получить случайные в необходимом количестве не представляется возможным ни при каких условиях.

С этой целью были изобретены таблицы, состоящие из последовательности целых чисел, которые представляют результат простого случайного выбора из совокупности цифр от 0 до 9. Простой случайный выбор обеспечивает каждой цифре близкую к одинаковой вероятность появления в последовательности, в данном случае – к 0,1. Это значит, что в идеальной таблице случайных чисел, состоящих в совокупности из n цифр, каждая цифра должна появиться минимум $\lfloor n \cdot 10^{-1} \rfloor$ и максимум $\lceil n \cdot 10^{-1} \rceil$ раз.

Первопроходцем в создании таблиц случайных чисел был статистик Л. Г. К. Типпетт. Публикация Типпетта (Random Sampling Numbers) 1927 года содержала 41 600 цифр, взятых из отчетов о переписи и объединенных в 10 400 четырехзначных чисел.

В следующей таблице, за авторством Роналда Фишера и Фрэнка Йейтса, были использованы данные с логарифмических таблиц, составившие 7500 двузначных случайных чисел. И, наконец, в 1939 году Морис Джордж Кендалл и Б. Бабингтон-Смит

изобрели механический генератор случайных чисел для создания таблицы в 100 000 случайных цифр.

Компьютер Ferranti Mark 1, 1951 года производства, имел программу, использующую резистивный генератор дробового шума для поступления 20 случайных битов на сумматор. Данный функционал был предложен Аланом Тьюрингом. С помощью этой программы, к слову, в 1953 году было создано одно из первых произведений цифрового искусства – генератор любовных писем, написанный Кристофером Стрэчи, коллегой Тьюринга. Программа могла генерировать, как становится ясно из названия, различные любовные сообщения. Например,

«JEWEL LOVE

MY LIKING HUNGERS FOR YOUR ADORABLE INFATUATION. YOU ARE MY EROTIC ARDOUR. MY FOND RAPTURE. MY THIRST SIGHS FOR YOUR INFATUATION. MY HEART SEDUCTIVELY WISHES YOUR BREATHLESS LONGING.

YOURS CURIOUSLY

M. U. C.»

Или

«DEAR HONEY

YOU ARE MY COVETOUS LUST: MY UNSATISFIED ARDOUR: MY TENDER INFATUATION. MY LOVESICK CHARM CURIOUSLY CARES FOR YOUR WISH. MY AVID AFFECTION HUNGERS FOR YOUR ENCHANTMENT.

YOURSSEDUCTIVELY

M. U. C.»

Причем под инициалами в конце значился компьютер Манчестерского университета (Manchester University Computer). Эта программа по своему предназначению уже была самоиронична: оказывается, нет ничего клишированнее любовного письма!

В 1955 году аналитический центр RAND Corporation опубликовал книгу «A Million Random Digitswith 100,000 Normal Deviates» (Миллион случайных цифр и сто тысяч нормальных отклонений). Первые 50 пятизначных чисел этой таблицы приведены в таблице (см. Приложение, таблица 2).

В 1957 году свет увидело изобретение команды, стоявшей за созданием «Колосса» (компьютера 1943 года для расшифровки немецких радиосообщений), и в особенности

Томаса Флауэрса и Гарри Уильяма Фенсома – ERNIE (Electronic Random Number Indicator Equipment), которое генерировало случайные значения из шумового импульса в неоновой трубке. Этот генератор нашел применение в определении выигрышных билетов британской лотереи.

Технологический прогресс и изобретение компакт-дисков возродило применение таблиц случайных чисел. Диск 1995 года с 650 Мбайт случайных чисел и тестом diehard, разработанный Джорджем Марсальей, содержал числа, основанные на дробовом шуме диодной цепи и обработанной рэп-музыке, сам Марсалья называл это «белым и черным шумом».

И все же, таблица случайных чисел – это внешний ресурс для хранения данных, она выдает ограниченные и предопределенные значения. Возникла потребность в более продуктивном способе генерирования случайных чисел.

Все вышеописанные генераторы являются аппаратными, они выдают настоящие случайные числа, так как берут данные из физических и хаотических процессов. Они активно применяются в современном мире, но имеют несколько существенных минусов:

- сложность разработки аппаратного генератора случайных чисел, выдающего равномерные значения;
- медленный процесс генерирования.

Тогда было решено генерировать случайные числа математическим способом. Конечно же, истинно случайные числа подобным способом не получить, но оно и необязательно – главное, чтобы невозможно было предугадать или воспроизвести число. Значения, генерируемые таким образом, называют псевдослучайными, а генераторы, их производящие, – генераторами псевдослучайных чисел.

В 1946 году Джон фон Нейман предложил метод серединных квадратов. Метод заключался в том, что первое число возводилось в квадрат, из полученного числа выделялись цифры посередине, и они составляли следующее число, и так далее. Но на числе, оканчивавшемся нулями, а также на 6100, 2100, 4100 или 8100 данный алгоритм становился циклом с слишком малым периодом. В первом случае шла последовательность из нулей, во втором – последовательность 6100 – 2100 – 4100 – 8100 – 6100 и так далее.

Дональд Кнут в 1959 году также предложил свой метод «Алгоритм К».

К1. [Выбрать число итераций.] Присвоить Y наибольшую значащую цифру X. (Мы выполним шаги К2-К13 точно Y+1 раз, т. е. применим рандомизированные преобразования случайное число раз.)

К2. [Выбрать случайный шаг] Присвоить следующую наибольшую значащую цифру X. Переходим к шагу К(3 + Z), т. е. к случайно выбранному шагу в программе.

К3. [Обеспечить > 5 x 109] Если X < 5000000000, присвоить X значение X + 5000000000.

К4. [Средина квадрата.] Заменить X серединой квадрата X.

К5. [Умножить.] Заменить X числом (1001001001 X) mod 1010.

К6. [Псевдодополнение.] Если X < 100000000, то присвоить X значение X + 9814055677; иначе присвоить X значение 1010- X.

К7. [Переставить половины.] Поменять местами пять младших по порядку знаков со старшими.

К8. [Умножить.] Выполнить шаг К5.

К9. [Уменьшить цифры.] Уменьшить каждую не равную нулю цифру десятичного представления числа X на единицу.

К10. [Модифицировать на 99999.] Если A' < 105, присвоить X значение — X 2 +99999; иначе присвоить X значение X — 99999.

К11. [Нормировать.] (На этом шаге A' не может быть равным нулю.) Если X <109, то умножить X на 10.

К12. [Модификация метода средин квадратов.] Заменить X на средние 10 цифр числа X(X — 1).

К13. [Повторить?] Если Y > 0, уменьшить У на 1 и возвратиться к шагу К2. Если Y = 0, алгоритм завершен. Значение числа X, полученное на предыдущем шаге, и будет желаемым «случайным» значением.

Несмотря на свою сложную структуру, этот алгоритм порой сводился к исходному числу через малое количество шагов, что исключало возможность получения большого количества случайных чисел.

И все же, эта попытка не была бесполезной. На ее примере разработчики датчиков случайных чисел убедились, что для получения псевдослучайной последовательности недостаточно лишь бездумно написать много сложных математических операций.

Наиболее широкое распространение получил линейный конгруэнтный метод, предложенный в 1949 году Дерриком Генри Лемером. Последовательность случайных X_n чисел задается с помощью следующей формулы:

$X_{n+1} = (aX_n + c)\ mod\ m$,

где X_{n+1} – искомое случайное число, X_n – предыдущее число в последовательности, a –множитель, c – приращение, m – модуль.

В случае с a и c интуитивно понятен их функционал, и можно догадаться о том, зачем в формуле нужен модуль можно не сразу. Но мы попробуем.

Представим себе циферблат часов. Механические часы имеют только 12 часовых делений (это же и есть, по сути, ограничение по модулю), тогда как электронные обычно отображают 24-часовой формат. Представим, что сейчас время уже за полдень, наши механические часы немного отстают, а электронные показывают верное время. Как же тогда мы можем узнать, сколько делений нужно поставить на отстающих часах, чтобы они отображали верное время? Можно просто найти остаток числа на электронных часах от деления на 12!

Дело в том, что функция $X_{n+1}(X_n)$ при отсутствии $mod\ m$ будет мало того что монотонна на всей области определения, так еще и линейна. Последовательность типа 1 – 4 – 13 – 40 – 121 – … $(a = 3; c = 1)$ совершенно не соответствует определению случайной. А теперь представим, как будет вести себя уравнение с остатком от деления: теперь это и не функция вовсе, а все получаемые числа будут ограничены значением m,так как остаток от деления на число может быть лишь меньше самого числа. Для наглядности, создадим последовательность по $X_{n+1} = (X_n + 5)\ mod\ 12$, где $a = 1; c = 5;\ m = 12$.

Посмотрим на значение модуля и заметим, что для нахождения значений последовательности мы можем воспользоваться тем же циферблатом, только вместо 12 поставим число 0. Теперь, нужно придумать первое число нашей последовательности, оно обычно называется семенем (англ. seed). Пусть мы начнем с числа 1, тогда наша последовательность будет выглядеть так: 1 – 6 – 11 – 4 – 9 – 2 – 7 – 0 – 5 – 10 – 3 – 8 – 1 – 6 – 11 – … Получившаяся у нас последовательность повторяется каждый раз через 12 значений, то есть мы получили все целочисленные из промежутка $[0;m]$, но так происходит не при всех возможных значениях a и c. Например, если мы возьмем $c = 6$, при все тех же $m = 12,\ a = 1$, то последовательность будет повторяться через каждые два

значения при любом семени: 1 – 7 – 1 – ... или 0 – 6 – 0 – ... Это значит, что числа к таким генераторам также надо подбирать с умом.

Поэтому экспериментальным путем выясним, какие же значения наименее благоприятны для линейного конгруэнтного метода. Чтобы сэкономить время и снизить вероятность ошибки я написала простую программу в Паскале, которая составит последовательность, используя линейный конгруэнтный метод и введенные нами данные, а для наглядности создала графики с такой же формулой в Excel.

Текст программы:

```
var
a, c, m: integer;
rand: array[1..100] of integer;
begin
write ('Множитель: ');
readln (a);
write ('Приращение: ');
readln (c);
write ('Модуль: ');
readln (m);
write ('Начальное значение: ');
readln (rand[1]);
for var i:=2 to 100 do
rand [i]:=(a*rand[i-1]+c)mod m;
writeln ('Последовательность, полученная линейным конгруэнтным методом с использованием введенных данных:');
for var i:=1 to 100 do
writeln (i,') ',rand[i],';');
end.
```

Для проверки работоспособности и оценки характеристик датчтикам случайных чисел были построены графики,:

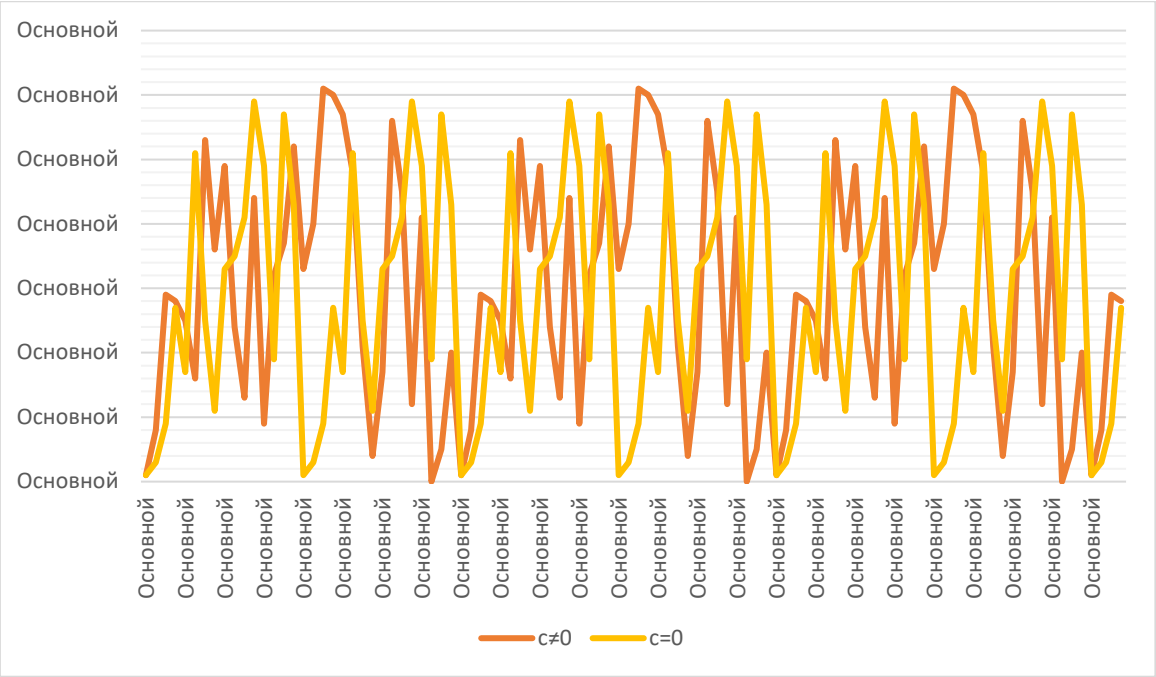

Рис.1. a=3, c=0, m=64, X_0=1

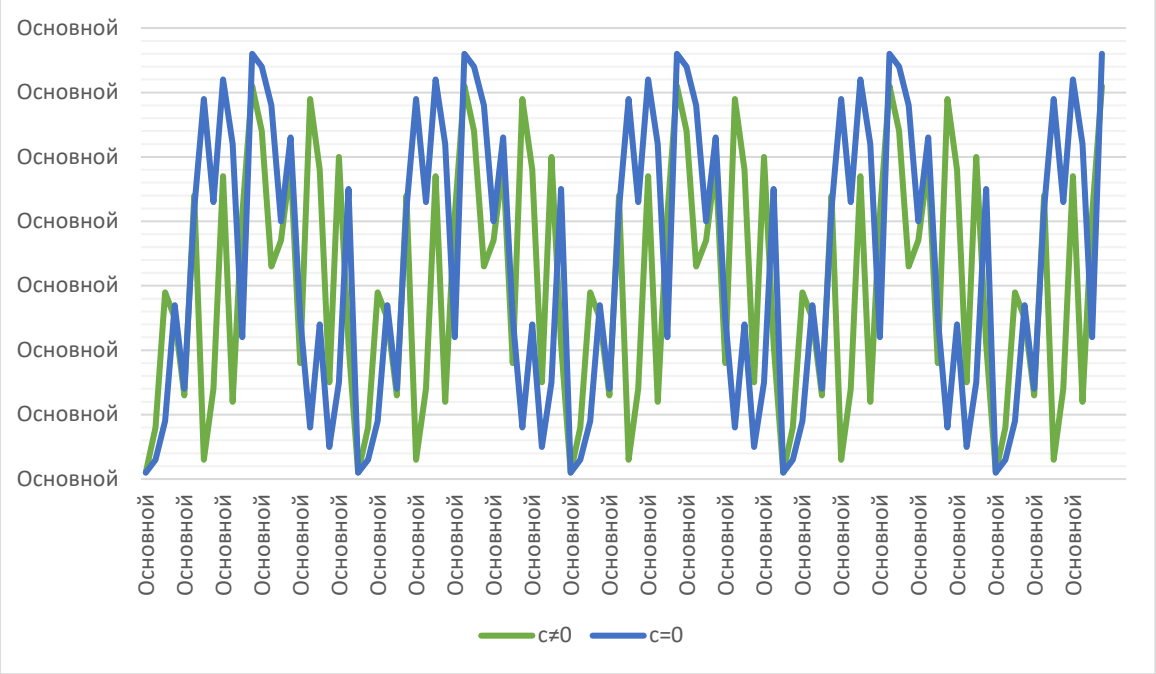

Рис.2. a=3, c=5, m=67, X_0=1

Эксперименты, проведенный с использованием данной программы, позволили сделать несколько выводов об эффективности коэффициентов линейного конгруэнтного метода, в частности:

- m должно быть достаточно большим, так как не более чем через m элементов последовательность начнет повторяться;
- a не должно принимать значения, равные нулю илиm, иначе последовательность будет состоять из значений c;
- при $c = 0$ последовательность, как правило, имеет меньший период, исключения составляют последовательности, в которых m - простое число;
- нет каких-то определенных строгих критериев для удачных значений коэффициентовa, c и m.

Несмотря на то, что при удачно выбранных значениях данный метод выдает последовательность, сильно похожую на случайную, он не обладает криптографической стойкостью. Например, взломать последовательность можно, зная лишь несколько значений из нее.

Таким образом в настоящей статье рассмотрены принципы работы разных генераторов случайных чисел. Как оказалось, случайные числа получают либо с помощью заранее составленных таблиц, либо с помощью энергозатратных и непрактичных физических явлений, либо с помощью математических алгоритмов. В последнем случае генератор случайных чисел плох тем, что его последовательность закономерна, циклична и обратима, но он также имеет существенные преимущества, такие как скорость и количество, поэтому он кажется наиболее удобным и надежным.

Работа генераторов псевдослучайных чисел рассмотрена на примере линейного конгруэнтного метода. В любом случае этот метод – своего рода, прорыв. Благодаря ему были созданы уже более сложные алгоритмы, которые как раз и обеспечивают нашу защиту.

ЛИТЕРАТУРА

1. Кнут Д. Искусство программирования, т. 2. Получисленные алгоритмы, 3-е изд.: Пер. с англ. – М.: ООО «И. Д. Вильямс», 2018. – 832 с.: ил. – Парл. тип. англ.

2. Слеповичев И.И. «Генераторы псевдослучайных чисел», учебное пособие, май 21, 2017.

3. Шнайер Б. Секреты и ложь. Безопасность данных в цифровом мире. — СПб.: Питер, 2003. — 368 с.

Электронные ресурсы:

1. A Brief History of Random Numbers by Carl Tashian. – Режим доступа: https://www.freecodecamp.org/news/a-brief-history-of-random-numbers-9498737f5b6c/.

2. From The Rand Corporation, A Million Random Digits with 100,000 Normal Deviates (New York: The Free Press, 1955).–Режим доступа:https://teorica.fis.ucm.es/ft8/tablern2.pdf.

3. I Programmer, ERNIE - A Random Number Generator, by Harry Fairhead. – Режим доступа: https://www.i-programmer.info/history/machines/6317-ernie-a-random-number-generator.html.

4. Semantic Scholar, FIRST 40 NUMBERS FROM THE TIPPETT'S TABLE. –Режим доступа: https://www.semanticscholar.org/paper/Parity-check-matrix-construction-of-LDPC-codes-in-Prasartkaew-Kovintavewat/043f813326fd45fd06989409c8a16d7ad4dc6595/figure/3.

5. StackExchange, Cryptography, Crack linear congruential generator knowing every other word in sequence. – Режим доступа: https://crypto.stackexchange.com/questions/24767/crack-linear-congruential-generator-knowing-every-other-word-in-sequence.

6. Wikipedia, the free encyclopedia, Random number table.Режим доступа: https://en.wikipedia.org/wiki/Random_number_table.