1,3,6,5). Сразу легко проверить, что они существуют и их длина 20 единиц. Внимательно изучив граф, находим ещё пути из 1 в 5 и из 1 в 4, убеждаемся, что они больше.

Заключение. В статье предложен алгоритм с умеренным перебором вариантов, который для графов с ограничениями на количество связей между вершинами и учёте особенностей моделируемого объекта на их характер обеспечивает следующие преимущества:

1) для всех заданных пар вершин графа называются кратчайшие пути и перечень вершин, через которые они проходят;
2) не требуется особых отметок для вершин, которые уже рассмотрены;
3) возможно указание всех цепочек с одинаковым расстоянием от заданной вершины;
4) не требуется в виде какого-то числа называть машинную бесконечность для условий конкретного графа;
5) исключение цепочек с путём более кратчайшего деается всего лишь по равенству последних номеров их вершин при совмещении с операцией сравнения их путей;
6) естественным путём заканчивается процесс построения цепочек. Следует отметить, что иногда из-за равнозначных ответов их количество может превышать число вершин заданного графа.

СПИСОК ЦИТИРОВАННЫХ ИСТОЧНИКОВ

1. Зыков, А. А. Основы теории графов. – М. : Наука,1987.
2. Алгоритм Дейкстры / Википедия. – Дата доступа : 10.11.2018.

Материал поступил в редакцию 08.02.2019

MATYUSHKOVA G.L., MATYUSHKOV A. L., VOYTSEKHOVICH O. Yu. Algorithm of search of the shortest way from the set count's top to the others

The article deals with the wave algorithm for finding the shortest path between the given vertices of the graph with a list of vertices on which it passes.

УДК 519.673

Петренко Т. Ю., Петренко И. А., Дереченник С. С.

О МЕТОДЕ ФОРМИРОВАНИЯ ДЛИННОПЕРИОДНОЙ КОНГРУЭНТНОЙ ПСЕВДОСЛУЧАЙНОЙ ПОСЛЕДОВАТЕЛЬНОСТИ

Введение. Псевдослучайные последовательности (ПСП) чисел имеют широкое применение в разных приложениях – от метода Монте Карло и имитационного моделирования до криптографии. Существует достаточное количество требований качества, предъявляемых к генераторам псевдослучайных последовательностей. Одним из этих требований является достаточно длинный период, гарантирующий отсутствие зацикливания последовательности в рамках решения поставленной проблемы. Задача формирования длиннопериодных ПСП является актуальной, поскольку от длины периода используемых ПСП напрямую зависит качество результатов. Известно, что ни одна псевдослучайная последовательность не может быть абсолютно случайной, поэтому последовательности должны проходить статистическую проверку. В данной статье впервые представлен алгоритм и анализ модифицированного конгруэнтного метода формирования псевдослучайных чисел, который способен формировать длиннопериодные псевдослучайные последовательности.

Линейный конгруэнтный метод. В данном давно изученном методе выбираются, как описывал Д. Кнут [1], четыре «волшебных числа»:

m, модуль; $0 < m$;

a, множитель; $0 \le a < m$;

c, приращение; $0 \le c < m$;

X_0, начальное значение; $0 \le X < m$.

Получают линейную конгруэнтную последовательность по формуле 1.

$$X_{n+1} = (aX_n + c) \bmod m,\ n \ge 0. \qquad (1)$$

Нам интересен случай с принятым названием «мультипликативный» линейный конгруэнтный метод, т. е. при $c = 0$. Обычно модуль выбирается равным длине машинного слова $w = 2^e$. Д. Кнут описал, что суммирование по модулю w (кроме случаев, когда машина использует процедуру единичного дополнения) и умножение по модулю w простое, т. к. затрагиваются только младшие разряды произведения. Существует случай, при $c = 0$, когда $m = w + 1$, т. е. значение aX_n может находиться между 0 и w включительно. Объясняется это тем, что переполнение происходит только тогда, когда результат равен w, и удобно его отбросить, когда оно появляется в конгруэнтной последовательности по модулю $m = w + 1$. В таком случае младшие разряды X_n ведут себя так же случайно, как и старшие. При этом важно учесть свойства коэффициентов, которые для получения максимального периода следующие:

1) числа c и m взаимно простые;
2) b кратно p для каждого простого p, являющегося делителем m;
3) если m кратно 4, то и b должно быть кратно 4.

По условиям, описанным еще в 1990 году [2], для получения случайных характеристик множитель a при делении на 8 должен давать остаток 5, т. е. $a \equiv 5 \bmod 8$. Начальное число X_0, задаваемое оператором, для получения хороших статистических характеристик, должно давать при делении на 4 остаток 1, т. е. $X_0 \equiv 1 \bmod 4$. Для того чтобы не заниматься подбором этих двух чисел с условиями, воспользуемся формулами $a = 5^{2p+1}$, а начальное число заменим на $RX_0 = 4X_0 + 1$ (тогда любое число, которое мы запишем в X_0, запишет в RX_0 число, которое при делении на 4 дает остаток 1). Рассмотрим модуль m, для заданного значения множителя $m = 2^N$, где N – это разрядность двоичного числа, т.е. количество числовых разрядов, необходимых для записи этого числа в двоичной системе счисления. Для определения степени $2p+1$ должно выполняться условие $5^{2p+1} < 2^N$, т. е. $a < m$. Прологарифмировав левую и правую части неравенства, найдем наибольшее значение $5^{2p+1} = fix[N \ln 2 / \ln 5]$, где операция $fix[5.65] = 5$ оставляет только целую часть числа. Период

Петренко Татьяна Юрьевна, *аспирант кафедры электронных вычислительных машин и систем Брестского государственного технического университета.*

Дереченник Станислав Станиславович*, к. т. н., доцент, зав. кафедрой электронных вычислительных машин и систем Брестского государственного технического университета.*

Беларусь, БрГТУ, 224017, г. Брест, ул. Московская, 267.

Петренко Илья Александрович, *студент кафедры прикладной информатики и математики Варшавской главной школы сельского хозяйства.*

Poland, 02-787 Warszawa, ul. Nowoursynowska, 166.

последовательности при таком способе формирования равен $m/4$. Далее, поскольку данная степень с использованием числа p подразумевает, что число должно получиться нечетным (т. к. $2p$ всегда дает четное число, а плюс единица делает его нечетным), то должна выполняться проверка на четность. В нашем случае, в программной среде Matlab, воспользуемся командой `bitand(RDLO,1)==0`. После проверки, если условие выполняется, необходимо отнять единицу. Прибавление единицы невозможно, поскольку в таком случае вышеописанное неравенство, $a < m$, не будет выполняться.

Ранее [3] был описан алгоритм формирования псевдослучайной последовательности с увеличенным периодом, для реализации которого использовался линейный конгруэнтный метод формирования псевдослучайных последовательностей чисел (ПСЧ). В данном способе формирования последовательности операция взятия модулярного остатка заменена на операцию последовательного вычитания. Данный способ формирования последовательности использует одно начальное положение генератора, т. е. только одно состояние. Реализация данного алгоритма в программной среде Matlab, для дальнейшего упрощения описания длиннопериодного генератора, приведена на рисунке 1.

```
%  N      -   Разрядность двоичного числа
%  DXO    -   Начальное случайное число
%  R      -   Количество ПСЧ
%  RDNO   -   нормированное ПСЧ
N=53;
DXO=33;
RDMO=2^N;
RXO=DXO*4+1;
RDLO=fix(N*log(2)/log(5));
RDIO=RDLO-2*fix(RDLO/2);
if (bitand(RDLO,1)==0)
    RDLO=RDLO-1
end
RDNO=[];
R=70000;
for i=1:R
                for k=0:RDLO-1
                    y = RXO;
                    for c=0:3
                        y = y + RXO;
                        if (y > RDMO)
                            y = y - RDMO;
                        end
                    end
                    RXO = fix((y-1)/4)*4+1;
                end
                RDNO(i) = RXO/RDMO;
end
```

Рисунок 1 – Алгоритм формирования ПСЧ с периодом 2^{51}

Для проверки условия переполнения в программе, в отличие от способа генерации чисел, описанного Д. Кнутом в [1], выполняется строгое сравнение со значением максимального состояния генератора. Это позволяет избежать выставления состояния генератора в нулевое значение, которое приводит к некорректной работе генератора.

Для перевода целых чисел в двоичную систему полученные числа необходимо поделить на максимальное значение, т. е. на модуль 2^N, умножить на два и отбросить дробную часть числа с помощью той же команды *fix*, используемой для проверки на четность степени мультипликатора. Последовательность бит выводится для того, чтобы иметь возможность в дальнейшем протестировать полученную последовательность ПСЧ и убедиться в том, что сгенерированные числа статистически независимы и некоррелированные.

Варьируя разрядность двоичного числа, можно получать максимальный период 62 бит. 64 бит получить невозможно, т. к. при проверке возможности переполнения мы можем не увидеть, что данное переполнение произошло. Далее опишем алгоритм, с помощью которого можно получать длиннопериодные последовательности. На основе генератора [3] предлагается метод, увеличивающий период за счет изменения количества внутренних состояний генератора.

Длиннопериодный конгруэнтный генератор. Реализовав замену модулярного деления на последовательное вычитание, появилась возможность формирования последовательности, используя не одно начальное положение генератора, а несколько. Так, задавая матрицей несколько начальных различных случайных чисел, можно последовательно формировать ПСЧ, используя поочередно числа с разных начальных состояний генератора. Степень периода при таком способе формирования увеличивается в количество используемых дополнительных случайных чисел (дополнительных начальных состояний), или, если говорить в единице ‘бит’, то, например, при одном начальном состоянии период 51 бит, но при двух начальных состояниях период станет 104 бит и т. д.

Для проверки условия переполнения, описанного в коде предыдущего раздела, $y > RDMO$, была создана отдельная функция. Отдельная функция позволит при переполнении прибавить это переполнение к следующему состоянию. Данная функция вызывается рекурсивно. Если случилось переполнение в текущем элементе, функция прибавит единицу к следующему элементу и вызовет эту же функцию на следующем элементе. Конечно, вероятность выпадения двойного переполнения критически мала, но выпадение такой случайности нарушило бы внутреннее состояние генератора, и дальнейшая сгенерированная последовательность не гарантировала бы получение полного периода.

В компьютере хранятся значения, кратные двум. Для хранения 53 бит достаточно 8 байт (позволяют хранить 64 бит). Поэтому в битовом представлении чисел необходимо избавиться от последних 11 бит (в англоязычной литературе такие биты называются most significant bits). Для мультипликативного конгруэнтного генератора, как ранее было описано, можно достичь период, который равен 2^{N-2}. Поэтому при использовании нескольких состояний во все состояния, кроме первого, переносятся единицы.

На рисунках 2 и 3 приведены схемы, описывающие способы формирования ПСЧ генераторов.

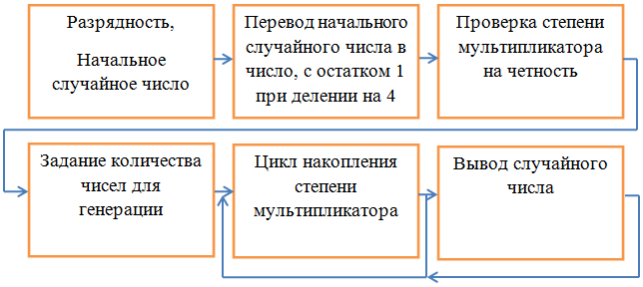

Рисунок 2 – Схема работы генератора внутреннего состояния 53 бит

Рисунок 3 – Схема работы длиннопериодного генератора с шестью внутренними состояниями

Основываясь на генераторе с одним начальным состоянием и периодом 53 бит, степень периода длиннопериодного генератора увеличивалась прямо пропорционально в число раз, равное количеству добавляемых начальных случайных состояний. Данный способ формирования последовательности способен увеличивать период, фактически, в неограниченное число раз.

Некоторые статистические тесты для длиннопериодного генератора. Основополагающие правила для статистических свойств периодических псевдослучайных последовательностей были представлены американским ученым Соломоном Голомбом. Он опубликовал в 1967 году [4], в постулатах, описание структуры псевдослучайных последовательностей, использование которой для создания коротких (по современным понятиям) псевдослучайных последовательностей приводило к удовлетворительным результатам. Голомб положил в основу количественную симметрию – количество «1» и «0» в каждом периоде должно отличаться не более чем на единицу. На практике постулаты Голомба служат первичной оценкой, а не инструкцией построения псевдослучайных последовательностей.

Для эксперимента используем 6 начальных различных состояний, модулярное значение последовательности $m = 2^{318}$. Проверим подпоследовательность 2^{15} чисел на соответствие длиннопериодной последовательности ПСЧ одному из постулатов Голомба, величину функции автокорреляции, которая должна принимать два значения $\exists K \in m$ по формуле 2.

$$C(t) = \frac{1}{m}\sum_{i=0}^{m-1}(2s_i - 1)(2s_{i+t} - 1) = \begin{cases} 1, & \text{если} \quad t = 0 \\ \dfrac{K}{m}, & \text{если} \quad 1 \le t \le m-1 \end{cases} \qquad (2)$$

Для проверки последовательности на соответствие третьему постулату Голомба предположим, что у нас есть две копии одной и той же последовательности длины 2^{15} чисел, сдвинутые относительно друг друга на значение $t = 33$. Тогда мы можем посчитать количество согласованностей между этими двумя последовательностями и количество несогласованностей. Согласованность означает, что для последовательности и сдвинутой ее копии биты с одинаковым порядковым номеров совпадают, а несогласованности – не совпадают. При суммировании по модулю два в ответе будет «0» там, где есть согласованность, и «1» там, где она отсутствует. Коэффициент автокорреляции k для каждого t определяется по формуле 3.

$$k = \frac{A + D}{m}, \qquad (3)$$

где A и D – это количество согласованностей и несогласованностей соответственно.

Полученный график по вышеописанным правилам приведен на рисунке 4.

По графику видно, что величина автокорреляции принимает различные значения по мере того, как t проходит все допустимые значения. Тогда для любой последовательности, воспользовавшись формулой 2, найдем значение функции автокорреляции на интервале $1 \le t \le m-1$:

```
n=fread(f);
M=2^20;
s=double(n(1:M))-48;
v=(1:33);
for d=1:33
len=2^15;
 for i =1:len
 fun(i)=(1/len)*sum((2*s(i)-
1)*(2*s(i+d)-1));
 end
end
```

Полученное модульное значение $|C(t)| = 3.051757812500000e-05$, близко к нулю для всех $1 \le t \le m-1$. Можно сделать вывод о том, что часть длины периода проходит проверку на соответствие третьему постулату Голомба.

Заключение. Проведена оценка периода генерации длиннопериодного генератора псевдослучайных чисел. Основываясь на генераторе с одним начальным состоянием 53 бит, степень периода длиннопериодного генератора увеличивается прямо пропорционально в число раз, равное количеству добавляемых начальных случайных состояний. Данный способ формирования последовательности потенциально позволяет формировать последовательности, фактически с неограниченным периодом. Период практически ограничивается лишь объемом оперативной памяти. Для первичной оценки правильности работы генератора и проверки его статистических характеристик была построена функция зависимости коэффициента корреляции от величины сдвига, модульное значение функции автокорреляции составило 3.051757812500000e-05. Схожесть сдвинутых копий последовательности не выявлена, коэффициент автокорреляции варьируется в пределах $[-0.01; 0.01]$. Реализованный алгоритм формирования длиннопериодных псевдослучайных последовательностей и полученные начальные статистические характеристики дают основание на дальнейшего изучения генератора, более подробного разбора статистических свойств последовательностей.

СПИСОК ЦИТИРОВАНЫХ ИСТОЧНИКОВ

1. Knuth, D. E. The Art of Computer Programming. – Vol. 2. Seminumerical Algorithms, 3nd ed. – Massachusetts : Addison Wesley, Reading., 1998. – P. 29–45.
2. Fishman, G. S. Multiplicative congruential random number generators with modulus 2^{β}: An exhaustive analysis for $\beta = 32$ and a partial analysis for $\beta = 48$ // Mathematics of Computation. – 1990. – P. 331–344.
3. Хазан, В. Л. Математические модели дискретных каналов связи декаметрового диапазона волн: учебное пособие / В. Л. Хазан. – Омск, ОмГТУ, 1998. – 107 с.
4. Филатов, О. В. Вывод формул для постулатов Голомба. Способ создания псевдослучайной последовательности из частот Мизеса. Основы «Комбинаторики длинных последовательностей» / О. В. Филатов // Проблемы современной науки и образования. – 2016. – № 17.

Материал поступил в редакцию 04.12.2018

PIATRENKA T. J., PIATRENKA I. A., DERECHENNIK S. S. On the method of generating long-period pseudorandom sequences

This paper is about new modification of linear congruential method, witch enables generation of long-period pseudorandom sequences. This algorithm is based on 53-bit internal state size generator with a number of internal states. The main analysis, method description and algorithms are deduced.