Моделирование случайных событий и переменных на ЭВМ

Имитация случайных переменных (СП) и событий на ЭВМ при моделировании сложных систем необходима по крайней мере по трем причинам: 1) сложные системы практически всегда стохастические, включающие в себя случайные события, сигналы, взаимодействия; 2) упрощение сложной системы при ее моделировании требует замены неучтенных в модели элементов и подсистем эквивалентными воздействиями, которые носят случайный характер; 3) значения случайных величин (СВ) необходимы при решении детерминированных задач, если алгоритм решения предполагает использование случайных шагов, например, в методах случайного поиска для задач оптимизации или методе статистических испытаний (Монте-Карло).

Имитация случайных событий и значений СП на ЭВМ реализуется в два этапа. На первом этапе получают значения равномерно распределенной в диапазоне (0;1) непрерывной СВ (РРНСВ). На втором этапе эти значения преобразуют в значения СП или случайные события с необходимыми вероятностными свойствами.

Для осуществления первого этапа известны три способа: 1) использование природных источников случайных значений; 2) отбор значений из заранее подготовленных таблиц случайных чисел; 3) применение специальных итерационных алгоритмов, последовательно генерирующих числовые значения, которые обладают свойствами РРНСВ.

В первом способе регистрируют тепловой шум электронных или полупроводниковых приборов, который затем преобразуют в числовой код. Этот способ непригоден для моделирования, поскольку дает поток значений с нестабильными свойствами и не допускает воспроизведения. В настоящее время этот способ не применяется.

Второй способ хорош тем, что табличные значения могут быть заранее оттестированы и быстро извлекаться из памяти ЭВМ. Однако моделирование требует огромного количества (миллионов) значений РРНСВ, хранение которых требует больших объемов памяти. Поэтому данный способ также практически не применяется.

Третий способ – специальные программно реализованные алгоритмы, получившие название **генераторов** (датчиков) **псевдослучайных чисел** (ГПСЧ), - является основным при моделировании систем и рассматривается ниже.

Проблема разработки ГПСЧ в том, что ЭВМ – детерминированный автомат, работающий по заданным алгоритмам, а ГПСЧ должны генерировать числовые значения, которые бы можно было рассматривать как реализации РРНСВ. Поэтому алгоритмы, используемые для ГПСЧ, должны быть такими, чтобы закономерность, детерминированность алгоритма не проявлялась в результате их работы –в получаемых числах, которые в отличие от значений «настоящих» СВ получили назание **псевдослучайных чисел** (ПСЧ).

Кроме этой основной особенности, алгоритмы ГПСЧ должны отвечать ряду требований:

1. Генерируемые числовые значения должны обладать свойствами реализаций случайной величины, в том числе:

1.1. Соответствовать заданному распределению – равномерному в диапазоне (0;1) – Un(0;1)
1.2. Образовывать последовательность независимых значений
1.3. Быть однородной, т.е. сохранять свои свойства на протяжении всей последовательности
1.4. Не содержать повторяющиеся значения или отрезки значений (отсутствие циклов в последовательности)

2. ГПСЧ должен быть быстродействующим, чтобы затраты компьютерного времени на генерацию занимали минимум времени, и использовать минимальные объемы памяти
3. ГПСЧ должен допускать возможность воспроизведения последовательности ПСЧ.ГПСЧ должен генерировать последовательности с указанными свойствами длиной в миллионы и десятки миллионов чисел.

Разработке алгоритмов ГПСЧ посвящено огромное количество теоретических и практических научных исследований. Предложено и проверено большое число алгоритмов. Установлено, что качество ГПСЧ зависит от числа разрядов представления двоичных кодов в памяти и процессоре ЭВМ. Конечное число разрядов физически ограничивает длину последовательности ПСЧ без повторения значений. Существует противоречие между качеством и длиной последовательности ПСЧ и быстродействием ГПСЧ. Однородность и независимость последовательности ПСЧ в сильной степени зависит не только от используемого алгоритма, но и выбранных значений параметров этого алгоритма – коэффициентов и т.н. стартовых значений ГПСЧ.

Конгруэнтный мультипликативный алгоритм ГПСЧ

Одним из наиболее популярных алгоритмов является т.н. конгруэнтный мультипликативный алгоритм.
(*Примечание*. Пусть числа a, b, m, k – целые положительные. Два целых числа a и b называют **конгруэнтными** (сравнимыми) по модулю m тогда и только тогда, когда существует такое число k, что $a - b = km$. Это означает, что числа a и b конгруэнтны, если их разность делится на m и остатки от деления одинаковы. Пример: $\mathbf{mod}_{10}(1988) = 8; \mathbf{mod}_{10}(5008) = 8.$) .
Алгоритм является примером применения рекуррентного соотношения 1-го порядка:

$$x_{i+1} = \varphi(x_i) = \mathrm{mod}_m(Ax_i + b)$$

Здесь x_i – i-е ПСЧ генерируемой последовательности, A,b – коэффициенты (параметры) преобразования, $\mathbf{mod}_m()$- операция преобразования аргумента по модулю m. У мультипликативных алгоритмов параметр $b = 0$.

На рис.1 приведен график этой функции при A=10, b=0, m=1. Видно, что малым изменениям аргумента отвечают большие изменения функции. Эта "неустойчивость" функционального преобразования и создает эффект случайности последовательности генерируемых значений. В реально используемых мультипликативных ГПСЧ в качестве значения параметра A выбирают большое (допустимое для выбранной разрядности представления чисел в регистрах ЭВМ) простое целое число, параметр b подбирают так, чтобы обеспечить независимость значений. Для большей скорости генерации арифметические операции выполняют в целых числах или непосред-

ственно с кодами, применяя операции двоичной арифметики. Достоинством алгоритма, обусловившим его популярность, являются высокое быстродействие и приемлемое качество ПСЧ при правильном подборе параметров и стартового числа x_0.

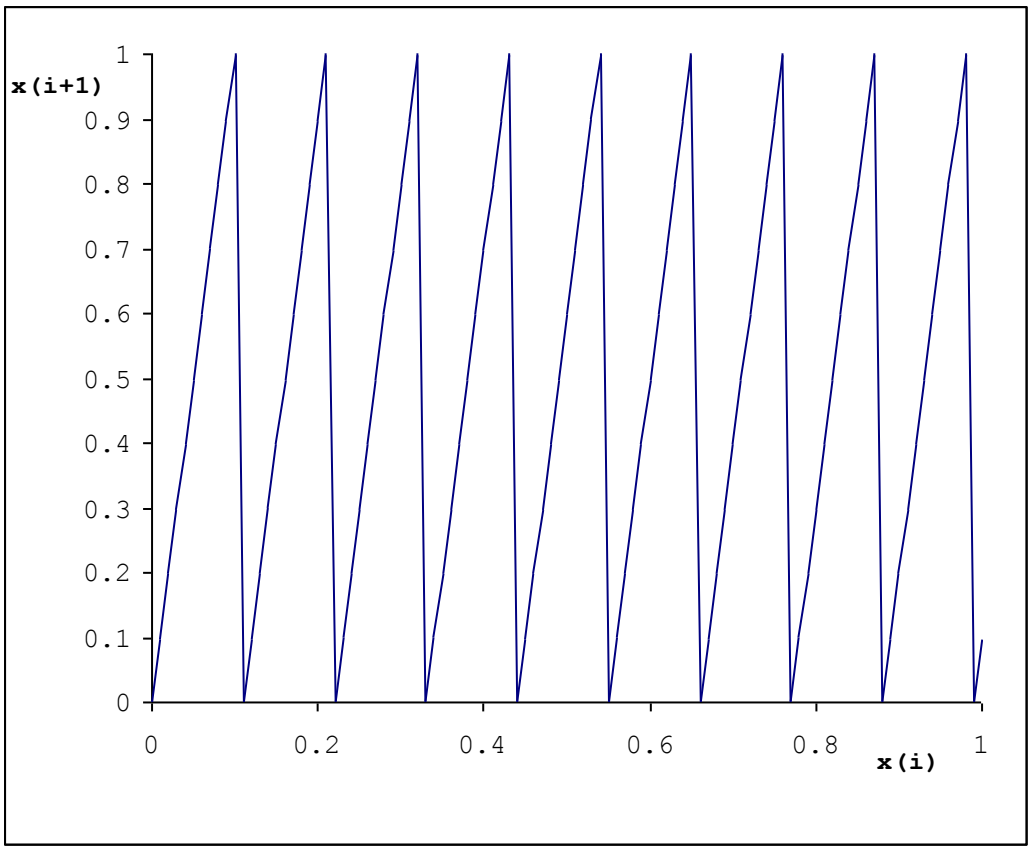

Рис. 1

В большинстве современных алгоритмических языков программирования имеются встроенные функции, реализующие мультипликативный алгоритм. Стартовые числа либо задаются пользователем, либо выбираются автоматически. Например, в качестве стартового значения компьютер выбирает значение системного таймера на момент запуска функции. Для получения последовательности значений РРНСВ выполняется программным способом циклическое обращение к функции.

Проверка статистических свойств последовательности ПСЧ

Проверка закона распределения

Последовательность ПСЧ должна соответствовать равномерному закону распределения Un(0;1). Для проверки могут быть использованы статистические критерии А.Н. Колмогорова и хи – квадрат Пирсона. Поскольку проверяется, как правило, большое количество чисел (тысячи и более), то пригодны асимптотические методы. Кроме того, целесообразно разбивать последовательность на подпоследовательности и выполнять проверку для каждой из них отдельно. Тем самым прверяется однородность распределения на всех участках последовательности.

Проверка гипотезы о постоянстве среднего значения

Отсутствие дрейфа среднего значения в последовательности ПСЧ можно выполнить, применяя критерий серий. Теоретическое значение математического ожидания известно и равно 0,5. Поэтому последовательность заменяется чередованием (+) и (-) для членов последовательности соответственно больших и меньших 0,5. Затем подсчитывается число серий одного знака. Последовательность не противоречит гипотезе, если число серий *v(n)* и наибольшая длина серии *r(n)*, где *n* – длина последовательности, удовлетворяют неравенствам (уровень значимости 0.05):

$$v(n) > [\frac{1}{2}(n+2-1.96\sqrt{n-1})],$$
$$r(n) < [1.43\ \ln(n+1)]$$

Здесь прямоугольные скобки [*z*] означают операцию взятия целой части числа *z*.

Для обнаружения не только монотонного смещения среднего, но и более общего характера, например, периодического, можно использовать критерий "восходящих" и "нисходящих" серий. Исходная последовательность заменяется чередованием (+) и (-), если последовательные разности *x(i+1)-x(i)*>0 или <0 соответственно. Последовательность не противоречит гипотезе, если одновременно выполняются неравенства для числа серий и максимальной длины серии (уровень значимости 0.05):

$$v(n) > [\frac{1}{3}(2n-1) - 1.96\sqrt{\frac{16n-29}{90}}],$$
$$r(n) < 7.$$

Если последовательность ПСЧ соответствует нормальному закону распределения, то более мощным, чем критерий серий, является критерий Аббе. Для проверки гипотезы с помощью данного критерия подсчитывают величину:

$$\tau(n) = \frac{q^2(n)}{s^2(n)}, \quad \text{где} \quad q^2(n) = \frac{1}{2(n-1)}\sum_{i=1}^{n-1}(x(i+1)-x(i))^2,$$
$$s^2(n) = \frac{1}{n-1}\sum_{i=1}^{n}(x(i)-\bar{x})^2, \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n}x(i)$$

Если $\tau(n) < \tau_c(n)$, то гипотеза отвергается. Критические значения определяются по формуле:

$$\tau_c(n) = 1 + \frac{u_\alpha}{\sqrt{n+0{,}5(1+u_\alpha^2)}}$$

Здесь u_α - квантиль для вероятности α стандартного нормального распределения *N(0;1)*. Например, при α=0.05 величина u_α = 1,96.

Проверка гипотезы о случайности значений ППСЧ

Для проверки случайности (стохастичности) ППСЧ используют тот же критерий серий, но для бинаризации (присвоения знаков + и -) в качестве порогового значения используют не только математическое ожидание 0.5, но и другие значения 0<*p*<1. Если при различных *p* гипотеза об отсут-

ствии смещения не отвергается, то это свидетельствует о хорошем "перемешивании" значений, т.е. псевдослучайности ППСЧ.

Другой способ, требующий анализа двоичных кодов значений ППСЧ, состоит в проверке случайности серий 0 и 1 в двоичных кодах ПСЧ. Если эти серии случайны как внутри одного числа, так и в последовательности чисел, то ППСЧ можно считать случайной.

На рис.2 приведены точки с координатами ($x(i);x(i-1)$), $i=1,\ldots,n$. Значения $x(i)$ получены с помощью мультипликативного алгоритма с параметрами $A=737$, $b=0$, $m=1,1$. Стартовое значение – случайное. Длина последовательности N=2000 чисел. Видно, что точки распределены равномерно в пределах квадрата области генерации и отсутствует видимый закономерный «узор» в их расположении. Рис.3 получен при значениях параметров $A=4$, $b=0$, $m=1,1$.

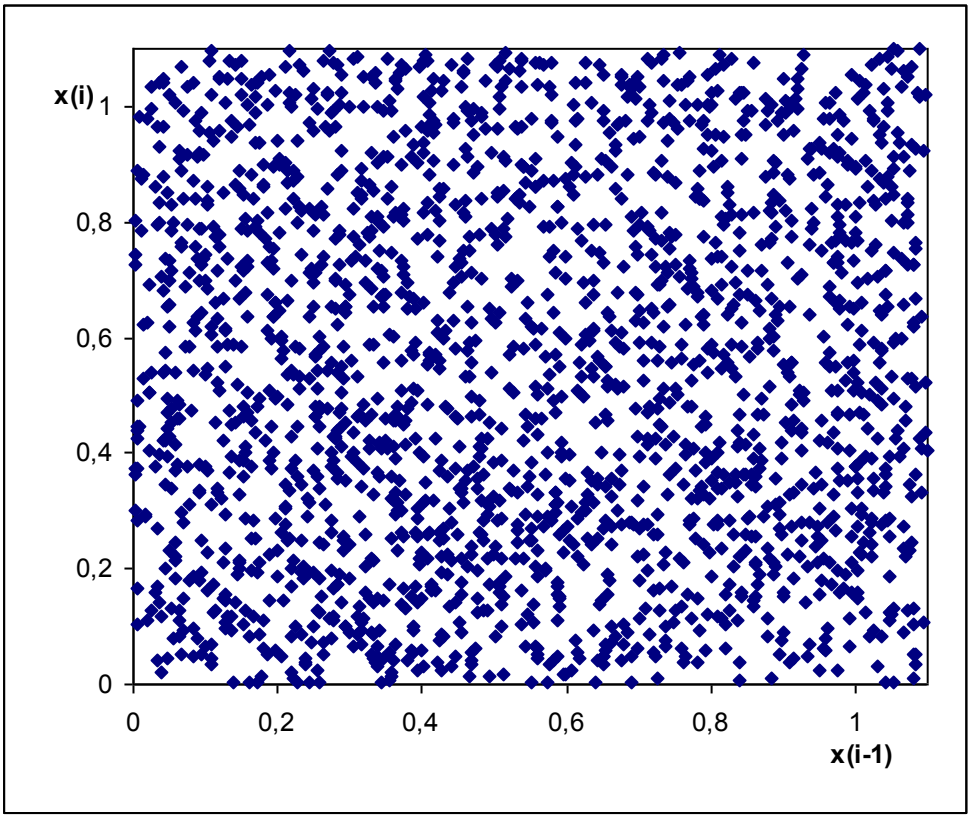

Рис.2

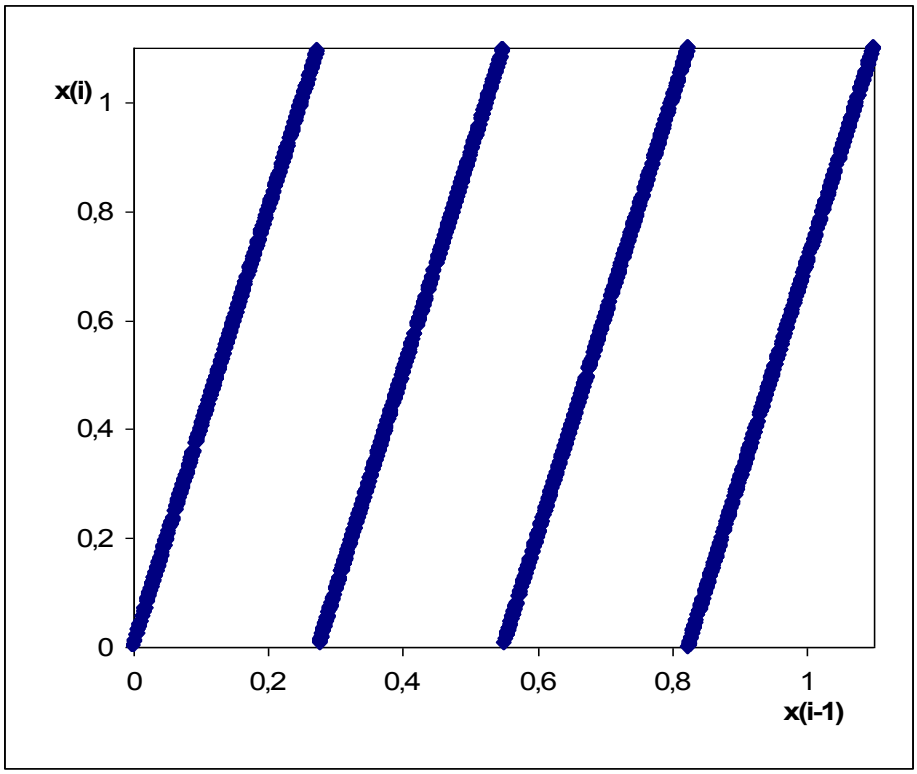

Рис.3

Рисунки наглядно доказывают важное значение правильного выбора параметров генератора. Во втором случае в ППСЧ возникает периодичность в чередовании генерируемых значений, что абсолютно недопустимо при моделировании. Расчеты и графика выполнены в среде Excel-2000.

Определение периода и отрезка апериодичности ППСЧ

Для хранения чисел в ЭВМ отводится конечное число двоичных разрядов и количество разных чисел, которые могут храниться в отведенной для числа области памяти, хотя и большое, но конечное. Поэтому при использовании длинных ППСЧ, что обычно бывает при моделировании, числа в ППСЧ могут повторяться. Моделировать систему на повторящихся значениях, очевидно, нет смысла, т.к. это не дает новой статистически значимой информации. Если алгоритм или параметры ГПСЧ подобраны неудачно, то повторение генерируемых значений может наступить быстро, и количество неповторяющихся значений будет малым. При использовании рекуррентных алгоритмов 1-го порядка достаточно первого появления повторяющегося числа и все последующие числа также будут повторяться. Структура получаемой ППСЧ изображена на рис.4.

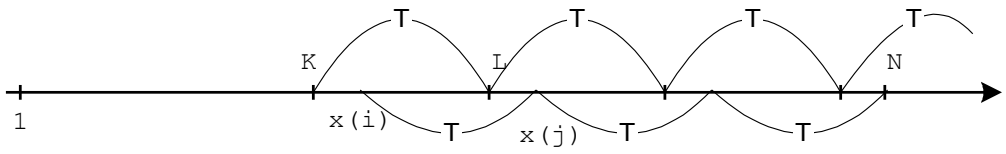

Рис. 4

Здесь N – длина ППСЧ (количество сгенерированных ПСЧ); T – период; K-номер первого числа, которое повторяется под номером L в ППСЧ. Количество неповторяющихся чисел L называют **отрезком апериодичности** ППСЧ. Он и является рабочей областью для использования в моделировании систем.

Для определения T и L используют следующий алгоритм. Все ППСЧ генерируют с одним и тем же стартовым числом x(0).

Сначала определяют T.

1. Генерируют ППСЧ x(1),...,x(N).
2. Заново генерируют ту же ППСЧ, сравнивают каждое из чисел с x(N) и находят два числа x(i)=x(N) и x(j)=x(N), причем i<j и j≤N. Разность T=j-i равна периоду ППСЧ.

Затем определяют отрезок апериодичности L.

1. Генерируют ППСЧ до x(T).
2. Одновременно генерируют две ППСЧ: одна стартует с x(0), другая с найденного x(T), - и сравнивают x(1) и x(T+1), x(2) и x(T+2), и т.д., пока не выполнится равенство x(K) и x(L=T+K). Получаем номер L – длину отрезка апериодичности.

Ориентировочная оценка $L=\sqrt{\pi N/2}$.

Генерация случайных событий

Пусть A – случайное событие, происходящее с известной вероятностью $p(A)$. Следовательно, при имитации случайного события на ЭВМ оно должно появляться случайным образом с этой вероятностью. Вместо события A рассмотрим событие B, состоящее в попадании ПСЧ на интервал $(0; p(A))$. Вероятность появления события B равна $p(A)$. Тогда процедура генерации появления события A следующая (**П.1**):

1. Генерация ПСЧ x.
2. Если $x \le p(A)$, то событие A произошло, иначе считается, что событие A не произошло.

Рассмотрим процедуру генерации полной группы событий $A(1),\ldots,A(k)$ при известных вероятностях $p(A(1)),\ldots, p(A(k))$. В полной группе события несовместны и $p(A(1))+\ldots+p(A(k))=1$. Цель процедуры – определить номер m происшедшего случайного события. Процедура имеет вид (**П.2**):

1. Генерация ПСЧ x.
2. Для $m=1,2,\ldots,k$ проверяется выполнение неравенства $p(A(1))+\ldots+p(A(m))\ge x$. То значение m, при котором оно впервые выполняется, есть искомое значение номера происшедшего случайного события $A(m)$.

Генерация значений дискретной случайной величины

Пусть для генерируемой случайной величины θ (СВ) известны принимаемые ею значения $t_1,\ldots, t_k$ и вероятности появления этих значений

$$p_1 = p(\theta = t_1),\ldots,p_k = p(\theta = t_k), \quad p_1+\ldots+p_k = 1 .$$

Тогда генерация значений СВ θ эквивалентна генерации случайных событий, образующих полную группу $A_1 \sim (\theta = t_1),\ldots,A_k \sim (\theta = t_k)$. Процедура гене-

рации имеет вид (**П.3**):

1. Генерация ПСЧ x.
2. $m = 1;\quad c = x$
3. $c = c - p_m$
4. Если $c \le 0$, то перейти к п.5, иначе: $m = m+1$; перейти к п.3
5. $\theta = t_m$.

Для многих типовых распределений (биномиального, пуассоновского, геометрического и др.) вероятности p_m связаны рекуррентной формулой $p_{m+1} = p_m \varphi(m)$, а $t_m = m = 0,1,2,\ldots$. В этом случае процедура упрощается (**П.4**):

1. Генерация ПСЧ x
2. $m = 0;\quad c = x;\quad p = p_0$
3. $c = c - p$
4. Если $c < 0$, то перейти к п.5, иначе: $p = p\ \varphi(m)$; $m = m+1$; перейти к п.3
5. $\theta = m$.

Для биномиального $Bi(n,p)$, геометрического $Ge(p)$, пуассоновского $Po(a)$ распределений функция $\varphi(m)$ равна соответственно:

$$\varphi(m) = \frac{(n-m)p}{(m+1)(1-p)},\ m = 0,1,\ldots,n;\quad \varphi(m) = p;\quad \varphi(m) = \frac{a}{m+1},\ m = 0,1,\ldots$$

Для некоторых случаев генерацию можно ускорить за счет специальных приемов. Так, равномерно распределенную дискретную СВ θ со значениями $t = 0,1,\ldots,n$ и вероятностями $p(\theta = t) = 1/(n+1)$ можно прогенерировать с помощью процедуры (**П.5**):

1. генерация ПСЧ x;
2. $\theta = \mathrm{Int}(x\ (n+1))$.

Здесь функция $\mathrm{Int}(z)$ вычисляет ближайшее к z меньшее целое число.

Можно показать, что значение СВ θ, распределенной по геометрическому закону, можно получить с помощью процедуры (**П.6**):

1. генерация ПСЧ x;
2. $\theta = \mathrm{Int}(Ln(x)/Ln(p))$.

Быстродействие последней процедуры невелико из-за необходимости вычислять логарифм ПСЧ, что само является итерационной процедурой.

Значение пуассоновской СВ может быть сгенерировано с помощью процедуры (**П.7**):

1. $m = 0;\quad s = 1;\quad c = \exp(-a)$
2. генерация ПСЧ x
3. $s = s\ x$
4. Если $s \le c$, то перейти к п.5, иначе: $m = m+1$; перейти к п.2
5. $\theta = m$.

Процедура эффективна при $a<9$.

Быстродействие процедур можно повысить, если осуществить переиндексацию значений t_m СВ θ так, чтобы меньшим номерам m отвечали большие вероятности появления. В этом случае среднее число циклов в процедуре при однократном обращении к ней, будет сокращаться.

Например, пусть случайная величина θ задана распределением:

θ	t_1=0	t_2=2	t_3=3	t_4=7	t_5=9	t_6=12
p_m	0.1	0.05	0.3	0.15	0.3	0.1

При генерации θ в «естественном» порядке с помощью процедуры (**П.3**) получим среднее число циклов
R_{sr} = 1*0.1+2*0.05+3*0.3+4*0.15+5*0.3+6*0.1 = 3,80. При изменении индексации значений случайной величины

θ	t_1=3	t_2=9	t_3=7	t_4=0	t_5=12	t_6=2
p_m	0.3	0.3	0.15	0.1	0.1	0.05

Получим среднее число циклов: R_{sr} = 1*0.3 + 2*0.3 + 3*0.15 + 4*0.1 + 5*0.1 + 6*0.05 = 2.55. Таким образом, средние затраты времени на генерацию одного значения случайной величины сокращаются в 1,5 раза.

Описанный прием использован в алгоритме генерации пуассоновской случайной величины x ~ Po(a), который представлен блок-схемой на рис.5.

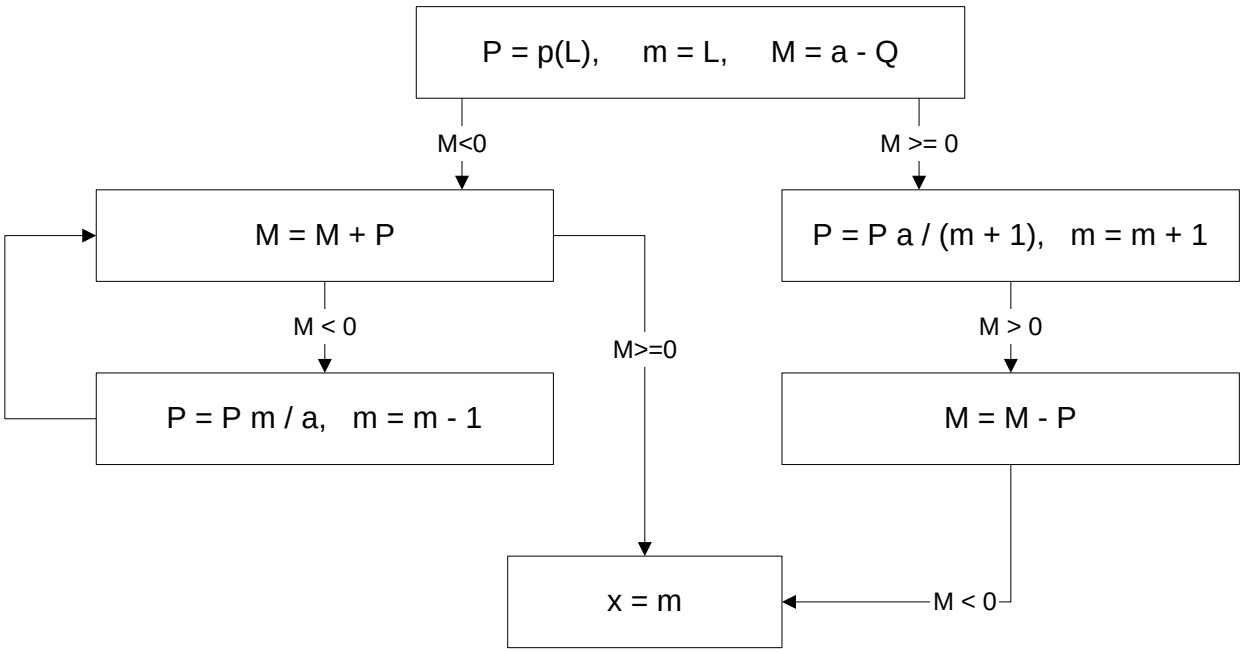

Рис.5

Здесь L – целая часть параметра a распределения: a > 1.

Генерация значений непрерывной случайной величины

Найдем функциональное преобразование $\eta=\varphi(\xi)$ непрерывной равномерно распределенной на интервале (0;1) СВ (РРНСВ) ξ ~ Un(0;1) в непрерывную СВ η с заданной функцией распределения (ФР) $F_\eta(y)$.

Используя правила правила преобразования ФР СВ при ее функциональном преобразовании, можно записать цепочку равенств:

$$p(\eta < y) = F_\eta(y) = p(\xi < x = \varphi^{-1}(y)) = F_\xi(x = \varphi^{-1}(y)) = x = \varphi^{-1}(y); \qquad \varphi(y) = F_\eta^{-1}(x),$$

из которых следует процедура генерации значений y СВ η с заданной ФР ("метод обратной функции") (**П.7**):

1. Генерация ПСЧ x
2. Вычисление $y = F_\eta^{-1}(x)$

Пример. Генерация значений СВ, распределенной по показательному закону η ~ Ex(a). ФР имеет вид $F_\eta(y) = 1-\exp(-ay)$, $y \geq 0$. Для получения обратной функции запишем уравнение $x = 1-\exp(-ay)$ и решим его относительно y: $y = -a^{-1}\,\mathrm{Ln}(1-x)$. Таким образом, подставляя в эту формулу значение x РРНСВ, сгенерированное с помощью ГПСЧ, вычисляем значение СВ η с заданным законом распределения. Учитывая, что СВ $1-x$ также РРНСВ в интервале (0;1), формулу можно упростить, исключив вычитание и сокращая тем самым время генерации (**П.8**): $y = -a^{-1}\,\mathrm{Ln}(x)$.

Метод обратной функции пригоден, если $F_\eta^{-1}(x)$ можно найти аналитически или достаточно точно аппроксимировать и на требуемые при этом арифметические операции не затрачивается слишком много компьютерного времени, что делает метод неприемлемым для моделирования. В противном случае применяют различные альтернативные или специальные методы. Ниже приведены некоторые алгоритмы генерации значений непрерывных СВ с наиболее часто используемыми в моделировании ФР. В них x - это значения РРНСВ, генерируемые ГПСЧ, y – значения непрерывной СВ с желаемой ФР.

Нормальное распределение N($m;\sigma$). Используется результат центральной предельной теоремы. Процедура генерации определяется формулой (**П.9**):

$$z = \Big(\sum_{j=1}^{k} x_j - a(k)\Big) \Big/ \sqrt{D(k)},\ \text{где}\quad a(k) = k/2;\quad D(k) = k/12;\quad y = \sigma\, z + m$$

Здесь $a(k)$ и $D(k)$ – математическое ожидание и дисперсия суммы k РРНСВ.

Число слагаемых k может быть не слишком большим. Обычно выбирают k=12. Тогда существенно снижается число операций, требуемых на получение одного значения нормальной СВ (**П.10**):

$$y = m + \Big(\sum_{j=1}^{12} x_j - 6\Big)\sigma$$

Процедура плохо отображает "хвосты" нормального распределения, т.к. значений y ограничены диапазоном (-6σ ; $+6\sigma$).

Следующий метод основан на факте, что декартовы координаты случайного вектора с равномерно распределенными направлениями в пространстве и гамма-распределением длины распределены по нормальному закону. Для двумерного пространства это приводит к экономичной процедуре генерации двух независимых значений нормальной СВ (**П.11**):

1. Генерация $x(1)$, $x(2)$.
2. $u = \sqrt{-2\ln x(1)};\quad y(1) = u\cos(2\pi x(2));\quad y(2) = u\sin(2\pi x(2))$

Показательное распределение Ex(a). Более экономичная по времени (сокращение времени более чем в 2 раза) процедура, чем **П.8**, требует генерации пяти x и дает сразу три независимых значения y при одном вычислении логарифма (**П.11**):

1. Генерация $x(1)$, $x(2)$, $x(3)$, $x(4)$, $x(5)$
2. Если $x(4) < x(5)$, то $z1=x(4)$, $z2=x(5)$, иначе $z1=x(5)$, $z2=x(4)$.
3. $u=\mathrm{Ln}(x(1)*x(2)*x(3))$
4. $y(1)=-z1*u$; $y(2)=(z1-z2)*u$; $y(3)=(z1-1)*u$