ИССЛЕДОВАНИЕ МОДИФИКАЦИЙ КОНГРУЭНТНОГО ГЕНЕРАТОРА ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ*

Н.В. Трачева
Институт вычислительной математики и математической геофизики
e-mail: `tnv@osmf.sscc.ru`

1. Конгруэнтный генератор псевдослучайных чисел

Моделирование любой случайной величины ξ с заданным распределением осуществляется путем преобразования одного или нескольких независимых значений случайного числа α, равномерно распределенного в интервале $(0, 1)$. Последовательность "выборочных" значений α обычно получается с помощью теоретико-числовых алгоритмов. В настоящее время, наиболее часто используемым и проверенным алгоритмом генерирования псевдослучайных чисел считается конгруэнтный генератор или метод вычетов [1]:

$$u_0 = 1,\ u_n \equiv u_{n-1}M (\mathrm{mod}\ 2^r),\ \alpha_n = u_n 2^{-r}. \tag{1.1}$$

Здесь r, как правило, число двоичных разрядов, используемых для представления числа в компьютере, M – множитель генератора, достаточно большое число, взаимно-простое с 2^r, α_n – псевдослучайное число. Для генератора (1.1) длина периода последовательности $\{\alpha_n\}$ равна 2^{r-2}.

В течение многих лет использовался конгруэнтный генератор (1.1) с параметрами $M = 5^{17}$ и $r = 40$ (см., например, [1]). Неоднократно результаты сравнивались с другими расчетными или экспериментальными данными. Согласие всегда получалось удовлетворительным. Однако, поскольку период генератора (1.1) при $r = 40$ ограничен $L = 2^{38} \approx 2,7 \cdot 10^{11}$ случайных чисел, а возможности вычислительных систем значительно возрасли, возникла необходимость в использовании генераторов с бóльшим периодом, например, генератора (1.1) с параметрами $M = 5^{100109} (\mathrm{mod}\ 2^{128})$ и $r = 128$, т.е. с длиной периода $L = 2^{126} \approx 10^{38}$ (см. [2, 3]).

2. 64-битная модификация генератора с параметрами $M = 5^{100109}$ и $r = 128$

Реализация алгоритма (1.1) с параметрами $M = 5^{100109} (\mathrm{mod}\ 2^{128})$ и $r = 128$ требует умножения чисел порядка 5^{100109}. Данная арифметика традиционно реализуется с использованием 32-битных целых типов. При этом 128-битные целые числа представляются в системах счисления по основанию 2^k, где k выбирается таким

*Работа выполнена при поддержке РФФИ (проекты 12-05-00169-а, 12-01-00034-а, 12-01-00727, 12-01-31328).

образом, чтобы избежать переполнения при умножении чисел. Например, при реализации генератора с параметрами $M = 5^{100109} (\mathrm{mod}\ 2^{128})$ и $r = 128$ используется основание 2^{13} [2, 4].

С появлением поддержки быстрой 64-битной арифметики в современных процессорах появилась возможность достичь ускорения вычислений за счет переформулирования старых алгоритмов в терминах 64-битных типов данных.

Приведем, например, разложение по основанию 2^{26} для 64-битной реализации множителя M:

$$M_{64} = 14919573 + 10735332 \cdot 2^{26} + 2380602 \cdot 2^{2\cdot 26} + 21375263 \cdot 2^{3\cdot 26} + \\ + 16382667 \cdot 2^{4\cdot 26}.$$

Аналогичное разложение можно привести для 32-битной реализации по основанию 2^{13} [2].

Такого рода представление сомножителей и переформулированный 64-битный алгоритм дают значительное ускорение при проведении вычислений на процессорах Intel Itanium 2 Новосибирского кластерного суперкомпьютера НКС-160 [5]. Длина "прыжка" bf-генератора, также как и в работе [2] была выбрана равной 10^{26}. В работе [5] приводились цифры начальных значений десяти псевдослучайных подпоследовательностей, полученные с помощью данного bf-генератора.

3. Векторизация генераторов псевдослучайных чисел

Вследствие архитектурных особенностей современных вычислительных процессоров, процедуры векторного типа часто оказываются более эффективны, чем их скалярные аналоги.

В таблице 1 приведены результаты тестирования двух алгоритмов: с использованием 32-битных и 64-битных целых типов. Здесь k – размерность генерируемого вектора псевдослучайных чисел. Компиляция проводилась с помощью Intel C Compiler 10.1 и GNU C Compiler 3.4.5, с оптимизационным ключом -O3. Вычисления проводились на процессоре Intel Itanium 2. Численный эксперимент показал значительное сокращение временных издержек при векторизации алгоритмов.

Т а б л и ц а 1. Время, затрачиваемое на генерирование $N = 10^8$ псевдослучайных чисел, секунды.

k	1	10	100	1000	10000	100000	1000000	10000000
-icc -O3								
$Rand128_{32}$	35.399	34.596	34.477	34.467	34.465	34.471	34.526	34.538
$Rand128_{64}$	7.093	3.359	2.985	2.948	2.944	2.945	2.955	2.952
-gcc -O3								
$Rand128_{32}$	89.536	89.033	88.945	88.936	88.936	88.949	88.950	88.996
$Rand128_{64}$	30.714	30.273	30.194	30.180	30.179	30.202	30.188	30.185

4. Модификации “сдвиг влево на 32 бита” и “нарезка по 52 бита”

Поскольку $\alpha \in (0,1)$, двоичное представление каждого выборочного значения этой случайной величины имеет вид

$$\alpha = 0, \alpha^{(1)} \dots \alpha^{(k)} \dots = \sum_{k=1}^{\infty} \alpha^{(k)}\, 2^{-k},$$

причем каждый разряд $\alpha^{(k)}$ мантиссы числа равен нулю или единице.

В дальнейших рассуждениях воспользуемся следующим утверждением [1].

Утверждение. Для того, чтобы случайная величина α была равномерно распределенной в интервале $(0,1)$, необходимо и достаточно, чтобы двоичные цифры $\alpha^{(1)}, \dots, \alpha^{(k)}, \dots$ представляли собой последовательность независимых бернуллиевских случайных величин с вероятностью успеха 1/2: $\mathbf{P}(\alpha^{(k)} = 1) = \mathbf{P}(\alpha^{(k)} = 0) = 1/2$.

Считается, что для применяемых на практике генераторов псевдослучайных чисел ошибки, связанные с конечностью мантиссы незначительны.

Заметим, что алгоритм генерирования псевдослучайного числа (1.1) с $r = 128$ предполагает реализацию 128-битных чисел. Подобную точность невозможно получить на практике, поскольку числа, воспроизводимые на ЭВМ имеют ограниченную мантиссу. В связи с этим, можно предложить алгоритмы, позволяющие моделировать два случайных числа α_n^1, α_n^2 за один шаг.

Модифицируем алгоритм генерирования псевдослучайных чисел (1.1) следующим образом:

$$u_0 = 1,\ u_n \equiv u_{n-1} M (\mathrm{mod}\ 2^{128}),\ v_{n-1} = u_{n-1} \ll 32,\ v_n \equiv v_{n-1} M (\mathrm{mod}\ 2^{128}),$$

$$\alpha_n^1 = u_n 2^{-128},\ \alpha_n^2 = v_n 2^{-128}.$$

Здесь $\ll$ – операция побитового сдвига влево, v_n – вспомогательная величина.

В данной модификации используется побитовый сдвиг влево на 32-бита, поскольку сдвиг на меньшее количество бит дает сильно коррелированные величины псевдослучайных чисел.

Рассмотрим другую модификацию алгоритма (1.1). Поскольку генератор должен выдавать псевдослучайные числа двойной точности, а 11 бит экспоненты и 1 бит знака для числа α, равномерно распределенного в интервале $(0,1)$, не являются значимыми, нас интересуют значащие 52 бита мантиссы. Соответственно, можно “нарезать” 128-битное число по 52 бита, получив два числа двойной точности, мантиссы которых будут состоять из указанных 52 бит, при этом рекомендуется использовать старшие биты.

5. Использование “смешанного” конгруэнтного генератора псевдослучайных чисел

В работе [6] показано, что можно построить достаточно экономичный генератор псевдослучайных чисел, если в качестве стартового числа для моделирования очередного эксперимента использовать случайное число, полученное из надежного физического датчика. Там же показано, что возможная неудовлетворительная равномерность распределения генерируемых им псевдослучайных чисел, эффективно преодолевается с помощью конгруэнтного суммирования.

Рассмотрим генератор с параметрами $M = 5^{17}$ и $r = 40$. Крупномасштабные вычисления будем проводить следующим образом: на каждом процессоре данный генератор будет инициализироваться с использованием некоторого стартового псевдослучайного числа. Будем считать, что данные числа обеспечивают нам достаточную "случайность" последовательности псевдослучайных чисел.

В качестве источников начальных чисел будем рассматривать следующие возможности:

- раздадим каждому процессору числа, сгенерированные мультипликативным датчиком с параметрами $M = 5^{100109}$, $r = 128$,
- раздадим каждому процессору числа, полученныенные конгруэнтным сложением двух сгенерированных датчиком с параметрами $M = 5^{100109}$, $r = 128$ псевдослучайных чисел,
- раздадим каждому процессору числа, полученныенные с помощю логического сложения бинарного представления сгенерированных датчиком с параметрами $M = 5^{100109}$, $r = 128$ псевдослучайных чисел.

Проведем тестирование полученных последовательностией псевдослучайных чисел на многомерную равномерность.

6. Тест на k-равномерность

Далее рассмотрим одно из важнейших свойств последовательности случайных чисел. Пусть $\alpha_1, \alpha_2, ...$ – последовательность "настоящих" выборочных значений для равномерного распределения в $(0, 1)$ и векторы $\eta_1^k, \eta_2^k, ...$ получены из нее следующим образом:

$$\begin{aligned} \eta_1^k &= (\alpha_1, ..., \alpha_k), \\ \eta_2^k &= (\alpha_{k+1}, ..., \alpha_{2k}), \\ \eta_n^k &= (\alpha_{k(n-1)+1}, ..., \alpha_{nk}), ... \end{aligned}$$

С вероятностью 1 такие векторы равномерно заполняют единичный k-мерный куб, т.е. частота попадания вектора в любую прямоугольную область куба стремиться к объему этой области при $n \to \infty$. Бесконечные последовательности чисел, обладающие такими свойствами, называются k-равномерными.

Разобьем единичный k-мерный куб на $s = r^k$ одинаковых кубиков объема r^{-k}. Пусть $m_1, m_2, \ldots, m_s$ – количество точек из последовательности $\eta_1^k, \ldots, \eta_N^k$, попавших в соответствующие кубики и $m_1 + \ldots + m_s = N$. Исследуем, насколько количество попаданий псевдослучайной точки в кубик отклоняется от теоретической частоты попадания в класс для равномерного распределения: $m_1 = m_2 = \ldots = m_s = \frac{N}{s}$.

Известно, что для достаточно большого числа N величина

$$\tilde{\chi}^2_{s-1} = \frac{s}{N} \sum_{i=1}^{s} \Big(m_i - \frac{N}{s} \Big)^2$$

при выполнении гипотезы о равномерности и независимости векторов $\eta_1^k, \eta_2^k, \ldots$ приближенно распределена как χ^2_{s-1} с $s-1$ степенями свободы.

Если $\tilde{\chi}^2_{s-1} > \chi^2_{s-1}(p)$, где $\chi^2_{s-1}(p)$ определяется уравнением

$$\mathbf{P}\big(\chi^2_{s-1} \geq \chi^2_{s-1}(p)\big) = p \ll 1,$$

то вышеуказанная гипотеза ставится под сомнение.

Тест на k-мерную равномерность [1, 2] модифицированных генераторов псевдослучайных чисел проводился для выборки объема 10^{11}, которая была получена объединением начальных 10^{10} чисел из первых 10 подпоследовательностей псевдослучайных чисел. Такая проверка ориентирована на одновременное использование 10 вычислительных процессоров, на каждом из которых предполагается использовать не более 10^{10} псевдослучайных чисел, с последующим осреднением полученных статистических оценок.

Многомерные распределения проверялись на равномерность для $k = 1, 2, \ldots, 9$ по критерию χ^2 с разбиением по каждой оси на 100 частей для $k = 2, 3$ и на 10 частей для $k = 4, \ldots, 9$.

Для $k = 1$ было использовано оптимальное в смысле максимума мощности критерия χ^2 число классов, определяемое формулой [2, 7]

$$s \sim 4\sqrt[5]{2}(N/d_\alpha)^{2/5},$$

где N – объем выборки, а постоянную $d_\alpha = O(1)$ можно практически полагать равной 2. В данном случае $N = 10^{11}$ и $s \approx 87469$.

При анализе результатов статистической проверки использовалось то обстоятельство, что для “настоящих” случайных чисел величина

$$\tilde{\chi}^2_{0,s-1} = \frac{\chi^2_{s-1} - \mathbf{E}(\chi^2_{s-1})}{\sigma(\chi^2_{s-1})} = \frac{\chi^2_{s-1} - (s-1)}{\sqrt{2(s-1)}}$$

при используемых значениях s с большой степенью точности является стандартно нормальной и для нее выполнены следующие соотношения:

$$\mathrm{P}(|\tilde{\chi}^2_{0,s-1}| > 1) \approx 0.32,\ \mathrm{P}(|\tilde{\chi}^2_{0,s-1}| > 2) \approx 0.05,\ \mathrm{P}(|\tilde{\chi}^2_{0,s-1}| > 3) \approx 0.003.$$

На k-мерную равномерность исследовались генераторы псевдослучайных чисел, обладающие большим периодом и удобной методикой распределения потоков псевдослучайных чисел по процессорам при параллельных вычислениях. Указанным критериям отвечают описанные в данной работе модификации алгоритма генерирования псевдослучайного числа (1.1) с $r = 128$, а также генератор псевдослучайных чисел MT2203 математической библиотеки Intel MKL, представляющий собой набор из 1024 псевдослучайных генераторов “Вихрь Мерсенна”, период каждого из которых составляет $2^{22203} - 1$ [8].

Численное тестирование проводилось на десяти процессорах Intel Itanium 2 многопроцессорной системы НКС-160 (ССКЦ) [9]. Компиляция проводилась с помощью Intel C Compiler 10.1 с оптимизационным ключем -O3. Полученные численные результаты сведены в таблице 2.

Анализ полученных результатов, показывает, что все рассматриваемые генераторы псевдослучайных чисел проходят тест на k-мерную равномерность: во всех случаях значения $\tilde{\chi}^2_{0,s-1}$ выходят за пределы интервала $(-1, 1)$ дважды, что не превышает теоретическую вероятность 32%.

Т а б л и ц а 2. Результаты теста на k-мерную равномерность.

n	1	2	3	4	5	6	7	8	9
N	10^{11}	$5\cdot10^{10}$	$\approx3.3\cdot10^{10}$	$2.5\cdot10^{10}$	$2\cdot10^{10}$	$\approx1.67\cdot10^{10}$	$\approx1.43\cdot10^{10}$	$1.25\cdot10^{10}$	$\approx1.11\cdot10^{10}$
s	87469	10^4	10^6	10^4	10^5	10^6	10^7	10^8	10^9
MT2203 (Intel MKL)									
$\tilde{\chi}^2_{0,s-1}$	−1.769	−0.634	−0.744	−1.070	0.081	0.420	−0.543	−0.055	−0.480
$M = 5^{100109}(\mathrm{mod}\ 2^{128})$, $r = 128$									
$\tilde{\chi}^2_{0,s-1}$	−0.186	−0.764	−0.882	0.487	1.176	0.358	1.817	−0.447	N/A
Модификация $Rand128_{52}$ с $M = 5^{100109}(\mathrm{mod}\ 2^{128})$, $r = 128$									
$\tilde{\chi}^2_{0,s-1}$	0.217	0.597	−0.758	−1.394	0.035	−0.484	−0.214	1.300	0.897
Модификация $Rand128_{shift_{32}}$ с $M = 5^{100109}(\mathrm{mod}\ 2^{128})$, $r = 128$									
$\tilde{\chi}^2_{0,s-1}$	−1.346	0.332	−0.093	1.249	0.558	−0.915	−0.229	−0.646	0.035

Список литературы

[1] Михайлов Г.А., Войтишек А.В. Численное статистическое моделирование. М.: Учебно-издательский центр “Академия”, 2006.

[2] Марченко М.А. Михайлов Г.А. Распределенные вычисления по методу Монте-Карло// Автоматика и телемеханика. 2007, № 5. С. 157–170.

[3] Dyadkin I.G., Hamilton K.G. A study of 128-bit multipliers for congruential pseudorandom number generators// Comp. Phys. Comm., 2000, Vol. 125, pp.239-258.

[4] Пригарин С.М. Введение в численное моделирование случайных процессов и полей. - Новосибирск: Изд-во Новосибирского гос. ун-та, 1999.

[5] Трачева Н.В. Модификации конгруэнтного генератора псевдослучайных чисел. - Труды конференции молодых ученых ИВМ и МГ СО РАН, 2009.

[6] Михайлов Г.А. Весовые методы Монте-Карло. Новосибирск, Издательство СО РАН, 2000.

[7] Кендалл М., Стюарт А. Статистические выводы и связи. М., Наука; 1973.

[8] Intel® Math Kernel Library. Vector Statistical Library Notes.
– http://download.intel.com/software/products/mkl/docs/vslnotes.pdf.

[9] Сибирский суперкомпьютерный центр. – http://www2.sscc.ru.