УДК 004.421.5

ИССЛЕДОВАНИЕ АЛГОРИТМОВ КОМПЬЮТЕРНОГО ГЕНЕРИРОВАНИЯ СТАНДАРТНЫХ РАВНОМЕРНЫХ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

Алексеев С.А.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Боровиков С.М. – к. т. н., доцент, доцент кафедры ПИКС

Аннотация. Представлены результаты экспериментальных исследований по определению для различных алгоритмов количества стандартных равномерных псевдослучайных чисел, входящих в совокупность (период) до начала повторения значений этих чисел. Разработан программный комплекс на языке Python, реализующий методы анализа периодов: полный перебор, алгоритм Брента и статистический анализ. Проведено тестирование восьми классических алгоритмов генерирования псевдослучайных чисел, даны рекомендации по использованию алгоритмов.

Ключевые слова: стандартные равномерные псевдослучайные числа, алгоритмы генерирования случайных чисел, период повторения чисел

Введение. Статистическое моделирование выходных параметров и эксплуатационных свойств электронных устройств (ЭУ) основано на получении реализаций случайных параметров с использованием генераторов псевдослучайных чисел. Важной характеристикой является длина последовательности (периода), определяющая количество чисел до начала повторения. При моделировании, требующем сотен тысяч или миллионов реализаций (например, в методах Монте-Карло), недостаточная длина периода может приводить к корреляциям и искажению результатов. Поэтому актуальной задачей является выбор алгоритма генерации псевдослучайных чисел с учётом требуемого объёма данных, производительности и достоверности моделирования ЭУ.

Целью работы является экспериментальное исследование периодов классических и современных алгоритмов генерации равномерных псевдослучайных чисел, разработка методики их тестирования и формулировка рекомендаций по выбору генератора для задач, требующих большого числа реализаций.

Основная часть. При выполнении исследований использованы научные работы [1–4]. Для проведения исследования был разработан программный комплекс на языке Python, включающий проверку восьми классических алгоритмов различных битностей: Middle-Square (1946), LCG-8bit (1949), LCG-16bit (1969), Xorshift-16 (2003), LCG-22bit (1986), LCG-24bit (1986), Xorshift-32 (2003), Xorshift-64 (2003). Для анализа периодов использовались три следующие метода:

1 Полный перебор (для алгоритмов с периодом ≤ 65 тыс. случайных чисел).

2 Алгоритм Брента (для периодов ≤ 100 млн случайных чисел) [1].

3 Статистический анализ на выборке 1 млн чисел (для периодов > 4 млрд. случайных чисел).

Эксперименты проводились, используя компьютер с процессором Intel Core i7-12700K. Время выполнения одного теста ограничивалось 30-ю секундами. Результаты тестирования (таблица 1) показали, что для классических алгоритмов с правильно подобранными параметрами (LCG-8bit, LCG-16bit, LCG-22bit, LCG-24bit) экспериментально полученные периоды полностью соответствуют теоретическим. В то же время алгоритм Middle-Square продемонстрировал период всего в 56 значений вместо примерно теоретически возможных 10 000, что подтверждает ограниченность его использования для важных инженерных задач.

Таблица 1 – Экспериментальные результаты тестирования классических алгоритмов

Алгоритм	**Количество неповторяющихся случайных чисел в периоде:**		**Метод определения количества случайных чисел в периоде**
	теоретический подход	**определено экспериментально**	
1 Middle-Square	~10 000	56	Полный перебор
2 LCG-8bit	256	256	Полный перебор
3 LCG-16bit	65536	65536	Полный перебор
4 Xorshift-16	65535	65535	Полный перебор
5 LCG-22bit	4194304	4194304	Алгоритм Брента
6 LCG-24bit	16777216	16777216	Алгоритм Брента
7 Xorshift-32	$4{,}29\times10^{9}$	Статистический тест продолжительностью 30 секунд	Статистический
8 Xorshift-64	$1{,}8\times10^{19}$	Статистический тест продолжительностью 30 секунд	Статистический

Для современных алгоритмов (PCG32, Xoroshiro128+, Philox, ChaCha20 и др.) период настолько велик (10^{19}…10^{77}), что его прямой поиск методами Брента или перебора физически невозможен. Поэтому для их оценки применялись статистические тесты.

По вычислительной сложности и скорости генерации современные алгоритмы можно разделить на следующие категории:

1 Сверхбыстрые алгоритмы (1…3 такта на число).

К ним относятся Xoroshiro128+ [2] и WyRand. Благодаря оптимизированной структуре, они требуют всего 1…3 такта процессора на генерацию одного числа, что делает их идеальными для задач, требующих максимальной производительности (игровая логика, компьютерная графика и т.п.).

2 Алгоритмы средней сложности (5…15 тактов на число).

В эту группу входят PCG32 и Philox. PCG32 комбинирует быстрый линейный конгруэнтный шаг с дополнительной битовой перестановкой, что улучшает статистические свойства [3] ценой увеличения сложности до 5–10 тактов на число. Philox использует несколько раундов умножения и XOR для обеспечения параллелизма, что требует 10…15 тактов. Эти алгоритмы предлагают оптимальный баланс между скоростью и качеством случайных чисел, подходят для научных расчётов и моделирования.

3 Сложные криптографические алгоритмы (20…60 тактов на число).

ChaCha20 и AES-CTR относятся к этой категории. Они выполняют множественные раунды преобразований (сложения, вращения, подстановки) для обеспечения криптографической стойкости, что увеличивает сложность до 20…60 тактов на число. Хотя их скорость существенно ниже, они гарантируют, что выходная последовательность практически неотличима от истинно случайной. Ниже приводятся рекомендации по выбору алгоритма генерирования псевдослучайных чисел.

1 Для учебных целей и маломасштабных симуляций (до 50…100 тыс. реализаций) достаточно классических алгоритмов с периодом порядка 10^{6}…10^{7}, таких как LCG-24bit или Xorshift-32. Их период может быть проверен экспериментально за приемлемое время, а скорость генерации составляет 5…10 тактов на число.

2 Для производственных задач моделирования (от 100 тыс. до миллиардов реализаций) следует выбирать современные алгоритмы с гарантированно бóльшим периодом и проверенными статистическими свойствами:

а) PCG32, обеспечивает баланс скорости 5…10 тактов на число и высокое качество распределения, период ~$1{,}8\times10^{19}$;

б) Philox, является идеальным для параллельных вычислений, обеспечивает детерминированную генерацию на каждом ядре/узле, период ~$3{,}4\times10^{38}$.

3 Для криптографических задач или моделирования с повышенными требованиями к безопасности следует использовать специализированные алгоритмы: ChaCha20 или AES-CTR. Их сложность выше (20…60 тактов на число), но они обеспечивают криптографическую стойкость и надёжность.

4 Для задач, где скорость генерации критична, а требования к криптостойкости отсутствуют, можно использовать WyRand или Xoroshiro128+ (1…3 такта на число).

5 При выборе алгоритма необходимо убедиться, что его период существенно (минимум на порядок) превышает требуемое количество генерируемых чисел. Для проверки качества генерации следует применять статистические тестовые наборы. Также важно учитывать разрядность случайных чисел: малые разрядности (64…128 бит) подходят для встраиваемых систем, большие (512+ бит) – для криптографии и высоконадёжных приложений.

Заключение. Проведённое исследование подтвердило, что для задач моделирования с большим числом реализаций необходимы алгоритмы с достаточной длиной периода, приемлемой вычислительной сложностью и подтверждёнными статистическими характеристиками. Классические алгоритмы с малыми периодами пригодны в основном для учебных целей. Современные генераторы (PCG, Xoroshiro, Philox) обеспечивают необходимую длину периода и производительность, а криптографические алгоритмы (ChaCha20, AES-CTR) – повышенную надёжность. Разработанная методика тестирования позволяет оценивать пригодность генераторов и избегать ошибок, связанных с повторением псевдослучайных чисел.

Список литературы

1. Brent, R.P. An improved Monte Carlo factorization algorithm / R.P. Brent // BIT Numerical Mathematics. – 1980. – Vol. 20, N 2. – Pp. 176–184.

2. L'Ecuyer, P. TestU01: A C Library for Empirical Testing of Random Number Generators / P. L'Ecuyer, R. Simard // ACM Transactions on Mathematical Software. – 2007. – Vol. 33, N 4. – Article 22.

3. Vigna, S. An experimental exploration of Marsaglia's xorshift generators, scrambled / S. Vigna // ACM Transactions on Mathematical Software. – 2016. – Vol. 42, N 4. – Article 30.

4. O'Neill, M.E. PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation / M.E. O'Neill // Technical Report, Harvey Mudd College. – 2014. – 58 p.

UDC 004.421.5

STUDY OF ALGORITHMS FOR COMPUTER GENERATION OF STANDARD PSEUDORANDOM NUMBERS WITH UNIFORM DISTRIBUTION

Alekseev S.A.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Borovikov S.M. – Cand. of Sci., Associate Professor, Associate Professor of the Department of ICSD

Annotation. The article presents the results of experimental studies to determine, for various algorithms, the number of standard, uniformly distributed pseudorandom numbers included in a set (period) before the values of these numbers begin to repeat. A Python software package has been developed that implements period analysis methods: exhaustive search, Brent's algorithm, and statistical analysis. Eight classical pseudorandom number generation algorithms have been tested, and recommendations for their use are provided.

Keywords: Standard uniformly distributed pseudo-random numbers, random number generation algorithms, number repetition period