Математические структуры и моделирование 2020. № 4(56). С. 64–74

УДК 004.056
DOI 10.24147/2222-8772.2020.4.64-74

ПОСТРОЕНИЕ ГЕНЕРАТОРА ПСЕВДОСЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ НА ОСНОВЕ КЛЕТОЧНОГО АВТОМАТА

Н.Ф. Богаченко
к.ф.-м.н., доцент, e-mail: nfbogachenko@mail.ru
И.О. Горохов
студент, e-mail: vixxman@ya.ru

Омский государственный университет им. Ф.М. Достоевского, Омск, Россия

Аннотация. Работа посвящена реализации и тестированию генератора псевдослучайных последовательностей, построенного на основе модели клеточного автомата.

Ключевые слова: псевдослучайная последовательность, клеточный автомат.

Введение

Клеточные автоматы — самостоятельные объекты теоретического изучения. Изначально клеточные автоматы были разработаны для исследования самовоспроизводящихся машин. В то же время клеточные автоматы представляют собой мощный инструмент для моделирования многих физических процессов. Главное преимущество клеточных автоматов — это относительная простота в сочетании с большими возможностями для моделирования. И, конечно, такая потенциально мощная дискретная модель неоднократно находила применение в криптографии [1–7]. В данной работе модель клеточного автомата использована для построения генератора псевдослучайных последовательностей (ПСП).

Клеточный автомат является дискретной моделью, состоящей из решётки ячеек (клеток), каждая из которых может находиться в одном из нескольких состояний. В простейшем случае таких состояний 2: состояние 0 – клетка «мертва», состояние 1 – клетка «жива». Решётка автомата может быть любой размерности. Возможно создание не только плоского автомата, но и двумерного, и многомерного. Также для каждой ячейки определяется окрестность — множество клеток, которые будут влиять на дальнейшее состояние ячейки. Для работы автомата требуется первоначальное заполнение всех его ячеек, а также задание правил перехода из одного состояния в другое. Правила перехода являются общими для всех ячеек. В каждый дискретный момент времени к каждой клетке применяется правило перехода и исходя из состояний ячеек в окрестности текущая клетка меняет своё состояние. Поколением автомата называется множество ячеек в каждый дискретный момент времени. А эволю-

ция клеточного автомата — это изменение конфигурации автомата с течением времени.

Клеточные автоматы классифицируются по скорости стабилизации состояний и наличию устойчивых структур.

1. Класс 1 — в ходе эволюции быстрая стабилизация состояний автомата. Случайные структуры быстро исчезают.

2. Класс 2 — в ходе эволюции быстрая стабилизация состояний автомата. Большинство случайных структур исчезает, но некоторые остаются.

3. Класс 3 — в ходе эволюции любые стабильные структуры почти сразу уничтожаются шумом. Автомат заполнен псевдослучайными, хаотическими последовательностями.

4. Класс 4 — в ходе эволюции образуются устойчивые структуры, взаимодействующие друг с другом сложным образом. Данные структуры способны выживать долгое время.

В рамках криптографических задач несомненную ценность представляет 3-й класс клеточных автоматов.

1. Реализация клеточного автомата

Проект был реализован в среде Visual Studio Community Edition 2017 на языке C# с использованием технологии Windows Forms. Интерфейс программы представлен на рисунках 1–3.

1. Форма PRNG (рис. 1) отвечает за инициализацию Visualiser, извлечение состояний ячеек клеточного автомата и запись в файл сгенерированных чисел.

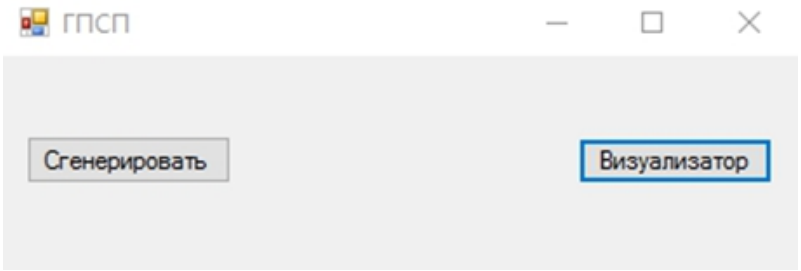

Рис. 1. Главная форма PRNG

2. Форма Visualiser (рис. 2) отвечает за инициализацию клеточного автомата, установку «по умолчанию» правил перехода и количества состояний клетки, отрисовку клеточного автомата. Имеет набор горячих клавиш, таких как пересоздание клеточного автомата, пауза, пошаговая эволюция и запуск формы Setting.

3. Форма Setting (рис. 3) отвечает за формирование правил перехода и смену количества состояний клетки.

Цикл работы клеточного автомата.

1. Точка входа main инициализирует форму PRNG.

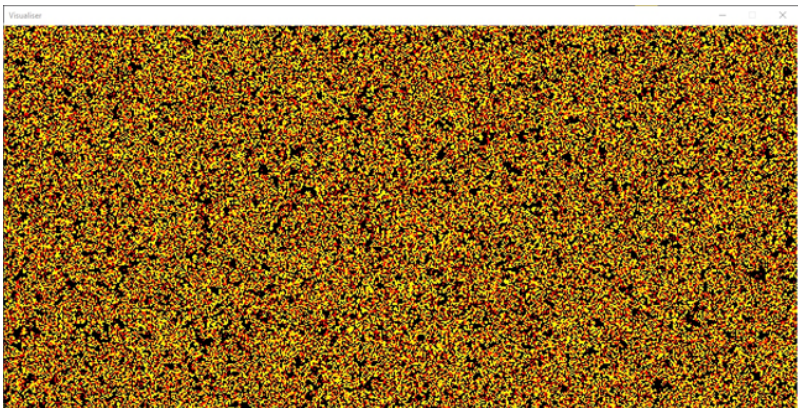

Рис. 2. Форма Visualiser

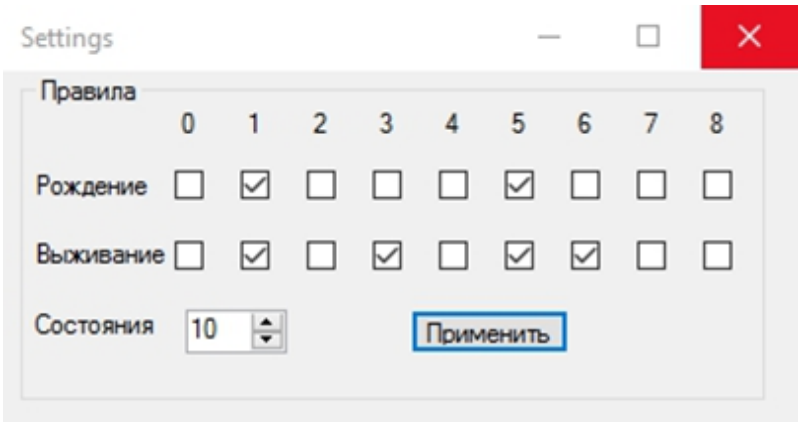

Рис. 3. Форма Setting

2. Класс PRNG инициализирует класс Visualiser.

3. Класс Visualiser создаёт Bitmap, задаёт «по умолчанию» правила перехода и количество состояний ячейки клеточного автомата и вызывает конструктор класса Automat. Создаёт класс Palette, представляющий из себя кортеж из 3-х байт (RGB). Количество элементов в списке цветов соответствует текущему количеству состояний.

4. Начальные состояния всех клеток заполняются случайным образом системной функцией System.Random.

5. Каждые 16 мс срабатывает таймер, который вызывает функцию расчёта следующего поколения автомата и перерисовку Bitmap.

Плоскость автомата представляет собой двумерное поле размерности 600×400. Клеточный автомат в текущей конфигурации использует полную окрестность Мура — окрестность 3×3, включая состояние текущей клетки. Данная окрестность была выбрана не случайно. Исследователями доказано, что именно она из всех стандартных окрестностей обладает оптимальным лавинным эффектом [8]. Этот лавинный эффект характеризуется равномерным распространением изменений состояний клеточного автомата с максимально возможной

скоростью. Каждая ячейка автомата может принимать целочисленное значение в диапазоне [0, 256]. В ходе вычислительных экспериментов выяснилось, что чем большее число состояний используется, тем более плавные и периодичные изменения происходят по всей плоскости автомата. Поэтому максимальным на данный момент выбран предел в 11 состояний: состояния клеток — это целые числа от 0 до 10. Каждому состоянию ставится в соответствие цвет в процессе отрисовки автомата.

Важно пояснить принцип работы правил. На рисунке 3 можно увидеть 18 checkbox'ов, по 9 в каждой строке. Первая строка «рождение» отвечает за условия, при которых клетка может «ожить», т. е. сменить состояние «0» на «1». В представленном примере для «рождения» клетки необходимо, чтобы количество ячеек не в состоянии «0» в окрестности 3×3 было равно 1 или 5. По похожему принципу работает механизм «выживание». Вторая строка checkbox'ов устанавливает условия, при которых текущая клетка сменит своё i-ое состояние на $i + 1$-е состояние, в противном случае клетка примет состояние «0». Для правил на рисунке 3 предположим, что текущая клетка находится в состоянии «4». Тогда клетка перейдёт в состояние «5», если в её окрестности 1, 3, 5 или 6 клеток находятся не в состоянии «0». В противном случае, если количество не 0-х клеток будет, например 2, то текущая клетка перейдёт в состояние «0». Если клетка находится в максимально возможном состоянии (в текущей конфигурации это «10»), то клетка принимает состояние «0» в любом случае.

2. Реализация генератора ПСП

В текущей реализации происходит генерация 40 000 1-байтовых чисел с последующей конвертацией в строку двоичного представления и записью в файл. Каждое псевдослучайное число является продуктом разных поколений автомата.

На первом этапе вычислительных экспериментов алгоритм генерации псевдослучайного числа (ПСЧ) состоял из следующих шагов (рис. 4)

1. На вход поступает массив 600×400 клеток, из которого случайным образом выбираются 18 клеток, и извлекаются их состояния.

2. На основе полученных целых чисел формируются 2 матрицы размерностью 3×3 и перемножаются между собой.

3. Элементы полученной результирующей матрицы подвергаются операции XOR. Каждый текущий элемент «ксорится» с последующим, результат операции пишется в последующий, т. е. $[i, j] = [i, j] \oplus [i, j - 1]$. Кроме редактирования самой матрицы каждое полученное число переводится в двоичный вид и записывается в строку.

4. Двоичная строка, полученная на 3 этапе, переводится в целое число и «ксорится» с целочисленным числом, полученном на 3 этапе предыдущей итерации алгоритма.

5. Полученное число переводится опять в двоичную строку, и от неё берётся хеш-функция SHA256.

6. Из полученной хэш-строки случайным образом извлекаются 8 символов,

из которых тоже случайным образом извлекается по 1 биту в случайной позиции. Полученная последовательность бит является сгенерированным целым псевдослучайным числом.

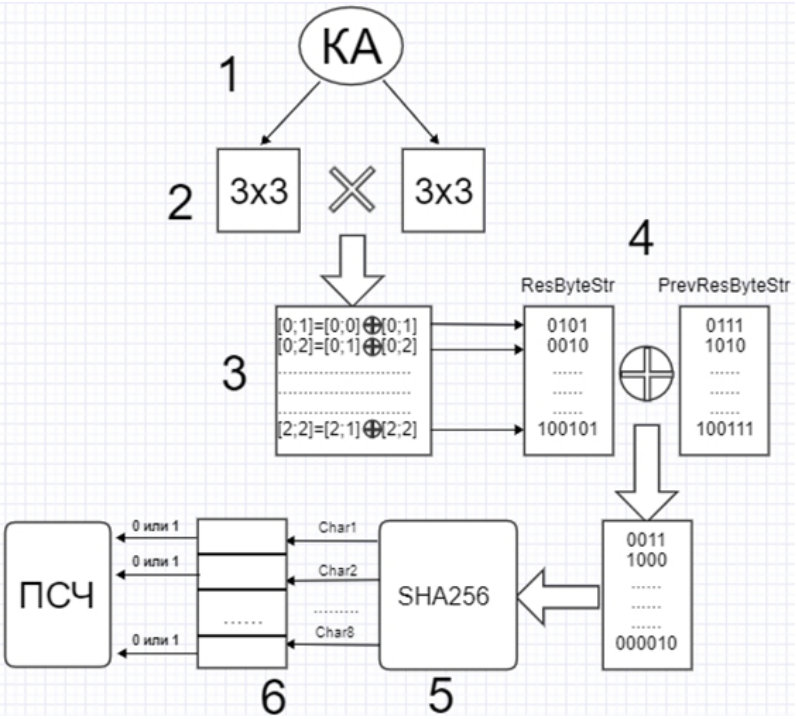

Рис. 4. Поэтапная схема генерации ПСЧ

3. Тестирование генератора ПСП

В качестве инструмента тестирования был выбран пакет статистических тестов NIST STS. На данный момент это одно из самых популярных средств тестирования генераторов ПСП [9–11]. Данный пакет содержит набор тестов, направленных на выявление каких-либо дефектов в последовательности. В таблице 1 приведены список тестов и выявляемые ими дефекты случайности [9].

Для проведения тестирования было сгенерировано 40 000 однобайтовых целых чисел, разбитых на 100 подпоследовательностей длиною в 400 чисел. Показателем качества последовательности является значение P-Value, находящееся в диапазоне от 0 до 1 включительно. Если P-Value принимает значение 0 — последовательность является абсолютно предсказуемой. Если P-Value принимает значение 1 — последовательность случайная. Тест считается пройденным, если P-Value > 0,01. Усреднённые результаты тестирования 100 подпоследовательностей приведены в таблице 2.

В рамках исследования несомненный интерес представляет наглядное сравнение статистических показателей с существующими алгоритмами генерации ПСП. Для сравнения были выбраны 5 генераторов, имеющихся в пакете NIST

Таблица 1. Статистические тесты пакета NIST STS

№	*Название теста*	*Определяемый дефект*
1	Частотный (монобитный тест)	Слишком много нулей либо единиц во всей последовательности. Их количество должно быть примерно одинаковым.
2	Блочный тест на частоту	Слишком много нулей либо единиц внутри m-битных блоков. Их количество должно быть примерно одинаковым.
3	Тест на серийность	Отклонение числа серий из единиц (из нулей) различной длины, в частности, частоты чередования 1 и 0.
4	Тест кумулятивных сумм	Слишком много нулей или единиц в начале последовательности.
5	Спектральный тест на основе дискретного преобразования Фурье	Выявление периодических слагаемых (шаблонов) в двоичной последовательности.
6	Тест на линейную сложность	Сниженная линейная сложность последовательности.
7	Тест на длиннейшую серию единиц в блоке	Отклонение от теоретического закона распределения максимальных длин серий единиц
8	Тест не пересекающихся шаблонов	Непериодические шаблоны встречаются слишком часто.
9	Тест пересекающихся шаблонов	Слишком часто встречаются m-битные последовательности единиц.
10	Тест рангов бинарных матриц	Отклонение значений рангов матрицы, анализ линейной зависимости элементов последовательности.
11	Тест приблизительной энтропии	Неравномерность распределения m-грамм (регулярность свойств генератора последовательностей).
12	Тест на периодичность	Неравномерность распределения m-битных слов.

Таблица 2. Статистические тесты пакета NIST STS

№	*Название теста*	*P-Value*	*Результат*
1	Частотный (монобитный тест)	0,47918666	Тест пройден
2	Блочный тест на частоту	0,47789048	Тест пройден
3	Тест на серийность	0,53582583	Тест пройден
4	Тест кумулятивных сумм	0,48842218	Тест пройден
5	Спектральный тест на основе дискретного преобразования Фурье	0,52557954	Тест пройден
6	Тест на линейную сложность	0,58009578	Тест пройден
7	Тест на длиннейшую серию единиц в блоке	0,53939177	Тест пройден
8	Тест непересекающихся шаблонов	0,53161099	Тест пройден
9	Тест пересекающихся шаблонов	0,49977973	Тест пройден
10	Тест рангов бинарных матриц	0,46442867	Тест пройден
11	Тест приблизительной энтропии	0,50172644	Тест пройден
12	Тест на периодичность	0,53398629	Тест пройден

STS. Тестирование генераторов проводилось с теми же параметрами, что и тестирование генератора ПСП на основе клеточного автомата. Результаты тестирования приведены в таблице 3, где 12 строк — это 12 тестов, порядок и название которых соответствует таблице 1. Численные значения в таблице отражают усреднённые значения P-Value по каждому из тестов.

Как видно из таблицы 3, реализованный эвристическим путём алгоритм генерации ПСП на основе КА полностью оправдывается статистическими показателями и в рамках отдельных тестов демонстрирует лучшие значения среди сравниваемых генераторов. Важно понимать, что сложно реализовать алгоритм, который показывал бы отличные статистические результаты по всем тестам. Поэтому не лишним будет перейти к сравнению средних значений по всем тестам каждого генератора. Средние значения P-Value по 12 тестам каждого из генераторов приведены далее.

1. Линейный конгруэнтный генератор: 0,49059889.
2. Квадратичный конгруэнтный генератор: 0,4927076.
3. Micali-Shcnorr: 0,51080925.
4. Blum-Blum-Shub: 0,48831351.
5. Модулярно-экспоненциальный генератор: 0,50401959
6. Генератор на основе КА: 0,51316036.

Следующим этапом являлось исследование зависимости характеристик ПСП от параметров клеточного автомата и от алгоритма генерации ПСЧ.

Непосредственное влияние на эволюцию клеточного автомата и выходную последовательность оказывают количество состояний и правила перехода. Для краткости изложения введём условное обозначение текущих параметров авто-

Таблица 3. Статистические тесты пакета NIST STS (Г1 — линейный конгруэнтный генератор, Г2 – квадратичный конгруэнтный генератор, Г3 — Micali-Shcnorr, Г4 — Blum-Blum-Shub, Г5 — модулярно экспоненциальный генератор) и Г6 – генератор на основе КА

№	Г1	Г2	Г3	Г4	Г5	**Г6**
1	0,4808706	0,4865253	0,54483113	0,43734769	0,50222298	0,47918666
2	0,48264618	0,42435623	0,48836388	0,41852682	0,48641801	0,47789048
3	0,48647511	0,51587904	0,51144753	0,50576546	0,51370477	0,53582583
4	0,47519589	0,49126063	0,54919581	0,45471316	0,50266348	0,48842218
5	0,47599264	0,49951031	0,47713432	0,46424687	0,49055585	0,52557954
6	0,53060168	0,5259644	0,52680149	0,50991242	0,52988905	0,58009578
7	0,48308109	0,45039169	0,52299557	0,51815044	0,45264045	0,53939177
8	0,53227178	0,53203322	0,53045616	0,53667222	0,53409214	0,53161099
9	0,46629077	0,51013325	0,49721174	0,5094293	0,52428777	0,49977973
10	0,47734816	0,47754565	0,47875372	0,50464686	0,50132254	0,46442867
11	0,46561875	0,50866382	0,50724831	0,50300328	0,46291012	0,50172644
12	0,53079404	0,49022771	0,49527144	0,49734770	0,54752802	0,53398629

мата на примере параметров на рисунке 3. Текущие настройки можно записать следующим образом 1-5/1-3-5-6/10, где 1-5 – количество живых клеток, необходимых для рождения клетки, 1-3-5-6 — количество живых клеток, необходимых для смены i-го состояния на $i+1$-е, 10 — количество состояний. Были проведены следующие изменения параметров.

1. Увеличение количества состояний без смены правила (1-5/1-3-5-6/30).

2. Уменьшение количества состояний без смены правила (1-5/1-3-5-6/5).

3. Смена правила, при котором в автомате появляются неустойчивые, но цикличные структуры (1-2-4-6/0-3-5/10).

4. Уменьшение количества состояний в правиле с неустойчивыми структурами (1-2-4-6/0-3-5/5).

5. Смена правила, при котором в автомате появляются устойчивые и цикличные структуры (2-7/2-3-5-6/20).

Средние значения P-Value по 12-ти тестам по каждой из 5 конфигураций, а также среднее значение P-Value для исходной конфигурации представлены далее.

1. 1-5/1-3-5-6/30: 0,50580696.

2. 1-5/1-3-5-6/5: 0,49698226.

3. 1-2-4-6/0-3-5/10: 0,50058212.

4. 1-2-4-6/0-3-5/5: 0,5039827.

5. 2-7/2-3-5-6/20: 0,50384019.

6. 1-5/1-3-5-6/10: 0,513160363 (исходная конфигурация).

По данным вычислительного эксперимента можно сделать вывод, что никакие из изменений не привели к ощутимым отклонениям ни в лучшую, ни в худшую сторону. А это, в свою очередь, говорит о том, что при текущем алгоритме

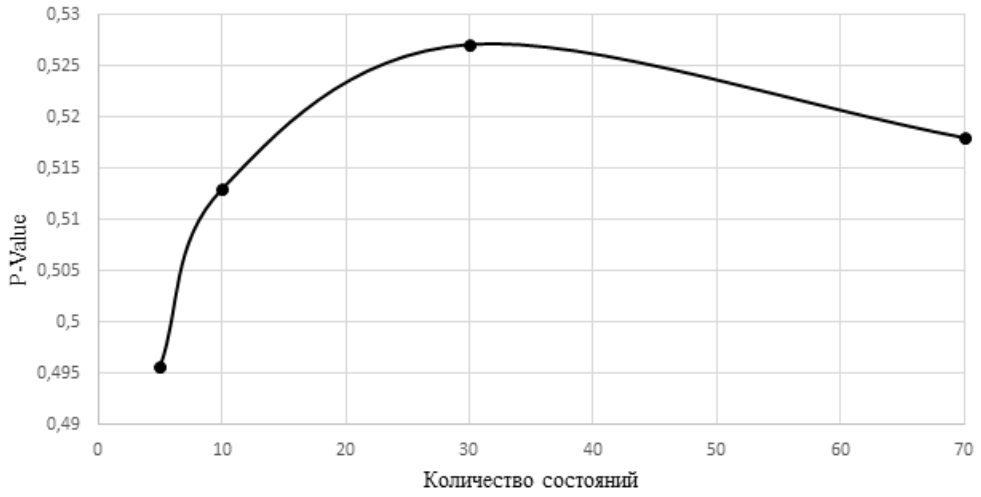

Рис. 5. Зависимость P-Value от количества состояний КА

параметры клеточного автомата не играют существенной роли в формировании выходной последовательности.

Перейдём к исследованию и модификации алгоритма генерации ПСЧ. Первым шагом по упрощению алгоритма стало удаление 5-го и 6-го этапов (рис. 4). Также было реализовано «вбрасывание» 40 случайных состояний в случайные клетки каждые 20 поколений автомата. После вышеупомянутых модификаций было проведено тестирование и сравнение изменённого алгоритма на разных параметрах клеточного автомата.

Средние значения P-Value по 12-ти тестам для каждой конфигурации представлены далее.

1. Без SHA256: 0,51309172.
2. Без SHA256 на 5 состояниях: 0,49571853.
3. Без SHA256 на 30 состояниях: 0,5271106.
4. С SHA256 на 10 состояниях (исходная конфигурация): 0,513160363.
5. Без SHA256 на 70 состояниях: 0,518555851.
6. Без SHA256 2-4-7/1-3-6/70: 0,51142996.
7. Без SHA256 1-2-4-6/0-3-5/30: 0,507571585.

На данном этапе исследования нам удалось убедиться в том, что уменьшение количества состояний клеток негативно сказывается на последовательности, как и увеличение после определённого порога (см. рис. 5).

Заключение

Итогом данной работы является программная реализация генератора ПСП на основе клеточного автомата, а также исследовательский вклад в изучение клеточного автомата как источника программных шумов для генератора ПСП. Текущая версия генератора обладает хорошими статистическими показателями, сравнимыми с известными генераторами ПСП.

Литература

1. Ефремова А.А., Гамова А.Н. Генератор псевдослучайных чисел на основе клеточных автоматов // Компьютерные науки и информационные технологии. Материалы Международной научной конференции, 2016. С. 131–134.
2. Жуков А.Е. Клеточные автоматы в криптографии. Часть 1 // Вопросы кибербезопасности. 2017. № 3(21). С. 70–76.
3. Жуков А.Е. Клеточные автоматы в криптографии. Часть 2 // Вопросы кибербезопасности. 2017. № 4(22). С. 47–66.
4. Ключарёв П.Г. Метод построения криптографических хэш-функций на основе итераций обобщённого клеточного автомата // Вопросы кибербезопасности. 2017. № 1(19). С. 45–50.
5. Мухамеджанов Д.Д, Левина А.Б. Генератор псевдослучайных чисел на основе клеточных автоматов // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18, № 5. С. 894–900.
6. Сухинин Б.М. Высокоскоростные генераторы псевдослучайных последовательностей на основе клеточных автоматов // Прикладная дискретная математика. 2010. № 2. С. 34–41.
7. Selvavinayagam G., Umamaheswari K. Performance Enhancement in Image Encryption using Parallel Non Uniform Cellular Automata based PRN Generation // Asian Journal of Research in Social Sciences and Humanities. 2017. Vol. 7, No. 2. P. 1079–1093.
8. Сухинин Б.М. О лавинном эффекте в клеточных автоматах // Объединённый научный журнал. 2010. № 8. С. 41–46.
9. Будько М.Б., Будько М.Ю., Гирик А.В., Грозов В.А. Методы генерации и тестирования случайных последовательностей. СПб : Университет ИТМО, 2019. С. 44.
10. Григорьев А.Ю. Методы тестирования генераторов случайных и псевдослучайных последовательностей // Учёные записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2017. № 1. С. 22–28.
11. Миненко А.И. Экспериментальное исследование эффективности тестов для проверки генераторов случайных чисел // Вестник СибГУТИ. 2010. № 4. С. 36–46.

CONSTRUCTION OF A PSEUDO-RANDOM SEQUENCE GENERATOR BASED ON A CELLULAR AUTOMATON

N.F. Bogachenko
Ph.D.(Phys.-Math.), Associate Professor, e-mail: nfbogachenko@mail.ru
I.O. Gorohov
Student, e-mail: vixxman@ya.ru

Dostoevsky Omsk State University, Omsk, Russia

Abstract. The work is devoted to the implementation and testing of a pseudo-random sequence generator based on a cellular automaton model.

Keywords: pseudo-random sequence, cellular automaton.

References

1. Efremova A.A. and Gamova A.N. Generator psevdosluchainykh chisel na osnove kletochnykh avtomatov. Komp'yuternye nauki i informatsionnye tekhnologii, Materialy Mezhdunarodnoi nauchnoi konferentsii, 2016, pp. 131–134. (in Russian)
2. Zhukov A.E. Kletochnye avtomaty v kriptografii. Chast' 1. Voprosy kiberbezopasnosti, 2017, no. 3(21), pp. 70–76. (in Russian)
3. Zhukov A.E. Kletochnye avtomaty v kriptografii. Chast' 2. Voprosy kiberbezopasnosti, 2017, no. 4(22), pp. 47–66.
(in Russian)
4. Klyucharev P.G. Metod postroeniya kriptograficheskikh khesh-funktsii na osnove iteratsii obobshchennogo kletochnogo avtomata. Voprosy kiberbezopasnosti, 2017, no. 1(19), pp. 45–50.
(in Russian)
5. Mukhamedzhanov D.D and Levina A.B. Generator psevdosluchainykh chisel na osnove kletochnykh avtomatov. Nauchno-tekhnicheskii vestnik informatsionnykh tekhnologii, mekhaniki i optiki, 2018, vol. 18, no. 5. pp. 894–900. (in Russian)
6. Sukhinin B.M. Vysokoskorostnye generatory psevdosluchainykh posledovatel'nostei na osnove kletochnykh avtomatov. Prikladnaya diskretnaya matematika, 2010, no. 2, pp. 34–41. (in Russian)
7. Selvavinayagam G. and Umamaheswari K. Performance Enhancement in Image Encryption using Parallel Non Uniform Cellular Automata based PRN Generation. Asian Journal of Research in Social Sciences and Humanities, 2017, vol. 7, no. 2, pp. 1079–1093.
8. Sukhinin B.M. O lavinnom effekte v kletochnykh avtomatakh. Ob"edinennyi nauchnyi zhurnal, 2010, no. 8, pp. 41–46. (in Russian)
9. Bud'ko M.B., Bud'ko M.Yu., Girik A.V., and Grozov V.A. Metody generatsii i testirovaniya sluchainykh posledovatel'nostei. SPb., Universitet ITMO, 2019, pp. 44. (in Russian)
10. Grigor'ev A.Yu. Metody testirovaniya generatorov sluchainykh i psevdosluchainykh posledovatel'nostei. Uchenye zapiski UlGU. Ser. Matematika i informatsionnye tekhnologii, UlGU, Elektron. zhurn., 2017, no. 1, pp. 22–28. (in Russian)
11. Minenko A.I. Eksperimental'noe issledovanie effektivnosti testov dlya proverki generatorov sluchainykh chisel. Vestnik SibGUTI, 2010, no. 4, pp. 36–46. (in Russian)

Дата поступления в редакцию: 01.11.2020