УДК 004.421.5
doi:10.15217/issn1684-8853.2018.2.2

ГЕНЕРАЦИЯ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ НА ОСНОВЕ ПРЕОБРАЗОВАНИЙ ГРАФИЧЕСКИХ ОБЪЕКТОВ

Д. А. Крюков[a], *канд. техн. наук, dm.bk@bk.ru*
И. А. Тескер[a], *магистрант, teskerivan.t@gmail.com*
[a]*Московский технологический университет, Вернадского пр., 78, Москва, 119456, РФ*

Постановка проблемы: *одним из методов защиты данных является их шифрование. Для шифрования применяются алгоритмы, работа которых предполагает использование случайных или псевдослучайных чисел. Данные числа, критически влияющие на криптостойкость шифра, создаются с помощью генераторов случайных и псевдослучайных чисел. Для того чтобы сгенерированные числа могли использоваться в криптоалгоритмах, они должны иметь признаки истинно случайной последовательности, т. е. быть устойчивыми к реверсивному определению всего потока сгенерированных чисел по известным его частям.* ***Цель:*** *разработка метода генерации псевдослучайных чисел, пригодных для дальнейшего использования в криптографических алгоритмах.* ***Результаты:*** *представлена методика генерации псевдослучайных чисел, в основе которой лежит процедура обработки изображений методом K-средних и алгоритм вихрь Мерсенна. В качестве источника генерации псевдослучайных чисел предложено применять случайные графические объекты. Приведен пример, визуализирующий результаты работы методики. Новизна подхода состоит в том, что энтропия псевдослучайной величины повышается путем использования преобразованных сведений о графических объектах, обладающих повышенной персонализацией. Полученный алгоритм обладает относительно малой скоростью, но при этом остается относительно устойчивым благодаря уникальным изображениям и двукратному применению алгоритма вихрь Мерсенна.* ***Практическая значимость:*** *разработанный метод генерации псевдослучайных чисел может быть использован в криптоалгоритмах, ориентированных в том числе на работу с большими объемами данных.*

Ключевые слова *— криптография, генератор псевдослучайных чисел, K-метод, обработка изображений, кластеризация, вихрь Мерсенна.*

Цитирование: Крюков Д. А., Тескер И. А. Генерация псевдослучайных чисел на основе преобразований графических объектов // Информационно-управляющие системы. 2018. № 2. С. 2–7. doi:10.15217/issn1684-8853.2018.2.2
Citation: Kryukov D. A., Tesker I. A. Pseudo Number Generation based on Graphic Object Processing. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2018, no. 2, pp. 2–7 (In Russian). doi:10.15217/issn1684-8853.2018.2.2

Введение

Развитие вычислительных технологий провоцирует возникновение рисков, связанных с повышением уязвимости существующих способов защиты данных. Такие методы защиты информации, как шифрование данных, электронная цифровая подпись, нуждаются в постоянном модифицировании для сохранения своей эффективности. Одним из важнейших компонентов большинства криптографических алгоритмов являются случайные или псевдослучайные числа, формируемые соответствующими генераторами. Данные генераторы способны как сделать зашифрованные данные полностью недоступными для третьих лиц, так и оказаться слабым звеном криптографического алгоритма, существенно снизив его устойчивость к различным атакам.

Вероятностные последовательности

Необходимость использования случайных и псевдослучайных чисел имеется в большинстве криптографических алгоритмов и прикладных приложений. К примеру, многие криптографические протоколы используют генераторы случайных или псевдослучайных чисел для вспомогательных данных при создании цифровых подписей, векторов инициализации или для иных методов повышения устойчивости шифров.

Существует два базовых алгоритма создания вероятностных последовательностей: генераторы случайных чисел и генераторы псевдослучайных чисел. Фактически результатом обоих алгоритмов являются строки, состоящие из нулей и единиц, которые могут быть впоследствии разделены на подстроки или блоки случайных чисел.

Первый тип — генераторы случайных чисел. Процессы, лежащие в основе генераторов случайных чисел, основаны на анализе естественных процессов, обладающих энтропией, т. е. событий, исход которых заранее неизвестен. Например, такими процессами могут быть вероятностные процессы квантовой физики, дробовой или тепловой шум элементов ЭВМ, на котором будет производиться генерация случайных величин. Зарегистрированные случайные величины, порождаемые данными процессами, после опреде-

ленных математических манипуляций образуют числовую последовательность.

Другим типом является генератор псевдослучайных чисел. Алгоритмы образования псевдослучайных чисел используют так называемые зерна, псевдослучайные числа, являющиеся аналогом данных о случайном процессе у генератора случайных чисел.

Целью вышеописанных алгоритмов является создание последовательности случайных или псевдослучайных чисел. Последовательность, чтобы называться случайной, должна состоять из элементов с одинаковым распределением, при этом элементы этой последовательности должны быть независимы, т. е. изменение значения одного числа не должно влиять на другие элементы [1].

С момента появления необходимости в генерации случайных чисел с достаточной скоростью и в достаточном объеме, чтобы использовать ее в криптографии и других сферах, создано большое количество генераторов как случайных, так и псевдослучайных чисел [2].

Возможно, самый распространенный метод генерации псевдослучайных чисел — конгруэнтный. По сути метод является преобразованием некого числа формулой вида

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

где X — зерно (некое случайное число); a, c, m — некие константы. Данный метод малоэффективен ввиду зависимости последующего элемента генерируемой строки от предыдущего [3].

Другим распространенным алгоритмом генерации псевдослучайных чисел является так называемый вихрь Мерсенна [4–6]. Этот алгоритм построен на числах, именуемых числами Мерсенна. Эти числа образуют последовательность A000668 в Энциклопедии целочисленных последовательностей Нила Слоуна и вычисляются по правилу [7–10] $M_n = 2^n - 1$.

Алгоритм включает в себя регистр сдвига с линейной обратной связью и процесс, именуемый закалкой. Вихрь Мерсенна был разработан японскими учеными Макото Мацумото и Такудзи Нисимура в 1997 г. и с тех пор был существенно модернизирован. Вихрь Мерсенна обладает достаточно высокой скоростью генерации и имеет реализации на многих платформах и языках программирования [11–14].

Вышеперечисленные алгоритмы генерации псевдослучайных чисел имеют относительно простую реализацию, позволяют генерировать большой поток псевдослучайных чисел, однако не обладают криптостойкостью, что делает возможным их применение только как часть более сложного генератора.

Алгоритм генерации псевдослучайных чисел

В качестве составляющей алгоритма генерации псевдослучайных чисел, изложенного в данной работе, предлагается использовать метод *K*-средних, или *K*-метод [15–18]. *K*-метод заключается в группировании неких данных в несколько сплоченных кластеров. В качестве обрабатываемых данных возможно выполнить выборку любых типов данных. В статье предложен алгоритм генерации псевдослучайных чисел на основе преобразований графических материалов — изображений, поскольку изображения удобны для работы с входными данными, что подтверждается следующими объективными обстоятельствами. Во-первых, уникальное изображение (например, снятое камерой смартфона пользователем и сохраненное в памяти аппарата) обладает повышенной персонализацией. Во-вторых, одно изображение способно предоставить достаточно большое количество чисел для дальнейшей обработки (далее — зерен). Например, изображение 128×128, применяющее цветовой формат RGB, предоставит минимум 49 152 числа, не имеющих прямой зависимости друг от друга и готовых к дальнейшему преобразованию.

Предположим, что в распоряжении алгоритма имеется база уникальных изображений. Наиболее эффективным методом выборки будет применение алгоритма случайной (или псевдослучайной) выборки. Таким образом, первым шагом алгоритма является использование вышеупомянутого вихря Мерсенна. Причиной выбора данного алгоритма является, как было указано ранее, его высокая скорость и большое количество реализаций. На данном этапе от вихря Мерсенна требуется произвести выборку из имеющихся в распоряжении изображений. Снижение энтропии от применения вихря Мерсенна не повлияет на итоговый результат по причине уникальности изображений. Для простоты алгоритма количество изображений выбирается вида N^2.

После получения N^2 изображений производится преобразование всех изображений к единому размеру: p пикселей на p пикселей на 3 (здесь и далее рассматривается кодирование изображений в формате RGB). Затем производится склейка в один трехмерный массив размерности $(N \times N \times p \times p) \times 3$.

Следующей частью метода генерации псевдослучайных чисел является *K*-метод. Сначала вихрь Мерсенна случайным образом инициализирует K кластерных точек (центроидов): $\lambda_1, \lambda_2, \ldots, \lambda_k \in R^n$, которые определяют кластеры имеющейся матрицы данных, состоящей из пикселей изображения. Путем заданного числа итераций эти кластерные точки будут перемещены ближе к геометрическому центру кластеров.

Для каждого i имеем

$$c^i = \arg\min \left\| x^{(i)} - \lambda_{C_i} \right\|^2.$$

Для каждого j имеем

$$\lambda_j = \frac{1}{c_k} \sum_{i \in C_K} x^{(i)}.$$

Алгоритм повторяется до полной сходимости процесса кластеризации [19]. Таким образом, получено K центроидов, разделивших трехмерный массив на k групп, не связанных друг с другом. Полученные данные следует преобразовать в массив $(N \times N \times p \times p) \times 3$, затем обработать для упрощения классификации логистической сигмоидой вида

$$y = \frac{1}{1 + e^{-x}}. \qquad (*)$$

Следующим шагом требуется определить итоговые весы значений, с которыми будет производиться бинарная оценка значений массива, а затем преобразовать данные в бинарную строку.

Работа алгоритма генерации псевдослучайных чисел

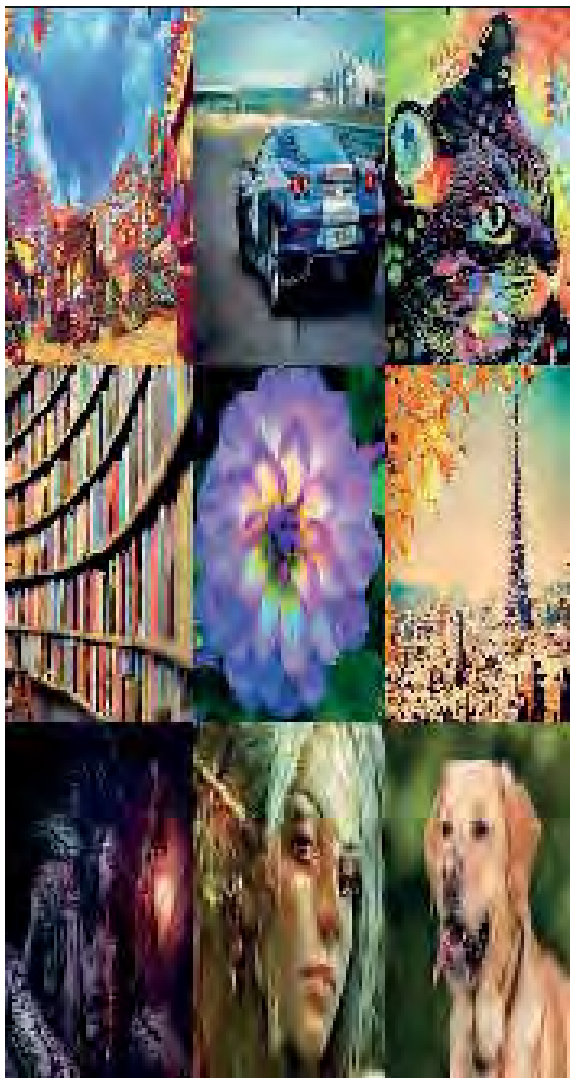
Работа данного алгоритма основана на обработке изображений. Для его эффективности и устойчивости к анализу следует использовать большой объем изображений, однако для примера работы алгоритма применялась случайная выборка, состоящая из девяти изображений (таблица).

■ Исходные размеры изображений
■ Initial sizes of pictures

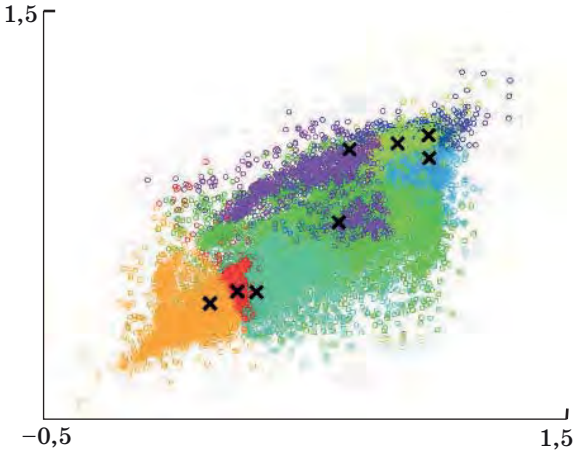
Образец изображения	Размер
1	1175×1680×3
2	223×226×3
3	183×275×3
4	720×960×3
5	183×275×3
6	252×200×3
7	1200×1920×3
8	781×1100×3
9	672×896×3

В первую очередь, следует определить индексы изображений с помощью алгоритма вихрь Мерсенна. Затем выполнить изменение размера изображений и объединение их в один трехмерный массив. Данные процессы возможно реализовать различными алгоритмами. В настоящей статье использованы методы, предложенные системой математических вычислений Octave [20]. Все изображения сначала приводятся к размеру 128×128×3, а затем склеиваются в изображение размера 384×384×3. Результат начального этапа показан на рис. 1.

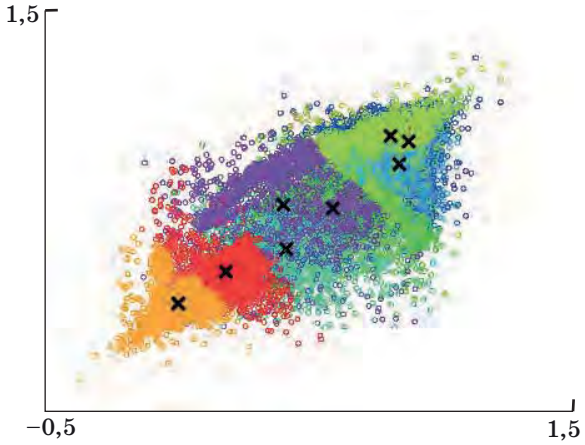
На следующем этапе производится инициализация восьми центроидов с использованием алгоритма вихрь Мерсенна. После инициализации центроидов проводится кластеризация методом K-средних. Оптимальным по соотноше-



■ ***Рис. 1.*** Склеенные и сжатые изображения
■ ***Fig. 1.*** Joint and compressed images



■ ***Рис. 2.*** Инициализированные центроиды
■ ***Fig. 2.*** Initialized centroids



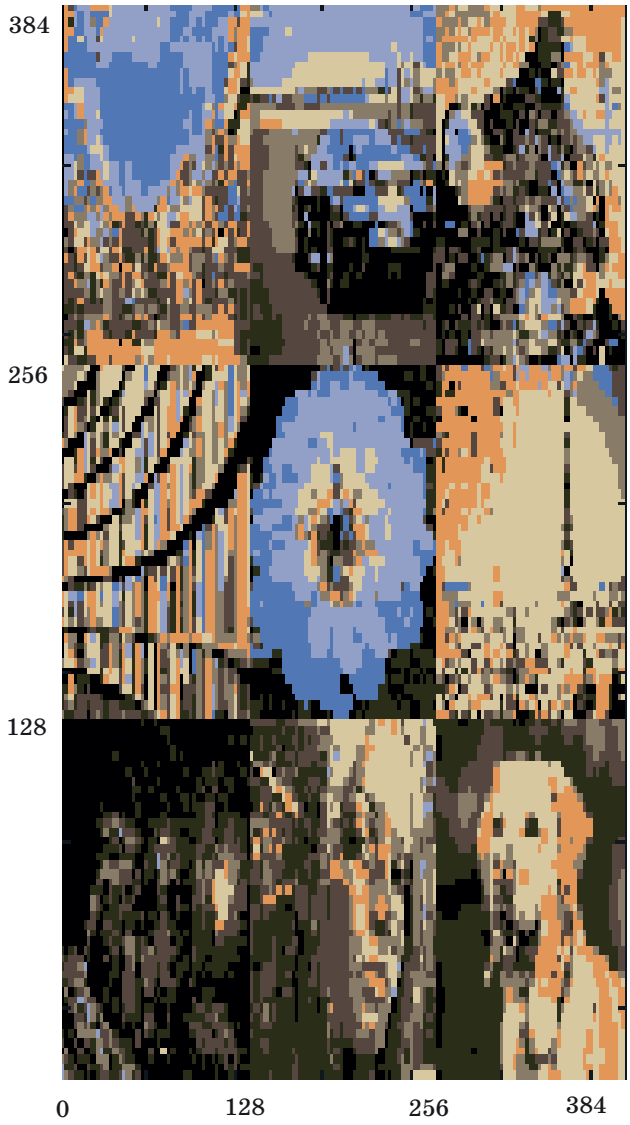
■ ***Рис. 3.*** Итоговое положение центроидов
■ ***Fig. 3.*** Final centroids positions

нию скорости и кластеризации данных количеством итераций предлагается не более четырех. Начальное положение центроидов показано на рис. 2.

Центроиды после четырех итераций процесса кластеризации (рис. 3) разделили все пиксели нашего изображения на восемь больших кластеров. В зависимости от исходного положения центроидов и количества итераций, которые произведет алгоритм, в каждом из кластеров будут разные пиксели.

После определения кластеров пикселям присваиваются значения, соответствующие ближайшим кластерным точкам (рис. 4).

После получения обновленных данных они преобразуются логистической сигмоидой (*) и объединяются в числовую последовательность. На выходе получается поток длиной 6144 Б, го-



■ ***Рис. 4.*** Визуализация действия *K*-метода
■ ***Fig. 4.*** Visualization of *K*-means algorithm

товый для использования в криптографических алгоритмах.

Заключение

В статье рассмотрен один из методов применения алгоритма вихрь Мерсенна для создания числовых последовательностей. В качестве источника образования зерен использованы графические объекты. Они претерпели изменения методом *K*-средних для кластеризации значений. В основе метода *K*-средних также находятся инициализированные алгоритмом вихрь Мерсенна центроиды кластера. Полученная в результате действия данного алгоритма числовая последовательность

может быть задействована в качестве компонентов работы криптографических алгоритмов пользовательских приложений, ориентированных на обмен тестовыми сообщениями и совместное использование медиафайлов. Полученный алгоритм, несмотря на относительно малую скорость, обладает относительной устойчивостью благодаря использованию уникальных изображений и двукратному применению алгоритма вихрь Мерсенна.

Литература

1. **Rukhin A.** A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. — Gaithersburg, MD: NIST, 2010. — 131 p.
2. **McAndrew A.** Introduction to Cryptography with Open-Source Software. — Boca Raton, FL: CRC Press, 2016. — 461 p.
3. **Menezes A.** Handbook of Applied Cryptography. — Boca Raton, FL: CRC Press, 1996. — 780 p.
4. **Matsumoto M., Nishimura T.** Mersenne Twister: a 623-dimensionally Equidistributed Uniform Pseudo-Random Number Generator // Journal ACM Transactions on Modeling and Computer Simulation (TOMACS). 1998. Vol. 8. Iss. 1. P. 3–30.
5. Mersenne Twister Home Page. http://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/emt.html. (дата обращения: 21.10.2017).
6. Cleve's Corner: Cleve Moler on Mathematics and Computing. http://blogs.mathworks.com/cleve/2015/04/17/random-number-generator-mersenne-twister (дата обращения: 20.04.2017).
7. **Schroeder M.** Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise. — N. Y.: W H Freeman and Company, 1991. — 429 p.
8. On-Line Encyclopedia of Integer Sequences. http://oeis.org/A000668 (дата обращения: 10.11.2017).
9. **Schroeder M.** Number Theory in Science and Communication. — Berlin: Springer-Verlag, 2009. — 431 p.
10. **Балонин Н. А., Сергеев М. Б.** Матрицы Мерсенна и Адамара //Информационно-управляющие системы. 2016. № 1. С. 2–14. doi:10.15217/issn1684-8853.2016.1.2
11. Octave-forge. https://octave.sourceforge.io/octave/function/randperm.html (дата обращения: 20.10.17).
12. Python Standart Library. https://docs.python.org/3/library/random.html (дата обращения: 20.10.2017).
13. Class MersenneTwister. http://commons.apache.org/proper/commons-math/javadocs/api-3.3/org/apache/commons/math3/random/MersenneTwister.html (дата обращения: 20.10.2017).
14. Haskell Documentation. https://hackage.haskell.org/package/mersenne-random-1.0.0.1/docs/System-Random-Mersenne.html (дата обращения: 20.10.2017).
15. **Bishop C. M.** Pattern Recognition and Machine Learning. — Berlin: Springer, 2006. — 738 p.
16. **Szeliski R.** Computer Vision: Algorithms and Applications. — Berlin: Springer, 2010. — 812 p.
17. **Cord M., Cunningham P.** Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval. — Berlin: Springer Science & Business Media, 2008. — 289 p.
18. **Petrosino A.** Progress in Image Analysis and Processing. — Berlin: Springer, 2013. — 766 p.
19. CS 229 Machine Learning Course Materials. http://cs229.stanford.edu/materials.html (дата обращения: 25.10.2017).
20. **Quarteroni A., Saleri F., Gervasio P.** Scientific Computing with MATLAB and Octave. — Berlin: Springer Science & Business Media, 2014. — 450 p.

UDC 004.421.5
doi:10.15217/issn1684-8853.2018.2.2

Pseudo Number Generation based on Graphic Object Processing

Kryukov D. A.[a], PhD, Tech., dm.bk@bk.ru
Tesker I. A.[a], Master Student, teskerivan.t@gmail.com
[a]Moscow Technological University, 78, Vernadskogo Av., 119456, Moscow, Russian Federation

Introduction: Data are often protected by encryption with algorithms using random or pseudo-random numbers. These numbers, which critically affect the cryptographic strength of the cipher, are created by generators of random or pseudorandom numbers. In order to use the generated bits in crypto algorithms, they must have the characteristics of a truly random sequence, i.e. be resistant to reverse engineering of the entire flow of generated numbers by its known fragments. **Purpose:** Developing a method for generating pseudo-random numbers suitable for further use in cryptographic algorithms. **Results:** The paper presents a method for generating pseudo-random numbers based on the K-method and Mersenne twister. As a source for generating pseudo-random numbers, we propose to use random graphic objects. An example is given, visualizing the results of the method. The novelty of our approach is that the entropy of a pseudo-random value is increased by using transformed information about graphic objects with enhanced personalization. The resulting algorithm has a relatively low speed, but it is highly stable due to the use of unique images and double use of Mersenne

twister algorithm. **Practical relevance:** The developed method of generating pseudo-random numbers can be used in crypto algorithms, including those operating with large amounts of data.

Keywords — Cryptography, Pseudo Random Number Generator, K-method, Image Processing Clustering, Mersenne Twister.

Citation: Kryukov D. A., Tesker I. A. Pseudo Number Generation based on Graphic Object Processing. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2018, no. 2, pp. 2–7 (In Russian). doi:10.15217/issn1684-8853.2018.2.2

References

1. Rukhin A. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. Gaithersburg, MD, NIST, 2010. 131 p.
2. McAndrew A. *Introduction to Cryptography with Open-Source Software*. Boca Raton, FL, CRC Press, 2016. 461 p.
3. Menezes A. *Handbook of Applied Cryptography*. Boca Raton, FL, CRC Press, 1996. 780 p.
4. Matsumoto M., Nishimura T. Mersenne Twister: a 623-dimensionally Equidistributed Uniform Pseudo-Random Number Generator. *Journal ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 1998, vol. 8, iss. 1, pp. 3–30.
5. *Mersenne Twister Home Page*. Available at: http://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/emt.html (accessed 21 October 2017).
6. *Cleve's Corner: Cleve Moler on Mathematics and Computing*. Available at: http://blogs.mathworks.com/cleve/2015/04/17/random-number-generator-mersenne-twister (accessed 20 April 2017).
7. Schroeder M. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. New York, W H Freeman and Company, 1991. 429 p.
8. *On-Line Encyclopedia of Integer Sequences*. Available at: http://oeis.org/A000668 (accessed 10 November 2017).
9. Schroeder M. *Number Theory in Science and Communication*. Berlin, Springer-Verlag, 2009. 431 p.
10. Balonin N. A., Sergeev M. B. Mersenne and Hadamard Matrices *Informatsionno-upravliaiushchie systemy* [Information and Control Systems], 2016, no. 1, pp. 2–14 (In Russian). doi:10.15217/issn1684-8853.2016.1.2
11. *Octave-forge*. Available at: https://octave.sourceforge.io/octave/function/randperm.html (accessed 20 October 2017).
12. *Python Standart Library*. Available at: https://docs.python.org/3/library/random.html (accessed 20 October 2017).
13. *Class MersenneTwister*. Available at: http://commons.apache.org/proper/commons-math/javadocs/api-3.3/org/apache/commons/math3/random/MersenneTwister.html (accessed 20 October 2017).
14. *Haskell Documentation*. Available at: https://hackage.haskell.org/package/mersenne-random-1.0.0.1/docs/System-Random-Mersenne.html (accessed 20 October 2017).
15. Bishop C. M. *Pattern Recognition and Machine Learning*. Berlin, Springer, 2006. 738 p.
16. Szeliski R. *Computer Vision: Algorithms and Applications*. Berlin, Springer, 2010. 812 p.
17. Cord M., Cunningham P. *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval*. Berlin, Springer Science & Business Media, 2008. 289 p.
18. Petrosino A. *Progress in Image Analysis and Processing*. Berlin, Springer, 2013. 766 p.
19. *CS 229 Machine Learning Course Materials*. Available at: http://cs229.stanford.edu/materials.html (accessed 25 October 2017).
20. Quarteroni A., Saleri F., Gervasio P. *Scientific Computing with MATLAB and Octave*. Berlin, Springer Science & Business Media, 2014. 450 p.