УДК 519.21

А. С. ВАМБОЛЬ

Национальный аэрокосмический университет им. Н. Е. Жуковского «ХАИ», Украина

ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ МАЖОРАНТО-СУПЕРПОЗИЦИОННОГО АЛГОРИТМА ГЕНЕРАЦИИ РОБАСТНОГО РАСПРЕДЕЛЕНИЯ СОЛИТОНА

В статье представлены результаты аналитического и экспериментального сравнительного исследования производительности мажорантно-суперпозиционного алгоритма генерации робастного распределение солитона и стандартного алгоритма генерации дискретной случайной величины, применённого для этого распределения. Результаты исследования демонстрируют преимущество в производительности мажорантно-суперпозиционного алгоритма, обладающего амортизационной сложностью O(1), по сравнению со стандартным алгоритмом, имеющим амортизационную сложность O(log K), при любых значениях параметров данного распределения с основным параметром K. Мажорантно-суперпозиционный алгоритм может быть рекомендован к использованию в программной реализации кодера кодов LT.

Ключевые слова: *мажорантно-суперпозиционный алгоритм, робастное распределение солитона, коды LT, фонтанные коды, исследование производительности, амортизационная сложность, генерация случайных величин.*

Введение

Коды LT являются исторически первым классом фонтанных кодов – нового класса помехоустойчивых кодов, позволяющих закодировать любое сообщение конечного размера потенциально неограниченным потоком независимых кодовых символов, что принципиально отличает их от классических блоковых или свёрточных кодов. Эти коды имеют простые алгоритмы декодирования и позволяют на практике получать результаты, близкие к предельным возможностям помехоустойчивого кодирования [1]. Коды LT находят широкое практическое применение для хранения информации на цифровых носителях и широковещании в компьютерных сетях [2].

Отличительной особенностью кодов LT является использование робастного распределения солитона в качестве закона распределения степени кодового символа – количества исходных символов, участвующих в формировании данного кодового символа. Преимущества данных кодов обусловлены этой особенностью, обеспечивающей вычислительную сложность их алгоритмов кодирования и декодирования, составляющую O(K log K), где K – количество исходных символов [2].

Поскольку генерация робастного распределения солитона является неотъемлемой частью алгоритма кодирования фонтанных кодов LT, проблема создания высокопроизводительных алгоритмов генерации дискретной случайной величины, подчиняющейся данному распределению, является актуальной.

Мажорантно-суперпозиционный алгоритм генерации робастного распределения солитона, предложенный в [3], демонстрирует преимущество в производительности по сравнению со стандартным алгоритмом генерации дискретной случайной величины, применённым для данного распределения. Однако сравнительное исследование производительности данных алгоритмов, приведённое в [3], не содержит аналитической составляющей, являясь исключительно экспериментальным. Кроме того, данное исследование было выполнено только для одной пары значений дополнительных параметров робастного распределения солитона, представленной $c = 0.2$ и $\delta = 0.05$. Отсюда следует актуальность более глубокого сравнительного исследования производительности этих алгоритмов.

Целью статьи является аналитическое и экспериментальное сравнительное исследование производительности мажорантно-суперпозиционного алгоритма генерации робастного распределение солитона и стандартного алгоритма генерации дискретной случайной величины, применённого для этого распределения.

1. Стандартный и мажорантно-суперпозиционный алгоритмы

Робастное распределение солитона является трёхпараметрическим и имеет следующий вид:

$$\mu(d) = \frac{\rho(d) + \tau(d)}{Z},$$

где $Z = \sum_{d=1}^{K} \left(\rho(d) + \tau(d)\right)$,

$$\rho(d) = \begin{cases} \frac{1}{K} & \text{для } d = 1, \\ \frac{1}{d(d-1)} & \text{для } d = 2, 3, \ldots, K. \end{cases}$$

$$\tau(d) = \begin{cases} \frac{S}{K}\frac{1}{d} & \text{для } d = 1, 2, \ldots, \left(\frac{K}{S}\right) - 1, \\ \frac{S}{K}\ln\left(\frac{S}{\delta}\right) & \text{для } d = \frac{K}{S}, \\ 0 & \text{для } \frac{K}{S} < d \le K. \end{cases}$$

где

$$S = c \cdot \ln\left(\frac{K}{\delta}\right)\sqrt{K}.$$

Параметрами данного распределения являются:

K – основной параметр, равный количеству исходных символов;

δ – верхняя граница вероятности сбоя процесса декодирования при условии получения KZ кодовых символов;

c – константа робастного распределения солитона [2].

Математическое ожидание данного распределения – величина порядка O(log K) [4].

Стандартный алгоритм генерации дискретной случайной величины, применённый для робастного распределения солитона, представлен на рис. 1 [3].

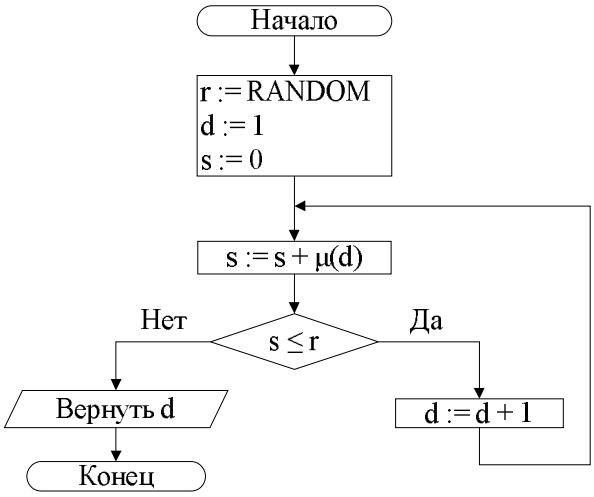

Рис. 1. Стандартный алгоритм генерации робастного распределения солитона

Суть этого алгоритма заключается в разбиении единичного отрезка на интервалы, длины которых равны вероятностям появления соответствующих значений случайной величины с распределением μ(d). Для получения нового значения генерируется равномерно распределённая точка на данном отрезке, производится поиск интервала, в который она попала, и возвращается значение, соответствующее данному интервалу [3].

Мажорантно-суперпозиционный алгоритм генерации робастного распределения солитона представлен на рис. 2, состоящем из трёх частей [3].

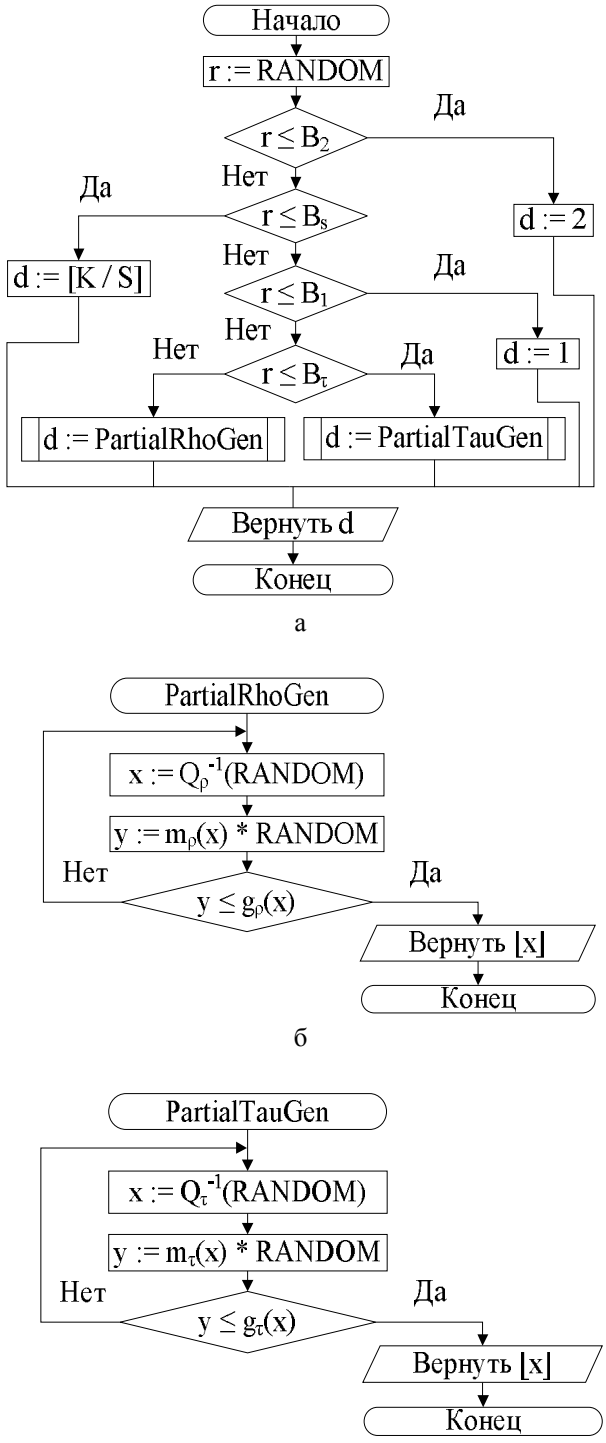

Рис. 2. Мажорантно-суперпозиционный алгоритм

Параметры этого алгоритма совпадают с параметрами робастного распределения солитона. Функции и переменные, используемые в данном алгоритме, определяются следующим образом [3]:

$$g_\rho(x)=\frac{1}{\lfloor x\rfloor(\lfloor x\rfloor-1)},\qquad g_\tau(x)=\frac{1}{\lfloor x\rfloor},$$

$$m_\rho(x)=\frac{1}{(x-1)(x-2)},\qquad m_\tau(x)=\frac{1}{x-1},$$

$$Q_\rho^{-1}(x)=1+\frac{2}{2-U^x},\qquad Q_\tau^{-1}(x)=1+2V^x,$$

$$U=2-\frac{2}{K},\qquad V=\frac{1}{2}\left\lceil\frac{K}{S}\right\rceil-0.5,$$

$$B_2=\frac{S+K}{2KZ},\qquad B_S=B_2+\frac{S}{KZ}\ln\left(\frac{S}{\delta}\right),$$

$$B_1=B_S+\frac{1+S}{KZ},\qquad B_\tau=B_1+\frac{S}{KZ}\sum_{d=3}^{\left\lceil\frac{K}{S}\right\rceil-1}\frac{1}{d}.$$

Функции PartialRhoGen и PartialTauGen предназначены для генерации распределений $f_\rho(d)$ и $f_\tau(d)$ при помощи совмещения мажорантного метода исключения и метода обратного преобразования [3].

$$f_\rho(d)=\begin{cases}0 & \text{для } d=1,2;\\ \dfrac{1}{(1-B_\tau)Z}\dfrac{1}{d(d-1)} & \text{для } d=3,\ldots,K.\end{cases}$$

$$f_\tau(d)=\begin{cases}0 & \text{для } d=1,2,\left\lceil\dfrac{K}{S}\right\rceil,\ldots,K;\\ \dfrac{S}{(B_\tau-B_1)KZ}\dfrac{1}{d} & \text{для } d=3,\ldots,\left\lceil\dfrac{K}{S}\right\rceil-1.\end{cases}$$

Алгоритм работы данных функций заключается в генерации равномерно распределённых случайных точек в области A, ограниченной осью OX, графиком функции m(x) и прямыми x = a и x = b, и возвращении в качестве результатов целых частей координат x точек, оказавшихся под графиком функции g(x). Если сгенерированная точка оказалась над графиком g(x), осуществляется повторная генерация случайной точки. Функция m(x) является мажорантой для g(x) на интервале [a; b), $a_\rho = a_\tau = 3$, $b_\rho = K + 1$, $b_\tau = [K / S]$. Таким образом, функции PartialRhoGen и PartialTauGen содержат циклы [3].

2. Аналитическое сравнительное исследование производительности

Амортизационная сложность стандартного алгоритма генерации дискретной случайной величины составляет O(m), где m – математическое ожидание номера i значения генерируемой случайной величины x_i [5]. Таким образом, генерация робастного распределения солитона при помощи этого алгоритма имеет амортизационную сложность O(log K). Затраты памяти для данного алгоритма составляют O(1).

Среднее количество итераций цикла для функций PartialRhoGen и PartialTauGen, изображённых на рис. 2б и рис. 2в, равно отношению площади области A к площади её части, находящейся под графиком функции g(x), поскольку генерируемая точка распределена равномерно в области A, а условием завершения цикла является попадание сгенерированной точки в область под графиком функции g(x). Площади области A для функций PartialRhoGen и PartialTauGen можно вычислить по этим формулам:

$$G_\rho=\int_{a_\rho}^{b_\rho+1}m_\rho(x)dx=\int_3^{K+1}\frac{dx}{(x-1)(x-2)}=\ln\left(2-\frac{2}{K}\right),$$

$$G_\tau=\int_{a_\tau}^{b_\tau+1}m_\tau(x)dx=\int_3^{\left\lceil\frac{K}{S}\right\rceil}\frac{dx}{x-1}=\ln\left(\left\lceil\frac{K}{S}\right\rceil-1\right)-\ln(2).$$

Площади части области A, находящейся под графиком g(x), для функций PartialRhoGen и PartialTauGen могут быть представлены в виде суммы площадей столбцов единичной ширины, высоты которых равны g(a), g(a + 1), ..., g(b). Для вывода формул этих площадей уместно воспользоваться формулой Эйлера для частичной суммы гармонического ряда, согласно которой

$$\sum_{k=1}^{n}\frac{1}{k}\approx\ln(n)+\gamma,$$

где $\gamma \approx 0.577$ является постоянной Эйлера-Маскерони [6]. Таким образом, формулы для данных площадей имеют следующий вид:

$$D_\rho=\sum_{x=a_\rho}^{b_\rho}g_\rho(x)=\sum_{x=3}^{K}\frac{1}{\lfloor x\rfloor(\lfloor x\rfloor-1)}=\sum_{x=2}^{K-1}\frac{1}{x}-\sum_{x=3}^{K}\frac{1}{x}=$$

$$=\left(1-\frac{1}{K}\right)-0.5,$$

$$D_\tau=\sum_{x=a_\tau}^{b_\tau}g_\tau(x)=\sum_{x=3}^{\left\lceil\frac{K}{S}\right\rceil-1}\frac{1}{\lfloor x\rfloor}\approx\ln\left(\left\lceil\frac{K}{S}\right\rceil-1\right)+\gamma-1.5\approx$$

$$\approx\ln\left(\left\lceil\frac{K}{S}\right\rceil-1\right)-1.$$

Пусть

$$H_\rho=2-\frac{2}{K},\qquad H_\tau=\ln\left(\left\lceil\frac{K}{S}\right\rceil-1\right),$$

тогда формулы среднего количества итераций цикла для функций PartialRhoGen и PartialTauGen имеют следующий вид:

$$L_\rho=\frac{G_\rho}{D_\rho}=\frac{2\ln(H_\rho)}{H_\rho-1},\qquad L_\tau=\frac{G_\tau}{D_\tau}\approx1+\frac{1-\ln(2)}{H_\tau-1}.$$

Значение H_ρ возрастает при увеличении K для любого положительного K, асимптотически приближаясь к 2. Поскольку минимальное значение K, используемое в функции PartialRhoGen, равно 3, знаменатель формулы L_ρ имеет значение в пределах интервала [⅓; 1). Числитель формулы L_ρ возрастает с увеличением K асимптотически приближаясь к ln(4), поэтому его значения находятся в интервале [ln(16 / 9); ln(4)). Таким образом, для любого допустимого K значение L_ρ не может превысить константу 6ln(2) ≈ 4.159. Отсюда следует, что амортизационная сложность выполнения функции PartialRhoGen составляет O(1).

Минимальное значение [K / S], используемое в функции PartialTauGen, равно 4, следовательно, минимальное значение H_τ составляет ln(3). Для значений H_τ, превышающих 1, L_τ уменьшается при увеличении H_τ, асимптотически приближаясь к 1. Следовательно, для любого допустимого K значение L_τ не может превысить константу $\log_{(3 / e)}(1.5) \approx 4.112$. Таким образом, амортизационная сложность выполнения функции PartialTauGen составляет O(1).

Мажорантно-суперпозиционный алгоритм генерации робастного распределения солитона, схематически изображённый на рис. 2, имеет 5 ветвей, выполнение трёх из которых заключается в возвращении констант, а две остальные состоят из однократного вызова функции PartialRhoGen или PartialTauGen, амортизационная сложность которых составляет O(1). Отсюда следует, что амортизационная сложность мажорантно-суперпозиционого алгоритма генерации робастного распределения составляет O(1). Затраты памяти для данного алгоритма также составляют O(1).

Таким образом, мажорантно-суперпозиционный алгоритм, обладая константной амортизационной сложностью, превосходит в производительности стандартный алгоритм генерации робастного распределения солитона, имеющий логарифмическую амортизационную сложность.

3. Экспериментальное сравнительное исследование производительности

Экспериментальное сравнительное исследование производительности мажорантно-суперпозиционного алгоритма генерации робастного распределения солитона и стандартного алгоритма генерации дискретной случайной величины, применённого для данного распределения, было выполнено путём измерения времени генерации 10^7 случайных значений при помощи реализаций данных алгоритмов на языке C# для заданных значений основного параметра робастного распределения солитона K при фиксированных значениях дополнительных параметров c и δ. В качестве реализации стандартного алгоритма была использована реализация генератора робастного распределения солитона в рамках проекта по разработке кодера и декодера кодов LT на языке C# с открытым исходным кодом [7]. Измерения осуществлялись на компьютере с процессором Intel Core i5-2430M 2.4GHz. На основании полученных измерений были построены графики зависимости времени генерации 10^7 случайных значений от основного параметра K.

На графике, изображённом на рис. 3, представлены результаты измерений при c = 0.1 и трёх различных значениях параметра δ. График, представленный на рис. 4, построен на основании результатов измерений, выполненных при δ = 0.1 и трёх различных значениях параметра c.

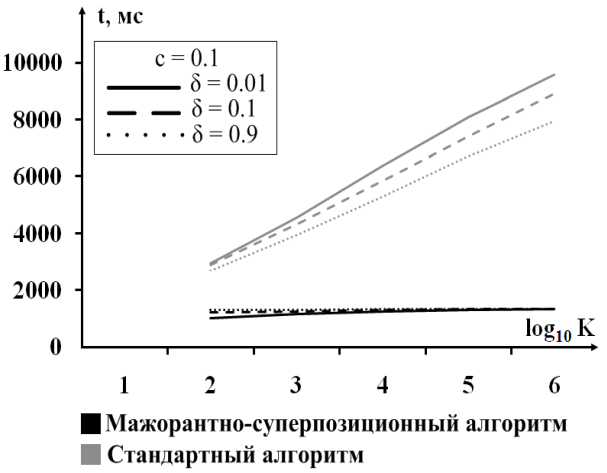

Рис. 3. График зависимости времени генерации 10^7 случайных значений от параметра K для сравниваемых алгоритмов при c = 0.1

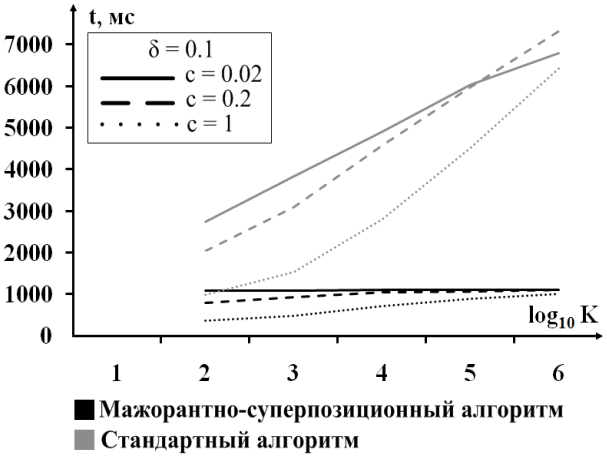

Рис. 4. График зависимости времени генерации 10^7 случайных значений от параметра K для сравниваемых алгоритмов при δ = 0.1

Результаты эксперимента свидетельствуют о более высокой производительности мажорантно-суперпозиционного алгоритма по сравнению со стандартным алгоритмом генерации робастного распределения солитона при всех значениях параметров данного распределения, рассмотренных в ходе эксперимента, что согласуется с результатами аналитического сравнительного исследования.

Для мажорантно-суперпозиционного алгоритма графики, представленные на рис. 3 и рис. 4, стремятся к асимптоте, параллельной оси абсцисс, что подтверждает выводы аналитического исследования производительности данного алгоритма о равенстве его амортизационной сложности O(1).

Заключение

Результаты аналитического и экспериментального сравнительного исследования производительности мажорантно-суперпозиционного и стандартного алгоритмов генерации робастного распределение солитона демонстрируют преимущество в производительности первого, обладающего амортизационной сложностью O(1), по сравнению со вторым, имеющим амортизационную сложность O(log K), при любых значениях параметров данного распределения с основным параметром K. Мажорантно-суперпозиционный алгоритм может быть рекомендован к использованию в программной реализации кодера кодов LT.

Литература

1. Жураковський, Б. Ю. Дослідження використання нових завадостійких кодів для каналів зі стиранням [Текст] / Б. Ю. Жураковський // Вісник Державного університету інформаційно-комунікаційних технологій. – 2012. – Т. 10, № 2. – С. 93-96.

2. MacKay, D. J. C. Fountain Codes [Text] / D. J. C. MacKay // IEE Proceedings: Communications. – 2005. – vol. 152, No. 6. – P. 1062-1068.

3. Вамболь, А. С. Мажоранто-суперпозиционный алгоритм генерации случайной величины, имеющей робастное распределение солитона [Текст] / А. С. Вамболь // Системи обробки інформації. – 2016. – № 2. – С. 83-88.

4. Du Toit, J. A Practical Implementation of Fountain Codes over WiMAX Networks with an Optimised Probabilistic Degree Distribution [Text] / Jaco du Toit, Riaan Wolhuter // ICSNC 2011: The Sixth International Conference on Systems and Networks Communications. – 2011. – P. 32-37.

5. Войтишек, А. В. Дополнительные сведения о численном моделировании случайных элементов. Учеб. пособие [Текст] / А. В. Войтишек // Новосибирский государственный университет. – 2007. – 92 с.

6. Fikhtengol'ts, G. M The Fundamentals of Mathematical Analysis: International Series of Monographs in Pure and Applied Mathematics [Text] / G. M. Fikhtengol'ts, I. N. Sneddon, M. Stark, S. Ulam // Elsevier. - 2014. – vol. 2 (73). – 540 p.

7. Lapa, C. Soliton.cs [Electronic resource] / Chris Lapa. – Access mode: https://github.com/GusBricker/FountainCodes/blob/master/LubyTransform/Distributions/Soliton.cs. – 10.03.2016.

References

1. Zhurakovs'kyy, B. Yu. Doslidzhennya vykorystannya novykh zavadostiykykh kodiv dlya kanaliv zi styrannyam [Research of the use of new antijamming codes for channels with elimination]. *Visnyk Derzhavnoho universytetu informatsiyno-komunikatsiynykh tekhnolohiy – Bulletin of State University of Information and Communication Technologies*, 2012, vol. 10, no. 2, pp. 93-96.

2. MacKay, D. J. C. Fountain Codes. *IEE Proceedings: Communications,* 2005, vol. 152, no. 6, pp. 1062-1068.

3. Vambol', A. S. Mazhoranto-superpozicionnyj algoritm generacii sluchajnoj velichiny, imejushhej robastnoe raspredelenie solitona [Majorant-superposition algorithm for generating a random variable that has a robust soliton distribution]. *Systemy obrobky informatsiyi – Information processing systems*, 2016, no. 2, pp. 83-88.

4. Du Toit, J., Wolhuter, R. A Practical Implementation of Fountain Codes over WiMAX Networks with an Optimised Probabilistic Degree Distribution. *ICSNC 2011: The Sixth International Conference on Systems and Networks Communications*, 2011, pp. 32-37.

5. Vojtishek, A. V. *Dopolnitel'nye svedenija o chislennom modelirovanii sluchajnyh jelementov. Ucheb. posobie* [Additional information about numerical modeling of random elements. Tutorial]. *Novosibirskij gosudarstvennyj universitet – Novosibirsk State University,* 2007. 92 p.

6. Fikhtengol'ts, G. M., Sneddon, I. N., Stark, M., Ulam, S. *The Fundamentals of Mathematical Analysis: International Series of Monographs in Pure and Applied Mathematics.* Elsevier, 2014, vol. 2 (73). 540 p.

7. Lapa, C. *Soliton.cs.* Available at: https://github.com/GusBricker/FountainCodes/blob/master/LubyTransform/Distributions/Soliton.cs (accessed 10.03.2016).

Поступила в редакцию 15.03.2016, рассмотрена на редколлегии 14.04.2016

ДОСЛІДЖЕННЯ ПРОДУКТИВНОСТІ МАЖОРАНТНО-СУПЕРПОЗИЦІЙНОГО АЛГОРИТМУ ГЕНЕРАЦІЇ РОБАСТНОГО РОЗПОДІЛУ СОЛІТОНА

О. С. Вамболь

У статті представлено результати аналітичного та експериментального порівняльного дослідження продуктивності мажорантно-суперпозиційного алгоритму генерації робастного розподілу солітона і стандартного алгоритму генерації дискретної випадкової величини, застосованого для цього розподілу. Результати дослідження демонструють перевагу в продуктивності мажорантно-суперпозиційного алгоритму, що має амортизаційну складність O(1), у порівнянні зі стандартним алгоритмом, який має амортизаційну складність O(log K), при будь-яких значеннях параметрів даного розподілу з основним параметром K. Мажорантно-суперпозиційний алгоритм може бути рекомендований до використання в програмній реалізації кодера кодів LT.

Ключові слова: мажорантно-суперпозиційний алгоритм, робастний розподіл солітона, коди LT, фонтанні коди, дослідження продуктивності, амортизаційна складність, генерація випадкових величин.

INVESTIGATION OF PERFORMANCE OF MAJORANT-SUPERPOSITION ALGORITHM FOR GENERATING A ROBUST SOLITON DISTRIBUTION

A. S. Vambol

The results of analytical and experimental comparative investigation of performance of the majorant-superposition algorithm for generating a robust soliton distribution and the standard algorithm for generating a discrete random variable, applied to this distribution, are presented in the paper. The results of the investigation demonstrate the performance advantage of the majorant-superposition algorithm, whose amortized complexity is O(1), over the standard algorithm, which has amortized complexity O(log K), for any values of the parameters of this distribution with a key parameter K. The majorant-superposition algorithm can be recommended for use in a software implementation of the LT codes encoder.

Key words: majorant-superposition algorithm, robust soliton distribution, LT codes, fountain codes, performance investigation, amortized complexity, generating random variables.

Вамболь Алексей Сергеевич – магистр специализированных компьютерных систем, кафедра Компьютерных систем и сетей, Национальный аэрокосмический университет им. Н. Е. Жуковского «Харьковский авиационный институт», Харьков, Украина, e-mail: a.vambol@mail.ru.

Vambol Aleksei Sergeevich – Master of Specialized Computer Systems, Department of Computer Systems and Networks, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine, e-mail: a.vambol@mail.ru.