Научная статья
УДК 004.056:539.12.04
DOI: 10.26583/bit.2025.4.08

АЛГОРИТМ ГЕНЕРАЦИИ БЕНТ-ФУНКЦИЙ С ПОМОЩЬЮ ВЕЙВЛЕТ–ПРЕОБРАЗОВАНИЯ

Илья В. Шибакин[1], Алла Б. Левина[2]
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), ул. Профессора Попова, 5 литера Ф, Санкт-Петербург, 197022, Россия
[1]shibakin12@mail.ru, https://orcid.org/0009-0002-8692-7474
[2]alla_levina@mail.ru, https://orcid.org/0000-0003-4421-2411

Аннотация. Объект данного исследования – бент-функции (максимально нелинейные булевы функции), а также булевы функции с высокой нелинейностью. Предметом статьи является возможность применения вейвлет–преобразования для создания бент-функций. Настоящая работа рассматривает два существующих алгоритма генерации бент-функций, первый из которых использует модификацию таблицы истинности (ТИ) исходной функции, а второй – добавление подходящих слагаемых к алгебраической нормальной форме (АНФ) функции. Для обоих алгоритмов приведены теоретические положения, лежащие в их основе, описание принципа работы, а также их сильные и слабые стороны. Вейвлет–преобразование оказалось возможно применить для создания бент-функций. В работе представлен новый алгоритм генерации бент-функций, основанный на вейвлет–преобразовании и функции следа, выполнена его реализация на C++. Для алгоритма генерации бент-функций, основанного на вейвлет–преобразовании, проведена оценка временных затрат на создание бент-функций от аргументов различной длины, сравнение его с рассмотренными в статье известными алгоритмами (алгоритм, задействующий модификацию таблицы истинности; алгоритм, добавляющий слагаемые в АНФ). Алгоритм, изменяющий ТИ исходной функции, улучшает ее нелинейность, но результат его работы скорее всего не будет бент-функцией, а алгоритм, основанный на вейвлет–преобразовании и функции следа всегда генерирует бент-функции. Скорость генерации функций у алгоритма, добавляющего слагаемые в АНФ высока, но новый алгоритм, задействующий вейвлет–преобразование и функцию следа превосходит его в этом отношении. Генерируемые полученным алгоритмом функции, как и сам алгоритм, могут найти применение в теории кодирования, криптографии.

Ключевые слова: *криптография, бент-функции, вейвлет–преобразование, функция следа, генерация бент-функций.*

Для цитирования: *Шибакин, Илья В.; Левина, Алла Б. Алгоритм генерации бент-функций с помощью вейвлет–преобразования. Безопасность информационных технологий, [S.l.], т. 32, № 4, с. 106–121, 2025. ISSN 2074-7136. URL: https://bit.spels.ru/index.php/bit/article/view/1872. DOI: https://doi.org/10.26583/bit.2025.4.08.*

Благодарности. Работа выполнена в рамках государственного задания Министерства науки и высшего образования Российской Федерации № 075-00003-24-01 от 08.02.2024 (проект FSEE-2024-0003).

Scientific article

ALGORITHM FOR GENERATION OF BENT FUNCTIONS USING WAVELET TRANSFORM

Ilya V. Shibakin[1], Alla B. Levina[2]
Saint Petersburg Electrotechnical University "LETI", Professora Popova St., 5, Unit F, St. Petersburg, Russia, 197022

[1]*shibakin12@mail.ru, https://orcid.org/0009-0002-8692-7474*
[2]*alla_levina@mail.ru, https://orcid.org/0000-0003-4421-2411*

Abstract. The object of this study is bent functions (maximally nonlinear Boolean functions), as well as Boolean functions with high nonlinearity. The subject of the article is the possibility of using the wavelet transform to create bent functions. This work considers two existing algorithms for generation of bent functions, the first of which uses a modification of the truth table (TT) of the original function, and the second - adding suitable terms to the algebraic normal form (ANF) of the function. For both algorithms, the theoretical basis underlying them, a description of the working principle, as well as their strengths and weaknesses are given. The wavelet transform turned out to be possible to apply to create bent functions. In the paper, a new algorithm for generation of bent functions based on the wavelet transform and the trace function is presented, its implementation in C++ is performed. For the algorithm for generation of bent functions based on the wavelet transform, the time costs for creating bent functions from arguments of different lengths are estimated, and it is compared with the existing algorithms considered in the article (the algorithm that uses the modification of the truth table; the algorithm that adds terms to the ANF). The algorithm that changes the TT of the original function improves its nonlinearity, but the result of its work will most likely not be a bent function, and the algorithm based on the wavelet transform and the trace function always generates bent functions. The speed of generation of functions for the algorithm that adds terms to the ANF is high, but the new algorithm that uses the wavelet transform and the trace function surpasses it in this respect. The functions generated by the resulting algorithm, as well as the algorithm itself, can find application in coding theory and cryptography.

Keywords: *cryptography, bent functions, wavelet transform, trace function, bent function generation.*

For citation: *Shibakin, Ilya V.; Levina, Alla B. Algorithm for generation of bent functions using wavelet transform. IT Security (Russia), [S.l.], v. 32, no. 4, p 106–121, 2025. ISSN 2074-7136. URL: https://bit.spels.ru/index.php/bit/article/view/1872. DOI: https://doi.org/10.26583/bit.2025.4.08.*

Acknowledgement. The work was carried out within the framework of the state assignment of the Ministry of Science and Higher Education of the Russian Federation No. 075-00003-24-01 dated 08.02.2024 (project FSEE-2024-0003).

Введение

Американский математик Оскар Ротхаус ввел понятие бент-функций и начал их исследование в 60-х годах прошлого века [1]. В 70-х годах над изучением этой области работали Дж. Диллон [2] и Р.Л. МакФарланд [3], а к 80-м годам интерес к бент-функциям возрос, и их изучением занялись ученые-математики по всему миру.

Несмотря на значительный прогресс, в исследовании бент-функций все еще существует множество открытых вопросов, перечислим некоторые из них:

1) В настоящее время существует только асимптотическая оценка числа бент функций от n переменных, что затрудняет дальнейшее теоретическое развитие используемого ими математического аппарата, имеющего множество приложений как теоретических, так и практических в различных областях (в том числе в криптографии, криптоанализе, теории кодирования).

2) Сложность представляет разработка новых конструкций, эффективных алгоритмов порождения бент-функций, являющихся, как уже было упомянуто, полезным видом булевых функций с практической точки зрения.

3) Разработка методов построения уравновешенных функций с высокой нелинейностью из бент-функций также является важным вопросом, так как в криптографических приложениях от булевых функций помимо нелинейности часто требуется уравновешенность.

Ряд существующих проблем далеко не исчерпывается перечисленными [4].

В [5–7] рассматривается использование вейвлет–преобразования и бент-функций для создания *R*-надежных кодов (то есть кодов, для которых R – максимальное число кодовых слов кода, при наложении на которые фиксированной ошибки ($e \neq 0$) получаются кодовые слова).

Авторы в [5–7] предлагают использовать результат вейвлет–преобразования (элементы вейвлетного потока, полученные путем выбрасывания из статической или заданной на основе информационной части кодового слова сетки в качестве части аргументов фиксированных бент-функций различной степени нелинейности). Вычисленные значения этих бент-функций составляют в зависимости от конкретного метода всю дополнительную часть кодового слова или ее фрагмент.

Приведем конкретные примеры применения бент-функций в области криптографии.

Бент-функции являются основой построения *S*-блоков (*S*-box), являющихся частью многих симметричных шифров. На основе бент-функций построены, в частности, *S*-блоки стандарта AES и канадского шифра CAST [4]. С точки зрения математики *S*-блок представляет из себя векторную булеву функцию, имеющую n входных и m выходных бит.

Качества булевых функций, используемых для построения *S*-блоков важны для их криптографической стойкости. Например, *S*-блоки шифра DES были уязвимы для линейного криптоанализа в силу низкой нелинейности функций, использованных в них [4, 8].

Построение *S*-блоков не является единственным применением бент-функций в криптографии. Согласно [4] бент-функции применяются также в хэш-функциях, псевдослучайных генераторах [9] и схемах аутентификации [10].

Бент-функции востребованы, но их доля в общей массе булевых функций невелика. Построение алгоритмов, позволяющих получать функции с высокой нелинейностью, или бент-функции представляется актуальным.

В данной работе рассмотрено два существующих алгоритма генерации бент-функций, первый из которых использует модификацию таблицы истинности исходной функции, второй – добавление походящих слагаемых к алгебраической нормальной форме (АНФ) функции. В работе представлен новый алгоритм построения бент-функций, основанный на вейвлет–преобразовании и функции следа.

1. Алгоритмы генерации булевых функций с высокой нелинейностью

1.1 Модификация таблицы истинности булевой функции

Для описания алгоритмов, приведенных в этом и последующих разделах, понадобятся определения и обозначения, приведенные ниже.

Под таблицей истинности (ТИ) следует понимать таблицу, строка которой представляют собой двоичный вектор, являющийся конкатенацией аргумента $\boldsymbol{x} \in \mathbb{Z}_2^n$ ($\mathbb{Z}_2^n$ – множество двоичных векторов длины n) и соответствующего значения $f(\boldsymbol{x})$ булевой функции. Строки в таблице истинности идут в порядке возрастания аргумента. Всего строк 2^n.

Преобразованием Уолша—Адамара булевой функции f от n переменных называется целочисленная функция W_f , заданная на множестве $\mathbb{Z}_2^n$ равенством

$$W_f(\boldsymbol{v}) = \sum_{\boldsymbol{u} \in \mathbb{Z}_2^n} (-1)^{(\boldsymbol{u}\cdot\boldsymbol{v}) \oplus f(\boldsymbol{u})},$$

где $\boldsymbol{v} \in \mathbb{Z}_2^n$.

Линейная булева функция, задаваемая вектором $\boldsymbol{v}$:

$$L_{\boldsymbol{v}}(x) = v_1 x_1 + \cdots + v_n x_n.$$

Нелинейность N_f произвольной функции f, выраженная через коэффициенты Уолша-Адамара

$$N_f \;=\; 2^{n-1} \;-\frac{1}{2}\max_{\boldsymbol{v}\in\mathbb{Z}_2^n}|W_f\,(\boldsymbol{v})|\;.$$

Нелинейность бент-функций максимальна и равна

$$N_f \;=\; 2^{n-1} - 2^{\frac{n}{2}-1}.$$

Множество аргументов, способных улучшить нелинейность функции $f(x)$, $1 - IS_f$:

$$1 - IS_f = \{x_1 | g(x_1) = f(x_1) \oplus 1, \forall\, x \neq x_1\; g(x) = f(x)\;\; N_g > N_f\;,$$

Иными словами, множество входов (аргументов) функции $f(x)$ для которых изменение значения соответствующего выхода (значения) увеличивает нелинейность $f(x)$.

Максимальное значение коэффициента Уолша-Адамара:

$$\max_{\boldsymbol{v}\in\mathbb{Z}_2^n}|W_f\,(\boldsymbol{v})| \;=\; WH_{\max}\,.$$

Множества векторов, для которых абсолютные значения коэффициентов Уолша-Адамара равны или близки к максимальному:

$$\begin{aligned}
W_1^+ &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = \;WH_{\max}\};\\
W_1^- &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = -WH_{\max}\};\\
W_2^+ &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = \;WH_{\max} - 2\};\\
W_2^- &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = -(WH_{\max} - 2)\};\\
W_3^+ &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = \;WH_{\max} - 4\};\\
W_3^- &= \{\boldsymbol{v} | W_f\,(\boldsymbol{v}) = -(WH_{\max} - 4)\}.
\end{aligned}$$

Исходя из определения нелинейности булевой функции, для увеличения нелинейности нужно минимизировать значение $\max_{\boldsymbol{v}\in\mathbb{Z}_2^n}|W_f\,(\boldsymbol{v})|$.

В [11] представлены алгоритмы, результатом работы которых является список позиций таблицы истинности, значение функции в которых, будучи изменено, повысит нелинейность функции. Подход основывается на наблюдении, что изменения в ТИ (под изменением/модификацией таблицы истинности здесь и далее следует понимать изменение бита значения функции в некоторой строке ТИ на противоположный) меняют и значения коэффициентов в $\boldsymbol{W}_f\,(\boldsymbol{v})$. Авторы алгоритмов [11] заметили, что при внесении изменения в одну из строк таблицы истинности функции, значения каждого из коэффициентов Уолша-Адамара $W_f\,(\boldsymbol{v})$ изменяется на ± 2 ($\forall \boldsymbol{v} \in \mathbb{Z}_2^n\; \Delta W_f\,(\boldsymbol{v}) = \pm 2$); если заменить два значения в таблице истинности, изменение каждого из коэффициентов составит $\Delta W_f\,(\boldsymbol{v}) \in \{-4,0,4\}$.

Чтобы увеличить нелинейность функции требуется увеличить значения коэффициентов в W_1^-, W_2^- на 2, уменьшить значения коэффициентов в W_1^+, W_2^+ на 2. Условия, касающиеся множеств W_2^+, W_2^-, нужны, чтобы остальные модули коэффициентов Уолша-Адамара оставались меньше $WH_{\max}$.

Алгоритмы, представленные в [11], используют технику восхождения к вершине (Hill Climbing), последовательно модифицируя таблицу истинности изначально произвольной булевой функции так, чтобы с каждым шагом нелинейность возрастала.

При этом алгоритмы позволяют сохранить или повысить уравновешенность исходной функции при соблюдении дополнительных условий.

Данные алгоритмы основывается, главным образом, на трех теоремах. Первая из них определяет условия принадлежности входа (аргумента) x к множеству $1 - IS_f$.

Теорема 1:

Пусть $f(x)$ – булева функция, $\boldsymbol{W_f}(\boldsymbol{v})$ – вектор коэффициентов Уолша-Адамара, заданы множества $W^+ = W_1^+ \cup W_2^+, W^- = W_1^- \cup W_2^-$. Чтобы вход x был элементом множества $1 - IS_f$, должны быть выполнены следующие условия:

$$\forall\, \boldsymbol{v} \in W^+ \; f(x) = L_{\boldsymbol{v}}(x);$$
$$\forall\, \boldsymbol{v} \in W^- \; f(x) \neq L_{\boldsymbol{v}}(x).$$

Если нужно повысить сбалансированность функции, потребуется наложить еще одно ограничение (так как сбалансированная функция имеет $W_f(0) = 0$):

$$W_f(0) > 0, \qquad f(x) = 0.$$

Для криптографических задач возникает необходимость получить высокую нелинейность при сохранении сбалансированности функции, что может быть достигнуто только четным числом изменений таблицы истинности. Определим множество пар аргументов, способных улучшить нелинейность функции $2 - IS_f$ как множество пар входов (аргументов), изменение соответствующих выходов (значений функции) которых повышает нелинейность функции при сохранении ее сбалансированности. Критерием принадлежности произвольной пары входов этому множеству является Теорема 2.

Теорема 2:

Пусть $f(x)$ – булева функция, $\boldsymbol{W_f}(\boldsymbol{v})$ – вектор коэффициентов Уолша-Адамара, заданы множества $W_1 = W_1^+ \cup W_1^-, W_{2,3}^+ = W_2^+ \cup W_3^+, W_{2,3}^- = W_2^- \cup W_3^-$. Пара входов x_1, x_2 является элементом множества $2 - IS_f$, если и только если выполнены следующие условия:

$$f(x_1) \neq f(x_2);$$
$$\forall \boldsymbol{v} \in W_1 \; L_{\boldsymbol{v}}(x_1) \neq L_{\boldsymbol{v}}(x_2);$$
$$\forall \boldsymbol{v} \in W_1^+ \; f(x_i) = L_{\boldsymbol{v}}(x_i), i \in \{1,2\};$$
$$\forall \boldsymbol{v} \in W_1^- \; f(x_i) \neq L_{\boldsymbol{v}}(x_i), i \in \{1,2\};$$
$$\forall \boldsymbol{v} \in W_{2,3}^+ \; L_{\boldsymbol{v}}(x_1) \neq L_{\boldsymbol{v}}(x_2) \Rightarrow f(x_i) = L_{\boldsymbol{v}}(x_i), i \in \{1,2\};$$
$$\forall \boldsymbol{v} \in W_{2,3}^- \; L_{\boldsymbol{v}}(x_1) \neq L_{\boldsymbol{v}}(x_2) \Rightarrow f(x_i) \neq L_{\boldsymbol{v}}(x_i), i \in \{1,2\}.$$

Теорема 3 описывает правило, по которому изменяются элементы вектора коэффициентов Уолша-Адамара при замене результата функции для определенного входа.

Теорема 3:

Пусть $g(x)$ получена из $f(x)$ заменой выхода для единственного входа x_1. Каждый из коэффициентов Уолша-Адамара может быть получен так: $W_g(\boldsymbol{v}) = W_f(\boldsymbol{v}) + \Delta W$; при этом $\Delta W = -2, f(x_1) = L_{\boldsymbol{v}}(x_1); \Delta W = 2, f(x_1) \neq L_{\boldsymbol{v}}(x_1)$.

Алгоритмы в [11] реализованы как рекурсивные процедуры HillClimb и HillClimb2. HillClimb в ходе работы применяет одиночные модификации таблицы истинности, а HillClimb2 – двойные. Процедуры представляют из себя проход по битам таблицы истинности с проверкой соблюдения условий Теоремы 2 и перерасчетом вектора коэффициентов Уолша-Адамара согласно Теореме 3.

Анализ результатов работы процедур HillClimb и HillClimb2

Авторы [11] провели сравнение распределений нелинейности случайных и случайных сбалансированных функций с функциями, полученными процедурами HillClimb и HillClimb2.

Как видно на рис. 1 функции, полученные методом HillClimbing, будут иметь высокую нелинейность с большей вероятностью, чем случайным образом сгенерированные.

На рис. 2 показано, как меняется среднее число шагов до локального максимума в зависимости от размера входа функции.

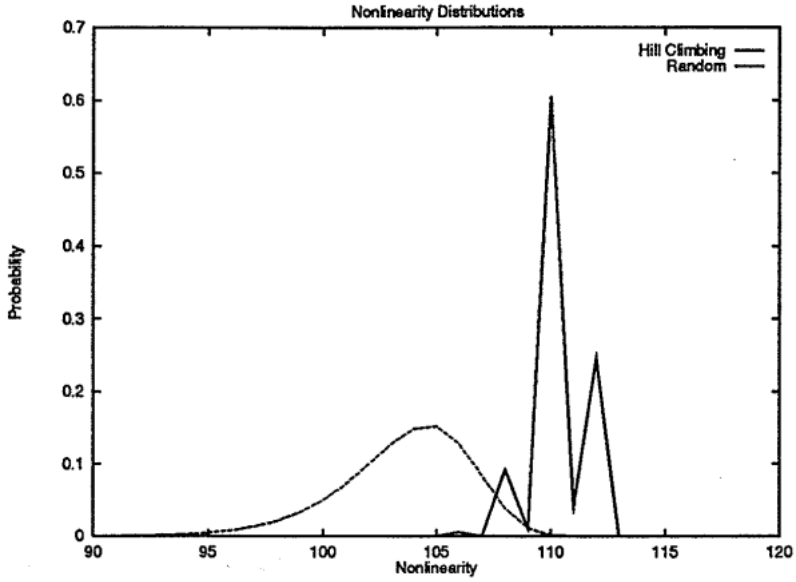

Рис. 1. Сравнение распределения нелинейности HillClimbing и случайной генерации, n=8[11]

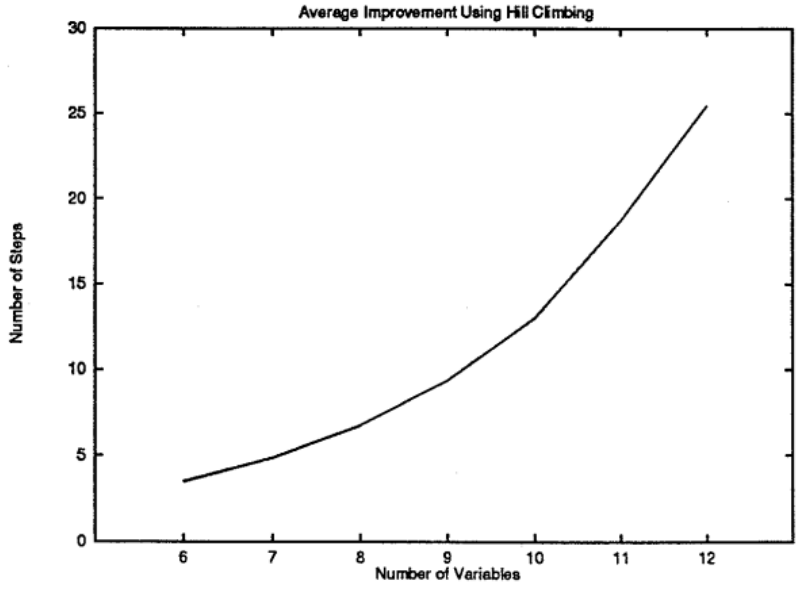

Рис. 2. Зависимость числа шагов от размера входа n [11]

Расстояние от случайной функции до локального максимума нелинейности (число шагов алгоритма) увеличивается экспоненциально, что позволяет сделать вывод о большей эффективности алгоритма при больших n: согласно Теореме 3 n шагов будут сделаны приблизительно за то же время, что и случайная генерация функции с вычислением ее вектора коэффициентов Уолша-Адамара.

Исходя из изложенного, *сильными сторонами алгоритма являются:*

– Значительное превосходство в эффективности над случайной генерацией, возрастающее с ростом n;

– Функции, полученные методом HillClimbing, будут иметь высокую нелинейность с большей вероятностью, чем случайным образом сгенерированные;

– Возможность улучшить нелинейность произвольной функции;

– Теоретическая возможность повысить сбалансированность функции (см. условие 3 теоремы 1);

– Возможность сохранить изначальную сбалансированность функции (процедура HillClimb2).

Слабыми сторонами алгоритма являются:

– Находит только локальный максимум, так как используется техника оптимизации HillClimbing. В большинстве случаев полученная функция будет иметь относительно высокую нелинейность, но бент-функцией не будет (глобальный максимум);

– Алгоритм, вероятно, может быть улучшен: не определена связь между мощностью множеств улучшений и нелинейностью функции: как указывают авторы, для функций, имеющих одинаковую нелинейность, размеры множеств могут отличаться.

В следующем разделе рассмотрим алгоритм, основанный на ином представлении булевой функции – алгебраической нормальной форме (она же полином Жегалкина) и ином подходе к генерации бент-функций.

1.2 Добавление слагаемых к АНФ

Булева функция может быть однозначно задана своей АНФ:

$$f(v) = a_0 \oplus \sum_{i=1}^{n} a_i x_i \oplus \sum_{1<i<j<n}^{n} a_{ij} x_i x_j \oplus \ldots \oplus a_{1,\ldots,n} x_1 \cdot \ldots \cdot x_n,$$

где знак сложение производится по модулю 2, а коэффициенты $a \in \{0,1\}$.

В [12] предложен алгоритм, позволяющий генерировать бент-функции из квадратичной бент-функции, добавляя к АНФ слагаемые более высоких степеней.

Таблицу истинности булевой функции $f(x)$ от n переменных можно рассматривать как конкатенацию таблиц истинности двух булевых подфункций $f_1(x), f_0(x)$ от n-1 переменной. В общем случае $f(x)$ может быть расщеплена вдоль направления, заданного вектором $\boldsymbol{v}$, тогда подфункции будут определены следующим образом:

$$f = [f_0|f_1]_v;$$

$$f_0(x) = \begin{cases} f(x), L_{\boldsymbol{v}}(x) = 0; \\ \text{иначе не определена.} \end{cases}$$

$$f_1(x) = \begin{cases} f(x), L_{\boldsymbol{v}}(x) = 1; \\ \text{иначе не определена.} \end{cases}$$

Конкатенация получается естественным образом, если вектор $\boldsymbol{v}$ – единичный (обозначим его $\boldsymbol{v_1}$).

Пусть $f = [f_0|f_1]_{\boldsymbol{v_1}}$ расщеплена единичным вектором на две функции $f_1(x), f_0(x)$. Для вектора $\boldsymbol{v}$ найдется невырожденная двоичная матрица T размера $n \times n$, которая переводит его в единичный: $T(\boldsymbol{v}) = \boldsymbol{v_1}$. То есть можно применить обратимое линейное преобразование $g(x) = T\big(f(x)\big) = f\big(T(x)\big)$, причем $g = [g_0|g_1]_v$. В силу обратимости преобразования, расщепление вдоль единичного вектора $\boldsymbol{v_1}$ функции $g(x)$ равносильно расщеплению вдоль некого вектора $T^{-1}(\boldsymbol{v_1}) = \boldsymbol{v}$ преобразованной функции $f(x) = T^{-1}\big(g(x)\big)$, то есть можем, не умаляя общности, говорить о $\boldsymbol{v}$ как о единичном векторе.

Также, если $h(x) = g(x) \oplus f(x)$ и каждая из функций расщеплена вдоль вектора $\boldsymbol{v}$, $h_i(x) = g_i(x) \oplus f_i(x), i \in \{1,2\}$.

Приведенная ниже теорема связывает вес Хэмминга функции (количество единиц в столбце значений ТИ функции) с значениями коэффициентов Уолша-Адамара.

Теорема 4: вес Хэмминга hwt подфункций, полученных любым $\boldsymbol{v}$-расщеплением функции f, выражается через ее коэффициенты Уолша-Адамара следующим образом:

$$hwt(f_0) = \frac{2^n - W_f\,(0) - W_f\,(\boldsymbol{v})}{4};$$

$$hwt(f_1) = \frac{2^n - W_f\,(0) + W_f\,(\boldsymbol{v})}{4}.$$

Результатами применения этой теоремы к бент-функциям являются два следствия, которые непосредственно использованы при построении алгоритма.

Следствие 1: Если $|W_f\,(0)| = |W_f\,(\boldsymbol{v})|$ для $\boldsymbol{v} \neq \mathbf{0}$, тогда $hwt(f_0) = 2^{n-2}$ или $hwt(f_1) = 2^{n-2}$. Иными словами, $f = [f_0|f_1]_{\boldsymbol{v}}$ полусбалансирована.

Следствие 2: $f = [f_0|f_1]_{\boldsymbol{v}}$ – бент-функция, если и только если $\forall \boldsymbol{v} \neq \mathbf{0}\ hwt(f_0) = 2^{n-2}$ или $hwt(f_1) = 2^{n-2}$.

То есть булева функция будет бент-функцией если и только если она полусбалансирована.

Описание алгоритма, основанного на добавлении слагаемых к АНФ

Слагаемые добавляются в АНФ исходной квадратичной бент-функции, которая может быть как постоянной, так и полученной методом случайной генерации.

Важно заметить, что не любое слагаемое может быть добавлено к АНФ, иначе в вычислительном смысле алгоритм не отличался бы от случайной генерации, которая потребовала бы вычисления вектора Уолша-Адамара для расчета нелинейности получившейся функции. Подходит слагаемое или нет определяется на основе двух критериев. Во-первых, слагаемое, добавляемое в АНФ, не должно иметь степень, превосходящую $\frac{n}{2}$ (степень бент-функций не превосходит $\frac{n}{2}$ [4], что также используется в алгоритме, рассмотренном в [13]). Во-вторых, бент-функции полусбалансированы, а значит ее подфункции должны удовлетворять следствию 2.

Слагаемое будет добавлено в АНФ только при соблюдении обоих условий: степень функции не станет больше, чем $\frac{n}{2}$; веса Хэмминга для подфункций будут удовлетворять следствию 2.

Автор в [12] обращает внимание, что, хотя согласно следствию 2 должны быть проверены все вектора направления расщепления $\boldsymbol{v} \neq \mathbf{0}$, на практике эффективнее оказывается сначала проверить все единичные вектора, а затем рассчитать нелинейность функции и сравнить ее со значением для бент-функций.

Алгоритм принимает на вход количество переменных и максимальную степень функции и генерирует множество бент-функций, с его псевдокодом, как и с доказательствами теорем и следствий можно ознакомиться в [12].

В табл. 1 представлены результаты оценки эффективности алгоритма.

Максимальная степень функции ord_{max} для всех размеров n ($8 \leq n \leq 16$) лежит в границах $3 \leq ord_{max} \leq \frac{n}{2}$. Третий и четвертый столбцы таблицы отражают количество функций, прошедших отбор по первому (степень функции) и второму (веса Хэмминга для расщепления по единичным векторам) критериям отбора соответственно.

Исходя из полученных данных были сделаны следующие выводы:

– Для всех рассмотренных значений входных параметров алгоритм сгенерировал множества бент-функций;

– Количество бент-функций увеличивается с ростом входных параметров n, ord_{max}, но для $n > 12$ количество сгенерированных функций для значения $ord_{max} - 1$ превосходит такое для ord_{max}, что предположительно связано с меньшим абсолютным количеством бент-функций такой степени, а следовательно меньшей вероятностью их генерации.

– В среднем 66% функций были отброшены по второму критерию (веса Хэмминга), а из функций, прошедших эту проверку, бент-функциями были от 0.41% до 63.16%, причем эта доля снижалась с ростом n и ord_{max}.

Таблица 1. Результаты работы алгоритма, добавляющего слагаемые в АНФ

n	Макс. порядок слагаемого	Количество слагаемых АНФ, соответствующих условию порядка	Количество слагаемых АНФ, соответствующих условию полусбалансированности	Всего бент-функций	Макс. количество слагаемых в АНФ	Среднее время на генерацию функции (мс)
8	3	148	19	12	16	0.00
8	4	461	106	17	21	0.59
10	3	221	45	19	24	0.63
10	4	1234	333	32	37	1.56
10	5	2736	609	51	56	1.76
12	3	403	92	37	43	2.16
12	4	1374	530	56	62	6.07
12	5	7230	2087	78	84	17.97
12	6	4796	2042	66	72	19.58
14	3	657	181	71	78	10.01
14	4	3984	1889	57	64	111.04
14	5	9866	3760	118	125	110.25
14	6	24927	8956	187	194	165.32
14	7	58120	16583	146	153	404.27
16	3	1007	328	113	121	52.38
16	4	4595	2546	165	173	248.36
16	5	13204	5897	292	300	332.22
16	6	43550	21258	365	373	930.08
16	7	155116	61076	424	432	2361.32
16	8	194022	83545	339	347	3947.06

Сильными сторонами алгоритма являются:

– Скорость работы явно превосходит для случайной генерации и является достаточно высокой в целом (для наибольших входных параметров $n = 16$ и $ord_{max} = 8$ генерация происходила за 3,95 с в среднем, что с учетом использованной аппаратной части Pentium III 500, 128Mb RAM быстро);

– Алгоритм позволяет ограничить степень и задать явно количество входов функции, что может быть полезно в практических приложениях.

Слабыми сторонами алгоритма являются:

– Необходимость наличия бент-функции для начала работы (скорее особенность, не являющаяся существенной проблемой).

Рассмотренные подходы (модификация таблицы истинности и добавление слагаемых к АНФ) имеют свои недостатки: первый порождает функции, имеющие не максимальную нелинейность, второй использует не всегда необходимую форму представления функции (АНФ), оба могут быть улучшены с точки зрения скорости работы.

В следующем разделе предложен разработанный алгоритм, в некоторых отношениях превосходящий изложенные ранее, он основывается на функции следа и вейвлет-преобразовании.

2. Алгоритм генерации бент-функций, использующий вейвлет–преобразование

2.1 Вейвлет–преобразование

Вейвлет–преобразование применяется, в самом общем случае, для обработки числовых потоков и призвано сократить объем передаваемой информации, путем отбрасывания той ее части, которая с какой-либо точки зрения является несущественной.

Идея вейвлет–преобразования заключается в разделении информационного потока c_i на основной a_i и вейвлетный b_i, служащий для уточнения [14]:

$$a_j = \frac{c_{2j} + c_{2j+1}}{2}, b_j = \frac{c_{2j} - c_{2j+1}}{2}.$$

Восстановление потока происходит по формулам:

$$c_{2j} = a_j + b_j, c_{2j+1} = a_j - b_j .$$

Исходный поток разделяется на основной и вейвлетный согласно формулам декомпозиции, а переход от двух потоков к исходному происходит по формулам реконструкции.

Для вейвлетного преобразования необходима также сетка $X = \{x_j\}$, то есть дискретное множество чисел. Удаляя из этой сетки узлы, мы сможем отбрасывать не интересующие нас элементы информационного потока.

Опишем подробнее сплайн-вейвлетное преобразование, так как оно будет применено в предложенном алгоритме.

B-сплайны первого порядка на сетке X могут быть заданы следующими формулами:

$$w_j(t) = \frac{t - x_j}{x_{j+1} - x_j}, \qquad t \in [x_j, x_{j+1}];$$

$$w_j(t) = \frac{x_{j+2} - t}{x_{j+2} - x_{j+1}}, \qquad t \in [x_{j+1}, x_{j+2}];$$

$$w_j(t) = 0, \qquad t \notin [x_j, x_{j+2}].$$

Удалим из сетки элемент x_k,так что:

$$\tilde{x}_j = x_j, \qquad j \le k - 1;$$

$$\tilde{x}_j = x_{j+1}, \qquad j \ge k,$$

где $\tilde{x}_j$ элементы новой сетки $\widetilde{X}$.

Представим новые сплайны $\widetilde{w_j}(t)$, получившиеся на укрупненной сетке с помощью сплайнов $w_j(t)$:

$$\widetilde{w}_j(t) = w_j(t), \qquad j \le k - 3;$$

$$\widetilde{w}_{k-2}(t) = w_{k-2}\ \frac{(x_k - \tilde{x}_{k-2})}{\tilde{x}_k - \tilde{x}_{k-2}} + \frac{(\tilde{x}_{k+1} - x_k)}{\tilde{x}_{k+1} - \tilde{x}_{k-1}} w_{k-1};$$
$$\widetilde{w}_{k-1}(t) = w_{k-1}\frac{(x_k - \tilde{x}_{k-1})}{\tilde{x}_{k+1} - \tilde{x}_{k-1}} + w_k;$$
$$\widetilde{w}_j(t) = w_{j+1}(t), \qquad j \geq k.$$

Пространство $P(X)$ является линейной оболочкой множества функций w_j и называется, в данном случае, пространством B-сплайнов первой степени на сетке X:

$$P(X) = \{v | v = \sum_j c_j w_j \ \forall\ c_j \in \mathbb{R}\},$$
$$P(\tilde{X}) = \{v | v = \sum_j \tilde{c_j}\widetilde{w}_j \ \forall \tilde{c_j}\ \in \mathbb{R}\}.$$

где $P(\tilde{X})$ – пространство B-сплайнов первой степени на сетке $\tilde{X}$, вложенное в пространство $P(X)$, $P(\tilde{X}) \subset P(X)$, при этом прямая сумма $P(\tilde{X}) \oplus \widetilde{W}$ является сплайн-вейвлетным разложением пространства $P(X)$, а $\widetilde{W}$ является пространством вейвлетов.

Приведенные выше формулы, выражающие сплайны на укрупненной сетке через сплайны на мелкой сетке, позволяют получить выражения для декомпозиции и реконструкции.

Формулы декомпозиции:

$$a_i = c_i\ , \qquad i \leq k-2;$$
$$a_i = c_{i+2}, \qquad i \geq k-1$$
$$b_{k-1} = c_{k-1} - \frac{x_{k+1} - x_k}{x_{k+1} - x_{k-1}} c_{k-2} - \frac{x_k - x_{k-1}}{x_{k+1} - x_{k-1}} c_k;$$
$$b_i = 0, \qquad i \neq k-1.$$

Формулы реконструкции:

$$c_i = a_i, \qquad i \leq k-2;$$
$$c_i = a_{i-2}, \qquad i \geq k-1;$$
$$c_k = \left(a_{k-1} - \frac{x_{k+1} - x_k}{x_{k+1} - x_{k-1}} a_{k-2} - b_{k-1}\right)\left(\frac{x_{k+1} - x_{k-1}}{x_k - x_{k-1}}\right).$$

2.2 Описание алгоритма генерации бент-функций, основанного на вейвлет–преобразовании и функции следа

Алгоритм опирается на существующую конструкцию [15], позволяющую применить вейвлет–преобразование для создания бент-функций. Ее основной частью является функция следа $tr: GF(2^n) \to GF(2)$:

$$tr(x) = x + x^2 + x^4 + \cdots + x^{2^{n-1}}.$$

Вейвлет–преобразование может быть применено здесь для получения a: при наличии числового потока можно выбрать некий произвольный (случайный) индекс I выбрасываемого элемента и рассчитать элемент вейвлетного потока $b \in GF(2^n)$, который и станет исходным значением для $a \in GF(2^n) \setminus \{x^3 | x \in GF(2^n)\}$.

Взаимно простое с n число k, необходимое для расчета степени $d \equiv 2^{2k} - 2^k + 1$, возможно получить, например следующим образом: создать достаточно большое множество элементов, взаимно простых с n, назовем его K, и при надобности случайным образом выбирать из него элемент k_i. Возможно при необходимости менять множество и способ его построения.

Особенностью алгоритма будет то, что он может генерировать повторяющиеся функции, если не налагать условие уникальности на числовой поток, поступающий в вейвлет–преобразование, или на число k. Чем больше порядок поля и размер множества K тем реже будут генерироваться функции, уже полученные ранее.

Можно изменять алгоритм в зависимости от задачи, для которой используются бент-функции, генерируемые им, а именно вычислять значение полученных функций на ходу без их запоминания или заполнять множество функций, возможно меняя форму их представления (например, получать АНФ или ТИ).

На рис. 3 ниже представлена блок-схема для варианта с записью функций в множество в форме пары (a, d).

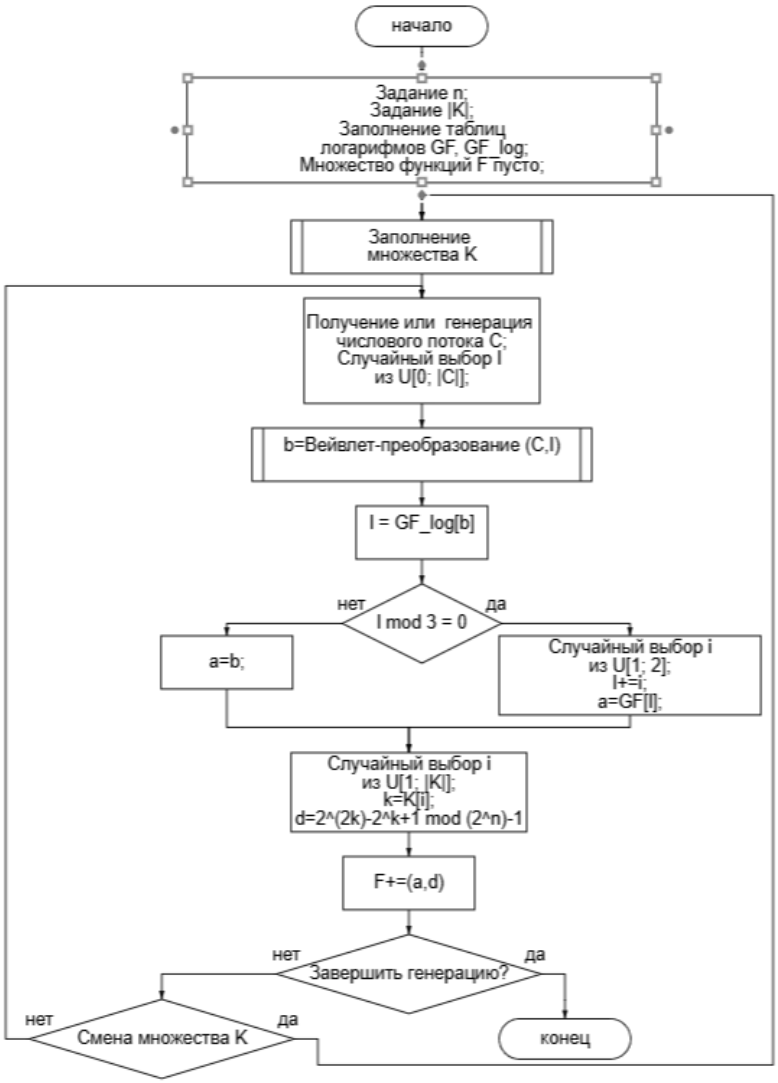

Рис. 3. Блок-схема алгоритма

Обоснуем изложенные выше положения, на которых основывается алгоритм. Докажем, что функции, генерируемые им, будут действительно являться бент-функциями.

Теорема:

Для любого $a \in GF(2^n) \setminus \{x^3 | x \in GF(2^n)\}, n$ неделящегося на 3, k взаимно простого с n функция $tr\left(ax^{2^{2k}-2^k+1}\right): GF(2^n) \to GF(2)$ является бент.

Доказательство:

Ссылка на работы, в которых содержится доказательство в [15]. **чтд.**

Необходимо наложить дополнительное ограничение на n помимо уже упомянутого в формулировке теоремы: n должно быть четно, иначе все элементы $GF(2^n)$ будут кубами, то есть множество $GF(2^n) \setminus \{x^3 | x \in GF(2^n)\}$ будет пусто.

Лемма:

Для нечетных n множество $|\{x^3 | x \in GF(2^n)\}| = |GF(2^n)|$, то есть $GF(2^n) \setminus \{x^3 | x \in GF(2^n)\} = \emptyset$.

Для четных n множество $|\{x^3 | x \in GF(2^n)\}| = \frac{2^n-1}{3}$.

Доказательство:

По т. о гомоморфизме: $\{x^3 | x \in GF(2^n)\} \cong GF(2^n)^* / Ker(x \to x^3)$;

$$|\{x^3 | x \in GF(2^n)\}| = \frac{|GF(2^n)^*|}{|Ker(x \to x^3)|}.$$

Пусть g – примитивный элемент поля $GF(2^n)$. Найдем мощность ядра, соответствующую количеству k, удовлетворяющие сравнению:

$$(g^k)^3 \equiv g^0 \equiv 1 \; mod \; 2^n$$
$$3k \equiv 0 \; mod \; 2^n - 1.$$

Число решений этого линейного сравнения равно $\gcd(2^n - 1, 3)$.

$2^n - 1 = 4^k - 1 \equiv 0 \; mod \; 3$, n четно

$2^n - 1 = 2^{2k+1} - 1 \equiv 2 \; mod \; 3$, n нечетно

$$|Ker(x \to x^3)| = \gcd(2^n - 1, 3) = \begin{bmatrix} 3, n \text{ четно} \\ 1, n \text{ нечетно} \end{bmatrix}$$

$$|\{x^3 | x \in GF(2^n)\}| = \frac{|GF(2^n)^*|}{|Ker(x \to x^3)|} = \begin{bmatrix} \frac{2^n - 1}{3}, n \text{ четно} \\ 2^n - 1, n \text{ нечетно} \end{bmatrix}$$

чтд.

В случае, если n четно, степень примитивного элемента любого куба $y \in \{x^3 | x \in GF(2^n)\}$ будет делится на 3. Иными словами, пусть $b = g^k$, где g – примитивный элемент поля $GF(2^n)$, n четно, тогда $b^3 = g^{3k \; mod \; (2^n-1)}$ и $3 \mid (3k) \; mod \; (2^n - 1)$. Исходя из этого, можно предложить такой порядок действий над элементом вейвлетного потока $b = g^k \in GF(2^n)$: если $b \in GF(2^n) \setminus \{x^3 | x \in GF(2^n)\}$, то есть $3 \nmid k \; mod \; (2^n - 1)$, берем без изменений $a = b$, иначе рассчитываем $a = bg^i = g^{k+i}$, где i- случайное целое число из диапазона [1:2].

Лемма 2:

Для четных n, чтобы произвольный элемент $b \in GF(2^n)$ отобразить в элемент $a \in GF(2^n) \setminus \{x^3 | x \in GF(2^n)\}$, достаточно следующего преобразования:

$$a = \begin{bmatrix} \mathrm{b}, \log(b) \not\equiv 0 \; mod \; 3 \\ \mathrm{b} \cdot \mathrm{g}^{\mathrm{l}}, \; \log(b) \equiv 0 \; mod \; 3 \end{bmatrix},$$

где g – примитивный элемент поля $GF(2^n), l \in \{1,2\}, \log(b)$ возвращает степень примитивного элемента, соответствующую b.

Доказательство:

Элементы $\{x^3 | x \in GF(2^n)\}$ имеют показатель степени примитивного элемента вида $3k, k \in \mathbb{Z}$, если n четно.

Если $\log(b) \not\equiv 0 \; mod \; 3$, можем определить $\mathrm{a} = b$, так как b кубом не является.

Если $\log(b) \equiv 0 \; mod \; 3$, то b является кубом, и возможно взять одну из ближайших степеней примитивного элемента, которая кубом не будет, то есть $g^{\log(b)+l} = \mathrm{b} \cdot \mathrm{g}^{\mathrm{l}}$, $l \in \{1,2\}$. **чтд.**

2.3 Сравнение предложенного алгоритма с алгоритмами, основанными на модификации ТИ, добавлении слагаемых в АНФ

Результаты замеров времени работы предложенного алгоритма и их сравнение с алгоритмами, основанными на модификации ТИ, добавлении слагаемых в АНФ, приведены в табл. 2. Под подготовкой понимается главным образом заполнение таблиц для вычислений в конечных полях. Значения в табл. 2 осреднены по размеру числового потока, который варьировался от 2^{n-4} до 2^n с шагом 2^{n-4} и по количеству сгенерированных функций (100 для каждого значения n и размера числового потока).

Размер числового потока не оказывает значительного влияния на время генерации, если он достаточно велик (2^{n-4} может считаться таковым).

Незначительный рост времени генерации функции объясняется особенностью реализации: арифметические операции в полях $GF(2^n)$ реализованы как обращение к элементам таблиц, рассчитанных предварительно.

Таблица 2. Замеры времени работы разработанного алгоритма

n	Среднее время генерации функции с учетом подготовки, мс	Среднее время генерации функции, мс	Среднее по n время генерации для алгоритма, добавляющего слагаемые в АНФ	Выигрыш, раз
8	0,113	0,074	1,28	11,33
10	0,169	0,077	11,5	68,05
14	1,18	0,083	160	135,59
16	9,58	0,085	1312	136,95

По сравнению с алгоритмом, использующим метод HillClimbing, новый алгоритм имеет следующие преимущества:

– в большинстве случаев нелинейность генерируемых функций будет выше, так как новый алгоритм порождает бент-функции;

– скорость генерации нового алгоритма выше.

По сравнению с алгоритмом, использующим добавление подходящих слагаемых в АНФ, новый алгоритм имеет следующие преимущества:

– нет необходимости иметь предварительно заданную квадратичную бент-функцию;

– значительно более высокая скорость генерации (см. пятый столбец табл. 2).

У разработанного алгоритма есть особенность, для представления функций в виде АНФ или таблицы истинности нужно проводить дополнительные вычисления, которые могут быть относительно затратны. В этом отношении второй из рассмотренных существующих алгоритмов выигрывает, так как сразу оперирует АНФ в ходе работы.

Заключение

В работе рассмотрены и проанализированы два существующих алгоритма генерации бент-функций. Первый использует модификацию таблицы истинности исходной функции, а второй – добавление подходящих слагаемых к АНФ функции. Разработан новый алгоритм, в основе которого лежит вейвлет–преобразование и функция следа, а также проведено его сравнение с существующими. Алгоритм, изменяющий ТИ исходной функции, улучшает ее нелинейность, но результат его работы скорее всего не будет бент-функцией, а алгоритм, основанный на вейвлет–преобразовании и функции

следа всегда генерирует бент-функции. Скорость генерации функций у алгоритма, добавляющего слагаемые в АНФ высока, но новый алгоритм, задействующий вейвлет–преобразование и функцию следа превосходит его в этом отношении. Генерируемые полученным алгоритмом функции, как и сам алгоритм, могут найти применение в теории кодирования, криптографии.

СПИСОК ЛИТЕРАТУРЫ:

1. Rothaus O. On bent functions. IDA CRD W.P. No.169.1966. DOI: https://doi.org/10.1016/0097-3165(76)90024-8.
2. Dillon J.F. A survey of bent functions. The NSA Technical J. 1972. Special Issue. P. 191–215. URL: https://cryptome.org/2015/11/nsa-survey-of-bent-functions.pdf (дата обращения: 12.05.2025).
3. McFarland R.L. A family of difference sets in non-cyclic groups. J. Combin. Theory. Ser. A. 1973, v. 15, no. 1, p. 1–10. DOI: https://doi.org/10.1016/0097-3165(73)90031-9.
4. Токарева Н.Н. Нелинейные булевы функции: бент-функции и их обобщения. Издательство LAP LAMBERT Academic Publishing (Saarbrucken, Germany), ISBN: 978-3-8433-0904-2. 2011. – 180 с.
5. Левина А.Б., Ряскин Г.А. Сплайн-вейвлетные надежные бент-коды. Научно-технический вестник информационных технологий, механики и оптики. 2021, т. 21, № 6, с. 936–941. DOI: https://doi.org/10.17586/2226-1494-2021-21-6-936-941.
6. Левина А.Б., Ряскин Г.А. Создание надежных кодов на основе бент-функций и вейвлет–преобразований. Научно-технический вестник информационных технологий, механики и оптики. 2018, т. 18, № 6, с. 1008–1015. DOI: https://doi.org/10.17586/2226-1494-2018-18-6-1008-1015.
7. Levina, A.; Ryaskin, G. Robust Code Constructions Based on Bent Functions and Spline Wavelet Decomposition. Mathematics 2022, 10, 3305. DOI: https://doi.org/10.20944/preprints202208.0348.v1.
8. Matsui M. Linear cryptanalysis method for DES cipher. Advances in Cryptology – EUROCRYPT'93. Workshop on the theory and application of cryptographic techniques (Lofthus, Norway. May 23–27, 1993). Proc. Berlin: Springer, 1994, p. 386–397 (Lecture Notes in Comput. Sci. V. 765). DOI: https://doi.org/10.1007/3-540-48285-7_33.
9. Молдовян А.А., Молдовян Н.А., Еремеев М.А. Криптография: от примитивов к синтезу алгоритмов. СПб.: БХВ-Петербург, 2004. ISBN 5-94157-524-6. URL: https://books.4nmv.ru/books/kriptografiya_ot_primitivov_k_sintezu_algoritmov_3642674.pdf (дата обращения: 12.05.2025).
10. Carlet C., Ding C., Niederreiter H. Authentication schemes from highly nonlinear functions. Designs, Codes and Cryptography. 2006, v. 40, no. 1, p. 71–79. DOI: https://doi.org/10.1007/s10623-005-6407-0.
11. Millan W., Clark A., Dawson E. Smart hill climbing finds better Boolean functions. Workshop on Selected Areas in Cryptology. 1997. Workshop record. P. 50–63. URL: https://www.researchgate.net/publication/2357166_Smart_Hill_Climbing_Finds_Better_Boolean_Functions (дата обращения: 12.05.2025).
12. Fuller J.E. Analysis of affine equivalent Boolean functions for cryptography. Ph. D. thesis, Queensland University of Technology. Brisbane, Australia. 2003. URL: https://scispace.com/papers/analysis-of-affine-equivalent-boolean-functions-for-5u4cm5dfp0 (дата обращения: 12.05.2025).
13. Grocholewska-Czury lo A. A study of differences between bent functions constructed using Rothaus method and randomly generated bent functions. J. Telecommunications and Information Technology. 2003, no. 4, p. 19–24. URL: https://www.researchgate.net/publication/239550136_A_study_of_dierences_between_bent_functions_constructed_using_Rothaus_method_and_randomly_generated_bent_functions (дата обращения: 12.05.2025).
14. Демьянович Ю.К., Ходаковский В.А. Введение в теорию вэйвлетов: учебное пособие. СПб.: Изд-во ПГУПС, 2008. – 50 p. URL: https://dl.libcats.org/genesis/366000/6382fd068c32a25b1a8312a510fa8ec9/_as/[Demyanovich_YU.K.,_Hodakovsky_V.A.]_Vvedenie_v_te(libcats.org).pdf (дата обращения: 12.05.2025).
15. Carlet C. Boolean Functions for Cryptography and Error Correcting Codes. Chapter of the monograph «Boolean Methods and Models», Cambridge Univ. Press (P. Hammer, Y. Crama eds.). URL: https://www.researchgate.net/publication/228720083_Boolean_Functions_for_Cryptography_and_Error_Correcting_Codes (дата обращения: 12.05.2025).

REFERENCES

[1] Rothaus O. On bent functions. IDA CRD W.P. No.169.1966. DOI: https://doi.org/10.1016/0097-3165(76)90024-8.

[2] Dillon J.F. A survey of bent functions. The NSA Technical J. 1972. Special Issue. P. 191–215. URL: https://cryptome.org/2015/11/nsa-survey-of-bent-functions.pdf (accessed: 12.05.2025).
[3] McFarland R.L. A family of difference sets in non-cyclic groups. J. Combin. Theory. Ser. A. 1973, v. 15, no. 1, p. 1–10. DOI: https://doi.org/10.1016/0097-3165(73)90031-9.
[4] Tokareva N.N. Nonlinear Boolean functions: bent functions and their generalizations: Theoretical results. Saarbrucken, Germany: LAP LAMBERT Academic Publishing, ISBN: 978-3-8433-0904-2. 2011. – 180 p (in Russian).
[5] Levina A.B., Ryaskin G.A. Spline-wavelet bent robust codes. Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2021, v. 21, no. 6, p. 936–941. DOI: https://doi.org/10.17586/2226-1494-2021-21-6-936-941 (in Russian).
[6] Levina A.B., Ryaskin G.A. Robust codes creation based on bent-functions and wavelet transformation. Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2018, v. 18, no. 6, p. 1008–1015. DOI: https://doi.org/10.17586/2226-1494-2018- 18-6-1008-1015 (in Russian).
[7] Levina, A.; Ryaskin, G. Robust Code Constructions Based on Bent Functions and Spline Wavelet Decomposition. Mathematics 2022, 10, 3305. DOI: https://doi.org/10.20944/preprints202208.0348.v1.
[8] Matsui M. Linear cryptanalysis method for DES cipher. Advances in Cryptology – EUROCRYPT'93. Workshop on the theory and application of cryptographic techniques (Lofthus, Norway. May 23–27, 1993). Proc. Berlin: Springer, 1994, p. 386–397 (Lecture Notes in Comput. Sci. V. 765). DOI: https://doi.org/10.1007/3-540-48285-7_33.
[9] Moldovyan A. A., Moldovyan N. A., Eremeev M. A. Cryptography: from primitives to synthesis algorithm. St. Petersburg: BHV-Petersburg, 2004. ISBN 5-94157-524-6. URL: https://books.4nmv.ru/books/kriptografiya_ot_primitivov_k_sintezu_algoritmov_3642674.pdf (accessed 12.05.25) (in Russian).
[10] Carlet C., Ding C., Niederreiter H. Authentication schemes from highly nonlinear functions. Designs, Codes and Cryptography. 2006, v. 40, no. 1, p. 71–79. DOI: https://doi.org/10.1007/s10623-005-6407-0.
[11] Millan W., Clark A., Dawson E. Smart hill climbing finds better Boolean functions. Workshop on Selected Areas in Cryptology. 1997. Workshop record. P. 50–63. URL: https://www.researchgate.net/publication/2357166_Smart_Hill_Climbing_Finds_Better_Boolean_Functions (accessed: 12.05.2025).
[12] Fuller J.E. Analysis of affine equivalent Boolean functions for cryptography. Ph. D. thesis, Queensland University of Technology. Brisbane, Australia. 2003. URL:https://scispace.com/papers/analysis-of-affine-equivalent-boolean-functions-for-5u4cm5dfp0 (accessed: 12.05.2025).
[13] Grocholewska-Czury lo A. A study of differences between bent functions constructed using Rothaus method and randomly generated bent functions. J. Telecommunications and InformationTechnology. 2003, no. 4, p. 19–24. URL: https://www.researchgate.net/publication/239550136_A_study_of_dierences_between_bent_functions_constructed_using_Rothaus_method_and_randomly_generated_bent_functions (accessed: 12.05.2025).
[14] Dem'yanovich Yu.K., Khodakovsky V.A. Introduction to the Theory of Wavelets: a tutorial. St. Petersburg: Publishing House of Emperor Alexander I St. Petersburg State Transport University, 2008. – 50 p. URL: https://dl.libcats.org/genesis/366000/6382fd068c32a25b1a8312a510fa8ec9/_as/[Demyanovich_YU.K.,_Hodakovsky_V.A.]_Vvedenie_v_te(libcats.org).pdf (accessed: 12.05.2025) (in Russian).
[15] Carlet C. Boolean Functions for Cryptography and Error Correcting Codes. Chapter of the monograph «Boolean Methods and Models», Cambridge Univ. Press (P. Hammer, Y. Crama eds.). URL: https://www.researchgate.net/publication/228720083_Boolean_Functions_for_Cryptography_and_Error_Correcting_Codes (accessed: 12.05.2025).

Статья поступила в редакцию 14.05.2025; одобрена после рецензирования 14.06.2025;
принята к публикации 01.08.2025
The article was submitted 14.05.2025; approved after reviewing 14.06.2025;
accepted for publication 01.08.2025