Московский Государственный Университет
Факультет Вычислительной Математики и Кибернетики

КУРСОВАЯ РАБОТА

студента группы М113
Гуламова Рустама Тохировича

Однородные бент-функции

Научный руководитель – Лавриков Иван Викторович

Москва – 2016 г.

Содержание

Введение

Как известно из практики математических исследований, линейность (в математическом смысле) изучаемого объекта позволяет получить в большинстве случаев достаточно полное описание его свойств, параметров или поведения. Линейность успешно используется в алгебре, теории систем, математической кибернетике и других разделах математики. Более того, близость изучаемых объектов и процессов к линейным также позволяет изучать их параметры и свойства.

С точки зрения построения надежных криптографических систем важным свойством является нелинейность, а существование у некоторой криптографической функции свойств, близких к линейным, безусловно считается слабостью. Выполнение подобных свойств также противоречит фундаментальным принципам построения криптографических функции.

Задача построения булевых функций, обладающих нелинейными свойствами, естественным образом возникает во многих областях дискретной математики. Часто наибольший интерес представляют те функции, для которых эти свойства экстремальны. Такие булевы функции называются *бент-функциями*. Бент-функции были введены О. Ротхаусом еще в 60-х годах XX века, хотя его работа на эту тему была опубликована лишь в 1976 году.

Бент-функция – булева функция с четным числом переменных, для которой расстояние Хэмминга от множества аффинных булевых функций с тем же числом переменных максимально. Бент-функцию можно определить как функцию, которая крайне плохо аппроксимируется аффинными функциями. В этом смысле рассматриваемые функции обладают максимальной степенью нелинейности среди всех функций с данным числом переменных. В блочных и поточных шифрах бент-функции и их векторные аналоги позволяют повысить стойкость этих шифров к линейному и

дифференциальному методам криптоанализа — современным методам криптоанализа шифров.

Актуальность исследования бент-функций подтверждается их многочисленными теоретическими и практическими приложениями в комбинаторике, алгебре, теории кодирования, теории информации, криптографии и криптоанализе. Например, классическая комбинаторная задача, в дискретной математике – построение матриц Адамара порядка N, в случае $\mathrm{N} = 2^n$ и при некоторых ограничениях, сводится к задаче построения бент-функции от n переменных. В теории информации: семейства бент-последовательностей из элементов +1 и −1, построенные на основе бент-функций, имеют предельно низкие значения как взаимной корреляции, так и автокорреляции. Поэтому такие семейства успешно применяются в коммуникационных системах коллективного доступа. Бент-функции применяются в технологии цифровой сотовой связи CDMA (Code Division Multiple Access — множественный доступ с кодовым разделением каналов). Для оценки сигнала в CDMA используется коэффициент отношения пиковой и средней мощностей сигнала PAPR (peak-to-average power ratio). Чем данный коэффициент ниже, тем сигнал считается лучше, так как снижается стоимость на усилители мощности и повышается надежность связи. Минимальное значение PAPR достигается на кодовых словах специального вида. Такими словами являются векторы значений бент-функций. Коды, состоящие из таких слов, называются кодами постоянной амплитуды. Так возникает задача построения кодов постоянной амплитуды наибольшей мощности и обладающих хорошей структурой, т.е. задача выбора специальных подмножеств множества бент-функций. В криптографии: например, в 1993 году была обнаружена существенная слабость к линейному криптоанализу блочного шифра DES. Шифр DES оказался нестойким и к дифференциальному методу криптоанализа. Причина заключалась в слабых S-блоках шифра.

Стойкость шифров может повышаться в случае использования бент-функций, их аналогов и обобщений при построении S-блоков.

В настоящее время бент-функции исследуются по всему миру очень активно. Тем не менее, в этой области остается множество открытых вопросов.

В данной курсовой работе рассматриваются основные свойства некоторого подкласса бент-функций, которые названы однородными бент-функциями. Однородные бент-функции были впервые определены в работе [1]. Изучению свойств данных функций посвящено большое число работ (см., например, [3],[5],[6],[8]).

Основные понятия и определения

2.1 Булевы функции

Пусть $\mathbb{E}_2 = \{0,1\}$. Через $\mathbb{E}_2^n$ обозначим множество всех двоичных векторов $\boldsymbol{x} = (x_1, \dots, x_n)$ длины n.

Определение 2.1. *Булевой функцией* от n переменных называется отображение из $\mathbb{E}_2^n$ в $\mathbb{E}_2$. Множество всех булевых функций от n переменных обозначим через $\mathcal{F}_n$.

Для записи значения булевой функции f на векторе $\boldsymbol{x} \in \mathbb{E}_2^n$ будем использовать следующее выражение

$$f(\boldsymbol{x}) = f(x_1, \dots, x_n).$$

Каждую булеву функцию от n переменных можно определить вектором ее значений длины 2^n.

Пусть $\oplus$ обозначает сложение по модулю 2.

Известно, что каждая булева функция однозначно может быть задана своей *алгебраической нормальной формой* (АНФ), а именно представлена в виде:

$$f(\boldsymbol{x}) = \left(\bigoplus_{k=1}^{n} \bigoplus_{i_1,\dots,i_k} a_{i_1,\dots,i_k} x_{i_1} \cdot \dots \cdot x_{i_k}\right) \oplus a_0$$

где при каждом k индексы $i_1, \dots, i_k$ различны и в совокупности пробегают все $k-$ элементные подмножества множества $\{1, \dots, n\}$.

Коэффициенты $a_{i_1,\dots,i_k}, a_0$ принимают значения 0 или 1. Часто в литературе АНФ называют *полиномом Жегалкина.*

Определение 2.2. *Алгебраической степенью функции* f (обозначение $\deg f$) называется число переменных в самом длинном мономе полинома Жегалкина (АНФ) функции f.

Функция называется *аффинной, квадратичной, кубической*, если ее степень равна соответственно 1, 2, 3.

Множество аффинных функций будем обозначать через $\mathcal{A}_n$.

Определение 2.3. Булева функция $f(x_1, x_2, \dots, x_n)$ называется симметрической, если при любой перестановке $\sigma \in S_n (S_n$ – группа перестановок натуральных чисел $1, 2, \dots, n)$ $f(x_1, x_2, \dots, x_n) = f\left(x_{\sigma(1)}, x_{\sigma(2)} \dots, x_{\sigma(n)}\right)$.

Определение 2.4. Скалярным произведением *векторов* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{E}_2^n$ называется сумма

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = u_1 v_1 \oplus \dots \oplus u_n v_n$$

Расстояние (Хэмминга) между двумя векторами $\boldsymbol{x}$ и $\boldsymbol{y}$ равно числу компонент, в которых они различаются, обозначим данную характеристику через dist($\boldsymbol{x}, \boldsymbol{y}$).

Определение 2.5. *Вес* $wt(f)$ *булевой функции* f определяется следующим равенством:

$$wt(f) = \#\{\boldsymbol{x} \in \mathbb{E}_2^n \mid f(x) = 1\}$$

Определение 2.6. Расстояние между булевой функцией $f \in \mathcal{F}_n$ и множеством аффинных функций $\mathcal{A}_n$ будем называть *нелинейностью функции* f и обозначать через N_f.

2.2 Бент-функции

Определение 2.7. Функция $f \in \mathcal{F}_n$ называется максимально-нелинейной функцией, если ее нелинейность N_f достигает максимально возможного для функций из $\mathcal{F}_n$ значения.

Определение 2.8. *Преобразованием Уолша-Адамара* булевой функции f называется целочисленная функция на $\mathbb{E}_2^n$, определяемая следующим равенством:

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

Для каждого $\boldsymbol{u} \in \mathbb{E}_2^n$ значение $W_f(\boldsymbol{u})$ называется *коэффициентом Уолша-Адамара.*

Нелинейность N_f произвольной функции f тесно связана с ее коэффициентами Уолша-Адамара, а именно:

$$N_f = 2^{n-1} - \frac{1}{2} \max_{\boldsymbol{v} \in \mathbb{E}_2^n} \left| W_f(\boldsymbol{v}) \right|$$

Очевидно, что чем меньше максимум модуля коэффициента $W_f(\boldsymbol{v})$, тем выше нелинейность функции f. В случае четного n максимально возможное значение нелинейности функции f (см. [2]) равно : $N_f = 2^{n-1} - 2^{\frac{n}{2}-1}$

Справедливо равенство Парсеваля (см. [2]):

$$\sum_{\boldsymbol{v} \in \mathbb{E}_2^n} \left(W_f(\boldsymbol{v}) \right)^2 = 2^{2n}.$$

Поскольку число всех коэффициентов $W_f(\boldsymbol{v})$ равно 2^n, из равенства следует, что максимум модуля коэффициента Уолша-Адамара не может быть меньше величины $2^{n/2}$.

Определение 2.9. Функция $f \in \mathcal{F}_n$ называется *бент-функцией* (bent function), если модуль каждого коэффициента Уолша-Адамара этой функции равен $2^{n/2}$.

Другими словами, $f-$ бент-функция, если максимум модуля $W_f(\boldsymbol{v})$ достигает своего минимально возможного значения. В силу равенства Парсеваля это имеет место, только если модули всех коэффициентов Уолша—Адамара этой функции совпадают и равны $2^{n/2}$. Таким образом, при четном n данное определение является эквивалентным определению максимально нелинейной функции.

Множество всех бент-функций от n переменных будем обозначать через $\mathcal{B}_n$. Поскольку коэффициенты Уолша-Адамара являются целыми рациональными числами, то при нечетном n бент-функций не существует.

Определение 2.10. Бент-функция называется *однородной*, если все одночлены ее алгебраической нормальной формы имеют одинаковые степени.

Пример:

$$f(x_1,x_2,x_3,x_4,x_5,x_6) = x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_2x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_4 \oplus x_1x_3x_5 \oplus$$
$$\oplus x_1x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_3x_4 \oplus x_2x_3x_6 \oplus x_2x_4x_5 \oplus x_2x_5x_6 \oplus x_3x_4x_5 \oplus x_3x_4x_6 \oplus$$
$$\oplus x_3x_5x_6 \oplus x_4x_5x_6$$

является однородной бент-функцией (обоснование приведено в Приложении А)

Определение 2.11. *Перестановкой* множества из n элементов называется любой упорядоченный набор из всех элементов этого множества, среди которых нет одинаковых.

Существует ровно $n!$ перестановок множества из n элементов, $|S_n| = n!$

Определение 2.12. *Группой инерции функции* f в группе S_n называется следующее множество

$$\mathcal{J}_{S_n}(f) = \{g \in S_n \mid f^g = f\}.$$

Введем на множестве $\mathcal{F}_n$ бинарное отношение: $f_1 \sim f_2$ в группе S_n, тогда и только тогда, когда существует такой элемент $g \in S_n$, что $f_1^g = f_2$. Бинарное отношение $\sim$ является отношением эквивалентности. Обозначим класс эквивалентности в группе постановок S_n , содержащий функцию f, через $\{f\}_{S_n}$.

Теорема 2.13. Для произвольной функции $f \in \mathcal{F}_n$ выполняется равенство:

$$\#\{f\}_{S_n} = \frac{\#S_n}{\#\mathcal{J}_{S_n}(f)}.$$

Свойства однородных бент-функций

Данное направление впервые было изучено в работе [1]. В ней дано определение однородной бент-функции и изучены свойства таких функций степени три от числа переменных, кратных шести. Данные исследования были продолжены в работе [8].

Рассмотрим булевы функции от n переменных. Известно несколько свойств однородных бент-функций.

Утверждение 1 [8]. Для любого четного $n > 4$ существует однородная бент-функция степени 3.

Утверждение 2 [6]. Для любого четного $n > 6$ не существует однородной бент-функции степени $\frac{n}{2}$.

Утверждение 3 [3]. Для любого четного $n > 8$ не существует однородной бент-функции степени $\frac{n}{2} - 1$.

Утверждение 4 [5]. Для любого $k \geq 0$, найдется наибольшее $N \leq \frac{n}{2}$ (n – четное), удовлетворяющее неравенству:

$$2^{N-1} > \binom{N+1}{0} + \binom{N+1}{1} + \cdots + \binom{N+1}{k+1}$$

для которых не существует однородных бент-функций от n переменных степени не менее $\frac{n}{2} - k$.

Утверждение 5 [5]. Если функция f является однородной бент-функцией степени $n-k$ (n – четное), тогда вес Хэмминга функции f удовлетворяет неравенству:

$$wt(f) \leq \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{k}$$

где $k, n > 0, k < n$.

Утверждение 6 [1]. Количество однородных бент-функции степени 3 в $\mathbb{E}_2^{6k}$ не меньше чем $30^k \cdot \binom{6k}{6}$, где $k = 1,2, \ldots$.

Утверждение 7. Для любых $f(\boldsymbol{x}), h(\boldsymbol{x})$: $\mathbb{E}_2^n \to \mathbb{E}_2$, для которых функция $g(\boldsymbol{x}) = f(\boldsymbol{x}) \oplus h(\boldsymbol{x})$ является симметрической, выполняется равенство $\mathcal{J}_{S_n}(f(\boldsymbol{x})) = \mathcal{J}_{S_n}\big(h(\boldsymbol{x})\big)$.

Доказательство

Введем обозначение: $f^{\sigma}(\boldsymbol{x}) = f\big(x_{\sigma(1)}, x_{\sigma(2)} \ldots, x_{\sigma(n)}\big)$.

Пусть $\sigma \in \mathcal{J}_{S_n}(f(\boldsymbol{x}))$. Тогда из $g = f \oplus h$ получим $g^{\sigma} = f^{\sigma} \oplus h^{\sigma}$. Так как функция g симметрическая и $\sigma \in \mathcal{J}_{S_n}(f(\boldsymbol{x}))$, тогда $g^{\sigma} = g$ и $f^{\sigma} = f$.

$g^{\sigma} = f^{\sigma} \oplus h^{\sigma}$, следовательно $g = f \oplus h^{\sigma}$, откуда следует $h^{\sigma} = g \oplus f = h$. Следовательно получили:

$$\mathcal{J}_{S_n}(f(\boldsymbol{x})) \subset \mathcal{J}_{S_n}\big(h(\boldsymbol{x})\big). \ (*)$$

Пусть $\sigma \in \mathcal{J}_{S_n}\big(h(\boldsymbol{x})\big)$. Тогда из $g = f \oplus h$ получим $g^{\sigma} = f^{\sigma} \oplus h^{\sigma}$. Функция g симметрическая и $\sigma \in \mathcal{J}_{S_n}(h(\boldsymbol{x}))$, тогда $g^{\sigma} = g$ и $h^{\sigma} = h$.

$g^{\sigma} = f^{\sigma} \oplus h^{\sigma}$, следовательно $g = f^{\sigma} \oplus h$, откуда следует $f^{\sigma} = g \oplus h = f$. Следовательно получили

$$\mathcal{J}_{S_n}(h(\boldsymbol{x})) \subset \mathcal{J}_{S_n}\big(f(\boldsymbol{x})\big). \quad (**)$$

Из (*),(**) получим $\mathcal{J}_{S_n}(f(\boldsymbol{x})) = \mathcal{J}_{S_n}\big(h(\boldsymbol{x})\big)$. ■

Утверждение 7 упрощает нахождение количества однородных бент-функций. Пусть функция $g -$ сумма всевозможных мономов степени k от n переменных, $f -$ однородная бент-функция степени k от n переменных и $h = f \oplus g$, h будем называть дополнением функции f. Очевидно, что g является симметрической функцией. Если f – однородная бент-функция, то по теореме 2.13 для нахождения количества однородных бент-функций степени k от n переменных нужна мощность группы инерции функции f. Если в АНФ f много мономов, то это усложнит нахождение ее мощности. Утверждение 7 показывает, что группы инерции у функции f и ее дополнения h совпадают. Тогда для упрощения расчетов можно искать мощность группы инерции функции h.

Матрица инцидентности (см.[1]) функции f показывает вхождение определенной переменной в соответствующий моном. Строки матрицы соответствуют переменным, а столбцы соответствуют мономам функции. На пересечении i-й строки и j-о столбца стоит 1 если i -ая переменная входит в j -й моном. *Матрица смежности функции* – это матрица инцидентности функции, умноженная на транспонированную матрицу инцидентности функции. Матрица смежности показывает частоту появления переменной или пары переменных в АНФ функции.

Эмпирическим путем было установлено, что для $k = 3, n = 6$ определители матриц смежности для всех однородных бент-функций совпадают.

Несмотря на большое количество исследований в данном области, на настоящий момент не получена точная верхняя оценка для степени однородной бент-функции. Известно следующее:

Предположение. Для любого $k \geq 2$ найдется такое $N \geq 3$, что однородная бент-функция степени k от n переменных существует при каждом четном $n > N$.

В ходе проведенных исследований была написана программа на платформе .NET языка C#, которая строит все однородные бент-функции от n переменных степени k, либо сообщает, что подобных функций не существует. Исходный код представлен в Приложении Б.

Программа работает в соответствии со следующим алгоритмом.

Алгоритм

1-шаг. Составляем всевозможные мономы степени k от n переменных. Перебираем все наборы двоичного представление чисел от 1 до $2^n - 1$. Находим вес каждого набора. Если $wt(\alpha) = k$, то добавляем моном, соответствующий α, в список мономов. Число таких мономов равно C_n^k.

2-шаг. Составляем всевозможные полиномы, с участием построенных мономов. Перебираем двоичные наборы длины C_n^k, где i –ый элемент набора показывает вхождение соответствующего монома в полином. Таких полиномов будет $2^{C_n^k}$

3-шаг. С помощью преобразования Уолша-Адамара проверяем какие из построенных функций являются однородными бент-функциями.

Утверждение 8. Сложность работы алгоритма:

$$2^{2(n+k)+\binom{n}{k}} \leq O(n,k) \leq 2^{4(n+k)+\binom{n}{k}}$$

Доказательство.

1) Сложность первого шага $L_1(n) = n2^n$

Так как наборов 2^n и для каждого набора перебираем все элементы для вычисления веса.

2) Сложность второго шага $L_2(n,k) = 2^{C_n^k}$
3) Для фиксированного $\boldsymbol{u} \in \mathbb{E}_2^n$, правая часть равенства имеет сложность 2^n, т.к. перебираются $\boldsymbol{v}$ – все наборы из $\mathbb{E}_2^n$.

Для вычисления всех $W_f(\boldsymbol{u})$ перебираем все наборы $\boldsymbol{u}$**.** Сложность третьего шага $2^{2(n+k)} \leq L_3(n,k) \leq 2^{4(n+k)}$.

Итоговая сложность: $O(n,k) \approx L_1(n) + L_2(n,k)L_3(n)$ =>

$$n2^n + 2^{2(n+k)+C_n^k} \leq O(n,k) \leq n2^n + 2^{4(n+k)+C_n^k}.$$

■

Замечание. Нижняя оценка достигается когда для n, k не существует однородных бент-функций степени k от n переменных. Верхняя оценка достигается, когда все полиномы являются однородными бент-функциями.

Заключение

Данная работа посвящена исследованию свойств однородных бент-функций. Приведены необходимые определения и известные результаты в данной области. Сформулированы основные известные свойства однородных бент-функций.

В ходе проведенных исследований доказано утверждение о совпадении групп инерции двух функций, сумма которых есть симметрическая функция. Данное утверждение позволяет значительно ускорить поиск однородных бент-функций и подсчета их количества, что обеспечивается проверкой условий для более простых функций, являющихся дополнениями к рассматриваемым.

Для поиска всех однородных бент-функций указанной степени с заданным количеством переменных, написана программа на платформе .NET языка C#, которая находит всех однородные бент-функции (если существуют) с указанным количеством переменных и указанной степени. Для реализованного алгоритма получены верхняя и нижняя оценки трудоемкости.

В качестве направлений дальнейших исследований могут быть выделены:

- построение однородных бент-функций степени больше трех;

- изучение свойств определителей матриц смежности однородных бент-функций.

Список литературы

[1] Chengxin Qu, Jennifer Seberry, Josef Pieprzyk, Homogeneous bent functions. Discrete Applied Mathematics 102(1-2): 133-139 (2000)

[2] Логачев О. А., Сальников А. А., Ященко В. В. Булевы функции в теории кодирования и криптологии. М.:МЦНМО,2012.-584 с.

[3] Meng Q., Zhang H., Yang M. C., Cui J. On the degree of homogeneous bent functions **.** Discrete Applied Mathematics 155(5): 665-669 (2007)

[4] Токарева Н.Н. Нелинейные булевы функции: бент-функции и их обобщения. 2011. ISBN: 978-3-8433-0904-2. 180 с.

[5] Meng Q., Zhang H., Yang M. C., Cui J. On the degree of homogeneous bent functions **.** IACR Cryptology ePrint Archive 2004: 284 (2004)

[6] Xia T, Seberrya J., Pieprzykb J., Charnes Ch. Homogeneous bent functions of degree n in $2n$ variables do not exist for $n > 3$ **.** Discrete Applied Mathematics 142(1-3): 127-132 (2004).

[7] O.S Rothaus, On "bent" functions **.** J. Comb. Theory, Ser. A 20(3): 300-305 (1976).

[8] Charnes Ch., Rotteler M., Beth Thomas. Homogeneous bent functions and Design, 2002 **.** Des. Codes Cryptography 26(1-3): 139-154 (2002)

Приложение А

Рассмотрим функцию

$$f(x_1,x_2,x_3,x_4,x_5,x_6) = x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_2x_5 \oplus x_1x_2x_6 \oplus x_1x_3x_4 \oplus x_1x_3x_5 \oplus$$

$$\oplus x_1x_4x_6 \oplus x_1x_5x_6 \oplus x_2x_3x_4 \oplus x_2x_3x_6 \oplus x_2x_4x_5 \oplus x_2x_5x_6 \oplus$$

$$\oplus x_3x_4x_5 \oplus x_3x_4x_6 \oplus x_3x_5x_6 \oplus x_4x_5x_6$$

Докажем, что данная функция является однородной бент-функцией.

Проверим является ли заданная функция однородной бент-функцией.

В данном примере $n = 6$.

1-шаг. Строим таблицу значений функции на всевозможных наборах.

Покажем на примере набора (101101):

$$f(1,0,1,1,0,1) = 1$$

Таким образом получим вектор значений данной функции: (00000001000101100001001101011110000101010011111001101110111 00000)

2-шаг. Для каждого набора вычисляем коэффициент Уолша-Адамара.

Приведем пример для набора (001010)

$$W_f(\boldsymbol{u}) = \sum_{\boldsymbol{v}\in\mathbb{E}_2^n} (-1)^{f(\boldsymbol{v})\oplus\langle \boldsymbol{u},\boldsymbol{v}\rangle} \text{ , где } \boldsymbol{u} \in \mathbb{E}_2^n$$

$$W_f(0,0,1,0,1,0) = (-1)^{\langle 001010,000000\rangle \oplus f(000000)} + (-1)^{\langle 001010,000001\rangle \oplus f(000001)}$$

$$+(-1)^{\langle 001010,000010\rangle \oplus f(000010)} + (-1)^{\langle 001010,000011\rangle \oplus f(000011)} + \cdots$$

$$\ldots. + (-1)^{\langle 001010,111111\rangle \oplus f(111111)} = 8$$

$$\left|W_f(0,0,1,0,1,0)\right| = 8 = 2^{\frac{6}{2}} = 2^3.$$

Таким образом получим значение Уолша-Адамара для всех наборов. Как видно, все модули коэффициентов Уолша-Адамара равны 2^3, а это означает что данная функция является однородной бент-функцией.

$x_1x_2\ x_3\ x_4\ x_5\ x_6$	$f(x_1,x_2,x_3,x_4,x_5,x_6)$	$W_f(x_1,x_2,x_3,x_4,x_5,x_6)$
0 0 0 0 0 0	0	8

0 0 0 0 0 1	0	8
0 0 0 0 1 0	0	8
0 0 0 0 1 1	0	8
0 0 0 1 0 0	0	8
0 0 0 1 0 1	0	8
0 0 0 1 1 0	0	8
0 0 0 1 1 1	1	-8
0 0 1 0 0 0	0	8
0 0 1 0 0 1	0	8
0 0 1 0 1 0	0	8
0 0 1 0 1 1	1	-8
0 0 1 1 0 0	0	8
0 0 1 1 0 1	1	-8
0 0 1 1 1 0	1	-8
0 0 1 1 1 1	0	8
0 1 0 0 0 0	0	8
0 1 0 0 0 1	0	8
0 1 0 0 1 0	0	8
0 1 0 0 1 1	1	-8
0 1 0 1 0 0	0	8
0 1 0 1 0 1	0	-8
0 1 0 1 1 0	1	8
0 1 0 1 1 1	1	-8
0 1 1 0 0 0	0	8
0 1 1 0 0 1	1	8
0 1 1 0 1 0	0	-8
0 1 1 0 1 1	1	-8
0 1 1 1 0 0	1	-8
0 1 1 1 0 1	1	-8
0 1 1 1 1 0	1	-8
0 1 1 1 1 1	0	8
1 0 0 0 0 0	0	8
1 0 0 0 0 1	0	8
1 0 0 0 1 0	0	8
1 0 0 0 1 1	1	-8
1 0 0 1 0 0	0	8
1 0 0 1 0 1	1	8
1 0 0 1 1 0	0	-8
1 0 0 1 1 1	1	-8
1 0 1 0 0 0	0	8
1 0 1 0 0 1	0	-8
1 0 1 0 1 0	1	8
1 0 1 0 1 1	1	-8

1 0 1 1 0 0	1	-8
1 0 1 1 0 1	1	-8
1 0 1 1 1 0	1	-8
1 0 1 1 1 1	0	8
1 1 0 0 0 0	0	8
1 1 0 0 0 1	1	-8
1 1 0 0 1 0	1	-8
1 1 0 0 1 1	0	8
1 1 0 1 0 0	1	-8
1 1 0 1 0 1	1	-8
1 1 0 1 1 0	1	-8
1 1 0 1 1 1	0	8
1 1 1 0 0 0	1	-8
1 1 1 0 0 1	1	-8
1 1 1 0 1 0	1	-8
1 1 1 0 1 1	0	8
1 1 1 1 0 0	0	8
1 1 1 1 0 1	0	8
1 1 1 1 1 0	0	8
1 1 1 1 1 1	0	8

Приложение Б

```
using System;
using System.Linq;
using Org.BouncyCastle.Math;
namespace ConsoleApplication1
{
    class Program
    {
        static string[] temp;
        static int count = 0;
        static string[] all_monoms;
        static int variable,degree;

        static void Main(string[] args)
        {
            Console.Write("n:");
            variable = int.Parse(Console.ReadLine());
            Console.Write("k:");
            degree = int.Parse(Console.ReadLine());



            all_monoms = count_of_monoms(variable,degree);
            temp= new string[all_monoms.Length];
            BigInteger max_step = new BigInteger("2", 10).Pow(all_monoms.Length);
            BigInteger k = new BigInteger("1", 10);
            BigInteger step=k;
            while (!step.Equals(max_step))
```

```
        {
            check_function(step);
            step =step.Add(k);

        }

        Console.WriteLine("Количество однородных бент-функций:{0}",count);
        Console.ReadLine();
    }

    static string[] count_of_monoms(int peremena,int degree)
    {
        string[] monoms; int count=0;
        string s,s1;
        for (int z = 1; z < Math.Pow(2, variable); z++)
        {
            s = Binary(z);
            if (s.Where(x => x == '1').Count() == degree)
            {
                ++count;
            }
        }
        monoms = new string[count];
        int j = 0;
        for (int z = 1; z < Math.Pow(2, peremena); z++)
        {
            s = Binary(z);

            if (s.Where(x => x == '1').Count() == degree)
            {
                s1 = "";
                for (int i = 0; i < s.Length; i++)
                {
                    if (s[i] == '1') s1 += (i + 1).ToString()+".";
                }
                monoms[j] = s1;  ++j;
            }
        }

        return monoms;

    }

    static void check_function(BigInteger k)
    {

        int check; string function_monoms = "";

        string s = Binary_bigint(k);


        for (int i = 0; i < s.Length; i++)
        {
            if (s[i] == '1') function_monoms += all_monoms[i] + " ";

        }

       string[] temp_1 = function_monoms.Split(' ');
        string[] temp_2 = new string[temp_1.Length - 1];
        for (int i = 0; i < temp_1.Length - 1; i++)
        { temp_2[i] = temp_1[i]; }
        temp = null;
        temp = temp_2;
```

```
            int[] coefficients_Uolsha_Adamar = new int[(int)Math.Pow(2,variable)];

            for (int i = 0; i < Math.Pow(2, variable); i++)
            {
                check = converting_Uolsha_Adamar(i);
                if (Math.Abs(check) != Math.Pow(2, variable / 2)) return;
                coefficients_Uolsha_Adamar[i] = check;
            }


            ++count;
            Console.Write("Найдена однородная бент-функция с мономами:");
            foreach (string x in temp)
                Console.Write(" {0}", x.re);

            Console.WriteLine();

        }


        static int skalyar(string u, string v)
        {
            int sum = 0;
            for (int i = 0; i < u.Length; i++)
            {
                sum += (Convert.ToInt32(u[i]) - 48) * (Convert.ToInt32(v[i]) - 48);
            }
            return sum;
        }


        static string Binary(int a)
        {
            string v = "";
            string s = Convert.ToString(a, 2);
            for (int i = 0; i < variable - s.Length; i++)
            {
                v += "0";
            }

            v += s;
            return v;
        }

        static string Binary_bigint(BigInteger a)
        {
            string s1 = "";
            string s = a.ToString(2);
            for (int i = 0; i < all_monoms.Length - s.Length; i++)
            {
                s1 += "0";
            }

            s1 += s;
            return s1;
        }
        static int converting_Uolsha_Adamar(int y)
        {
            string v = Binary(y);
            string u; int sum = 0,temp;
            for (int x = 0; x < Math.Pow(2,variable); x++)
            {
                u = Binary(x);
```

```
                temp = (int)Math.Pow(-1, ((skalyar(u, v) + function(u)) % 2));

                sum += temp;
            }

            return sum;
        }


        static int function(string s)
        {
            int[] x = new int[variable+1];
            int multiply=1,f=0;
            string[] number_variable;
            int[] monomi = new int[degree];
            int[] z = new int[degree];
            for (int i = 1; i < variable+1; i++)
            {
               x[i] = Convert.ToInt32(s[i - 1]) - 48;
            }


            for (int t = 0; t < temp.Length; t++)
            {
                number_variable = temp[t].Split('.');

                for (int j = 0; j < degree; j++)
                {
                  z[j]= Convert.ToInt32(number_variable[j]);
                }

                for (int i = 0; i < degree; i++)
                {
                    multiply *= x[z[i]];
                }

                f += multiply;
                multiply = 1;


            }

            f = f % 2;

            return f;
        }


    }
}
```