Некоторые приложения двоичной лунной арифметики

В.В. Данг[1], Н.Л. Додонова[2], М.В. Додонов[2], С.Ю. Корабельщикова[3]

[1]Государственный политехнический институт Хошимин, 268 Ly Thuong Kiet, dist 10, Hochiminh city, Vietnam
[2]Самарский национальный исследовательский университет им. академика С.П. Королева, Московское шоссе 34А, Самара, Россия, 443086
[3]Северный (Арктический) федеральный университет имени М.В. Ломоносова, наб. Северной Двины 17, Архангельск, Россия, 163007

Аннотация. В общем виде задача извлечения корня n–й степени из языка формулируется следующим образом: для заданного языка A и заданного $n \in N$ требуется найти все языки B такие, что $A = B^n$. Эта теоретическая задача тесно связана с практической задачей декодирования сообщений, где требуется найти разбиение кодового сообщения на элементарные коды, соответствующие символам алфавита источника. Ранее мы получили решение задачи извлечения корня n–й степени для языков специального вида, а именно содержащих всевозможные слова длины от $n \cdot n_1$ до $n \cdot n_2$ $(n_1 \le n_2)$. Рассматриваемая задача была сведена к задаче о рюкзаке и решена методом программной реализации предложенных алгоритмов. Были получены количественные оценки числа корней n–й степени, при различных значениях n и k, где k – мощность множества $\{n_1, n_1+1, \ldots, n_2\}$. При n=2 последовательность количества квадратных корней совпала с последовательностью, опубликованной на сайте онлайн энциклопедии целочисленных последовательностей http://oeis.org/A191701, и представляющей собой число двоичных чисел длины k, квадраты которых не содержат нулей в двоичной лунной арифметике. В данной статье нами установлено и теоретически обосновано соответствие между операциями в двоичной лунной арифметике и в кольце многочленов с целыми коэффициентами. На основе установленного соответствия разработан алгоритм, который позволяет решать, как частную задачу нахождения корней из языка, так и более общую задачу о рюкзаке с применением операций в лунной двоичной арифметике. Проведено сравнение программной реализации предложенного алгоритма с известными классическими вариантами решения задачи о рюкзаке.

1. Введение

Понятие «лунная арифметика» было введено американскими учеными D. Applegate, M. Lebrun и N. J. A. Sloane в работе [1], причем изначально они использовали термин «dismal arithmetic», впоследствии заменив его на менее пессимистичный «lunar arithmetic».

В т.н. лунной арифметике операции сложения и умножения над числами заменяются на операции нахождения максимума и минимума соответственно. Например, 2+7=7, а 2·7=2. С многозначными числами производят операцию сложения поразрядно, то есть выбирают максимум

в каждом разряде. Таким образом, 6179 + 348 = 6379. При умножении многозначных чисел первое число поразрядно умножается (в смысле, выбирается минимум) на каждый разряд второго числа, после чего полученные значения складываются, то есть в каждом разряде(столбце) выбирается максимум:

			6	1	7	9
		x		3	4	8
			6	1	7	8
		4	1	4	4	
+	3	1	3	3		
	3	4	6	4	7	8

Подробнее об особенностях лунной арифметики можно узнать в [1]. Нас же заинтересовал двоичный случай, когда операции над числами производятся по сформулированным выше правилам, но цифры принадлежат множеству {0, 1}.

Ранее в работах [2,3] нами рассматривалась задача извлечения всех корней из языков специального вида. В общем виде задача извлечения корня n–й степени из языка формулируется следующим образом: для заданного языка A и заданного $n \in N$ требуется найти все языки B такие, что $A = B^n$. При этом язык B называется корнем n-й степени из языка A. Эта теоретическая задача тесно связана с практической задачей декодирования сообщений, где требуется найти разбиение кодового сообщения на элементарные коды, соответствующие символам алфавита источника.

Введем необходимые обозначения. Пусть M – конечное подмножество множества натуральных чисел, Σ - некоторый алфавит, возможно и бесконечный. Будем рассматривать языки вида $\Sigma(M)$, содержащие всевозможные слова длины i над алфавитом Σ, где $i \in M$. Множество M, определяющее язык $\Sigma(M)$, будем называть множеством индексов. Решение задачи извлечения всех корней было получено для языков $A=\Sigma[t_1;t_2]$, содержащих всевозможные слова длины от t_1 до t_2 $(t_1 \le t_2)$. Рассмотрим два примера.

Пример 1. Из языка $A=\Sigma[2;14]$ извлекаются 5 квадратных корней с множествами индексов {1, 2, 3, 4, 5, 6, 7}, {1, 2, 3, 4, 6, 7}, {1, 2, 4, 5, 6, 7}, {1, 2, 4, 6, 7} и {1, 2, 3, 5, 6, 7}.

Рассмотрим, например, язык $B=\Sigma(M)$, где M={1, 2, 4, 6, 7}. Покажем, что $B^2=A$. Операция умножения здесь – конкатенация. Так как язык B содержит все слова длины 1, то язык B^2 содержит конкатенацию слов длины 1, то есть все слова длины 2. Аналогично язык B^2 содержит все слова длины 4, 8, 12 и 14. Все слова длины 3 можно получить конкатенацией слов длины 1 и 2 (1, $2 \in M$), длины 5 – конкатенацией слов длины 1 и 4 (1, $4 \in M$), и так далее для оставшихся натуральных чисел из промежутка $[2;14]$.

Пример 2. Найдем все корни третьей степени из языка $A=\Sigma[27;42]$.

Очевидно, что множество M_1={9, 10, 11, 12, 13, 14} задает корень третьей степени $B=\Sigma[9;14]$ из языка A. Аналогично примеру 1, можно проверить, что множество M_2 = {9, 10, 13, 14} задает еще один корень третьей степени из языка A. Поэтому корнями будут еще два языка с множествами индексов M_3={9, 10, 11, 13, 14} и M_4={9, 10, 12, 13, 14}.

В общем случае, для того, чтобы из языка $A=\Sigma[t_1;t_2]$ корень n–й степени извлекался, необходимо и достаточно, чтобы t_1 и t_2 делились на n. Пусть M – подмножество множества натуральных чисел $\{n_1, n_1 + 1, \ldots, n_2\}$, где $n_1=t_1/n$, $n_2=t_2/n$. Язык $B=\Sigma(M)$, содержащий все слова длины a_j, $a_j \in M$, является корнем n–й степени из языка A тогда и только тогда, когда выполняется условие:

$$(\forall i)\ (n \cdot n_1 \le i \le n \cdot n_2 \rightarrow i = a_1 + a_2 + \ldots + a_n), \qquad (1)$$

где $a_1, a_2, \ldots, a_n$ – некоторые элементы из M (не обязательно различные).

Задача проверки условия (1) эквивалентна частному случаю задачи о неограниченном рюкзаке, для которого известно решение методом динамического программирования, находящее ответ за

полиномиальное время. Условие (1) очевидно выполняется для множества $\{n_1, n_1 + 1, ..., n_2\}$ и соответствует тривиальному корню $B=\Sigma[n_1,n_2]$, содержащему всевозможные слова длины от n_1 до n_2. Проверка условия (1) для других подмножеств множества $\{n_1, n_1 + 1, ..., n_2\}$ была проведена методом программной реализации известных алгоритмов решения задачи о неограниченном рюкзаке. В таблице 1 приведены результаты работы программы, то есть количество корней n–й степени из языков вида $A=\Sigma[t_1;t_2]$ для некоторых n и k ($k = n_2 – n_1 + 1$). При k от 1 до 3 значения равны 1.

Таблица 1. Количество корней n–й степени при различных k.

n \ k	4	5	6	7	8	9	10	11	12	13	14
2	1	2	3	5	9	15	28	50	95	174	337
3	1	2	4	7	13	25	49	95	185	365	721
4	1	2	4	8	15	29	57	113	225	447	889
5	1	2	4	8	16	31	61	121	241	481	961
6	1	2	4	8	16	32	63	125	249	497	993
7	1	2	4	8	16	32	64	127	253	505	1009

Последовательность, полученная при n = 2, была обнаружена на сайте онлайн энциклопедии целочисленных последовательностей http://oeis.org/A191701, которая ссылается, в свою очередь, на работу [1]. Эта последовательность была получена D. Applegate, M. Lebrun и N. J. A. Sloane для k от 1 до 41:

1, 1, 1, 1, 2, 3, 5, 9, 15, 28, 50, 95, 174, 337, 637, 1231, 2373, 4618, 8974, 17567, 34387, 67561,… (1)

В данной работе мы установим причину этого совпадения, интерпретируем полученные нами результаты в терминах лунной арифметики, а также применим установленное соответствие для решения задачи о рюкзаке.

2. Двоичная лунная арифметика и операции над многочленами

Пусть исходное множество {0,1}. В двоичном случае лунному сложению (выбору максимального) соответствует дизъюнкция, а умножению (выбору минимального) - конъюнкция. Рассмотрим лунные операции над многозначными двоичными числами. При сложении в каждом разряде выбираем максимум: 1011+101=1111. При умножении действуем по правилу умножения многозначных чисел в лунной арифметике, приведенному выше.

			1	0	1	1
		x		1	0	1
			1	0	1	1
	+	0	0	0	0	
+	1	0	1	1		
	1	0	1	1	1	1

Таким образом, 1011· 101= 101111 (при сложении по столбцам находим максимум).

Видим, что исходный вектор 1011 при умножении на разряд 1 сохраняется в неизменном виде, сдвигаясь влево на позицию этого разряда. Аналогичная ситуация происходит при умножении в кольце многочленов $Z[x]$. Если исходный многочлен $f(x)$ умножить на x^i, то все его коэффициенты останутся неизменными, только справа допишутся i нулей, что соответствует сдвигу вектора коэффициентов $f(x)$ на i позиций влево. Сопоставим двоичному вектору 1011 многочлен x^3+x+1, а вектору 101 многочлен x^2+1. Получим:

$$(x^3+x+1)\cdot 1= x^3+x+1=1011$$
$$(x^3+x+1)\cdot x^2= x^5+x^3+ x^2=101100$$

Сложив коэффициенты при одинаковых степенях, имеем: $x^5+2x^3+x^2+x+1=102111$. В кольце многочленов, при подсчете коэффициентов при одинаковых степенях, производим обычное сложение, а в лунной двоичной арифметике если есть 1 в разряде, то получим 1, если все нули, то 0. Таким образом, справедлива теорема.

Теорема 1. Сопоставим двоичному вектору $(a_n,a_{n-1},\dots,a_1,a_0)$ многочлен $a_nx^n+a_{n-1}x^{n-1}+\dots+a_1x+a_0$. При заданном таким образом отображении, операциям в двоичной лунной арифметике соответствуют операции над многочленами в кольце Z[x] с последующей заменой ненулевых коэффициентов на 1.

Доказательство.

Без ограничения общности, рассмотрим два двоичных вектора $a=(a_n,a_{n-1},\dots,a_1,a_0)$ и $b=(b_n,b_{n-1},\dots,b_1,b_0)$ одинаковой длины. В противном случае допишем к более короткому вектору нули слева. Тогда $a+b=(a_n\vee b_n, a_{n-1}\vee b_{n-1},\dots,a_1\vee b_1, a_0\vee b_0)$ и 0 будет в i-той позиции тогда и только тогда, когда $a_i= b_i=0$.

С другой стороны, для многочленов $a(x)=a_nx^n+a_{n-1}x^{n-1}+\dots+a_1x+a_0$ и $b(x)=b_nx^n+b_{n-1}x^{n-1}+\dots+b_1x+b_0$ $a(x)+ b(x)= (a_n+ b_n) x^n+(a_{n-1}+b_{n-1})x^{n-1}+\dots+ (a_1+b_1)x+a_0+b_0$. Так как исходные векторы a и b двоичные, то коэффициенты многочлена $a(x)+ b(x)$ принадлежат множеству {0,1,2}, причем коэффициент при x^i равен 0 тогда и только тогда, когда $a_i= b_i=0$. Поэтому после замены всех ненулевых коэффициентов на 1, вектор коэффициентов многочлена $a(x)+ b(x)$ совпадет с вектором $a+b$.

Соответствие результатов при умножении доказывается аналогично.

3. Двоичная лунная арифметика и задача нахождения всех корней из языка

Далее интерпретируем содержательно последовательность (1). В терминологии авторов, она представляет собой количество двоичных векторов длины k, лунные квадраты которых состоят из одних единиц. Лунный квадрат – это результат умножения вектора на себя.

Пример 3. Лунные квадраты следующих пяти векторов длины k = 7: 1111111, 1111011, 1101111, 1101011 и 1110111 равны $1111111111111=1^{(13)}$, поэтому седьмой член в последовательности (1) равен 5. Для всех остальных двоичных векторов длины 7 лунный квадрат будет содержать нули.

Перейдем к операциям над многочленами.

Например, четвертому вектору a=1101011 сопоставим многочлен $a(x)=x^6 + x^5 + x^3 + x + 1$. Умножив его сам на себя, получим: $(x^6 + x^5 + x^3 + x + 1)(x^6 + x^5 + x^3 + x + 1) = c_{12}x^{12} + c_{11}x^{11} + c_{10}x^{10} + c_9x^9 + c_8x^8 + c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$.

По установленному в теореме 1 закону, будем считать ненулевой коэффициент равным 1.

$$c_0= a_0\cdot a_0=1,$$
$$c_1= a_0\cdot a_1+ a_1\cdot a_0=1+1=1,$$
$$c_2= a_0\cdot a_2+ a_1\cdot a_1+ a_2\cdot a_0=0+1+0=1, \text{ и так далее.}$$

Получим $c_0= c_1=\dots= c_{12}=1$.

Степень x^i присутствует в произведении $a(x)\cdot a(x)$ тогда и только тогда, когда i можно представить в виде суммы степеней сомножителей, то есть в виде суммы степеней многочлена $a(x)$. Для некоторых, не обязательно различных, j и k выполняется: $i=j+k$, где $j, k\in\{0, 1, 3, 5, 6\}$.

В многочлене $a(x)\cdot a(x)$, все коэффициенты c_i ненулевые, значит любое i от 0 до 12 можно представить в виде суммы двух (не обязательно различных) чисел из {0, 1, 3, 5, 6}.

Вспомним, что к этому же вопросу сводится задача извлечения всех корней из языка $A=\Sigma[t_1;t_2]$. В примере 1 из языка $A=\Sigma[2;14]$ извлекаются 5 квадратных корней с множествами индексов {1, 2, 3, 4, 5, 6, 7}, {1, 2, 3, 4, 6, 7}, {1, 2, 4, 5, 6, 7}, {1, 2, 4, 6, 7} и {1, 2, 3, 5, 6, 7}.

Сопоставим этим подмножествам множества {1, 2, 3, 4, 5, 6, 7} их двоичный характеристический вектор (α_1, α_2, …, α_7), и получим 5 двоичных векторов длины 7 из примера 3.

Дадим интерпретацию полученных в таблице 1 результатов в терминах лунной арифметики. Не сложно проверить, что вторая последовательность числа кубических корней из языков вида $A=\Sigma[t_1;t_2]$ совпадает с количеством двоичных чисел длины k, чьи лунные кубы состоят из одних единиц. В частности, при $k = 6$ лунные кубы четырёх векторов: 110011, 111011, 110111 и 111111 дают единичный вектор (он будет длины 16). Эти векторы являются характеристическими для множеств индексов из примера 2.

В общем случае справедлива следующая теорема.

Теорема 2. Пусть Σ – произвольный алфавит. Язык вида $B=\Sigma(M)$, где M – подмножество множества $\{n_1, n_1 + 1, ..., n_2\}$, является корнем n–й степени из языка $A=\Sigma[n\cdot n_1;n\cdot n_2]$ тогда и только тогда, когда соответствующий подмножеству M двоичный вектор длины $k = n_2 – n_1 + 1$ имеет n–ю степень, состоящую только из единиц в лунной двоичной арифметике.

Теорема 2 даёт еще один способ проверить, является ли подмножество M множеством индексов корня из языка вида $A=\Sigma[t_1;t_2]$.

4. Приложение лунной арифметики к решению задачи о рюкзаке

Классическую задачу о рюкзаке в общем виде можно сформулировать следующим образом: из заданного множества предметов со свойствами «стоимость» и «вес» требуется выбрать некоторое подмножество предметов таким образом, чтобы получить максимальную суммарную стоимость при соблюдении ограничения на суммарный вес.

Задача о неограниченном рюкзаке – обобщение классической задачи, когда любой предмет может быть взят любое количество раз. Рассмотрим частный случай этой задачи, когда стоимость предмета – число натуральное, и равное его весу. Поэтому в дальнейшем стоимость предметов можно не учитывать, а работать только с весами предметов. Также добавим условие, что обязательно нужно взять ровно M предметов. Сформулируем данную задачу на математическом языке.

Задача 1. Дано N предметов, W – вместимость рюкзака, M – количество предметов, которое требуется взять, w_1, w_2, …, w_N – натуральные веса соответствующих предметов. Нужно найти набор величин x_1, x_2, …, x_N, где x_i равно количеству взятых предметов данного вида в набор, и такой что:

1. $x_1 w_1 + ... + x_N w_N \le W$;
2. $x_1 + ... + x_N = M$;
3. $x_1 w_1 + ... + x_N w_N$ максимальна.

Замечание. Заранее можно договориться, что все веса предметов различны между собой. Если же есть какие–то предметы с одинаковым весом, то будем всегда брать из них только какой–то определенный. Такое допущение ничего не изменит в задаче 1, так как любой предмет можно брать любое количество раз.

Алгоритм

Шаг 1. По набору весов $\hat{w}= \{w_1, w_2, …, w_N\}$ составим многочлен $y = x^{w_1} + x^{w_2} + \cdots + x^{w_N}$

Шаг 2. Сопоставим многочлену y двоичный вектор его коэффициентов u. Тогда в векторе u для всех u_i выполняется следующее условие: u_i=1, если $i \in \hat{w}$; иначе u_i=0.

Шаг 3. Возведем вектор u в степень M в лунной двоичной арифметике. Получим двоичный вектор $z = u^M$.

В векторе z все ненулевые z_k обозначают, что существует какой–то набор ровно из M предметов, не обязательно различных, сумма весов которых равна k. Это утверждение обосновывает следующий шаг.

Шаг 4. Выбираем максимальный k, удовлетворяющий условиям: $k \le W$ и $z_k \ne 0$.

Шаг 5. Методом «обратный ход» находим линейную комбинацию $x_1 w_1 + \ldots + x_N w_N = k$.

Пример 4. Пусть даны 4 предмета весов 2, 3, 6, 7. Вместимость рюкзака $W=17$ и требуется взять ровно 3 предмета.

Шаг 1. Составим многочлен $y = x^2 + x^3 + x^6 + x^7$.

Шаг 2. Сопоставим многочлену y двоичный вектор его коэффициентов $u=(11001100)$.

Шаг 3. Найдем u^3. Выполнив операции в лунной двоичной арифметике, получим:

$u^2 = (111011101110000)$, или (по теореме 1) $y^2 = x^{14} + x^{13} + x^{12} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4$.

$u^3 = (111111111111111000000) = (1^{16}0^6)$.

Шаг 4. $k=17$.

Шаг 5. 17=10+7; 10=7+3. Значит, 17=7+3+7, то есть надо взять 2 предмета веса 7 и один веса 3.

Есть второе решение 17=14+3; 14=7+7. Получим 17=7+7+3, что по сути совпадает с первым решением.

Оценим асимптотическую сложность данного алгоритма. Обозначим за $w_{max} = max\{ w_i \mid 1 \le i \le N\}$ – вес самого тяжелого предмета, то есть максимальную степень многочлена y. Тогда $w_{max} \cdot M$ – максимальная степень многочлена $y^M = z$. Возведение в степень может быть реализовано наивным образом, что предполагает последовательное вычисление степеней y от 2 до M, либо с использованием приема «бинарное возведение в степень», позволяющего возводить любое число в степень n за $O(log\ n)$ умножений вместо n умножений при обычном последовательном умножении. С использованием этого приема, асимптотическая сложность данного алгоритма равна $O((w_{max} \cdot M)^2 \cdot log\ (M))$, где $(w_{max} \cdot M)^2$ – алгоритмическая сложность умножения двух многочленов, $log\ M$ – алгоритмическая сложность бинарного возведения в степень.

Для умножения двух многочленов можно применить быстрое преобразование Фурье (БПФ) [4, 5]. Для умножения двух многочленов в двоичной лунной арифметике, на основании теоремы 1, достаточно перемножить многочлены над стандартным комплексным полем и, после получения результирующего многочлена, изменить коэффициенты, которые больше единицы, на единицу. Умножение многочленов в двоичной лунной арифметике с помощью БПФ имеет асимптотическую сложность $O(w_{max} \cdot log(w_{max}))$. Тем самым, получилось улучшить асимптотическую сложность исходного алгоритма до $O(w_{max} \cdot log(w_{max}) \cdot log(M))$. Отметим также, что БПФ может быть реализовано с использованием параллельного алгоритма из [6].

5. Сравнительный анализ решений задачи о рюкзаке

Сравнительный анализ времени выполнения программ среди различных решений задачи 1 о рюкзаке был проведен магистром 2 курса по направлению 01.04.02 «Прикладная математика и информатика» Чесноковым А. И. Тестирование проводилось на случайных многочленах степени N, которые требуется возвести в M–ю степень. Приведем ограничения на N и M в тестах:

тест 1 – N = 500, M = 500;
тест 2 – N = 500, M = 1000;
тест 3 – N = 1000, M = 500;
тест 4 – N = 1000, M = 1000;
тест 5 – N = 1500, M = 500;
тест 6 – N = 1500, M = 1000;
тест 7 – N = 2000, M = 500;
тест 8 – N = 2000, M = 1000.

Были рассмотрены 7 различных решений задачи 1 о рюкзаке [7,8].

– Решение A – решение с помощью динамического программирования;

– Решение B – решение с помощью динамического программирования, используя битовые маски;

– Решение C – решение с помощью произведения многочленов в лунной арифметике. Произведение многочленов производится с помощью быстрого преобразования Фурье. В степень *M* многочлен возводится наивным образом;

– Решение D – решение с помощью произведения многочленов в лунной арифметике. Произведение многочленов производится с помощью быстрого преобразования Фурье. В степень *M* многочлен возводится с помощью бинарного возведения в степень;

– решение E – решение с помощью произведения многочленов в лунной арифметике. Произведение многочленов производится с помощью быстрого преобразования Фурье. В степень *M* многочлен возводится с помощью бинарного возведения в степень. Используется параллельный алгоритм БПФ [6]. Вычисления производятся на 2, 4 или 8 процессорах.

Тестирование проводилось на кластере САФУ, имеющем следующие характеристики:

– 20 вычислительных узлов;

– на каждом узле установлено 2 десятиядерных процессора Intel Xeon и 64 ГБ ОЗУ;

– на восьми узлах дополнительно установлены математические сопроцессоры Intel Xeon Phi 5110P;

– внутренняя компьютерная сеть для вычислений: Infiniband 56 Гб/с;

– сетевая файловая система FEFS (Fujitsu Exabyte File System) объёмом более 50 ТБ и пропускной способностью 1,67 ГБ/с (13,36 Гб/с);

– производительность кластера на CPU в тесте LINPACK 8,02 Tflops; на CPU+Xeon Phi 7,68 Tflops, совокупная 15,7 Tflops;

Результаты времени выполнения программ представлены в таблице 2.

Таблица 2. Результаты тестирования.

Тест	Решение A, сек.	Решение B, сек.	Решение C, сек.	Решение D, сек.	Решение E 2 пр., сек.	Решение E 4 пр., сек	Решение E 8 пр., сек
1	46.780	0.340	14.800	0.180	0.153	0.137	0.131
2	187.260	1.760	62.300	0.380	0.319	0.275	0.264
3	192.670	1.250	32.990	0.370	0.316	0.277	0.265
4	772.610	5.610	137.330	0.830	0.657	0.585	0.550
5	445.530	1.820	58.310	1.000	0.654	0.571	0.540
6	1793.230	8.160	268.720	1.900	1.476	1.246	1.244
7	779.010	3.280	77.380	0.910	0.689	0.576	0.566
8	3133.810	13.120	322.010	1.940	1.512	1.301	1.241

По таблице 2 можно сделать вывод, что более быстрым, по сравнению с остальными решениями, является решение задачи 1 с использованием БПФ в двоичной лунной арифметике и бинарного возведения в степень. Параллельный алгоритм БПФ также во всех тестах дает заметное ускорение, растущее при увеличении числа потоков (в некоторых случаях незначительно).

6. Заключение

Подведем итог полученным результатам. В теоретической части работы нами установлено соответствие между операциями в двоичной лунной арифметике и в кольце многочленов с целыми коэффициентами (теорема 1). Также установлено соответствие между двоичными числами x длины k, для которых x^n не содержит нулей в двоичной лунной арифметике, и множествами индексов корней n-й степени из языка специального вида (теорема 2). Обобщением задачи извлечения корня n-й степени из языка специального вида является математическая модель частного случая задачи о

неограниченном рюкзаке (задача 1). Разработан алгоритм решения этой задачи на основе лунной двоичной арифметики.

Программная реализация предложенного алгоритма была проведена в нескольких вариантах, с применением оптимизационных методов вычислений и параллельных технологий. Таблица 2 позволяет сравнить полученное авторами решение с другими известными решениями (решение А и решение В) задачи о неограниченном рюкзаке.

Как известно, задача о рюкзаке относится к классу *NP*–полных задач. Это означает, что не существует полиномиального алгоритма для получения точного результата (решения). Приведенные в таблице 2 результаты демонстрируют, что, несмотря на этот пессимистический факт, в некоторых случаях решение такой задачи может быть получено за секунды.

Результаты, полученные в ходе данного исследования, могут быть применены для решения многих прикладных задач, сводящихся к задаче о рюкзаке. В частности, с использованием этого метода в работе [9] была решена задача оценки числа различных циклических кодов с заданными параметрами. Некоторые дополнительные алгоритмы были предложены в работах [10,11].

7. Литература

[1] Applegate, D. Dismal Arithmetic / D. Applegate, M. LeBrun, N.J.A. Sloane, 2011. – 33 p.
[2] Корабельщикова, С.Ю. О первообразных корнях из языков специального вида / С.Ю. Корабельщикова, А.И. Чесноков, А.Г. Тутыгин // Сборник трудов IX международной конференции «Дискретные модели в теории управляющих систем», 2015. – С. 116-118.
[3] Melnikov, B.F. On the task of extracting the root from the language / B.F. Melnikov, S.Yu. Korabelshchikova, V.N. Dolgov // International Journal of Open Information Technologies. – 2019. – Vol. 7(3). – P. 1-6.
[4] Нуссбаумер, Г. Быстрое преобразование Фурье и алгоритмы вычисления сверток – Москва: Радио и связь, 1985.
[5] Корабельщикова, С.Ю. Алгоритмы лунной арифметики и быстрого преобразования Фурье в задаче о рюкзаке / С.Ю. Корабельщикова, А.И. Чесноков // Эвристические алгоритмы и распределенные вычисления. – 2015. – Т. 2, № 3. – С. 41-49.
[6] Воеводин, В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин – СПб.: БХВ-Петербург, 2002.
[7] Pisinger, D. Knapsack problems, 1995 [Electronic resource]. – Access mode: http://www.diku.dk/users/pisinger/95–1.pdf.
[8] Martelo, S. Knapsack problems / S. Martelo, P. Toth – Wiley, 1990.
[9] Корабельщикова, С.Ю. О числе различных циклических кодов заданной длины / С.Ю. Корабельщикова, А.И. Чесноков // Вектор науки ТГУ. – 2013. – Т. 4, № 26. – С. 25-26.
[10] Korabelshchikova, S.Y. Linear codes and some their applications / S.Y. Korabelshchikova, L.V. Zyablitseva, B.F. Melnikov, S.V. Pivneva // Journal of Physics: Conference Series electronic edition, 2018. – P. 012174.
[11] Мельников, Б.Ф. Алгоритмы получения числа помехоустойчивых кодов общего и специального вида / Б.Ф. Мельников, С.Ю. Корабельщикова // Информатизация и связь. – 2019. – Т. 1. – С. 55-60.

Some applications of binary lunar arithmetic

V.V. Dang[1], N.L. Dodonova[2], M.V. Dodonov[2], S.Y. Korabelshchikova[3]

[1]State Polytechnic Institute of HochiMinh, 268 Ly Thuong Kiet, dist 10, Hochiminh city, Vietnam
[2]Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086
[3] Northern (Arctic) Federal University named after M.V. Lomonosov, Severnaya Dvina Emb. 17, Arkhangelsk, Russia, 16300

Abstract. In general, the problem of extracting the root of *n*-th order from a language is stated as follows: for a given language *A* and a given $n \in N$ it is required to find all the languages *B*, such that $A = B^n$. This theoretical problem is closely related to the practical task of decoding messages, where it is required to find the fragmentation of the coded message into elementary codes, which correspond with certain symbols of the source alphabet. We previously found a solution to the problem of extracting the *n*-th root for languages of a special form, which is containing all the possible words of the length from $n \cdot n_1$ to $n \cdot n_2$ ($n_1 \leq n_2$). The considered problem was converted to a knapsack problem and solved by software implementation of the algorithms, suggested in the paper. Quantitative estimates of the number of roots of the *n*-th degree were obtained, at different values of *n* and *k*, where *k* is the cardinal number of the set $\{n_1, n_1+1,..., n_2\}$. For n=2 the sequence of the number of square roots coincided with the sequence that was published on the website of the online encyclopedia of integer sequences http://oeis.org/A191701. This sequence represented in base 2 lunar arithmetic, number of binary numbers *x* of length *k* such that $x \cdot x$ has no zeros. In the paper, we have established and theoretically proven a correspondence between operations in binary lunar arithmetic and in the ring of polynomials with integer coefficients. Based on the established correspondence we have developed an algorithm, which enables us to solve both special problem of finding roots from a language and more general knapsack problem, using operations in binary lunar arithmetic. Program implementation of the suggested algorithm has been compared to well-known classical alternate solutions of a knapsack problem.