На правах рукописи

АМЕР ИСМАИЛ Ф. О.

РАЗРАБОТКА ЭФФЕКТИВНЫХ АЛГОРИТМОВ ВЫЧИСЛЕНИЯ НОД НАТУРАЛЬНЫХ ЧИСЕЛ ДЛЯ КРИПТОГРАФИИ И ТЕОРИИ ЧИСЕЛ

Специальность 05.13.11 —
«Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей»

Автореферат
диссертации на соискание учёной степени
кандидата технических наук

Казань — 2020

Работа выполнена в Институте вычислительной математики и информационных технологий Казанского (Приволжского) федерального университета

Научный руководитель: доктор физико-математических наук, профессор **Ишмухаметов Шамиль Талгатович**

Официальные оппоненты: доктор физико-математических наук, старший научный сотрудник, главный научный сотрудник Всероссийского научно-исследовательского Института экспериментальной физики ВНИИЭФ **Дерюгин Юрий Николаевич**

кандидат технических наук, доцент, доцент кафедры систем информационной безопасности Казанского национального исследовательского технического университета им. А.Н. Туполева (КНИТУ-КАИ) **Тумбинская Марина Владимировна**

Ведущая организация: Башкирский государственный университет, г. Уфа, 450076, г. Уфа, ул.Заки Валиди, 32

Защита состоится «20» мая 2020 г. в 14:30 часов на заседании диссертационного совета КФУ.01.04. № 01-03/803 на базе ФГАОУ ВО «Казанский (Приволжский) федеральный университет» по адресу: 420008, Казань, ул. Кремлевская, 35.

С диссертацией можно ознакомиться в библиотеке ФГАОУ ВО «Казанский (Приволжский) федеральный университет» по адресу: 420008, Казань, ул. Кремлевская, 35.

Автореферат разослан «___» _____ 2020 года.

Ученый секретарь диссертационного совета
КФУ.01.04. № 01-03/803,
канд. физ.-мат. наук,
доцент  Еникеев Арслан Ильясович

Общая характеристика работы

Актуальность темы. Диссертационная работа посвящена решению задачи построения эффективного программного обеспечения для реализации алгоритмов вычисления НОД (наибольшего общего делителя) натуральных чисел с точки зрения их приложений к вычислениям с длинными числами, используемых в криптографии и теории чисел. Задача вычисления наибольшего общего делителя (НОД) натуральных чисел представляет собой одну из наиболее распространённые задач, имеющих различные приложения в современной вычислительной математике. Процедура вычисления НОД присутствует во многих криптосистемах и алгоритмах факторизации. Именно поэтому, исследование алгоритмов вычисления НОД является важной темой.

С развитием мощных компьютеров, глобальных компьютерных сетей и информационных технологий становится более важной и задача защиты информации. Криптография работает с длинными числами, выполняя задачи генерации криптографических ключей, операции шифрования и дешифрования данных. Современные криптографические алгоритмы работают с конечными полями большой размерности, выполняя многочисленные модулярные вычисления. Обратные вычисления в конечных полях необходимо используют расширенный алгоритм Евклида и процедуру вычисления НОД в рассматриваемых алгебраических структурах.

Вычисление НОД является важной фундаментальной задачей теории чисел, успешное решение которой позволяет усовершенствовать алгоритмы широкого класса прикладных задач, связанных с использованием современной асимметричной криптографии (алгоритмы RSA, Рабина, Эль-Гамаля, электронной цифровой подписи ЭЦП, исследование порядка эллиптической кривой)[1,2]. как правило, современные криптографические алгоритмы работают с длинными числами и предусматривают процедуру вычисления их НОД[3,4].

Наиболее распространенным из таких алгоритмов является классический алгоритм Евклида. Однако известны и другие алгоритмы, общей целью создания которых было стремление уменьшить сложность вычисления НОД, сократить время его поиска.

---

[1] Ишмухаметов Ш.Т. Технологии защиты информации в сети, методическое пособие, Казань. — 2010.

[2] Рубцова Р.Г., Ишмухаметов Ш.Т. и др. Математические основы защиты информации. — 2012.

[3] Ishmukhametov Sh., Mubarakov B., Mochalov A. Euclidian algorithm for recurrent sequences, Applied Discrete Mathematics and Heuristic Algorithms // International Scientifi c Journal.–Samara . — 2015. —Vol. 1, no. 2. — Pp. 57–62.

[4] Ишмухаметов Ш.Т. Методы факторизации натуральных чисел: учебное пособие. — 2011.

Одним из самых перспективных таких алгоритмов является k-арный алгоритм поиска GCD, опубликованный в 1990-х годах Джонатаном Соренсоном[5,6,7]. По сравнению с классическим алгоритмом Евклида данный алгоритм позволяет заметно снизить время вычислений при аккуратном выборе параметра k. В дальнейшем самим Соренсоном и другими учёными было предложено несколько вариантов улучшения и модификации k-арного алгоритма[8,9,10]. Одним из них является аппроксимирующий k-арный алгоритм, предложенный проф. Ш.Т. Ишмухаметовым[11].

Цель работы. Целью данной работы является сравнительное исследование трех алгоритмов вычисления НОД длинных натуральных чисел: классического алгоритма Евклида, k-арного алгоритма Дж. Соренсона, аппроксимирующего k-арного алгоритма Ш.Т. Ишмухаметова и разработки эффективного программного обеспечения для реализации этих алгоритмов и ускорения операций вычисления наибольшего общего делителя с длинными целыми числами.

Степень разработанности темы. Задача сравнительного изучения трех алгоритмов вычисления НОД и построения эффективного программного обеспечения для их выполнения выполнена в полном объеме. Разработан пакет программ, реализующий базовые версии рассматриваемых алгоритмов, система тестирования, генерирующая пары чисел заданной длины и выполняющая контроль значений различных параметров вычисления таких, как время выполнения всей процедуры, среднее и максимальное число итераций для каждого из рассмотренных алгоритмов и наборов пар чисел заданной длины от 100 до 3000 бит, влияние различных методов ускорения на сходимость процедуры вычисления НОД.

С помощью этих программ были получены численные оценки сходимости трех рассматриваемых алгоритмов для различных входных наборов данных, выведены точные оценки времени выполнения процедуры вычисления НОД и числа итераций, доказана эффективность использования k-арного алгоритма Соренсона КАРИ и аппроксимирующего алгоритма

[5]Sorenson J. The k-ary GCD algorithm. — University of Wisconsin-Madison, Computer Sciences Department, 1990. — Pp. 1–20.

[6]Sorenson J. Two fast GCD algorithms // Journal of Algorithms. — 1994. — Vol. 16, no. 1. — Pp. 110–144.

[7]Sorenson J.P. An analysis of the generalized binary GCD algorithm // High Primes and Misdemeanors: Lectures in Honour of the 60th Birthday of Hugh Cowie Williams. — 2004. — Pp. 327–340.

[8]Weber K. The accelerated integer GCD algorithm // ACM Transactions on Mathematical Software (TOMS). — 1995. — Vol. 21, no. 1. — Pp. 111–122.

[9]Jebelean T. A generalization of the binary GCD algorithm // Proceedings of the 1993 international symposium on Symbolic and algebraic computation / ACM. — 1993. — Pp. 111–116.

[10]Wang X., Pan V.Y. Acceleration of Euclidean algorithm and rational number reconstruction // SIAM Journal on Computing. — 2003. — Vol. 32, no. 2. — Pp. 548–556.

[11]Ishmukhametov Sh. An approximating k-ary GCD algorithm // Lobachevskii Journal of Mathematics. — 2016. — Vol. 37, no. 6. — Pp. 723–729.

АКА для решения практических задач теории чисел и криптографии. Также определен новый комбинированный алгоритм КОМБИ, соединяющий простоту классического алгоритма Евклида и достоинства k-арного алгоритма и выполняющийся быстрее обоих алгоритмов.

Для достижения поставленной цели были решены следующие **задачи**:

1. Выполнен анализ современных алгоритмов вычисления НОД натуральных чисел — классического алгоритма Евклида КАЕ, бинарного алгоритма Стейна, k-арного алгоритма и аппроксимирующего k-арного алгоритма;
2. Разработан программный комплекс для выполнения полноценного тестирования всех трех рассматриваемых в диссертации алгоритмов, оценки влияния входных и промежуточных параметров на скорость выполнения процедуры, число итераций и время, затрачиваемое на каждой итерации и всей процедуры в целом.
3. Разработаны эффективные реализации алгоритмов вычисления НОД с использованием библиотеки длинных чисел MPIR[12] в среде Visual Studio, на основе современных принципов программирования;
4. Выполнены экспериментальные вычисления и сбор статистического материала по скорости выполнения этих алгоритмов и числу итераций в зависимости от выбора параметров;
5. Выполнено ускорение алгоритмов вычисления НОД на основе подбора эффективных параметров и использования предтаблиц;
6. Разработаны практические приложения алгоритмов для решения сложных вычислительных задач (поиска строго псевдопростых чисел для ускорения теста простоты Миллера-Рабина.

Научная новизна: Аппроксимирующий k-арный алгоритм является современным эффективным алгоритмом вычисления НОД, разработанный Ш.Т. Ишмухаметовым. До настоящего времени не было ни одной полной реализации этого алгоритма для работы с длинными числами. Также никем не выполнялось сравнительное тестирование этого алгоритма в сравнении с другими известными алгоритмами вычисления НОД. Эта работа была выполнена впервые в нашей диссертации. Нами доказано абсолютное превосходство данного алгоритма над классическим алгоритмом Евклида и k-арным алгоритмом Соренсона путем выполнения экспериментальных вычислений с числами различной длины.

Также нами были разработаны методы, ускоряющие выполнение этого алгоритма с использованием особенностей представления длинных чисел в библиотеке MPIR и использования рядов Фарея для аппроксимации параметра α, играющего основную роль в этом алгоритме.

[12] Torbjorn Granlund and the GMP Development Team. The Multiple Precision Integers and Rationals Library Edition 2. — 2015. — URL: http://mpir.org/mpir-2.7.2.pdf

Основные достижения диссертации содержатся в следующем перечне:

1. Разработан программный комплекс, реализующий рассмотренные алгоритмы и выполняющий оценки их эффективности на языке программирования язык C++ в среде Visual Studio с использованием библиотеки длинных чисел MPIR);
2. Найдены оптимальные значения параметров, обеспечивающих быстрое выполнение k-арного и аппроксимирующего алгоритмов для пар чисел различной длины;
3. Доказано практическое ускорение процедуры вычисления НОД с использованием k-арного и аппроксимирующего k-арного алгоритмов по отношению к классическому алгоритму Евклида и бинарному алгоритму;
4. Выполнено исследование скорости сходимости k-арного алгоритма Соренсона при сдвиге области значений параметров x, y, найдены оптимальные параметры и доказано практическое ускорение k-арного алгоритма до 20
5. Выполнены экспериментальные оценки скорости вычисления НОД для чисел различной разрядности для аппроксимирующего k-арного алгоритма, получены численные результаты сходимости этих алгоритмов в виде таблиц и графиков по числу итераций и времени вычислений для чисел различной длины до 2000 десятичных разрядов (примерно, 6500 бит).
6. Разработан новый комбинированный алгоритм КОМБИ вычисления НОД, основанный на сочетании на разных итерациях алгоритмов Евклида и k-арного алгоритма, работающий до двух раз быстрее, чем исходные алгоритмы Евклида и Соренсона.

Теоретическая и практическая значимость работы. В диссертации были исследованы и построены новые эффективные версии алгоритмов вычисления НОД на основе алгоритмов Евклида, Соренсона и Ишмухаметова. Эти алгоритмы могут быть использованы в криптографии, теории чисел и других численных разделах математики при генерации параметров криптосистем с открытым ключом (RSA), при поиске нетривиальных делителей натуральных чисел (факторизации чисел), при построении электронной цифровой подписи ЭЦП, построении архивов типа блокчейн и решении других задач с числами большой размерности.

Методология и методы исследования. При выполнение работы использовались методы теории чисел, арифметические операции в конечных полях, теории алгоритмов, библиотека длинных чисел MPIR в Visual studio и компьютерного моделирования.

Основные положения, выносимые на защиту:

1. Разработка эффективного программного обеспечения для задачи вычисления наибольшего общего делителя пар длинных натуральных чисел;
2. Разработка компьютерной системы тестирования для различных алгоритмов вычисления наибольшего общего делителя;
3. Разработка эффективных реализаций k-арного алгоритма Соренсона с ускорением базового алгоритма до двух раз;
4. Разработка эффективных реализаций аппроксимирующего алгоритма Ишмухаметова с ускорением базового алгоритма до пяти раз;
5. Разработка программного комплекса для поиска строго псевдопростых чисел с использование аппроксимирующего алгоритма;

Достоверность. Степень достоверности полученных в работе результатов обеспечивается строгостью постановки задач и математических методов их решения, а также экспериментальный системный подход к построению программного комплекса, экспериментально подтверждающего теоретические оценки.

Соответствие паспорту специальности. Работа выполнена в рамках направления области исследований «Модели, методы и алгоритмы проектирования и анализа программ и программных систем, их эквивалентных преобразований, верификации и тестирования» паспорта специальности 05.13.11 — Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей.

Апробация результатов работы. Основные результаты работы докладывались на конференциях:

1. Международная конференция по алгебре, анализу и геометрии КФУ, Казань, июнь-июль 2016;
2. Итоговая научная конференция Казанского федерального университета (КФУ) за 2016 год, Казань, январь 2017;
3. Международная научная конференция, Scientific research of the SCO countries: Synergy and Integration — Пекин, Китай, июнь 2018;
4. XXVII Международная научно-практическая конференция «Вопросы современных научных исследований», Научный центр «Орка», Омск, июль 2018;
5. VII Международная конференция “Последние тенденции в области науки и технологий управления” — Лондон, Великобритания, июль 2018;
6. XXIX Международная научно-практическая конференция «Вопросы современных научных исследований», Научный центр «Орка», Омск, август 2018;
7. Международная конференция, International Conference on Innovations in Engineering, Technology and Sciences (ICIETS-2018), Karnataka, India, сентябрь 2018;

8. Международная научно-практическая конференция “Математическое моделирование, программирование и прикладная математика”, Великий Новгород, июнь 2019;
9. Международная математическая конференция стран БРИКС, The 3rd BRICS Mathematics Conference, Иннополис, июль 2019;
10. XXII Международный научно-исследовательский конкурс «Лучшая научно-исследовательская работа 2019», МЦНС «Наука и Просвещение», Пенза, октябрь 2019;
11. II Международная конференция «Передовые технологии в аэрокосмической отрасли, машиностроении и автоматизации - MIST-2019», Красноярск, ноябрь 2019;

Личный вклад. Автор принимал активное участие в теоретических разработках и анализе предложенных методик, разработал и реализовал программный комплекс, необходимый для получения статистических численных оценок, выполнил цикл экспериментальных вычислений процедуры вычисления НОД с использованием трех алгоритмов, выполнил анализ полученных данных, разработал методику ускорения вычисления НОД для k-арного алгоритма Соренсона на основе сдвигов области выбора параметров, доказал практическую эффективность применения k-арного алгоритма Соренсона и аппроксимирующего алгоритма в сложных вычислительных задачах теории чисел и криптографии, использующих процедуру вычисления НОД.

Публикации. Основные результаты по теме диссертации изложены в 11 печатных изданиях, 5 из которых изданы в журналах, рекомендованных ВАК, 6 — в прочих изданиях.

✓ Статьи [1–5] в журналах, рекомендованных ВАК, а статьи [6–11] — в прочих изданиях.

Структура и объем диссертации. Диссертация состоит из введения, трёх глав с выводами, заключения, и практических рекомендаций. Она изложена на **124** страницах машинописного текста, включает **14** рисунков, **29** таблиц, приложение и содержит список литературы из **131** наименований, среди которых **38** отечественных и **93** иностранных авторов.

Содержание диссертации

Во **введении** обосновывается актуальность исследований, проводимых в рамках данной диссертационной работы, приводится обзор научной литературы по изучаемой проблеме, формулируется цель, ставятся задачи работы, сформулированы научная новизна, практическая значимость представляемой работы, и раскрыты основные положения, выносимые на защиту.

Первая глава посвящена постановке задачи и обзору основных алгоритмов вычисления наибольшего общего делителя (НОД) натуральных чисел. Целью данной главы является анализ и сравнение эффективности существующих методов поиска НОД. Самым древним из которых является классический алгоритм Евклида (КАЕ). Этот алгоритм имеет многочисленное применение в теории чисел и криптографии и остается одним из самых эффективных алгоритмов вычисления НОД. Его расширенная версия (РАЕ) используется во многих криптографических и теоретико-числовых алгоритмах, поэтому оценка его производительности играет важную роль в расчетах эффективности криптографических алгоритмов.

Алгоритм Евклида имеет многочисленное применение в теории чисел и криптографии. Приведем некоторые примеры.
В теории чисел:

- Его расширенная версия используется для нахождения обратных элементов в конечных полях.
- Алгоритм используется при решении линейных диофантовых уравнений. Частный случай линейного диофантового уравнения – это соотношение Безу[13].
  НОД$(A,B) = xA + yB$,
  которое легко решается при помощи расширенной версии алгоритма.
- При помощи алгоритма можно представить любое вещественное число в виде непрерывной(цепной) дроби, если это число рационально.

$$\frac{A}{B} = q_0 + \frac{1}{q_1 + \frac{1}{q_2 + \frac{1}{q_n}}} = [q_0; q_1; q_2; \ldots; q_n]$$

  где q_i - значение $A\ div\ B$ на i-ой итерации алгоритма Евклида, при $i = 0 \ldots n$.
- Также при помощи алгоритма можно определить являются ли два числа взаимно простыми, то есть не имеют никаких общих делителей, кроме 1.

В криптографии:

- Алгоритм используется для генерации открытого и секретного ключей в методе шифрования RSA.
- Используется для генерации параметров точек на эллиптических кривых и вычисления их кратного.

[13] Ишмухаметов Ш.Т., Мубараков Б.Г., Аль-Анни К.М. Вычисление коэффициентов Безу для k-арного алгоритма нахождения НОД // Известия высших учебных заведений. Математика. — 2017. — № 11. — С. 30–38.

– Одним из наиболее перспективных направлений, в которых быстрые алгоритмы НОД играют важную роль, является поиск сильных псевдопростых целых чисел, что улучшает эффективность тестов простоты для криптографии[14,15].

В первой главе еще рассматривается бинарный алгоритм вычисления НОД, который вычисляет НОД двух натуральных чисел, используя более простые арифметические операции, чем КАЕ. Он заменяет деление на арифметические сдвиги, сравнения и вычитание. Еще рассмотрены k-арный алгоритм Евклида, который был разработан Д. Соренсоном в 1990 году и аппроксимирующий k-арный алгоритм (новая версия k-арного алгоритма вычисления НОД натуральных чисел, которая была разработана Ш. Ишмухаметовым в 2016 году).

Вторая глава посвящена исследованию эффективного программирования k-арного алгоритма вычисления НОД двух или более целых чисел и предложим ряд модификаций, ускоряющих выполнение алгоритма.

k-арный алгоритм Евклида был разработан в 1990 году Дж. Соренсоном[16,17] и улучшен независимо друг от друга Вебером[18] и Джебелеаном[19] в 1993 и 1995 годах. Этот алгоритм является естественным обобщением бинарного алгоритма. Основная идея алгоритма состоит в том, чтобы искать на шаге вычисления искать линейную комбинацию $Ax+By$ заданных чисел A и B такую, что выполняется соотношение

$$Ax + By \equiv 0 \bmod k, \tag{1}$$

где k – заранее выбранный и фиксированный параметр алгоритма.

В простейшем случае значение k равно 2, и тогда алгоритм совпадает с бинарным. Значение k может быть выбрано любым положительным целым числом. Первые исследователи этого метода (Дж. Соренсон, Т. Джебелеан, К. Вебер) рассматривали разные значения этого параметра (например, Соренсон предложил выбрать k равным степени большого простого числа), однако, никаких преимуществ выбор k простым, составным или степенью простого числа не давал, поэтому наилучшим вариантом стали считать значение k равным степени двойки (Вебер). Это позволило

[14] Ishmukhametov Sh., Mubarakov B. On practical aspects of the Miller-Rabin Primality Test // obachevskii Journal of Mathematics. — 2013. — Vol. 34, no. 4. — Pp. 304–312.

[15] Sorenson J., Webster J. Strong pseudoprimes to twelve prime bases // Mathematics of Computation. — 2015. — Vol. 86, no. 304. — Pp. 985–1003.

[16] Sorenson J. The k-ary GCD algorithm. — University of Wisconsin-Madison, Computer Sciences Department, 1990. — Pp. 1–20.

[17] Sorenson J. Two fast GCD algorithms // Journal of Algorithms. — 1994. — Vol. 16, no. 1. — Pp. 110–144.

[18] Weber K. The accelerated integer GCD algorithm // ACM Transactions on Mathematical Software (TOMS). — 1995. — Vol. 21, no. 1. — Pp. 111–122.

[19] Jebelean T. A generalization of the binary GCD algorithm // Proceedings of the 1993 international symposium on Symbolic and algebraic computation / ACM. — 1993. — Pp. 111–116.

ускорить выполнение рутинных операциях на шагах алгоритма, поскольку деление на степень двойки означало просто сдвиг числа вправо.

Итак, пусть k - четная степень двойки, например, $k = 2^m$, и $A \geq B > 0$ - нечетные целые числа. В главе 1 была сформулирована и доказана основная теорема Соренсона, служащая базисом k-арного алгоритма:

Теорема 1. ***(Дж. Соренсон [1990]).*** *Пусть $m > 1$ – фиксированное натуральное число, $k = m^2$. Для любых целых чисел A и B, $A \geq B > 0$, взаимно простых с m, существуют ненулевые $x, y, |x|, |y| \leq m$ такие, что выполняется соотношение (1).*

Следуя теореме Соренсона, на шаге алгоритма для фиксированного k и заданных нечетных A и B вычисляются коэффициенты x, y, удовлетворяющие теореме, и вычисляется $C = (Ax + By)/k$. Далее вычисляется $d =$НОД(C, m), и если он не равен 1, то выполняется сокращение C на d, пока C не станет взаимно простым с m. В завершении итерации пара (A,B) заменяется парой (B,C), которая имеет меньший размер по сравнению с исходной парой.

Назовем отношение большего числа к меньшему в паре $(A; B)$ на шаге алгоритма коэффициентом редукции и обозначим символом $\rho = A/B$. Чем больше среднее значение этого коэффициента, тем быстрее сходится алгоритм.

Согласно теореме Соренсона:

$$\frac{A}{C} = \frac{Ak}{Ax + By} \geq \frac{Ak}{2A\sqrt{k}} = \frac{\sqrt{k}}{2}.$$

откуда

$$\rho = \sqrt{A/C} \geq \frac{k^{1/4}}{\sqrt{2}} \approx 0{,}707k^{1/4}.$$

Таким образом, что при увеличении k в 16 раза, нижняя граница для ρ увеличивается в два раза.

Значение k можно брать большим, оставаясь в размере одного машинного слова (до 2^{64}), однако экспериментальные вычисления, которые мы приведем в этой главе, показывают, что скорость алгоритма оптимальна при значениях k от $k = 1024$ до $k = 4096$. При дальнейшем увеличении k поиск подходящей пары (x,y) и выполнение промежуточных вычислений выполняется дольше, что замедляет выполнение алгоритма в целом.

Кроме того, поскольку НОД d, вычисленный в k-арном алгоритме не обязательно равен исходному НОД$(A, B) = d_0$, а является его кратным. Поэтому после завершения процедуры вычисления НОД с помощью k-арного алгоритма и получения его ответа d, необходимо выполнить дополнительное вычисление, используя алгоритм Евклида

$$d_0 = GCD(A; GCD(B,d)),$$

получая правильное значение НОД.

Дополнительные множители, входящие в ответ k-арного алгоритма, называются посторонними множителями. Их размер также растет с увеличение k. Это также замедляет общую процедуру k-арного алгоритма.

Во второй главе подробно рассмотрен базовый k-арный метод вычисления НОД натуральных чисел и методы его ускорения. Был разработан программный комплекс по которому были проведены разные экспериментальные вычисления и сравнительный анализ результатов итерации и времени выполнения. Мы исследовали значения коэффициента редукции в зависимости от k и поиск подходящей пары (x, y) в k-арном алгоритме. Представлены методы ускорения процедуры поиска пары (x, y) и проведены экспериментальные данные по времени вычисления НОД при использовании предтаблиц обратных элементов и значений x и y. Были получены экспериментальные результаты времени при сдвиге области значений y. Еще был рассмотрен смешанный алгоритм на основе k-арного алгоритма и схемы Евклида.

В этой главе нами были получены следующие результаты:

1. Среднее количество итераций k-арного алгоритма для чисел длины 500 десятичных разрядов уменьшается от 1358 до 573 при увеличении k от 16 до 2^{16}. Это число значительно меньше среднего числа итераций для таких пар по схеме Евклида, которое составляет 1942 итерации.

2. Среднее значение коэффициента редукции $\rho = A/B$ в процессе вычисления превышает примерно в 5 раз наименьшее значение этого коэффициента, обеспечиваемое теоремой Соренсона для всех k от 16 до 2^{16}.

3. Базовый k-арный алгоритм по схеме Соренсона выполняется медленнее классического алгоритма Евклида для пар чисел длины 1600 бит почти для всех значений k. Наилучшим значением является $k = 16384$, при котором этот алгоритм работает немного быстрее алгоритма Евклида (101 мс на 50 парах против 112 мс у алгоритма Евклида).

4. При использовании предтаблиц обратных элементов по модулю k алгоритм ускоряется от 3,8% до 9,9% в зависимости от значения k. Наилучший результат (91 мс и 9,9% выигрыша) на парах длины 1600 бит достигается при $k = 16384$. Немного проигрывает время вычисления при $k = 4096$ - 94 мс. При увеличении k время опять начинает увеличиваться.

5. Те же расчеты, выполненные для чисел 3200 бит, показывают ухудшение времени работы k-арного алгоритма относительно схемы Евклида. Наилучшим средним временем на числах такой длины является время 288 мс, достигнутое при $k = 16384$, но даже это время проигрывает времени работы алгоритма Евклида (264 мс). Отсюда следует, что базовая версия алгоритма Соренсона даже с использованием предтаблиц работает хуже

алгоритма Евклида при длине входных аргументов больше примерно 3000 бит.

6. Однако для чисел длины меньше 3000 бит (напомним, что стандартные длины ключей RSA составляют 1024 и 2048 бит) алгоритм Соренсона обгоняет алгоритм Евклида. Время выполнения этих алгоритмов на числах 640 бит доказывает преимущество k-арного алгоритма по отношению к алгоритму Евклида почти для всех k, а при $k = 4096$ выполняется за 25 мс против 34 мс у алгоритма Евклида (выигрыш составляет 26,5%).

7. Исследован метод непосредственного анализа значений параметра $q = AB^{-1} \bmod k$ для ускорения вычисления коэффициентов (x,y). Однако, данный алгоритм не дал значительного ускорения вычисления и в дальнейшем не рассматривался.

8. Также был исследован метод использования предтаблиц для прямого нахождения коэффициентов (x,y) по значению параметра $q = AB^{-1} \bmod k$. Этот подход требует значительного объема машинной памяти, так как необходимо хранить таблицы, содержащие примерно $k/4$ строк, что при больших k становится не эффективным. Однако при $k = 4096$ мы смогли достичь более быстрого вычисления НОД 50 пар чисел длины 1600 бит, равной 85 мс.

9. Далее нами был исследован метод сдвига параметра y, заключающийся в том, что вместо интервала $[-\sqrt{k}; \sqrt{k}]$ подходящее значение y мы искали в сдвинутом влево интервале $[-\sqrt{k}-t; \sqrt{k}-t]$. При подходящем значении t параметры x и y имеют противоположные знаки, что обеспечивает уменьшение значения линейной комбинации $|Ax+By|$. Этот метод хорошо тем, что не требует никаких значительных изменений в алгоритме, а просто изменения границ в процедуре поиска значений коэффициентов (x,y).

Сначала мы рассмотрели небольшие значения k и сдвиги $t < 2\sqrt{k}$ на числах длины 1600 бит. При таких ограничениях получено небольшое ускорение (до 12,8%) при $k = 16$ и $t = 5$ (то есть y искался на интервале $[-9; -1]$ вместо $[-4; 4]$).

При следующем значении $k = 64$ удалось достичь только 3-х процентного выигрыша при $t = 3$. Поэтому при $k \geq 256$ сдвиг на небольшие значения t является неэффективным.

10. Следующими был рассмотрен метод сдвига на большой интервал, сравнимый по величине с параметром k. В этом случае рассматривались различные значения k и пары чисел различной длины.

Общий выигрыш, полученный от таких сдвигов, зависит как от величины сдвига, так и от длины исходных чисел. Наилучший результат для пар длины 4800 бит был достигнут при сдвиге на величину половины k и составит примерно 25 процентов.

11. Последним был предложен комбинированный метод вычисления НОД, основанный на использовании на чередовании k-арного алгоритма и алгоритма Евклида. Теоретическая выгода такого сочетания обоснована в секции 2.11, а экспериментальные подсчеты, приведенные в таблице, показывают выигрыш такого алгоритма по отношению к схеме Соренсона с предвычислениями обратных элементов по k от 10 до 18 процентов. По этой схеме было получено наилучшее значение для среднего времени вычисления НОД 50 пар чисел длины 1600 бит, равное 81 мс (28 процентов выигрыша у алгоритма Евклида вычисляющего НОД за 112 мс).

Третья глава посвящена исследованию эффективной реализации на компьютере аппроксимирующего k-арный алгоритм, являющегося естественным обощением k-арного алгоритма Соренсона. Этот алгоритм был разработан в 2016 году Ш.Т. Ишмухаметовым[20]. Главное отличие аппроксимирующего k-арного алгоритма от алгоритма Соренсона заключается в выборе на шаге алгоритма пары параметров (x,y), обеспечивающих тождество

$$Ax + By \equiv 0 \ (\mathrm{mod} k). \tag{2}$$

Идея Ишмухаметова состояла в том, чтобы выбирать эти значения не перебором, а таким способом, чтобы линейная комбинация $|Ax + By|$ принимала минимальное значение. Напомним, что тождество (2) эквивалентно равенству

$$y = -qx + ks, \text{ где } \ q = AB^{-1} \bmod \ k, \ s \in \mathbb{N}.$$

Варьируя значения двух параметров x и s, можно добиться высокой скорости сходимости k-арного алгоритма, во много раз превосходящего базовую версию Соренсона.

В третьей главе мы подробно исследовали аппроксимирующий k-арный алгоритм вычисления НОД и его эффективную реализацию на языке программирования C++ в среде программирования Visual Studio с установленной библиотекой длинных чисел MPIR. Приведены теоретические расчеты. Проведены оценки производительности вычисления НОД для пар чисел различной длины по числу итераций и времени работы. Исследованы методы ускорения аппроксимирующего алгоритма и проведены разные оценки времени вычисления. Были реализованы приложения этого алгоритма такие, как тест простоты Миллера-Рабина и поиск строго псевдопростых чисел по которым были проведены экспериментальные результаты.

Библиотека длинных чисел MPIR является портированием под операционную систему Windows известной библиотеки длинных чисел GMP, разработанной первоначально для ОС Linux. Эта библиотека содержит тип

[20] Ishmukhametov Sh. An approximating k-ary GCD algorithm // Lobachevskii Journal of Mathematics. — 2016. — Vol. 37, no. 6. — Pp. 723–729.

длинных положительных чисел mpz_t и набор основных операций с этим форматом чисел. Описание переменной типа mpz_t выполняется так же, как и обычных типов, то есть командой

```
mpz_t  список переменных, разделенных запятой;
```

после чего переменные, перечисленные в этом списке должны быть инициализированы командой mpz_init. Переменная типа mpz_t является указателем на блок памяти, выделенной переменной этого типа. После использования переменной память, занимаемая переменной, может быть освобождена командой mpz_clear.

Приведем пример программирования алгоритма Евклида вычисления НОД для пары длинных чисел A и B, $A > B$:

```
mpz_t Euclid (mpz_t A,  mpz_t B){
     mpz_t  C;   mpz_init(C);
     size_t  L=mpz_size(B);
     while (L>0){
            mpz_fdiv_r(C, A, B);
            mpz_set(A, B); mpz_set(B, C); L=mpz_size(B);
     }
     return A;
     mpz_clear(C);
}
```

В этой процедуре команда mpz_set присваивает значение второго операнда первому, а команда mpz_fdiv_r вычисляет остаток от деления второго операнда на третий и заносит полученное значение в первую переменную.

Длина переменной типа mpz_t определяется функцией mpz_size и вычисляется в лимбах, где лимб – это длина регистра процессора компьютера, в котором установлена библиотека, т.е. лимб равен 32 или 64 битам. Если число не превышает 2^{64} для 64-битового процессора, то длина такого числа равна 1. Условие mpz_size(B)> 0 эквивалентно условию $B > 0$, но выполняется быстрее.

Перейдем теперь к программированию аппроксимирующего алгоритма.

```
mpz_t AKA(mpz_t A,  mpz_t B, int k){
    mpz_t AA, BB;
    int i=0, a, b, q, m, n,x,y,y-;
    double r,alpha, beta;
    mpz_init_set(AA, A); mpz_init_set(BB, B);
    // copy A, B to AA, BB
    size_t  L1,L2;
```

```
    while(L2>0){
        i++;
        L1==mpz_size(A); L2=mpz_size(B);
        double r=mpz_fdivq_ui(A,B);
        // Take rests of least limbs of A, B ny module k
        a=(A->_mp_d[0])%k; b = (B->_mp_d[0]) % k;
        q = (a*InvK(b,k) % k; // Compute q=AB^{-1} (mod k)
        beta=(r-q)/k;
        int s0=int(beta);  alpha=beta-s0;
        // beta=alpha+s0 sum of fraction and int parts of beta
        Farey(k, alpha, m,n);
    // proc Farey computes approximating fraction m/n for alpha
        x=n; y-=q*n+(m+s0*n)*k;// y-=-y
        mpz_mul_ui(T1,A,x);// T1=Ax
        mpz_mul_ui(T2,B,y-);// T2=-By
        int d=mpz_cmp(T1,T2);// Compare T1 and T2
        if(d>0)mpz_sub(C,T1,T2);
        if(d<0)mpz_sub(C,T2,T1);  // C=Ax+By
        if(d==0) goto fin1;// Case Ax+By=0.
        mpz_fdiv_q_ui(C,C,k);// C=(Ax+By)/k
        while((C->_mp_d[0])%2==0) mpz_fdiv_q_ui(C,C,2);
        // Divide C by 2 while C is even
        mpz_set(A,B); mpz_set(B,C);
        \\ go to the next iteration
   }
   return A;
}
```

Основной цикл представляет собой одну итерацию, в которой по паре нечетных чисел A,B, $A > B$, вычисляются целые числа x, y, такие что

$$0 < x \le k, y < 0, \;\; Ax + By \equiv 0 (\mathrm{mod}\ k), \;\; Ax \approx -By. \tag{3}$$

После этого вычисляется число $C = |(Ax + By)/k|$, C проверяется на четность и если четно, то сокращается на 2 до тех пор, пока не станет нечетным. После этого выполняется присвоение значений $A = B$, $B = C$, и выполняется переход к следующей итерации.

Нами была приведена грубая схемы вычисления НОД с помощью аппроксимирующего алгоритма.

Приведем пример вычислений по этому алгоритму для 33-битовых чисел A и B и $k = 64$:

Таблица 1 — Вычисления НОД с помощью аппроксимирующего алгоритма для 33-битовых чисел A и B и $k = 64$.

	A	B	A/B	q	α	m	$x = n$	y	C
1	1736704041	1210259647	1,43	23	0,587	41	62	-82	2430861
2	1210259647	2430861	498	59	0,857	6	7	-3485	4174
3	2430861	4174	583	7	0,997	62	63	-36665	419
4	4174	419	10,0	57	0,265	9	34	-338	3

Ответ: GCD= 3.

Из примера видно, что даже при небольших значениях k аппроксимирующий алгоритм обеспечивает быструю сходимость вычисления, значительно обгоняя предыдущие алгоритмы по значению коэффициента редукции $\rho = A/B$.

В этой главе нами были получены следующие результаты:

1. Была разработана процедура генерации пар чисел, взаимно простых с параметров k, имеющих различную длину от 100 до 1000 десятичных разрядов.

2. Реализована базовая процедура вычисления НОД с использованием алгоритма Евклида с выводом в ответ общего и среднего числа итераций и суммарного времени по серии из N испытаний для произвольного N. Подобрано значение $N = 50$, достаточное для того, чтобы сгладить флуктуации средних значений параметров относительно средних значений и получать реальные оценки сходимости алгоритма в зависимости от длин исходных пар.

3. Разработана базовая модель реализации аппроксимирующего алгоритма на языке C++ в системе MS Visual Studio 2012 с использованием библиотеки длинных чисел MPIR. Выполнены реализации на компьютере основных процедур, входящих в базовую схему аппроксимирующего алгоритма. Такими процедурами являются:

3.1. Процедура вычисления параметров $r = A/B$ и $q = AB^{-1} \bmod k$ путем прямых вычислений и с использованием предвычисленных таблиц.

3.2. Процедура ускоренного вычисления параметра $r = A/B$ с использованием особенностей представления длинных чисел в библиотеке MPIR с обращением к отдельным частям (лимбам) длинных чисел.

3.3. Процедура аппроксимации действительного параметра α дробью с ограниченным значением знаменателя.

4. Разработана и реализована эффективная модель аппроксимации действительного параметра α правильной дробью с ограниченным знаменателем с использованием рядов Фарея.

5. Выполнены серии вычислений НОД с использованием различных значений длин входных чисел, параметра k и различных ускоряющих приемов для оптимальной сходимости всего алгоритма.

6. Подсчитаны среднее и общее число итераций, наилучшее время для каждого из входных наборов исходных данных, значения параметра k и реализаций тех или иных процедур аппроксимирующего алгоритма. Результаты вычислений собраны в таблицы.

7. Доказано абсолютное превосходство аппроксимирующего алгоритма при выборе оптимальных параметров вычисления при различных длинах исходных чисел от 100 до 1000 десятичных разрядов у алгоритмов Евклида и k--арного алгоритма Соренсона как по числу итераций, так и по времени вычислений. По числу итераций аппроксимирующий алгоритм обгоняет алгоритм Евклида до 5 раз и комбинированный k-арный алгоритм до двух раз, а по времени - до трех раз.

8. Был реализован эффективный поиск строго псевдопростых чисел на основе аппроксимирующего алгоритма, с помощью которого были найдены все строго псевдопростые числа, меньшие 10^{12}, включая числа ψ_5 и ψ_6 из последовательности $\{\psi_n\}_n$, в настоящее время построенной до члена ψ_{13} (J. Sorenson, J. Webster)[21]. Этим доказана высокая эффективность аппроксимирующего алгоритма, выполняющего подобные вычисления примерно в три раза быстрее классического алгоритма Евклида, который обычно используется в подобных вычислениях.

В **заключении** приведены основные выводы по диссертации, сформулированы полученные научные и практические результаты, которые заключаются в следующем:

1. Разработана эффективная реализация k-арного алгоритма Дж. Соренсона.

Для реализации этой цели была построена система тестирования, содержащая большой набор тестовых пар чисел различной длины от 100 до 3000 бит, программное обеспечение на языке C++ на основе программной среды MS Visual Studio 2012 и библиотеки длинных чисел MPIR для выполнения операций вычисления наибольшего общего делителя и системой контроля влияния отдельных параметров на ход процедуры, системы оценки качества разрабатываемых алгоритмов, оценивающая общее и среднее число итераций, требуемых для вычисления НОД для наборов пар натуральных чисел, обще е и среднее время выполнения в сравнении для классического алгоритма Евклида.

[21] Sorenson J., Webster J. Strong pseudoprimes to twelve prime bases // Math ematics of Computation. — 2015. — Vol. 86, no. 304. — Pp. 985–1003.

2. Были предложены и реализованы многочисленные методы модернизации k-арного алгоритма, выполнены оценки их влияния на скорость выполнения алгоритма и число итераций. Среди предложенных модернизаций перечислим основные:

2.1. Была реализована система предвычислений, содержащая таблицы обратных по модулю k чисел, используемая для поиска пар чисел x,y. Это позволило добиться ускорения базового алгоритма до 25 процентов.

2.2. Были выполнены оценки средних значений коэффициента редукции $\rho = A/B$ на отдельных итерациях алгоритма. Полученные результаты показали, что его значение зависит только от значения параметра k и составляет величину приблизительно равную $2{,}5\sqrt{k}$, что примерно в 5 раз превышает минимальное теоретическое значение, построенное Дж.Соренсоном.

2.3.Было доказано увеличение эффективности k-арного алгоритма при увеличении от 16 до $2^{14} = 16384$. При дальнейшем увеличение этого параметра общая эффективность алгоритма начинала ухудшаться. Поэтому оптимальным значением для k-арного алгоритма является $k = 16384$.

2.4. Была разработана и реализована система предвычислений параметров x,y, позволяющая выполнить поиск искомых значений параметров для каждого заданного значения k и пары значений $a = A \bmod k$ и $b = B \bmod k$. Это позволило получить ускорение до 10 процентов скорости выполнения k-арного алгоритма для $k \leq 1024$). Однако, при увеличении k эффективность алгоритма падала, объем таких таблиц стал слишком большой (для $k = 16384$ составляем более гегабайта объема), поэтому в целом эта модернизация была признана не эффективной и в дальнейшем не использовалась.

2.5. Была разработан и реализован алгоритм сдвига области изменения параметров (x,y), который позволил получить до 25 процентов ускорения без какого-то существенного изменения выполняемых кодов.

2.6. Были проанализировано изменение коэффициента редукции $\rho = A/B$ на отдельных итерациях алгоритма и выяснено, что происходит чередование итераций с небольшим значением ρ и итерациями, на которых ρ получает большие значения. Для уменьшения таких провалов был предложен новый модифицированный комбинированный алгоритм, чередующий использование на нечетных шагах k-арного алгоритма с алгоритмом Евклида, выполняемым на четных шагах. Такой алгоритм оказался самым эффективным из всех предложенных модификаций. Эта модификация работает примерно на 30 процентов быстрее алгоритма Евклида.

Полученные результаты могут быть использованы для дальнейших теоретических исследований в области эффективных алгоритмов вычисления наибольшего общего делителя НОД с приложениями в криптографии и теории чисел такие, как RSA, El Gamal, Elliptic Curves Ciphering, алгоритмы электронной цифровой подписи ЭЦП и другие высокопроизводительные вычисления на эллиптических кривых над конечными полями. Они еще играют важную роль в перспективных направлениях для поиска сильных псевдопростых целых чисел, что улучшает тест простоты для криптографии.

Публикации автора по теме диссертации

1. *Amer, I.* Lenstra Factorization Method Convergence Investigation on Elliptic Curves/ I. Amer, Sh.T. Ishmukhametov, R.G. Rubtsova // *Research Journal of Applied Sciences.* — 2015. — Vol. 10, no. 8. — Pp. 365–370.

2. *Amer, I.* Analysis of Schönhage-Strassen multiplication algorithm in finite Galois fields/ M.P. Vasilev, S.T. Ishmukhametov, I. Amer // *Astra Salvensis.* — 2017. — Vol. 4, no. 13. — Pp. 591–602.

3. *Амер, И.* Об ускорении k-арного алгоритма вычисления НОД натуральных чисел/ И. Амер, Ш.Т. Ишмухаметов // *Ученые записки Казанского университета. Серия Физико-математические науки.* — 2019. — Т. 161, № 1. — С. 110–118.

4. *Amer, I.* Analysis of the k-ary Euclid for tuples of integers/ I. Amer, Sh.T. Ishmukhametov // *Journal of Physics: Conference Series.* — 2019. — Vol. 1352. — P. 012001.

5. *Amer, I.* On acceleration of the k-ary GCD algorithm/ I. Amer // *IOP Conference Series: Materials Science and Engineering.* — 2020. — Vol. 734. — P. 012149.

6. *Амер, И.Ф.* Об одном обобщении k-арного алгоритма Евклида/ И.Ф. Амер, М.К. Аль Анни, А.М. Аль Халиди // Международная конференция по алгебре, анализу и геометрии, КФУ. — 2016. — С. 89–90.

7. *Amer, I.* Selecting the interval of the coefficient y of the k-ary GCD algorithm for natural numbers // *Recent trend in Science and Technology management/ I. Amer.* — 2018. — Pp. 12–15.

8. *Amer, I.* On one acceleration of the k-ary GCD algorithm/ I. Amer // Scientific research of the SCO countries: synergy and integration. — 2018. — Pp. 132–134.

9. *Амер, И.Ф.* Методы ускорения k-арного алгоритма вычисления НОД/ И.Ф. Амер // *Вестник современных исследований.* — 2018. — Т. 7.1 (22). — С. 431–433.

10. *Амер, И.Ф.* О реализации аппроксимирующего k-арного алгоритма вычисления НОД/ И.Ф. Амер // *Вестник современных исследований.* — 2018. — Т. 8.1 (23). — С. 224–228.

11. *Амер, И.* Эффективная реализация аппроксимирующего k-арного алгоритма вычисления НОД в библиотеке MPIR/ И. Амер, А.М. Альхалиди, Ш.Т. Ишмухаметов // *МЦНС Наука и Просвещение.* — 2019. — С. 10–14.