МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Национальный исследовательский
Нижегородский государственный университет им. Н.И. Лобачевского»
(ННГУ)**

Институт информационных технологий, математики и механники

Кафедра: алгебры, геометрии и дискретной математики

Направление подготовки: «Программная инженерия»
Профиль подготовки: «Разработка программно-информационных систем»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

на тему:
«Алгоритмы для нахождения Эрмитовой нормальной формы и ближайшего вектора решетки»

Выполнил(а): студент(ка) группы ____
_____ Д.В. Огнев

Подпись

Научный руководитель:
Доцент, кандидат физико-математических наук

_____ С.И. Весёлов

Подпись

Нижний Новгород
2022

Аннотация (ДОПИСАТЬ)

Тема выпускной квалификационной работы бакалавра – «Алгоритмы для нахождения Эрмитовой нормальной формы и ближайшего вектора решетки».

Ключевые слова: решетки, Эрмитова нормальная форма, проблема ближайшего вектора.

Данная работа посвящена изучению задач теории решеток и методов их решения. В работе изложены основные понятия, связанные с решетками, и разбор алгоритмов для нахождения Эрмитовой нормальной формы и ближайшего вектора решетки.

Целью работы является программная реализация алгоритмов для решения задач.

Объем работы - ...

Содержание

1. Список условных обозначений и сокращений (TODO)

ПБВ (CVP) – проблема ближайшего вектора (Closest vector problem)

ЭНФ (HNF) – Эрмитова нормальная форма (Hermite normal form)

B&B – Branch and bound

2. Введение (TODO)

Текст введения

3. Основные определения (TODO)

Решетка. Пусть $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_n] \in \mathbb{R}^{d \times n}$ - линейно независимые вектора из $\mathbb{R}^d$. Решетка, генерируемая от $\mathbf{B}$ есть набор

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{Bx} : \mathbf{x} \in \mathbb{Z}^n\} = \left\{ \sum_{i=1}^{n} x_i \cdot \mathbf{b}_i : \forall i \; x_i \in \mathbb{Z} \right\}$$

всех целочисленных линейных комбинаций столбцов матрицы $\mathbf{B}$. Матрица $\mathbf{B}$ называется базисом для решетки $\mathcal{L}(\mathbf{B})$. Число n называется рангом решетки. Если $n = d$, то решетка $\mathcal{L}(\mathbf{B})$ называется решеткой полного ранга или полноразмерной решеткой в $\mathbb{R}^d$.

4. Постановка задачи (TODO)

Основная задача - реализовать алгоритмы для нахождения ЭНФ и решения ПБВ за полиномиальное и суперполиномиальное время. Для этого необходимо изучить предложенные в псевдокоде алгоритмы, найти необходимые инструменты для программной реализации, научиться их эффективно использовать и написать программу, реализующие разобранные алгоритмы в виде подключаемой библиотеки. Полученную библиотеку использовать для решения задач теории решеток и найти практическое применение в криптографии.

5. Обзор литературных источников (TODO)

6. Обзор инструментов (TODO)

Для программной реализации был выбран язык C++. Сборка проекта осуществляется с помощью системы сборки CMake. Для работы с матрицами была выбрана библиотека Eigen, для работы с большими числами используется часть библиотеки Boost Boost.Multiprecision, которая подключается в режиме Standalone.

6.1. Обзор библиотеки Eigen

6.2. Обзор библиотеки Boost.Multiprecision

7. Нахождение ЭНФ (TODO)

Будет разобрано два алгоритма - общий и алгоритм для матриц полного ранга строки, который используется в общем алгоритме.

7.1. Алгоритм для матриц полного ранга строки

Дана матрица $\mathbf{B} \in \mathbb{Z}^{m \times n}$. Предположим, что у нас есть процедура AddColumn, которая работает за полиномиальное время и принимает на вход квадратную невырожденную ЭНФ матрицы $\mathbf{H} \in \mathbb{Z}^{m \times m}$ и вектор $\mathbf{b}$, а возвращает ЭНФ матрицы $[\mathbf{H}|\mathbf{b}]$. ЭНФ от $\mathbf{B}$ может быть вычислена следующим образом:

1. Применить алгоритм Грама-Шмидта к столбцам $\mathbf{B}$, чтобы найти m линейно независимых столбцов. Пусть $\mathbf{B}$' - матрица размера $m \times m$, заданная этими столбцами.

2. Вычислить $d = \det(\mathbf{B}')$, используя алгоритм Грама-Шмидта или любую другую процедуру с полиномиальным временем. Пусть $\mathbf{H}_0 = d \cdot \mathbf{I}$ будет диагональной матрицей с d на диагонали.

3. Для $i = 1, \ldots, n$ пусть $\mathbf{H}_i$ это результат применения AddColumn ко входу $\mathbf{H}_{i-1}$ и $\mathbf{b}_i$.

4. Вернуть $\mathbf{H}_n$.

Разберем подпункты:

1. Необходимо найти линейно независимые столбцы матрицы. Их количество всегда будет равно m, т.к. наша матрица полного ранга строки, а значит матрица, состоящая из этих столбцов, будет размера $m \times m$. Для нахождения этих строк можно использовать алгоритм ортогонализации Грама-Шмитда: если $\mathbf{b}_i^* = 0$, то i-ая строка является линейной комбинацией других строк, и ее необходимо удалить. Реализация данного алгоритма находится в пространстве имен Utils в функции get_linearly_independent_columns_by_gram_schmidt. Полученная матрица будет названа $\mathbf{B}$'.

2. Для вычисления det напишем функцию det_by_gram_schmidt, которая принимает на вход матрицу и вычисляет det по формуле $d = \prod_i \|\mathbf{b}_i^*\|$ - сумма произведений длин всех элементов, полученных после применения ортогонализации Грама-Шмидта. Матрица $\mathbf{H_0}$ будет единичной матрицей размера $m \times m$, умноженной на определитель. В результате все диагональные элементы будут равны d.

3. Применяем функцию AddColumn (реализация находится в функции add_column) к $\mathbf{H}_0$ и первому столбцу матрицы $\mathbf{B}$ – $\mathbf{b}_0$, получаем $\mathbf{H}_1$, повторяем для всех столбцов, получаем $\mathbf{H}_n$.

4. $\mathbf{H}_n$ является ЭНФ($\mathbf{B}$).

Алгоритм AddColumn на вход принимает квадратную невырожденную ЭНФ матрицы $\mathbf{H} \in \mathbb{Z}^{m \times m}$ и вектор $\mathbf{b} \in \mathbb{Z}^m$ и работает следующим образом. Если $m = 0$, то тут ничего не надо делать, и мы можем сразу вернуть $\mathbf{H}$. В противном случае, пусть $\mathbf{H} =$ и дальше:

8. Решение ПБВ (TODO)

9. Обзор программной реализации (TODO)

10. Заключение (TODO)

В ходе выполнения выпускной квалификационной работы бакалавра были реализованы алгоритмы на языке C++ для нахождения ЭНФ и решения ПБВ.

Полученную программную реализацию можно использовать как библиотеку и подключать в другие проекты.

Был создан Github репозиторий, который содержит в себе все исходные файлы. Программная реализация использует CMake для автоматической сборки.

11. Список источников (TODO)

Тут будет список источников

Приложения (TODO)

Тут будет листинг кода