Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ВЕРИФИКАЦИЯ АЛГОРИТМА ПОСТРОЕНИЯ БАЗИСА ГРЁБНЕРА И ЕГО ПРИМЕНЕНИЙ В СИСТЕМАХ КОМПЬЮТЕРНОЙ АЛГЕБРЫ НА ЯЗЫКЕ ИНТЕРАКТИВНОГО ДОКАЗАТЕЛЬСТВА ТЕОРЕМ LEAN

Автор: Федоров Глеб Владимирович ______________

Направление подготовки: 01.03.02 Прикладная математика и информатика

Квалификация: Бакалавр

Руководитель ВКР: Трифанов А.И., канд. физ.-мат. наук ______________

Санкт-Петербург, 2022 г.

Обучающийся Федоров Глеб Владимирович

Группа M34351 Факультет ИТиП

Направленность (профиль), специализация

Математические модели и алгоритмы в разработке программного обеспечения

Консультанты:

а) Гилев П.А., без звания ______________

ВКР принята «____» ___________ 20__ г.

Оригинальность ВКР ____%

ВКР выполнена с оценкой ____

Дата защиты «» июня 2022 г.

Секретарь ГЭК Павлова О.Н. ______________

Листов хранения ____

Демонстрационных материалов/Чертежей хранения _____________________

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

УТВЕРЖДАЮ

Руководитель ОП

проф., д.т.н. Парфенов В.Г. _______________

«_____» ____________ 20___ г.

ЗАДАНИЕ

НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Обучающийся Федоров Глеб Владимирович

Группа M34351 **Факультет** ИТиП

Квалификация: Бакалавр

Направление подготовки: 01.03.02 Прикладная математика и информатика

Направленность (профиль) образовательной программы: Математические модели и алгоритмы в разработке программного обеспечения

Тема ВКР: Верификация алгоритма построения базиса Грёбнера и его применений в системах компьютерной алгебры на языке интерактивного доказательства теорем lean

Руководитель Трифанов А.И., канд. физ.-мат. наук, ординарный доцент Университета ИТМО

2 Срок сдачи студентом законченной работы до: «31» мая 2022 г.

3 Техническое задание и исходные данные к работе

4 Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов)

5 Перечень графического материала (с указанием обязательного материала)

Графические материалы и чертежи работой не предусмотрены

6 Исходные материалы и пособия

а) -

7 Дата выдачи задания «22» октября 2022 г.

Руководитель ВКР _______________

Задание принял к исполнению _______________ «22» октября 2022 г.

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

АННОТАЦИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Обучающийся: Федоров Глеб Владимирович

Наименование темы ВКР: Верификация алгоритма построения базиса Грёбнера и его применений в системах компьютерной алгебры на языке интерактивного доказательства теорем lean

Наименование организации, в которой выполнена ВКР: Университет ИТМО

ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

1 Цель исследования: Разработать программное обеспечение на языке lean4, вычисляющее базис Грёбнера. Код данного программного обеспечения должен быть верифицирован на том же языке.

2 Задачи, решаемые в ВКР:

а) Реализация упорядочения lex и grlex для мономов. Доказательство, что реализованные доказательства являются линейными упорядочениями на множестве мономов;

б) Реализация алгоритма деления. Доказательство корректности алгоритма;

в) Реализация алгоритма построения базиса Грёбнера(алгоритм Бухбергера). Доказательство корректности алгоритма;

г) Реализация возможности пользовательского взаимодействия с кодом.

3 Число источников, использованных при составлении обзора: 0

4 Полное число источников, использованных в работе: 0

5 В том числе источников по годам:

Отечественных			**Иностранных**		
Последние 5 лет	От 5 до 10 лет	Более 10 лет	Последние 5 лет	От 5 до 10 лет	Более 10 лет
0	0	0	0	0	0

6 Использование информационных ресурсов Internet: нет

7 Использование современных пакетов компьютерных программ и технологий:

Пакеты компьютерных программ и технологий	**Раздел работы**
Пакет `tabularx` для чуть более продвинутых таблиц	**??**, Приложения А, **??**
Пакет `biblatex` и программное средство `biber`	Список использованных источников

8 Краткая характеристика полученных результатов

9 Гранты, полученные при выполнении работы

10 Наличие публикаций и выступлений на конференциях по теме выпускной работы

-

Обучающийся Федоров Г.В. ______________

Руководитель ВКР Трифанов А.И. ______________

«_____» ____________ 20___ г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 5
1. Инструмент интерактивного доказательства теорем lean4 6
 1.1. Введение 6
 1.2. Основания 6
 1.3. Доказательство в lean4 6
2. Теория базисов Грёбнера 7
 2.1. Основные определения 7
 2.2. Деление полиномов от одной переменной 8
 2.3. Упорядочения мономов 8
 2.4. Алгоритм деления полиномов от нескольких переменных 10
 2.5. Алгоритм Бухбергера 10
 2.6. Решение задачи о принадлежности многочлена идеалу 10
 Выводы по главе 2 10
3. Реализация 11
 3.1. Реализация полинома 11
 3.2. Мономиальные упорядочения 12
 Выводы по главе 3 12
ЗАКЛЮЧЕНИЕ 14
ПРИЛОЖЕНИЕ А. Пример приложения 15

ВВЕДЕНИЕ

В данном разделе размещается введение.

ГЛАВА 1. ИНСТРУМЕНТ ИНТЕРАКТИВНОГО ДОКАЗАТЕЛЬСТВА ТЕОРЕМ LEAN4

В данной главе будет приведён обзор интерактивного помощника доказательства теорем lean4.

1.1. Введение

Язык lean – это функциональный язык программирования, разработанный Microsoft Research. До четвёртой версии данный языке данный язык можно было использовать, в основном, только как интерактивный помощник доказательства теорем, но сейчас он может быть применим как язык общего назначения. Доказательством этому служит тот факт, что текущая версия компилятора lean написана на lean.

Данный язык, помимо Microsoft, поддерживает довольно большое сообщество разработчиков и математиков. Так, например, на lean3 была разработана библиотека mathlib – проект по формализации математики, который позволяет переиспользовать различные математические факты в своей программе.

На момент написания данной работы lean4 находится в финальной стадии разработки, и уже пригоден для использования. Библиотека mathlib так же находится в финальной стадии переписывания на четвёртую версию.

1.2. Основания

описывается что такое исчисление конструкций и как это связано с lean

1.3. Доказательство в lean4

Зачем нужен theorem. Основные тактики, использованные в работе. Тактика rw. Тактика simp. Тактика split. Тактика rename_i. Тактика apply. Тактика exact. Тактика contradiction. Тактика cases.

ГЛАВА 2. ТЕОРИЯ БАЗИСОВ ГРЁБНЕРА

Данная глава будет посвящена введению основных понятий теории колец от нескольких переменных.

2.1. Основные определения

Будем называть вектором степеней конструкцию следующего вида

$$\alpha = (\alpha_1 \dots \alpha_n), \alpha_i \in \mathbb{N}. \tag{1}$$

Назовём вектором переменных следующий вектор

$$x = (x_1 \dots x_n). \tag{2}$$

Мономом от переменных $x_1 \dots x_n$ называется конструкция следующего вида

$$x^\alpha = (x_1^{\alpha_1} \dots x_n^{\alpha_n}). \tag{3}$$

Полиномом f, с коэффициентами из поля K, называется конечная линейная комбинация мономов, которая записывается следующим образом

$$f = \sum_\alpha c_\alpha * x^\alpha, c_\alpha \in K. \tag{4}$$

Множеством всех полиномов от переменных $x_1 \dots x_n$ над полем K будем обозначать как $K[x_1 \dots x_n]$. Отметим, что на данном множестве можно естественным образом ввести операции $+$ и $*$ таким образом, чтобы структура $\langle K[x_1 \dots x_n], +, * \rangle$ удовлетворяла аксиомам кольца.

Подмножество $I \subset K[x_1 \dots x_n]$ называется идеалом, если выполнены следующие условия:

а) $0 \in I$;

б) если $f, g \in I$, то $f + g \in I$;

в) если $f \in I$ и $h \in K[x_1 \dots x_n]$, то $hf \in I$.

Пусть $f_1 \dots f_s$ – набор полиномов от нескольких переменных, тогда множество $\langle f_1 \dots f_s \rangle = \{\sum_i^s h_i * f_i | h_1 \dots h_s \in K[x_1 \dots x_n]\}$ является идеалом в $K[x_1 \dots x_n]$, а полиномы $\langle f_1 \dots f_s \rangle$ называются образующими идеала.

2.2. Деление полиномов от одной переменной

Теория базисов Грёбнера во многом опирается на операцию деления многочленов. Перед тем, как определить алгоритм деления в кольце полиномов от нескольких переменных рассмотрим алгоритм в кольце полиномов от одной переменной.

Важнейшей частью алгоритма является понятие старшего члена полинома. Пусть $f = \alpha_0 x^m + \alpha_1 x^{m-1} \ldots + a_m$, где $a_i \in K$, $a_0 \neq 0$. Тогда

$$LT(f) = \alpha_0 x^m$$

называется старшим членом полинома f.

Опишем алгоритм деления в $K[x]$. Пусть $g \in K[x]$ – ненулевой полином. Тогда любой полином $f \in K[x]$ может быть записан в виде

$$f = qg + r,$$

где $q, r \in K[x]$ и либо $r = 0$, либо $deg(r) < deg(g)$, причём q и r определены однозначно. Многочлены q и r могут быть найдены следующим алгоритмом.

Листинг 1 – Деление в $K[x]$

```
function Divide(q, f)
    q = 0;
    r = f;
    while r ≠ 0  &  LT(g)|LT(r) do
        q = q + LT(r)/LT(g)
        r = r − (LT(r)/LT(g))g
    end while
    return q, r
end function
```

Доказательство корректности данного алгоритма можно найти в ...

2.3. Упорядочения мономов

После прочтения предыдущего параграфа может сложится впечатление, что алгоритм полиномов из $K[x]$ будет работать и в $K[x_1 \ldots x_n]$. К сожалению, это не совсем так. Заметим, что в кольце $K[x]$ у мономов есть естественный порядок – по степеням, которые являются натуральными числами. Но в

$K[x_1 \dots x_n]$ степень монома – не число. Поэтому, в данном параграфе будет дано определение упорядочения мономов в $K[x_1 \dots x_n]$.

Мономиальным упорядочением на $K[x_1 \dots x_n]$ называется любое бинарное отношение $\leqslant$ на $\mathbb{Z}^n_{\geqslant 0}$, обладающее следующими свойствами:

а) $\leqslant$ является линейным упорядочением на $\mathbb{Z}^n_{\geqslant 0}$;

б) если $\alpha \leqslant \beta$ и $\gamma \in \mathbb{Z}^n_{\geqslant 0}$, то $\alpha + \gamma \leqslant \beta + \gamma$;

в) $\leqslant$ вполне упорядочивает $\mathbb{Z}^n_{\geqslant 0}$.

Условие а нужно для того, чтобы мы могли для любого полинома расположить мономы в порядке $\leqslant$. То есть, для любой пары мономов x^α, y^β должно выполняться одно из следующих соотношений

$$x^\alpha < y^\beta, x^\alpha = y^\beta, x^\alpha > y^\beta$$

.

Условие б нужно для того, чтобы упорядоченность мономов была согласована с аксиомами кольца. Заметим, что задача умножения полинома на полинома естественным образом сводится к задаче умножения полинома на моном. Но, если упорядочение не удовлетворяет свойству б, то умножение монома на старший член полинома не будет являться старшим членом. Примером упорядочения, не удовлетворяющего свойству б, может служить max упорядочение: $\leqslant_{max}$: $\alpha \leqslant_{max} \beta \Leftrightarrow max(\alpha) \leqslant max(\beta) \vee max(\alpha) = max(\beta) \wedge \alpha \leqslant_{lex} \beta$. Тогда $x^2y^3 \leqslant_{max} xyz^5$, но $x^2y^3 * y^8 \geqslant_{max} xyz^5 * y^8$.

Условие в необходимо для доказательства корректности алгоритмов их следующих параграфов. А именно, критерий остановки остановки алгоритма будет основан на том, что старший член полинома убывает на каждом шаге алгоритма.

В данной работе будут рассмотрены два упорядочения:

а) Лексикографическое упорядочение – $a \leqslant_{lex} b \Leftrightarrow$ первая ненулевая координата вектора $b - a$ положительна;

б) Градуированное лексикографическое упорядочение – $a \leqslant_{grlex} b \Leftrightarrow$ $|a| < |b| \vee (|a| = |b| \wedge a \leqslant_{lex} b)$.

Доказательства того, что эти упорядочения удовлетворяют условиям, определённым выше, будут предъявлены в следующей главе.

2.4. Алгоритм деления полиномов от нескольких переменных

Перейдём к алгоритму деления полиномов от нескольких переменных. Зафиксируем некоторое мономиальное упорядочение $\leqslant$. Пусть $F = (f_1 \dots f_s)$ – упорядоченный набор полиномов из $k[x_1 \dots x_n]$. Тогда любой полином $f \in k[x_1 \dots x_n]$ может быть представлен в виде

$$f = \alpha_1 f_1 + \ldots + \alpha_s f_s + r,$$

где $a_i, r \in k[x_1 \dots x_n]$, причём либо $r = 0$, либо r есть линейная комбинация мономов, ни один из которых не делится ни на один из старших членов $LT(f_1) \dots LT(f_s)$. Данно представление можно найти следующим алгоритмом:

Листинг 2 – Алгоритм деления

2.5. Алгоритм Бухбергера

2.6. Решение задачи о принадлежности многочлена идеалу

Выводы по главе 2

Вывод:

ГЛАВА 3. РЕАЛИЗАЦИЯ

Данная глава будет посвящена введению реализации описанной выше теории на языке интерактивного доказательства теорем lean. В данной работе будут рассматриваться только полиномы над полем рациональных чисел.

3.1. Реализация полинома

В библиотеке mathlib уже есть реализация полинома от нескольких переменных под названием MvPolynomial, которая удовлетворяет всем аксиомам кольца. Но, к сожалению, это реализация явно использует аксиому выбора, что делает её неконструктивной. Иначе говоря, её можно использовать только для доказательства теорем, но сгенерировать исполняемый код при её использовании не выйдет. Поэтому в данной работе была написана собственная реализация.

Введём основные определения. Вектором степеней назовём вектор натуральных чисел длины n упорядоченный согласно ord. Произведением двух векторов одинаковой длины с одинаковым упорядочением назовём их покомпонентную сумму. Мы называем сумму векторов произведением потому-что здесь имеется ввиду произведение $x_1^{\alpha_1} \dots x_n^{\alpha_n}$ на $x_1^{\beta_1} \dots x_n^{\beta_n}$, что равно $x_1^{\alpha_1+\beta_1} \dots x_n^{\alpha_n+\beta_n}$.

Листинг 3 – Вектор степеней

```
def Variables (n: Nat) (ord: Type) := Vector Nat n

def Variables.mul (v1 v2: Variables n ord): Variables n ord :=
  map2 (fun x y => x + y) v1 v2
```

Заметим, что вектор переменных нам нужен только для удобства восприятия. В работе алгоритмов он никак не учавствует. Поэтому мы можем определить моном как пару из рационального числа и вектора степеней длинны n упорядоченную согласно ord. Произведение мономов определенно естественным образом.

Полином был основан на основе красно-чёрного дерева. А именно, полином – это красно-чёрное дерево, элемены которого – это мономы, у которых ровно n переменных. Причём функция сравнения – это некоторое мономиальное упорядочение ord.

Основные функции работы с полиномом можно найти в приложении ......

Листинг 4 – Моном

```
def Monomial (n: Nat) (ord: Type) := Rat × (Variables n ord)

def Monomial.mul (m1 m1: Monomial n ord) : Monomial n ord :=
  (m1.fst * m2.fst , Variables.mul m1.snd m2.snd)
```

Листинг 5 – Моном

```
def Polynomial (n: Nat) (ord: Type) [MonomialOrder $ Variables n
   ord] := Std.RBSet (Monomial n ord) ordering.m_cmp
```

3.2. Мономиальные упорядочения

Для доказательства того, что реализованные упорядочения удовлетворяют аксиомам мономиального упорядочения, был реализован typeclass(класс типов) MonomialOrder, наследованный от typeclass LinearOrder(линейное упорядочение) и typeclass WellFoundedRelation(вполне упорядочивание). Для LinearOrder lean накладывает дополнительные ограничения. А именно:

а) Отношение $\leqslant$ должно быть вычислимым(decidable);

б) Lean автоматически строит отношение $<$. Поэтому, нужно проверка согласованности отношений $<$ и $\leqslant$. А именно – $a < b \Leftrightarrow (a \leqslant b \wedge b \nleqslant a)$.

В данной работе были определены два мономиальных упорядочения: лексикографическое(далее Lex) и градуированное лексикогрфическое(далее GrLex).

Листинг 6 – Lex упорядочение

```
def Order.lex_impl (v1 v2: Vector Nat n): Prop :=
  match v1, v2 with
    | <[], _>  , <[], _>   => True
    | <x::_, _>, <y::_, _> => if x = y then lex_impl v1.tail v2.
       tail
                              else x ≤ y

def Order.lex (v1 v2: Variables n order.Lex): Prop := Order.
   lex_impl v1 v2
```

Выводы по главе 3

Вывод:

Листинг 7 – Lex упорядочение

```
def Order.lex_impl (v1 v2: Vector Nat n): Prop :=
  match v1, v2 with
    | <[], _>  , <[], _>   => True
    | <x::_, _>, <y::_, _> => if x = y then lex_impl v1.tail v2.
       tail
                              else x ≤ y

def Order.lex (v1 v2: Variables n order.Lex): Prop := Order.
   lex_impl v1 v2
```

Листинг 8 – GrLex упорядочение

```
def Order.grlex (vs1 vs2: Variables n order.GrLex): Prop :=
  let sum1 := elem_sum vs1
  let sum2 := elem_sum vs2
  if sum1 < sum2 then True
  else if sum1 = sum2 then if Order.lex vs1 vs2 then True
                           else False
  else False
  where
    elem_sum (vs: Variables n order.GrLex): Nat :=
      List.foldl (fun x y => x + y) 0 vs.toList
```

ЗАКЛЮЧЕНИЕ

В данном разделе размещается заключение.

ПРИЛОЖЕНИЕ А. ПРИМЕР ПРИЛОЖЕНИЯ