1. Министерство науки и высшего образования Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт компьютерных наук и кибербезопасности

РАСЧЕТНОЕ ЗАДАНИЕ № 4

«**Базис Гребнера**»

1. по дисциплине «Алгебраические основы криптографии»
2. Выполнил
3. студент гр. 5151004/10101 Веремейчук Я.Ю.

<подпись>

1. Проверил
2. Старший преподаватель Ярмак А.В.

<подпись>

1. Санкт-Петербург
2. 2024
3. Цель работы

Цель данной работы состоит в изучении способа построения базиса Гребнера.

1. Задача

Разработать программу, которая проверяет, является ли заданное множество базисом Гребнера при лексикографическом упорядочении.

1. Теоретические исследования

Конечное множество 𝐺 = {𝑔1, 𝑔2, . . , 𝑔𝑡 } является базисом Гребнера для полиномиального идеала 𝐼 , если старший член любого из полиномов идеала 𝐼 делится на старший член одного ил многочленов 𝐺 .

1. Ход работы

Была разработана программа на языке Python с использованием библиотеки groebner.

При попытке построить базис из множества, которое уже должно являться базисом, алгоритм нахождения базиса возвращает это же множество. В противном случае получается новый базис.

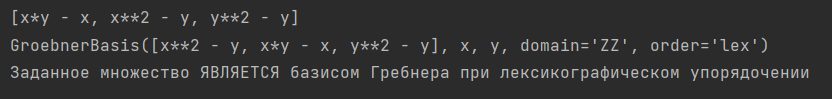


Рисунок 1 – Пример запуска программы

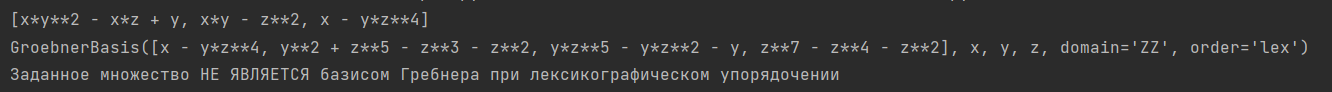


Рисунок 2 – Пример запуска программы

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 3 – Реализуемые алгоритмы

По умолчанию используется алгоритм Бухбергера.

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

Рисунок 4 – Алгоритм Бухбергера

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 5 – Расчет S-полинома в алгоритме Бухбергера

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 6 – Реализация функции нахождения S-полинома

Также доступно нахождение базиса с помощью алгоритма f5b.

f5b – это реализация алгоритма F5B, разработанного Яо Суном и Динкан Вангом. Аналогично алгоритму Бухбергера, алгоритм работает путем вычисления критических пар, вычисления S-многочлена, уменьшения его и присоединения к уменьшенному S-многочлену, если он не равен 0.

В отличие от алгоритма Бухбергера, каждый многочлен содержит дополнительную информацию, а именно сигнатуру и число. Подпись указывает путь вычисления (т.е. из какого многочлена в исходном базисе он был получен и как), число указывает, когда многочлен был добавлен к базису. С помощью этой информации (часто) можно решить, уменьшится ли значение S-полинома до 0 и можно ли его отбросить.

Оптимизации включают в себя:

* cокращение числа генераторов перед вычислением базиса Гребнера,
* удаление избыточных критических пар при вводе в базис нового многочлена и сортировку критических пар и текущего базиса.

Как только будет найден базис Гребнера, он будет сокращен.

Приложение А

Листинг программы

from sympy import groebner  
from sympy.abc import x, y, z  
  
polynomials = [x\*y - x, x\*\*2 - y, y\*\*2 - y] # yes  
# polynomials = [y - x\*\*2, z - x\*\*3] # no  
# polynomials = [x\*(y\*\*2) - x\*z + y, x\*y - z\*\*2, x - y\*(z\*\*4)] # no  
print(polynomials)  
  
G = groebner(polynomials, order='lex')  
print(G)  
  
G\_polynomials = list(G.args[0])  
  
polynomials = set(polynomials)  
G\_polynomials = set(G\_polynomials)  
  
if (G\_polynomials==polynomials): print("Заданное множество ЯВЛЯЕТСЯ базисом Гребнера при лексикографическом упорядочении")  
else: print("Заданное множество НЕ ЯВЛЯЕТСЯ базисом Гребнера при лексикографическом упорядочении")