МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Алгоритм разложения на свободные от квадратов множители

ЛАБОРАТОРНАЯ РАБОТА

студента 4 курса 431 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Серебрякова Алексея Владимировича

Научный руководитель
доцент, к. п. н. _____ А. С. Гераськин
подпись, дата

Саратов 2022

PSQFF. Разложение полиномов на свободные от квадратов множители (Polynomial Squarefree Factorization)

Вход: $p(x)$ — примитивный полином положительной степени от одной переменной над областью J характеристики нуль с однозначным разложением на множители.

Выход: Полиномы $s_i(x)$ и число e, такие, что $p(x) = \prod_{1 \leqslant i \leqslant e} [s_i(x)]^i$ — разложение полинома $p(x)$ на свободные от квадратов множители.

1. [Инициализация] $r(x) := \gcd[p(x), p'(x)]$; $t(x) := p(x)/r(x)$; $j := 1$.
2. [Конец?] Если $\deg[r(x)] = 0$, то $\{e := j$; $s_j(x) = t(x)$; выход;$\}$.
3. [Вычисление $s_j(x)$] $v(x) := \gcd[r(x), t(x)]$; $s_j(x) := t(x)/v(x)$.
4. [Обновление] $r(x) := r(x)/v(x)$; $t(x) := v(x)$; $j := j + 1$; перейти к шагу 2.

```
[alse0722@alse0722-ms7a34 coding_teory]$ /bin/python /home/alse0722/Desktop/univer/coding_teory/fin/n17.py
[1, -1, -2, 2, 1, -1]
(1, [([1, 1], 2), ([1, 2], 3)])
[alse0722@alse0722-ms7a34 coding_teory]$ /bin/python /home/alse0722/Desktop/univer/coding_teory/fin/n17.py
[1, 1, -1, -2, 2, 1, -1]
(1, [([1, 1, 2, 1, 2, 1, 2], 1)])
[alse0722@alse0722-ms7a34 coding_teory]$ /bin/python /home/alse0722/Desktop/univer/coding_teory/fin/n17.py
[2, 1, -1]
(2, [([1, 1], 2)])
[alse0722@alse0722-ms7a34 coding_teory]$
```

```
from sympy.polys.galoistools import gf_monic
from sympy.polys.galoistools import gf_degree
from sympy.polys.galoistools import gf_diff
from sympy.polys.galoistools import gf_gcd
from sympy.polys.galoistools import gf_quo
from sympy.polys.domains import ZZ


def gf_sqf_list(f, p, K, all=False):

    n, sqf, factors, r = 1, False, [], int(p)

    lc, f = gf_monic(f, p, K)

    if gf_degree(f) < 1:
        return lc, []

    while True:
        F = gf_diff(f, p, K)

        if F != []:
            g = gf_gcd(f, F, p, K)
            h = gf_quo(f, g, p, K)

            i = 1

            while h != [K.one]:
                G = gf_gcd(g, h, p, K)
                H = gf_quo(h, G, p, K)
```

```
                if gf_degree(H) > 0:
                    factors.append((H, i*n))

                g, h, i = gf_quo(g, G, p, K), G, i + 1

            if g == [K.one]:
                sqf = True
            else:
                f = g

        if not sqf:
            d = gf_degree(f) // r

            for i in range(0, d + 1):
                f[i] = f[i*r]

            f, n = f[:d + 1], n*r
        else:
            break

    if all:
        raise ValueError("'all=True' is not supported yet")

    return lc, factors


f = [ 2, 1, -1]
print(f)
print(gf_sqf_list(f, 3, ZZ))
```