Contents

1 Functions **2**

1.1 poly.factor – polynomial factorization 2

1.1.1 brute_force_search – search factorization by brute force . 2

1.1.2 divisibility_test – divisibility test 2

1.1.3 minimum_absolute_injection – send coefficients to minimum absolute representation 3

1.1.4 padic_factorization – p-adic factorization 3

1.1.5 upper_bound_of_coefficient –Landau-Mignotte bound of coefficients 3

1.1.6 zassenhaus – squarefree integer polynomial factorization by Zassenhaus method 3

1.1.7 integerpolynomialfactorization – Integer polynomial factorization 4

Chapter 1

Functions

1.1 poly.factor – polynomial factorization

The factor module is for factorizations of integer coefficient univariate polynomials.

This module using following type:

polynomial :
`polynomial` is the polynomial generated by function poly.uniutil.polynomial.

1.1.1 brute_force_search – search factorization by brute force

brute_force_search(`f`: ***poly.uniutil.IntegerPolynomial***, `fp_factors`: ***list***, `q`: ***integer***)
→ ***[factors]***

Find the factorization of `f` by searching a factor which is a product of some combination in `fp_factors`. The combination is searched by brute force.
The argument `fp_factors` is a list of poly.uniutil.FinitePrimeFieldPolynomial
.

1.1.2 divisibility_test – divisibility test

divisibility_test(`f`: ***polynomial***, `g`: ***polynomial***) → ***bool***

Return boolean value whether `f` is divisible by `g` or not, for polynomials.

1.1.3 minimum_absolute_injection – send coefficients to minimum absolute representation

minimum_absolute_injection(`f`**:** ***polynomial*****)** → ***F***

Return an integer coefficient polynomial F by injection of a $\mathbf{Z}/p\mathbf{Z}$ coefficient polynomial `f` with sending each coefficient to minimum absolute representatives.

The coefficient ring of given polynomial `f` must be intresidue.IntegerResidueClassRing or finitefield.FinitePrimeField .

1.1.4 padic_factorization – p-adic factorization

padic_factorization(`f`**:** ***polynomial*****)** → ***p*, *factors***

Return a prime `p` and a p-adic factorization of given integer coefficient squarefree polynomial `f`. The result `factors` have integer coefficients, injected from $\mathbb{F}_p$ to its minimum absolute representation.

†The prime is chosen to be:

1. `f` is still squarefree mod `p`,
2. the number of factors is not greater than with the successive prime.

The given polynomial `f` must be poly.uniutil.IntegerPolynomial .

1.1.5 upper_bound_of_coefficient –Landau-Mignotte bound of coefficients

upper_bound_of_coefficient(`f`**:** ***polynomial*****)** → ***long***

Compute Landau-Mignotte bound of coefficients of factors, whose degree is no greater than half of the given `f`.

The given polynomial `f` must have integer coefficients.

1.1.6 zassenhaus – squarefree integer polynomial factorization by Zassenhaus method

zassenhaus(`f`**:** ***polynomial*****)** → ***list of factors f***

Factor a squarefree integer coefficient polynomial `f` with Berlekamp-Zassenhaus method.

1.1.7 integerpolynomialfactorization – Integer polynomial factorization

integerpolynomialfactorization(`f`**:** ***polynomial*****)** $\rightarrow$ ***factor***

Factor an integer coefficient polynomial `f` with Berlekamp-Zassenhaus method.

factor output by the form of list of tuples that formed (factor, index).

Bibliography