

Basics of polynomials

Application of discrete models

1 Introduction

Definition 1.1. A mathematical expression is called a **polynomial** if it contains only the following:

- constants (e.g. numbers),
- a variable (e.g. x),
- addition, subtraction, multiplication,
- exponentiation to an exponent in $\mathbb{N}$,
- parentheses.

Example. $3x^5 - 5x^3 + x + 1/2$, $x^2(x+3)^5$ and 15 are polynomials, but the following ones are not: $1/x$ (it contains division), x^{-5} (the exponent is not in $\mathbb{N}$) and $\sqrt{x}$ (square root is not allowed).

Note: this definition can be generalized to allow more than one variables, e.g. then $x^3 + xy^2 + y$ is a *bivariate polynomial*, i.e. a polynomial with two variables. Then, the above definition defines *univariate polynomials*, i.e. polynomials with one variable. But in this subject, we only deal with the latter, so we omit the qualifier "univariate".

Definition 1.2. Let $K[x]$ be the set of polynomials whose variable is x , and whose constants are all from the set K . Terminology: $K[x]$ is the set of polynomials **over** K .

Example. $\mathbb{R}[x]$ is the set of polynomials with real numbers, and $\mathbb{Z}[x]$ is its subset where the constants can only be integers. For example, $2x^2 - 5 \in \mathbb{Z}[x]$ and $\sqrt{2}x + 3/2 \in \mathbb{R}[x]$, but $\sqrt{2}x + 3/2 \notin \mathbb{Z}[x]$.

The same polynomial can be written in multiple forms, for example:

$$(2x + 1)^2 = 4x^2 + 4x + 1 = 4x + 1 + 4x^2 = x^5 + 1 + x - x^5 + 4x^2 + 3x.$$

If we need a unique representation, then we can use the following form:

Definition 1.3. The **canonical form** of a polynomial $p \in K[x]$ is the following:

$$p = a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0,$$

where all $a_k \in K$ and $a_n \neq 0$.

Example. The canonical form of $p = (2x + 1)^2$ is $p = 4x^2 + 4x + 1$.

Any polynomial can be converted to canonical form by the following steps: expand the brackets, perform the multiplications in each term (e.g. $x \cdot 2x \cdot 3 = 6x^2$), combine the terms with equal powers, and sort the terms in decreasing powers of x .

Definition 1.4. In the above canonical form:

- a_k 's are the **coefficients** of the polynomial (specifically, a_k is the **coefficient of degree k**);
- n is the **degree** of the polynomial (notation: $\deg p$);
- a_n is the **leading coefficient** of the polynomial;
- a_0 is the **constant term** of the polynomial.

Example. The coefficients of $p = 2x^3 - 6x + 4$ are 2, 0, -6 and 4 (note the 0 for the missing x^2 term), its degree is 3, its leading coefficient is 2 and its constant term is 4.

But there is a polynomial which, according to the above definition, cannot be written in canonical form: the 0 (because we required that $a_n \neq 0$). So we need to handle it separately:

Definition 1.5. $0 \in K[x]$ is the **zero polynomial**, its canonical form is 0, and its degree is $-\infty$.

(Note: it is $-\infty$ to make it smaller than for any other polynomial. It cannot be defined 0, because that would break certain statements that are true for all other polynomials, see e.g. the degree theorem about polynomial multiplication in the next document. Some sources leave the degree of the zero polynomial undefined, and Sage defines it to be -1 .)

Definition 1.6.

- If $\deg p \leq 0$, then p is called a **constant polynomial**;
- if $\deg p = 1$, then p is called a **linear polynomial**;
- if $\deg p = 2$, then p is called a **quadratic polynomial**;
- if $\deg p = 3$, then p is called a **cubic polynomial**.

Example. 2 and 0 are constant polynomials, $2x - 3$ and x are linear polynomials, $x^2 + 2x + 1$ is a quadratic polynomial, and $2x^3 - 1$ is a cubic polynomial.

2 Polynomials over rings

We have not yet specified exactly what kind of set K can be in the notation $K[x]$. For example, it can be one of the known number sets: $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ or $\mathbb{C}$.

To discuss polynomials in the most general way, we only assume that K is an arbitrary *commutative ring with identity*. (Rings and their properties were introduced in the previous document.)

Example. $\mathbb{Z}_5[x]$ is the set of polynomials over the congruence classes modulo 5 ($\mathbb{Z}_5 = \{\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}\}$, see: Congruences), e.g. $p = \bar{4}x^2 + \bar{2}x + \bar{3} \in \mathbb{Z}_5[x]$. It works similarly to $\mathbb{Z}[x]$, except that the coefficients must be taken modulo 5, e.g. $p + p = \bar{3}x^2 + \bar{4}x + \bar{1}$, because $\bar{4} + \bar{4} = \bar{3}$ etc.

Note: the reason for requiring K to be commutative and to have an identity element is that otherwise even the most basic properties would be problematic. For example, without commutativity, we wouldn't be able to convert all polynomials to canonical form, e.g. even the $2x \cdot 2x = 4x^2$ simplification uses that $x \cdot 2 = 2 \cdot x$. And if K had no identity element, then e.g. the polynomial x wouldn't have a leading coefficient (which is 1 in $\mathbb{Z}[x]$, because $x = 1x$).

It is easy to prove that $K[x]$ inherits most properties of the coefficient ring K :

Theorem 2.1 (properties of $K[x]$).

1. If K is a ring, then $K[x]$ is also a ring.
2. If K is a commutative ring, then $K[x]$ is also a commutative ring.
3. If K is a ring with identity, then $K[x]$ is also a ring with identity (with the same identity).
4. If K has no zero divisors, then $K[x]$ has no zero divisors either.
5. If K is an integral domain, then $K[x]$ is also an integral domain.

But there is one property that cannot be inherited: $K[x]$ is never a *field*, not even if K is a field. The reason for this is that most polynomials, including $x \in K[x]$ have no multiplicative inverse (because if we multiply x by any polynomial, we never get the identity element, which is a constant polynomial – or put it differently, $1/x$ is not a polynomial).

Example. $\mathbb{Z}[x]$, $\mathbb{Q}[x]$, $\mathbb{R}[x]$ and $\mathbb{C}[x]$ are integral domains, but none of them are fields.

Example. $\mathbb{Z}_p[x]$ (for prime p) is also an integral domain, but not a field (even though $\mathbb{Z}_p$ is a field).

3 Polynomials in Sage

Sage provides multiple ways to deal with polynomials. The "naive" way is to use the general symbolic expressions:

```
sage: (2*x - 1)*(x^2 + 2)
(x^2 + 2)*(2*x - 1)
sage: expand(_)
2*x^3 - x^2 + 4*x - 2
sage: type(_)
<class 'sage.symbolic.expression.Expression'>
```

(Reminder: `x` is a built-in symbolic variable, as if created by `var("x")`, see the first document: SageMath introduction.)

But this is slightly cumbersome (e.g. it requires `expand()` after each multiplication), and not very efficient for polynomials. Instead, we can use Sage's structures that are optimized specifically for polynomials: `ZZ["x"]`, `QQ["x"]` etc., corresponding to the mathematical sets $\mathbb{Z}[x]$, $\mathbb{Q}[x]$ etc.:

```
sage: P = ZZ["x"]
sage: p = P(2*x - 1)      # convert a symbolic expression to a polynomial
sage: p
2*x - 1
sage: q = P([2, 0, 1])    # create from coefficients (starting from the constant)
sage: q
x^2 + 2
sage: p * q                # no need for expand() (and much faster)
2*x^3 - x^2 + 4*x - 2
sage: type(_)
<class 'sage.rings.polynomial.[...]'>
```

Warning: `x` has two different meanings here: in the entered commands, `x` is still a symbolic variable (type: `Expression`), but in the printed results, `x` is the variable of the polynomials (type: `polynomial`). Don't mix them:

```
sage: p*(x^2 + 2)          # DON'T do this! (polynomial * symbolic expression)
(x^2 + 2)*(2*x - 1)
sage: type(_)              # the optimized polynomial structure is lost
<class 'sage.symbolic.expression.Expression'>
```

Note: as a slightly simplified explanation, Sage represents a symbolic expression as a sequence of operations, e.g. $x^2 + 2$ is represented as "square x , then add 2". This is very general, but not very efficient to calculate with. Polynomials on the other hand are represented with the sequence of coefficients, e.g. $x^2 + 2$ is represented as `2, 0, 1`. This makes much more efficient calculations possible (see the next document).

To avoid accidentally mixing symbolic expressions with polynomials, we can redefine `x` to be a polynomial too:

```
sage: P = ZZ["x"]          # let P be the set of integer polynomials
sage: x = P.gen()          # let x be the variable of the polynomials in P
```

There is also a simplified syntax that defines both P and x :

```
sage: P.<x> = ZZ[]      # same as: P = ZZ["x"]; x = P.gen()
sage: P
Univariate Polynomial Ring in x over Integer Ring
sage: type(x)           # x is also redefined
<class 'sage.rings.polynomial.[...]'>
sage: (2*x - 1)*(x^2 + 2) # now these are polynomials too
2*x^3 - x^2 + 4*x - 2
```

We can also have polynomials whose coefficients follow modular arithmetic, e.g. $\mathbb{Z}_5[x]$:

```
sage: P.<x> = Zmod(5)[]
sage: p = 9*x^2 - 3*x + 33
sage: p
4*x^2 + 2*x + 3
sage: p + p
3*x^2 + 4*x + 1
```

We can query various properties of polynomials:

```
sage: P.<x> = ZZ[]
sage: p = 2*x^3 - 6*x + 4
sage: p.degree()
3
sage: p.list()      # list of coefficients (starting from the constant term)
[4, -6, 0, 2]
sage: p[1]          # coefficient of degree 1
-6
sage: p.leading_coefficient() # or: p.lc()
2
sage: p.constant_coefficient() # or: p[0]
4
```

4 Roots

Definition 4.1. The **value** of the polynomial $p \in K[x]$ at c is obtained by replacing all occurrences of x in p by the value c . Notation: $p(c)$. This computation is called the **evaluation** of p at c .

Example. If $p = 2x^2 - 3x + 1$ and $c = 2$, then $p(2) = 2 \cdot 2^2 - 3 \cdot 2 + 1 = 3$.

Example. It is not necessary that $c \in K$: we can evaluate the same $p = 2x^2 - 3x + 1 \in \mathbb{Z}[x]$ at $c = 1/3 \notin \mathbb{Z}$ too: $p(1/3) = 2 \cdot (1/3)^2 - 3 \cdot (1/3) + 1 = 2/9$.

Definition 4.2. The value c is a **root** of the polynomial $p \in K[x]$ if $p(c) = 0$.

Example. $c = 1$ is a root of $p = 2x^2 - 3x + 1$, because $p(1) = 2 \cdot 1^2 - 3 \cdot 1 + 1 = 0$.

Example. The same $p \in \mathbb{Z}[x]$ has another root, $c = 1/2 \notin \mathbb{Z}$: $p(1/2) = 2 \cdot (1/2)^2 - 3 \cdot (1/2) + 1 = 0$.

Example. $q = x^2 + 1 \in \mathbb{R}[x]$ has no real roots, only complex: $x_1 = i$ and $x_2 = -i$.

Theorem 4.3. $c \in K$ is a root of $p \in K[x]$ if and only if $\exists q \in K[x] : p = (x - c)q$.

Example. $c = 1$ is a root of the above $p = 2x^2 - 3x + 1$, and indeed: $p = (x - 1)(2x - 1)$.

Definition 4.4. The polynomial $x - c \in K[x]$ is called the **root factor** corresponding to c .

That is, c is a root of p if and only if the root factor $x - c$ can be extracted from p .

It might happen that c is also root of the polynomial q left after the extraction $p = (x - c)q$. In this case, the same root factor can be extracted again: $p = (x - c)q = (x - c)^2 r$. Then we say that c is a **multiple root** of p , or more precisely:

Definition 4.5. $c \in K$ is a **root of multiplicity k** of the polynomial $p \in K[x]$ if $\exists q \in K[x]$ such that $p = (x - c)^k q$, but c is not a root of q . Terminology: if $k = 1$, then c is a **simple root**, if $k = 2$, then it is a **double root** etc., and if $k \geq 2$, then c is a **multiple root** of p .

Example. $c = 1$ is a double root of $p = 2x^3 - 6x + 4$, because $p = (x - 1)^2(2x + 4)$, but 1 is not a root of $2x + 4$.

That is, c is a root of multiplicity k if $x - c$ can be extracted exactly k times.

A very important theorem follows from the root factor extraction:

Theorem 4.6. If K is an integral domain (e.g. a number ring: $K = \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$), then a polynomial $p \in K[x] \setminus \{0\}$ of degree n can have at most n roots in K . This is true even if they are counted with multiplicities, i.e. the sum of the multiplicities of all roots is also at most n .

Example. A cubic polynomial can have at most 3 roots. And if it has a double root (e.g. $c = 1$ for the above p), then it can only have one more root, because the double root "counts twice".

(Proof: extract as many root factors from p as possible: $p = (x - c_1)(x - c_2) \cdots (x - c_m)q$, where q has no more roots. Now what roots can p have? If $p(c) = 0$, then one of the factors must be 0 (because K is an integral domain, i.e. it has no zero divisors). But q cannot be 0 (it has no roots), so c must be equal to one of $c_1, c_2, \dots, c_m$. And there are at most n of them, because $n = \deg p$, and each extraction decreases the degree by one.)

Note: the theorem requires that K has no zero divisors (i.e. it is an integral domain, see the definition in the previous document). Otherwise, there may be more than n roots:

Example. In $\mathbb{Z}_6[x]$ (which has zero divisors: $\bar{2} \cdot \bar{3} = \bar{0}$), the polynomial $p = x^2 + x$ has not two, but four roots: $\bar{0}, \bar{2}, \bar{3}, \bar{5} \in \mathbb{Z}_6$.

Here are some special cases for the number of roots:

- A quadratic polynomial ($p = ax^2 + bx + c$) can have at most two roots (two simple or one double root), as we know from the quadratic formula.
- A polynomial of degree one, i.e. $p = ax + b$ can have at most one root (a simple root): $x = -b/a$ (it is "at most" one, because the division may not be valid in rings like $\mathbb{Z}$).
- A polynomial of degree zero (i.e. a nonzero constant polynomial, e.g. $p = 5$) has no roots.
- But the zero polynomial ($p = 0$) has all $c \in K$ as roots (this is why it had to be excluded from the above theorem).

Earlier, we defined the canonical form of polynomials. Now we define another important form:

Definition 4.7. The **root factor form** of a polynomial $p \in K[x]$ is the following:

$$p = a(x - c_1)(x - c_2) \cdots (x - c_m),$$

where $a, c_1, c_2, \dots, c_m \in K$ and $a \neq 0$.

Example. The polynomial $p = 2x^3 - 6x + 4$ has the root factor form $p = 2(x - 1)^2(x + 2)$.

The point of this form is that it shows all roots of the polynomial, with multiplicities.

But the root factor form doesn't always exist, only if the polynomial has at least n roots in the base set K (counted with multiplicities). And it exists *uniquely* (up to the order) if it has *exactly* n roots. Note that the above theorem only guarantees that it has *at most* n roots, and only if K is an integral domain. Also, the number of roots depends on the base set K :

Example. The polynomial $p = 16x^4 - 1$ has

- no roots in $\mathbb{Z}$;
- two simple roots in $\mathbb{Q}$ and $\mathbb{R}$: $x_1 = 1/2$ and $x_2 = -1/2$;
- and four roots in $\mathbb{C}$: $x_1 = 1/2$, $x_2 = -1/2$, $x_3 = i/2$, $x_4 = -i/2$;

therefore, the root factor form only exists in $\mathbb{C}$ (and it is unique, up to the order of the factors):

$$p = 16(x - 1/2)(x + 1/2)(x - i/2)(x + i/2).$$

Example. In $\mathbb{Z}_6[x]$, which is not an integral domain, $p = x^2 + x$ has four roots (see above), and indeed, the root factor form is not unique: $x^2 + x = (x - \bar{2})(x - \bar{3}) = x(x - \bar{5})$.

In the example before the last one, the set of complex numbers ($\mathbb{C}$) played a special role. The following theorem says that this is *always* the case:

Theorem 4.8 (Fundamental theorem of algebra). *Every polynomial $p \in \mathbb{C}[x] \setminus \{0\}$ of degree n has exactly n roots in $\mathbb{C}$, when counted with multiplicities.*

This theorem guarantees that *any* (nonzero) polynomial with complex coefficients has a unique root factor form. (But calculating this form, i.e. finding the roots is a very difficult task.)

Warning: don't confuse the fundamental theorem of **algebra** with the fundamental theorem of **arithmetic** – the latter is about the unique factorization of integers, see: Divisors, prime numbers.

In Sage, the method `p.roots()` calculates the roots of a polynomial:

```
sage: P.<x> = ZZ[]
sage: p = 2*x^3 - 6*x + 4
sage: p.roots()
[(-2, 1), (1, 2)]      # -2 is a simple, 1 is a double root
sage: p.roots(multiplicities=False)
[-2, 1]                # roots without multiplicities
sage: p(-2)             # evaluate at the root
0
sage: p = 16*x^4 - 1
sage: p.roots()
[]                      # no integer roots
sage: p.roots(QQ)        # but we can specify a larger set
[(1/2, 1), (-1/2, 1)]
sage: p.roots(QQ, multiplicities=False)
[1/2, -1/2]
sage: p.roots(RR)        # in RR, Sage can only calculate numerically
[(-0.5000000000000000, 1), (0.5000000000000000, 1)]
sage: p.roots(CC)        # so in CC
[(-0.5000000000, 1), (0.5000000000, 1), (-0.5000000000*I, 1), (0.5000000000*I, 1)]
```

5 Lagrange interpolation

In this section, the polynomials are over a *field* (e.g. $\mathbb{Q}$ or $\mathbb{R}$, but not $\mathbb{Z}$, see: Algebraic structures.)

We have seen that a polynomial of degree n can have at most n roots. This also means that if a polynomial of degree *at most* n has $n+1$ roots, then it can only be the zero polynomial (0) (which has arbitrarily many roots). This statement can be generalized from roots to other values:

Theorem 5.1. *If two polynomials of degree $\leq n$ are equal in $n+1$ places, then they are equal.*

Example. If p and q have degree ≤ 2 , and they are equal in three places: $p(1) = q(1)$, $p(2) = q(2)$ and $p(5) = q(5)$, then the theorem says that $p = q$. For degree ≤ 1 , two values are sufficient (visually: two points define a line), and for constant polynomials, one is enough.

(The proof of the theorem is simple: subtract the two polynomials, then the result has zeros on these $n+1$ places, i.e. roots, but then it can only be the zero polynomial.)

So a polynomial of degree n is uniquely determined by $n+1$ values.

But how can we calculate this unique polynomial from the $n+1$ values? First, consider a special case where all values are zero except one. For example, find the quadratic polynomial p for which $p(1) = 0$, $p(2) = 0$ and $p(5) = 8$. The first two conditions can be handled easily: $p := (x-1)(x-2)$. But the third one is not yet satisfied: $p(5) = (5-1)(5-2) = 12 \neq 8$. So we multiply this polynomial by a constant c to make it true, i.e. $c \cdot p(5) = 8$, and the solution of this is $c = 8/p(5) = 8/12 = 2/3$. Note that this transformation not only fixes the third value, but it also preserves the zeros. So let p be actually $p := 2/3 \cdot (x-1)(x-2) = 2x^2/3 - 2x + 4/3$.

But what if we don't have zero values? For example, which quadratic polynomial p has $p(1) = 4$, $p(2) = 2$ and $p(5) = 8$? For this, we use the above method three times, and combine the results:

$$\begin{aligned} p_1(1) = 4, \quad p_1(2) = 0, \quad p_1(5) = 0 &\implies p_1 = 4 \cdot \frac{(x-2)(x-5)}{(1-2)(1-5)} = x^2 - 7x + 10 \\ p_2(1) = 0, \quad p_2(2) = 2, \quad p_2(5) = 0 &\implies p_2 = 2 \cdot \frac{(x-1)(x-5)}{(2-1)(2-5)} = -\frac{2}{3}x^2 + 4x - \frac{10}{3} \\ p_3(1) = 0, \quad p_3(2) = 0, \quad p_3(5) = 8 &\implies p_3 = 8 \cdot \frac{(x-1)(x-2)}{(5-1)(5-2)} = \frac{2}{3}x^2 - 2x + \frac{4}{3} \end{aligned}$$

$$p(1) = 4, \quad p(2) = 2, \quad p(5) = 8 \implies p = p_1 + p_2 + p_3 = x^2 - 5x + 8$$

This method is called **Lagrange interpolation**. In general:

Theorem 5.2 (Lagrange interpolation). *If a polynomial p of degree $\leq n$ over a field takes the following values on given $n+1$ different places: $p(x_1) = y_1$, $p(x_2) = y_2$, $\dots$, $p(x_{n+1}) = y_{n+1}$, then:*

$$p = \sum_{i=1}^{n+1} y_i \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{x - x_j}{x_i - x_j}.$$

Note: we need a field, otherwise the divisions may not be possible. That is, the interpolation doesn't work in $\mathbb{Z}[x]$, only in $\mathbb{Q}[x]$, $\mathbb{R}[x]$ etc. But for a field, this polynomial uniquely exists for *any* given $x_1, \dots, x_{n+1}$ and $y_1, \dots, y_{n+1}$ values (provided that all x_i 's are different).

Sage has a built-in function for Lagrange interpolation:

```
sage: P.<x> = QQ[]      # not ZZ[]!
sage: P.lagrange_polynomial([(1,4), (2,2), (5,8)])
x^2 - 5*x + 8
```

6 Shamir's secret sharing (SSS)

Assume that we have a secret piece of information (a number, e.g. an encryption key) that we need to share among multiple people (e.g. among the employees of a company).

The simplest way to do this is to give the secret to all participants. But then the obvious problem is that if any of them misuses it, or it is stolen from them, then the whole secret is compromised. We need a solution where no single person can access the secret.

The opposite approach is to split the secret into equal parts, and give each participant a single part. Then none of them can individually misuse their own part – not even a small group of them can do that –, so the secret can only be restored when all participants are involved. But the problem with this is that if any of them loses their part (or the person is temporarily unavailable), then the common secret cannot be restored.

But there is also another problem with this approach: although none of the participants have the *whole* secret, they still have a part of it, which brings them one step closer to finding out the secret using brute force. For example, if a number lock has a four-digit code, then someone who doesn't know anything about the code needs to try $10^4 = 10000$ possibilities to open it (in the worst case). But if someone knows one digit of the code, then they only need to try $10^3 = 1000$ possibilities, which is ten times faster. With two digits, it is hundred times faster.

The ideal solution would need k participants out of the n to restore the secret (e.g. three out of five), meaning that any k participants would be able to calculate the secret, but $k-1$ or less people would not, nor could they even get one step closer to finding it out than an outsider with no information at all.

A solution to this problem is **Shamir's secret sharing** (SSS).

Let n be the number of participants, and k the threshold to restore the secret ($1 \leq k \leq n$). Let $S \in \mathbb{Z}$ be the secret to be shared, and let's "hide" it inside a polynomial of degree $k-1$ as follows:

$$p = a_{k-1}x^{k-1} + \dots + a_2x^2 + a_1x + S,$$

where the constant term is the secret S , and the other coefficients $(a_1, \dots, a_{k-1})$ are integers chosen randomly. This polynomial is then evaluated in n different places, e.g. $p(1), p(2), \dots, p(n)$, and each participant is given one of these values (a pair (x_i, y_i)), called a *secret share*. Then any k participants can use their k secret shares to reconstruct the original polynomial using Lagrange interpolation (see above), and thus get its constant term, S . But no group of less than k people can restore the polynomial, because they can fill the missing values in any way, getting different valid-looking polynomials each time, and they have no way of knowing which one was the real one.

Example. Assume that there are $n = 5$ participants, and the threshold is $k = 3$. Let the secret value be $S = 67$, and generate two more random coefficients, e.g. $a_1 = 38$ and $a_2 = 13$, to construct the quadratic polynomial $p = 13x^2 + 38x + 67$. Then evaluate p in five places:

$$p(1) = 118, \quad p(2) = 195, \quad p(3) = 298, \quad p(4) = 427, \quad p(5) = 582,$$

and give each participant one of these values. Then any three of them can restore S , e.g. the 1st, 2nd and 5th can do this as follows (using Lagrange interpolation, see above):

$$p = 118 \cdot \frac{(x-2)(x-5)}{(1-2)(1-5)} + 195 \cdot \frac{(x-1)(x-5)}{(2-1)(2-5)} + 582 \cdot \frac{(x-1)(x-2)}{(5-1)(5-2)} = 13x^2 + 38x + 67.$$

But we don't actually need to calculate the whole polynomial, only its constant term, S , which is the same as $p(0)$ (i.e. the evaluation of p at $x = 0$), and it can be calculated much faster:

$$S = p(0) = 118 \cdot \frac{(-2)(-5)}{(1-2)(1-5)} + 195 \cdot \frac{(-1)(-5)}{(2-1)(2-5)} + 582 \cdot \frac{(-1)(-2)}{(5-1)(5-2)} = 67.$$

But the method as described above still has a problem. This comes from the fact the we used integer coefficients, whereas we have seen above that the Lagrange interpolation requires a field.

This is not just a theoretical problem, but it actually allows an attack on the scheme! If two participants cooperate, then although they can't recover the whole secret (which needs three people), but as it happens, they can still figure out some properties of it. For example, if they know $p(1)$ and $p(5)$, then they can use the fact that the polynomial is quadratic with integer coefficients, i.e. it can be written as $p = ax^2 + bx + S$ with $a, b \in \mathbb{Z}$, to calculate the following:

$$\begin{aligned} p(5) &= 25a + 5b + S = 582 \\ p(1) &= a + b + S = 118 \\ p(5) - p(1) &= 24a + 4b = 464 \implies b = 116 - 6a \\ \implies p(1) &= a + (116 - 6a) + S = 118 \implies S = 5a + 2. \end{aligned}$$

That is, they learn that S gives the remainder 2 when divided by 5, i.e. $S \equiv 2 \pmod{5}$, which means that if they want to find S by brute force, they only need to check one fifth of the possibilities.

This trick was enabled by the fact that the coefficients are integers, where division is not always possible. If they were rational numbers, then the above equation $S = 5a + 2$ would give no information about S at all, because for any S , there would be an $a \in \mathbb{Q}$ satisfying this equation (namely: $a = (S - 2)/5$).

Then why don't we use rational numbers for the coefficients? For many reasons: it is inconvenient to calculate with fractions, the secret value is already typically an integer, and random number generation would also be problematic (e.g. what distribution do we use at all?).

In practice, Shamir's secret sharing is done in $\mathbb{Z}_p[x]$, where p is a large enough prime number to contain the secret S . (We have seen in the previous document that $\mathbb{Z}_p$ is a field for prime p .)

Example. Let $n = 5$, $k = 3$ and $S = 67$ as above, and generate again a quadratic polynomial, but this time in $\mathbb{Z}_{89}[x]$ instead of $\mathbb{Z}[x]$:

$$p = \overline{13}x^2 + \overline{38}x + \overline{67} \in \mathbb{Z}_{89}[x].$$

It is then evaluated in five places (giving the same values as above, but modulo 89):

$$p(\overline{1}) = \overline{29}, \quad p(\overline{2}) = \overline{17}, \quad p(\overline{3}) = \overline{31}, \quad p(\overline{4}) = \overline{71}, \quad p(\overline{5}) = \overline{48}.$$

The secret can be recovered in the same way, but now the calculations are in $\mathbb{Z}_{89}$ instead of $\mathbb{Q}$. For example, from $p(1)$, $p(2)$ and $p(5)$, it can be done as follows:

$$\begin{aligned} S &= \overline{29} \cdot \frac{(-\overline{2})(-\overline{5})}{(\overline{1} - \overline{2})(\overline{1} - \overline{5})} + \overline{17} \cdot \frac{(-\overline{1})(-\overline{5})}{(\overline{2} - \overline{1})(\overline{2} - \overline{5})} + \overline{48} \cdot \frac{(-\overline{1})(-\overline{2})}{(\overline{5} - \overline{1})(\overline{5} - \overline{2})} = \\ &= \overline{29} \cdot \overline{10} \cdot \overline{4}^{-1} + \overline{17} \cdot \overline{5} \cdot \overline{86}^{-1} + \overline{48} \cdot \overline{2} \cdot \overline{12}^{-1} = \overline{28} + \overline{31} + \overline{8} = \overline{67}. \end{aligned}$$

Since $\mathbb{Z}_p$ is a field, the attack described above is no longer possible: although we still get that $S = \overline{5}a + \overline{2}$, but now this leaks no information about S at all, because the linear congruence $5a \equiv S - 2 \pmod{89}$ is solvable for any S (see: Congruences). Another advantage of $\mathbb{Z}_p$ is that being finite, it is trivial to choose random numbers from it.

7 Author

This document was written by Uray M. János, partly using the following sources:

- [1] <https://compalg.elte.gitlab-pages.hu/dimoa-web/tematika/dimoa> (Hungarian)
- [2] <https://en.wikipedia.org/wiki/Polynomial>
- [3] https://en.wikipedia.org/wiki/Lagrange_polynomial
- [4] https://en.wikipedia.org/wiki/Shamir%27s_secret_sharing
- [5] A. Shamir (1979): How to share a secret. *Communications of the ACM*, 22 (11), pp. 612–613.
<https://dl.acm.org/doi/10.1145/359168.359176>

Errors and suggestions can be reported here: uray.janos@inf.elte.hu.