Sorbonne University

Fondement de l’algorithmique algébrique

MU4IN902

Instructors: Prof. Vincent Neiger

Hayato Ishida

Ver.1.0

Contents

1 Introduction

1.1 Ring and Field

Rough definition : Field is a set F with two operations $(+, \times)$ with "usual" properties (e.g. $a(b+c) = ab + ac$, $a(bc) = (ab)c$ etc...). And every element has an inverse. If one of the elements does not have an inverse, it is a ring.

- So, elements in the field can be inverted except 0.
- Every element $a \in F$ has an opposite for $+$, these act on b s.t. $a + b = 0$. This is true for all rings and fields.
- Therefore, invertibility is about "×"

Remark 1.1

Ring $\longleftarrow$ Field

Examples

- $\mathbb{R}$ the set of real numbers (field). $\mathbb{Q}$ rational (field). $\mathbb{C}$ complex (field).
- $\mathbb{Z}$ integers is not a field, thus a ring. Why?

This is because 2 is not invertible. For instance, there is no $x \in \mathbb{Z}$ s.t. $2x = 1$.

- $\mathbb{Z}/p\mathbb{Z}$ is a field. In general, $a \in \{1, ..., p-1\}$, there are u, v s.t. $au + pv = 1$, so $au = 1 \; mod \; p$.
- $\mathbb{Z}/4\mathbb{Z}$ is not a field, why? $\longrightarrow$ because 2 is not invertible.

1.2 Complexity

In this course, we manipulate algebraic objects. For instance, polynomials (ring), matrices (field), and more. Complexity measures the efficiency of algorithms. Algebraic complexity is counting the number of operations in the base ring and field.

The complexity here does not take into account of ...

- all memory-related operations

- the size of the elements in the base ring or field. (e.g. : addition of polynomials of degree $\leq d$ in $\mathbb{Q}[x]$, algebraic complexity is $O(d)$ and ignore the size of the rational coefficients.

*You should be familiar with the complexity from MODEL.

1.3 Matrices

For a field $\mathbb{K}$ or denote by $\mathbb{K}^{m \times n}$ the set of $m \times n$ matrices over $\mathbb{K}$.

Recall : Not a ring because .... ??

It is represented as a two-dimensional array with rows and columns. $A = (a_{i\,j}) \in \mathbb{K}^{m \times n}$ where $0 \leq i \leq m$ and $0 \leq j \leq n$.

Addition

Complexity is $O(m\,n)$, operations in $\mathbb{K}$.

Multiplication

Multiplication of $A \in \mathbb{K}^{m\times n}$ by $B \in \mathbb{K}^{n\times p}$

- Naive algorithm : $O(m\,n\,p)$

- Strassen's algorithm (recall form MODEL) has complexity $O(n^{log_2(7)})$ where $m = n = p$

- As an alternative, there is a lot of work on matrix multiplication algorithms. The best performance today is $O(n^{2.37})$ but for the moment it is impractical.

1.4 Polynomials

For a field $\mathbb{K}$ or a ring $\mathbb{R}$, the sets $\mathbb{K}[x]$, $\mathbb{R}[x]$ are those of polynomials in one variable x.

$\mathbb{K}[x] = \{P = P_0 + P_1 x + P_2 x^2 + \ldots.. + P_d x^d\}$ in $\mathbb{K}$. (same for $\mathbb{R}$)

They are represented as a one-dimensional array $[P_0, P_1, P_2, \ldots.., P_d]$.

Addition

of two polynomials in $\mathbb{R}[x]$ of degree $\leq d$ has a complexity $O(d)$ operations in $\mathbb{R}$.

Multiplication :

- Naive algorithm : $O(d^2)$ operations in $\mathbb{R}$ - Karatsuba (recall from MODEL) : $O(d^{log_2(3)}) \approx O(d^{1.584})$ operations in $\mathbb{R}$

- For alternative FFT (again, recall from MODEL) : $O(d\ log(d)\ log\ log(d))$

2 Euclid's division and GCD

- $A|B$ means A divides B. Meaning there is no reminder (and A and B are polynomials here)

2.1 Algebraic structure

Definition 2.1

$\mathbb{K}$ field

$f \in \mathbb{K}[x] \Rightarrow f = a_0 + a_1 x + ... + a_n x^n$ with $a_i \in \mathbb{K}$

$n = deg(F)$

- Roots : $\alpha \in \mathbb{K} s.t f(\alpha) = 0$

Lemma 2.0

α is a root of $f \iff (x - \alpha)|f$, in other word $f \in \mathbb{K}[x]$ and $\alpha \in \mathbb{K} s.t\ f(\alpha) = 0$, the $(X - a)|f$

Lemma 2.0

f has degree n, then f has at most n roots.

Definition 2.1: Algebraic closure

- $\overline{\mathbb{K}}$ is the algebraic closure of K if :

- $\mathbb{K} \subset \overline{\mathbb{K}}$

- $\forall f \in \bar{\mathbb{k}}\,[x]$, f has exactly $degf$ roots in $\bar{\mathbb{k}}$.

Example :

$\mathbb{C}$ is algebraic closed.

$\mathbb{R} \subset \mathbb{C}$ and $\mathbb{Q} \subset \mathbb{C}$ are algebraic closure.

But, $\mathbb{R}$ is not $\longrightarrow x^2 + 1$ has 0 roots in $\mathbb{R}$.

Definition 2.2: Recalling a definition of ring

Let R a set equipped with two binary operation s.t $(R, +, \times)$. It is said to be a ring if the followings hold $\longrightarrow$

- $R, +)$ commutative group with neutral 0_R (what is 0_R means ???) - $\times$ has a neutral element 1_R and is associative

$a \times b \times c = (a \times b) \times c = a \times (b \times c)$

- $\times$ is distributive with respect to $+$.

$a(b + c) = ab + ac$

Example :

$\mathbb{Z}, \mathbb{K}[x], \mathbb{R}, \mathbb{C}$, but $\mathbb{N}$ is not a ring because $-1 \in \mathbb{N}$.

Proposition 2.1

$f = a_0 + ... + a_n x^n$ with $a_i \in \mathbb{R}$, $\mathbb{R}[x]$

if R is a ring so is $R[x]$

Definition 2.3: Monic Polynomial

Monic polynomial is a polynomial with a leading coefficient equal to 1.

Example

- Monic polynomial : $x + 1$, $x^2 - 7x + 99$, $x^{1000} + x^{99} - 10000$

- Polynomial that are not monic : $5x^{99} + 4$, $8x^3 + x^2 - 7x + 0$

Definition 2.4: Quotient ring

Let R be a ring and $f \in R[x]$ be monic. Also, $s, t \in R[x]$.

- We say that s and t are equivalent modulo $f \iff s \equiv t \ \ mod\ f$, if f divides $s - t$.

- The set of classes of equivalences is denoted by $R[x]/\langle f \rangle$ and is called quotient ring.

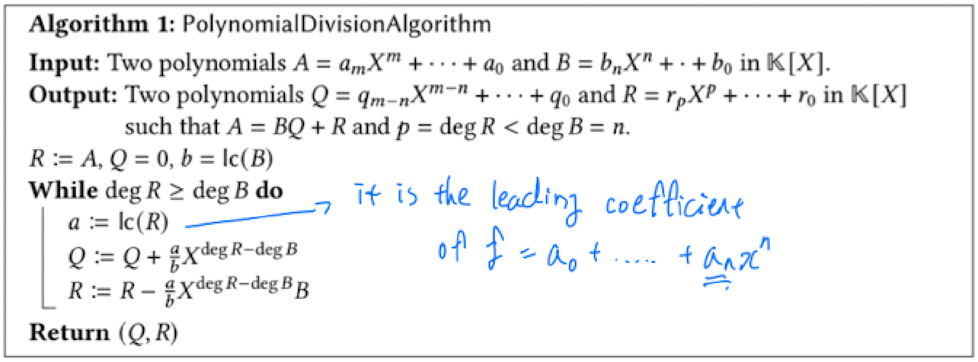
Example :

$x^2 + x + 1 \equiv 1 \ \ mod \ \ x + 1$, since $x^2 + x + 1 = x(x + 1) + 1$

2.2 Division algorithm for polynomials in $\mathbb{K}[x]$

Definition 2.5: Euclidean domain

$\mathbb{R}$ is a Euclidean domain if there exists a Euclidean division (and R is an integral domain).

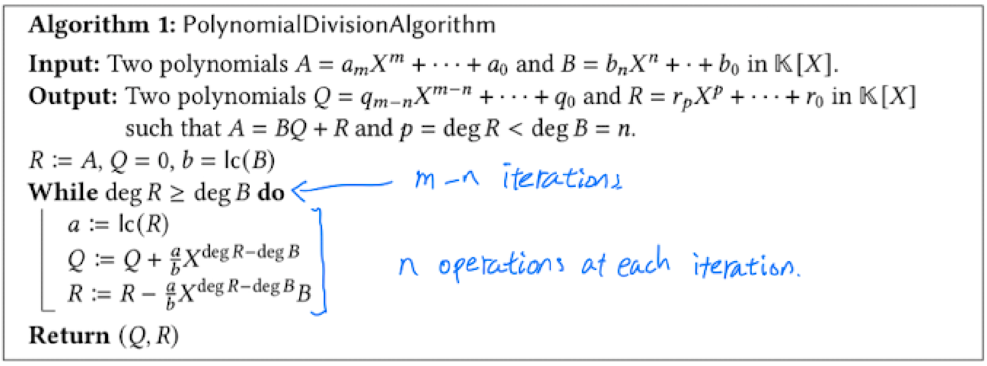


```
Algorithm 1: PolynomialDivisionAlgorithm
Input: Two polynomials $A = a_mX^m + \cdots + a_0$ and $B = b_nX^n + \cdot + b_0$ in $\mathbb{K}[X]$.
Output: Two polynomials $Q = q_{m-n}X^{m-n} + \cdots + q_0$ and $R = r_pX^p + \cdots + r_0$ in $\mathbb{K}[X]$
        such that $A = BQ + R$ and $p = \deg R < \deg B = n$.
$R := A$, $Q = 0$, $b = \mathrm{lc}(B)$
While $\deg R \geq \deg B$ do
    $a := \mathrm{lc}(R)$
    $Q := Q + \frac{a}{b}X^{\deg R - \deg B}$
    $R := R - \frac{a}{b}X^{\deg R - \deg B}B$
Return $(Q, R)$
```

it is the leading coefficient of $f = a_0 + \dots + \underline{a_n} x^n$

Proposition 2.2

On input A and B in $\mathbb{K}[x]$ with degree m and n, with $m > n$. Polynomial division algorithm perform $O(n(m-n))$ arithmetic operation in $\mathbb{K}$.



```
Algorithm 1: PolynomialDivisionAlgorithm
Input: Two polynomials $A = a_mX^m + \cdots + a_0$ and $B = b_nX^n + \cdot + b_0$ in $\mathbb{K}[X]$.
Output: Two polynomials $Q = q_{m-n}X^{m-n} + \cdots + q_0$ and $R = r_pX^p + \cdots + r_0$ in $\mathbb{K}[X]$
        such that $A = BQ + R$ and $p = \deg R < \deg B = n$.
$R := A$, $Q = 0$, $b = \mathrm{lc}(B)$
While $\deg R \geq \deg B$ do
    $a := \mathrm{lc}(R)$
    $Q := Q + \frac{a}{b}X^{\deg R - \deg B}$
    $R := R - \frac{a}{b}X^{\deg R - \deg B}B$
Return $(Q, R)$
```

$m - n$ iterations

n operations at each iteration.

Remark 2.1

(1) If $\mathbb{K}$ is just a ring, the algorithm works if and only if B is monic.

(2) $A \equiv R \ \ mod \ \ B$ $\longrightarrow$ Euclidean division allows to perform operations in $\mathbb{K}[x]/(B)$.

$A_1 + A_2 \equiv R_1 + R_2 \ \ mod \ \ B$

$A_1 \times A_2 \equiv R_1 \times R_2 \ \ mod \ \ B$

2.3 Euclidean algorithm

Definition 2.6

R Euclidean domain and $a, b \in \mathbb{R}$, g is a gcd of a and b : $g = gcd(a, b)$

if : $g \in R$

$g|a$

$g|b$

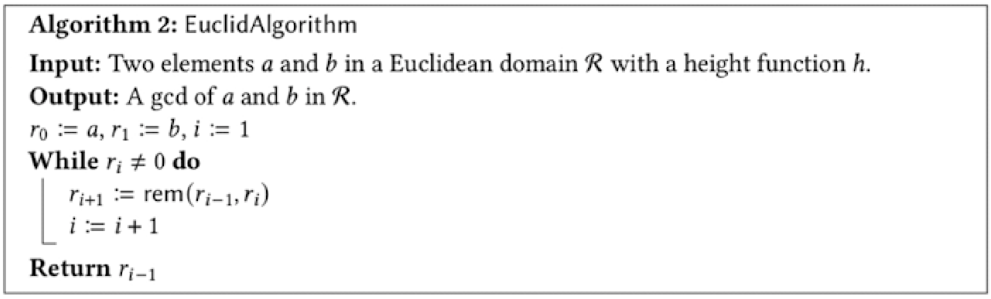
any common divisor of $aandb$ divides g.

Proposition 2.3

In R Euclidean, such a gcd always exist.

Remark : g may not be unique.

Proposition : If $a = bq + r$ with $h(r) < h(b)$ then, $gcd(a, b) = gcd(b, r)$.



```
Algorithm 2: EuclidAlgorithm
Input: Two elements $a$ and $b$ in a Euclidean domain $\mathcal{R}$ with a height function $h$.
Output: A gcd of $a$ and $b$ in $\mathcal{R}$.
$r_0 := a, r_1 := b, i := 1$
While $r_i \neq 0$ do
    $r_{i+1} := \mathrm{rem}(r_{i-1}, r_i)$
    $i := i + 1$
Return $r_{i-1}$
```

Complexity : $O(deg(a) \times deg(b))$

Proposition 2.4

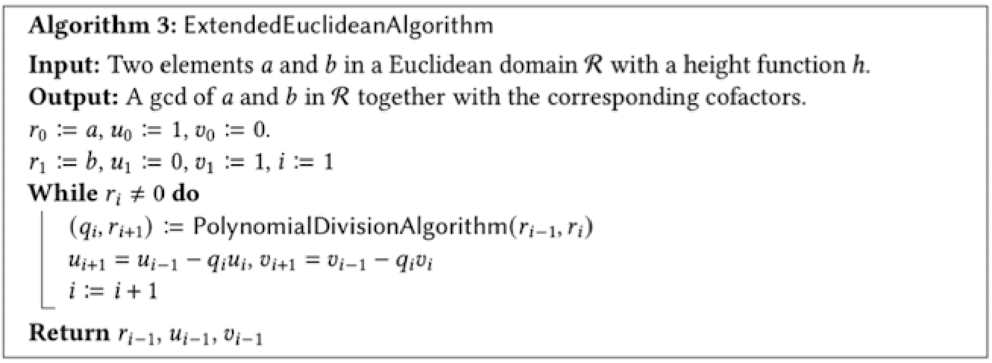
If $g = gcd(a, b)$ then, $\exists (u, v) \in R^2$ s.t $au + bv = g$ and $h(ug) < h(b), h(vg) < h(a)$

- u and v are cofactors.

Proposition 2.5

$a, b \in R$

EEA



```
Algorithm 3: ExtendedEuclideanAlgorithm
Input: Two elements $a$ and $b$ in a Euclidean domain $\mathcal{R}$ with a height function $h$.
Output: A gcd of $a$ and $b$ in $\mathcal{R}$ together with the corresponding cofactors.
$r_0 := a, u_0 := 1, v_0 := 0.$
$r_1 := b, u_1 := 0, v_1 := 1, i := 1$
While $r_i \neq 0$ do
    $(q_i, r_{i+1}) := \mathrm{PolynomialDivisionAlgorithm}(r_{i-1}, r_i)$
    $u_{i+1} = u_{i-1} - q_i u_i, v_{i+1} = v_{i-1} - q_i v_i$
    $i := i + 1$
Return $r_{i-1}, u_{i-1}, v_{i-1}$
```

Complexity : $O(deg(a) \times deg(b))$

Application of Extended Euclidean Algorithm: Modulo inversion

If a and b are coprimers, then $\exists (u, v) / au + bv = 1$.

So, $au \equiv 1 \mod b$ $bv \equiv 1 \mod a$

- u is a inverse of $a \mod b$

- v is a inverse of $b \mod a$

If $a \in \mathbb{Z}$ and $au + bv = 1$, then $a^{-1} \equiv u \mod n$

$\bar{a} = a + kn, K \in \mathbb{Z} \in \mathbb{Z}/n\mathbb{Z}, \bar{a}^{-1} = \bar{u} = u + kn, k \in \mathbb{Z}$

- If n is prime number then for all $a \in \mathbb{Z}$, $gcd(a, n) = 1$. So for all $\bar{a} \in \mathbb{Z}/n\mathbb{Z}$, $\bar{a}^{-1}$ exists.

Proposition 2.6

$\mathbb{Z}/n\mathbb{Z}$ is a field if and only if n is prime.

Definition 2.7

$P \in \mathbb{K}[x]$ is irreducible if for any $\mathbb{Q}, \mathbb{R} \in \mathbb{K}[x]$ s.t $P = QR$, then either $\mathbb{Q} \in \mathbb{K}$ or $\mathbb{R} \in \mathbb{K}$.

Proposition 2.7

If P is irreducible then $\forall\, \mathbb{Q} \in \mathbb{K}[x], \quad gcd(P, Q) = 1$

Theorem 2.3

$\mathbb{K}[x]/(P)$ is a field if and only if P is irreducible

Remark 2.1

we can computer the inverse with Extended Euclidean Algorithm.

3 Finite field

Definition 3.1

A finite field is a field with a finite number of elements.

3.1 Integer $\mathbb{Z}$

- for $n \in \mathbb{Z}/0 \approx 0, ..., n-1$ with add/ multiplication modulo n
- $a \in \mathbb{Z}/n\mathbb{Z}$ is invertible if and only if $gcd(a,n) = 1$
- n is prime $\iff$ $\mathbb{Z}/n\mathbb{Z}$ is a field
- computing a^{-1} : run EEA to obtain $1 = au + nv$

Theorem 3.1: Bezout's relation

Let $R = \mathbb{Z} or R = \mathbb{K}[x]$. If a and b in R, there exist u and v in R s.t $au + bv = gcd(a,b)$

Theorem 3.2

Let $R = \mathbb{Z} or R = \mathbb{K}[x]$. If a and b are coprime, then a invertible modulo b and b invertible modulo a. Thus, $\mathbb{Z}/n\mathbb{Z}$ is a field, if and only if n is a prime.

3.2 Polynomial $\mathbb{K}[x]$

(where $\mathbb{K}$ is a field) - for $f \in \mathbb{K}[x] \setminus \{0\}$, $\mathbb{K}[x]/\langle f \rangle \approx \{P(X) \in \mathbb{K}[x]/deg(p) < deg(f)\}$ with add/multiplication mod f.

- $P \in \mathbb{K}[x]/\langle f \rangle$: P is invertible if and only if $gcd(p,f) = 1$

-f is invertible $\iff$ $\mathbb{K}[x]/\langle f \rangle$ field.

- computing P^{-1}: run EEA to obtain $1 = pu + fv$

- $\mathbb{K}[x]/\langle f \rangle$ is a field, for irreducible polynomial f.

Proof:

Suppose f irreducible, let $P \in \mathbb{K}[x]/\langle f \rangle \setminus \{0\}$. To show that P is invertible which means $gcd(P, f) = 1$. The gcd of P and f divides both P and f. But f has only $\mathbb{K}\{0\}$ and $\propto f, \propto \in \mathbb{K} \setminus \{0\}$ as divides. Since $deg(P) < deg(f)$, P cannot be divisible by f, so $gcd(P, f) = 1$.

3.3 Building a finite field

- If $K = \mathbb{Z}/p\mathbb{Z}$ and $deg(f) = d$ then $\mathbb{Z}/p\mathbb{Z}[x]/\langle f \rangle$ is a finite field of cardinality p^d. This is because $\mathbb{Z}/p\mathbb{Z}[x]/\langle f \rangle = \{a_0 + a_1 x + \ldots + a_{d-1} x^{d-1}, (a_0, a_1 \ldots.., a_{d-1}) \in (\mathbb{Z}/p\mathbb{Z})^d\}$ and cardinality of $(\mathbb{Z}/p\mathbb{Z}))^d$ is p^d

Theorem 3.3

A finite field must have p elements for same prime p and $d \in \{1, 2, \ldots..\}$. If $d = 1$ then finite field is $\mathbb{Z}/p\mathbb{Z}$. If $d > 1$, then $\mathbb{Z}/p\mathbb{Z}$ is not a field.

- $\mathbb{F}_q$ for $q = p^d$ is the notation for a finite field of cardinality q

III.6.

4 Formal Power Series

4.1 Introduction

Definition 4.1

From a sequence $(s_i) \in \mathbb{K}^{\mathbb{N}}$. We define the power series:

$$\sum_{i \in \mathbb{N}} s_i x^i$$

Proposition 4.1

- The set of power series is a ring which we write $\mathbb{K}[|x|]$

- Power series is an infinite sequence.

Examples:

$1 + x$ is a power series with coefficients $(1, 1, 0, 0, ..., 0, ..)$.

Remark 4.1

Polynomials are power series. $\iff$ (s_i) finitely many nonzero s_i

Operations $(+, \times)$ for power series

$$(1 - x) \sum_{i \in \mathbb{N}} x^i \;=\; 1 \quad \Rightarrow 1 \cdot \sum_{i \in \mathbb{N}} x^i \;-\; x \cdot \sum_{i \in \mathbb{N}} x^i$$

$$= (1, 1, 1, ..., 1, ...) \;-\; (0, 1, , 1, 1, ..., 1, ...) = (1, 0, 0, 0, ..., 0, ...)$$

- Addition: it is coefficient by coefficient.

$$\sum_{i\in\mathbb{N}} s_i x^i + \sum_{i\in\mathbb{N}} t_i x^i = \sum_{i\in\mathbb{N}} (s_i + t_i) x^i$$

- Multiplication :

$$(\sum_{i\in\mathbb{N}} s_i x^i)(\sum_{i\in\mathbb{N}} t_i x^i) = \sum_{i\in\mathbb{N}} (\sum_{k=0}^{i} s_k t_{i-k}) x^i$$

- 0 in $\mathbb{K}[|x|]$ is $(0 + 0x + 0x^2 + 0x^3 + ...)$

- 1 in $\mathbb{K}[|x|]$ is $(1 + 0x + 0x^2 + 0x^3 + ...)$

Remark 4.2

in $(\sum_{i\in\mathbb{N}} s_i x^i)$, x is not invertible. This can be proven by contradiction.

Examples 1 :

Fibonacci sequence ⟶

$f_0 = 0, f_1 = 1, \quad \forall i, f_i = f_{i-1} + f_{i-2}$, then we can form:

$$S = \sum_{i\in\mathbb{N}} f_i x^i \in \mathbb{K}[|x|]$$

In this context with recurrent sequence, S is called the generating series of $(f_i)_{i\in\mathbb{N}}$.

Examples 2 :

Compute

$$(1 - x - x^2)S = \sum_{i\in\mathbb{N}} f_i x^i - \sum_{i\in\mathbb{N}} f_i x^{i+1} - \sum_{i\in\mathbb{N}} f_i x^{i+2}$$

$$= f_0 + f_1 x + \sum_{i\in\mathbb{N}} f_{i+2} x^{i+2} - (f_0 x + \sum_{i+1} x^{i+2}) - \sum_{i\in\mathbb{N}} f_i x^{i+2}$$

$$= f_0 + f_1 x - f_0 x + \sum_{i\in\mathbb{N}} (f_{i+2} - f_{i+1} - f_i) x^{i+2} = x$$

Sometimes, you can write a series in fractions,

$$S = \frac{x}{1-x-x^2}$$

*it doesn’t make sense to evaluate power series with specific value

some terminologies,

non-zero series : $[1, -1, -1, 0, 0, \ldots.., 0, ..]$

zero series: $[0, 0, 0, 0, \ldots.., 0..]$

infinite sequence: $\frac{1}{3} = 0.333333\ldots\ldots$

$\mathbb{K}[x], \quad \{k(x) = \frac{p}{\phi}, \quad \phi \neq 0, \quad p, \phi \in \mathbb{K}[x]\}$

We work with power series:

- As fraction $\frac{p}{\phi}$ of two polynomials
- As a truncated power series at precision n:

$$S = s_0 + s_1 x + s_2 x^2 + \ldots.. + O(x^n)$$

$$\text{where } O(x^n) : x^n T \text{ for some } T \in \mathbb{K}[|x|]$$

4.2 Inversion

A power series S is invertible if there exists $T \in \mathbb{K}[|x|]$ such that $ST = 1$

Examples:

- $S = 0$ is not invertible
- $S = c \in \mathbb{K} \setminus \{0\} : S^{-1} = c^{-1}$
- $S = 1 - x - x^2$: invertible....why?
- $S = x$: not invertible....why?
- $S = 1 - x \quad : \quad (1-x)^{-1} = \sum_{i \in \mathbb{N}} x^i$

Lemma 4.0

$S = \sum_{i \in \mathbb{N}} s_i x^i$ is invertible if and only if $S_0 \neq 0$

proof: Assume $S_0 \neq 0$. Construct $U = \sum_{i \in \mathbb{N}} u_i x^i \quad$ s.t. $\quad US = 1$

Coefficient of degree $0 : 1 = u_0 S_0 \longrightarrow u_0 = S_0^{-1}$

Coefficient of degree $1 : 0 = u_0 S_1 + u_1 S_0 \longrightarrow u_1 = \frac{-u_0 S_1}{S_0}$

Coefficient of degree $2 : 0 = u_0 S_2 + u_1 S_1 + u_2 S_0$

Coefficient of degree $3 : 0 = u_0 S_3 + u_1 S_2 + u_2 S_1 + u_3 S_0$

.... continued

proceeding this way we get $u_2, u_3, ...$ defined uniquely.

From S at precision n, this gives $U = S^{-1}$ at precision n

Therefore, we can invert $S \in \mathbb{K}[|x|]$ known at precision n (with inverse at precision n) using $O(n^2)$ operations in $\mathbb{K}$.

Lemma 4.0

Let $S \in \mathbb{K}[[x]]$ be an invertible power series, and let $T = S^{-1} + O(x^n)$. Then the power series $U = T + (1 - TS)T$ satisfies $U = S^{-1} + O(x^{2n})$

Remark 4.1

For a differentiable function F, Newton's iteration for an approximated root x_k of $F(x) = 0$ is (ANUM!):

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)}$$

In particular, for the case of power series inversion, power series S is the root of the function $F(x) = \frac{1}{x} - S$ so:

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)} = x_k + (1 - x_k S)x_k$$

Also, notice that $U = S^{-1} + O(x^{2n})$ is the Newton's iteration applied to T

```
**Algorithm 5:** Power Series Inversion via Newton iteration
**Input:** An integer $n > 0$, and a truncated series $S = s_0 + \cdots + s_{n-1}x^{n-1} + O(x^n) \in \mathbb{K}[[x]]$
        at precision $n$.
**Output:** The truncated power series $U$ at precision $n$ which satisfies $U = S^{-1} + O(x^n)$.
**If** $n = 1$ **then Return** $s_0^{-1}$
Compute recursively the inverse $T$ of $S + O(x^{\lceil n/2 \rceil})$.
**Return** $U := T + (1 - TS)T + O(x^n)$.
```

Complexity analysis:

$f(n)$ = complexity of input precision n

$f(n) = 1 \quad \text{if} \quad n = 1$ and

$f(n) = f(\lceil \frac{n}{2} \rceil) + 2M(n) + 2n$

Roughly..... $f(n) = 2(M(n) + n) + 2(M(\frac{n}{2} + \frac{n}{2})) + 2(M(\frac{n}{4}) + \frac{n}{4}) + ... = O(M(n))$

Example 1 (Example IV 6)

With Newton's iteration, pay attention to $O(x^{\lceil n/2 \rceil}$, ignore all terms degree higher than $\lceil n/2 \rceil$.

Also, $\mathbb{F}_n[[x]]$, which indiecate that all computations are in $\mod n$,

$$S = 3 + 2x^2 + x^3 + x^7 \in \mathbb{F}_5[[x]]$$

$n = 8 \quad (O(x^{\lceil n/2 \rceil})$

$$S = 3 + 2x^2 + x^3 + O(x^4)$$

$n = 4$

$$S = 3 + O(x^2)$$

$n = 1$

$$T = 3^{-1} + O(x) = 2 + O(x)(2 \text{ because } 3^{-1} \mod 5)$$

now algorithm returns.... (don't forget about $\mod 5$)

$$U = T + (1 - TS)T + O(x^2) = 2 + (1 - 2 \times 3)2 + O(x^2)$$

$$= 2 + (-5)2 + O(x^2) = 2 + O(x^2)$$

here, at $O(x^2), n = 4, S = 3 + O(x^2)$. On the next step $O(x^4)$, use S at $O(x^4)$ and previous computation of U.

$$V = U + (1 - US)U + O(x^4) = 2 + (1 - 2(3 + 2x^2 + x^3))2 + O(x^4)$$

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

$$2 + 2x^2 + 6x^3 + O(x^4) = 2 + 2x^2 + x^3 + O(x^4)$$

The last step, using the previous computation of V

$$W = V + (1 - VS)V + O(x^8)$$

$$= 2 + 2x^2 + x^3 + (1 - (2 + 2x^2 + x^3)(3 + 2x^2 + x^3 + x^7))(2 + 2x^2 + x^3) + O(x^8)$$

*it is in precision of $O(x^8)$, thus you don't have to consider terms about x^8

$$= 2 + 2x^2 + x^3 + (1 - 6 - 10x - 5x^3 - 4x^4 - 4x^5 - x^6 - 2x^7)(2 + 2x^2 + x^3) + O(x^8)$$

now you have to remind yourself and consider that it is in $\mod 5$

$$= 2 + 2x^2 + x^3 + (x^4 + x^5 + 4x^6 + 3x^7)(2 + 2x^2 + x^3) + O(x^8)$$

$$= 2 + 2x^2 + x^3 + (2x^4 + 2x^5 + 8x^6 + 6x^7 + 2x^6 + 2x^7 + x^7) + O(x^8)$$

$$= 2 + 2x^2 + x^3 + 2x^4 + 2x^5 + 4x^7 + O(x^8)$$

4.3 Polynomial division with a reminder: fast algo.

The best algorithm known is based on Newton's inversion of power series.

Theorem 4.3

Given (A, B) polynomials of degree $m \geq n \geq 0$, we can compute a Euclidean division

$A = BQ + R, \quad deg(R) < n$ in $O(M(m - n)) + M(n)$ operations in $\mathbb{K}$.

*division cost roughly the same as a multiplication for polynomials.

Idea of fast polynomial division:

- With two polynomials A with degree m and A with degree n, we want to find polynomials Q and R s.t. $A = BQ + R$, with $deg(R) < deg(B) = n$.
- $m \geq n$, otherwise the solution is $(Q, R) = (0, A)$ in other word $A = 0 + A$
- The idea of this algorithm is to exploit the gap between $deg(R)$ and $deg(A) = deg(BQ) = m$, thus gap $= m - deg(R) \geq m - n + 1$.

  We reverse the equality to put the gap in the low-degree coefficients:

  $x^m A(x^{-1}) = x^m B(x^{-1}) Q(x^{-1}) + x^m R(x^{-1})$
- Also, we can rewrite the solution of division as

$$\frac{A}{B} = Q + \frac{R}{B}$$

  Because $deg(R) < deg(B)$, with $x \longrightarrow \infty$, $\frac{R(x)}{B(x)} = 0$

Therefore Q corresponds to the asymptotic expansion of $\frac{A}{B}$ at infinity (∞). Hence, one can obtain Q by computing the Taylor expansion at infinity of the fraction $\frac{A}{B}$.

- To adapt above approach of an expansion at ∞, we can cange variable $y \longleftarrow x^{-1}$, thus:

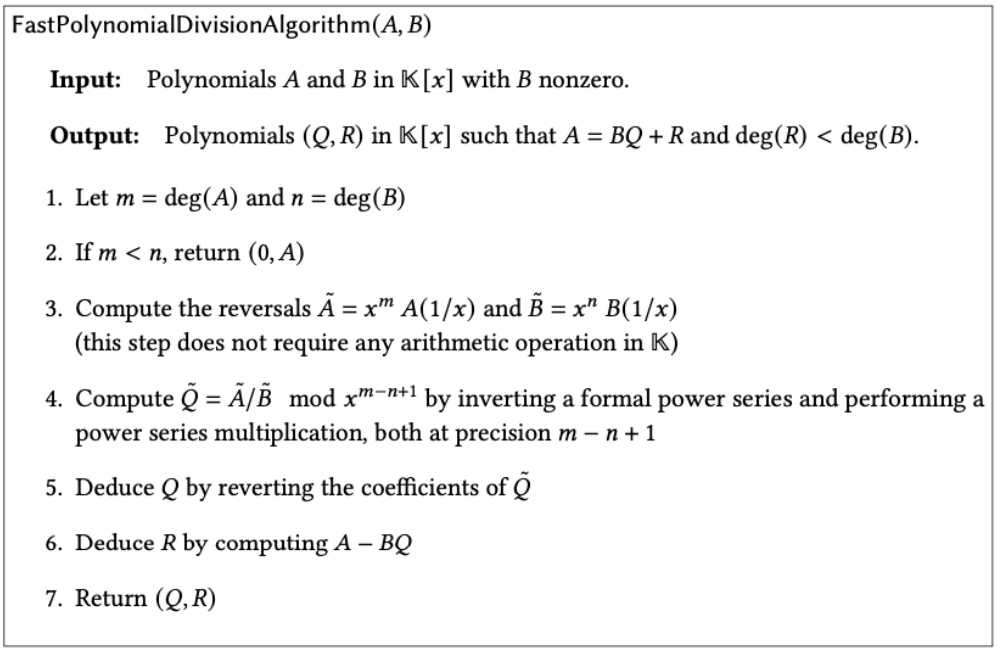
  $$\frac{A(x^{-1})}{B(x^{-1})} = Q(x^{-1}) + \frac{R(x^{-1})}{B(x^{-1})}$$

  Then, multiply each side by x^{m-n} $(\frac{x^m}{x^n})$ to ensure that we only manipulate polynomials in numerators and denominators:

  $$\frac{x^m A(x^{-1})}{x^n B(x^{-1})} = x^{m-n} Q(x^{-1}) + \frac{x^m R(x^{-1})}{x^n B(x^{-1})}$$

  Here, $deg(Q) = m - n$ and $x^n B(x^{-1})$ is invertible as a power series. (By assumption, B is nonzero.)

- Since $deg(R) < n$ and $m \geq n$, the polynomial $x^m R(x^{-1}) = x^{m-n+1} x^{n-1} R(x^{-1})$ has valuation at least $m - n + 1$ which is grater than the degree of polynomial $x^{m-n} Q(x^{-1})$. Thus expansion of $\frac{x^m A(x^{-1})}{x^n B(x^{-1})}$ at precision $m - n + 1$ will give us all coefficients of the polynomial $x^{m-n} Q(x^{-1})$, from which we can deduce to Q. Then $R = A - BQ$



```
FastPolynomialDivisionAlgorithm($A, B$)

    Input:   Polynomials $A$ and $B$ in $\mathbb{K}[x]$ with $B$ nonzero.

    Output:   Polynomials $(Q, R)$ in $\mathbb{K}[x]$ such that $A = BQ + R$ and $\deg(R) < \deg(B)$.

    1. Let $m = \deg(A)$ and $n = \deg(B)$
    2. If $m < n$, return $(0, A)$
    3. Compute the reversals $\tilde{A} = x^m\, A(1/x)$ and $\tilde{B} = x^n\, B(1/x)$
       (this step does not require any arithmetic operation in $\mathbb{K}$)
    4. Compute $\tilde{Q} = \tilde{A}/\tilde{B} \mod x^{m-n+1}$ by inverting a formal power series and performing a
       power series multiplication, both at precision $m - n + 1$
    5. Deduce $Q$ by reverting the coefficients of $\tilde{Q}$
    6. Deduce $R$ by computing $A - BQ$
    7. Return $(Q, R)$
```

$$A(x) = a_0 + a_1 x + 1_2 x^2 + \ldots + a_m x^m$$

$$A(1/x) = a_0 + \frac{a_1}{x} + \frac{a_2}{x^2} + \ldots + \frac{a_m}{x^m} \longrightarrow \tilde{A} = a_m + a_{m-1}x + \ldots + a_0 x^m$$

* multiplication of power series is same as polynomial multiplication

Example (Problem IV 9)

Compute a division of A/B where

$$A = 2 + x^3 + x^9 \quad B = 2 + 2x + x^3 \quad \text{in } \mathbb{F}_3[|x|]$$

Compute reversals $\tilde{A}$ and $\tilde{B}$. It is easier if you write coefficient on a array such that,

$$A = [2, 0, 0, 1, 0, 0, 0, 0, 0, 1] \quad B = [2, 2, 0, 1]$$

and then inverse the order.

$$\tilde{A} = 1 + x^6 + 2x^9 \quad \tilde{B} = 1 + 2x^2 + 2x^3$$

$m = 9, n = 3$, thus precision is $m - n - 1 = 7 \quad O(x^7)$

Compute $\tilde{Q} = \tilde{A}/\tilde{B} = \tilde{A}\tilde{B}^{-1}$. First, compute inverse of $\tilde{B} \longrightarrow \tilde{B}^{-1}$

$$S = 1 + 2x^2 + 2x^3 + O(x^7)$$

$n = 7 \quad$ meaning $\;O(x^{\lceil 7/2 \rceil})$

$$S = 1 + 2x^2 + 2x^3 + O(x^4)$$

$n = 4$

$$S = 1 + O(x^2)$$

$n = 2$

$$S = 1 + O(x)$$

$n = 1$

$$S = 1^{-1} + O(x) = 1 + O(x)$$

Return $\tilde{B}^{-1}$

$$U = T + (1 - ST)T + O(x^2) = 1(1 - 1 \times 1)1 + O(x^2) = 1 + O(x^2)$$

$V = U + (1 - US)U + O(x^4) = 1 + (1 - 1(1 + 2x + 2x^3)1 + O(x^4)) = 1 + (1 - 1 - 2x^2 - 3x^3)1 + O(x^4)$

Reminder, computations are in $\mod 3$!!

$T = V + (1 - VS)V + O(x^7) = 1 + x^2 + x^3 + (1 - (1 + x^2 + x^3)(1 + 2x^2 + 2x^3))(1 + x^2 + x^3) + O(x^7)$

$$= 1+x^2+x^3+(1-(1+2x^2+2x^3+x^2+2x^4+2x^5+x^3+2x^5+2x^6))(1+2x^2+2x^3)+O(x^7)$$

$$= 1 + x^2 + x^3 + (1 - 1 - 3x^2 - 3x^3 - 2x^4 - 4x^5 - 2x^6)(1 + x^2 + x^3) + O(x^7)$$

$$= 1 + x^2 + x^3 + (x^4 + 2x^5 + x^6)(1 + 2x^2 + x^3) + O(x^7)$$

$$= 1 + x^2 + x^3 + x^4 + x^6 + 2x^5 + x^6 + O(x^7)$$

$$= 1 + x^2 + x^3 + x^4 + 2x^5 + 2x^6 + O(x^7)$$

Thus,

$$\tilde{B}^{-1} = 1 + x^2 + x^3 + x^4 + 2x^5 + 2x^6$$

now, compute $\tilde{A}\tilde{B}^{-1}$

$$\tilde{A}\tilde{B}^{-1} = (1 + x^6 + 2x^9)(1 + x^2 + x^3 + x^4 + 2x^5 + 2x^6)$$

$$= 1 + x^2 + x^3 + x^4 + 3x^6 = 1 + x^2 + x^3 + x^4 + 2x^5$$

Write down coefficient in array, $[1, 0, 1, 1, 1, 2]$ then deduce Q by reverting $\tilde{Q} = \tilde{A}\tilde{B}^{-1}$

$$Q = 2x + x^2 + x^3 + x^4 + x^6$$

Next, compute $R = A - BQ$

$$BQ = (x^3+2x+2)(2x+x^2+x^3+x^4+x^6) = 2x^4+x^5+x^6+x^2+2x^3+2x^4+2x^5+x+2x^2+2x^3+2x^4+2x^6+2x^7+x^7+x^9$$

$$= x + x^3 + x^9$$

$$R = (x^9 + x^3 + 2) - (x^9 + x^3 + x) = 2 - x = 2x + 2$$

Final step is necessary because $\mathbb{F}_3[|x|]$. The solution is :

$$(Q, R) = (2x + x^2 + x^3 + x^4 + x^6, 2x + 2)$$

5 Fast Polynomial Evaluation and Interpolation

5.1 Introduction

There are two main questions we want to solve in this chapter.

Question 1: (Multipoint evaluation)

- Input: n elements $x_0, ..., x_{n-1} \in \mathbb{K}$
- Input: Polynoimial $P = p_{n-1}x^{n-1} + \ldots + p_0 \in \mathbb{K}[x]$ of degree less than n.
- How to efficiently compute $y_i = P(x_i)$ for $0 \leq i < n$?

Question 2 (Interpolation)

- Input: n pairwise distinct element $x_0, ..., x_{n-1} \in \mathbb{K}$
- Input: Polynoimial $P = p_{n-1}x^{n-1} + \ldots + p_0 \in \mathbb{K}[x]$ of degree less than n.
- How to efficiently compute a polynomial $P = p_{n-1}x^{P}n - 1 + \ldots + p_0 \in \mathbb{K}[x]$ s.t. $P(x_i) = y_i$ for all $0 \leq i < n$

*Those two questions are inverse to each other

5.1.1 Multipoint evaluation

The very naive algorithms take $2n$ multiplications and n addition. This should never be used. Instead of this algorithm, a less naive algorithm which is the Horner scheme is recommended. This take n multiplications and n addition.

Horner scheme

⋮

5.1.2 Interpolation

The first approach for interpolation is to rely on the Lagrange. In conclusion, it takes $O(n^2)$ in $\mathbb{K}$.

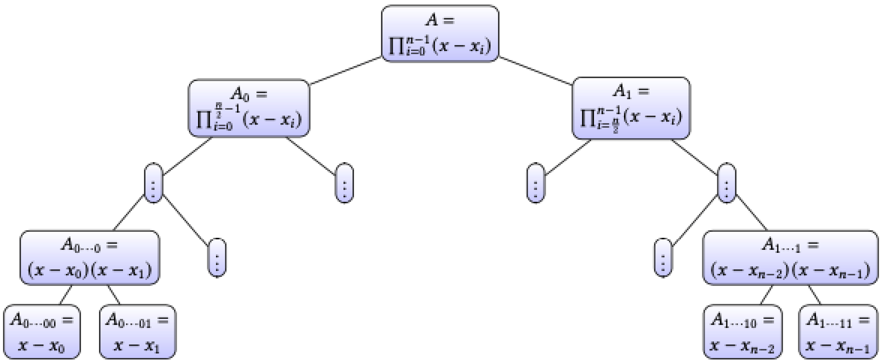
Lagrange interpolation : https://youtu.be/WCGKqJrf4N4

5.2 Fast multipoint evaluation

Computing $y_i = P(x_i)$ for $0 \leq i < n \quad \longrightarrow P \mod (x - x_i)$ for $0 \leq i < n$. We consider $n = 2^k$ for simplicity.

- First, compute $A = \prod_{i=0}^{n-1}(x - x_i)$ and a corresponding subproducts tree.
- Second, compute $P \mod (x - x_i)$ for $0 \leq i < n$ by exploiting the subproducts tree

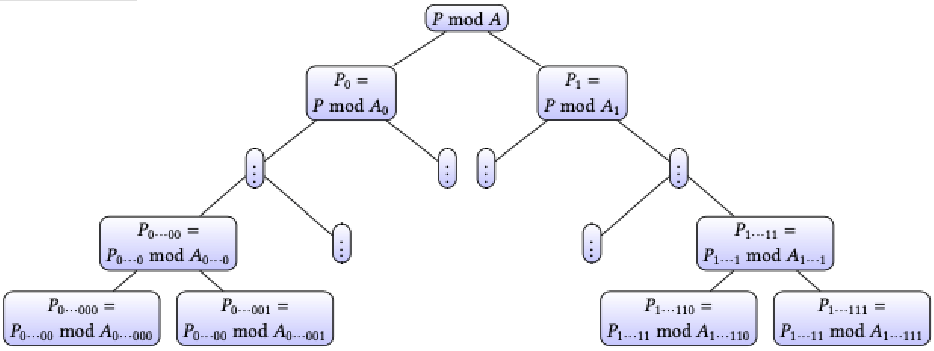
subproducts tree



Proposition 5.1

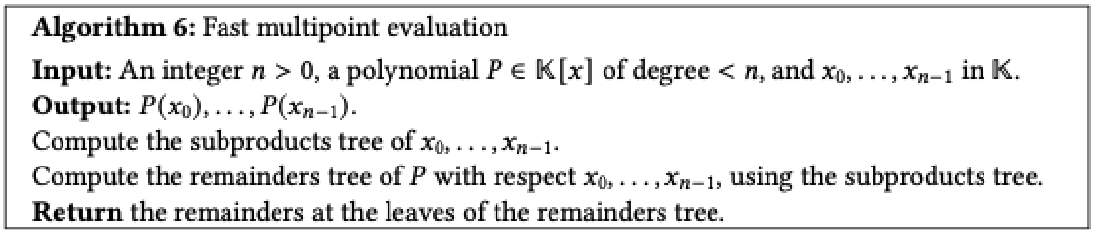
Building the subproducts tree from its leaves to to its root yield all the polynomials indicated in its nodes in a total of $O(M(n)log(n))$ operation in $\mathbb{K}$

reminders tree


P mod A
$P_0 =$ P mod A_0
$P_1 =$ P mod A_1
$P_{0\cdots00} =$ $P_{0\cdots0}$ mod $A_{0\cdots0}$
$P_{1\cdots11} =$ $P_{1\cdots1}$ mod $A_{1\cdots1}$
$P_{0\cdots000} =$ $P_{0\cdots00}$ mod $A_{0\cdots000}$
$P_{0\cdots001} =$ $P_{0\cdots00}$ mod $A_{0\cdots001}$
$P_{1\cdots110} =$ $P_{1\cdots11}$ mod $A_{1\cdots110}$
$P_{1\cdots111} =$ $P_{1\cdots11}$ mod $A_{1\cdots111}$

Proposition 5.2

Recall that we assume $deg(P) < n$. Having computed the suibproducts tree computing all the remainders in the remainders tree from its root to its leaves uses $O(M(n)log(n))$ operation in $\mathbb{K}$



```
Algorithm 6: Fast multipoint evaluation
Input: An integer $n > 0$, a polynomial $P \in \mathbb{K}[x]$ of degree $< n$, and $x_0, \ldots, x_{n-1}$ in $\mathbb{K}$.
Output: $P(x_0), \ldots, P(x_{n-1})$.
Compute the subproducts tree of $x_0, \ldots, x_{n-1}$.
Compute the remainders tree of $P$ with respect $x_0, \ldots, x_{n-1}$, using the subproducts tree.
Return the remainders at the leaves of the remainders tree.
```

Theorem 5.1

Fast polynomial multipoint evaluation can be performed in $O(M(n)log(n))$ operations in $\mathbb{K}$.

5.3 Fast Interpolation

⋮

6 Guessing Linear Recurrence Relations

6.1 The Berlekamp-Massey algorithm

$\vdots$

6.2 Sparse Matrix

Definition 6.1

A sparse representation of a matrix is such that only nonzero entries are stored.

Definition 6.2: Sparse Matrix

A sparse matrix is a matrix with many (most) coefficients that are zero.

*There are no specific numbers of zero to be sparse matrix

You can possibly compute matrix computations faster by taking into account that a matrix is sparse.

Data representation

You can use i, j coefficients for all non-zero coefficients.

$$\text{For example, } M = \begin{bmatrix} \alpha_1 & 0 & \dots & 0 \\ \alpha_2 & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \alpha_n & 0 & \dots & 0 \end{bmatrix}, \quad \begin{array}{c} 0, 0, \alpha_1 \\ 1, 0, \alpha_2 \\ 2, 0, \alpha_3 \\ \vdots \\ n-1, 0, \alpha_n \end{array}, \longrightarrow \text{representation of size } O(n)$$

6.2.1 Multiplication

Product of two sparse matrices is not necessary sparse.

Given a matrix with m nonzero entries, stored with a sparse representation, its product with a vector requires $O(m)$ operations in the base field. Recall that for $n \times n$ dense matrix, multiplication can be done $O(n^{2.38})$ with Coppersmith–Winograd algorithm.

Given two matrices A and B, with both size of $n \times n$ and we assume that the number of non-zero entries in those matrices are at most m_A, m_B. The naive algorithm of matrix multiplication can take some advantage of sparsity since the product of any element from A and B will be zero if either the corresponding entry in A or B is zero. Let, $\overline{a}_k$ and $\overline{b}_k$ be a number of non-zeror element in kth col (resp. row) of matrix A and B. In particular,

$$\sum_{k=1}^{n} \overline{a}_k = m_A \quad \text{and} \quad \sum_{k=1}^{n} \overline{b}_k = m_B$$

Then the number of multiplications in the base ring that one does in the naive matrix multiplication AB is

$$\sum_{k=1}^{n} \overline{a}_k \overline{b}_k \leq \left(\sum_{k=1}^{n} \overline{a}_k\right)\left(\sum_{k=1}^{n} \overline{b}_k\right) \leq m_A m_B$$

Note that one should also consider the number of additions, and that the details of such a naive sparse matrix multiplication algorithm will be highly dependent on the chosen sparse representation format.

6.2.2 Other computation

<u>Addition</u>

$A, B \in \mathbb{K}^{n \times n}$ with m_A, m_B non-zero entries.

$A + B$ in $O(m_A + m_B)$ operations and $A + B$ has $\leq m_A + m_B$ non-zero entries.

Matrix-vector multiplication

$A \in \mathbb{K}^{n \times n}$ with m_A non-zero entries, and $V \in \mathbb{K}^n$ (dense). AV in $O(m_A)$ operations.

Computing multiplication of dense and sparce matrices

$A \in \mathbb{K}^{n \times n}$ with m_A non-zero entries, and $B \in \mathbb{K}^{n \times n}$ (dense). Multiplication can be done in $O(m_a, n)$, faster than multiplication of dense matrices.

PLU decomposition

A sparse matrix can have a dense PLU decomposition. Example on the lecture note.

6.2.3 The Wiedemann algorithm

$\vdots$

7 Univariate and Bivariate Results

8 Structured linear algebra

9 Error Correcting Codes; decoding algorithm