Number Theoretic Algorithms

유한 체 다항식의 인수분해

October 15, 2019

1 좋은 다항식 찾기

이 자료에서는 지난 주 스터디 자료에서 느끼셨을 수도 있고, 어쩌면 이미 깨달아서 쓰고 계실 수도 있는 알고리즘을 엄밀하게 증명합니다. 유한 체 F 위에서는 모든 d 차 기약다항식의 곱을 약수로 가지는 다항식 $x^{|F|^d} - x$ 가 있습니다. 이 다항식은 F 의 characteristic이 2가 아니면 $x(x^u - 1)(x^u + 1)$ where $u := (|F|^d - 1)/2$ 로 쪼갤 수 있습니다. 이 다항식과의 최대공약수를 아무렇게나 구하는 방법으로 원하는 다항식을 인수분해할 수 있습니다. 이 알고리즘의 이름은 **Cantor-Zassenhaus**입니다.

이 알고리즘을 사용하기 위해 가장 궁극적인 질문, **왜 위 다항식이 모든 d 차 기약다항식을 약수로 가지는가?**에 대해 **초등적으로** 답하려고 합니다. 증명 자체는 어렵지는 않으나 매우 길어서 독립된 자료로 빼야 했습니다.

읽기 전에 반드시 7주차 연습문제 5번을 계산하고 오시기 바랍니다. 아래쪽에 계산 과정이 모두 있지만, 직접 계산하면서 계수가 어떻게 변화하는지 느끼는 것과 자료를 그냥 보는 것은 다르다고 생각합니다.

2 계수 다항식 다루기

7주차 연습문제 5번의 계산 과정입니다. 계산의 편의를 위해 $F = \mathbb{Z}_p$ 를 가정했으나, F 의 characteristic이 p 라고 놓으면 똑같은 계산을 할 수 있습니다.

$$\begin{aligned} \prod_{a,b \in F} (x^2 + xa + b) &= \prod_{a \in F} ((x^2 + xa)^p - (x^2 + xa)) \\ &= \prod_{a \in F} (x^{2p} + x^p a - (x^2 + xa)) \\ &= \prod_{a \in F} ((x^{2p} - x^2) + (x^p - x)a) \\ &= \prod_{a \in F} (x^p - x)((x^p + x) + a) \\ &= (x^p - x)^p ((x^p + x)^p - (x^p + x)) \\ &= (x^p - x)^p (x^{p^2} - x). \end{aligned}$$

$$\begin{aligned}
\prod_{a,b,c \in F} (x^3 + x^2a + xb + c) &= \prod_{a,b \in F} ((x^3 + x^2a + xb)^p - (x^3 + x^2a + xb)) \\
&= \prod_{a,b \in F} ((x^{3p} - x^3) + (x^{2p} - x^2)a + (x^p - x)b) \\
&= (x^p - x)^{p^2} \prod_{a,b \in F} ((x^{2p} + x^{p+1} + x^2) + (x^p + x)a + b) \\
&= (x^p - x)^{p^2} \prod_{a \in F} ((x^{2p^2} + x^{p^2+p} - x^{p+1} - x^2) + (x^{p^2} - x)a) \\
&= (x^p - x)^{p^2} \prod_{a \in F} ((x^{p^2} - x)(x^{p^2} + x^p + x) + (x^{p^2} - x)a) \\
&= (x^p - x)^{p^2} (x^{p^2} - x)^p \prod_{a \in F} ((x^{p^2} + x^p + x) + a) \\
&= (x^p - x)^{p^2} (x^{p^2} - x)^p (x^{p^3} - x).
\end{aligned}$$

계산 과정에서 몇 가지 눈여겨 보셔야 할 점을 짚겠습니다.

- 각 a, b, c 앞에 붙은 (x 에 대한) 다항식 계수가 변화하는 과정은 독립적입니다. 이 말은, 맨 처음에 다항식 계수가 같았다면, 소멸까지의 과정도 똑같다는 말입니다. 앞 계산의 1의 계수와 뒤 계산의 a 의 계수를 주목해서 한 번 더 읽어 보세요.
- 매 알파벳을 없앨 때마다, 각 계수의 공통 인수가 있고, 이는 현재까지 공통 인수를 묶어낸 횟수를 c 라 할 때 $(x^{p^{c+1}} - x)$ 입니다. 뒤 계산을 알파벳을 없애는 시점에 주목해서 한 번 더 읽어 보세요.

따라서, 계수 다항식에 대해 알파벳을 없애는 작업과 공통 인수를 묶어내는 작업을 조직적으로 다루는 방법이 필요해 보입니다.

앞으로의 논의의 편의를 위해 다음 표기법을 정의합니다.

$$[a_n, \dots, a_0] := x^{anp^n + \dots + a_0}$$

역시 표기법의 편의를 위해 $a_n = 0$ 인 것도 허용합니다.

2.1 진보 과정

진보 과정은 다음과 같이 정의됩니다.

$$\text{adv } f := f^p - f \text{ or } \text{adv} \left(\sum_{i=1}^k [a_{i,n}, \dots, a_{i,0}] \right) := \sum_{i=1}^k ([a_{i,n}, \dots, a_{i,0}, 0] - [a_{i,n}, \dots, a_{i,0}])$$

두 정의가 같음은 직접 확인해 보시기 바랍니다.

각 계수 다항식에 대해 알파벳을 하나 없앨 때마다 진보 과정이 일어나는데, 이는 식에 의해 자명합니다.

2.2 보수 과정

c 차 보수 과정은 다음 나눗셈이 다항식으로 잘 정의된다면 다음과 같이 정의됩니다.

$$\text{fix}_c f := \frac{f}{x^{p^c} - x}$$

만일 나누어떨어지지 않으면 c 차 보수 과정이 **실패한다**고 말하겠습니다.

알파벳이 c 개라면, 각 계수 다항식에 대해 보수 과정은 총 c 번 일어나는데, 1차 보수 과정, 2차 보수 과정, $\dots$, c 차 보수 과정이 순서대로 한 번씩 일어납니다. 물론 우리는 아직 이 과정이 실패하지 않는지 증명하지는 않았습니다.

2.3 단항식에 대한 증명

다음을 보이면 됩니다.

Claim. $c \leq d$ 라 하면, 임의의 다항식 x^d 에 대해 진보 과정, 1차 보수 과정, 진보 과정, 2차 보수 과정, $\dots$, 진보 과정, c 차 보수 과정은 실패하지 않으며, 그 결과 다항식은 a_i 들이 모두 음이 아닌 정수일 때

$$\sum_{a_0 + \dots + a_c = d - c} [a_c, a_{c-1}, \dots, a_1, a_0]$$

이 된다.

사실 이 명제는 $c > d$ 일 때도 성립하지만, 이를 다루는 것은 의미가 없는데, $c = d$ 이면 결과 다항식이 1이 되고, 따라서 진보 과정을 거치면 0이 되어 보수하거나 진보하는 의미가 없기 때문입니다.

Proof. c 에 대한 수학적 귀납법을 통해 합니다. $c = 0$ 인 경우 자명합니다. c 인 경우 성립한다고 합시다. 진보 과정을 거치면 다항식은

$$A := \sum_{a_0 + \dots + a_c = d - c} [a_c, a_{c-1}, \dots, a_1, a_0, 0] - \sum_{a_0 + \dots + a_c = d - c} [a_c, a_{c-1}, \dots, a_1, a_0]$$

이 되는데, 앞 합의 $a_c = 0$ 이면, 뒤 합의 $a_0 = 0$ 인 것과 상쇄되어 사라질 것이므로,

$$\begin{aligned} A &= \sum_{a_0 + \dots + a_c = d - c} [a_c, a_{c-1}, \dots, a_1, a_0, 0] - \sum_{a_0 + \dots + a_c = d - c} [a_c, a_{c-1}, \dots, a_1, a_0] \\ &= \sum_{a=1}^{d-c} \sum_{a_0 + \dots + a_{c-1} = d - c - a} [a, a_{c-1}, \dots, a_1, a_0, 0] - \sum_{a=1}^{d-c} \sum_{a_1 + \dots + a_c = d - c - a} [a_c, a_{c-1}, \dots, a_1, a] \\ &= \sum_{a=1}^{d-c} \sum_{a_0 + \dots + a_{c-1} = d - c - a} ([a, a_{c-1}, \dots, a_1, a_0, 0] - [0, a_{c-1}, \dots, a_1, a_0, a]) \\ &= \sum_{a=1}^{d-c} \sum_{a_0 + \dots + a_{c-1} = d - c - a} [a_{c-1}, \dots, a_1, a_0, 0] (x^{ap^{c+1}} - x^a). \end{aligned}$$

그런데 $a \geq 1$ 이면

$$\begin{aligned} x^{ap^{c+1}} - x^a &= (x^{p^{c+1}} - x)(x^{(a-1)p^{c+1}} + x^{(a-2)p^{c+1}+1} + \dots + x^{a-1}) \\ &= (x^{p^{c+1}} - x) \sum_{i=0}^{a-1} [i, \underbrace{0, \dots, 0}_{c-1 \text{ zeros}}, a-1-i] \end{aligned}$$

이므로, A 에 대해 $(c+1)$ 차 보수 과정을 진행할 수 있고, 그 결과는 원하는 대로

$$\begin{aligned} \text{fix}_{c+1} A &= \sum_{a=1}^{d-c} \sum_{a_0+\dots+a_{c-1}=d-c-a} \left([a_{c-1}, \dots, a_1, a_0, 0] \cdot \sum_{i=0}^{a-1} [i, \underbrace{0, \dots, 0}_{c-1 \text{ zeros}}, a-1-i] \right) \\ &= \sum_{a=1}^{d-c} \sum_{a_0+\dots+a_{c-1}=d-c-a} \sum_{i=0}^{a-1} [i, a_{c-1}, \dots, a_1, a_0, a-1-i] \\ &= \sum_{i=0}^{a-1} \sum_{a=1}^{d-c} \sum_{a_0+\dots+a_{c-1}=d-c-a} [i, a_{c-1}, \dots, a_1, a_0, a-1-i] \\ &= \sum_{i=0}^{a-1} \sum_{a_{-1}+a_0+\dots+a_{c-1}=d-c-i-1} [i, a_{c-1}, \dots, a_1, a_0, a_{-1}] \\ &= \sum_{a_{-1}+a_0+\dots+a_{c-1}+a_c=d-c-1} [a_c, a_{c-1}, \dots, a_1, a_0, a_{-1}] \end{aligned}$$

가 됩니다. □

3 기약다항식으로의 이동

위 사실을 이용하면 다음을 아주 쉽게 증명할 수 있습니다.

$$\prod_{a_0, \dots, a_{n-1} \in F} (x^n + a_{n-1}x^{n-1} + \dots + a_1x + a_0) = \prod_{i=1}^n (x^{p^i} - x)^{n+1-i}$$

그런데 이 다항식의 대부분은 중복 factor를 가지게 됩니다. 수학적 귀납법을 통해 $x^{p^n} - x$ 가 n 차 이하의 다항식으로 인수분해됨을 보이고 나면, n 차 기약다항식은 모두 $x^{p^n} - x$ 에 속하게 됨을 보일 수 있습니다. 따라서 그러한 다항식을 찾았기 때문에, Cantor-Zassenhaus 알고리즘을 시행할 수 있습니다.

$p = 2$ 인 경우가 역시 문제가 되는데, 지난 주와 마찬가지로 F 의 characteristic이 2라면 성립하는 다음 식

$$x^{2^n} - x = (x^{2^{n-1}} + x^{2^{n-2}} \dots + x)^2 + (x^{2^{n-1}} + x^{2^{n-2}} + \dots + x)$$

을 이용하면 쉽게 이 문제를 회피할 수 있습니다.

◇

이 스터디를 처음 기획할 때 생각했던 모든 내용을 다루었습니다. 정말로 수고 많으셨습니다.

처음에는 PS에 쓰이는 수론적 알고리즘을 제대로 이해할 수 있게 하거나, 왜 작동하는지에 대한 궁금증을 해소하고, computational number theory 분야의 PS에 소개되지 않은 알고리즘을 간단히 소개하는 선에서 그치려고 했는데, 스터디를 진행하다 보니 생각보다 재욕심이 과해져서 어찌다 보니 여러 알고리즘을 조금 깊게 다루게 되었습니다. 내용이 많이 어려웠나요?

저 역시 이 스터디를 준비하면서 처음에는 이해할 수 있으리라 생각지도 못한 내용을 이해하고 구현하며 성장했습니다. 또 성장의 기록을 모두 자료로 남길 수 있었습니다. 60페이지가량 되는 $(10 + \alpha)$ 개의 자료를 쓸 수 있게 해 주신 여러분께 감사의 말씀을 드립니다.

이 스터디는 이제 필요한 내용을 모두 했기 때문에 언제든지 종료될 수 있지만, 제가 준비할 수 있는 내용 중에 여러분이 원하시는 내용으로 몇 번 더 스터디를 진행할 예정입니다.

4 문제

1. n 을 입력하면, x^n 이 0이 될 때까지 진보 과정과 c 차 보수 과정을 번갈아가며 진행하는 프로그램을 작성하세요. 5 이상의 분석이 용이한 n 에 대해 프로그램을 돌려 보고, 2.3의 Claim에 대한 확신을 가지세요.
2. 임의의 유한 체 F 에 대해 $|F| = n$ 이라 합니다. 임의의 d 에 대해 다음을 증명하세요.

$$\prod_{a_0, \dots, a_{d-1} \in F} (x^d + a_{d-1}x^{d-1} + \dots + a_d x + a_d) = \prod_{i=1}^d (x^{n^i} - x)^{d+1-i}$$

증명 과정은 $\mathbb{Z}_p$ 와 거의 유사하나, 한 번에 $n = p^i$ 제곱을 하는 식을 어떻게 잘 처리할지가 관건입니다.

3. (a) $\gcd(x^a - 1, x^b - 1) = x^{\gcd(a,b)} - 1$ 임을 보이세요. Hint.¹
 (b) $\mathbb{Z}_p$ 에서 $x^{p^d} - x$ 가 d 의 약수 차수의 모든 기약다항식들의 곱임을 증명하세요.
 (c) 차수가 정확히 $d > 1$ 인 기약다항식의 개수는

$$\frac{1}{d} \sum_{i|d} \mu(d/i) (p^i - 1)$$

임을 보이세요.² $\mu(i)$ 는 뫼비우스 함수라고 부르며, i 가 제곱수로 나누어떨어지는 경우 $\mu(i) = 0$, 아닌 경우 i 의 소인수의 개수를 k 라 하면 $\mu(i) = (-1)^k$ 로 정의됩니다. 편의상 $\mu(1) = 1$ 로 정의됩니다.

¹ $a > b$ 라 하면, $\gcd(x^a - 1, x^b - 1) = \gcd(x^a - x^b, x^b - 1) = \gcd(x^b(x^{a-b} - 1), x^b - 1)$.

²이 식의 형태에 관심이 있으신 분은 Möbius inversion formula를 찾아보시기 바랍니다.