6.885 Algebra and Computation	October 17, 2005
Lecture 10	
Lecturer: Madhu Sudan	*Scribe: Ben Rossman*

1 Clarification on Resultants

Consider polynomials $f(x), g(x) \in R[x]$ where R is a unique factorization domain (with some notion of order on its elements). In the last lecture, the *resultant* of f and g was defined as the "minimal" element in the ideal I of $R[x]$ generated by f and g, i.e., the polynomial with smallest leading coefficient among all nonzero polynomials of lowest degree in I. While this notion of *resultant* serves our purposes, be cautioned that this is not the standard definition.

Notice that f and g have no common factor if and only if there exist $a(x), b(x) \in R[x]$ such that $\deg a < \deg g$, $\deg b < \deg f$ and $a(x)f(x) + b(x)g(x) \in R - \{0\}$. The problem of finding a and b such that $a(x)f(x) + b(x)g(x) = 1$ can be represented as a linear system of $\deg f + \deg g$ equations (for the coefficients of f and g) in $\deg f + \deg g$ unknowns (the coefficients of a and b). The determinant of the matrix for this system is what is usually called the *resultant*. In particular, this is an element of R which is zero if and only if f and g have a common factor.

2 Factoring in $\mathbb{F}_q[x, y]$

We are given a polynomial $f(x, y) \in \mathbb{F}_q[x, y]$ that is assumed to be monic in x. (We later show how to get rid of this assumption.) Our goal is to output a nontrivial factor of f if f is reducible, or else to report that f is irreducible. Our algorithm consists of five steps. (In the following, where we write "large enough" or "sufficiently large", exactly how large will be specified when we do the analysis.)

Step 1 Check if $f(x, y)$ and $\frac{\partial f}{\partial x}(x, y)$ are relatively prime (i.e. if $f(x, y)$ is squarefree). If not, then output $\gcd(f, \frac{\partial f}{\partial x})$ and stop. Otherwise, proceed to Step 2.

Step 2 For a large enough extension field $K \supseteq \mathbb{F}_q$, find $y_0 \in K$ such that $f(x, y_0)$ and $\frac{\partial f}{\partial x}(x, y_0)$ are relatively prime.

Step 3 Find an extension field $L \supseteq K$, element $\alpha \in L$ and $h(x) \in L[x]$ such that $f(x, y) = (x - \alpha) \cdot h(x) \mod (y - y_0)$.

Remark. Note that $(x - \alpha)$ and $h(x)$ are relatively prime, since $f(x, y_0)$ is squarefree. Therefore, we can apply Hensel's Lifting Lemma.

Step 4 Apply Hensel iterations to obtain $A(y) \in L[y]$ and $H(x, y) \in L[x, y]$ such that

$$f(x, y) = (x - A(y)) \cdot H(x, y) \mod (y - y_0)^t$$

where $A(y_0) = \alpha$ and $H(x, y_0) = h(x)$ for some sufficiently large t (a power of two).

Step 5 If there exist nonzero polynomials $g(x, y), H'(x, y) \in L[x, y]$ such that

$$\begin{aligned}
&\deg_x g < \deg_x f \\
&\deg_y g \leq \deg_y f \\
&g(x, y) = (x - A(y)) \cdot H'(x, y) \mod (y - y_0)^t,
\end{aligned}$$

then find any such $g(x, y)$ and output $\gcd(f, g)$. Otherwise (if no such g exists), output "f is irreducible".

Remark. If $g(x, y)$ and $H(x, y)$ exist, they can be found efficiently by solving a linear system over L.

2.1 Feasibility

We want to show that the algorithm outlined above is correct as well as feasible (i.e. has polynomial runtime). We first establish feasibility. There are three parameters in the outline of the algorithm we need to bound: the sizes of fields K and L, and the size of t.

We need K to be large enough so that there is guaranteed to exist y_0 such that $f(x, y_0)$ and $\frac{\partial f}{\partial x}(x, y_0)$ have no root in common. We claim that it suffices to take $\|K\| > 2d(d-1)$ where $d = \deg_x f$. To see this, consider the *discriminant* $\Delta(y)$ of f with respect to x, defined as the resultant of f and $\frac{\partial f}{\partial x}$ in the ring $\mathbb{F}_q[y][x]$. As we have seen, $\Delta(y)$ is a polynomial in $\mathbb{F}_q[y]$ of degree $\leq 2d(d-1)$. Therefore, since $\|K\| > 2d(d-1)$, there exists $y_0 \in K$ that is not a root of $\Delta(y)$. It follows that $f(x, y_0)$ and $\frac{\partial f}{\partial x}(x, y_0)$ have no common root.

What about the size of L? L is required to contain a root $f(x, y_0)$, which could be an irreducible polynomial of degree d. So, the most we can say about L is that it is an extension field of degree at most d over K. This is okay, however, since elements of L have polynomial-size representations (as d-tuples over K), and each field operation over L reduces to polynomially-many field operations over $\mathbb{F}_q$.

We now turn to t (i.e. two to the number of Hensel iterations). We set t equal to the least power of two greater than $2d^2$. (We will need this assumption in order to prove correctness of the algorithm.) Thus, our algorithm involves only $2\log_2(d) + 2$ Hensel liftings. Since each Hensel lifting requires poly(d) operations over L (as we saw in the previous lecture), the overall running time is polynomial in d and q.

Remark. The algorithm presented here is not optimized for efficiency. More sophisticated algorithms are known that achieve better running times.

2.2 Correctness

Let $f_1(x, y), ..., f_k(x, y)$ be the irreducible factors of $f(x, y)$, all monic in x. When we substitute y_0 for y, the resulting polynomials $f_i(x, y_0)$ are no longer necessarily irreducible. However, because $f(x, y_0)$ is squarefree (by our choice of y_0), we know that $(x - \alpha)$ divides exactly one of the $f_i(x, y_0)$, say $f_1(x, y_0)$, without loss of generality. The following two claims now establish the correctness of the algorithm.

Claim 1 *If $k > 1$ then $f_1(x, y)$ is a candidate for $g(x, y)$ in Step 5.*

Proof There exists $h_1(x) \in L[x]$ such that

$$f_1(x, y) = (x - \alpha)h_1(x) \mod (y - y_0).$$

$(x - \alpha)$ and $h_1(x)$ are relatively prime, since $f(x, y_0)$ is squarefree. So we can apply Hensel liftings to get polynomials $A_1(y)$ and $H_1(x, y)$ such that $A_1(y_0) = \alpha$, $H_1(x, y_0) = h_1(x)$ and

$$f_1(x, y) = (x - A_1(y))H_1(x, y) \mod (y - y_0)^t.$$

Setting $\widetilde{H}(x, y) = H_1(x, y) \cdot \big(f_2(x, y) \cdots f_k(x, y)\big)$, we have

$$f(x, y) = (x - A_1(y))\widetilde{H}(x, y) \mod (y - y_0)^t.$$

From the uniqueness of monic Hensel liftings, it follows that $A_1(y) = A(y) \mod (y - y_0)^t$. Since $k > 1$, we have $\deg_x(f_1) < \deg_x(f)$. $f_1(x, y)$ is thus a suitable candidate for $g(x, y)$ in Step 5. ■

Claim 2 *$f_1(x, y)$ divides every candidate $g(x, y)$ in Step 5.*

Proof Toward a contradiction, assume f_1 does not divide g. Since f_1 is irreducible, f_1 and g have no common factor. Therefore, by the results of the last lecture, the resultant $R(y) = \mathrm{Res}_x(f_1, g)$ is a polynomial of positive degree $\leq 2d^2$. Write $R(y)$ as $a(x,y)f_1(x,y) + b(x,y)g(x,y)$ for some polynomials $a(x,y)$ and $b(x,y)$. We have

$$\begin{aligned} R(y) &= a(x,y)(x - A(y))H_1(x,y) + b(x,y)(x - A(y))H'(x,y) \mod (y - y_0)^t \\ &= (x - A(y)) \cdot (a(x,y)H_1(x,y) + b(x,y)H'(x,y)) \mod (y - y_0)^t. \end{aligned}$$

By our choice of $t > \deg_y(R)$, we ensure that $R(y) \neq 0 \mod (y - y_0)^t$. Therefore,

$$a(x,y)H_1(x,y) + b(x,y)H'(x,y) \neq 0 \mod (y - y_0)^t.$$

It follows that $(x - A(y)) \cdot (a(x,y)H_1(x,y) + b(x,y)H'(x,y))$ has nonzero degree in x. However, this contradicts that $\deg_x(R) = 0$, i.e., R is a polynomial in y alone. ■

2.3 Factoring non-monic $f(x,y)$

In our description of the algorithm, we assume that input polynomial $f(x,y)$ is monic in x. We now show how to eliminate this assumption. Suppose $f(x,y)$ is not monic in x, and rewrite it as

$$f(x,y) = g(y)x^d + \mathcal{O}(x^{d-1}).$$

Define a new polynomial $\widetilde{f}(z,y)$ by

$$\widetilde{f}(z,y) = g(y)^{d-1} \cdot f(\tfrac{z}{g(y)}, y).$$

$\widetilde{f}(z,y)$ is clearly monic in z, so we apply our algorithm to factor $\widetilde{f}(z,y) = \widetilde{h}(z,y)\widetilde{\ell}(z,y)$. We now have

$$f(x,y) = \tfrac{1}{g(y)^{d-1}} \widetilde{h}(g(y)x, y)\widetilde{\ell}(g(y)x, y).$$

3 A few words on multivariate factoring

Our algorithm for factoring in $\mathbb{F}_q[x,y]$ generalizes to rings $R[x,y]$ where R is any ring over which we have an efficient univariate factorization algorithm. This immediately suggests an approach to factoring multivariate polynomials in $\mathbb{F}_q[x_1, ..., x_c]$, since c-variate factorization over $\mathbb{F}_q$ is the same as univariate factorization over $\mathbb{F}_q[x_1, ..., x_{c-1}]$. Unfortunately, the algorithm implies by this approach has the abysmal running time of $\mathrm{poly}(d)^{\left.2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}\right\}\text{height } c}$.

There is another, more efficient generalization of our algorithm. We outline the trivariant case. Suppose we are given input polynomial $f(x,y,z) \in \mathbb{F}_q[x,y,z]$ that is monic in x. After checking that f and $\frac{\partial f}{\partial x}$) have no common factor,find a pair of values y_0 and z_0 (in a suitable extension field of $\mathbb{F}_q$) such that $f(x, y_0, z_0)$ and $\frac{\partial f}{\partial x}$)$(x, y_0, z_0)$ have no common factor. Next find a nice factorization of f modulo the ideal generated by $(y - y_0)$ and $(z - z_0)$ in the ring $L[x,y,z]$ for another extension field L of $\mathbb{F}_q$. Now apply Hensel liftings (Step 4) and look for a polynomial g with the right profile (Step 5). This method results in an efficient running time $\mathrm{poly}(d^c)$ where $d = \deg(f)$ and c is the number of variables.

One might be tempted ask: is there a multivariate factorization algorithm with running time polynomial in c? The question is absurd if we continue to represent polynomials explicitly (i.e. by enumerating their coefficients), since a polynomial in c variables of degree d has $\mathcal{O}(d^c)$ coefficients in general. In order to make the question meaningful, we have to consider other ways of representing polynomials.

Kaltofen and Trager devised a clever algorithm for finding the factors of a polynomial $f(x_1, ..., x_c)$ that is given as black box (together with essential information like the degree of f). The output of the algorithm is, of course, not an explicit polynomial, but a procedure (involving calls to the original black-box function) for computing the values of some irreducible factor of f. The Kaltofen-Trager algorithm achieves $\text{poly}(d, c)$ running time. (The bivariate factorization in the previous section is also due to Kaltofen. Other bivariate factorization algorithms are due to Lenstra and Grigoriev.)

4 Factoring in $\mathbb{Z}[x]$

As out final topic today, we begin outlining an algorithm for factoring integer polynomials. Let $f \in \mathbb{Z}[x]$ be a monic polynomial of degree d whose coefficients are between -2^n and 2^n. The representation of f is thus a bit-string of length $\mathcal{O}(dn)$. The procedure for factoring f uses many of the same ideas as the bivariate factorization algorithm. The five steps below are adapted from the corresponding steps in the bivariate algorithm.

Step 1 Check for common factors of f and $\frac{\partial f}{\partial x}$. If $\gcd(f, \frac{\partial f}{\partial x})$ is nontrivial, then we report it and stop. Otherwise, we continue to Step 2.

Step 2 Find prime $p \in \mathbb{Z}$ such that f and $\frac{\partial f}{\partial x}$ have no common factors modulo p. (We use resultants and the Chinese Remainder Theorem to prove small p exists.)

Step 3 Factor $f(x)$ modulo p as $f(x) = g(x)h(x) \mod p$ where g, h are relatively prime and g is irreducible.

Step 4 Apply Hensel liftings to obtain $G(x)$ and $H(x)$ such that

$$\begin{aligned} G(x) &= g(x) \mod p \\ H(x) &= h(x) \mod p \\ f(x) &= G(x)H(x) \mod p^t. \end{aligned}$$

Here we have the added burden of proving that the coefficients of G and H cannot be too large; $\pm 2^{n^2}$ is a rough bound.

Step 5 Find $\widetilde{g}(x)$ such that $\widetilde{g}(x) = G(x)H'(x) \mod p^t$, $\deg(\widetilde{g}) < d$ and all coefficients of $\widetilde{g}$ have size $\leq 2^{n^2}$. (Note that $\widetilde{g}(x)$ satisfying the first two conditions can be found by solving a linear system. The third condition, i.e. bounding the size of coefficients, is what makes the problem difficult.)

Exercise Prove analogues of Claims 1 and 2 in this setting.