Polynomial factorization and lattices in the very early 1980s

Perspectives of an early user of lattice basis reduction

Arjen K. Lenstra

EPFL IC LACAL and Bell Laboratories
INJ 330, Station 14, 1015-Lausanne, Switzerland

In this brief note I describe some polynomial factorization and lattice basis reduction developments that I was closely involved with. It all started with a list of irreducible monic polynomials over the integers and the question if the algebraic number fields generated by their roots were isomorphic (cf. [1]). Although symbolic algebra packages that deal with such problems were in principle available, they were not available to us. With Henk Bos and my fellow numerical analysis student Rudolf Mak, we set out to study and implement methods to factor univariate polynomials over algebraic number fields. It turned out to be a bit more challenging than anticipated, not just because our Algol68 punch-cards jobs had annoyingly long turn-around times (often just to add a missing semi-colon), but also because of all the new literature that we had to familiarize ourselves with.

Our implementation got about halfway Berlekamp-Hensel to factor univariate polynomials over the rationals. But we had gained a much better understanding of the complexity involved in extending our work to polynomials over algebraic number fields. And, it had wetted my appetite for the subject: I much preferred toying around with integers to the messier type of error analysis that was expected from students in the numerical analysis group. Therefore, my master's thesis project was to complete the earlier, unfinished project and to produce a working program to factor univariate polynomials over algebraic number fields. That program would then be able to answer the isomorphism questions.

Unfortunately – though with hindsight it was most definitely fortunate for me – simply generalizing Berlekamp-Hensel from univariates over the rationals to univariates over algebraic number fields runs into a snag that is quite a bit more worrisome than the related snag that Berlekamp-Hensel has to deal with anyhow. This more serious snag is essential to this story. To appreciate it, we first need to understand its first incarnation in Berlekamp-Hensel over the rationals.

First, unlike integers it seems, repeated factors of polynomials are easily cast out by computing the greatest common divisor with the polynomial's derivative. Polynomials are therefore assumed to be square-free. To factor a square-free polynomial $f \in \mathbf{Q}[X]$, one selects a prime p such that f remains square-free when taken modulo p, uses Berlekamp's algorithm to find the factorization of f modulo p followed by Hensel's lemma to 'lift' the factorization to an appropriately high power of p (using Mignotte's bound to decide how far to go), and then derives the 'true' rational factorization by searching among combinations formed by products of the modular factors.

What is the snag? It is easy to find irreducible polynomials that split, modulo any prime, into lots of factors. At the time generating such polynomials was an entire cottage industry, also to test and compare competing implementations of Berlekamp-Hensel. As a consequence, due to the search in the last step, the worst-case runtime of Berlekamp-Hensel is hopelessly exponential in the degree of the polynomial to be factored. Because of this unavoidable bottleneck in the last step, no one even bothered to analyse the precise behavior of the other steps: the size of the smallest prime that maintains square-freeness, the resulting runtime of Berlekamp's algorithm (probabilistic polynomial time, but linear in p when its deterministic version is used), or the lifting to the proper bound (though the Mignotte-bound itself was obviously polynomial).

When generalizing Berlekamp-Hensel from the rationals to algebraic number fields, one can certainly not expect the runtime to become any better – i.e., it is clearly not realistic to expect that it gets better than exponential in the degree of the polynomial to be factored. On the other hand, one may hope that it does not deteriorate either. But, as a matter of fact, it did deteriorate quite badly. Here is how.

Let g be a monic irreducible polynomial of degree d over the integers, let $g(\alpha) = 0$, and let f be a square-free polynomial to be factored over $\mathbf{Q}(\alpha)$. To be able to use Berlekamp-Hensel, one would like to define a finite field that properly corresponds to $\mathbf{Q}(\alpha)$, factor f over the finite field, lift its factorization, and then search for factors. The obvious choice that would make everything work smoothly, is to take a prime p such that g remains irreducible modulo p. In that case $(\mathbf{Z}/p\mathbf{Z})[X]/(g(X))$ is a finite field of cardinality p^d over which f can be factored using Berlekamp's algorithm. The resulting factorization modulo g and p then indeed corresponds to the factorization of f over $\mathbf{Q}(\alpha)$. Assuming that square-freeness is maintained, this factorization can be lifted to a factorization modulo g and a sufficiently high power of p and the factorization over $\mathbf{Q}(\alpha)$ can be completed, as usual, by a search among combinations of the lifted factors. It would be exponential in the degree of f, but that is the best one can hope for anyhow.

As should be obvious now, the above convenient choice of prime is in general not possible, and Berlekamp-Hensel over algebraic number fields does not always run as smoothly as described above. Instead of just a single finite field $(\mathbf{Z}/p\mathbf{Z})[X]/(g(X))$, one gets a finite field for each irreducible factor of g modulo p. The polynomial f must be factored over each of these finite fields and then lifted (after also lifting the factors of g modulo p to factors modulo the same high enough power of p). The resulting modular factorizations of f modulo the lifted factors of g must then first be combined using Chinese remaindering to factors of f modulo g and the power of p, after which the search among combinations consisting of products can be done. Since there is no a priori way to tell which factors must be combined using Chinese remaindering, also the first combination step involves a combinatorial search, and the overall algorithm is exponential in the product of the degrees of g and f. In practice it often worked – I managed to answer the isomorphism questions and thus to complete my master's thesis

– but it was a rather unsatisfactory method. This Chinese remaindering based approach was described in [3]. Another approach suggested in [2] suffered from a similarly poor worst-case performance.

When discussing this issue with Hendrik, he suggested it may be possible to replace the Chinese remaindering by a lattice step in the following way. Without loss of generality, let c in $\mathbf{Z}[\alpha]$ be a coefficient of a factor of f over $\mathbf{Q}(\alpha)$, and assume that the minimum polynomial g has a monic linear factor h_k modulo some power p^k of p. When c is taken modulo h_k and p^k (regarding h_k as a polynomial in $\mathbf{Z}[\alpha]$), it results in an integer value c_k that also will appear as one of the coefficients of the (combinations of) factors of f modulo h_k and p^k: for each c there is an integer ℓ and a polynomial t of degree at most $d-1$ in $\mathbf{Z}[\alpha]$ such that $c = c_k + \ell \cdot p^k + t \cdot h_k$. Phrased differently, c and c_k are congruent modulo the d-dimensional integer lattice generated by the vectors $(p^k, 0, 0, \ldots, 0)$, $(h_{k0}, 1, 0, 0, \ldots, 0)$, $(0, h_{k0}, 1, 0, 0, \ldots 0)$, ... $(0, 0, \ldots 0, h_{k0}, 1)$, where $h_k = \alpha + h_{k0}$. Because c is fixed but c_k and h_{k0} may be expected to grow with k, one may expect that for large enough k, the coefficient c will be the unique shortest vector that is congruent to c_k modulo the lattice as generated above. And if that is indeed the case, then one can construct the c corresponding to a given c_k (and h_k) by finding a reduced lattice basis for which the fundamental domain contains a sphere around the origin that is large enough to contain c, and by reducing c_k modulo that reduced basis.

Lattice basis reduction we knew how to do from Hendrik's ILP paper. It did not run in polynomial time, but it had to be done just once and, moreover, who cares about such petty issues when dealing with an algorithm that runs in exponential time anyhow? So, the lattice approach was implemented, and it turned out to work beautifully. One of its fun features was the possibility to hunt for a linear factor of g modulo p, which meant that all polynomial factorizations could be done in prime fields as opposed to more cumbersome extension fields, and that more complicated 'polynomial' coefficients only had to be dealt with during the trial division steps for the combined factors.

In the experiments, all values (c_k and h_{k0}) grew as hoped for and c was consistently located as the unique shortest vector congruent to c_k modulo the reduced basis for the lattice generated by p^k and h_k. What more was needed? What was needed was a proof that the approach indeed always works as expected, including an estimate what value of k one should use to be able to derive valid irreducibility results.

It sufficed to prove that in the lattice generated by p^k and h_k, the lengths of the non-zero vectors grew with k, because then also the reduced basis vectors would grow. Based on the provable upper bound on their orthogonality defect, the required lower bound for the radius of the embedded sphere could then be established. I could not immediately get a handle on this problem, unsure what properties of the lattice vectors should or could be used to prove a lower bound on their lengths. My lack of understanding of the situation reached its zenith when, in my confusion, I mistakenly allowed a polynomial t of degree at most d in $c = c_k + \ell \cdot p^k + t \cdot h_k$, as opposed to $d-1$ as above. This leads to a $(d+1)$-

dimensional lattice, as opposed to the correct d-dimensional one, a lattice that, rather disturbingly, always contains a non-zero vector of short, fixed length, namely g itself. This observation baffled me for a while, but then quickly led to the desired result: apparently the property I needed was coprimality with g over the integers, yet a factor h_k in common with g modulo p^k. This property I could then indeed use to derive the lower bound proof – a very inelegant proof that is now happily lost in oblivion. In any case, I now knew for sure that my sphere would never collapse, and thus that I could factor polynomials over algebraic number fields faster than before. How much faster precisely, no one seemed to care, since the overall algorithm was still exponential in the degree of f.

The initially disturbing observation had an interesting side-result, namely that for sufficiently large k, the irreducible polynomial g can be found as a shortest non-zero vector in a lattice defined by any of its irreducible modular factors – or by an irreducible factor modulo p^k of a polynomial one wants to factor of which g is an unknown factor. This implied that if one lifts far enough, the combinatorial search in Berlekamp-Hensel can be avoided at the cost of shortest vector computations in various lattices. Furthermore, by pushing k even further, the shortest vector computations can be replaced by lattice basis reductions. Cute, but useless, since neither the shortest vector nor lattice basis reduction methods I used ran in polynomial time. So, as was the case for the traditional Berlekamp-Hensel approach, I did not even attempt to analyse the two different new lattice-based methods to factor polynomials over the rationals.

Initially, this disinterested attitude hardly changed when Hendrik got a letter from Laci that lattice basis reduction could be done in polynomial time. After all, at that time factoring polynomials over the rationals was so firmly established as something non-polynomial time, that it was hard to believe that the shortest vector trick would change anything – most certainly some other step would spoil the game. As it turned out, it didn't, as Hendrik suddenly realized that the smallest prime maintaining square-freeness can always be bounded in such a way that Berlekamp runs in polynomial time, deterministically.

And is the rest history? Not at all, unless one is just interested in polynomials over the rationals. We still do not have an unconditional deterministic polynomial time algorithm to factor polynomials over prime fields. And, more importantly, factoring reducible polynomials over the integers is still widely open, with the degree zero case the embarrassing inspiration for a popular cryptographic application, and no true progress since the late 1980s. I sincerely hope that we will see the rest in the not too distant future.

References

1. H.W. Lenstra, Jr., Euclidean number fields of large degree, University of Amsterdam, Report 76-09, May 1976.
2. P.S. Wang, Factoring polynomials over algebraic number fields, Math. Comp. **30** (1976), 324–336.
3. P.J. Weinberger, L.P. Rothschild, Factoring polynomials over algebraic number fields, ACM Transactions on Mathematical Software **2** (1976), 335–350.