In the previous lecture, we started the study of a complete factorization algorithm for polynomials over finite fields. The second stage of the algorithm is the distinct-degree factorization (DDF) process.

13.1 Distinct-degree factorization (DDF)

The following theorem plays a central role in this stage of the factoring process. It is due to Gauss, and was rediscover by many authors including Galois and Arwin (see [1, 5, 10, 16]). For a proof see [24], p. 91, Theorem 3.20).

Theorem 1 *For $i \geq 1$, the polynomial $x^{q^i} - x \in \mathbb{F}_q[x]$ is the product of all monic irreducible polynomials in $\mathbb{F}_q[x]$ whose degree divides i.*

This theorem immediately provides an algorithm for this stage. It suffices to successively compute x^{q^i} and take $\gcd(f, x^{q^i} - x)$, for $i = 1, 2 \ldots$. More precisely, we have the following algorithm.

```
procedure DDF(a : polynomial); [a squarefree]
  n := deg(a); g := a; h := x;
  for k := 1 to n do
1.      h := h^q mod g;
2.      b[k] := gcd(h-x,g);
3.      g := g/b[k]; [a w/o factors deg<=k]
4.      if b[k] <> 1 then h := h mod g fi;
  od;
  return(b[1].b[2]...b[n]);
end;
```

The computation in step 1 is done by means of the classical *repeated squaring* method. Let $\nu(q)$ be the number of ones in the binary representation of q. The number of products needed to compute $h^q \bmod g$ by this method is

$$\lambda(q) = \lfloor \log_2 q \rfloor + \nu(q) - 1.$$

By the results of last lecture on the first stage of the factoring algorithm, we consider only the cost of DDF applied to the squarefree part of the input polynomial f. Our subsequent analyses are all relative to the statistics induced by a random input f of degree n.

We provide a precise comparison of three strategies for the DDF phase: the basic rule, the "half-degree" rule and the "early abort" rule, to be explained later. The costs are found to be in the approximate proportion $1 :: \frac{3}{4} :: \frac{2}{3}$. A saving of about one third results from controlling the DDF phase by the early abort strategy.

We start with the basic strategy, that is, the one in the above algorithm with the for command running from 1 to n.

Theorem 2 *The expected cost of the basic DDF phase satisfies*

$$\overline{DDF}_n \sim \frac{5}{12}\,(\lambda(q)\tau_1 + \tau_2)\,n^3.$$

Proof. The cost of the basic DDF is $\sum_{j=1}^{4} C_j$, where C_j is the cost of line j. Let $\overline{C_j}$ be the expectation of C_j.

Since the mean number of factors of f is $O(\log n)$, $\overline{C_3} + \overline{C_4} = O(n^2 \log n)$. Thus, we can concentrate on the first two steps of the algorithm.

Let d_k denote the degree of polynomial g when the kth iteration starts; the parameter d_k is also the sum of the degrees of the distinct factors of f with degree $\geq k$. Then, the quantity $C_1 + C_2$ is equal to $(\lambda(q)\tau_1 + \tau_2)\sum_{k\geq 1} d_k^2$.

The bivariate generating function associated with d_k is

$$P_k(z,u) = \prod_{j<k}\left(\frac{1}{1-z^j}\right)^{I_j} \prod_{j\geq k}\left(1 + u^j \frac{z^j}{1-z^j}\right)^{I_j}.$$

The expected value of $C = \sum_{k=1}^{n} d_k^2$ is then given by $\overline{C} = \frac{1}{q^n}[z^n]Q(z)$, where

$$Q(z) = \sum_{k=1}^{n}\left(\frac{\partial^2 P_k(z,u)}{\partial u^2} + \frac{\partial P_k(z,u)}{\partial u}\right)_{u=1}.$$

Singularity analysis entails that $[z^n]Q(z) \sim \frac{5}{12} q^n\, n^3$.

Q.E.D.

A natural idea for improving the algorithm is to stop the DDF loop when $k = n/2$: then, the remaining factor is either 1 or irreducible. We call this the *half-degree* rule. We can now state the following theorem.

Theorem 3 *The expected cost of the DDF phase using the half-degree rule satisfies*

$$\overline{DDF}_n^{(HD)} \sim \frac{5}{16}\,(\lambda(q)\tau_1 + \tau_2)\,n^3.$$

Proof. The cost is now given by $(\lambda(q)\tau_1 + \tau_2)C^{(1)}$, where $C^{(1)} = \sum_{k\leq n/2} d_k^2$.

Let D_1 be the highest degree of all irreducible factors of f. We study the difference $C^{(2)} = C - C^{(1)}$. If $D_1 \leq n/2$, we have $C^{(2)} = 0$, otherwise we have $C^{(2)} = (D_1 - \lfloor n/2 \rfloor)\, D_1^2$ since there can be only one factor of degree larger than $n/2$, namely D_1. Thus the mean value of $C^{(2)}$ is given by

$$\overline{C^{(2)}} = \sum_{n/2<k\leq n} \Pr(D_1 = k) \left(k - \left\lfloor \frac{n}{2} \right\rfloor\right) k^2.$$

The probability $\Pr(D_1 = k)$ is derived from the generating function $\chi_k(z)$ of polynomials whose factors have all degree $\leq k$, that is,

$$\chi_k(z) = \prod_{j=1}^{k} \left(\frac{1}{1-z^j}\right)^{I_j},$$

as

$$\Pr(D_1 = k) = \frac{1}{q^n}[z^n]\left(\chi_k(z) - \chi_{k-1}(z)\right).$$

Note that since $\chi_k(z)$ is the generating function of polynomials with factors of degree $\leq k$, the expression $\chi_k(z) - \chi_{k-1}(z)$ describes exactly those polynomials that have at least one factor of degree k. When $k > n/2$, the nth coefficient of $\chi_k(z) - \chi_{k-1}(z)$ is obtained from the relation

$$\begin{aligned}
\chi_k(z) - \chi_{k-1}(z) &= P(z)\left(1 - (1-z^k)^{I_k}\right) \prod_{j>k}(1-z^j)^{I_j} \\
&= P(z)\left(I_k z^k + O(z^{n+1})\right).
\end{aligned}$$

For $n/2 < k \leq n$, $\Pr(D_1 = k) = I_k/q^k \sim 1/k$. Thus, $\overline{C^{(2)}} \sim \frac{5}{48} n^3$, and finally

$$\overline{C^{(1)}} = \overline{C} - \overline{C^{(2)}} \sim \frac{15}{48} n^3.$$

Asymptotically, the half-degree rule results in a saving of 25%.

Q.E.D.

An additional way of improving the performance of the algorithm is using the *early abort* strategy. The early abort strategy consists of stopping the main loop of DDF when $2k$ exceeds the degree of the remaining factor. The proof of this rule is more technical than the previous rules (see [7]). The analysis has some analogy to that of integer factoring given by Knuth and Trabb-Pardo [23].

Theorem 4 *The expected cost of the DDF phase using the early-abort rule satisfies*

$$\overline{DDF}_n^{(EA)} \sim \delta\left(\lambda(q)\tau_1 + \tau_2\right) n^3,$$

where

$$\begin{aligned}
\delta &= \frac{5}{12} - \frac{1}{3}\int_0^\infty e^{-2x} \exp\left(-\int_x^\infty \frac{e^{-u}}{u}\, du\right) \left[3e^{-x}(2+3x+2x^2) + 2x^3 - 3x + 6\right] \frac{dx}{x} \\
&= 0.26688798\ldots.
\end{aligned}$$

Proof. Let D_1 and D_2 be the degrees of the two irreducible factors of f of highest degree, setting $D_2 = 0$ if **a** is irreducible. The iteration is now aborted at step $k = \max\{\lfloor D_1/2 \rfloor, D_2\} + 1$. Let $C^{(3)} = \sum_{k \le \max\{\lfloor D_1/2 \rfloor, D_2\}} d_k^2$. The cost of DDF with this stopping rule becomes

$$(\lambda(q)\tau_1 + \tau_2) C^{(3)}.$$

Consider the difference $C^{(4)} = C - C^{(3)}$. We have

$$C^{(4)} = \begin{cases} (D_1 - \lfloor D_1/2 \rfloor) D_1^2 & \text{if } D_1/2 > D_2, \\ (D_1 - D_2) D_1^2 & \text{if } D_1/2 \le D_2. \end{cases}$$

The generating function $\phi_{D_1}(z)$ of polynomials for which $D_1 > 2D_2$ is given by

$$\left[\prod_{1 \le \ell < D_1/2} \left(\frac{1}{1 - z^\ell} \right)^{I_\ell} \right] I_{D_1} \left(\frac{z^{D_1}}{1 - z^{D_1}} \right).$$

The generating function $\psi_{D_1,D_2}(z)$ of polynomials for which $D_2 < D_1 \le 2D_2$ is given by

$$\left[\prod_{1 \le \ell < D_2} \left(\frac{1}{1 - z^\ell} \right)^{I_\ell} \right] \left[\left(\frac{1}{1 - z^{D_2}} \right)^{I_{D_2}} - 1 \right] \left[I_{D_1} \frac{z^{D_1}}{1 - z^{D_1}} \right].$$

Hence, the generating function $\Phi(z)$ of the cumulated values of the parameter $C^{(4)}$ is

$$\sum_{D_1} \left(D_1 - \left\lfloor \frac{D_1}{2} \right\rfloor \right) D_1^2 \, \phi_{D_1}(z) + \sum_{D_1 = D_2 + 1}^{2D_2} (D_1 - D_2) D_1^2 \, \psi_{D_1,D_2}(z).$$

The analysis of this generating function near its positive dominant singularity $1/q$ is done by approximating sums with integrals (Euler-Maclaurin summation) after the change of variables $z = e^{-t} q^{-1}$. A delicate analysis shows that $\Phi(z/q) \sim c_0 (1 - z)^{-4}$ as $z \to 1^-$, where $c_0 = \frac{5}{2} - 6\delta$. A Tauberian theorem is needed to complete the argument since the positive singularity is not isolated.

Alternatively, this result can be obtained estimating the Cauchy integral of the generating function using a particular range of integration (see [7, 30]).

Q.E.D.

The value of the constant δ in Theorem 4 was confirmed by running $5 \cdot 10^6$ simulations on permutations of size $10,000$ which gave the empirical value $\delta \approx 0.2668897$.

The global saving of the early abort rule is of 36%, and the expected cost of $O(\log q \cdot n^3)$ for DDF clearly dominates the whole factorization chain.

DDF does not completely factor a polynomial that has different factors of same degree. The following theorem determines the chances that this happens.

Theorem 5 *(i) The probability that DDF yields the complete factorization is asymptotic to*

$$c_q = \prod_{k \geq 1} \left(1 + \frac{I_k}{q^k - 1}\right)(1 - q^{-k})^{I_k},$$

$c_2 \doteq 0.6656$, $c_{257} \doteq 0.5618$, $c_\infty = e^{-\gamma} \doteq 0.5614$.

(ii) The number of degree values for which there is more than one irreducible factor in the polynomial produced by DDF has an average that is asymptotic to the constant

$$\sum_{k \geq 1} (1 - q^{-k})^{I_k} \left((1 - q^{-k})^{-I_k} - 1 - \frac{I_k q^{-k}}{1 - q^{-k}}\right).$$

(iii) The degree of the part of the polynomial that remains to be factored by the EDF algorithm has expectation $\log n + O(1)$*, and standard deviation of approximately* $\sqrt{n}$*.*

Proof. (Sketch)

(i) The generating function of polynomials with irreducible factors of distinct degrees is

$$\prod_{k \geq 1} \left(1 + I_k \frac{z^k}{1 - z^k}\right).$$

(ii) The bivariate generating function of the number of degree values for which there are repeated elements is

$$\prod_{k \geq 1} \left(1 + \frac{I_k z^k}{1 - z^k} + u\left((1 - z^k)^{-I_k} - 1 - \frac{I_k z^k}{1 - z^k}\right)\right).$$

(iii) The bivariate generating function associated to the total degree of the nontrivial part of DDF is

$$\prod_{k \geq 1} \left[\left(1 + u^k \frac{z^k}{1 - z^k}\right)^{I_k} - (u^k - 1) I_k \frac{z^k}{1 - z^k}\right].$$

Q.E.D.

13.2 Equal-degree factorization (EDF)

After the first two stages of the general algorithm, the factorization problem has been eventually reduced to factoring a collection of monic squarefree polynomials b_k all of whose irreducible factors have the same (known) degree k. The third step in the factorization process, the equal-degree factorization algorithm (EDF), focuses on polynomials with this special form. Our reference for this stage is the classical Cantor-Zassenhaus algorithm [5] (see also [14]).

The EDF algorithm is described below, and we briefly recall its principle here. For convenience, we assume that q is odd (a similar treatment can be done when the characteristic is 2). Let b be a polynomial that is a product of j irreducible factors $f_1, \ldots, f_j$, each of degree k. The Chinese remainder theorem implies the ring homomorphism,

$$\mathbb{F}_q[x]/(b) \cong \mathbb{F}_q[x]/(f_1) \times \cdots \times \mathbb{F}_q[x]/(f_j),$$

and a random element h of $\mathbb{F}_q[x]/(b)$ is associated to a j-tuple $(h_1, \ldots, h_j)$, where each h_i is a random element of $\mathbb{F}_q[x]/(f_i)$.

Now comes the splitting principle that eventually makes it possible to isolate the various f_i. Since each f_i is irreducible, the multiplicative group of each component $\mathbb{F}_q[x]/(f_i)$ is a field isomorphic to $\mathbb{F}_{q^k}$. Such a group being cyclic, there are the same number $(q^k - 1)/2$ of squares and nonsquares. The test $h_i^{(q^k-1)/2} = 1$ discriminates the squares in this multiplicative group. Thus, taking a random h and computing $a := h^{(q^k-1)/2} - 1 \bmod b$, we have that $\gcd(a, b)$ "extracts" the product of all the f_i for which h is a square in $\mathbb{F}_q[x]/(f_i)$.

```
procedure EDF(b : polynomial, k : integer);
[b is a product of irreducibles of degree k]
        if degree(b) <= k then return(b) fi;
        h := randpoly(degree(b)-1);
1.      a := h^((q^k-1)/2)-1 mod b;
2.      d := gcd(a,b);
        return(EDF(d,k).EDF(b/d,k));
end;
```

From the probabilistic point of view, a component h_i that is random in $\mathbb{F}_q[x]/(f_i)$ has probability $\alpha = \frac{1}{2} - \frac{1}{2q}$ of being discriminated by the gcd test and the dual probability, $\beta = \frac{1}{2} + \frac{1}{2q}$ of being a nonsquare. The (small) difference between α and β is accounted for by the possibility of having noninvertible components.

In summary, irreducible factors of each degree can be extracted successively. For each degree where two or more such factors occur, the recursive splitting process will be launched. Then, the analysis of the complete EDF phase requires a purely combinatorial analysis of what takes place at each degree combined with an estimate of the probability that there are j irreducible factors of degree k in a random polynomial of degree n. These probabilities give interesting information on random polynomials and have been obtained by Knopfmacher and Knopfmacher [21] whose results we now recall.

Theorem 6 (Knopfmacher and Knopfmacher) *The probability that there are j distinct irreducible factors of degree k in a random polynomial of degree n is, for n large enough*

($n \geq kI_k$), given by the binomial distribution $\mathcal{B}(I_k, q^{-k})$, namely,

$$\Pr\{\kappa_n(k) = j\} = \binom{I_k}{j}(q^{-k})^j(1-q^{-k})^{I_k-j}.$$

As q becomes large, the binomial probability distribution converges to a Poisson law of parameter $1/k$,

$$\Pr\{\kappa_n(k) = j\} = e^{-1/k}\frac{k^{-j}}{j!},$$

in accordance with the known distribution of cycle lengths in the random permutation model [34].

The analysis of the EDF stage combines a recursive partitioning problem akin to digital trees —also known as "tries" [22]— together with estimates on the degrees of irreducible factors of random polynomials given by the previous theorem. The net result is that the global cost of EDF is quadratic, a sharp contrast with the cubic cost of DDF. For the proof of the theorems, see [7].

Theorem 7 *The expected cost of the EDF algorithm applied to any product of j irreducible factors of degree k is*

$$E_{j,k} = \frac{1}{2\alpha\beta}j(j-1) + (\mu_k\tau_1 + \tau_2)k^2 \cdot j \sum_{m\geq 0}\sum_{\ell=0}^{m}\binom{m}{\ell}\alpha^{m-\ell}\beta^{\ell}\left(1-(1-\alpha^{m-\ell}\beta^{\ell})^{j-1}\right),$$

where $\mu_k = C_{(q^k-1)/2} = \left\lfloor \log_2 \frac{q^k-1}{2}\right\rfloor + \nu\left(\frac{q^k-1}{2}\right) - 1.$

Finally, the main theorem of this stage of the factorization process is obtained.

Theorem 8 *The expected cost of the EDF phase satisfies*

$$\overline{EDF}_n \sim \frac{\tau_1}{\alpha\beta}\sum_{k=1}^{\lceil n/2\rceil}\mu_k,$$

where $\mu_k = \left\lfloor \log_2 \frac{q^k-1}{2}\right\rfloor + \nu\left(\frac{q^k-1}{2}\right) - 1$. *In addition, this cost is $O(n^2)$, and for $-\frac{1}{3} \leq \xi_n \leq \frac{1}{3}$,*

$$\overline{EDF}_n \sim \left(\frac{3}{4}\tau_1\frac{q^2}{q^2-1}\log_2 q \cdot n^2\right)(1 + \xi_n + o(1)).$$

The EDF step analyzed in [7] is the one in [5]. The versions in [4] and [2] could also be considered. These algorithms are also probabilistic. We point out that a well-known open problem is to find a polynomial time algorithm for the factorization problem.

13.3 Summary of results

- The expected cost of the *ERF* phase is dominated by a single gcd between the polynomial f to be factored and its derivative f', so that it is $O(n^2)$ on average. The non-squarefree part has average degree $O(1)$.
- The second phase *DDF* is the one that is most intensive computationally. We analyze three strategies: the basic rule, the "half-degree" rule and the "early abort" rule whose costs are found to be in the approximate proportion $1 :: \frac{3}{4} :: \frac{2}{3}$. Cost: $O(n^3 \log q)$.
- The third phase *EDF* is executed less than 50% of the times and its cost is small compared to that of DDF. Cost: $O(n^2 \log q)$.

The results for the worst and average-case analysis of the three steps of the algorithm are summarized in the table below.

Step	Worst-case	Average-case
ERF	$O(n^2)$	$\tau_2 n^2$
DDF	$O(n^3 \log q)$	$0.26688798\ldots\ (\lambda(q)\tau_1 + \tau_2)\, n^3$
EDF	$O(n^3 \log q)$	$\left(\frac{3}{4}\tau_1 \frac{q^2}{q^2-1} \log q \cdot n^2\right)\ \left(\frac{4}{3} + o(1)\right)$

The results on this table are done under classical arithmetic. The extension of these results to FFT based arithmetic is needed to encompass the fastest time algorithms for the problem.

13.4 Analysis of intervals

Factorization algorithms of the 1990's involve the analysis of irreducible factors whose degrees lie in intervals. Given a partition of $[1, n]$ into intervals, interesting parameters are:

- probability of no *multi-factor* interval, i.e., an interval with more than one irreducible factor,
- average number of multi-factor intervals,
- average number of irreducible factors in multi-factor intervals,
- average number of total degree of irreducible factors in multi-factor intervals, and so on.

The proofs of these theorems are based on the methodology of the course (work in progress).

Interval information may not only help in the average-case analysis but it can also lead to a better partition than in [15] and [18]. Further research on the area might translate into an algorithm with improved average-case cost.

13.5 Irreducibility tests for polynomials

For a prime power q and an integer $n \geq 2$, let $\mathbb{F}_{q^n}$ be the extension of degree n of $\mathbb{F}_q$. A fundamental problem in finite fields is the construction of extension fields.

Extensions of finite fields (especially $\mathbb{F}_2$) are important in implementing cryptosystems and error correcting codes.

There are two general ways of constructing extensions of finite fields:

1. via an irreducible polynomial over the ground field with degree equal to the degree of the extension, or
2. via a normal element $\alpha \in \mathbb{F}_{q^n}$, i.e., when $(\alpha, \alpha^q, \ldots, \alpha^{q^{n-1}})$ is a basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.

Normal bases are useful when computing exponentiations. Indeed, the computation of qth powers can be done with a cyclic shift of the coefficients

$$\left(\sum_{i=0}^{n-1} a_i \alpha^{q^i}\right)^q = \sum_{i=0}^{n-1} a_i^q \alpha^{q^{i+1}} = \sum_{i=0}^{n-2} a_i \alpha^{q^{i+1}} + a_{n-1}\alpha,$$

since $\alpha^{q^n} = \alpha$. However, polynomial basis are preferable when operations like multiplications and divisions are needed.

The construction of polynomial basis is based on the isomorphism

$$\mathbb{F}_{q^n} \cong \mathbb{F}_q[x]/(f)$$

where f is an irreducible polynomial with degree n, the degree of the extension. Therefore, finding irreducible polynomials is a central problem in finite fields.

A probabilistic algorithm for constructing irreducible polynomials is presented in Rabin [33]. The central idea is to take polynomials at random and test them for irreducibility. The probability of a polynomial of degree n (over any finite field) being irreducible is close to $1/n$, as we have seen in the previous lecture. Therefore, if we pick n polynomials at random, we expect one of them to be irreducible.

We need an irreducibility test! The following theorem provides an irreducibility test.

Theorem 9 (Rabin) *Let $p_1, \ldots, p_k$ be all the prime divisors of n, and denote $n/p_i = n_i$, for $1 \leq i \leq k$. A polynomial $f \in \mathbb{F}_q[x]$ of degree n is irreducible in $\mathbb{F}_q[x]$ if and only if gcd $(f, x^{q^{n_i}} - x \bmod f) = 1$, for $1 \leq i \leq k$, and f divides $x^{q^n} - x$.*

Since $x^{q^{n_i}} - x \in \mathbb{F}_q[x]$ is the product of all monic irreducible polynomials over $\mathbb{F}_q$ of degree dividing n_i, each gcd tests the existence of irreducible factors in f of several degrees. This makes the analysis somehow technical (see [32]). A Tauberian argument is needed to derive the asymptotic results.

Ben-Or [2] propose an algorithm that at first sight seems naive. The algorithm is based on the same ideas of the distinct-degree factorization algorithm, that is, in Theorem 1. Ben-Or's algorithm tests the irreducibility of a polynomial by searching for irreducible factors degree by degree. Therefore, one might expect to be more expensive than Rabin's method.

This is incorrect in an average-case sense. Indeed, the mean number of irreducible factors of degree k of a random polynomial of degree n approaches $1/k$ as n tends to infinity [21]. Consequently, for $k = 1$ this number is very close to 1, which means that most reducible polynomials have a factor of degree 1. Thus, in most cases it suffices to do only one test (i.e., for $i = 1$) in order to reject a reducible polynomial. So, the gcds involved in Ben-Or's algorithm are less expensive then in Rabin's algorithm. Remember that we are using a trial and error algorithm to find an irreducible polynomial, thus we are picking at random the polynomials to be tested. Intuitively, this justifies the efficiency of Ben-Or algorithm.

The polynomials that are hard for the algorithm are polynomials of degree n containing no irreducible factors of degree up to certain value m (such polynomials are sometimes called *m-rough*). In particular, the smallest degree explains the expected moment where we stop the process of checking factors degree by degree.

Previous results on rough polynomials:

- Car [6]: Buchstab function for large m, say $m > c\,\frac{n \log\log n}{\log n}$.
- Gao & Panario [11]: $e^{-\gamma}/m$ for small m, $m < c \log n$.

Panario & Richmond [31] give a precise study of the probability that a random polynomial be m-rough, $1 \le m \le n$, in terms of the Buchstab function when $m \to \infty$, and singularity analysis when m is fixed.

The *Buchstab* function is the unique continuous solution of the difference-differential equation

$$\begin{aligned} u\omega(u) &= 1 && 1 \le u \le 2, \\ (u\omega(u))' &= \omega(u-1) && u > 2. \end{aligned}$$

The asymptotic behavior of the Buchstab function is well–known (see [37], §6.3, p. 404). In particular,

$$\lim_{u\to\infty} \omega(u) \quad = \quad e^{-\gamma}.$$

Figure 1 shows the relation between $\omega(u)$ and $e^{-\gamma}$.

Theorem 10 *The smallest degree S_n among the irreducible factors of a random polynomial of degree n over $\mathbb{F}_q$ satisfies, when $m \to \infty$ with n,*

$$Pr(S_n \ge m) = \begin{cases} \frac{1}{m}\,\omega\left(\frac{n}{m}\right) + O\left(\frac{1}{m^2}\right) & \textit{if } m = O(\frac{n}{\log n}), \\ \frac{1}{m}\,\omega\left(\frac{n}{m}\right) + O\left(\frac{\log n}{mn}\right) & \textit{otherwise,} \end{cases}$$

where ω is the Buchstab function.

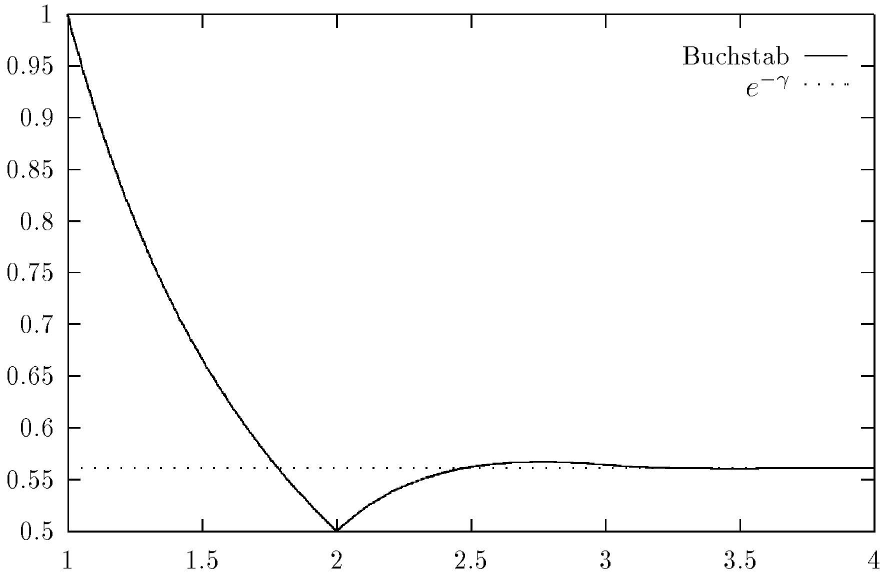

Figure 1: The relation between the Buchstab function and $e^{-\gamma}$ in the interval $[1,4]$.

The summation of the above probabilities give the expectation of the smallest degree among the irreducible factors of a random polynomial.

Theorem 11 *The expectation of the smallest degree among the irreducible factors of a random polynomial satisfies:*

$$E(S_n) = e^{-\gamma} \log n + O(h(n)),$$

where $h(n)$ is any function going to infinity with n, γ is the Euler's constant, and $e^{-\gamma} = 0.56416\ldots$.

For the variance we have

$$Var(S_n) = C\, n\, (1 + o(1)),$$

with

$$C = 2 \int_2^{\infty} \frac{\omega(u)}{u^2}\, du = 0.556816\ldots,$$

where ω is the Buchstab function.

The expected smallest degree of the factors of a random polynomial directly provides the average-case analysis of Ben-Or's algorithm [31].

Theorem 12 *The expected cost of Ben-Or's algorithm to test a random polynomial of degree n over $\mathbb{F}_q$ is asymptotic to*

$$\begin{cases} e^{-\gamma} \log n\, (C_q \tau_1 + \tau_2 \log n)\, n \log n \log\log n\, (1 + o(1)), \\ e^{-\gamma} \log n\, (C_q \tau_1 + \tau_2)\, n^2\, (1 + o(1)), \end{cases}$$

for FFT and classical arithmetic, respectively.

For some polynomial factorization algorithms (see [12]) it is important to know the chances that a polynomial be irreducible when it has no factors of degree smaller than certain bound. The following immediate corollary provides such a probability.

Corollary 1 *The probability that a polynomial of degree n over $\mathbb{F}_q$ be irreducible if it has no factors of degree less than or equal to m is asymptotic to*

$$\begin{cases} e^{\gamma}\frac{m}{n} & m = o(n), \\ \frac{c}{\omega(1/c)} & m = c\,n,\ 0 < c \leq 1/2, \end{cases}$$

where $e^{\gamma} = 1.781\ldots$.

These studies generalize to the size of smallest components in random decomposable structures in the exp–log class (as defined by Flajolet & Soria [9]). This class includes permutations, polynomials over finite fields, 2-regular graphs, random mappings, arithmetical semigroups, among other classes of combinatorial objects. The results include expectation, variance and other moments of the rth smallest size, among other studies (work in progress).

14 Discrete logarithm problem

Let α be a generator of the nonzero elements of $\mathbb{F}_q$. For any element $b \in \mathbb{F}_q$, $b \neq 0$, there exists an integer x, $0 \leq x \leq q-2$, such that $b = \alpha^x$. We call x the *discrete logarithm* of b in the base α.

Discrete logarithm problem: find a computationally feasible algorithm to compute the discrete logarithm of b for any $b \in \mathbb{F}_q$, $b \neq 0$.

We restrict ourselves to the case of discrete logarithm problem over $\mathbb{F}_{2^n}$. We view the elements in $\mathbb{F}_{2^n}$ as polynomials of degree n over $\mathbb{F}_2$. Many public-key cryptosystems base their security on the (supposed) intractability of the discrete logarithm problem (for large q).

We present here the *index calculus algorithm*. This method consists of two parts: a construction of a large database of logarithms, and the computation of individual logarithms.

Index calculus algorithm.

Step 1: find the logarithms of all irreducible polynomials of degree at most m, where m is a fixed positive integer.

Step 2: for computing the logarithm of an element $g \in \mathbb{F}_{2^n}$, $g \neq 0$, take a random integer

a, and compute $h = g \cdot \alpha^a$, where α generates $\mathbb{F}_{2^n}$. Factor h in

$$h = \prod_{i=1}^{t} p_i^{e_i}.$$

If each irreducible factor p_i has deg $p_i \leq m$, then

$$\log g = \sum_{i=1}^{t} e_i \log p_i - a,$$

which can be easily evaluated by looking up $\log p_i, 1 \leq i \leq t$ in the database. If not all p_i have degree $\leq b$, then generate another integer a and repeat.

The probability that a polynomial have all its irreducible factors of degree smaller than certain bound m gives useful information on the feasibility of computing discrete logarithms with this method. Polynomials with all irreducible factors with degree smaller than or equal to m are called *m-smooth*. Clearly, we have that the generating function of m-smooth polynomials is

$$f_m(z) = \prod_{k=1}^{m} (1 - z^k)^{-I_k}.$$

Using the saddle point method (see [8, 29]), Odlyzko [28] prove the following result for $q = 2$ (the generalization for any q is in [25]).

Theorem 13 *Let $\mathbb{F}_q$ be fixed, $f_m(z) = \prod_{k=1}^{m}(1 - z^k)^{-I_k}$, and $r_0 = r_0(n, m)$ be the unique solution in $(0, 1)$ of the equation*

$$r_0 \frac{f_m{}'}{f_m}(r_0) = n,$$

and let

$$b(r_0) = \left(\frac{f_m{}'}{f_m}(r) \right)' \Bigg|_{r=r_0}.$$

Then, for

$$n^{1/100} \leq m \leq n^{99/100},$$

as $n \to \infty$,

$$[z^n] f_m(z) = (1 + o(1)) \frac{f_m(r_0) r_0^{-n}}{\sqrt{2\pi b(r_0)}}.$$

Soundararajan [35] completed the full range of m estimating $[z^n]f_m(z)$ using recurrences relations. This could be done using partial fraction expansions for $1 \leq m \leq (\log n)(\log \log n)^{-1}$, and singularity analysis for $n \log \log n (\log n)^{-1}$. He also established the precise bounds

$$(\log n)(\log \log n)^{-1} \leq m \leq n \log \log n (\log n)^{-1},$$

for the range of usage of the saddle point method.

14.1 Conclusion

14.1.1 Final Remarks

We commented on a methodology for counting random polynomials over finite fields based on generating functions and asymptotic analysis that provides precise average-case analysis of polynomial algorithms.

Other related polynomial problems are:

- smooth polynomials;
- finding roots;
- gcd of polynomials; etc.

Other connected problems are:

- study of random decomposable combinatorial structures (size of smallest and largest components);
- study of number theoretic functions via analytic combinatorics (Buchstab; Dickman; Renyi; a class of functions defined by Hildebrand & Tenenbaum [17]).

14.1.2 Further algebraic problems

There are many problems that have not been successfully analyzed in the sense of our course. It may be possible to obtain information on the behavior of such algorithms with the techniques we have developed so far. They remain as interesting open problems.

- Linear algebraic based problems (Berlekamp's [3] and Niederreiter's [26, 27] factorization algorithms);
- polynomials in several variables (irreducibility test, factorization, and so on);
- analysis of multiplication algorithms (Karatsuba's [19, 20] and Strassen's [36] algorithms);
- construction of normal elements [13];
- analysis of algorithms for polynomial factorization over the integers, and discrete logarithm problem,

and many other problems!

References

[1] A. Arwin. Über Kongruenzen von dem fünften und höheren Graden nach einem Primzahlmodulus. *Arkiv för matematik, astronomi o. fysik*, 14:1–46, 1918.

[2] M. Ben-Or. Probabilistic algorithms in finite fields. In *Proc. 22nd IEEE Symp. Foundations Computer Science*, pages 394–398, 1981.

[3] E.R. Berlekamp. *Algebraic coding theory.* McGraw Hill, New York NY, 1968.

[4] E.R. Berlekamp. Factoring polynomials over large finite fields. *Math. Comp.*, 24:713–735, 1970.

[5] D.G. Cantor and H. Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Math. Comp.*, 36:587–592, 1981.

[6] M. Car. Théorèmes de densité dans $\mathbb{F}_q[x]$. *Acta Arith.*, 48:145–165, 1987.

[7] P. Flajolet, X. Gourdon, and D. Panario. The complete analysis of a polynomial factorization algorithm over finite fields. Submitted March 1998. [Extended abstract in *Proc. 23rd ICALP Symp.*, Lecture Notes in Computer Science, vol. 1099, p. 232–243, 1996.] Full version in technical report 3370, INRIA, March 1998.

[8] P. Flajolet and R. Sedgewick. The average case analysis of algorithms: saddle point asymptotics. Technical Report 2376, INRIA, Rocquencourt, Le Chesnay, France, 1994.

[9] P. Flajolet and M. Soria. Gaussian limiting distributions for the number of components in combinatorial structures. *Journal of Combinatorial Theory,* Series A, 53:165–182, 1990.

[10] É. Galois. Sur la théorie des nombres. In R. Bourgne and J.P. Arza, editors, *Écrits et mémoires d'Évariste Galois*, pages 112–128. Gauthier-Villars, 1830.

[11] S. Gao and D. Panario. Tests and constructions of irreducible polynomials over finite fields. In F. Cucker and M. Shub, editors, *Foundations of Computational Mathematics*, pages 346–361. Springer Verlag, 1997.

[12] J. von zur Gathen and J. Gerhard. Arithmetic and factorization of polynomials over $\mathbb{F}_2$. In Lakshman Y.N., editor, *Proc. ISSAC'96, Zürich, Switzerland*, pages 1–9. ACM press, 1996.

[13] J. von zur Gathen and M. Giesbrecht. Constructing normal bases in finite fields. *J. Symb. Comp.*, 10:547–570, 1990.

[14] J. von zur Gathen and D. Panario. A survey on factoring polynomials over finite fields. To appear in the special issue of the MAGMA conference in *J. Symb. Comp.*, 1999.

[15] J. von zur Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. *Comput complexity*, 2:187–224, 1992.

[16] C.F. Gauss. *Untersuchungen úber Höhere Mathematik.* Chelsea, New York, 1889.

[17] A. Hildebrand and G. Tenenbaum. On a class of differential-difference equations arising in number theory. *J. D'Analyse Mathématique*, 61:145–179, 1993.

[18] E. Kaltofen and V. Shoup. Subquadratic-time factoring of polynomials over finite fields. In *Proc. 27th ACM Symp. Theory of Computing*, pages 398–406, 1995.

[19] A. Karatsuba. The complexity of computations. In *Proc. Steklov Inst. Math.*, volume 211, pages 169–183, 1995.

[20] A. Karatsuba and Y. Ofman. Умножение многозначных чисел на абто-матах. *Dokl. Akad. Nauk USSR*, 145:293–294, 1962. English translation: Multiplication of multidigit numbers on automata, Soviet Physics–Doklady **7** (1963), 595-596.

[21] J. Knopfmacher and A. Knopfmacher. Counting irreducible factors of polynomials over a finite field. *SIAM Journal on Discrete Mathematics*, 112:103–118, 1993.

[22] D.E. Knuth. *The art of computer programming, vol.3: sorting and searching.* Addison-Wesley, Reading MA, 2 edition, 1998.

[23] D.E. Knuth and L. Trabb-Pardo. Analysis of a simple factorization algorithm. *Theor. Computer Science*, 3:321–348, 1976.

[24] R. Lidl and H. Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its Applications.* Addison-Wesley, 1983.

[25] R. Lovorn. *Rigourous, subexponential algorithmsfor discrete logarithm algorithms in* $\mathbb{F}_{p^2}$. PhD thesis, University of Georgia, 1992.

[26] H. Niederreiter. Factorization of polynomials and some linear-algebra problems over finite fields. *Lin. Alg. Appl.*, 192:301–328, 1993.

[27] H. Niederreiter. A new efficient factorization algorithm for polynomials over small finite fields. *Appl. Alg. Eng. Comm. Comp.*, 4:81–87, 1993.

[28] A. Odlyzko. Discrete logarithms and their cryptographic significance. In *Advances in Cryptology, Proceedings of Eurocrypt 1984*, volume 209 of *Lecture Notes in Computer Science*, pages 224–314. Springer-Verlag, 1985.

[29] A. Odlyzko. Asymptotic enumeration methods. In R. Graham, M. Grötschel, and L. Lovász, editors, *Handbook of Combinatorics*, volume 2, pages 1063–1229. Elsevier, 1995.

[30] D. Panario, X. Gourdon, and P. Flajolet. An analytic approach to smooth polynomials over finite fields. In J. P. Buhler, editor, *Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*, pages 226–236. Springer-Verlag, 1998. Proceedings of ANTS III.

[31] D. Panario and B. Richmond. Analysis of Ben-Or's polynomial irreducibility test. *Random Struct. Alg.*, 13:439–456, 1998.

[32] D. Panario and A. Viola. Analysis of Rabin's irreducibility test for polynomials over finite fields. Submitted August 1998. [Extended abstract in *Proc. LATIN'98*, Lecture Notes in Computer Science, vol. 1380, p. 1–10, 1998.], 1998.

[33] M.O. Rabin. Probabilistic algorithms in finite fields. *SIAM J. Comp.*, 9:273–280, 1980.

[34] L.A. Shepp and S.P. Lloyd. Ordered cycle lengths in a random permutation. *Trans. Amer. Math. Soc.*, 121:340–357, 1966.

[35] K. Soundararajan. Asymptotic formulae for the counting function of smooth polynomials. To appear in J. London Math. Soc.

[36] V. Strassen. Gaussian elimination is not optimal. *Numer. Mathematik*, 13:354–356, 1969.

[37] G. Tenenbaum. *Introduction to analytic and probabilistic number theory.* Cambridge University Press, 1995.