AM 106/206: Applied Algebra **Prof. Salil Vadhan**

Lecture Notes 20

December 2, 2009

1 Finite Fields

- **Reading:** Gallian Ch. 22
- Recall: only possible sizes for finite fields are prime powers p^n.
- **Thm 22.1:** Existence of finite field of order p^n for every prime power p^n, unique up to isomorphism. Often denoted $\mathrm{GF}(p^n)$ for "Galois field" or $\mathbb{F}_{p^n}$.
- **Proof:** Let F be splitting field of $f(x) = x^{p^n} - x$ over $\mathbb{Z}_p$, $F' =$ roots of $f(x)$ in F.
  1. Claim 1: F' is a subfield of F (and hence $F' = F$ by def of splitting field).
  2. Claim 2: the roots of $f(x)$ are all distinct in F.
  3. Claim 3: every finite field of order p^n is a splitting field of $f(x)$.
- **Thm 22.2:** The additive group of $\mathrm{GF}(p^n)$ is isomorphic to $\mathbb{Z}_p^n$. The multiplicative group $\mathrm{GF}(p^n)^*$ is cyclic.
- **Proof:**
- **Corollaries:**
  1. For every n, there is an element of $\mathrm{GF}(p^n)$ of degree n over $\mathbb{Z}_p$.
  2. For every n, there is an irreducible polynomial of degree n in $\mathbb{Z}_p[x]$.
- Thus, instead of constructing $\mathrm{GF}(p^n)$ as a splitting field by adjoining several roots, we can take a *single* irreducible polynomial $f(x)$ of degree n and $\mathbb{Z}_p[x]/\langle f(x)\rangle \cong \mathrm{GF}(p^n)$.
- **Examples:**
  1. $\mathrm{GF}(7^3) \cong \mathbb{Z}_7[x]/\langle x^3 + 2\rangle$.
  2. $\mathrm{GF}(7^3) \cong \mathbb{Z}_7[x]/\langle x^3 + x^2 + 1\rangle$.

3. Adding and multiplying $x^2 + 5$ and $3x + 2$ in above representations of $\mathrm{GF}(7^3)$.

- **Thm:** $\mathrm{GF}(p^n)$ has a (unique) subfield isomorphic to $\mathrm{GF}(p^m)$ if (and only if) $m|n$.

- **Computational Issues:**
  - Computations in the finite field $\mathrm{GF}(p^n)$ can be done efficiently given the prime p and an irreducible polynomial $f(x)$ over $\mathbb{Z}_p$ of degree n.
  - How to find p and $f(x)$?
    1. Choose randomly and test for primality/irreducibility (which can be done in polynomial time). Primes and irreducible polynomials have noticeable density, so this doesn't take too many trials (see ps8).
    2. Use a small value of p (e.g. $p = 2$) and known explicit irreducible polynomials, e.g. $f(x) = x^{2 \cdot 3^\ell} + x^{3^\ell} + 1$.

2 Error-Correcting Codes

- **Goal:** encode data so that it can be recovered even after much of it has been corrupted.
- **Def:** An *code* is an injective mapping $\mathrm{Enc} : \Sigma^k \to \Sigma^n$ for some finite *alphabet* Σ, *message length* k and *block length* n.
- **Def:** For two strings $x, y \in \Sigma^n$, we define their *(relative) Hamming distance* to be

$$d(x, y) = \#\{i \in [n] : x_i \neq y_i\}/n.$$

- **Def:** A code Enc is δ-*error-correcting* if there is a *decoding function* $\mathrm{Dec} : \Sigma^n \to \Sigma^k$ such that for every message $m \in \Sigma^k$ and every received word $r \in \Sigma^n$ such that $d(r, \mathrm{Enc}(m)) \leq \delta$, we have $\mathrm{Dec}(r) = m$.
- **Prop:** A code Enc is δ-*error-correcting* if and only if its *minimum distance* $\min_{m \neq m'} d(\mathrm{Enc}(m), \mathrm{Enc}(m'))$ is greater than 2δ (provided $2\delta n$ is an integer, to avoid round-off errors).
- **Proof:**

- **Goals:** Construct error-correcting codes for arbitrarily large message lengths k and:
  1. Maximize the error-correcting distance δ (ideally constant independent of k, e.g. $\delta = .1$).
  2. Maximize the *rate* $\rho = k/n$ (ideally constant independent of k, e.g. $\rho = .1$).
  3. Minimize the *alphabet size* $|\Sigma|$ (ideally constant independent of k, e.g. $\Sigma = \{0, 1\}$).
  4. Have efficient (e.g. polynomial time or even linear time) encoding and decoding algorithms.

- **Reed-Solomon Code:** $\Sigma = \mathrm{GF}(q)$. View message $m = (m_0, \ldots, m_{k-1}) \in \mathrm{GF}(q)^k$ as coefficients of a polynomial $p_m(x) = \sum_{i=0}^{k-1} m_i x^i$. The encoding is $\mathrm{RS}(m) = (p_m(\alpha_1), \ldots, p_m(\alpha_n))$ where $\alpha_1, \ldots, \alpha_n$ are fixed distinct elements of $\mathrm{GF}(q)$. (Thus we need $q \geq n$.)

- **Prop:** The minimum distance of the Reed-Solomon code is $1 - (k-1)/n$.

- Thus, taking e.g. $k = n/2$, we have constant rate ($\rho = 1/2$) and constant minimum distance ($\geq 1/2$). The only downside is the nonconstant alphabet size, but this can be improved by combining Reed–Solomon codes with other codes (see ps8).

- **Efficient RS Decoding Algorithm:** Given n pairs $(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n) \in \mathrm{GF}(q) \times \mathrm{GF}(q)$ with $\alpha_1, \ldots, \alpha_n$ distinct, we want to find all polynomials f of degree at most $d = k - 1$ such that $f(\alpha_i) = \beta_i$ for at least $t = (1 - \delta)n$ values of i. We will show how to do this provided $t > 2\sqrt{dn}$.

- **Step 1:** Find a nonzero *bivariate* polynomial $Q(x, y)$ such that (a) $Q(\alpha_i, \beta_i) = 0$ for all i, and (b) the degree of Q in x is at most $\sqrt{dn}$ and the degree of Q in y is at most $\sqrt{n/d}$.

  - We are looking to find the coefficients c_{ij} of

$$Q(x, y) = \sum_{i=0}^{\sqrt{dn}} \sum_{j=0}^{\sqrt{n/d}} c_{ij} x^i y^j.$$

    Each constraint $Q(\alpha_i, \beta_i) = 0$ is a homogeneous linear constraint on the coefficients c_{ij}. Since there are $(\sqrt{dn} + 1)(\sqrt{n/d} + 1) > n$ coefficients c_{ij} and only n constraints, there exists a nonzero solution and we can find it by Gaussian elimination.

- **Step 2:** Factor Q into irreducibles, look for any factors of the form $y - f(x)$, and output all such f that appear.

  - Observe that if $f(x)$ is a polynomial of degree d such that $f(\alpha_i) = \beta_i$ for at least t values of i, then the *univariate* polynomial $P(x) = Q(x, f(x))$ has at least t roots (namely the values of α_i such that $f(\alpha_i) = \beta_i$). The degree of $P(x)$ is at most $\sqrt{dn} + d \cdot \sqrt{n/d} = 2\sqrt{dn}$. Since $t > 2\sqrt{dn}$, $P(x)$ must be the zero polynomial.
  - The fact that $Q(x, f(x)) = 0$ means that $f(x)$ is a root of Q, considering Q as polynomial in y with coefficients that are polynomials in y. That is, we consider $Q(x, y)$ as an element of the polynomial ring $R[y]$, where $R = \mathrm{GF}(q)[x]$. We know that for any integral domain R, if $g(y) \in R[y]$ has a root $\alpha \in R$, then $y - \alpha$ divides $g(y)$ in $R[y]$. Taking $\alpha = f(x)$,we get that $y - f(x)$ divides $Q(x, y)$. Thus it will appear when we factor $Q(x, y)$. (Multivariate polynomial rings $F[x, y]$ have unique factorization, and this factorization can be done in polynomial time for most common fields, including finite fields.)

- Reed–Solomon Codes and versions of the above decoding algorithm are widely used in practice, e.g. on CDs and in satellite communications.