Complexity Issues in Coding Theory

Steve Szabo
Eastern Kentucky University

Complexity Issues in Coding Theory

We will discuss a chapter from the Handbook on Coding Theory by Alexander Barg on the complexity issues in the theory of linear error correcting codes. The chapter deals with the complexity of algorithmic problems of constructing "good" codes as well as the encoding and decoding of these codes.

The focus of the chapter is on asymptotic behavior of the algorithms considered. Most of the algorithms discussed will have provable performance and complexity estimates.

When we speak complexity, the order will either be "easy" meaning polynomial in the code length n or "difficut" meaning exponential. The codes considered will mainly be linear block codes.

Conventions

Defenitons/Notions:

- **algorithmic problem** - includes a class of objects and a property the objects may have. The goal is to produce objects from the class that have this property
- **mass problem** - the class of instances of a particular algorithmic problem, the answer to which may be an object, a number or a simple yes/no.
- **decision problem** - a mass problem whose answer is a yes/no.
- **computation** - a way of solving a given algorithmic problem, formalized by an abstract computing device (machine).
- **time complexity (worst-case)** - Given A, an algorithm problem, is a function of l, defined as the maximal time of computation over all instances of length l.
- **space complexity** - Given A, an algorithm problem, the number of registers used in computation.

Conventions

Random Access Machine(RAM) - an abstract machine that performs certain boolean operations and additions and some flow control assumed to have an unlimited number of registers with instant access.

The RAM is used as opposed to the Turing machine mainly because of the ease of transforming algorithms for a RAM presents less difficulty than for a Turing Machine. Using the RAM seems realistic when the space complexity is not too large and the model fits into the main memory of the computer.

Each instruction of the RAM is simulated by a Turing Machine program. This way the time complexity of the two machines are related.

Background

E_q^n - the set of n length vectors over the q-ary alphabet. We consider this space a linear space over F_q.

A subset $C \subset E_q^n$ is called a **block code**. If in addition C is a k dimensional F_q-subspace, we say C is an $[n, k]$ **linear block code** or **linear code**. A **generator matrix** for an $[n, k]$ linear code C is a $k \times n$ matrix G whose row space is C. For a linear code C denote the dual space by $C^\perp$. Note $C^\perp$ is $[n, n-k]$ linear code which has a $n-k \times n$ generator matrix H. We call H the **parity check matrix** of C. Note: $GH^T = 0$.

$\mathcal{N} = \{1, \ldots, n\}$. For $W \subset \mathcal{N}$ and A a matrix with n columns, $A(W)$ denotes the submatrix of A formed by the columns of A indexed by W.

For C be a $[n, k]$ linear code given, a k-set $W \subset \mathcal{N}$ s.t. for generator matrix G of C, $G(W)$ is non-singular is a **information set**. The remaining $n-k$ coordinates are called a **check set**.

Algorithmic Problems

1. **Code Construction** - Given parameters q, n, k, d, find a generator matrix for a q-ary $[n, k, d]$ linear code.
2. **Encoding** - A linear code can be viewed as a linear injective mapping form E_q^k to E_q^n. The problem is to implement this mapping.
3. **Decoding** - The decoding map is $f : E_q^n \to C$ s.t. for $x \in E_q^n$, $f(x) \in C$ is chosen to be the closest codeword to x. If for certain x, there is more than one value of $f(x)$ that satisfies this, an arbitrary value is chosen. This is called **complete minimum distance decoding**. We call it **bounded distance decoding** if for some t, we restrict the domain of the decoding map to $\{x \in E_q^n | d(x, C) \leq t\}$. Note: If $t = \lfloor \frac{d-1}{2} \rfloor$ it can be shown that the decoding result is unique for each vector in the domain. This combinatorial setting corresponds to what is know **hard decision decoding** ("hard input" and "hard output. In contrast, a **soft decision decoder** accepts "soft input" from the channel while producing "hard output" estimates of the correct symbols based on probability information of the code symbols).
4. **Numerical Parameters** - compute parameters such as packing radius (minimum distance), covering radius, weight distribution, etc.

Asymptotically Good Codes

Given a family of q-ary block codes, this family is **asymptotically good** if there exists a subset $S = \{C_i\}_{i=1}^{\infty}$ of the family with the length of C_i, n_i, the number of codewords in C_i, M_i and $d(C_i) = d_i$ s.t.

1. $\liminf_{i\to\infty} n_i = \infty$
2. $R = \liminf_{i\to\infty} \frac{\log_q M_i}{n_i} > 0$
3. $\delta = \liminf_{i\to\infty} \frac{d_i}{n_i} > 0$

If C_i is a linear code of dimension k_i, $\log_q M_i = \log_q q^{k_i} = k_i$. So for a sequence of linear codes, the second property becomes $\liminf_{i\to\infty} \frac{k_i}{n_i} > 0$. For a code C of length n, size M and distance d, the **rate** of C is $(\log_q M)/n$ (in the linear case k/n) and the **relative distance** is d/n.

Gilbert-Varshamov Bound

The Hilbert entropy function is

$$H_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x).$$

Define the sphere of radius r

$$V_q(n, r) = \sum_{i=0}^{r} \binom{n}{i} (q-1)^i$$

and

$$\alpha_q(\delta) = \limsup_{n \to \infty} \frac{\log_q A_q(n, n\delta)}{n}$$

which is the function on δ, the value the relative distance of a family of codes approaches, giving the largest possible rate for the family as code length approaches infinity.

Lemma: Let $0 < \delta \leq 1 - 1/q$ where $q \geq 2$. Then

$$\lim_{n \to \infty} \frac{\log_q V_q(n, \lfloor \delta n \rfloor)}{n} = H_q(\delta).$$

The proof is technical so we will omit it.

Gilbert-Varshamov Bound

Asymptotic-Gilbert Varshamov Bound:
If $0 < \delta \leq 1 - 1/q$ where $q \geq 2$ then $\alpha_q(\delta) \geq 1 - H_q(\delta)$.

Proof: By the Gilbert bound $A_q(n, \delta n) = A_q(n, \lceil \delta n \rceil) \geq \frac{q^n}{V_q(n, \lceil \delta n \rceil - 1)}$. Since $\lceil \delta n \rceil - 1 \leq \lfloor \delta n \rfloor$, $A_q(n, \delta n) \geq \frac{q^n}{V_q(n, \lfloor \delta n \rfloor)}$. By the lemma then,

$$\begin{aligned}
\alpha_q(\delta) &= \limsup_{n \to \infty} \frac{\log_q A_q(n, n\delta)}{n} \\
&\geq \limsup_{n \to \infty} \frac{\log_q \frac{q^n}{V_q(n, \lfloor \delta n \rfloor)}}{n} \\
&= \limsup_{n \to \infty} \frac{\log_q q^n - \log_q V_q(n, \lfloor \delta n \rfloor)}{n} \\
&\geq \limsup_{n \to \infty} 1 - \frac{\log_q V_q(n, \lfloor \delta n \rfloor)}{n} = 1 - H_q(\delta)
\end{aligned}$$

Gilbert-Varshamov Bound

The **Varshamov Bound** is

$$A_q(n,d) \geq B_q(n,d) \geq q^{n-\left\lceil \log_q\left(1+\sum_{i=0}^{d-2}\binom{n-1}{i}(q-1)^i\right)\right\rceil}$$

Using the Varshamov bound, the Asymptotic Gilbert-Varshamov Bound can be shown, hence the name.