| | |
|---|---|
| **6.897 Algorithmic Introduction to Coding Theory** | December 12, 2001 |
| Lecture 23 | |
| *Lecturer: Madhu Sudan* | *Scribe: Mihaela Enăchescu* |

Topics for today:

- Amplification of Hardness
- Concluding Thoughts & Further Directions

Today we will continue discussing the applications of coding theory to complexity theory.

## Complexity Theory

Motivation: **Complexity theory** studies the fundamental problem of finding $f : \{0,1\}^n \to \{0,1\}^m$ such that computing $f$ is *hard*.

A related problem is, given $f : \{0,1\}^n \to \{0,1\}^m$ hard, construct $f' : \{0,1\}^{n'} \to \{0,1\}^{m'}$ which is also hard (or very hard). This problem has interesting applications: Blum & Micali used it to construct a **pseudorandom generator**. The Blum-Micali construction, assumes the existence of a **one-way function**, which is defined below.

**Definition 1** *$g$ is a one-way function, if:*

- *given $x$, $g(x)$ is easy to compute*
- *$g(x)$ is very hard to invert (no circuit $C$ of small size inverts $g$ with non-negligible probability).*

## The Blum-Micali Construction

Suppose there exists a one-way permutation $g : \{0,1\}^n \to \{0,1\}^n$, and also suppose there exists a boolean function $B : \{0,1\}^n \to \{0,1\}$ such that $B(x)$ is easy to compute given x, but very hard to predict given $g(x)$ (i.e. for every small size circuit $C$, $\Pr_x[c(g(x)) = B(x)] \leq \frac{1}{2} + \frac{1}{p(m)}$ for any polynomial $p$). On input x, the Bloom-Micali construction generates the pseudorandom string:

$$B(x) \circ B(g(x)) \circ B(g^2(x)) \circ \cdots$$

This method gives a way to transform a random seed $x$ into a string of arbitrary length. The string looks random to any polynomial time computable function. Specific suggestion for $g$ and $B$: take $g$ to be the discrete-log function, and $B$ to be the most significant bit.

However, it seems that two different assumptions are needed in order for the method to work. We need not only to find a hard one-way function $g$ but we also need to find $B$, a hard-core predicate for $g$.

Question: Can we get $B(x)$ generically for any $g(x)$?

**Claim 2** *If $g^{-1} : \{0,1\}^n \to \{0,1\}^n$ is hard, define $\hat{g} : \{0,1\}^n \times [n] \to \{0,1\}$ such that $\hat{g}(x,i) = g^{-1}(x)_i$, then $\hat{g}$ is hard.*

Exercise: Find $B$ very hard, not just hard.

## Quantification of Hardness

Let us give the formal definition of the two notions: hard and very hard, which we have been using in our discussion.

**Definition 3** *A function $f$ is hard for circuits of size $s$ if, for any circuit $C$ of size $\leq s, \exists x \in \{0,1\}^n$ such that $f(x) \neq C(x)$ (i.e. $C$ doesn't compute $f$ exactly).*

**Definition 4** *A (boolean) function $f$ is very hard for circuits of size $s$ if, for any circuit $C$ of size $\leq s$,*

$$\Pr_x[f(x) = C(x)] < \frac{1}{2} + \epsilon$$

*(i.e. $C$ cannot predict $f$).*

**Amplification** : Given a function $f : \{0,1\}^n \to \{0,1\}^n$ which is hard in the worst case, we want a function $\hat{f} : \{0,1\}^n \to \{0,1\}$ which is very hard.

**Theorem 5** *If nice list-decodable codes exist, then amplification is possible*

More clarifications:

**Definition 6** *Nice list-decodable codes are a family of binary codes specified by two parameters, $k$ (length of the message) and $\epsilon$ which satisfy the following:*

- **Encoding property:** *Given $k$ and $\epsilon$, the generator matrix is constructible in time polynomial in $\frac{k}{\epsilon}$.*
- **Decoding property:** *Given $r \in \{0,1\}^n$, the (implicitly) received vector, compute (implicitly) all $M_1, \cdots, M_l \in \{0,1\}^k$ such that $\frac{\Delta(E_{k,\epsilon}(M_i),r)}{n} \leq \frac{1}{2} - \epsilon$ in time polynomial in $\frac{\log k}{\epsilon}$.*

We need:

- **Implicit input**: a procedure $r - box$ which upon input $i$ returns $r_i$ in unit time.
- **Implicit output**: For each of the $l$ possible $M$'s, we need a procedure, the $M_j box$ which upon input $i$ generates $(M_j)_i$. The $M_j box$ can send queries to the $r - box$, queries which would depend on $i$.

Nice list-decodable codes exist, for example Reed-Muller Codes concatenated with Hadamard codes!

We are now ready to prove the theorem.

**Proof** Given $f$ hard, we want $\hat{f}$ which is very hard. The function $f : \{0,1\}^n \to \{0,1\}$ can be represented as a $2^n$-bit string. Let $K = 2^n$, and let $C_{K,\epsilon}$ be the appropriate member of the nice code family. Define $\hat{f} = C_{K,\epsilon}$.

Observe that $C_{K,\epsilon}$ maps $K$ bits into $N = 2^{n'}$-bit string, which can be regarded as representing a boolean function defined on $\{0,1\}^{n'}$.

Why is this hard, in fact very hard?

Assume we have a small circuit $A$ of size $\leq s$ that predicts $\hat{f}$ with probability $\frac{1}{2} + \epsilon$ (think of $A$ as the $r - box$. Use the list-decoding algorithm on this input. We get a collection of small circuits that compute the $M_i$'s. The condition that $A$ predicts $\hat{f}$ with high probability says that $f$ is one of these message nodes. Recall that the size of the $M_i - box$-es is polynomial in $\frac{\log K}{\epsilon} \cdot s$. This implies that we have found a small circuit which computes $f$. Contradiction with our initial assumption that $f$ is hard!

■

## Conclusions & Further Directions

- **Codes**
  - We discussed mainly block codes
  - **Further Directions** Convolution codes [Blahut's; Schilman] are another type of codes, appropriate for situations where a fixed block length code is not appropriate. For example, telephone companies do not know in advance how long the message to be send is. Dividing the input in certain block sizes would be either waistful or would introduce delays. Thus, they use convolution codes which allow the encoding/decoding to be done on the fly (no delays).
- **Bounds**
  - **Further Directions** Linear Programming Bounds
- **Algorithms**
  - We saw some very interesting algorithms. In particular, the fast decoding of Reed-Solomon codes is a surprising, powerful, somewhat unexpected result.
  - New results regarding iterative algorithms & LDPC Codes.
- **Applications**
  - We saw some applications to complexity theory.
  - New applications of coding theory discovered every day.
- **Other Topics**
  - Quantum Information Theory
    - DIMACS Quantum Computing Tutorial and Workshop
    - Nielsen Chuang
  - Space-Time Codes: new family of codes [Vahid Tarokh (vahid@eecs.mit.edu)]