Algorithms of Information Security: Error-correcting codes III

Faculty of Information Technology
Czech Technical University in Prague

October 13, 2021

Reed–Muller codes

- Reed-Muller codes are named after David E. Muller, who developed the codes in 1954, and Irving S. Reed, who designed the first efficient decoding algorithm.
- Reed-Muller codes are error correcting codes that are used in wireless communication applications, especially in space communication.
- Reed-Muller codes with parameters r and m are denoted by $R(r, m)$, where r and m are integers such that $0 \leq r \leq m$.
- Reed-Muller codes can be considered as a generalization of Reed-Solomon codes.
- Reed-Muller codes are linear codes defined by evaluating polynomials of several variables. In the lecture we consider mainly binary Reed-Muller codes.

Basic Definitions

Definition

The Boolean function of m variables is a map $F_2^m \to F_2$.

Definition

Polynomial $f(x_1, \ldots, x_m)$ in m variables over F_2 is *boolean polynomial*, if in each member of the sum

$$f(x_1, \ldots, x_m) = \sum_{(i_1, \ldots, i_m)} a_{i_1 \ldots i_m} x_1^{i_1} \ldots x_m^{i_m}$$

all exponents $i_1, \ldots, i_m$ are equal to 0 or 1.

Basic Definitions

- Boolean polynomial $f(x_1, \dots, x_m)$ is thus the sum of monomials in a form

$$x_{j_1} x_{j_2} \dots x_{j_k}$$

where $1 \leq j_1 < \dots < j_k \leq m$.

- Each set $I \subset \{1, \dots, m\}$ corresponds to a monomial

$$x_I = \prod_{i \in I} x_i.$$

- Monomial $x_\emptyset$ is denoted by the symbol 1.
- Polynomial 0 denotes the sum of an empty set of monomials.
- The total degree of the polynomial $f \in F_q[x_1, \dots, x_m]$ is the value $\max \sum_{j=1}^{m} i_j$, where the maximum is over all members $x_1^{i_1} \dots x_m^{i_m}$, which have a non-zero coefficient.

Basic Definitions

- Since in the field F_2 holds that $0^2 = 0$ and $1^2 = 1$, then for $i = 1, \ldots, m$ the following equality holds:

$$x_i{}^2 = x_i.$$

- Using this property, we can (uniquely) modify the product of two Boolean polynomials into a polynomial, which is again Boolean. For example:

$$x_1 x_3 \cdot (x_1 + x_2) = x_1 x_3 + x_1 x_2 x_3.$$

- Each Boolean polynomial f determines the Boolean function $\hat{f}$: if we substitute for individual variables, the resulting value is uniquely determined.
- The number of Boolean functions of m variables is the same as the number of Boolean polynomials in the variables $x_1, \ldots, x_m$.

Basic Definitions

Theorem

For every Boolean function h with m variables, there is a Boolean polynomial $f \in F_2[x_1, \ldots, x_m]$ having the property that $h = \hat{f}$.

Note. The above theorem allows us to identify a Boolean function with a uniquely determined Boolean polynomial.
Notation. If $b = (b_1, \ldots, b_m)$ is an ordered m-tuple of elements of the field F_q, then the symbol $f(b)$ denotes the value $f(b_1, \ldots, b_m)$.

Definition

Let $B_0, \ldots, B_{q^m-1}$ be the numbering of all ordered m-tuples over F_q. Reed-Muller code $R_q(r, m)$ consists of the words in a form:

$$(f(B_0), f(B_1), \ldots, f(B_{q^m-1}))$$

where words are obtained from all polynomials f in $F_q[x_1, \ldots, x_m]$, whose total degree is at most r. The lenght of the code $R_q(r, m)$ is therefore q^m.

Binary Reed–Muller codes

Notation. For any polynomial $f \in F_2[x_1, \dots, x_m]$ let's denote

$$N(f) = \{(i_1, \dots, i_m) \in F_2^m : f(i_1, \dots, i_m) = 1\}.$$

The lower bound on the size of the set $N(f)$ implies an estimate of the minimum distance of the (binary) Reed-Muller codes.

Theorem

Let $f \in F_2[x_1, \dots, x_m]$ be nonzero Boolean polynomial of total degree at most r. Then

$$|N(f)| \geq 2^{m-r}.$$

Consequence. A set $B_r \subset R(r, m)$, consisting of the evaluations of all monomials of the total degree at most r is the base of the code $R(r, m)$.
Consequence. Reed–Muller code $R(r, m)$ has length 2^m, dimension $\binom{m}{0} + \dots + \binom{m}{r}$ and minimal weight 2^{m-r}.

Theorem

The codes $R(r, m)$ and $R(m - r - 1, m)$ are dual to each other.

Binary Reed–Muller codes

Example. Let $r = 1$ and $m = 3$, then the length of $R_2(1, 3)$ code is $n = 8$. Monomials in $F_2[x_1, x_2, x_3]$ of degree atmost 1 are $\{1, x_1, x_2, x_3\}$. When evaluating, consider the elements of the set F_2^3 in the order:

$$(x_3x_2x_1) : 000, 001, 010, 011, 100, 101, 110, 111.$$

Vectors over F_2^8 associated with these monomials are:

$$\begin{array}{rl}
\text{(evaluation of) } 1 \rightarrow & (1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1) \\
\text{(evaluation of) } x_1 \rightarrow & (0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1) \\
\text{(evaluation of) } x_2 \rightarrow & (0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1) \\
\text{(evaluation of) } x_3 \rightarrow & (0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1).
\end{array}$$

Therefore, the generating matrix of the code $R_2(1, 3)$ is as follows:

$$\begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{pmatrix}$$

Fuzzy extractors - motivation

- Fuzzy extractors present an approach for handling secret biometric data in cryptographic applications.
- Fuzzy extractor extracts a uniformly random string R from its input w in a noise-tolerant way.
- If the input w changes to w', which is only "slightly" different from w, the string R can be reproduced exactly.
- Fuzzy extractors are used for encryption and authentication, using biometric input as a key.

Fuzzy extractors - basic definitions and notations

- U_ℓ denotes the uniform distribution $\{0,1\}^\ell$.
- If a function f is *randomized*, we denote by $f(x;r)$ the result of computing f on input x with randomness r.
- *Predictability* of a random variable A is $\max_a \mathbb{P}[A=a]$.
- *min-entropy* $H_\infty(A)$ is $-\log(\max_a \mathbb{P}[A=a])$. $H_\infty(A)$ can be viewed as the "worst-case" entropy.
- A random variable with min-entropy at least m is called an m-source.

Fuzzy extractors - basic definitions and notations

- Consider now a pair of (possibly correlated) random variables A and B. If the adversary finds out the value b of B, then the predictability of A becomes $\max_a \mathbb{P}[A = a|B = b]$.
- On average, the adversary's chance of success in predicting A is $\mathbb{E}_{b \leftarrow B} [\max_a \mathbb{P}[A = a|B = b]]$. (We are taking the average over B (which is not under adversarial control), but the worst case over A).
- *Conditional min-entropy*

$$\widetilde{H}_\infty(A|B) \stackrel{\text{def}}{=} -\log \mathbb{E}_{b \leftarrow B} [\max_a \mathbb{P}[A = a|B = b]] =$$

$$= -\log \mathbb{E}_{b \leftarrow B} \left[2^{-H_\infty(A|B=b)}\right]$$

Conditional min-entropy

- Conditional min-entropy satisfies a *weak chain rule*, namely, revealing any λ bits of information about A can cause its entropy to drop by at most λ.
- The definition of conditional min-entropy is suitable for cryptographic purposes and, in particular, for extracting "nearly" uniform randomness from A.
- "nearly" here corresponds to the *statistical distance* between two probability distributions A and B, defined as $SD[A, B] = \frac{1}{2} \sum_v |\mathbb{P}[A = v] - \mathbb{P}[B = v]|$.
- SD can be interpreted as a measure of distinguishability. We write $A \approx_\varepsilon B$ to say that A and B are at distance at most ε.

Strong extractor

Definition

A randomized function $Ext : \mathcal{M} \to \{0,1\}$ with randomness of length r is an (m, ℓ, ε)-strong extractor if for all m-sources W on $\mathcal{M}$, $(Ext(W; I), I) \approx_\varepsilon (U_\ell, U_r)$, where $I = U_r$ is independent of W.

We think of the output of the extractor as a key generated from $w \leftarrow W$ with the help of a seed $i \leftarrow I$.

Lemma

Strong extractors can extract at most $\ell = m - 2\log(1/\varepsilon) + \mathcal{O}(1)$ bits from (arbitrary) m-sources.

Properties of strong extractor

Definition

$Ext(w;i)$ with an ℓ-bit output is universal if for each $w_1 \neq w_2$, $\mathbb{P}_i[Ext(w_1;i) = Ext(w_2;i)] = 2^{-\ell}$.

If elements of $\mathcal{M}$ can be represented as n-bit strings, *universal hash functions* can be built using seeds of the length n: for instance, simply view w and x as members of $GF(2^n)$ and let $Ext(w;x)$ be ℓ least significant bits of wx.
Using universal hash functions we can extract
$\ell = m + 2 - 2\log(1/\varepsilon)$ bits:

Lemma

Let for any E (possibly dependent on W), if $\widetilde{H}_\infty(W|E) \geq m$ and $\ell = m + 2 - 2\log(1/\varepsilon)$, then $(Ext(W;I), I, E) \approx_\varepsilon (U_\ell, I, E)$.

Secure sketches and fuzzy extractors

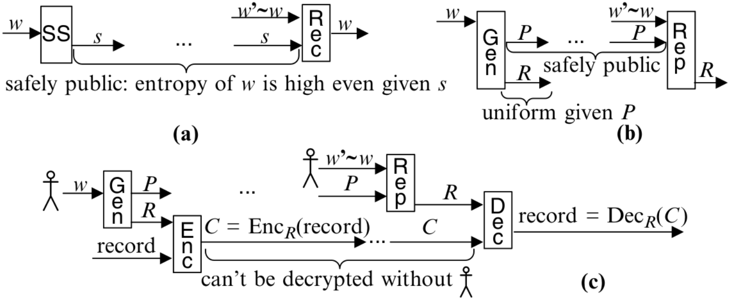


Fig. 5.1. **(a)** Secure sketch; **(b)** fuzzy extractor; **(c)** a sample application. The user encrypts a sensitive record using a key R extracted from biometric w via a fuzzy extractor; both P and the encrypted record may be sent or stored in the clear.

Secure sketch

Let $\mathcal{M}$ be a metric space with distance function dis. Informally, a *secure sketch* enables recovery of a string $w \in \mathcal{M}$ from any "close" string $w' \in \mathcal{M}$ without leaking too much information about w.

Definition

An $(m, \widetilde{m}, t)$-secure sketch is a pair of efficient randomized procedures (SS, Rec) ("sketch" and "recover") such that the following hold:

1. The sketching procedure SS on input $w \in \mathcal{M}$ returns a string $s \in \{0,1\}^*$. The recovery procedure Rec takes an element $w' \in M$ and $s \in \{0,1\}^*$.
2. Correctness: If $dis(w, w') \leq t$, then $Rec(w', SS(w)) = w$.
3. Security: For any m-source over $\mathcal{M}$, the min-entropy of W given s is high: For any (W, E), if $\widetilde{H}_\infty(W|E) \geq m$, then $\widetilde{H}_\infty(W|SS(W), E) \geq \widetilde{m}$.

Fuzzy extractor -informal

Fuzzy extractors do not recover the original input but, rather, enable generation of a close-to-uniform string R from w and its subsequent reproduction given any w' close to w.

The reproduction is done with the help of the helper string P produced during the initial extraction; yet P need not remain secret, because R is nearly uniform even given P.

Fuzzy extractor

Definition

An $(m, \ell, t, \varepsilon)$-fuzzy extractor is a pair of efficient randomized procedures (Gen, Rep) ("generate" and "reproduce") such that the following hold:

1. Gen, given $w \in \mathcal{M}$, outputs an extracted string $R \in \{0,1\}^\ell$ and a helper string $P \in \{0,1\}^*$. Rep takes an element $w' \in \mathcal{M}$ and a string $P \in \{0,1\}^*$.
2. Correctness: If $dis(w, w') \leq t$ and $(R, P) \leftarrow Gen(w)$, then $Rep(w', P) = R$.
3. Security: For all m-sources W over $\mathcal{M}$, the string R is nearly uniform even given P; that is, if $\widetilde{H}_\infty(W|E) \geq m$, then $(R, P, E) \approx_\varepsilon (U_\ell, P, E)$.

Fuzzy extractor - notes

- *Entropy loss* of a secure sketch (resp. fuzzy extractor) is $m - \widetilde{m}$ (resp. $m - \ell$).
- the nearly-uniform random bits output by a fuzzy extractor can be used in a variety of cryptographic contexts that require uniform random bits (e.g., for secret keys).
- The slight nonuniformity of the bits may decrease security, but by no more than their distance ε from uniform.
- By choosing ε sufficiently small (e.g., 2^{-100}) one can make the reduction in security irrelevant.
- If more than ℓ random bits are needed, then pseudorandom bits can be obtained by inputting R to a pseudorandom generator.

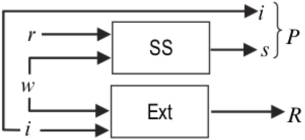
Secure Sketches Imply Fuzzy Extractors

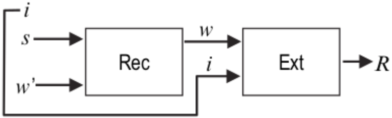
Given a secure sketch, we can always construct a fuzzy extractor that generates a key of length almost $\widetilde{m}$ by composing the sketch with a good (standard) strong extractor. The following lemma is stated for universal hash functions:

Lemma

Suppose we compose an $(m, \widetilde{m}, t)$-secure sketch (SS, Rec) for a space $\mathcal{M}$ and a universal hash function $Ext : M \to \{0,1\}^*$ as follows: In Gen, choose a random i and let $P = (SS(w), i)$ and $R = Ext(w; i)$; let $Rep(w', (s, i)) = Ext(Rec(w', s), i)$. The result is an $(m, \ell, t, \varepsilon)$-fuzzy extractor with $\ell = \widetilde{m} + 2 - 2\log(1/\varepsilon)$.

Secure Sketches Imply Fuzzy Extractors





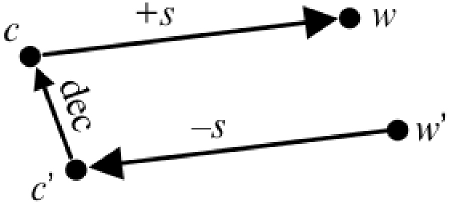
Construction of secure sketch for Hamming distance

- Constructions of secure sketches are based on error-correcting codes.
- To obtain a secure sketch for correcting Hamming errors over $\mathbb{F}^n$ ($\mathbb{F}$ is a field), we start with a $[n, k, 2t+1]$ error-correcting (linear) code C.
- The idea is to use C to correct errors in w, even though w may not be in C, by shifting the code so that a codeword matches up with w and storing the shift as the sketch.

Construction of secure sketch for Hamming distance

Definition

Construction 1 (Code-offset construction). On input w, select a uniformly random codeword $c \in C$, and set $SS(w)$ to be the shift needed to get from c to w: $SS(w) = w - c$. To compute $Rec(w', s)$, subtract the shift s from w' to get $c' = w' - s$, decode c' to get c (note that since $dis_{\mathrm{Ham}}(w', w) \leq t$ then $dis_{\mathrm{Ham}}(c', c) \leq t$), and compute w by shifting back to get $w = c + s$.



Construction of fuzzy extractor for Hamming distance

Theorem

For any m, given an $[n, k, 2t+1]$ error-correcting code, Construction 1 is an $(m, m-(n-k)\log|\mathbb{F}|, t)$-secure sketch for the Hamming distance over $\mathbb{F}^n$. Combined with Lemma "Secure Sketches Imply Fuzzy Extractors", this construction give, for any ε, an $(m, m-(n-k)\log|\mathbb{F}|+2-2\log(1/\varepsilon), t, \varepsilon)$ fuzzy extractor for the same metric.

Construction of fuzzy extractor for Hamming distance

- The trade-off between the error tolerance and the entropy loss depends on the choice of error-correcting code.
- For large alphabets ($\mathbb{F}$ is a field of size $\geq n$), one can use Reed-Solomon codes to get the optimal entropy loss of $2t \log |\mathbb{F}|$.
- No secure sketch construction can have a better trade-off between error tolerance and entropy loss than *Construction 1* (there are more constructions, see [2])), as searching for better secure sketches for the Hamming distance is equivalent to searching for better error-correcting codes.

Source

[1] [Czech] Samoopravné kódy, učební text prof. Kaisera http://home.zcu.cz/ kaisert/kody/kody.pdf

[2] Tuyls, P., Škoric, B., & Kevenaar, T. (Eds.). (2007). *Security with noisy data: on private biometrics, secure key storage and anti-counterfeiting*. Springer Science & Business Media.