CR10: Cryptanalysis Autumn 2018

Lecture 4 — October 17th , 2018

Lecturer G. Hanrot, E. Kirshanova *Scribe: Tymofii Prokopenko*

Information Set Decoding. The representation technique.

1 The Syndrome Decoding Problem

Definition 1. *(Syndrome Decoding Problem)*

In the Syndrome Decoding Problem (SDP) with parameters n*,* $k = k(n)$*, we are given a parity-check matrix* $H \in \mathbb{F}_2^{(n-k)\times n}$*, a syndrome* $s \in \mathbb{F}_2^{n-k}$ *and a target weight* $w = w(n) \in \mathbb{N}$*. We search for* $e \in \mathbb{F}_2^n$ *s.t.*

$$H \cdot e = s \text{ and } wt(e) \leq w,$$

here $wt(x)$ *denotes the Hamming weight of* $x \in \mathbb{F}_2^n$*.*

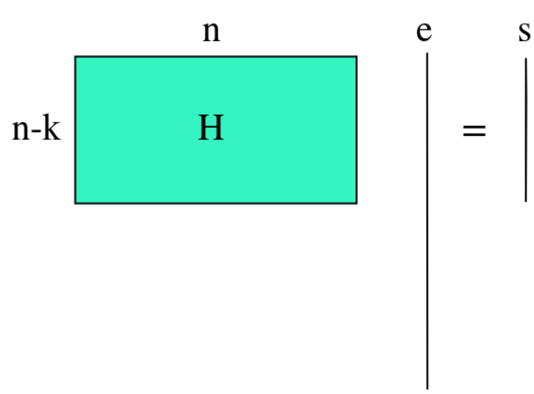

Figure 1: A visual representation of the SDP

Remarks

1. The SDP is motivated by the task of decoding random $[n, k, d]$ - linear codes: Given a generator matrix $G = \mathbb{F}_2{}^{n\times k}$, an $[n, k, d]$ code C is a $k - dim$ subspace of $\mathbb{F}_2^n$:

$$C = \{\, G \cdot m \mid m \in \mathbb{F}_2{}^k \,\}$$

or alternatively

$$C = \{\, c \in \mathbb{F}_2^n \mid H \cdot c = 0\},$$

where $d = \min_{\substack{c,c' \in C \\ c \neq c'}} \{wt(c \oplus c')\}$ - minimal distance of the code.

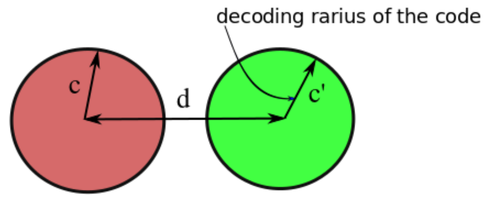

Figure 2: Decoding radius of the code.

When a code-word $c \in C$ is transmitted via a nosy channel, we obtain

$$x = c + e$$

The decoding proceeds by considering $H \cdot x = H \cdot c + H \cdot e = s$;
s is called the syndrome of x.

2. Decoding for random linear codes is NP-hard [2].

3. For sufficiently large n, H has full row-rank $(n-k)$ with probability at least $1 - e^{-\Omega(n)}$ (will be the case throughout).

4. The SDP has 3 parameters $n, k = k(n), w = w(n)$. Consider w first.

   - if $w - const$, i.e. $w = \Theta(1)$, the SDP can be solved in time $\mathcal{O}(\binom{n}{w}) = \mathcal{O}(n^w) = poly(n)$
   - if $w \geq (1/2)n$, a randomly chosen pre-image of s has expected weight $n/2$, hence, a solution is expected to be found in $\mathcal{O}(poly(n))$ time.
   - in this lecture, we assume that $w = \lfloor (d-1)/2 \rfloor (< \frac{n}{2})$, and without loss of generality also that we know w (otherwise loop over all possible integers in $[\lfloor \frac{d-1}{2} \rfloor])$. That is, we assume the solution is unique.
   - for a random $[n, k, d]$ - code and the large enough n, k, it holds $\frac{k}{n} = 1 - H(\frac{d}{n})$ - GV bound $(\frac{k}{n} = 1 - H(\frac{2w-1}{n}))$

The equations allows to express w as a function of n, k. The complexity of algorithms for SDP are usually expressed in the form

$$2^{n+o(n)} = 2^{(c+o(1)) \times n}$$

for the worst case k. Information set $\simeq I$ of k positions of a code word c s.t.
code word $c = \{c_i : i \in I\}$ that specifies c entirely. Improving C is an active research topic.

Applications in Crypto: McEliece cryptosystem

2 Syndrome Decoding Algorithms (Information Set Decoding)

Trivial Brute-force. Enumerate all $e \in \mathbb{F}_2^n$ with $wt(e) = w$. The correct e must satisfy $H \cdot e = s$

$$T_{\text{Brut-Force}} = \binom{n}{w},\ M_{\text{Brut-Force}} = poly(n)$$

2.1 Prange's Syndrome Decoding [6]

Algorithm 1 Prange's Algorithm '62

Observation: permuting the columns of H permutes positions of $1's$ in e

Input: H, s, w

Output: e

1: Apply a random permutation Π to $H, (H \leftarrow \Pi \cdot H)$

2: Bring H to the systematic form: $U_G \cdot H = [Q|I_{n-k}]$.(It can be achieved by Gaussian elimination, U_G- the corresponding invertible matrix)

3: Apply U_G to $s : s' = U_G \cdot s$

4: **if** all the 1's of e are on the last "I_{n-k}" coordinates **then**

5: $\quad s'$ reveals e, i.e $wt(s') = w$ **return** $e = \Pi s'$

6: **else**

7: $\quad$ **Goto** Step 1

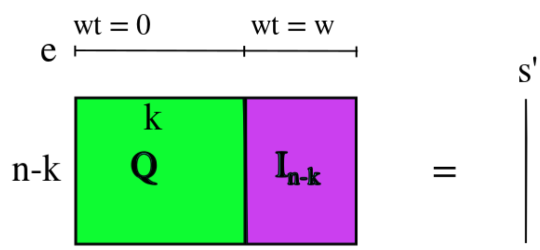

Figure 3: Systematic form for H.

Theorem 2. *Prange's algorithm solves the SDP:*

$$T_{PRANGE} = \widetilde{\mathcal{O}}\left(\frac{\binom{n}{w}}{\binom{n-k}{w}}\right),\ M_{PRANGE} = poly(n)$$

In particular,

$$T_{PRANGE} = \widetilde{\mathcal{O}}(2^{0.0588n})$$

Proof.

$$\Pr_{\substack{e \in \mathbb{F}_2^n \\ wt(e)=w}} \{e \text{ has all its } w-\text{many 1's on the last } (n-k) \text{ coordinates}\} = \frac{\binom{n-k}{w}}{\binom{n}{w}}.$$

Bringing H to the systematic form can be done in $poly(n)$ time.

$\square$

Remark Prange improves over the BF attack by a factor of $\binom{n-k}{w}$

2.2 Stern's algorithm [7]

Idea. Once again, we permute H and bring it to the systematic form: $[Q|I_{n-k}]$ by multiplying by some U_G. Contrary to Prange's algorithm, we allow non-zero weight for e on the "Q" part: we cut it into three pieces,

$$\begin{cases} e_1 \in \mathbb{F}_2^{k/2} \times 0^{k/2} \times 0^{n-k}; \\ e_2 \in 0^{k/2} \times \mathbb{F}_2^{k/2} \times 0^{n-k}; \\ e_3 \in \mathbb{F}_2^{n-k}, \end{cases}$$

So $H \cdot e = s$ becomes

$$Qe_1 + Qe_2 + e_3 = s'$$

where $s' = U_G \cdot s$. The previous equality can be read $Qe_1 \approx Qe_2 + s'$, up to some "error" e_3.

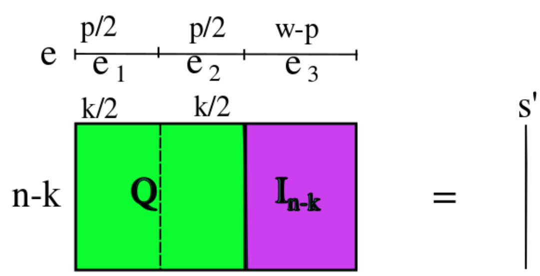

Figure 4: Stern's Algorithm visual representation.

If the starting permutation Π additionally gives $wt(e1) = wt(e2) = p/2$, then we enumerate all Qe_1 with $e_1 \in \mathbb{F}_2^{k/2} \times 0^{k/2} \times 0^{n-k}$ into a list L. For each $Qe_2 + s'$ with $e_2 \in 0^{k/2} \times \mathbb{F}_2^{k/2} \times 0^{n-k}$ we check if there is an element in L "close" to $Qe_2 + s'$.

Notation: $V_{[l]}$ - projection of V onto the first l coordinates.

Stern looks for two elements Qe_1, $Qe_2 + s'$ that are equal on fixed $l-$coordinates (the idea is that if two binary vectors are close, there must exist coordinates on which they are equal)

$$(Qe_1)_{[l]} = (Qe_2 + s')_{[l]}$$

.

Algorithm 2 Stern's Algorithm '89

Input: H, s; (p and l - parameters to be optimized)
Output: e

1: Permute the columns of H
2: Bring H to the systematic form; apply the same transformation on s.
3: Let $L \leftarrow \{\}$
4: **for all** $e_1 \in \mathbb{F}_2^{k/2} \times 0^{k/2} \times 0^{n-k}$ s.t. $wt(e_1) = p/2$ **do**
5: $\quad L \leftarrow L \cup \{(e_1, (Qe_1)_{[l]})\}$
6: $\quad$ Sort L wrt the component $(Qe_1)_{[l]}$
7: **for all** $e_2 \in 0^{k/2} \times \mathbb{F}_2^{k/2} \times 0^{n-k}$ s.t. $wt(e_2) = p/2$ **do**
8: $\quad$ **if** $\exists j : L[j][2] == (Qe_2 + s')_{[l]}$ **then**
9: $\quad\quad$ Let $e_3 = Q(L[j][1] + e_2) + s'$
10: $\quad\quad$ **if** $wt(e_3) = w - p$ **then**
11: $\quad\quad\quad$ **return** $e = [L[j][1], e_2, e_3]$
12: $\quad\quad$ **else**
13: $\quad\quad\quad$ Go to Step 1.

Theorem 3. *Stern's algorithm solves SDP in time*

$$T_{STERN} = \max\left\{ \widetilde{\mathcal{O}}\left(\frac{\binom{n}{w}}{\binom{k/2}{p/2}\binom{n-k-l}{w-p}} \right), \widetilde{\mathcal{O}}\left(\frac{\binom{n}{w}}{2^l\binom{n-k-l}{w-p}} \right) \right\}$$

In particular, for the optimal choices of $p = 0.03n$ and $l = 0.013n$. Stern's algorithm achieves

$$T_{STERN} = \widetilde{\mathcal{O}}(2^{0.05563n}),\ M_{STERN} = \widetilde{\mathcal{O}}(2^{0.013n})$$

Proof. $P = \Pr\{\Pi$ from Step 1 gives the desired distribution of 1's in $e\} = \left(\frac{\binom{k/2}{p/2}\binom{k/2}{p/2}\binom{n-k-l}{w-p}}{\binom{n}{w}} \right)$, where
$\binom{k/2}{p/2}$ - $(p/2)$ 1's on the first $k/2$ coordinates
$\binom{k/2}{p/2}$ - $(p/2)$ 1's on the second $k/2$ coordinates
$\binom{n-k-l}{w-p}$ - all the 1's on the $n-k-l$ coordinates ($0's$ on l)
Step 2 is $poly(n)$.
Step 4 costs $\widetilde{\mathcal{O}}(\binom{k/2}{p/2})$ (and determines the memory).

The number of false positives checks performed on Step 7 is $\frac{\binom{k/2}{p/2}^2}{2^l}$. It leads to the running time of the algorithm is $P^{-1}\max\left\{\binom{k/2}{p/2}, \frac{\binom{k/2}{p/2}^2}{2^l}\right\}$ □

2.3 Representation technique MMT (May-Mauer-Thomas) [5]

Stern's algorithm, given a random matrix $Q \in \mathbb{F}_2{}^{l\times k}$ and a target vector $s \in \mathbb{F}_2{}^l$, searches for an index set $I \subset [k]$ of size $|I| = p$ s.t. $\sum\limits_{i\in I} q_i = s$
(q_i - columns of Q).

This is a vectorial version of the subset sum problem. Stern exploits MitM approach: it splits I into 2 disjoint sets I_1 and I_2 i.e. $I = I_1 \cup I_2$, $|I_1| = |I_2| = p/2$

Idea. Choose I_1, I_2 from the full set $[k]$ ($|I_1| = |I_2| = p/2$). This has the effect of obtaining many ways to express the solution e as the sum of two binary vectors. We want to explore several (maybe all) such expressions.

Example:

$$\begin{aligned} e &= (1,0,0,0,0,1) \qquad wt = 2 \\ &\;\| \\ e_1 &= \left(\frac{1}{0}, 0,0,0,0, \frac{0}{1}\right) \qquad wt = 1 \\ &\;+ \\ e_2 &= \left(\frac{0}{1}, 0,0,0,0, \frac{1}{0}\right) \qquad wt = 1 \end{aligned}$$

(Two equalities can be read in this example: one by considering the digits above the fraction bar, and one by considering the digits below.)

In particular, there are $\binom{p}{p/2} \approx 2^p$ different identities

$$\sum_{i \in I_1} q_i = \sum_{i \in I_2} q_i + s \tag{1}$$

for I_1, $I_2 \subset [k]$.
We do not consider all possible sums of the form (1), but only a $\frac{1}{2^p}$ - fraction of them. In fact, we construct the lists for an appropriately chosen $l_2 \in \mathbb{N}$.

$$\begin{aligned} L_1 &= \{(I_1, \sum_{i \in I_1} q_i), I_1 \subset [k], |I_1| = p/2 \text{ and } (\sum_{i \in I_1} q_i)_{[l_2]} = 0 \in {\mathbb{F}_2}^{l_2}\} \\ L_2 &= \{(I_2, \sum_{i \in I_2} q_i), I_2 \subset [k], |I_2| = p/2 \text{ and } (\sum_{i \in I_2} q_i + s')_{[l_2]} = 0 \in {\mathbb{F}_2}^{l_2}\} \end{aligned}$$

That is, we only consider the identities (1) which are equal to 0 on l_2 - bits. Therefore, we expect to remove a $\frac{1}{2^{l_2}}$ - fraction of all solutions. L_1 (analog L_2) is constructed in the MitM way by merging the two lists:

$$\begin{aligned} L_{11} &= \{(I_{11}, \sum_{i \in I_{11}} q_i) : I_{11} \subset [1, k/2], |I_{11}| = p/4\} \\ L_{12} &= \{(I_{12}, \sum_{i \in I_{12}} q_i) : I_{12} \subset [k/2+1, k], |I_{12}| = p/4\} \\ L_{21} &= \{(I_{21}, \sum_{i \in I_{21}} q_i) : I_{21} \subset [1, k/2], |I_{21}| = p/4\} \\ L_{22} &= \{(I_{22}, \sum_{i \in I_{22}} q_i) : I_{22} \subset [k/2+1, k], |I_{22}| = p/4\} \end{aligned}$$

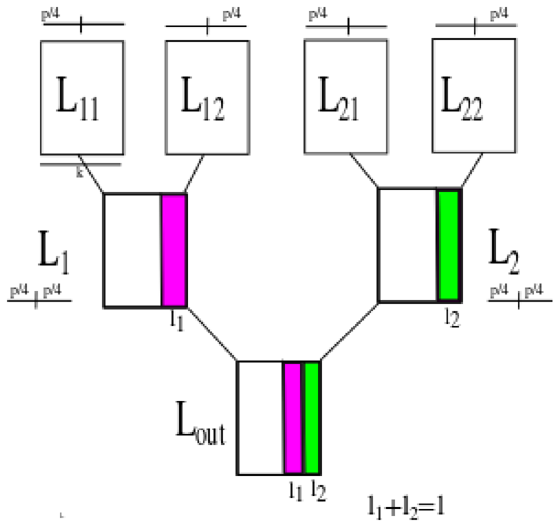

Figure 5: A visualization of Representation technique for a vectorial subset sum.

Algorithm 3 Representation technique for a vectorial subset sum.

Input: $Q \in {\mathbb{F}_2}^{l \times k}$, $s' \in {\mathbb{F}_2}^{l}$; l_1, l_2 - parameters to be optimized

Output: $I : \sum q_i = s$ or $\perp$

Construct L_{11},..., L_{22} // size/time: $\binom{k/2}{p/4}$

Sort L_{12}, L_{22} wrt the $\sum q_i$ and $\sum q_i + s'$// $\widetilde{\mathcal{O}}(\binom{k/2}{p/4})$

for all $(I_{11}, \sum\limits_{i \in I_{11}} q_i) \in L_{11}$ **do**

  Find all $(I_{12}, \sum\limits_{i \in I_{12}} q_i) \in L_{11}$ s.t. $(\sum\limits_{i \in I_{11}} q_i)_{l_1} = (\sum\limits_{i \in I_{12}} q_i)_{l_2}$

  $Insert(I_{11} \cup I_{12}, \sum\limits_{i \in I_{11}} q_i + \sum\limits_{i \in I_{12}} q_i)$ into L_1

for all $(I_{21}, \sum\limits_{i \in I_{21}} q_i) \in L_{21}$ **do**

  Find all $(I_{22}, \sum\limits_{i \in I_{22}} q_i) \in L_{21}$ s.t. $(\sum\limits_{i \in I_{21}} q_i)_{l_1} = (\sum\limits_{i \in I_{22}} q_i)_{l_2}$

  $Insert(I_{21} \cup I_{22}, \sum\limits_{i \in I_{21}} q_i + \sum\limits_{i \in I_{22}} q_i)$ into L_2

Sort L_2 wtr the $\sum\limits_{i \in I_2} q_i + s'$

for all $(I_1, \sum\limits_{i \in I_1} q_i) \in L_1$ **do**

  Find all $(I_2, \sum\limits_{i \in I_2} q_i) \in L_2$ s.t. $(\sum\limits_{i \in I_1} q_i)_{[l]} = (\sum\limits_{i \in I_2} q_i)_{[l]}$

  $Insert(I_1 \cup I_2, \sum\limits_{i \in I_1 \cup I_2} q_i)$ into L_{out}

return $L_{out}[1]$

Theorem 4. *The representation technique from Algorithm 3 leads to the complexity of the SDP*

$$T_{REPR} = \widetilde{\mathcal{O}}(2^{0.0537n}),\ M_{REPR} = \widetilde{\mathcal{O}}(2^{0.021n})$$

for optimal chooses $p = l_2 = 00.6n$ *and* $l_1 = 0.028n$

2.4 Complexity analysis (briefly)

- the lists $L_{11}, ..., L_{22}$ are of sizes $\widetilde{\mathcal{O}}(\binom{k/2}{p/4})$ - (all binary strings of length $k/2$ of $wt\ p/4$) this is also the time needed to create them;
- time to create the list L_1, L_2 is $\widetilde{\mathcal{O}}(\max\{(\frac{\binom{k/2}{p/4}^2}{2^{l_2}}), (\binom{k/2}{p/4})\})$;
- the expected size of L_1, L_2:

$$\mathbb{E}[|L_1|] = \frac{\binom{k/2}{p/4}}{2^{l_2}}$$

- time to create L_{out}:

$$\widetilde{\mathcal{O}}(\max\{(\frac{\binom{k/2}{p/4}^2)}{2^{l_2}}), (\frac{\binom{k/2}{p/4}^4}{2^{2l_2 - l_1}})\});$$

$\Pr\{\Pi \text{ is a good permutation}\} = \frac{\binom{k/2}{p/2}^2 \binom{n-k-l}{w-p}}{\binom{n}{w}}$ (same as for Stern). The success probability of Algorithm 3 (i.e. analysis on the number of representations that remain in the list) is more involved and can be found in [5].

Remarks

1. One can increase the number of representations by considering the '0' bits as well, i.e. $0 = 0+0$ or $0 = 1 + 1$. This brings the complexity of the SDP down to $2^{0.0494n+o(n)}$ [1]
2. The best known algorithm for SDP has the complexity $2^{0.0465n}$ [3] and makes use of Near-Neighbour techniques.
3. The representation technique improves density-1 SS problem from $T = \widetilde{\mathcal{O}}(2^{n/2})$ down to $T = \widetilde{\mathcal{O}}(2^{0.3113n})$ [4]
4. Algorithms for the SDP can be applied to LPN when the number of LPN samples in $\Theta(n)$

References

[1] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in 2n/20: How 1+ 1= 0 improves information set decoding. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 520–536. Springer, 2012.

[2] Elwyn Berlekamp, Robert McEliece, and Henk Van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[3] Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for lpn security. In *International Conference on Post-Quantum Cryptography*, pages 25–46. Springer, 2018.

[4] Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 235–256. Springer, 2010.

[5] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\widetilde{\mathcal{O}}(2^{0.054n})$. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 107–124. Springer, 2011.

[6] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.

[7] Jacques Stern. A method for finding codewords of small weight. In *International Colloquium on Coding Theory and Applications*, pages 106–113. Springer, 1988.