

# Code-based Cryptography

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

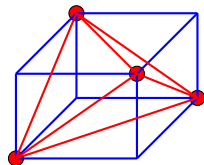
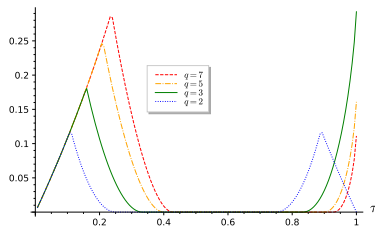
Prange's  
Algorithm

Public-key  
Encryption  
Schemes

Thomas Debris-Alazard

Inria GRACE & LIX CNRS, Ecole Polytechnique  
Palaiseau, France

August 2, 2022



Hard

Easy

Hard

$\tau$

# The Team

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

- Maxime Bombar (PhD at Inria):  
`maxime.bombar@inria.fr`
- Kévin Carrier (Assistant Professor at Cergy-Paris):  
`kevin.carrier@ensea.fr`
- Thomas Debris-Alazard (Researcher at Inria):  
`thomas.debris@inria.fr`

# The Content

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

## A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

3 lecture notes (long, for further reading): <http://tdalazard.io/>

1. *An Intractable Problem Related to Codes, Decoding*
2. *Random Codes*
3. *Information Set Decoding Algorithms*

→ Tuesday and Friday' presentations are an overview!

2 exercise sheets

1. starting exercises to get familiar with codes & crypto,
2. two cryptanalysis (if you are already familiar with codes).

# Our Lectures

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

*Today*

## Codes

*basic definitions*

## Decoding Problem

- *Worst-Case*
- *An easy case*
- *Average case*

*Friday*

**Random codes**

**Average-case  
hardness**

*Prange's algo*

**Public-key  
Encryptions**

*McEliece*

*Alekhnovich*

Don't hesitate to interrupt!

# Code-based Cryptography?

# An Old History

## Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

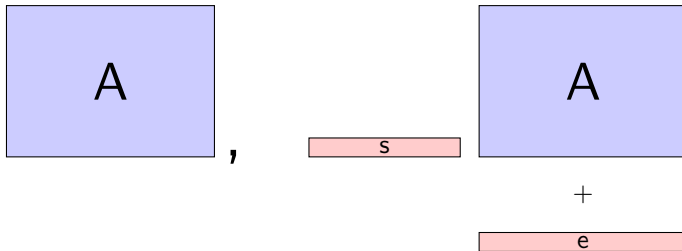
Average-case

Search-to-Decision  
Reduction

## A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

Shannon (1948/1949) introduced the following problem,



The diagram illustrates the linear system  $A \cdot s = e$ . It consists of three main components: a large light blue square labeled 'A' on the left, a small light red rectangle labeled 's' in the middle, and another large light blue square labeled 'A' on the right. A comma is placed between the first 'A' and the 's' rectangle. Below the 's' rectangle is a plus sign '+', and below the plus sign is another small light red rectangle labeled 'e'.

→ Matrix  $A$  and vectors  $s, e$  are **binary** ( $\in \mathbb{F}_2$ )

**Aim**

Recover

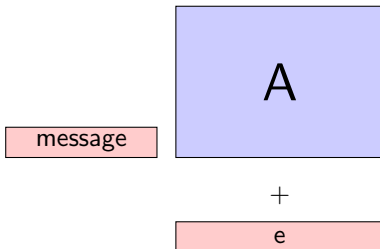


A small light red rectangle labeled 's'.

# There are trapdoors(I)!

McEliece (1978):

$A \leftarrow \text{Trapdoor}():$  public-key



- With the trapdoor, easy to recover message if e “short”,
- Without, hard

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

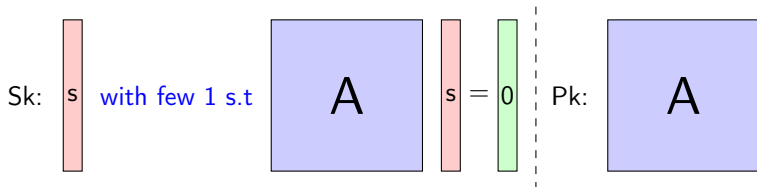
About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

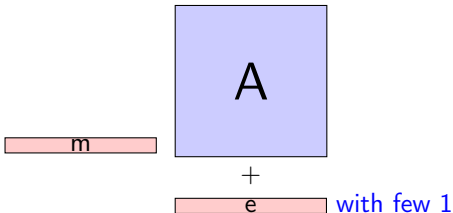
# There are trapdoors(II)!

Alekhnovich (2003):

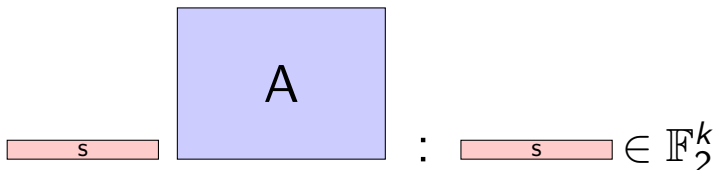


- To encrypt  $b = 1$ , send  $u \leftarrow \text{Unif}$

- To encrypt  $b = 0$ , send



# You said code?



is known as a **code**!

To understand what is a code will be useful to

1. build trapdoors
2. understand the hardness of the problem

# Codes are used in telecommunications

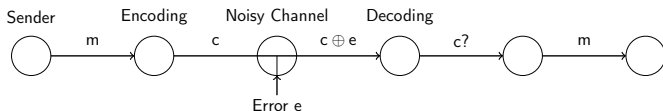
# Codes: used in telecommunications!

How to transmit  $k$  bits over a noisy channel?

# Codes: used in telecommunications!

How to transmit  $k$  bits over a noisy channel?

1. Fix  $\mathcal{C}$  subspace  $\subseteq \mathbb{F}_2^n$  of dimension  $k$
2. Map  $(m_1, \dots, m_k) \rightarrow c = (c_1, \dots, c_n) \in \mathcal{C}$   
(adding  $n - k$  bits redundancy)
3. Send  $c$  across the noisy channel



→ from  $c \oplus e$ : how to recover  $e$  and then  $c$ ?  
(Decoding Problem)

# Hamming distance

## Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

## A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

Real life scenario,  $c + e$  with  $e = (e_1, \dots, e_n)$  s.t:

$$\forall i, \quad \mathbb{P}(e_i = 1) = p \text{ and } \mathbb{P}(e_i = 0) = 1 - p$$

→ Each bit of  $c$  is flipped with probability  $p$

## Given a received corrupted word $y$

$$\mathbb{P}(c \text{ was sent} \mid y \text{ is received}) = p^{d_H(c,y)}(1-p)^{n-d_H(c,y)}$$

where  $d_H(c,y) \stackrel{\text{def}}{=} \#\{i : c_i \neq y_i\}$  (Hamming distance)

# Hamming distance

Real life scenario,  $c + e$  with  $e = (e_1, \dots, e_n)$  s.t:

$$\forall i, \quad \mathbb{P}(e_i = 1) = p \text{ and } \mathbb{P}(e_i = 0) = 1 - p$$

→ Each bit of  $c$  is flipped with probability  $p$

## Given a received corrupted word $y$

$$\mathbb{P}(c \text{ was sent} \mid y \text{ is received}) = p^{d_H(c,y)}(1-p)^{n-d_H(c,y)}$$

where  $d_H(c,y) \stackrel{\text{def}}{=} \#\{i : c_i \neq y_i\}$  (Hamming distance)

Any decoding candidate  $c \in \mathcal{C}$  is even more likely as it is close to the received message  $y$  for the Hamming distance.

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# Basic Definitions

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

$\mathbb{F}_q^n$  denotes the finite field with  $q$  elements

## Linear Code

A linear code  $\mathcal{C}$  of length  $n$  and dimension  $k$  ( $[n, k]$ -code):  
subspace of  $\mathbb{F}_q^n$  of dimension  $k$

$n$  length ;  $k$  dimension

# Example of Codes

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

First examples of codes:

1.  $\{(f(x_1), \dots, f(x_n)) : f \in \mathbb{F}_q[X] \text{ and } \deg(f) < k\}$  where the  $x_i$ 's are distinct elements of  $\mathbb{F}_q$ ,
2.  $\{(u, u + v) : u \in U \text{ and } v \in V\}$  where  $U$  (resp.  $V$ ) is an  $[n, k_U]_q$ -code (resp.  $[n, k_V]_q$ -code).

→ What are the lengths and dimensions?  
(exercise)

# How to represent a code(I)?

$\mathcal{C}$  be an  $[n, k]$ -code

**Basis representation:**  $g_1, \dots, g_k$  basis of  $\mathcal{C}$

→  $\mathcal{C} = \{mG : m \in \mathbb{F}_q^k\}$  where the **rows** of  $G \in \mathbb{F}_q^{k \times n}$  are the  $g_i$ 's

Reciprocally, any  $G \in \mathbb{F}_q^{k \times n}$  of **rank  $k$**  defines the  $[n, k]$ -code

$$\mathcal{C} \stackrel{\text{def}}{=} \{mG : m \in \mathbb{F}_q^k\}$$

$G$ : **generator (matrix)**

# How to represent a code(II)?

## Dual code

Given  $\mathcal{C}$ , its dual  $\mathcal{C}^*$  is the  $[n, n - k]$ -code

$$\mathcal{C}^* \stackrel{\text{def}}{=} \left\{ \mathbf{c}^* \in \mathbb{F}_q^n : \forall \mathbf{c} \in \mathcal{C}, \langle \mathbf{c}, \mathbf{c}^* \rangle = \sum_{i=1}^n c_i c_i^* = 0 \in \mathbb{F}_q \right\}.$$

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

# How to represent a code(II)?

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

## Dual code

Given  $\mathcal{C}$ , its dual  $\mathcal{C}^*$  is the  $[n, n - k]$ -code

$$\mathcal{C}^* \stackrel{\text{def}}{=} \left\{ c^* \in \mathbb{F}_q^n : \forall c \in \mathcal{C}, \langle c, c^* \rangle = \sum_{i=1}^n c_i c_i^* = 0 \in \mathbb{F}_q \right\}.$$

Parity-check representation:  $h_1, \dots, h_{n-k}$  basis of  $\mathcal{C}^*$

$\longrightarrow \mathcal{C} = \{c : Hc^T = 0\}$  where the **rows** of  $H \in \mathbb{F}_q^{(n-k) \times n}$  are the  $h_i$ 's

Reciprocally, any  $H \in \mathbb{F}_q^{(n-k) \times n}$  of **rank**  $n - k$  defines the  $[n, k]$ -code

$$\mathcal{C} \stackrel{\text{def}}{=} \{c : Hc^T = 0\}$$

H: **parity-check (matrix)**

# A remark

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

- $G \in \mathbb{F}_q^{k \times n}$  generator matrix of  $\mathcal{C}$  ( $\mathcal{C} = \{mG : m\}$ )

→  $SG$  still generator matrix of  $\mathcal{C}$  when

$$S \in \mathbb{F}_q^{k \times k} \text{ non-singular}$$

- $H \in \mathbb{F}_q^{(n-k) \times n}$  parity-check matrix of  $\mathcal{C}$  ( $\mathcal{C} = \{c : Hc^T = 0\}$ )

→  $SH$  still parity-check matrix of  $\mathcal{C}$  when

$$S \in \mathbb{F}_q^{(n-k) \times (n-k)} \text{ non-singular}$$

# From one representation to the other?

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

## A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

$$G \in \mathbb{F}_q^{k \times n} \text{ generator} \quad \overset{\text{easy to compute?}}{\longleftrightarrow} \quad H \in \mathbb{F}_q^{(n-k) \times n} \text{ parity-check}$$

# From one representation to the other?

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

$G \in \mathbb{F}_q^{k \times n}$  generator  $\xleftrightarrow{\text{easy to compute?}}$   $H \in \mathbb{F}_q^{(n-k) \times n}$  parity-check

YES!

1. Show that if  $H \in \mathbb{F}_q^{(n-k) \times n}$  has rank  $n - k$  and  $GH^T = 0$ , then  $H$  parity-check (exercise),
2. Perform a Gaussian elimination (see the board).

# Generator or parity-check?

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

Would you rather choose generator or parity-check representation?

# Generator or parity-check?

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

### About Random Codes

### Prange's Algorithm

### Public-key Encryption Schemes

Would you rather choose generator or parity-check representation?

**Sorry for the team generator matrix :(**

Usually, the parity-check representation is more “natural”...

# Hamming code

Let  $\mathcal{C}_{\text{Ham}}$  be the  $[7, 4]$ -code of generator matrix:

$$G \stackrel{\text{def}}{=} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$H \stackrel{\text{def}}{=} \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

has rank 3 and verifies  $GH^T = 0$ .

Let  $c + e$  where  $\begin{cases} c \in \mathcal{C}_{\text{Ham}} \\ |e| = 1 \end{cases}$  : how to easily recover  $e$ ?

# Modulo the code

Given  $c + e$ : recover  $e$ .

→ Make modulo  $\mathcal{C}$  to extract the information about  $e$

**Coset space:**  $\mathbb{F}_q^n / \mathcal{C}$

$$\# \mathbb{F}_q^n / \mathcal{C} = q^{n-k} \quad \text{and} \quad \mathbb{F}_q^n / \mathcal{C} = \{x_i + \mathcal{C} : 1 \leq i \leq q^{n-k}\}$$

A natural set of representatives via a parity-check  $H$ : **syndromes**

$$x_i + \mathcal{C} \in \mathbb{F}_q^n / \mathcal{C} \mapsto Hx_i^T \in \mathbb{F}_q^{n-k} \text{ (called a syndrome)}$$

is an isomorphism

# Syndrome or noisy codewords?

$\mathcal{C}$  be an  $[n, k]$ -code of parity-check matrix  $H$

Noisy codeword	Syndrome
$c + e$	$He^T$

- From  $c + e$ :  $H(c + e)^T = Hc^T + He^T = He^T$
- From  $He^T$ : compute with linear algebra  $y$  s.t.  
 $Hy^T = He^T \iff H(y - e)^T = 0 \iff y - e \in \mathcal{C} \iff y = c + e$

# Minimum distance

## Hamming weight

Given  $x \in \mathbb{F}_q^n$ , its Hamming weight:

$$|x| \stackrel{\text{def}}{=} \# \{i : x_i \neq 0\}$$

## Minimum distance

The minimum distance of  $\mathcal{C}$  is

$$d_{\min}(\mathcal{C}) \stackrel{\text{def}}{=} \min \{|c| : c \in \mathcal{C}, c \neq 0\}.$$

$d_{\min}(\mathcal{C})$  important quantity:

“geometry” of  $\mathcal{C}$  ; “efficiency” of  $\mathcal{C}$  ; “security” of  $\mathcal{C}$

## ① Basic on Codes

## ② The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## ③ A quick recap

## ④ About Random Codes

## ⑤ Prange's Algorithm

## ⑥ Public-key Encryption Schemes

# The Decoding Problem

Two formulations:

## Problem (Noisy Codeword Decoding)

Given  $G \in \mathbb{F}_q^{k \times n}$  of rank  $k$ ,  $t \in \llbracket 0, n \rrbracket$ ,  $y \in \mathbb{F}_q^n$  where  $y = c + e$  with  $c = mG$  for some  $m \in \mathbb{F}_q^k$  and  $|e| = t$ , find  $e$ .

## Problem (Syndrome Decoding)

Given  $H \in \mathbb{F}_q^{(n-k) \times n}$  of rank  $n - k$ ,  $t \in \llbracket 0, n \rrbracket$ ,  $s \in \mathbb{F}_q^{n-k}$  where  $He^T = s^T$  with  $|e| = t$ , find  $e$ .

→ They are equivalent!

$n$  length ;  $k$  dimension ;  $t$  decoding distance

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

# Equivalent representations

Let,  $\mathcal{A}$  be s.t  $\mathcal{A}(G, mG + e) \rightarrow e$

Given  $(H, He^T)$ : our aim, recover  $e$  using  $\mathcal{A}$

1. Compute with linear algebra  $G$  (rank  $k$ ) s.t  $GH^T = 0$
2. Compute (again) with linear algebra  $y$  s.t  $Hy^T = He^T$ .
3. Notice that  $H(y - e)^T = 0 \iff y - e = mG$  for some  $m$
4. Feed  $(G, y)$  to  $\mathcal{A}$ , it recovers  $e$ .

Exercise: show that the reciprocal holds

In what follows, we only keep the parity-check representation!

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

# NP-completeness

## Problem (Worst-case decisional decoding problem)

- Input:  $H \in \mathbb{F}_q^{(n-k) \times n}$ ,  $s \in \mathbb{F}_q^{n-k}$  where  $n, k \in \mathbb{N}$  with  $k \leq n$  and an integer  $t \leq n$ .
- Decision: *it exists*  $e \in \mathbb{F}_q^n$  of Hamming weight  $t$  such  $He^T = s^T$ ?

This problem is NP-complete...

Is it useful? Be careful of the input set...

# Drawback of the NP-completeness

The above NP-completeness shows that (if  $P \neq NP$ )

We cannot easily solve the decoding problem for **all codes and all decoding distances...**

Not a safety guarantee for cryptographic applications...

## ① Basic on Codes

## ② The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## ③ A quick recap

## ④ About Random Codes

## ⑤ Prange's Algorithm

## ⑥ Public-key Encryption Schemes

There are codes and associated distance for which  
we know how to decode!

## Generalized Reed-Solomon

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

## GRS codes

$z \in (\mathbb{F}_q^*)^n$  and  $x \in \mathbb{F}_q^n$  s.t  $x_i \neq x_j$  (in particular  $n \leq q$ ) and  $k \leq n$ .

The code  $\text{GRS}_k(x, z)$  is defined as

$$\text{GRS}_k(x, z) \stackrel{\text{def}}{=} \{(z_1 f(x_1), \dots, z_n f(x_n)) : f \in \mathbb{F}_q[X] \text{ and } \deg(f) < k\}$$

→ These codes are used in QR-codes!

Exercise:  $\text{GRS}_k(x, z)$  has generator matrix

$$G \stackrel{\text{def}}{=} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^k & x_2^k & \cdots & x_n^k \end{pmatrix} \begin{pmatrix} z_1 & & & 0 \\ & z_2 & & \\ & & \ddots & \\ 0 & & & z_n \end{pmatrix}$$

# Berlekamp-Welsh algorithm

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## Decoding algorithm

Given,  $\text{GRS}_k(x, z)$  and  $c + e$  s.t  $\begin{cases} c \in \text{GRS}_k(x, z) \\ |e| \leq \lfloor \frac{n-k}{2} \rfloor \end{cases}$

Then, easy to recover  $(c, e)$ .

## Proof.

On board!



## ① Basic on Codes

## ② The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

**Average-case**

Search-to-Decision Reduction

## ③ A quick recap

## ④ About Random Codes

## ⑤ Prange's Algorithm

## ⑥ Public-key Encryption Schemes

# The average decoding problem

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

$$\text{DP}(n, q, R, \tau), k \stackrel{\text{def}}{=} Rn \text{ and } t \stackrel{\text{def}}{=} \tau n$$

Sample:  $\boxed{H} \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$   $\boxed{x} \leftarrow \text{Unif}(z : |z| = t)$

Given:  $\boxed{H}$ ,  $\boxed{s} = \boxed{H} \boxed{x}$

Recover:  $\boxed{e}$  s.t.  $\boxed{H} \boxed{e} = \boxed{s}$  and  $\boxed{e} \in \{z : |z| = t\}$

With respect to  $\tau$ , the solution will be unique or not...

## Average hardness?

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

Let,  $\varepsilon = \mathbb{P}_{H,x} (\mathcal{A}(H, s = xH^T) = e \text{ s.t. } |e| = t \text{ and } eH^T = s)$

Then, using the law of total probability:

$$\varepsilon = \frac{1}{q^k (n-k) (q-1)^t \binom{n}{t}} \sum_{\substack{H \in \mathbb{F}_q^{(n-k) \times n} \\ |x|=t}} \mathbb{P}(\mathcal{A}(H, s = xH^T) = e \text{ s.t. } |e| = t \text{ and } eH^T = s)$$

## Average hardness?

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

Let,  $\varepsilon = \mathbb{P}_{H,x} (\mathcal{A}(H, s = xH^T) = e \text{ s.t. } |e| = t \text{ and } eH^T = s)$

Then, using the law of total probability:

$$\varepsilon = \frac{1}{q^k (n-k) (q-1)^t \binom{n}{t}} \sum_{\substack{H \in \mathbb{F}_q^{(n-k) \times n} \\ |x|=t}} \mathbb{P}(\mathcal{A}(H, s = xH^T) = e \text{ s.t. } |e| = t \text{ and } eH^T = s)$$

All known algorithms have a complexity  $T/\varepsilon$  ( $T$  running time)

$$\frac{T}{\varepsilon} = 2^{\alpha(q,R,\tau) n(1+o(1))}$$

for some  $\alpha(q, R, \tau) \geq 0$

# Its hardness



**Figure:** Hardness of  $\text{DP}(n, q, R, \tau)$  as function of  $\tau$ .

## Its hardness

## Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

## A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

**Figure:** Hardness of  $\text{DP}(n, q, R, \tau)$  as function of  $\tau$ .

- McEliece encryption:  $\tau = \Theta\left(\frac{1}{\log n}\right)$ ,
- Other encryption schemes:  $\tau = \Theta\left(\frac{1}{\sqrt{n}}\right)$ ,
- Authentication protocol:  $\tau = C$  constant quite small,
- Signature Wave:  $\tau = C$  large constant,  $C \approx 0.95$ .

# And the generator representation?

$\text{DP}'(n, q, R, \tau)$ . Let  $k \stackrel{\text{def}}{=} \lfloor Rn \rfloor$  and  $t \stackrel{\text{def}}{=} \lfloor \tau n \rfloor$

- Input :  $(G, y \stackrel{\text{def}}{=} sG + x)$  where  $G, s$  and  $x$  are uniformly distributed over  $\mathbb{F}_q^{k \times n}$ ,  $\mathbb{F}_q^k$  and words of Hamming weight  $t$  in  $\mathbb{F}_q^n$ .
- Output : an error  $e \in \mathbb{F}_q^n$  of Hamming weight  $t$  such that  $y - e = mG$  for some  $m \in \mathbb{F}_q^k$ .

## Exercise

For any algorithm  $\mathcal{A}$  solving  $\text{DP}'$  with probability  $\varepsilon$  and time  $T$ : describe  $\mathcal{B}$  which solves  $\text{DP}$  in the same time with probability  $\geq \varepsilon - O(q^{-\min(k, n-k)})$  (and the reciprocal)

→ Same average hardness with syndromes or noisy codewords formalism!

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# Average Decisional Decoding Problem

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

$\text{DDP}(n, q, R, \tau)$ ,  $k \stackrel{\text{def}}{=} \lfloor Rn \rfloor$  and  $t \stackrel{\text{def}}{=} \lfloor \tau n \rfloor$ .

- Distributions:
  - $\mathcal{D}_0$  :  $(H, s)$  be uniformly distributed over  $\mathbb{F}_q^{(n-k) \times n} \times \mathbb{F}_q^{n-k}$ .
  - $\mathcal{D}_1$  :  $(H, xH^T)$  where  $H$  (resp.  $x$ ) being uniformly distributed over  $\mathbb{F}_q^{(n-k) \times n}$  (resp. words of Hamming weight  $t$ ).
- Input:  $(H, s)$  distributed according to  $\mathcal{D}_b$  where  $b \in \{0, 1\}$  is uniform,
- Decision:  $b' \in \{0, 1\}$ .

Is this problem strictly easier than its search version?

→ No! They are equivalent (Goldreich-Levin hardcore predicate)

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

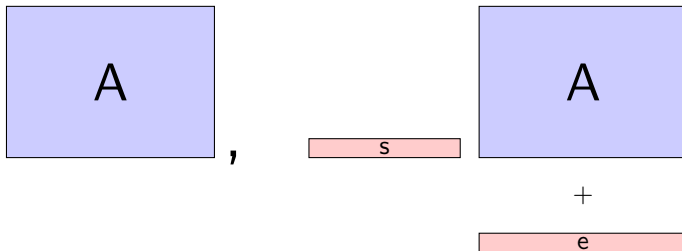
## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# An old problem: decoding

Shannon (1948/1949) introduced the decoding problem,



Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes**Aim**

Recover

S

There are cryptosystem whose security relies on this problem:  
`code`-based crypto (McEliece 78 ; Alekhnovich 03 ; etc)

# Two representations of codes

## Basic on Codes

### The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

### A quick recap

#### About Random Codes

#### Prange's Algorithm

#### Public-key Encryption Schemes

$\mathcal{C}$  be an  $[n, k]$ -code

$n$  length ;  $k$  dimension

$$\mathcal{C} \stackrel{\text{def}}{=} \{mG : m \in \mathbb{F}_q^k\}$$

$$G \in \mathbb{F}_q^{k \times n} \text{ rank } k : \text{generator (matrix)}$$

$$\mathcal{C} \stackrel{\text{def}}{=} \{c : Hc^T = 0\}$$

$$H \in \mathbb{F}_q^{(n-k) \times n} \text{ rank } n - k : \text{parity-check (matrix)}$$

## Average Decoding Problem

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes
$$\text{DP}(n, q, R, \tau), k \stackrel{\text{def}}{=} Rn \text{ and } t \stackrel{\text{def}}{=} \tau n$$

Sample:  $\boxed{H} \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$   $\boxed{x} \leftarrow \text{Unif}(z : |z| = t)$

Given:  $\boxed{H}, \boxed{s} = \boxed{H} \boxed{x}$

Recover:  $\boxed{e} \text{ s.t. } \boxed{H} \boxed{e} = \boxed{s} \text{ and } \boxed{e} \in \{z : |z| = t\}$

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# A motivation

## Average Decoding Problem (DP)

- Sample:  $H \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$ ,  $x \leftarrow \text{Unif}(\{z \in \mathbb{F}_q^n : |z| = t\})$ ,
- Given:  $(H, Hx^T)$ ,
- Find  $e \in \mathbb{F}_q^n$  s.t 
$$\begin{cases} He^T = Hx^T \\ |e| = t \end{cases}$$

A trivial algorithm:

pick  $e \in \{z \in \mathbb{F}_q^n : |z| = t\}$  and test if  $He^T = Hx^T$

# A motivation

## Average Decoding Problem (DP)

- Sample:  $H \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$ ,  $x \leftarrow \text{Unif}(\{z \in \mathbb{F}_q^n : |z| = t\})$ ,
- Given:  $(H, Hx^T)$ ,
- Find  $e \in \mathbb{F}_q^n$  s.t. 
$$\begin{cases} He^T = Hx^T \\ |e| = t \end{cases}$$

A trivial algorithm:

pick  $e \in \{z \in \mathbb{F}_q^n : |z| = t\}$  and test if  $He^T = Hx^T$

- If one solution: probability of success  $\frac{1}{\#\{z \in \mathbb{F}_q^n : |z|=t\}}$
- If  $N$  solutions: probability of success  $\approx \frac{N}{\#\{z \in \mathbb{F}_q^n : |z|=t\}}$

What is the value of  $N$ ?

# The value of $N$ ?

To compute  $N$ : use the theory of **random codes**!

## Random Code

$$\mathcal{C} = \{c \in \mathbb{F}_q^n : Hc^T = 0\} : H \leftarrow \text{Unif} \left( \mathbb{F}_q^{(n-k) \times n} \right)$$

defines a random code

# Random codes: two models

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

And generator matrices?

## Random Code

- $\mathcal{C} = \{mG_u : m \in \mathbb{F}_q^k\}$  where  $G_u \leftarrow \text{Unif}(\mathbb{F}_q^{k \times n})$
- or
- $\mathcal{C} = \{c \in \mathbb{F}_q^n : H_u c^T = 0\}$  where  $H_u \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$

Are the models equivalent? Do they define a random  $[n, k]$ -code?

## At first sight

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

## Random Code

- $\mathcal{C} = \{mG_u : m \in \mathbb{F}_q^k\}$  where  $G_u \leftarrow \text{Unif}(\mathbb{F}_q^{k \times n})$   
 $\rightarrow \dim \mathcal{C} \leq k$  as  $\text{rank}(G_u) \leq k$
- $\mathcal{C} = \{c \in \mathbb{F}_q^n : H_u c^T = 0\}$  where  $H_u \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$   
 $\rightarrow \dim \mathcal{C} \geq k$  as  $\text{rank}(H_u) \leq n - k$

Both models do not seem to be equivalent... (Spoil: they are!)

# An important tool: statistical distance

## Statistical distance

$X$  and  $Y$  be random variables

$$\Delta(X, Y) \stackrel{\text{def}}{=} \frac{1}{2} \sum_{a \in \mathcal{E}} |\mathbb{P}(X = a) - \mathbb{P}(Y = a)|.$$

## A crucial property: data processing inequality

$$\Delta(f(X), f(Y)) \leq \Delta(X, Y)$$

Consequence:  $\forall \mathcal{A}$  algorithm

$$|\mathbb{P}_X(\mathcal{A}(X) = \text{"success"}) - \mathbb{P}_Y(\mathcal{A}(Y) = \text{"success"})| \leq \Delta(X, Y).$$

## Same models

## Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

## A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes **$G_u$  or  $H_u$ -models  $\iff$  draw uniformly an  $[n, k]$ -code:** $G_k \in \mathbb{F}_q^{k \times n}$  ( $H_{n-k} \in \mathbb{F}_q^{(n-k) \times n}$ ) be uniform of rank  $k$  (resp.  $n - k$ ):

$$\Delta(G_u, G_k) = O\left(q^{-(n-k)}\right) \quad (\text{resp. } \Delta(H_u, H_{n-k}) = O\left(q^{-k}\right))$$

**Computation are the same in  $G_u$  and  $H_u$ -models:**Let  $\mathcal{E}$  be a set of codes (defined as an event). We have,

$$|\mathbb{P}_{G_u}(\mathcal{E}) - \mathbb{P}_{H_u}(\mathcal{E})| = O\left(q^{-\min(k, n-k)}\right).$$

DP: generator or  
parity-check?

DP'(n, q, R,  $\tau$ ). Let  $k \stackrel{\text{def}}{=} \lfloor Rn \rfloor$  and  $t \stackrel{\text{def}}{=} \lfloor \tau n \rfloor$

- Input :  $(G_u, y \stackrel{\text{def}}{=} sG_u + x)$  where  $G_u, s$  and  $x$  are uniformly distributed over  $\mathbb{F}_q^{k \times n}$ ,  $\mathbb{F}_q^k$  and words of Hamming weight  $t$ .
- Output : an error  $e \in \mathbb{F}_q^n$  of Hamming weight  $t$  such that  $y - e = mG_u$  for some  $m \in \mathbb{F}_q^k$ .

## Exercise

For any algorithm  $\mathcal{A}$  solving DP' with probability  $\varepsilon$  and time  $T$ : describe  $\mathcal{B}$  which solves DP in the same time with probability  $\geq \varepsilon - O(q^{-\min(k, n-k)})$  (and the reciprocal)

## The proof

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes(H, Hx<sup>T</sup>) be an instance of DPThe algorithm  $\mathcal{B}$ :

1. Compute with linear algebra  $G$  (rank  $k$ ) s.t  $GH^T = 0$ .
2. Compute with linear algebra  $y$  such that  $Hy^T = Hx^T$
3. Pick  $m \in \mathbb{F}_q^k$  uniformly,  $y = y + mG$
4. Feed  $(G, y)$  to  $\mathcal{A}$  and output its output

Probability of success of  $\mathcal{B}$ 

$$\begin{aligned}
 \mathbb{P}_{H_u}(\mathcal{B}(H_u, H_u x^T) = \text{"succ"}) \\
 &\geq \mathbb{P}_{H_{n-k}}(\mathcal{B}(H_{n-k}, H_{n-k} x^T) = \text{"succ"}) - \Delta(H_u, H_{n-k}) \\
 &= \mathbb{P}_{G_k}(\mathcal{A}(G_k, mG_k + x) = \text{"succ"}) - \Delta(H_u, H_{n-k}) \\
 &\geq \mathbb{P}_{G_u}(\mathcal{A}(G_u, mG_u + x) = \text{"succ"}) - \Delta(G_u, G_k) - \Delta(H_u, H_{n-k})
 \end{aligned}$$

# A first computation with random codes

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

$s$  and  $y \neq 0$  (fixed),  $H_u \leftarrow \text{Unif}(\mathbb{F}_q^{(n-k) \times n})$ , then:

$$\mathbb{P}_{H_u}(H_u y^T = s) = \frac{1}{q^{n-k}}.$$

**Proof.**On board! □

Lattice analogue:  $\frac{1}{q^{n-k}} = \frac{q^k}{q^n} = \frac{\#\mathcal{C}}{\#\mathbb{F}_q^n}$  plays the role of  $\frac{1}{|\Lambda|}$

# What do we expect?

Given  $(H, s)$  we are ready to compute:

$$N(H_u, t) = \# \{e \in \mathbb{F}_q^n : |e| = t \text{ and } H_u e^T = s\}.$$

## Proposition

We have

$$\mathbb{E}_{H_u}(N(H_u, t)) = \frac{\# \{e \in \mathbb{F}_q^n : |e| = t\}}{q^{n-k}}$$

## Proof.

On board! □

$\mathbb{E}_{H_u}(N(H_u, t))$  independent of  $s$

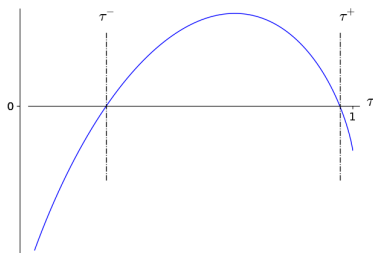
$s = 0$ : average number of codewords of weight  $t$ .

## Asymptotic behaviour

$$\# \{e \in \mathbb{F}_q^n : |e| = t\} = \binom{n}{t} (q-1)^t$$

$$\binom{n}{t} (q-1)^t = O\left(\frac{1}{\sqrt{n}}\right) q^n h\left(\frac{t}{n}\right)$$

$$h(x) \stackrel{\text{def}}{=} -x \log_q \left(\frac{x}{q-1}\right) - (1-x) \log_q (1-x).$$



**Figure:**  $\lim_{n \rightarrow +\infty} \frac{1}{n} \log_q \mathbb{E}_{H_u}(N(H_u, t))$  s.t  $q = 3$ ,  $k/n = 1/4$ , fct of  $\tau = t/n$ .

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

# Hardness of DP?

## Basic on Codes

## The Decoding Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

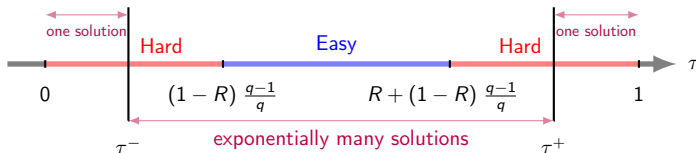
Search-to-Decision  
Reduction

## A quick recap

## About Random Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes



# Be more accurate: order 1

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

For now, only  $\mathbb{E}_{H_u}(N(H_u, t))$  is known

where  $N(H_u, t) = \#\{e \in \mathbb{F}_q^n : |e| = t \text{ and } H_u e^T = s\}$ .

Be more precise?

## Proposition (First Moment Technique)

For any  $a > 0$ ,

$$\mathbb{P}_{H_u}(N(H_u, t) > \textcolor{red}{a}) \leq \frac{1}{\textcolor{red}{a}} \frac{\binom{n}{t} (q-1)^t}{q^{n-k}}.$$

## Proof.

Markov:  $\mathbb{P}_{H_u}(N(H_u, t) > a) \leq \frac{\mathbb{E}_{H_u}(N(H_u, t))}{a}$



# Order 2?

We can be even more precise: Bienaymé-Tchebychev!  
(second moment technique)

# Expected minimum distance

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

$$\mathbb{E}_{H_u} (\#\{c : H_u c^T = 0 \text{ and } |c| = t\}) = \frac{\binom{n}{t}(q-1)^t}{q^{n-k}}$$

Expected minimum distance of  $\mathcal{C}$  defined by  $H_u$ ?

# Expected minimum distance

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

$$\mathbb{E}_{H_u} (\#\{c : H_u c^T = 0 \text{ and } |c| = t\}) = \frac{\binom{n}{t}(q-1)^t}{q^{n-k}}$$

Expected minimum distance of  $\mathcal{C}$  defined by  $H_u$ ?

## Gilbert-Varshamov distance

Smallest  $t$  such that  $\binom{n}{t}(q-1)^t = q^{n-k}$

$d_{\min}(\mathcal{C}) = t_{GV} = Cn$  for some constant  $C > 0$ .

# Balls and minimum distance (worst case)

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

$\mathcal{C}$  be a fixed code of minimum distance  $d_{\min}(\mathcal{C})$

$$\forall c, c' \in \mathcal{C}, c \neq c' : \mathcal{B}_H \left( c, \frac{d_{\min}(\mathcal{C})-1}{2} \right) \cap \mathcal{B}_H \left( c', \frac{d_{\min}(\mathcal{C})-1}{2} \right) = \emptyset$$

**Proof.**

On board!



# Balls and minimum distance (average case)

For a random code:

$$d_{\min}(\mathcal{C}) = t_{\text{GV}}$$

$\mathcal{C}$  be a random code

$$\forall c, c' \in \mathcal{C}, c \neq c' : \mathcal{B}_H(c, t_{\text{GV}}) \cap \mathcal{B}_H(c', t_{\text{GV}}) \approx \emptyset$$

Not  $\frac{t_{\text{GV}}}{2}$ !

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# Security?

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

Aim of any **code**-based cryptosystem:  
security relies on the hardness of the decoding problem (DP)

How to trust DP hardness?

→ By studying algorithms solving DP!

**An old history (since 60 years)**

Best algorithms: refinement of **Prange's** algorithm (1962)  
Information Set Decoding (ISD) algorithms

# Prange's algorithm

Our aim: describing Prange's algorithm

Two points of view:

- noisy codewords,
- syndromes and parity-check matrices.

# Prange's algorithm

Our aim: describing Prange's algorithm

Two points of view:

- noisy codewords,
- syndromes and parity-check matrices.

# Noisy codewords

- Given:  $\mathcal{C}$  an  $[n, k]$ -code and  $c + e$  where  $\begin{cases} c \in \mathcal{C} \\ |e| = t \end{cases}$
- Recover:  $e$

First remark of Prange: **Information Set!**

## Information Set

$\mathcal{I} \subseteq \{1, \dots, n\}$  of size  $k$ , information set of the  $[n, k]$ - $\mathcal{C}$  if:

$$\forall x \in \mathbb{F}_q^k : \exists (\text{unique}) c \in \mathcal{C} : c_{\mathcal{I}} = x \quad (\text{where } c_{\mathcal{I}} = (c_i)_{i \in \mathcal{I}})$$

Every codewords: uniquely determined by  $k = \dim(\mathcal{C})$  coordinates given by  $\mathcal{I}$

# Information Set

## Information Set

$\mathcal{I} \subseteq \{1, \dots, n\}$  of size  $k$ , information set of the  $[n, k]\text{-}\mathcal{C}$  if:

$$\forall x \in \mathbb{F}_q^k : \quad \exists (\text{unique}) c \in \mathcal{C} : c_{\mathcal{I}} = x$$

## Exercise

$\mathcal{I}$  inf set for  $\mathcal{C} \iff \forall G$  generator matrix of  $\mathcal{C}$ ,  $G_{\mathcal{I}}$  is invertible

$\iff \forall H$  parity-check matrix of  $\mathcal{C}$ ,  $H_{\overline{\mathcal{I}}}$  is invertible

$M_{\mathcal{I}}$  matrix whose columns are those of  $M$  which are indexed by  $\mathcal{I}$ .

# Information Set

## Information Set

$\mathcal{I} \subseteq \{1, \dots, n\}$  of size  $k$ , information set of the  $[n, k]$ - $\mathcal{C}$  if:

$$\forall x \in \mathbb{F}_q^k : \quad \exists (\text{unique}) c \in \mathcal{C} : c_{\mathcal{I}} = x$$

## Exercise

$\mathcal{I}$  inf set for  $\mathcal{C} \iff \forall G$  generator matrix of  $\mathcal{C}$ ,  $G_{\mathcal{I}}$  is invertible

$\iff \forall H$  parity-check matrix of  $\mathcal{C}$ ,  $H_{\overline{\mathcal{I}}}$  is invertible

$M_{\mathcal{I}}$  matrix whose columns are those of  $M$  which are indexed by  $\mathcal{I}$ .

$$\forall x \in \mathbb{F}_q^k : \quad \exists (\text{unique}) c \in \mathcal{C} \text{ that we compute easily} : c_{\mathcal{I}} = x$$

# Prange's algorithm

- Given:  $\mathcal{C}$  an  $[n, k]$ -code and  $y \stackrel{\text{def}}{=} c^{\text{sol}} + e^{\text{sol}}$  where  $\begin{cases} c^{\text{sol}} \in \mathcal{C} \\ |e^{\text{sol}}| = t \end{cases}$
- Recover:  $e^{\text{sol}}$

- Pick an information set  $\mathcal{I}$ ,
- Compute the unique  $c \in \mathcal{C}$  s.t

$$c_{\mathcal{I}} = y_{\mathcal{I}}$$

- You win if  $|y - c| = t$ , namely

$$y_{\mathcal{I}} = c_{\mathcal{I}}^{\text{sol}} \iff e_{\mathcal{I}}^{\text{sol}} = 0.$$

Otherwise, go back to 1.

Complexity of the algorithm: number of times we pick  $\mathcal{I}$

# Prange's algorithm

Our aim: describing Prange's algorithm

Two points of view:

- noisy codewords,
- syndromes and parity-check matrices.

# Prange's algorithm

Our aim: describing Prange's algorithm

Two points of view:

- noisy codewords,
- syndromes and parity-check matrices.

# Syndromes and parity-check matrices

Fixing  $(H, s \stackrel{\text{def}}{=} He^T)$  where  $|e| = t$ .

→ Linear system:  $n - k$  equations and  $n$  unknowns

$$(H \in \mathbb{F}_q^{(n-k) \times n})$$

But...

# Syndromes and parity-check matrices

Fixing  $(H, s \stackrel{\text{def}}{=} He^T)$  where  $|e| = t$ .

→ Linear system:  $n - k$  equations and  $n$  unknowns

$$(H \in \mathbb{F}_q^{(n-k) \times n})$$

But...

with a non-linear constraint  $(|e| = t)$

# Syndromes and parity-check matrices

Fixing  $(H, s \stackrel{\text{def}}{=} He^T)$  where  $|e| = t$ .

→ Linear system:  $n - k$  equations and  $n$  unknowns

$$(H \in \mathbb{F}_q^{(n-k) \times n})$$

But...

with a non-linear constraint  $(|e| = t)$

Prange's algorithm:

fixing  $k$  unknowns,

solving a square  $(n - k) \times (n - k)$  linear system,  
hoping the solution has the good Hamming weight.

# Extended Prange's algorithm

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

1. *Picking the information set.*

$\mathcal{I}$  of size  $k$ . If  $H_{\mathcal{I}} \in \mathbb{F}_q^{(n-k) \times (n-k)}$  is not of full-rank, pick another  $\mathcal{I}$ .

2. *Linear algebra.*

$S$  non-singular s.t  $SH_{\mathcal{I}} = 1_{n-k}$  (Gaussian elimination).

3. *Test Step.*

$x \in \mathbb{F}_q^k$  and  $e \in \mathbb{F}_q^n$  be s.t

$$e_{\mathcal{I}} = (s - xH_{\mathcal{I}}^T) S^T \quad ; \quad e_{\mathcal{I}} = x. \quad (1)$$

If  $|e| \neq t$  go back to Step 1, otherwise it is a solution.

**Correction**

On board!

# Extended Prange's algorithm

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## Exercise

Describe Prange's algorithm with generator matrices, three steps and the vector  $x$ .

# Hardness of DP(I)

Each iteration: we test if  $|e| = t$  where

$$e_{\overline{I}} = (s - xH_{\overline{I}}^T) S^T \in \mathbb{F}_q^{n-k} \quad ; \quad e_I = x.$$

Suppose  $s$  uniformly distributed, then:

$$\mathbb{E}(|e|) = |x| + \frac{q-1}{q} (n-k).$$

# Hardness of DP(I)

Each iteration: we test if  $|e| = t$  where

$$e_{\overline{I}} = (s - xH_{\overline{I}}^T) S^T \in \mathbb{F}_q^{n-k} \quad ; \quad e_I = x.$$

Suppose  $s$  uniformly distributed, then:

$$\mathbb{E}(|e|) = |x| + \frac{q-1}{q} (n-k).$$

Carefully choosing  $|x| \in \llbracket 0, k \rrbracket$  ( $k$  number of unknowns we can fix)  
we can easily reach any Hamming weight in

$$\left[ \frac{q-1}{q} (n-k), k + \frac{q-1}{q} (n-k) \right].$$

# Hardness of DP(II)

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

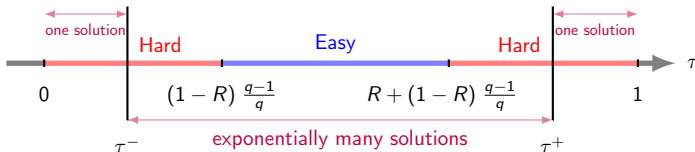
Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

$$k = Rn \quad \text{and} \quad t = \tau n$$

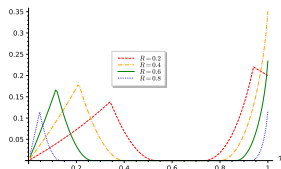
- $\frac{q-1}{q}(n-k) = n \frac{q-1}{q} (1-R),$
- $k + \frac{q-1}{q}(n-k) = n \left( R + \frac{q-1}{q} (1-R) \right).$



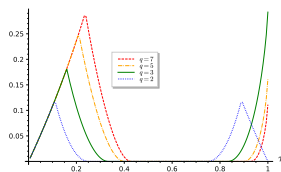
Since 60 years:

no known poly-time algorithm in the **red** area (even quantumly)

# Asymptotic Exponent



**Figure:** Exponent  $\alpha(\tau)$  of Prange's algorithm complexity  $2^{\alpha(\tau)n}$  to solve  $\text{DP}(n, q, R, \tau)$  for  $q = 3$  as function of  $\tau$ .



**Figure:** Exponent  $\alpha(\tau)$  of Prange's algorithm complexity  $2^{\alpha(\tau)n}$  to solve  $\text{DP}(n, q, R, \tau)$  for  $R = 1/2$  as function of  $\tau$ .

## 1 Basic on Codes

## 2 The Decoding Problem

Worst-case

An easy case: Reed-Solomon codes

Average-case

Search-to-Decision Reduction

## 3 A quick recap

## 4 About Random Codes

## 5 Prange's Algorithm

## 6 Public-key Encryption Schemes

# McEliece's Encryption

## Key Generation

- $(G_{pk}, t, T) \leftarrow \text{Trappdoor}()$  where  $G_{pk}$  represents a code s.t

$$(mG_{pk} + e, T) \xrightarrow{\text{easy}} m \quad (\text{if } |e| \leq t)$$

- Secret Key:  $T$
- Public Key:  $G_{pk}$

## Encryption of $m$

Pick random  $e \in \{z : |z| = t\}$  and output

$$mG_{pk} + e$$

## Decryption of $mG_{pk} + e$

Use  $T$  to compute

$$(mG_{pk} + e, T) \longrightarrow m$$

# Security of McEliece

## McEliece

pk:  $G_{pk}$  representation of a code, sk: a trapdoor  $T$

The security of McEliece relies on 2 assumptions:

1. The hardness of DP,
2. We can't distinguish  $G_{pk}$  and  $G_u$  (uniform).

*Can we distinguish the public code from a random one?  
Be extremely careful...*

## An instantiation

Codes that we know how to decode:  $\text{GRS}_k(x, z)$

- Public Key: a representation of  $\text{GRS}_k(x, z)$

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^k & x_2^k & \cdots & x_n^k \end{pmatrix} \begin{pmatrix} z_1 & & & 0 \\ & z_2 & & \\ & & \ddots & \\ 0 & & & z_n \end{pmatrix}$$

- Secret Key:

What is the secret key? Can we give the above matrix as a public key?

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
CodesPrange's  
AlgorithmPublic-key  
Encryption  
Schemes

# An instantiation

Codes that we know how to decode:  $\text{GRS}_k(x, z)$

- Public Key: a representation of  $\text{GRS}_k(x, z)$

$$G_{\text{pk}} = S \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^k & x_2^k & \cdots & x_n^k \end{pmatrix} \begin{pmatrix} z_1 & & & 0 \\ & z_2 & & \\ & & \ddots & \\ 0 & & & z_n \end{pmatrix}$$

- Secret Key:  $T = (x, z)$

This scheme is broken: exercise 1 in sheet 2

# A bad (but original) presentation of McEliece

[https://en.wikipedia.org/wiki/McEliece\\_cryptosystem](https://en.wikipedia.org/wiki/McEliece_cryptosystem)

There are no permutations in McEliece  
cryptosystem

# Don't forget Alekhnovich

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

Alekhnovich like encryption scheme:

Security does not rely on “structured” codes

# Codes at the NIST

Basic on Codes

The Decoding  
Problem

Worst-case

An easy case:  
Reed-Solomon codes

Average-case

Search-to-Decision  
Reduction

A quick recap

About Random  
Codes

Prange's  
Algorithm

Public-key  
Encryption  
Schemes

## McEliece

- Classic McEliece: Goppa codes,
- BIKE: QC-MDPC codes.

## Alekhnovich

- HQC: does not use structured codes as trapdoor.

# Conclusion

Many other topics:

- Search-to-decision reductions, average to average reductions using DP, ...
- Code-based primitives like signatures,
- Change the Hamming metric (rank metric, Exercise Sheet 2)
- etc...

If you are interested by the code-based crypto: lecture notes available here <http://tdalazard.io/>

# Conclusion

Many other topics:

- Search-to-decision reductions, average to average reductions using DP, ...
- Code-based primitives like signatures,
- Change the Hamming metric (rank metric, Exercise Sheet 2)
- etc...

If you are interested by the code-based crypto: lecture notes  
available here <http://tdalazard.io/>

# Thank You!

# About LPN

## Problem (Learning with Parity Noise Problem)

- Oracle: An oracle  $\mathcal{O}_{s,\tau}(\cdot)$  parametrized by  $s$  and  $\tau$  s.t on a call it outputs  $(a, s \cdot a + e)$  where  $a \leftarrow \text{Unif}(\mathbb{F}_2^k)$  and  $e$  Bernoulli of parameter  $\tau$ .
- Input:  $\mathcal{O}_{s,\tau}(\cdot)$
- Output:  $s$

Is it a decoding problem using codes?

# About LPN

## Problem (Learning with Parity Noise Problem)

- Oracle: An oracle  $\mathcal{O}_{s,\tau}(\cdot)$  parametrized by  $s$  and  $\tau$  s.t on a call it outputs  $(a, s \cdot a + e)$  where  $a \leftarrow \text{Unif}(\mathbb{F}_2^k)$  and  $e$  Bernoulli of parameter  $\tau$ .
- Input:  $\mathcal{O}_{s,\tau}(\cdot)$
- Output:  $s$

Is it a decoding problem using codes?

Yes! But be careful, there are differences with DP...

# In practice: DP not LPN

$n$  calls to the oracle  $\mathcal{O}_{s,\tau}(\cdot)$ :

$$\langle s, a_1 \rangle + e_1, \dots, \langle s, a_n \rangle + e_n.$$

These  $n$  samples can be rewritten as  $sG + e$  where columns of  $G \in \mathbb{F}_2^{k \times n}$  are the  $a_i$ 's and  $e \stackrel{\text{def}}{=} (e_1, \dots, e_n)$ .

$n$  is unlimited!

# In practice: DP not LPN

$n$  calls to the oracle  $\mathcal{O}_{s,\tau}(\cdot)$ :

$$\langle s, a_1 \rangle + e_1, \dots, \langle s, a_n \rangle + e_n.$$

These  $n$  samples can be rewritten as  $sG + e$  where columns of  $G \in \mathbb{F}_2^{k \times n}$  are the  $a_i$ 's and  $e \stackrel{\text{def}}{=} (e_1, \dots, e_n)$ .

$n$  is unlimited!

- DP: **fixed** number of samples  
problem used to design encryption or signature schemes, ensure the security
- LPN: **unlimited** number of samples  
problem **not used** to design encryption or signature schemes, sometimes useful in reductions