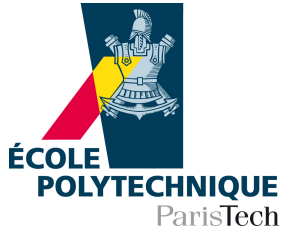
Sur l'algorithme de décodage en liste de Guruswami-Sudan sur les anneaux finis.

Guillaume Quintin

Équipe Grace,
INRIA SACLAY–Île-de-France,
Laboratoire d'informatique (LIX, UMR 7161),
École polytechnique.

Soutenance de thèse, le 22 novembre 2012



Sommaire

Ma Thèse a été dirigée par **Daniel Augot** et **Grégoire Lecerf**, cofinancée par

- L'institut national de recherche en informatique et automatique (INRIA) et la
- Direction générale de l'armement (DGA).

Plan :

Introduction générale.

Contexte : transmission

- La **transmission physique d'information** (ondes radios, fibre optique) est souvent accompagnée **d'erreurs**.
- Est-il possible de tolérer des erreurs ? Oui, dans une certaine limite.

Comment faire ?

- L'émetteur du message rajoute de la **redondance** à ce message.
  $\Longrightarrow$ L'émetteur encode le message.

- Le récepteur, si erreurs, peut corriger ces erreurs.
  $\Longrightarrow$ Le récepteur décode.

Les codes correcteurs d'erreurs

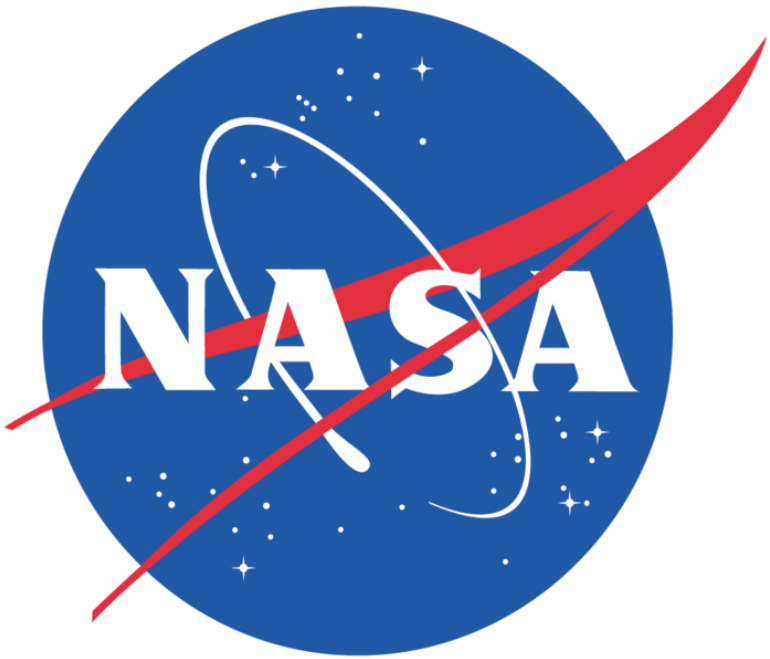
- La manière d'ajouter de la redondance s'appelle un **code correcteur d'erreurs**.
- Il existe **plusieurs familles** de codes correcteurs d'erreurs.
- On se concentre ici sur les **codes correcteurs algébriques**.
- Parmi les codes linéaires, ma thèse a porté sur l'étude des codes dits de Reed-Solomon.

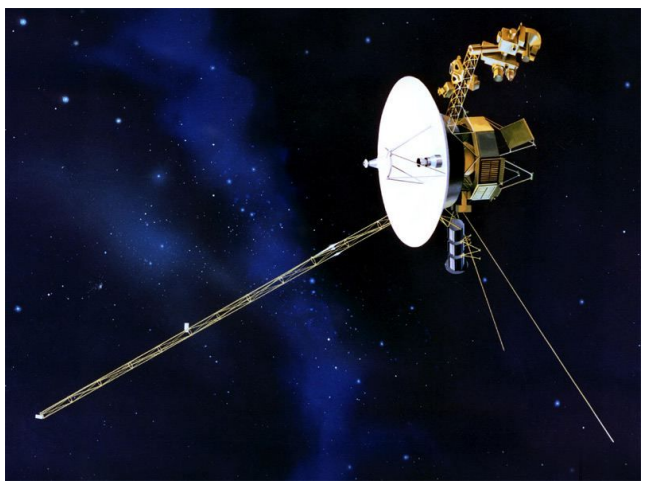
En pratique, sur terre

On retrouve des codes de **Reed-Solomon** dans les **boîtiers ADSL** pour se connecter à **Internet**.

En pratique, dans l'espace

Les sondes **Voyager** de la **NASA** utilisaient un code correcteur de **Reed-Solomon** pour communiquer avec la terre.

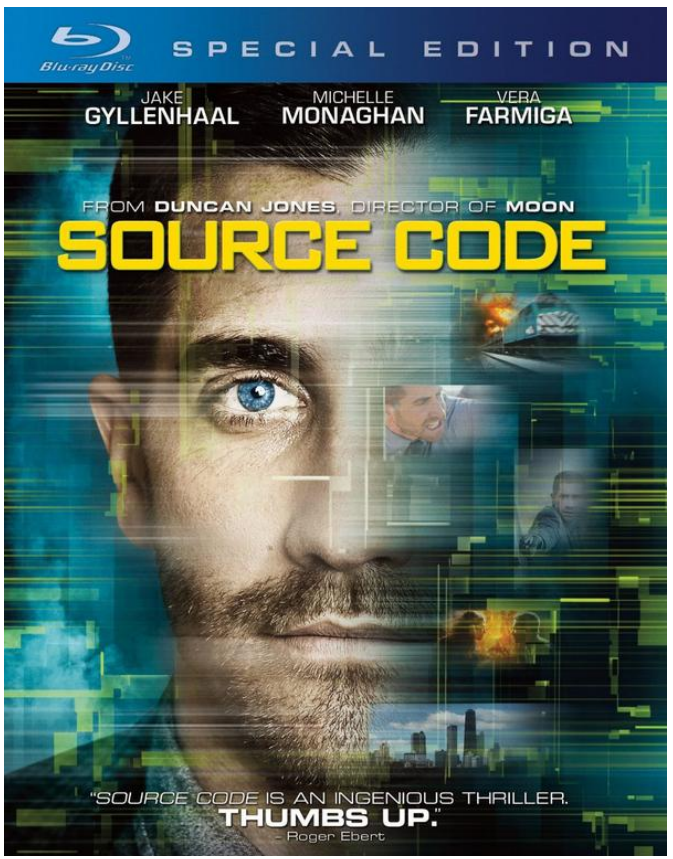




On retrouve aussi des codes de **Reed-Solomon** pour les missions **Mars Pathfinder**, **Galileo**, **Mars Exploration Rover** et **Cassini**.

Pour stocker des données, des films, des jeux vidéos

Sur les **CDs**, **DVDs** et **Blu-ray**, l'information est stockée en utilisant, entre autre, un **encodage** de **Reed-Solomon**.



Dans la vie de tous les jours

Les **codes QR** sont des code-barres à deux dimensions et utilisent un code de **Reed-Solomon**.

Ils sont utilisés dans les **musées** pour obtenir des informations complémentaires, par la **SNCF** pour ses billets, et dans beaucoup d'autres endroits.

Un exemple simple : le code à répétition

- On veut transmettre de l'information composée de **0** et de **1**.
- Une erreur est donc :
  - Un **0** qui se transforme en **1**.
  - Un **1** qui se transforme en **0**.

L'encodage :

- Chaque **symbole** est répété trois fois $\begin{cases} \mathbf{0} \longrightarrow \mathbf{000} \\ \mathbf{1} \longrightarrow \mathbf{111} \end{cases}$

  et **000** ou **111** est transmis.

Le décodage :

- On reçoit un groupe de **3 symboles**.
- On **considère** que le symbole original est celui qui apparaît le plus souvent dans le groupe.
  Exemple : $\mathbf{110} \rightarrow \mathbf{1}$ et $\mathbf{010} \rightarrow \mathbf{0}$.

Entrelacer pour corriger des erreurs en rafale

- **L'émetteur** veut transmettre 010.
- Il **encode** 010 en 000111000.
- Mais si une **erreur en rafale** arrive, par exemple 000 000 000, impossible de retrouver le message original.

- On **entrelace** les symboles :

$$000111000 \mapsto \begin{bmatrix} 000 \\ 111 \\ 000 \end{bmatrix} \mapsto \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mapsto 010\ 010\ 010$$

- Ainsi si une **erreur en rafale** se produit :

$$010\ 101\ 010 \mapsto \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \mapsto 0\,1\,0\,1\,0\,1\,0\,1\,0$$

Le poids et la distance de Hamming

- On fixe un anneau fini A et $n > 0$.
- Soit $x = (x_1, \ldots, x_n) \in A^n$.
- x est appelé un **mot**.
- Le **poids de Hamming** de x, $w(x)$, est le nombre de coordonnées non nulles de x.

$$w(0, 0, 1, 0, 1, 1, 0, 1) = 4.$$

- La **distance de Hamming entre** x et $y \in A^n$, $d(x, y)$, est définie comme $w(x - y)$.

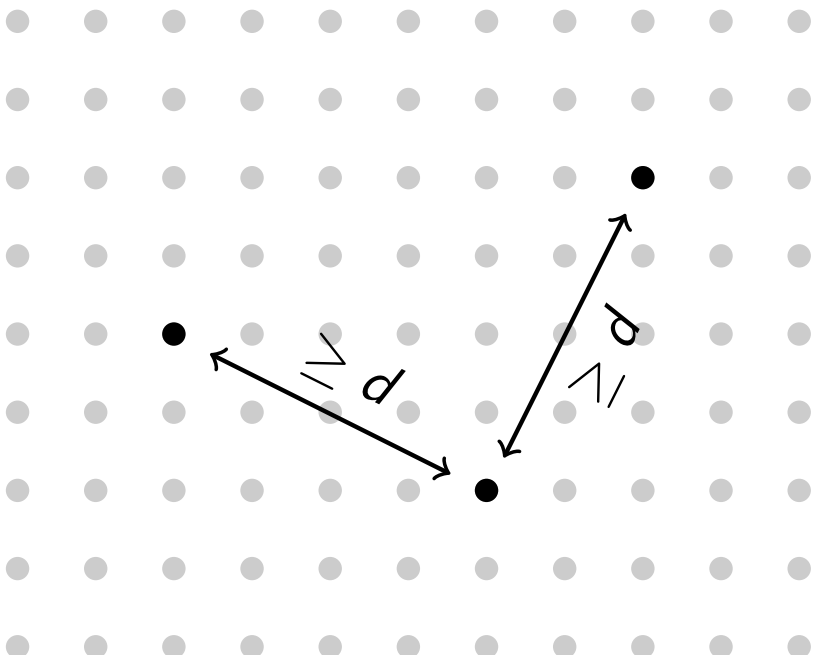
$$d\left((0, 0, 1, 0, 1, 1, 0, 1), (0, 0, 0, 1, 1, 1, 1, 0)\right) = 4.$$

Les codes correcteurs linéaires

- Soit $\mathcal{C}$ un sous-module **libre** à gauche de A^n de rang k.
- $\mathcal{C}$ est appelé un **code linéaire** et ses éléments des **mots de code**.
- La **distance minimale** de $\mathcal{C}$ est

$$d := \min\{d(x,y) : x, y \in \mathcal{C} \text{ et } x \neq y\}.$$

- **Borne de Singleton** : $d \leq n - k + 1$.



mot (de A^n)

mot de code (de $\mathcal{C}$)

L'encodage

- Soit $m \in A^k$, un **message**.
- On a une **injection** $\varphi : A^k \to A^n$ telle que $\varphi(A^k) = \mathcal{C}$.
- φ est une **fonction d'encodage**.

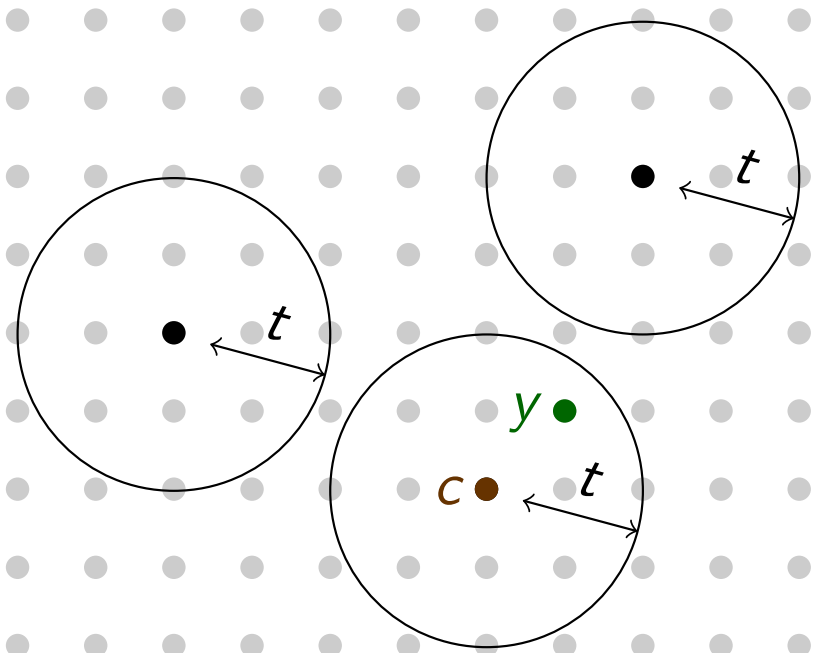
$$\varphi(m) = c \in \mathcal{C}.$$

- La transposée G d'une matrice de φ est appelée **matrice génératrice** de $\mathcal{C}$ et permet d'encoder les messages par

$$mG = c.$$

Le décodage unique

- Les boules de rayon $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ centrées en les mots de codes sont **disjointes**.
- Si le mot reçu y est à distance au plus t d'un mot de code c alors on retourne c.



mot (de A^n)

mot de code (de $\mathcal{C}$)

Les codes de Reed-Solomon sur les corps finis

- Considérons $A = \mathbb{F}_q$ et deux entiers $0 < k < n$.
- Choisissons un **support** $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$ tel que $\forall i \neq j \quad x_i \neq x_j$.
- Notons $\mathbb{F}_q[X]_{<k}$ les polynômes de degré au plus $k-1$.
- Le sous-espace vectoriel de $\mathbb{F}_q^n$ engendré par les vecteurs

$$(f(x_1), \ldots, f(x_n)) \text{ avec } f \in \mathbb{F}_q[X]_{<k}$$

  est appelé le code de **Reed-Solomon de paramètres** $(x_1, \ldots, x_n)$ **et** k.
- Sa distance minimale est $d = n - k + 1$.
  C'est un code à **distance séparable maximale (MDS)**.

Décodage unique pour les code de Reed-Solomon

Le **décodage unique** des codes de Reed-Solomon a été étudié et des algorithmes efficaces ont été donnés.

- Peterson [Pet60].
- Berlekamp [Ber68].
- Massey [Mas69].
- Sugiyama, Kasahara, Hirasawa et Namekawa [SKHN75].
- Berlekamp, Welch [BW86].
- ...

Complexité

Pour fixer les idées.

- Les complexités énoncées comptent le **nombres d'opérations** dans A sauf mention expresse du contraire.

- La notation "soft-Oh" $f(n) \in \widetilde{O}(g(n))$ signifie

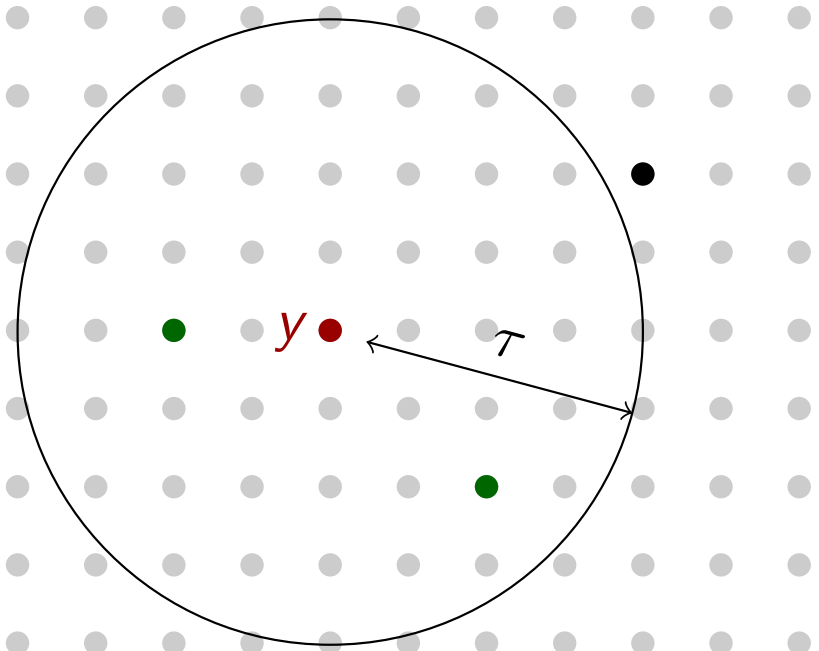
$$f(n) \in g(n) \log^{O(1)}(3 + g(n)).$$

- On note ω un réel positif tel que le nombre d'opérations dans A nécessaire pour multiplier deux matrices de taille $O(n^2)$ est $O(n^\omega)$. On peut prendre
  - $\omega = 3$ (algorithme naïf),
  - $\omega = 2,808$ (Strassen, 1969),
  - $\omega = 2,376$ (Coppersmith et Winograd, 1990) ou
  - $\omega = 2,3727$ (Williams, 2011).

Le décodage en liste et les codes de Reed-Solomon.

Le décodage en liste (1)

On change de stratégie : on cherche si il y a des mots de code à distance au plus τ du mot reçu y.



- mot (de A^n)
- mot de code (de $\mathcal{C}$)

Il est possible que plusieurs mots de code se retrouvent dans la boule.

Le décodage en liste (2)

Deux questions naturelles se posent alors pour un code de paramètres $[n, k, d]_A$.

1. **Quelle doit être la valeur de τ ?**
   - Si $\tau < n - \sqrt{n(n-d)}$, alors il ne peut y avoir qu'un nombre polynomial en n de mots de code dans la boule [Gur04].
   - La borne $n - \sqrt{n(n-d)}$ est appelée **borne de Johnson**.

2. **Si on trouve plusieurs mots de code, lequel choisir ?**
   - Pour les codes de Reed-Solomon, la probabilité de trouver plusieurs mots de code dans la boule est très petite [NH00].
     - $[8, 4, 5]_9$: $1,386 \times 10^{-6}$.
     - $[8, 4, 5]_{729}$: $4,759 \times 10^{-18}$.

Les algorithmes de décodages

Soit un code de paramètres $[n, k, d]_{\mathbb{F}_q}$.

- On ne connait pas d'algorithme polynomial en n, k, d et $\log q$ pour le **décodage unique**.
- En fait, on **ne sait pas** calculer la distance minimale d (codes BCH).
- On **ne connait pas** d'algorithme de décodage en liste en temps polynomial en n, k, d et $\log q$.
- **Une exception notable** sont les codes de **Reed-Solomon** pour lesquels il existe un **algorithme de décodage en liste** en temps polynomial en moyenne en n, k, d et $\log q$.

L'algorithme de Guruswami-Sudan

Soit $\mathcal{C}$ un code de Reed-Solomon de paramètres $[n, k, d = n - k + 1]_{\mathbb{F}_q}$.

Entrée : Un mot reçu $y = (y_1, \ldots, y_n)$ et un rayon $\tau < n - \sqrt{n(n-d)}$.
Sortie : Tous les mots de code dans la boule centrée en y de rayon τ.

3. Trouver une courbe $Q(X, Y)$ dans $\mathbb{A}^2_{\mathbb{F}_q}$ (**interpolation**) telle que :
   - la courbe passe par tous les points (x_i, y_i) avec, au moins, une multiplicité dépendante de n, k et τ, et que
   - la courbe soit de degré borné dépendant de n, k et τ.
4. Trouver les composantes irréductibles de la courbe $Q(X, Y)$ de la forme $Y - f(X)$ (**recherche de racines**).
5. Retourner les composantes de degré au plus $k - 1$ à bonne distance de y.

L'algorithme de Guruswami-Sudan en détails

Soit $\mathcal{C}$ un code de Reed-Solomon de paramètres $[n, k, d = n - k + 1]_{\mathbb{F}_q}$.

Entrée : Un mot reçu $y = (y_1, \ldots, y_n)$ et un rayon $\tau < n - \sqrt{n(n-d)}$.

Sortie : Tous les mots de code dans la boule centrée en y de rayon τ.

1. $m \leftarrow \left\lfloor \frac{(k-1)n + \sqrt{(k-1)^2 n^2 + 4((n-\tau)^2 - (k-1)n)}}{2((n-\tau)^2 - (k-1)n)} \right\rfloor + 1.$
2. $L \leftarrow \left\lceil \frac{m(n-\tau)-1}{k-1} \right\rceil - 1.$
3. Trouver $Q(X, Y) = \sum_{i=0}^{L} Q_i(X) Y^i \in \mathbb{F}_q[X, Y]$ tel que :
   - $Q(X + x_i, Y + y_i)$ a valuation au moins m et
   - $\deg Q_i \leq m(n - \tau) - 1 - i(k-1)$.
4. $Z \leftarrow$ les racines de Q vu dans $(\mathbb{F}_q[[X]])[Y]$.
5. Retourner $\{f \in Z \cap \mathbb{F}_q[X]_{<k} : Q(X, f(X)) = 0 \text{ et } d(f(x), y) \leq \tau\}$.

La complexité de l'algorithme de Guruswami-Sudan

- L'étape dominante en pratique est l'**interpolation** (étape 3).
  - Plusieurs algorithmes sont proposés pour résoudre cette étape :
    - **Roth** et **Ruckenstein** [RR98],
    - **Alekhnovich** [Ale05] et
    - **Zeh**, **Gentner** et **Augot** [ZGA11].
  - L'algorithme de **Kötter** [Köt96] permet aussi de résoudre cette étape et a une complexité de $\widetilde{O}\left(\frac{n^6 k^4}{(n-\sqrt{kn}-\tau)^4}\right)$.
- L'étape de **recherche de racines** (étape 4) est moins coûteuse en pratique et peut-être résolue avec une complexité de $\widetilde{O}\left(\frac{n^4 k^2}{(n-\sqrt{kn}-\tau)^2}\right)$.

Quelques motivations pour travailler sur les anneaux.

Les codes de Reed-Solomon sur les anneaux

La définition est (presque) la même que sur les corps. Pour un anneau A on note

- $A^\times$ le **groupe des unités** de A et
- $Z(A)$ le **centre** de A.

- Choisissons un **support** $(x_1, \ldots, x_n) \in A^n$ tel que

$$\forall i \neq j \quad (x_i - x_j) \in A^\times \text{ et } x_i x_j = x_j x_i.$$

- Notons $A[X]_{<k}$ les polynômes de degré au plus $k-1$.
- Le sous-module à gauche de A^n engendré par les vecteurs

$$(f(x_1), \ldots, f(x_n)) \text{ avec } f \in A[X]_{<k}$$

est appelé le code de **Reed-Solomon de paramètres** $(x_1, \ldots, x_n)$ **et** k.

Résultats théoriques

Proposition (Barbier, Chabot, Quintin [BCQ12a])

Un code de Reed-Solomon $\mathcal{C}$ sur un anneau A de longeur n et de rang k a pour distance minimale $n - k + 1$.

Théorème (Barbier, Chabot, Quintin [BCQ12a])

Il existe un algorithme de décodage unique pour $\mathcal{C}$.

Si $\forall i \quad x_i \in Z(A)$, alors il existe un algorithme de décodage en liste qui peut corriger jusqu'à $\left\lceil n - \sqrt{n(k-1)} \right\rceil - 1$ erreurs.

Théorème (Barbier, Chabot, Quintin [BCQ12a])

Soit $q = |A|$. Alors il existe un anneau commutatif B tel que $|B| = |A|$ et un code de Reed-Solomon $\mathcal{C}_B$ de longueur n et de dimension k sur B.

Les codes quasi-cycliques sur les corps finis (1)

Soit $n = m\ell$, on dit que $\mathcal{C} \subseteq \mathbb{F}_q^n$ est ℓ-**quasi-cyclique** si

$$(x_1, \ldots, x_\ell, \ldots, x_{\ell+1}, \ldots, x_{2\ell}, x_{n-\ell+1}, \ldots, x_n) \in \mathcal{C}$$
$$\Rightarrow (x_{n-\ell+1}, \ldots, x_n, x_1, \ldots, x_\ell, \ldots, x_{\ell+1}, \ldots, x_{2\ell}) \in \mathcal{C}.$$

- Leur structure a été étudiée par :
  - Lally and Fitzpatrick [LF01],
  - Ling and Solé [LS01] et
  - Cayrel, Chabot and Necer [CCN10].
- Application au cryptosystème de McEliece :
  - Berger, Cayrel, Gaborit and Otmani [BCGO09].

Pour certains $n, k \in \mathbb{N}$ les codes quasi-cycliques **donnent** des codes ayant la plus **grande distance minimale connue** [Gra07].

Les codes quasi-cycliques sur les corps finis (2)

Théorème (Barbier, Chabot, Quintin [BCQ12b])

Soit $n = m\ell$. Les codes ℓ-quasi-cycliques sont en correspondance biunivoque avec les idéaux à gauche de l'anneau $M_{\ell\times\ell}(\mathbb{F}_q)[X]/(X^m - 1)$.

Comme les codes cycliques, ils possèdent un **polynôme générateur**, qui engendre l'idéal correspondant de $M_{\ell\times\ell}(\mathbb{F}_q)[X]/(X^m - 1)$.

Théorème (Barbier, Chabot, Quintin)

Soit un code ℓ-quasi-cyclique de dimension k. Son polynôme générateur peut être calculé avec une complexité de $\widetilde{O}(k^{\omega-1}n + m\ell^\omega)$.

Racine primitive m-ième de l'unité

On appelle racine primitive m-ième de l'unité toute matrice $\Gamma \in M_{\ell\times\ell}(\mathbb{F}_{q^s})$ vérifiant:

- $\Gamma^m = Id_\ell$,
- $\forall 0 < i < m, \quad \Gamma^i \neq Id_\ell$ et
- $\forall 0 \leq i \neq j < m, \quad \det(\Gamma^i - \Gamma^j) \neq 0.$

Proposition

Il existe au moins une racine primitive $(q^{s\ell} - 1)$-ième de l'unité dans $M_{\ell\times\ell}(\mathbb{F}_{q^s})$.

Définition des codes quasi-BCH

Définition (Barbier, Chabot, Quintin [BCQ12b])

Soit $\Gamma \in M_{\ell\times\ell}(\mathbb{F}_{q^s})$ *une racine primitive* m*-ième de l'unité et* $\delta > 0$*. Le code* **quasi-BCH** *par rapport à* Γ *de distance contruite* δ *est*

$$\text{Q-BCH}(\Gamma,\delta) := \left\{ (c_1,\dots,c_m) \in \left(\mathbb{F}_q^\ell\right)^m : \sum_{j=0}^{m-1} (\Gamma^i)^j c_{j+1} = 0 \text{ pour } i = 1,\dots,\delta-1 \right\}.$$

Théorème (Barbier, Chabot, Quintin [BCQ12b])

Le code Q-BCH(Γ,δ) *a pour paramètres sur* $\mathbb{F}_q$

$$[m\ell, \geq (m - s(\delta-1))\ell, \geq \delta]_{\mathbb{F}_q}.$$

Propriétés des codes quasi-BCH

Théorème (Barbier, Quintin)

Soit un code quasi-BCH par rapport à $\Gamma \in M_{\ell\times\ell}(\mathbb{F}_{q^s})$ *de distance construite* δ*.*

Alors il existe

- *un code de Reed-Solomon* $\mathcal{C}$ *de paramètres* $[m, m-\delta+1, \delta]$ *sur l'anneau* $M_{\ell\times\ell}(\mathbb{F}_{q^s})$ *et*
- *une application* ϕ : Q-BCH$(\Gamma, \delta) \to \mathcal{C}$, $\mathbb{F}_q$*-linéaire injective isométrique.*

- On peut espérer **déduire des propriétés** pour Q-BCH(Γ, δ) de $\mathcal{C}$.
- Et, de même, déduire un **algorithme de décodage** pour Q-BCH(Γ, δ).

Les codes entrelacés (1)

- Dans la pratique une erreur n'arrive pas seule, mais plusieurs erreurs affectent **des symboles consécutifs**.
- On veut transmettre les $(r-1)$ mots de codes $c_0, \ldots, c_{r-1}$.

$c_{0,1}$	$c_{0,2}$	$\cdots$	$c_{0,n}$	$\rightarrow$	c_0
$c_{1,1}$	$c_{1,2}$	$\cdots$	$c_{1,n}$	$\rightarrow$	c_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
$c_{r-1,1}$	$c_{r-1,2}$	$\cdots$	$c_{r-1,n}$	$\rightarrow$	c_{r-1}
$\downarrow$	$\downarrow$		$\downarrow$		
s_1	s_2		s_n		

- Au lieu de transmettre $c_1, \ldots, c_{r-1}$, on transmet $s_1, \ldots, s_n$.
- On définit une **erreur en rafale** comme étant au plus r **erreurs arrivant sur toute une colonne** s_i.
- Dans ce contexte **une erreur en rafale** ne cause seulement **qu'une seule erreur sur chaque** c_i.

Les codes entrelacés (2)

Les codes entrelacés à base de codes de Reed-Solomon ont été étudiés entre autre par

- **Bleichenbacher**, **Kiayias** and **Yung** [BKY03].
- **Coppersmith** et **Sudan** [CS03].
- **Gopalan**, **Guruswami** et **Raghavendra** [GGR11].

Soit $\mathcal{C}$ un code sur l'anneau A. On dénote par $E_{\mathcal{C}}$ l'ensemble des matrices

$$\begin{pmatrix} c_{0,1} & c_{0,2} & \cdots & c_{0,n} \\ c_{1,1} & c_{1,2} & \cdots & c_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{r-1,1} & c_{r-1,2} & \cdots & c_{r-1,n} \end{pmatrix}$$

telles que $(c_{i,1}, \ldots, c_{i,n}) \in \mathcal{C}$ pour $i = 1, \ldots, r-1$.
Une **erreur en rafale** est donc un ensemble d'au plus $(r-1)$ erreurs sur **toute une colonne**.

Le lien avec les codes sur les anneaux

Proposition (Quintin [Qui12a])

Soit $\mathcal{C}$ le code tel que $c_0, \ldots, c_{r-1} \in \mathcal{C}$ et, pour $i = 1, \ldots, n$,

$$S_i(t) = c_{0,i} + c_{1,i}t + c_{2,i}t^2 + \cdots + c_{r-1,i}t^{r-1} \in \mathbb{F}_q[[t]]/(t^r).$$

Alors le nombre d'erreurs en rafale arrivant à $s_1, \ldots, s_n$ est égal au poids de Hamming de la même erreur arrivant à $S_1, \ldots, S_n$.

Si de plus $\mathcal{C}$ est un code de Reed-Solomon, alors le mot $(S_1, \ldots, S_n)$ est un mot de code d'un code de Reed-Solomon $\mathcal{C}_A$ sur l'anneau $\mathbb{F}_q[[t]]/(t^r)$.

Corollary (Quintin [Qui12a])

Tout algorithme de décodage pour $\mathcal{C}_A$ induit un algorithme de décodage pour $E_{\mathcal{C}}$.

Pour résumer

- Les codes sur les corps sont les plus étudiés et utilisés.
- Plusieurs familles de **codes sur les corps** peuvent être étudiées avec des codes sur des anneaux :
  - Les **codes quasi-cycliques** $\Longrightarrow$ codes de Reed-Solomon sur des anneaux non commutatifs.
  - Les **codes entrelacés** $\Longrightarrow$ codes sur l'anneau des séries tronquées.
- Aucun algorithme détaillé n'a été donné pour le décodage des codes de Reed-Solomon sur les anneaux.

L'algorithmique pour les codes correcteurs sur les anneaux.

L'algorithme de Guruswami-Sudan sur B

- Soit A un anneau de **valuation discrète** et π une **uniformisante** de A.
- Soit $\mathcal{C}$ un code de Reed-Solomon de paramètres $[n, k, d = n - k + 1]_A$.

Proposition

Pour tout $r \in \mathbb{N}$, $\mathcal{C}/\pi^r\mathcal{C}$ est un code de Reed-Solomon sur $B = A/(\pi^r)$ de paramètres $[n, k, d]_B$.

En particulier, $\mathcal{C}/\pi\mathcal{C}$ est un code de Reed-Solomon sur le corps $A/(\pi)$ de paramètres $[n, k, d]_{\mathbb{F}_p}$.

L'algorithme de Guruswami-Sudan sur B

On note $B = A/(\pi^r)$.

Entrée : Un mot reçu $y = (y_1, \ldots, y_n) \in B^n$ et un rayon $\tau < n - \sqrt{n(n-d)}$.
Sortie : Tous les mots de code dans la boule centrée en y de rayon τ.

1. $m \leftarrow \left\lfloor \frac{(k-1)n + \sqrt{(k-1)^2 n^2 + 4((n-\tau)^2 - (k-1)n)}}{2((n-\tau)^2 - (k-1)n)} \right\rfloor + 1.$
2. $L \leftarrow \left\lceil \frac{m(n-\tau)-1}{k-1} \right\rceil - 1.$
3. Trouver $Q(X, Y) = \sum_{i=0}^{L} Q_i(X) Y^i \in B[X, Y]$ tel que :
   - $Q(X + x_i, Y + y_i)$ a valuation au moins m et
   - $\deg Q_i \leq m(n-\tau) - 1 - i(k-1)$.
4. $Z \leftarrow$ les racines de Q vu dans $(B[[X]])[Y]$.
5. Retourner
   $\{f \in Z \cap B[X]_{<k} : Q(X, f(X)) = 0 \text{ et } d(f(x), y) \leq \tau\}$.

La recherche de racines sur un anneau commutatif (1)

- **Problème :** Par exemple, $|\{x \in \mathbb{Z}/p^n\mathbb{Z} : x^n = 0\}| = p^{n-1}$.

  Il y a un nombre exponentiel de racines !

- **Solution :** donner une description de l'ensemble des racines.

Théorème (Berthomieu, Lecerf, Quintin [BLQ11])

Soit A un anneau de valuation discrète et π un élément de valuation 1, $B = A/(\pi^r)$ et $f \in B[X]$ de degré d.

Alors il existe au plus d éléments $a_1, \ldots, a_m \in B$ et $e_1, \ldots, e_m \in \mathbb{N}$ tel que

$$\text{racines de } f \text{ dans } B \;=\; \coprod_{i=1}^{m} a_i + (\pi^{e_i}).$$

La recherche de racines sur un anneau commutatif (2)

Théorème (Berthomieu, Lecerf, Quintin [BLQ11])

- *Lorsque* $A = \mathbb{Z}_{p^s}$ *et* $B = A/(p^r)$, *il existe un algorithme décrivant les racines de* $f(X)$ *avec une* **complexité bit** *en moyenne de* $\widetilde{O}(r^2 ds \log p)$.

- *Lorsque* $A = \mathbb{F}_{p^s}[[t]]$ *et* $B = A/(t^r)$, *il existe un algorithme décrivant les racines de* $f(X)$ *avec une complexité en moyenne de* $\widetilde{O}(r^2 ds)$ **opérations sur** $\mathbb{F}_p$.

La recherche de racines sur un anneau commutatif (3)

Soit maintenant $Q \in B[X, Y]$, on recherche les racines de Q de la forme $f(X) \in B[X]$ de degré au plus $k-1$.

Théorème (Berthomieu, Lecerf, Quintin [BLQ11])

- *Lorsque $A = \mathbb{Z}_{p^s}$ et $B = A/(p^r)$, il existe un algorithme donnant une description des racines de Q de degré au plus $k-1$ avec une* **complexité bit** *en moyenne de $\widetilde{O}(r^2 d(dk+n)s \log p)$.*

- *Lorsque $A = \mathbb{F}_{p^s}[[t]]$ et $B = A/(t^r)$, il existe un algorithme donnant une description des racines de Q de degré au plus $k-1$ avec une* **complexité** *en moyenne de $\widetilde{O}(r^2 d(dk+n)s)$ opérations sur $\mathbb{F}_p$.*

Implantation.

Un mot sur l'implantation

Une bonne partie des algorithmes présentés a été implantée en C et en C++.

- Les algorithmes de recherche de racines de [BLQ11] ont tous été implantés en C++ dans Mathemagix [H+02].
- Les autres algorithmes font l'objet d'une librairie dédiée `Decoding` [Qui12b] conçue spécialement pour la correction d'erreurs.
- Toutes les implantations sont sous **licence GPL** et téléchargeables.

Mathemagix :
`http://www.mathemagix.org/`
Decoding :
`http://www.lix.polytechnique.fr/~quintin/decoding/`

Ma contribution à Mathemagix

Avec Grégoire Lecerf, je suis le mainteneur de plusieurs paquets.

- `mgf2x` est une interface pour la librarie `gf2x` [BGTZ09].
- `finitefieldz` est le paquet qui implante l'arithmétique des **corps finis** et quelques fonctions associées comme la **factorisation** et la **recherche de racines**.
- `quintix` est le paquet qui implante l'arithmétique des **anneaux de Galois** et quelques fonctions associées comme la **recherche de racines**.

Quelques temps de calcul pour la recherche de racines

degré d	20	40	80	160	320	640	1280
$\lfloor d/2 \rfloor$ racines, $r = 10$	4	8	18	38	82	178	373
$\left\lfloor \sqrt{d} \right\rfloor$ racines, $r = 10$	2	3	6	12	24	55	113
$\lfloor d/2 \rfloor$ racines, $r = 100$	229	474	984	2085	4431	9615	21135
$\left\lfloor \sqrt{d} \right\rfloor$ racines, $r = 100$	95	151	228	390	676	1346	2616

Table: Temps en millisecondes avec $B = \mathbb{Z}/73^r\mathbb{Z}$.

Quelques temps pour la recherche de racines (Sudan)

Length of the code	100	200	250
$\mathbb{Z}/p^{100}\mathbb{Z}$	$\mathbb{Z}/103^{100}\mathbb{Z}$	$\mathbb{Z}/211^{100}\mathbb{Z}$	$\mathbb{Z}/257^{100}\mathbb{Z}$
Y-degree	3	2	2
X-degree	29	116	83
k	9	49	59
extension degree	57	215	202
Temps en millisecondes	2046	9942	10861

Table: Application à la recherche de racines de polynômes issus de l'algorithme de Sudan sur $\mathbb{Z}/p^{100}\mathbb{Z}$.

La librairie decoding

- Constitue un **travail en cours** donc ne comporte pour l'instant que l'implantation pour les corps $\mathbb{F}_p$, $\mathbb{F}_{2^s}$.
- Est pour l'instant la **seule** librarie librement disponible offrant du décodage en liste.
- A pour l'instant un "concurrent" `Percy++` développé par I. Goldberg et N. Heninger [Gol07] dont le **but n'est pas** le décodage mais le **retrait d'informations privé** (*PIR*).
- Est conçue pour implanter les algorithmes sur les anneaux grâce à un mécanisme simple qui permet une **certaine généricité**.

Temps de calcul pour decoding sur $\mathbb{F}_p$

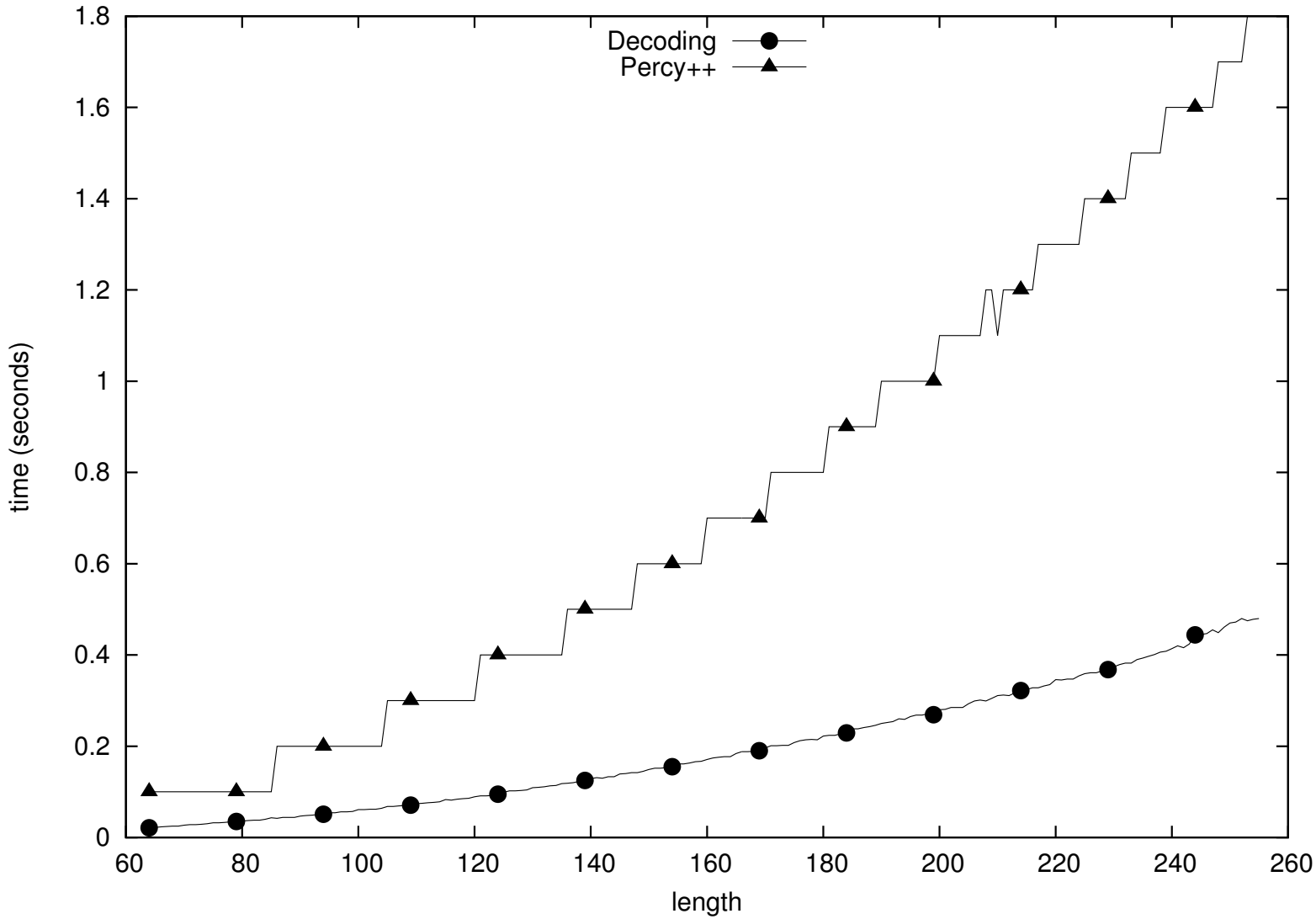


Figure: Algorithme de Guruswami-Sudan sur $\mathbb{F}_{257}$ sur des codes de Reed-Solomon de paramètres $[n, \lfloor n/5 \rfloor]_{\mathbb{F}_{257}}$ avec multiplicité 2.

Temps de calcul pour decoding sur $\mathbb{F}_{2^s}$

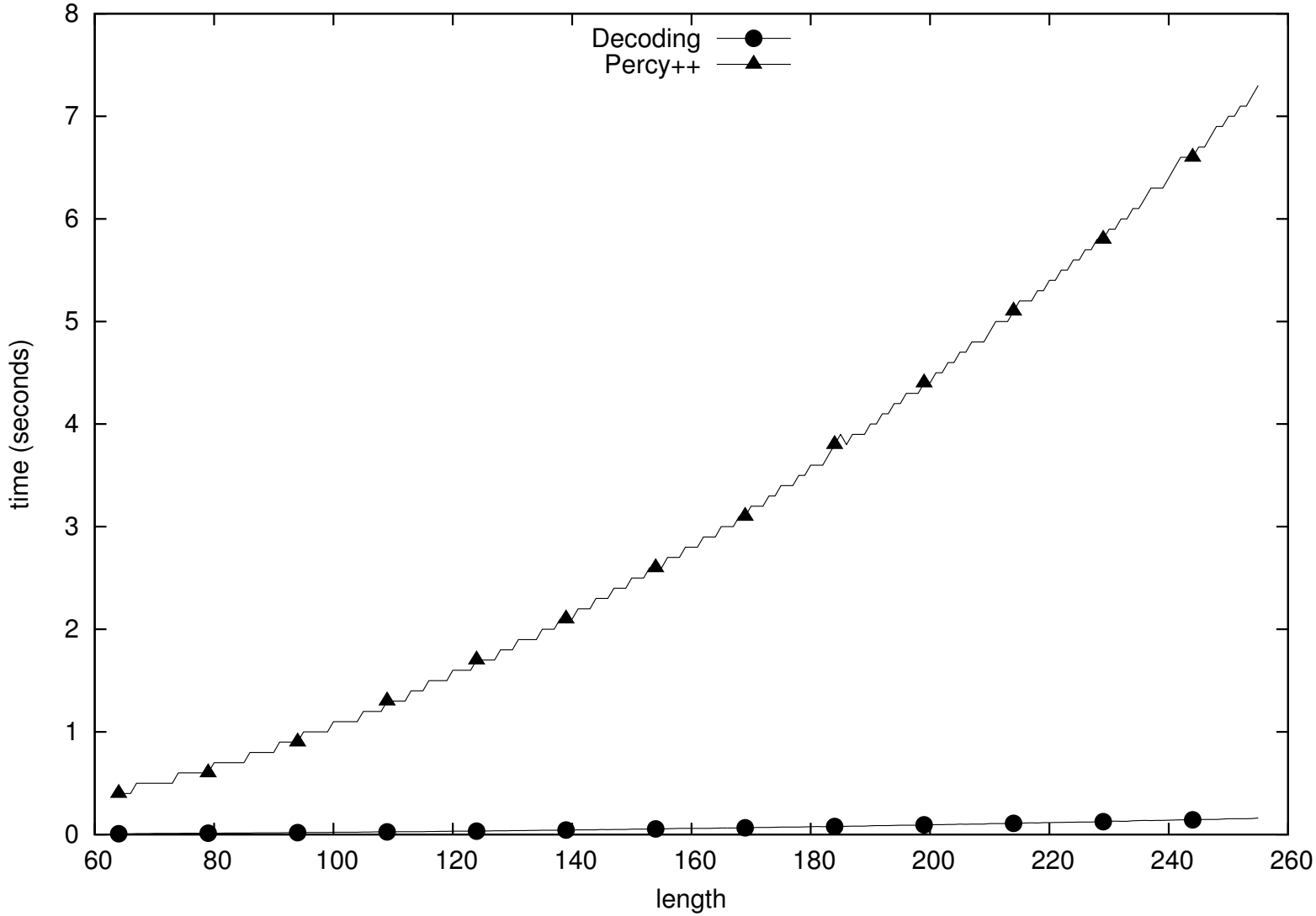


Figure: Algorithme de Guruswami-Sudan sur $\mathbb{F}_{256}$ sur des codes de Reed-Solomon de paramètres $[n, \lfloor n/5 \rfloor]_{\mathbb{F}_{256}}$ avec multiplicité 2.

Conclusion

- Étudier la dernière étape de l'algorithme de Guruswami-Sudan, la **sélection des "bonnes" racines**.
- Étudier et implanter un algorithme de **décodage unique** pour les codes **quasi-BCH**.
- Étudier l'algorithme de **Guruswami-Sudan** dans le cadre de la **métrique de Lee**.
- La recherche de racines dans $B[X]$ a un intérêt en elle-même pour le calcul formel. Par exemple pour étudier **l'algorithmique pour la factorisation** dans les anneaux

$$\left(\frac{\kappa[[t_1,\ldots,t_r]]}{(t_1,\ldots,t_d)^r}\right)[X] \text{ et } \kappa[[t_1,\ldots,t_d]][X].$$

Merci de votre attention !!!

Références I


M. Alekhnovich.
Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes.
IEEE Trans. Inform. Theory, 51(7):2257–2265, 2005.


T. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani.
Reducing Key Length of the McEliece Cryptosystem.
In *Proceedings of the 2nd International Conference on Cryptology in Africa: Progress in Cryptology*, AFRICACRYPT '09, pages 77–97, Berlin, Heidelberg, 2009. Springer-Verlag.


M. Barbier, C. Chabot, and G. Quintin.
On Generalized Reed-Solomon Codes Over Commutative and Noncommutative Rings, 2012.


M. Barbier, C. Chabot, and G. Quintin.
On quasi-cyclic codes as a generalization of cyclic codes.
Finite Fields and Their Applications, 18(5):904–919, 2012.

Références II


E.R. Berlekamp.
Algebraic Coding Theory.
McGraw Hill, New York, 1968.


R. Brent, P. Gaudry, E. Thomé, and P. Zimmermann.
gf2x.
`http://gf2x.gforge.inria.fr/`, 2009.


D. Bleichenbacher, A. Kiayias, and M. Yung.
Decoding of Interleaved Reed Solomon Codes over Noisy Data.
In Jos Baeten, Jan Lenstra, Joachim Parrow, and Gerhard Woeginger, editors, *Automata, Languages and Programming*, volume 2719 of *Lecture Notes in Computer Science*, pages 188–188. Springer Berlin / Heidelberg, 2003.

Références III


J. Berthomieu, G. Lecerf, and G. Quintin.
Polynomial root finding over local rings and application to error correcting codes.
http://hal.inria.fr/hal-00642075, 2011.


E. R. Berlekamp and L. R. Welch.
Error correction for algebraic block codes, 1986.
US Patent 4633470.


P.-L. Cayrel, C. Chabot, and A. Necer.
Quasi-cyclic codes as codes over rings of matrices.
Finite Fields and Their Applications, 16(2):100–115, 2010.

Références IV


D. Coppersmith and M. Sudan.
Reconstructing curves in three (and higher) dimensional space from noisy data.
In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, STOC '03, pages 136–142, New York, NY, USA, 2003. ACM.


P. Gopalan, V. Guruswami, and P. Raghavendra.
List Decoding Tensor Products and Interleaved Codes.
SIAM Journal of Computing, 40(5):1432–1462, 2011.


I. Goldberg.
Percy++.
Software available from `http://percy.sourceforge.net/`, 2007.

Références V


Markus Grassl.
Bounds on the minimum distance of linear codes and quantum codes.
Online available at http://www.codetables.de, 2007.
Accessed on 2011-04-19.


V. Guruswami.
List decoding of error-correcting codes: winning thesis of the 2002 ACM doctoral dissertation competition.
Lecture Notes in Computer Science. Springer, 2004.


J. van der Hoeven et al.
Mathemagix.
Software available from `http://www.mathemagix.org`, 2002.

Références VI


R. Kötter.
On Algebraic Decoding of Algebraic-Geometric and Cycling Codes.
PhD thesis, Linköping University, Sweden, 1996.


K. Lally and P. Fitzpatrick.
Algebraic structure of quasicyclic codes.
Discrete Applied Mathematics, 111(1–2):157–175, 2001.


S. Ling and P. Solé.
On the algebraic structure of quasi-cyclic codes .I. Finite fields.

IEEE Trans. Inform. Theory, 47(7):2751–2760, nov 2001.


J. Massey.
Shift-register synthesis and BCH decoding.
IEEE Trans. Inform. Theory, 15(1):122–127, jan 1969.

Références VII

Rasmus R. Nielsen and T. Hoeholdt.
Decoding Reed-Solomon codes beyond half the minimum distance.
In Johannes Buchmann, Tom Hoeholdt, Henning Stichtenoth, and Horacio Tapia Recillas, editors, *Coding Theory, Cryptography and Related Areas*. Springer-Verlag, April 2000.

W. Peterson.
Encoding and error-correction procedures for the Bose-Chaudhuri codes.
Information Theory, IRE Transactions on, 6(4):459–470, sep 1960.

Références VIII


G. Quintin.
A lifting decoding scheme and its application to interleaved linear codes.
In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pages 96–100, july 2012.


G. Quintin.
The decoding Library for List Decoding, 2012.
Accepted at ISSAC, International Symposium on Symbolic and Algebraic Computation 2012, software presentation.


R. M. Roth and G. Ruckenstein.
Efficient decoding of Reed-Solomon codes beyond half the minimum distance.
In *IEEE Trans. Inform. Theory*, page 56, 1998.

Références IX

Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa.
A method for solving key equation for decoding goppa codes.
Information and Control, 27(1):87–99, 1975.

A. Zeh, C. Gentner, and D. Augot.
An Interpolation Procedure for List Decoding Reed Solomon Codes Based on Generalized Key Equations.
IEEE Trans. Inform. Theory, 57(9):5946–5959, sep 2011.