Codis $\mathbb{Z}_2\mathbb{Z}_4$-Additius en Magma

Memòria del projecte de final de carrera corresponent als estudis d'Enginyeria Superior en Informàtica presentat per Bernat Gastón Brasó i dirigit per Mercè Villanueva Gay.

Bellaterra, 12 de Juny de 2008

El firmant, Mercè Villanueva Gay, professora del Departament d'Enginyeria de la Informació i de les Comunicacions de la Universitat Autònoma de Barcelona

CERTIFICA:

Que la present memòria ha sigut realitzada sota la seva direcció per Bernat Gastón Brasó

Bellaterra, 12 de Juny de 2008

Firmat: Mercè Villanueva Gay

Als meus pares, pel seu recolzament en tot moment i la seva determinació per a que mai em faltés cap oportunitat.

Agraïments

Especialment a la Mercè Villanueva i al Jaume Pujol per les seves inestimables correccions. Al Jaume Pernas i en Victor Ovalle per les nostres xerrades sobre codis i als meus amics i família per suportar-me en moments de molta feina.

Índex

Índex de figures

Capítol 1

Introducció

1.1 Codis Correctors

Els codis correctors d'errors ens serveixen per augmentar la fiabilitat en una transmissió d'informació. Actualment, els codis correctors estan presents en gran quantitat d'aplicacions informàtiques, des dels CD's fins a Internet. La necessitat de trobar un compromís entre rapidesa i fiabilitat de les dades transmeses, va fer pensar en algun sistema capaç de detectar i corregir errors o esborralls (dades no llegibles).

Els codis correctors d'errors afegeixen redundància en el missatge a transmetre, de manera que enviarem més dades que ens serviran per corregir possibles errors en la informació. Evidentment, tindrà un cost pel que fa a la velocitat de transmissió, doncs el volum d'informació enviada és inversament proporcional a la velocitat de la transmissió. O sigui:

```
Velocitat = Capacitat del Canal / Volum d'Informació
```

Per una banda, ens interessarà trobar codis en que cada missatge rebut tingui una única descodificació possible. Aquest tipus de codis s'anomenen codis perfectes. El problema de trobar-los tots però, no és trivial. Per altra banda, ens interessen els codis lineals i binaris. Els codis binaris permeten treballar directament a nivell de bit i els codis lineals compleixen una propietat fonamental:

- Tancament per la suma. És a dir, dues paraules del codi sumades han de donar una altra paraula del codi.

Aquesta propietat ens permet veure els codis lineals com un conjunt de generadors capaços de generar totes les paraules del codi combinant-se entre ells. Per tant no haurem de tenir tots els vectors del codi en memòria, sinó que en tindrem prou amb el conjunt de generadors. Mitjançant el que anomenem una matriu generadora, podrem representar i treballar amb codis lineals d'una manera minimalista, sense grans costos en memòria ni en còmput.

Els codis lineals i binaris han estan molt estudiats i per tant són força coneguts, per això la possibilitat de convertir, mitjançant una bijecció, codis binaris no lineals en codis lineals en una altra base, sembla força interessant. D'aquesta manera podem treballar de forma lineal sobre uns codis que no ho serien en base binària.

En aquest sentit les estructures de tipus $\mathbb{Z}_2\mathbb{Z}_4$-additiu són estructures que ens permeten representar codis binaris no necessariament lineals, en una base, on aquests codis compleixen propietats que permeten treballar de forma lineal amb ells.

1.2 Objectius

1. Documentació sobre els codis $\mathbb{Z}_2\mathbb{Z}_4$-additus.
    - Recopilació d'informació sobre l'estat de l'art.
    - Comprensió del funcionament dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.
2. Implementació d'una llibreria de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius en Magma.
    - Algorísmica i implementació de funcions de creació de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.
    - Algorísmica i implementació de funcions per al càlcul d'invariants i formes de representació dels codis (matrius generadores, de paritat, estàndards, etc...).

- Algorísmica i implementació de funcions de conversió entre codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i altres tipus d'estructures com per exemple codis binaris.
- Algorísmica i implementació de funcions per al càlcul de dualitats amb codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.
- Algorísmica i implementació eficient de funcions per al càlcul del rang i el nucli.
- Disseny d'un conjunt de testos seguint els estàndards de MAGMA.

3. Documentació, manual d'usuari i exemples de la llibreria, seguint el format establert de MAGMA

4. Redacció de la memòria del projecte.

1.3 Contingut de la Memòria

La memòria esta composta per 7 capítols:

Capítol 2: Fonaments Teòrics. En aquest capítol s'expliquen les bases teòriques en les que es sustenta tot el treball fet. Es tracta d'un resum de tots aquells conceptes que s'han adquirit per al correcte desenvolupament del projecte. En aquest capítol, el lector trobarà una introducció a la teoria de codis i informació científica que s'ha anat publicant sobre codis $\mathbb{Z}_2\mathbb{Z}_4$-additius en un format planer.

Capítol 3: Planificació del Projecte. En aquest capítol es descriu el procés que es seguirà per al correcte desenvolupament del projecte en els límits de temps establerts. El lector podrà observar les diverses tasques a desenvolupar en un format estàndard, establert per a la planificació de projectes.

Capítol 4: Entorn de Desenvolupament. En aquest capítol s'expliquen les característiques de l'entorn sobre el qual s'ha desenvolupat el pro-

jecte. Es dóna informació sobre la màquina utilitzada i l'entorn de programació. Es descriu de manera breu, els estàndards de programació que s'han utilitzat i les característiques específiques del llenguatge triat. El lector també podrà trobar una breu descripció d'altres possibilitats existents al mercat i la justificació sobre l'elecció del MAGMA com a entorn per al desenvolupament.

Capítol 5: Desenvolupament del Projecte. En aquest capítol es descriu l'algorísmica de les funcions més importants utilitzades per al desenvolupament del projecte. El lector podrà tenir una visió general del paquet de funcions així com entrar en detall sobre l'algorísmica i els criteris de programació utilitzats. També es descriu la documentació associada al projecte així com el joc de proves utilitzat per al test del paquet de funcions.

Capítol 6: Resultats. En aquest capítol s'exposen els resultats obtinguts. El lector podrà observar les millores de temps aconseguides respecte a versions anteriors d'algunes funcions, així com els límits obtinguts en l'execució del paquet de funcions.

Capítol 7: Conclusions. En aquest capítol s'exposen les conclusions obtingudes durant el desenvolupament del projecte, així com possibles millores i ampliacions a desenvolupar en el futur.

Capítol 2

Fonaments Teòrics

2.1 Codis: binàris i quaternaris

Quan parlem de $\mathbb{Z}_2 = \{0, 1\}$, estem parlant del cos format pels enters mòdul 2 i quan parlem de $\mathbb{Z}_4 = \{0, 1, 2, 3\}$, estem parlant de l'anell format pels enters mòdul 4.

Definition 1 *Definim distància de Hamming d_H entre dos vectors binaris de la mateixa longitud, com el nombre de coordenades diferents entre els dos vectors.*

Per exemple $d_H\left((1011101), (1001001)\right) = 2$.

Definition 2 *Un codi (n, M, d) $\mathtt{C}$ sobre un cos o anell A és un subconjunt amb M vectors sobre A de longitud n i distància mínima de Hamming d, tal que $d = min\left\{d_H(x, y) : \forall x, y \in \mathtt{C}, x \neq y\right\}$*

És a dir, un codi és un subconjunt de vectors sobre A que s'anomenen paraules codi. El conjunt d'aquestes paraules codi defineix i construeix aquest codi.

Si el codi $\mathtt{C}$ té una estructura lineal, vol dir que és un subespai de A^n, on n és la longitud de les paraules codi. Llavors, $\mathtt{C}$ compleix la propietats de

tancament per la suma abans descrita en la secció 1.1, i es podrà representar tots els vectors del codi amb un conjunt de generadors. Això permetrà rebaixar molt el cost:

- En memòria: El fet de poder representar el codi com un conjunt minimal de vectors linealment independents rebaixa l'espai en memòria
- En CPU: Podem organitzar els generadors del codi en una matriu anomenada matriu generadora que denotarem per $\mathcal{G}$ i la matriu generadora del codi dual $\mathcal{C}^{\perp}$ anomenada matriu de control i denotada per $\mathcal{H}$. Mitjançant aquesta representació, les operacions de codificar i descodificar seran multiplicacions vector-matriu.

Tan sols una petita part de tot el conjunt de codis possibles són lineals. Per tant, és interessant buscar algun tipus de representació eficient per tots aquells codis que són no lineals.

Definition 3 *Un codi binari (n, M, d) C és un subconjunt amb M elements de $\mathbb{Z}_2^n$ i distància mínima de Hamming d.*

Quan parlem d'un codi binari ens referim al fet que el codi es un subespai de vectors de n coordenades a $\mathbb{Z}_2$. Aquest cas serà especialment important degut al tractament binari de la informació que es dóna habitualment en les transmissions i en l'enmagatzematge de dades.

El conjunt de vectors sobre $\mathbb{Z}_2^n$ és de 2^n i els possibles subconjunts seran 2^{2^n}. D'aquests conjunts però, només una petita part tindran una estructura lineal. Es pretén trobar estructures capaces de representar codis binaris no lineals, de manera que ens donin eines per classificar-los i per treballar amb ells.

Definition 4 *El pes de Lee d'una coordenada d'un vector quaternari es determina seguint la següent funció:*

$$w_L(0) = 0 \;\; w_L(1) = w_L(3) = 1 \;\; w_L(2) = 2$$

El pes de Lee d'un vector quaternari x de n coordenades és:

$$w_L(x) = \sum_{i=1}^{n} w_L(x_i).$$

Definim la distància de Lee d_L entre dos vectors de la mateixa longitud x i y com

$$d_L = w_L(x - y).$$

Definition 5 *Un codi quaternari (β, M, d) $\mathcal{C}$ és un subconjunt amb M elements de $\mathbb{Z}_4^\beta$ i distància mínima de Lee d.*

Podem veure qualsevol codi quaternari com un codi binari un cop passat una certa aplicacio lineal bijectiva que anomenarem mapa de Gray i denotarem per ϕ. Aquesta aplicació codificarà les coordenades del codi quaternari i les representarà amb dos bits al codi binari. El mapa de Gray per a un codi quaternari queda definit de la següent manera: $\phi : \mathbb{Z}_4^\beta \longrightarrow \mathbb{Z}_2^n$, on $n = 2\beta$, i en cada coordenada ve donat per:

$$\phi(0) = (0,0),\ \phi(1) = (0,1),\ \phi(2) = (1,1),\ \phi(3) = (1,0)$$

La tria d'aquest mapa de Gray és especialment important, doncs compleix que transforma la distància de Lee definida sobre el codi quaternari a distància de Hamming sobre el codi binari.

Els codis quaternaris serviran de base per al desenvolupament dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Tal com va demostrar Solé en el seu article [8] hi han codis binaris no lineals, que són lineals com a quaternaris un cop transformats mitjançant ϕ. És més, hi ha famílies de codis sense relació en binari però que sí la tenen vistos com a codis quaternaris. El lector interessat en codis quaternaris pot trobar més informació a [22].

2.2 Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius

Segons la definició de codis additius donada per Delsarte el 1973 [2] els codis additius són subgrups dels grups abelians. En el cas del esquema de Hamming, és a dir quan el grup abelià és de tamany 2^n, les úniques estructures per al grup abelià són aquelles de la forma $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$, amb $\alpha + 2\beta = n$.

Els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius els podem veure com subgrups de $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$, pertant les paraules codi són una tupla de dos vectors, un dels quals es troba a $\mathbb{Z}_2^\alpha$ i l'altre a $\mathbb{Z}_4^\beta$. D'aquí s'extreu que quan $\alpha = 0$ estarem en el cas quaternari i quan $\beta = 0$ estarem en el cas binari.

- Un vector és d'ordre dos si generara dos vectors diferents al sumar-se amb ell mateix, i per tant compleix que $2v = 0$. Per exemple $v = (2, 0, 2, 0)$.

- Un vector és d'ordre quatre si genera quatre vectors diferents al sumar-se amb ell mateix, i per tant compleix que $4v = 0$. Per exemple $v = (1, 0, 2, 3)$.

Sigui $\mathcal{C}$ un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu, llavors $\mathcal{C}$ és un subgrup de $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ i també és isomòrfic a una estructura abeliana $\mathbb{Z}_2^\gamma \times \mathbb{Z}_4^\delta$. Per tant el codi $\mathcal{C}$ es pot generar a partir de $\delta + \gamma$ generadors, on γ és el mínim nombre de generadors d'ordre 2 i δ és el mínim nombre de generadors d'ordre 4. A més, tenim que el nombre de paraules del codi és $|\mathcal{C}| = 2^\gamma 4^\delta$ i el nombre de paraules d'ordre 2 dins de $\mathcal{C}$ serà $2^{\gamma+\delta}$.

Si X (respectivament Y) és el conjunt de coordenades de $\mathbb{Z}_2$ (respectivament $\mathbb{Z}_4$), és a dir $|X| = \alpha$ i $|Y| = \beta$. Anomenarem $\mathcal{C}_X$ (respectivament $\mathcal{C}_Y$) al codi $\mathcal{C}$ restringit a les coordenades X (respectivament Y). Llavors si $\mathcal{C}_b$ és el subcodi de $\mathcal{C}$ amb totes les paraules d'ordre 2, i el restringim a les coordenades X, això és $(\mathcal{C}_b)_X$ llavors κ és la dimensió de $(\mathcal{C}_b)_X$, que és un codi lineal i binari. Per al cas $\alpha = 0$ també tindrem $\kappa = 0$.

En resum, els parametres amb els que distingirem un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu seran:

- α: Nombre de coordenades de les paraules-codi a $\mathbb{Z}_2$.
- β: Nombre de coordenades de les paraules-codi a $\mathbb{Z}_4$.
- γ: Nombre mínim de generadors d'ordre 2.
- δ: Nombre mínim de generadors d'ordre 4.
- κ: Dimensió del subcodi de $\mathcal{C}$ restringit als generadors d'ordre dos de la part binaria, és a dir limitada per α i γ.

Podem dir llavors que $\mathcal{C}$ és del tipus $(\alpha, \beta; \gamma, \delta; \kappa)$. Un cop definits aquests paràmetres, podem veure el codi quaternari $\mathcal{C}_Y$ de tipus $(0, \beta; \gamma_Y, \delta; 0)$ on $\gamma_Y \leq \gamma$. Equivalenment podem veure $\mathcal{C}_X$ com un codi binari de tipus $(\alpha, 0; \gamma_X, 0; \gamma_X)$, on $\gamma_X \geq \kappa$.

2.2.1 Mapa de Gray

Com ja hem esmentat a la secció 2.2 els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius són subgrups $\mathcal{C}$ de $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$. Tal com passa amb els codis quaternaris, podem veure els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius com a codis binàris un cop els hem passat una certa aplicació $C = \Phi(\mathcal{C})$, on Φ és la extensió del Gray map habitual que anomenem ϕ, llavors tenim que $\Phi : \mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta \longrightarrow \mathbb{Z}_2^n$, on $n = \alpha + 2\beta$, donat per

$$\Phi(x, y) = (x, \phi(y_1), \ldots, \phi(y_\beta)) \quad \forall x \in \mathbb{Z}_2^\alpha,\ \forall y = (y_1, \ldots, y_\beta) \in \mathbb{Z}_4^\beta;$$

a on $\phi : \mathbb{Z}_4 \longrightarrow \mathbb{Z}_2^2$ és el mapa de Gray a $\mathbb{Z}_4$.

De la mateixa manera que en el cas quaternari el mapa de Gray és una isometria que transforma les distancies de Lee definides als codis $\mathcal{C}$ sobre $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$, a distàncies de Hamming definides pels codis binaris $C = \Phi(\mathcal{C})$.

2.2.2 Matriu Generadora

Definition 6 *Dos codis que només difereixen en una permutació de columnes de les seves coordenades, es diuen codis equivalents.*

La matriu generadora $\mathcal{G}$ d'un codi $\mathcal{C}$, conté la informació essencial per representar el codi. Juntament amb la matriu de control $\mathcal{H}$, seran la base per a totes les operacions de codificació i descodificació. En el cas dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius no només necessitarem les matrius, sinó que també necessitarem saber com a mínim el valor de α.

En general però, podem donar una matriu amb el mínim nombre de files on podrem fer dues particions: una partició vertical que separarà la part binària de la quaternària, i una partició horitzontal que separarà els generadors d'ordre 2 i d'ordre 4.

Així la matriu generadora $\mathcal{G}$ d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu es pot escriure com:

$$\mathcal{G} = \left(\begin{array}{c|c} B_1 & 2B_3 \\ \hline B_2 & Q \end{array}\right) \tag{2.1}$$

on B_1, B_2 és la part binària dels generadors d'ordre 2 i d'ordre 4 respectivament i B_3, Q és la part quaternària dels generadors d'ordre 2 i d'ordre 4 respectivament. Cal destacar el fet que B_3 tot i ser quaternària correspon als generadors d'ordre 2 i per tant les seves coordenades contindran només elements de $\{0, 2\}$.

Theorem 2.2.1 *Si C és un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu amb tipus $(\alpha, \beta; \gamma, \delta; \kappa)$. Llavors, podem obtenir un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu equivalent a C amb una matriu generadora de la forma*

$$\mathcal{G}_S = \left(\begin{array}{cc|ccc} I_\kappa & T_b & 2T_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2T_1 & 2I_{\gamma-\kappa} & \mathbf{0} \\ \hline \mathbf{0} & S_b & S_q & R & I_\delta \end{array}\right), \tag{2.2}$$

on T_b, T_1, T_2, R, S_b són matrius a $\mathbb{Z}_2$ i S_q és una matriu a $\mathbb{Z}_4$. Cal destacar que $I_\delta, I_{\gamma-\kappa}$ són les matrius identitat de tamany δ i $\gamma - \kappa$ respectivament.

Aquesta matriu ens serveix per representar d'una forma estàndard tots els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i per tant facilitarà la feina en varis sentits:

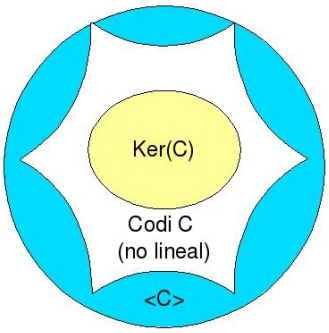

Figura 2.1: Representació d'un codi no lineal.

- Creació de codis concrets $\mathbb{Z}_2\mathbb{Z}_4$-additius mitjançant la construcció d'una matriu estàndard.
- Comparació de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.
- Codificació i descodificació sistemàtica, ja que sabem a on està la informació i a on la redundància.

2.3 Rang i Nucli per codis binaris

El problema de la classificació dels codis binaris no lineals no és trivial, es tracta de trobar invariants per a codis equivalents que els determinin de forma única. En aquest sentit s'ha investigat sobre dos invariants dels codis binaris: el rang i la dimensió del nucli.

El rang d'un codi binari C, que denotarem per r, és la dimensió del subespai lineal minimal que conté tot el codi. Per veure-ho d'una forma més gràfica és la dimensió de l'espai creat pels vectors del codi, més el mínim nombre de vectors a afegir per convertir l'espai en lineal. Per tant,

$$r = Dim(\langle C \rangle).$$

El nucli d'un codi binari C, que denotarem per $Ker(C)$, es defineix com la intersecció dels subespais lineals i maximals del codi a analitzar. Degut a aquesta definició, sempre podrem trobar un codi equivalent al nucli de C que sigui lineal. Per tant podem veure el nucli com aquella part lineal més representativa d'un codi.

En realitat ens interessarà, com en el cas del rang, la dimensió de l'espai del nucli que anomenarem k on

$$k = Dim(Ker(C)).$$

És fàcil veure que

$$Ker(C) \leq C \leq \langle C \rangle.$$

A més, si el codi és lineal

$$C = Ker(C) = \langle C \rangle.$$

i per tant,

$$Dim(C) = k = r.$$

Efectivament, en un codi lineal, el nucli, $\langle C \rangle$ i el codi seran exactament el mateix. Podem veure per tant $\langle C \rangle$ i el nucli $Ker(C)$ com dos subconjunts lineals que marquen els límits superiors i inferiors de la linealitat d'un codi (veure Figura 2.1).

Per altra banda, el nucli ens permetrà extreure d'un codi no lineal, un subcodi d'aquest que sí que ho sigui, i que per tant es pugui representar i treballar d'una manera eficient amb ell. Cal remarcar que el nucli només té sentit per a codis binaris.

Matemàticament, el nucli ve definit per:

$$Ker(C) = \{x : x + y \in C,\ \forall y \in C\} \tag{2.3}$$

Com es pot veure la definició de nucli porta a un problema de complexitat

clarament exponencial, serà doncs interessant veure com es pot reduir aquest cost. En aquest projecte s'estudiarà el nucli només per codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i per tant s'aprofitaran propietats especials d'aquests codis.

Per al cas d'un codi binari sense estructura, el cómput hauria de ser a priori més difícil, ja que la complexitat d'un algorisme és inversament proporcional a l'informació de la qual es disposa. Dit d'una altra manera, com més informació menys complexitat alhora d'implementar-ho. Si es vol aprofundir en el càlcul del nucli per a codis binaris en general es pot consultar el projecte [14].

Donat un codi binari, es pot fer un primer subconjunt lineal considerant els vectors que formen el nucli. Si el codi és lineal, no hi haurà més vectors que els continguts al nucli, però si no ho és, hi hauran vectors que pertanyen al codi però que no pertanyen al nucli del codi.

A partir de la definició del nucli, es pot veure que aquests vectors es poden classificar en subconjunts disjunts construïts a partir de traslladats del nucli. Un cop fet això, podem representar qualsevol codi com a unió del nucli i dels traslladats del nucli, anomenats cossets:

$$C = \bigcup_{i=0}^{t} \Big(K(C) + c_i \Big)$$

D'aquesta manera dividim un codi que no és lineal, en subconjunts d'aquest que representen el nucli, més el nucli traslladat tantes vegades com cossets tingui el codi (veure figura 2.2). Es conclou doncs, que qualsevol codi es pot representar simplement amb el nucli, més un vector que anomenarem líder, per cada cosset del codi.

2.4 Rang i Nucli per codis $\mathbb{Z}_2\mathbb{Z}_4$-additius

Com es comenta a la secció 2.3 el rang d'un codi només té sentit per a codis binàris. El fet de parlar del rang per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius doncs, sembla a priori il·lògic.

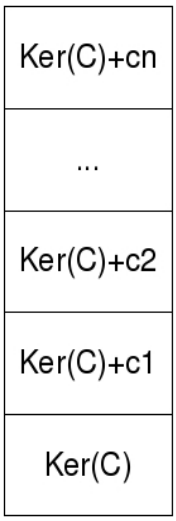

Figura 2.2: Representació d'un codi com a unió dels cossets del nucli.

Donat un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\mathcal{C}$, el rang r d'aquest codi serà el conjunt de vectors de la forma:

$$r = Dimensio(\langle \Phi(\mathcal{C}) \rangle)$$

A [4] es demostra que el rang d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu es pot calcular a partir de la dimensió de l'espai format pels $\gamma + \delta$ generadors d'ordre dos, els δ generadors d'ordre 4 i tots els vectors de la forma:

$$\{2x * y : \forall x, y \in GeneradorsOrdre4\}$$

Cal destacar que el rang serà un codi lineal i que no cal utilitzar totes les paraules del codi sinó que en tindrem prou amb un conjunt de generadors. En concret el conjunt tindrà $\gamma + 2\delta + \binom{\delta}{2}$ generadors.

Com es comenta a la secció 2.3 el nucli d'un codi només té sentit per a codis binàris. Donat un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\mathcal{C}$, el nucli d'aquest codi serà el subconjunt de $\mathcal{C}$ format pels vectors:

$$K(\mathcal{C}) = \{v \in \mathcal{C} : \Phi(v) \in Ker(\Phi(\mathcal{C}))\}$$

Aquest subconjunt $K(\mathcal{C})$ és també un subcodi $\mathbb{Z}_2\mathbb{Z}_4$-additiu de $\mathcal{C}$.

Com ja s'ha vist a la subsecció 2.2.1, es pot canviar de base d'un codi binari C a un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\mathcal{C}$ i viceversa amb el mapa de Gray. En resum, existeix el nucli a $\mathbb{Z}_2\mathbb{Z}_4$ anomenat $K(\mathcal{C})$ com a mapa de Gray del nucli binari anomenat $Ker(\Phi(\mathcal{C}))$.

Dit això, es pot passar a analitzar les propietats d'aquest nucli $\mathbb{Z}_2\mathbb{Z}_4$-additiu. Intuïtivament podem veure que serà més fàcil treballar, i per tant calcular, el nucli dins del codi $\mathcal{C}$ que en el seu corresponent codi no lineal binari C, ja que es disposarà d'una estructura de tipus $\mathbb{Z}_2\mathbb{Z}_4$-additiva que facilitarà els càlculs.

Es pot comparar el comportament amb el ben conegut sistema de Fourier per al tractament del senyal, el que es necessita és el senyal en domini temporal, però es pot passar a domini freqüencial, treballar amb ell (ja que és més fàcil) i en acabat tornar-lo a transformar al domini temporal.

Veiem doncs les propietats que ens facilitaran el càlcul:

- El nucli d'un codi lineal binari es pot calcular a partir dels generadors del codi. En el cas d'un codi lineal $Ker(C) = C$ com s'explica a 2.3. Sabem que podem representar C a partir dels seus generadors degut a les propietats de linealitat. Per tan podrem representar $Ker(C)$ a partir dels generadors de C.

- [4] Donat un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu podem calcular el seu nucli mitjançant els generadors del codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu. De fet, els vectors del nucli d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu compleixen:

$$K(\mathcal{C}) = \{x \in \mathcal{C} \mid \forall y \in \mathcal{G}, 2x * y \in \mathcal{C}\} \tag{2.4}$$

  Notem que tots els $\gamma + \delta$ generadors d'ordre dos d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu seran part del nucli, ja que l'operació $2x$ sobre un vector d'ordre 2 dóna la paraula zero.

Aquestes propietats facilitaran molt els càlculs ja que treballarem només amb els generadors del codi, però algorísmicament parlant de fet, l'únic que suposarà un cost elevat seran els generadors d'ordre 4 ja que els d'ordre 2 automàticament quedaran afegits al nucli.

Capítol 3

Planificació del Projecte

El projecte que es pretén portar a terme és un projecte d'investigació i desenvolupament. Per una banda com ja s'ha vist, el projecte té una forta base teòrica que s'ha d'anar assolint conforme s'avança en la seva implementació. La feina de documentació teòrica i d'assoliment dels conceptes serà sens dubte una de les bases per a completar els objectius establerts.

Pel que fa al desenvolupament s'haurà de separar la part d'algorísmica i d'implementació. Degut a la naturalesa del projecte d'investigació, la majoria de temps serà destinat al desenvolupament d'algorismes, capaços de tractar de la forma més òptima possible els diferents problemes de còmput. Ja que la complexitat en la teoria de codis és en molts casos no polinòmica, aquest requeriment d'optimització serà molt important.

Les úniques referències que es tindran, pel que fa a l'algorísmica d'aquests tipus de funcions, seran els projectes fets anteriorment al departament així com els articles teòrics publicats per part dels membres del CCG. Tot això ens servira de base per al correcte desenvolupament i optimització dels algorismes.

L'implementació per la seva banda haurà de ser de tipus matemàtica, aquest fet suposarà un esforç d'adaptació degut a que aquest tipus de llenguatges no són els més habituals durant la carrera. Es vol aconseguir un paquet capaç de ser distribuït per al seu ús, i pertant seguirà els estàndards

i *Best-Practices* coneguts pel que fa a programació. Concretament el llibre d'estil estarà basat en el *CCG Style Guide*.

Per altra banda s'haurà d'incloure l'explicació i documentació de totes les funcions implementades per al projecte. Per a aquest procés es seguirà l'estil que ja dóna MAGMA per a la seva documentació (veure Annex 1). El programa haurà de comptar amb un procés explicatiu d'instal·lació i utilització (fitxer README). S'inclouran també una sèrie d'exemples seguint l'estil de MAGMA per complementar la documentació (veure Paquet "Codis Z2Z4-Additius"). Tota el paquet està disponible a la web del grup CCG [23].

El programa haurà de comptar amb una llicència, ja que es pretén distribuir-lo, per evitar implicacions legals tot software ha de contindre una llicència. En aquest cas s'ha triat la versió GPLv3, una llicència que permet l'us, distribució i modificació del software sempre que sigui per a fins no comercials. Aquesta llicència està secundada per la *Free Software Fundation* [5].

3.1 Tasques

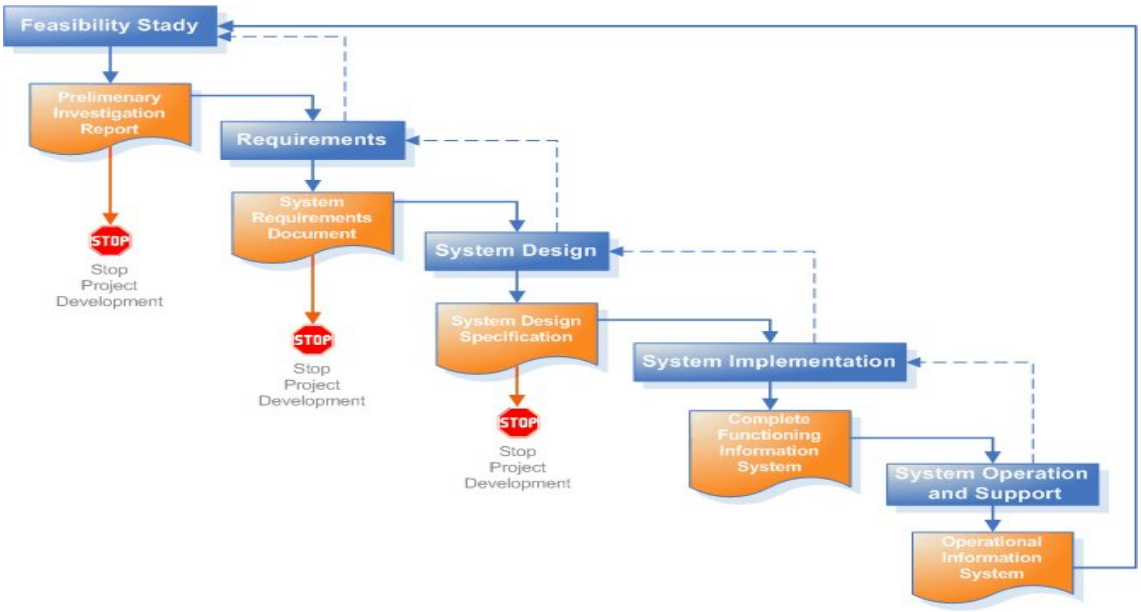

Figura 3.1: Model en cascada

El cicle de desenvolupament serà de tipus clàssic o cascada (veure Figura 3.1) i el desglossament en tasques el podem veure a la Figura 3.2. A continuació les tasques seran explicades detingudament:

WBS	Name	Start	Finish	Work	Duration	Slack	Cost	Assigned to
1	Codis Z2Z4-additius	ago 1	mai 30	374d	218d		0	
1.1	Fonaments Teòrics	ago 1	gen 14	105d	119d	99d	0	
1.1.1	Documentació Z2Z4	ago 1	oct 31	66d	66d	152d	0	
1.1.2	Documentació Kemel	nov 21	gen 14	39d	39d	99d	0	
1.2	Implementació Llibreria	ago 1	mar 17	144d	164d	53d	0	
1.2.1	Llibreria Z2Z4	ago 1	nov 20	80d	80d		0	
1.2.2	Implementació Kemel	nov 21	gen 21	44d	44d		0	
1.2.3	Implementació Rang	feb 19	mar17	20d	20d		0	
1.3	Test	nov 21	mar 31	70d	94d	44d	0	
1.3.1	Test de Z2Z4	nov 21	gen 15	40d	40d	98d	0	
1.3.2	Test del Kemel	gen 22	feb 18	20d	20d		0	
1.3.3	Test del Rang	mar18	mar31	10d	10d	44d	0	
1.4	Manual Aplicació	mar 18	mar 24	6d	5d	49d	0	
1.4.1	Documentació Llibreria Z2Z4	mar18	mar18	1d	1d	53d	0	
1.4.2	Documentació Rang i Kemel	mar18	mar24	5d	5d		0	
1.5	Memoria	mar25	mai 30	49d	49d		0	

Figura 3.2: Recopilació de tasques

1. **Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius**

   (a) **Fonaments Teòrics**

      i. **Documentació $\mathbb{Z}_2\mathbb{Z}_4$:** L'objectiu d'aquesta tasca és assolir les bases teòriques en que es sustenten els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.

      ii. **Documentació Nucli:** L'objectiu d'aquesta tasca és entendre el concepte de nucli i els avantatges que comporta al treballar amb codis no lineals.

   (b) **Implementació Llibreria**

      i. **Llibreria $\mathbb{Z}_2\mathbb{Z}_4$:** L'objectiu d'aquesta tasca és implementar totes les funcions bàsiques que permeten crear i treballar amb codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Es seguirà el format que MAGMA dóna

per altres tipus de codis, especialment per als codis quaternaris que ens serviran de base.

ii. **Implementació Nucli:** L'objectiu d'aquesta tasca és implementar una funció eficient per al còmput del nucli dels codis binaris associats als codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. L'algorisme que s'utilitzi serà bàsic per al èxit d'aquesta tasca i per tant hauria de ser el més òptim possible.

iii. **Implementació Rang:** En aquesta tasca s'implementarà un algorisme eficient per al comput del rang dels codis binàris associats als codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. L'algorisme que s'utilitzi serà bàsic per al èxit d'aquesta tasca i per tant hauria de ser el més òptim possible.

(c) **Test:**

i. **Test de $\mathbb{Z}_2\mathbb{Z}_4$:** S'ha de provar el correcte funcionament de totes les funcions de la llibreria $\mathbb{Z}_2\mathbb{Z}_4$, minimitzant els errors i evitant que l'usuari faci crides incorrectes. S'haurà de crear un conjunt de tests i exemples, aquests últims en format Magma i que seran inclosos dins la documentació.

ii. **Test del Nucli:** Aquesta tasca tindrà dues bessants, per una banda provar el correcte funcionament de la funció del nucli. Per altra banda s'haurà d'optimitzar al màxim possible la funció, refent si cal l'algorisme per a un millor aprofitament dels recursos.

iii. **Test del Rang:** Provar el correcte funcionament de l'implementació usada per a calcular el rang dels codis.

(d) **Manual de l'aplicació**

i. **Documentació Llibreria $\mathbb{Z}_2\mathbb{Z}_4$:** Es seguirà el model i estil de Magma pel que fa al manual de la llibreria. L'explicació inclosa dins de cada funció serà també el *help* de la funció des de la línia de comandes.

ii. **Documentació Rang i Nucli:** Es seguirà el mateix estil que en el cas de la Llibreria. S'haurà de tenir en compte l'especial dificultat d'aquests dos conceptes i la seva importància, amb una més extensa explicació al respecte.

(e) **Memòria:** Redacció de la memòria del projecte, seguint les pautes donades per a la presentació de "Projectes de final de carrera" a l'Escola Tècnica Superior d'Enginyeries (ETSE) dins de l'Universitat Autònoma de Barcelona (UAB).

3.2 Recursos

La majoria dels recursos utilitzats per a aquest projecte han estat subministrats pel Departament d'Enginyeria de l'Informació i les Comunicacions (dEIC). Bàsicament s'han usat tres tipus de recursos.

3.2.1 Recursos Documentals

Els recursos documentals utilitzats en aquest projecte han estat bàsicament els articles referents a codis quaternaris i especialment a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Aquesta sèrie d'articles s'han anat esmentant durant el capítol 2, i estan recollits dins la bibliografia del projecte. També s'ha utilitzat el manual de Magma especialment pel que fa a codis quaternaris [1]. Pel que fa a l'implementació s'ha usat de base, el codi obert que Magma proporciona en el cas d'algunes funcions.

3.2.2 Recursos de Software

Magma és un software privat (veure secció 4.2) amb dos tipus de llicència:

- Llicència d'estudiant: limitat a un ús de memòria de 2 GB i un rendiment computacional mitjà.

- Llicència completa: ús de memòria total (limitat per l'arquitectura del computador), i rendiment computacional alt.

Per al nostre cas es necessitarà una llicència completa de la qual ja disposa el departament.

En el cas dels editors de text s'han usat dos diferents, un per al codi i l'altre per a la documentació.

- Per a programar s'ha utilitzat el `GEDIT`, ja que l'ordinador amb llicència per a MAGMA exporta l'ambient gràfic del que disposa: el `GNOME`.

- Per a la documentació s'ha emprat l'editor de texts científics LaTeX el motiu és la gran quantitat de funcionalitats per a l'escriptura matemàtica de les que disposa. Com a frontal s'ha utilitzat el `Kile` un editor de LaTeX per GNU/Linux.

3.2.3 Recursos de Hardware

Per al desenvolupament del projecte, tenint en compte les necessitats de còmput, l'entorn triat i la disponibilitat de material, s'ha escollit:

- Ordinador Intel Dual Core 3700 amb 2 GB de Ram i sistema operatiu Fedora-GNU/LINUX core 6. Aquest ordinador és el servidor disponible al departament amb llicència completa de MAGMA.

- Es disposarà d'un computador d'altes prestacions (4 nuclis i 7 GB de RAM) per a càlculs especials, disponible al departament també amb una llicència completa.

Capítol 4

Entorn de Desenvolupament

4.1 Software Matemàtic

Al tractar-se d'un projecte sobre teoria de codis, treballarem sobre anells, cossos i matemàtica discreta. S'havia de buscar per tant, un software preparat per a la programació matemàtica.

La capacitat de treballar amb estructures matemàtiques i guardar-les com a tals (com guardar un codi lineal només amb els seus generadors) era bàsica per a la implementació del paquet de funcions. A més, calia valorar les utilitats ja implementades bàsicament pel que fa a codis correctors d'errors, la facilitat d'ús, l'eficiència i la flexibilitat per crear estructures pròpies.

Cal destacar les diferències entre la programació numèrica i la programació simbòlica. La programació numèrica està destinada al càlcul d'operacions purament numèriques, per contra, la programació simbòlica permet declarar variables dins d'una expressió. Un cas clar de programació simbòlica (i que els programes purament numèrics no poden fer), és el d'una equació de tipus $3 = 2 + x$.

A continuació es descriuran les alternatives més importants existents al mercat i se'n farà un breu anàlisi.

Maple [10] és un dels entorns de programació matemàtica més estesos al mercat. Es tracta d'un software de llicència privada desenvolupat a

Canadà. Té gran quantitat de possibilitats, des del càlcul simbòlic fins a la teoria de nombres, l'àlgebra o el càlcul. També afegeix una interfície gràfica amigable que permet veure gràfiques i programar d'una forma molt intuïtiva.

El Maple és molt utilitzat sobretot en les comunitats educatives, degut a la facilitat del llenguatge, la seva gran versatilitat i l'orientació gràfica de que disposa. Malauradament, els temps de càlcul no són excessivament bons; quan es tracta d'aplicacions petites no hi ha problemes, però per a grans quantitats de dades es queda bloquejat fàcilment.

Mathematica [11] és un software privat molt optimitzat fet als EUA. Té un llenguatge de programació propi i moltes llibreries fortament optimitzades per al càlcul simbòlic i numèric. Mathematica és molt apreciat degut al seu alt rendiment i a la capacitat de representar gràfiques complexes 2 i 3-dimensionals.

Mathematica dóna suport a gran quantitat de camps, però pel que fa a la teoria de nombres, a la criptografia i a la codificació, el seu comportament és bastant fluix; sobretot en la varietat d'utilitats per treballar en aquests camps.

Matlab [12] és un software privatiu desenvolupat als EUA. Està bàsicament orientat a la programació numèrica i cobreix un gran nombre de funcionalitats en aquest camp. El seu funcionament és orientat a vector, obtenint grans resultats amb les operacions que impliquen vectors i matrius.

Matlab també té una interfície gràfica i un llenguatge de programació propi. Permet enllaçar amb l'eina de càlcul simbòlic del Maple estenent la seva funcionalitat en aquest camp. Per altra banda però, Matlab no té cap tipus de de funcionalitat pel que fa a la teoria de codis.

Number Theory Library (NTL) over C++ [17] és una llibreria orientada a objecte i programada en C++. Es tracta d'una llibreria com-

posta per una sèrie d'objectes capaços de representar estructures matemàtiques i mètodes per al seu tractament.

NTL té la principal avantatge en la seva rapidesa i flexibilitat, ja que utilitza un llenguatge de programació molt ràpid com el C++ directament. El fet de ser de codi lliure permet configurar les estructures existents o afegir noves funcionalitats. El problema principal de NTL és el fet que les estructures definides són de molt baix nivell (matrius, vectors, etc) i no incorpora funcionalitats per a la teoria de codis, encara que en el futur, degut a la seva naturalesa, algú podria programar-les.

Software for Algebra and Geometry Experimentation (SAGE) [20] és un projecte de software lliure que pretén unificar les solucions de programació matemàtica existents al mercat. El projecte SAGE s'inspira en l'eslògan "Fabricar el cotxe sense re-inventar la roda" i va començar el 2005 liderat pel professor W. Stein de la universitat de Washington.

SAGE intenta donar resposta a tots els àmbits de la matemàtica actual i incorpora interfícies per a poder executar tot el software descrit en aquesta secció. Permet importar i exportar els resultats de SAGE a les altres solucions per a una programació integral. El principal problema de SAGE és la maduresa com a programa, ja que fa poc que s'ha tret al mercat comparat amb les altres solucions. Cal destacar però, que aquest projecte està cridat a ser un dels entorns de programació matemàtica del futur.

GAP [6], PARI [15], Singular [18], Macaulay2 [7], Maxima [13] són entorns de software lliure orientats a la programació matemàtica. Tots continuen actualitzats avui en dia i alguns d'ells són dels primers que van sortir al mercat. El principal problema de tots ells es que no cobreixen alguna part important de la programació matemàtica (veure Taula 4.1)

funció	Singular	GAP	PARI	Maxima
Factorització de polinomis	Si	No	No	Si
Grups	No	Si	No	No
Cossos i Anells	No	No	Si	No
Càlcul Simbòlic	No	No	No	Si

Taula 4.1: Taula de funcionalitats suportades per les diferents solucions.

4.2 Entorn Utilitzat: Magma

Magma [1] és un software d'alt rendiment amb llicència privada per part d'una companyia australiana. Segons la seva web es defineixen com:

Magma *és un software ampli i ben soportat dissenyat per solucionar problemes d'alt cost computacional relacionats amb l'algebra, la teoria de nombres, la geometria i la combinatòria. Proporciona un entorn matemàtic rigorós per treballar en tots aquests camps.*

Magma té un ampli conjunt de funcionalitats en els àmbits comentats i el seu llenguatge de programació propi es basa en l'escriptura matemàtica, tot i això, té un dèficit important pel que fa a programació simbòlica. No disposa de cap interfície gràfica però disposa d'una línia de comandes molt bona, amb ajudes i explicacions.

Actualment no hi ha la possibilitat per part dels usuaris de definir les seves pròpies estructures de dades i incloure-les al nucli. En els darrers anys però, s'ha avançat en aquest sentit i des de fa unes quantes versions es disposa d'una comanda de tipus *record* per a la creació d'estructures compostes. Magma deixa definir paquets de funcions i bases de dades per part de l'usuari i ofereix la possibilitat de carregar-les en l'arrancada del programa.

Magma està escrit en `C` i utilitza els millors algorismes coneguts per a la solució de problemes matemàtics, aquesta característica és molt important per al tractament de problemes no polinomials, com és el cas del nucli (veure secció 2.3). Degut al llenguatge `C`, Magma permet enllaçar llibreries existents com per exemple *General Multiprecision Package (GMP)*. Tot i això, com compilar el nucli no és possible per part dels usuaris, només ho poden fer els

programadors de MAGMA.

MAGMA és segurament, el millor programa actualment al mercat per treballar en certs camps com per exemple la geometria aritmètica, la teoria de grups o la teoria de codis. Bàsicament, aquest últim camp és el que es necessita per al desenvolupament del projecte i és l'entorn triat per a la seva implementació.

Capítol 5

Desenvolupament del Projecte

5.1 Qüestions Prèvies

Un cop estudiats els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius es pretén implementar una llibreria capaç de crear-los i treballar amb ells. Amb aquest objectiu s'estudia les llibreria de codis quaternaris implementada en MAGMA, ja que les propietats dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i dels codis quaternaris són força semblants.

Un codi de longitud β definit sobre l'anell $\mathbb{Z}_4$ conté β coordenades de valors $\{0, 1, 2, 3\}$, per contra un codi de longitud α definit sobre el cos $\mathbb{Z}_2$ conté α coordenades de valors $\{0, 1\}$. Es pot veure que el valors $\{0, 2\}$ quaternaris es comporten de la mateixa manera que els valors $\{0, 1\}$ en binari, analitzant-ho:

$\mathbb{Z}_2$	$\mathbb{Z}_4$
$\{0, 1\}$	$\{0, 2\}$
$0 + \{0, 1\} = \{0, 1\}$	$0 + \{0, 2\} = \{0, 2\}$
$1 + \{0, 1\} = \{1, 0\}$	$2 + \{0, 2\} = \{2, 0\}$

Per tant podem representar la part $\mathbb{Z}_2^\alpha$ d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu a $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$, convertint els uns en dosos a $\mathbb{Z}_4^\alpha$ es a dir: $\{0, 1\} \in \mathbb{Z}_2 \Leftrightarrow \{0, 2\} \in \mathbb{Z}_4$. Aquest codi estarà representat com un objecte compost d'un codi quaternari i del paràmetre α, que permetrà distingir les dues parts de les paraules codi.

Cal esmentar que MAGMA no treballa amb codis no lineals, encara que permet representar-los mitjançant un conjunt de paraules. Serà especialment important el fet de poder representar alguns codis binaris no lineals com a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, o sigui com a conjunció del paràmetre α i d'un codi quaternari lineal.

L'implementació, documentació, exemples i test de la llibreria, s'han fet basant-se en els estàndards utilitzats pels desenvolupadors de MAGMA, amb l'idea de ser el màxim integrables dins de l'entorn en el futur.

En aquest capítol s'explicarà de forma resumida les funcions implementades en el projecte, agrupades per grups d'utilitat tal i com fa MAGMA. No s'explicaran totes les funcions una a una, sinó que es pretén donar una vista general de les utilitats de la llibreria i entrar en detall tan sols de les funcions més significatives.

El lector interessat pot veure el "$\mathbb{Z}_2\mathbb{Z}_4$-additive Handbook" (Annex 1) que s'ha desenvolupat com a ajuda, índex i explicació de totes les funcions implementades. El paquet desenvolupat és accessible des de la web del grup [23].

5.2 Creació de Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius

Per a la creació d'aquests codis es decideix utilitzar una estructura de tipus *record* en MAGMA. El *record* és un conjunt de variables agrupades en un sola estructura, molt semblant a la comanda *struct* de `C`. Les variables o estructures emmagatzemades en el record seran públiques per definició, MAGMA no deixa declara-les privades. Aquest record o registre que anomenarem R contindrà dues variables.

- **Code:** Un codi quaternari de longitud $\alpha + \beta$ que representarà el codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu amb les cordenades binàries transformades a coordenades quaternàries.
- **Alpha:** Un enter que marca el paràmetre α del codi, si $\alpha = 0$, **Code** serà exactament un codi quaternari.

S'utilitzarà una arquitectura basada en una funció de creació de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, que accepti múltiples estructures d'entrada, i d'un valor α. La funció retornarà un registre R corresponent al codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu. A partir d'aquesta funció se'n crearan altres, que construiran certs tipus de codis especials (ex. codi zero, univers i de repetició).

$\mathbb{Z}_2\mathbb{Z}_4$AdditiveCode: Aquesta funció construirà un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu seguint les premises explicades a la secció 5.1. Per flexibilitzar-la al màxim, el primer paràmetre pot ser de diferents tipus, veiem doncs els dos paràmetres que pot rebre la funció:

- L:
  1. Seqüència de vectors quaternaris.
  2. Subespai quaternari.
  3. Matriu de mida $m \times n$.
  4. Codi quaternari.
  5. Codi binari.
- alpha: Enter que marca el nombre de coordenades binàries del codi.

Random$\mathbb{Z}_2\mathbb{Z}_4$AdditiveCode: Aquesta funció permet crear un codi aleatori passant una llista de paràmetres que conté de 2 a 5 paràmetres. Els paràmetres que falten es calculen de manera aleatòria amb les següents restriccions:

γ**:** $\gamma \geq 0$

δ**:** $\delta \geq 0, \gamma + \delta \leq \beta$

κ**:** $\kappa \geq 0, \kappa \leq min(\gamma, \alpha)$

Un cop es tenen els 5 paràmetres $[\alpha, \beta, \gamma, \delta, \kappa]$ es calcula un codi aleatori

a partir de les submatrius de la matriu estàndard (veure Teorema 2.2.1).

$$\mathcal{G}_S = \left(\begin{array}{cc|ccc} I_\kappa & T_b & 2T_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2T_1 & 2I_{\gamma-\kappa} & \mathbf{0} \\ \hline \mathbf{0} & S_b & S_q & R & I_\delta \end{array}\right) \tag{5.1}$$

S'utilitzaran funcions de diagonalització per a les matrius $I_\kappa, 2I_{\gamma-\kappa}$ i I_δ. Per a la resta, es definiran matrius random sobre l'anell que els correspon segons el Teorema 2.2.1.

5.3 Invariants Numèriques i Espai del Codi

En aquesta secció s'implementen funcions per al càlcul dels invariants bàsics d'un codi tals com la longitud, la dimensió, el nombre de generadors, el nombre de paraules o la taxa d'informació. Totes aquestes utilitats es basen en les funcions ja donades per a codis quaternaris de Magma.

Pel que fa a l'espai del codi, hi ha funcions per trobar el conjunt de paraules i generadors d'un codi, així com els generadors d'ordre dos i quatre. Les funcions de construcció de les diferents matrius generadores, especialment la de la matriu generadora amb mínim nombre de files, o sigui amb $\gamma + \delta$ files, seguiran algorismes molt semblants al càlcul del tipus que descrivim a continuació:

Z2Z4Type: Pel que fa als invariants es necessita una funció de càlcul del tipus del codi (veure secció 2.2). L'algorisme per trobar els diferents paràmetres és el següent.

1. $\delta = \#\{v \in \mathcal{G} \mid 2v \neq 0\}$.
2. $\gamma = NumeroFiles(\mathcal{G}) - \delta$.
3. $\beta = NumeroColumnes(\mathcal{G}) - \alpha$.
4. $\kappa = Dimensio(Subespai(< \gamma >| \alpha))$.

5.4 Forma Estàndard

Donat un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\mathcal{C}$ es pretén trobar la matriu estàndard corresponent a un codi equivalent a $\mathcal{C}$. Per a fer-ho, s'utilitzarà el teorema 2.2.1. Per aquesta funció es va intentar usar la seva equivalent per a codis quaternaris en MAGMA, però contenia un error. Aquest problema ja s'ha fet arribar als desenvolupadors de MAGMA.

El fet de no poder aprofitar l'equivalent per a codis quaternaris, implicarà diagonalitzar matrius sobre anells, en concret sobre l'anell $\mathbb{Z}_4^\beta$. Aquesta és una feina complicada, ja que encara que hi han algorismes capaços de diagonalitzar amb rapidesa, el fet que el cas quaternari sigui un anell, només permet la utilització del sistema de Gauss.

Pel que fa a l'algorisme, recordem la forma d'una matriu estàndard:

$$\mathcal{G}_S = \left(\begin{array}{cc|ccc} I_\kappa & T_b & 2T_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2T_1 & 2I_{\gamma-\kappa} & \mathbf{0} \\ \hline \mathbf{0} & S_b & S_q & R & I_\delta \end{array}\right), \tag{5.2}$$

S'hauran de complir les restriccions imposades per aquesta. S'ha de respectar per tant, la posició i valors d'aquelles submatrius que composen la matriu estàndard (bàsicament les submatrius zero i les diagonals). Per a fer-ho partirem de la matriu generadora amb mínim nombre de files (veure secció 5.3).

L'algorisme utilitzat cridarà per ordre a les subfuncions de diagonalització. Cada una d'aquestes subfuncions retornarà la matriu diagonalitzada per la seva part, més la permutació de columnes que hagi estat necessària. Cal recordar que en codis la permutació de columnes és important, ja que genera un codi diferent però equivalent.

Diagonalització I_κ: Aquesta funció diagonalitzarà la submatriu corresponent a I_κ i generarà les matrius zero de sota seu.

Diagonalització I_δ: Aquesta funció diagonalitzarà la submatriu corresponent a I_δ i generarà les matrius zero de sobre seu.

Diagonalització$I_{\gamma-\kappa}$: Aquesta funció diagonalitzarà la submatriu corresponent a $I_{\gamma-\kappa}$ i generarà la matriu zero de sobre seu.

La funció $\mathbb{Z}_2\mathbb{Z}_4$**StandardForm** doncs, retornarà la matriu estàndard equivalent, la permutació de columnes total i una isometria que converteix $\mathcal{C}$ al codi estàndard equivalent.

5.5 Mapa de Gray

En aquesta secció s'implementen funcions per al càlcul del mapa de Gray i equivalències entre codis binaris i codis $\mathbb{Z}_2\mathbb{Z}_4$-additius (veure secció 2.2.1). Aquest fet serà bàsic, doncs ens permetrà passar de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius teòrics als codis binàris que s'utilitzen a la pràctica. Les funcions més importants d'aquesta secció són:

$\mathbb{Z}_2\mathbb{Z}_4$**GrayMap:** Donat un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\mathcal{C}$, retorna la isometria Φ, que transforma $\mathcal{C}$ al codi binari corresponent C.

$\mathbb{Z}_2\mathbb{Z}_4$**GrayMapImage:** Utilitza Φ i $\mathcal{C}$ per a retornar tot el conjunt de paraules que composen C. Per tant, retorna el seu codi binari equivalent com un conjunt de vectors, ja que no necessàriament serà lineal.

Has$\mathbb{Z}_2\mathbb{Z}_4$**LinearGrayMapImage:** Aquesta funció comprova que el codi binari $C = \Phi(\mathcal{C})$ sigui lineal. En cas afirmatiu retorna el codi binari lineal.

5.6 Rang i Nucli

El rang d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu r, com ja s'ha comentat a la secció 2.4, és la dimensió del subespai lineal minimal $\langle C \rangle$ que conté el codi. Però r és equivalent a la dimensió del conjunt format per $\gamma + 2\delta + \binom{\delta}{2}$ generadors. Aquest fet permetrà rebaixar la complexitat.

Pel que fa al nucli, l'implementació ha estat depurada durant varies versions. Com ja es comenta a la secció 2.3, el càlcul del nucli és un problema

no polinòmic, però aprofitant certes propietats podem reduir-ne el cost de càlcul.

5.6.1 Càlcul Exhaustiu

Una primera aproximació al càlcul de nuclis podria ser la de prendre la definició de nucli, l'equació 2.3 i aplicar-la. L'algorisme per tant agafarà totes les paraules del codi binari C i les sumarà amb totes les paraules de C altre cop. Com veiem en aquest cas estem agafant tot el conjunt de paraules codi i per tant, es pot tractar d'un conjunt realment molt gran.

```
funcio Kernel (Code)
    ∀v ∈ C
        ∀w ∈ C
            si v + w ∉ C llavors
                Aquest vector no està al nucli
            fisi
        fi for
        si No ha estat descartat llavors
            nucli.add(v)
        fi si
    fi for
    return nucli
fi funcio
```

La complexitat d'aquest algorisme és exponencial sobre el nombre de paraules $\#\mathcal{C}$, en concret serà de $O\left(2^{\gamma+2\delta}\right)$. Aquest càlcul es pot arribar a millorar, però l'àmbit dels codis binàris queda fora d'aquest projecte, per més informació sobre càlcul de nuclis per a codis binaris el lector pot veure [3] i [14].

5.6.2 Càlcul per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius: Primera Versió

Aquesta serà la primera versió en la que acotarem l'anàlisi als codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Això ens permetrà usar propietats d'aquest tipus de codis que ens facilitaran els càlculs. Bàsicament ara es treballarà només amb els generadors del codi, per tant les operacions seran de multiplicació vector matriu. Per a més informació veure la secció 2.4.

```
funcio Z2Z4Kernelv1 (Code)
    MG = Matriu dels generadors d'ordre 4
    for totes les combinacions possibles de vectors de MG que
    anomenarem v
        for w = tots els generadors d'ordre 4
            si v * 2w ∉ Code llavors
                Aquest vector no està al nucli
            fisi
        fi for
        si No ha estat descartat llavors
            nucli.add(v)
        fi si
    fi for
    return nucli
fi funcio
```

Com veiem en aquest cas, es poden descartar tots els vectors $\gamma+\delta$ d'ordre dos aprofitant la propietat que es decriu a la secció 2.4. La complexitat d'aquest nou algorisme és $O(2^{\delta})$.

5.6.3 Càlcul per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius: Segona Versió

Un cop sabem que un vector v d'ordre 4 està al nucli, v no aporta més informació. Per tant, podem treure v de l'espai de les combinacions dels ge-

neradors d'ordre quatre. Aquest fet permet reduir el nombre de combinacions en certs casos especials.

En el cas d'un codi que només conté els generadors d'ordre 2 al nucli, el problema continuarà sent igual de costós, ja que haurem de provar els δ vectors sense poder descartar-ne cap. En el pitjor dels casos doncs, la complexitat continuarà sent $O(2^\delta)$, però en la majoria dels casos, aquesta es podrà veure reduïda fins a la $O(\delta)$ del cas lineal.

funcio $\mathbb{Z}_2\mathbb{Z}_4$Kernelv2 (Code)
    MG = Matriu dels generadors de ordre 4
    MK = Matriu que conté el nucli temporal (vectors d'ordre 2)
    Recórrer la matriu des de l'ultim vector fins al primer combinant aquest vector amb els que té per sota a la matriu
        Comprovar que el vector estigui al nucli
        Si esta al nucli treure vector de la matriu i afegir-lo al nucli temporal
        Sinó deixar-lo a la matriu
    return nucli, matriu resultant de vectors que no estan al nucli
fi funcio

Com veiem, aquesta funció no combina tots els vectors d'ordre 4, sinó que tan sols ho fa amb aquells que previament ha comprovat que no són al nucli. Si un vector és al nucli, no ens aporta cap informació, en canvi, un vector que no estigui dins el nucli, podria combinar-se amb un altre vector que tampoc estigui dins el nucli, per donar-ne un que sí hi sigui inclòs. Tot i això, en el pitjor dels casos la complexitat continua sent de $O(2^\delta)$.

Cal fixar-se també, amb el fet que retornem no només el nucli, sinó també la matriu dels vectors que no estan al kernel. Aquests vectors corresponen als líders dels cossets (veure secció 2.3).

5.7 Funcions de Conversió i Predicats Booleans

Les funcions de conversió tenen utilitats molt diverses, però comparteixen que permeten convertir diversos tipus d'estructures a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i viceversa. En aquest apartat destaquen les funcions que permeten crear codis $\mathbb{Z}_2\mathbb{Z}_4$-additius a partir de codis binaris i quaternaris.

En aquest sentit, es pot trobar una funció que converteix un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu representat com a objecte compost d'una α i d'un codi quaternari, a un producte de vectors a $\mathbb{Z}_2^\alpha$ i vectors a $\mathbb{Z}_4^\beta$. És a dir, mostra l'estructura real que tenen aquests codis com a subgrups de $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$.

A continuació trobem els predicats booleans. Els predicats booleans són un conjunt de funcions que calculen propietats dels codis $\mathcal{C}$, i en donen una resposta binària. En aquesta secció es pot trobar per exemple, funcions per a comprovar la igualtat de dos codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, per comprovar si un codi està o no contingut en un altre o si un codi és efectivament de tipus $\mathbb{Z}_2\mathbb{Z}_4$-additiu.

5.8 Construcció de nous codis a partir d'altres

En aquesta secció es creen nous codis $\mathbb{Z}_2\mathbb{Z}_4$-additius a partir d'altres codis també $\mathbb{Z}_2\mathbb{Z}_4$-additius. Aquest fet és especialment important, perquè ens permet implementar operacions molt usades com l'obtenció de subcodis restringits; el codi suma, intersecció o concatenació a partir de dos codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, o els codis retallats (*shorten* i *punctured codes*).

Cal destacar les funcions que implementen les formes de Plotkin. Aquestes seran la base per a crear famílies de codis, com per exemple els Reed-Muller, sobre $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$. Per saber més sobre famílies de codis amb estructura $\mathbb{Z}_2\mathbb{Z}_4$-additiva, el lector es pot adreçar a [16].

$\mathbb{Z}_2\mathbb{Z}_4$**PlotkinSum:** Donats dos codis C i D retorna el codi Plotkin $\mathbb{Z}_2\mathbb{Z}_4$-additiu que consisteix en tots els vectors de la forma $(u_\alpha, u_\alpha+v_\alpha|u_\beta, u_\beta+$

$v_\beta)$, on $(u_\alpha|u_\beta) \in C$ i $(v_\alpha|v_\beta) \in D$. Per aquesta funció es va intentar usar la seva equivalent per a codis quaternaris en MAGMA, però contenia un error. Aquest problema ja s'ha fet arribar als desenvolupadors de MAGMA. L'algorisme construirà aquest codi, tal i com es demostra a [16], a partir d'una matriu de la forma:

$$\mathcal{G}_{PC} = \begin{pmatrix} \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} \\ 0 & \mathcal{G}_\mathcal{B} \end{pmatrix}$$

$\mathbb{Z}_2\mathbb{Z}_4$**QuaternaryPlotkinSum:** Donats dos codis $\mathbb{Z}_2\mathbb{Z}_4$-additius A i B tots dos amb els mateixos paràmetres $\alpha = 0$ i β, construeix un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu que consisteix en els vectors de la forma $(u, u+v, u+2v, u+3v)$, on $u \in A$ i $v \in B$. En base a [16], l'algorisme trobarà el codi de suma Plotkin quaternària a partir d'una matriu de la forma:

$$\mathcal{G}_{QP} = \begin{pmatrix} \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} \\ 0 & \mathcal{G}_\mathcal{B} & 2\mathcal{G}_\mathcal{B} & 3\mathcal{G}_\mathcal{B} \end{pmatrix}$$

$\mathbb{Z}_2\mathbb{Z}_4$**DoublePlotkinSum:** Donats quatre codis $\mathbb{Z}_2\mathbb{Z}_4$-additius A, B, C i D amb els mateixos paràmetres $\alpha = 0$ i β, construeix un codi de suma doble Plotkin que consisteix en tots els vectors de la forma $(u, u+v, u+2v+z, u+3v+z+t)$, on $u \in A$, $v \in B$, $z \in C$ i $t \in D$. En base a [16], l'algorisme trobarà el codi de suma Plotkin doble a partir d'una matriu de la forma:

$$\mathcal{G}_{DP} = \begin{pmatrix} \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} & \mathcal{G}_\mathcal{A} \\ 0 & \mathcal{G}_\mathcal{B} & 2\mathcal{G}_\mathcal{B} & 3\mathcal{G}_\mathcal{B} \\ 0 & 0 & \mathcal{G}_\mathcal{C} & \mathcal{G}_\mathcal{C} \\ 0 & 0 & 0 & \mathcal{G}_\mathcal{D} \end{pmatrix}$$

$\mathbb{Z}_2\mathbb{Z}_4$**BQPlotkinSum:** Donats 3 codis $\mathbb{Z}_2\mathbb{Z}_4$-additius A, B i C tots amb els mateixos paràmetres $\alpha = 0$ i β, construeix la BQ suma Plotkin de A, B i C. En base a [16], l'algorisme trobara el codi a partir d'una matriu

de la forma:

$$\mathcal{G}_{BQ} = \begin{pmatrix} \mathcal{G}_{\mathcal{A}} & \mathcal{G}_{\mathcal{A}} & \mathcal{G}_{\mathcal{A}} & \mathcal{G}_{\mathcal{A}} \\ 0 & \mathcal{G}'_{\mathcal{B}} & 2\mathcal{G}'_{\mathcal{B}} & 3\mathcal{G}'_{\mathcal{B}} \\ 0 & 0 & \hat{\mathcal{G}}_{\mathcal{B}} & \hat{\mathcal{G}}_{\mathcal{B}} \\ 0 & 0 & 0 & \mathcal{G}_C \end{pmatrix},$$

Si G_B és la matriu generadora de B i tipus $(0, \beta; \gamma, \delta; \kappa)$, el codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu B' s'obté de B canviant els dosos per uns a les γ files d'ordre 2 de G_B. El codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu $\hat{B}$ s'obté de B esborrant les γ files d'ordre dos de G_B.

La BQ suma Plotkin és un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu que consisteix en tots els vectors de la forma $(u, u + v', u + 2v' + \hat{v}, u + 3v' + \hat{v} + z)$, on $u \in A$, $v' \in B'$ $\hat{v} \in \hat{B}$, i $z \in C$.

5.9 Test del Software

El procés de test, és un conjunt de proves aplicades al software per validar-ne el seu funcionament i són utilitzats per determinar paràmetres com la completesa, la seguretat o la qualitat del software desenvolupat. De totes maneres, mai podrem estar al cent per cent segurs que el software no tindrà cap malfuncionament o *bug*.

El producte ha de complir les especificacions requerides, per a això el responsable del producte ha d'assegurar el correcte funcionament, però també ha d'anar més enllà, intentant predir quins seran els punts foscos o el mal ús que es pot donar al producte. El test és per tant, un procés d'investigació.

El test és una part important del control de qualitat d'un projecte, alguns dels paràmetres que determina són la fiabilitat, l'estabilitat, la portabilitat, la mantenibilitat o la usabilitat.

Hi han diverses tècniques per al test de software [21], les més conegudes són les de caixa blanca i caixa negra, però també hi han tests d'unitat, de sistema, de regressió. Alguns d'aquests tests són responsabilitat del programador i uns altres del client.

El procés de test en MAGMA, no és massa complet. Encara que té instruccions per a controlar els paràmetres d'entrada, té molts problemes alhora de provar els errors. MAGMA no permet continuar l'execució d'un programa un cop s'ha produit un error predit o un malfuncionament no predit. S'espera que en el futur MAGMA millori en aquest sentit.

En aquest projecte s'han fet bàsicament tests d'unitat, per validar que un mòdul funcioni correctament. Es tracta d'escriure crides a les funcions del mòdul per provar-les, sobretot mirant aquells casos especialment complicats.

Capítol 6

Resultats

En aquest capítol es veuran els resultats obtinguts amb el desenvolupament de la llibreria. Com ja s'ha anat explicant en capítols anteriors, aquest projecte pretén establir les bases per a futures investigacions sobre els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.

Per la majoria de funcions implementades no té sentit presentar-ne resultats escrits, doncs són funcions fins ara no implementades. Per veure'n els seus resultats om hauria de descarregar-se el paquet i utilitzar-lo en l'entorn de Magma.

Hi ha una funció però, que sí pot ser analitzada a fons, es tracta del càlcul del nucli de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Aquesta serà doncs la funció que analitzarem detingudament, veient els rendiments de temps i memòria que s'aconsegueixen.

6.1 Rendiment en Memòria

El fet de poder treballar amb els generadors del codi, tant pel càlcul del rang com del nucli és un gran avantatge. Mantenir en memòria totes les paraules d'un codi que pot arribar a ser molt gran, és en molts casos intractable. Aquest problema és el que es millora ostensiblement entre la versió exhaustiva del càlcul de nuclis (veure subsecció 5.6.1) i les versions sobre codis $\mathbb{Z}_2\mathbb{Z}_4$-

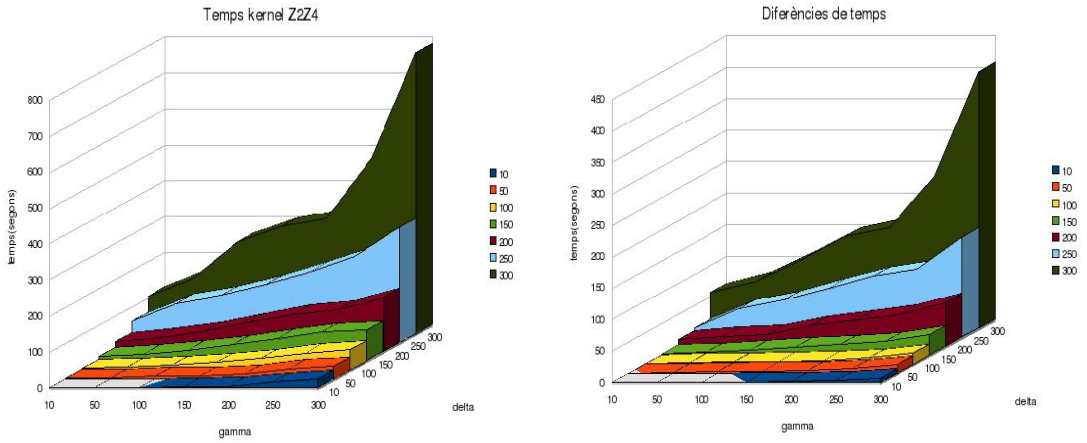

Figura 6.1: A l'esquerra, temps de la segona versió del algorisme. A la dreta, diferències de temps amb la primera versió.

additius (veure subseccions 5.6.2 i 5.6.3).

Fins ara en còmputs del nucli per codis binaris estem arribant a codis de 2^{16} paraules. En general però, serà impossible treballar amb tota aquella xifra de paraules codi que s'apropi a 2^{20} (per a un ordinador de 32 bits). El fet de poder utilitzar generadors per generar totes aquestes paraules en temps d'execució, millora molt l'espai en memòria.

6.2 Rendiment en Temps

Comparar els temps del càlcul del nucli per a codis binaris amb el del nucli per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, no té massa sentit. Primer, les escales de mesura són diferents; mentre que en el nucli d'un codi binari es compara el temps segons el nombre de paraules del codi, en el nucli per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius aquesta mesura no té massa sentit. Per a aquest segon cas, s'hauria de comparar segons el nombre de generadors del codi.

Al provar de construir un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu segons els paràmetres γ i δ i comparar el temps entre el càlcul del nucli per a codis binaris i per codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, ens trobem que per γ i δ petites (menors que 10) el temps del

algorisme per codis binaris és molt dolent, quan en canvi per al algorisme per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius no passem de les dècimes de segon.

Arribem a la conclusió que els dos algorismes no es poden comparar, doncs un cop podem transformar un codi no lineal en lineal, les propietats que es donen en el cas lineal fan els dos algorismes incomparables. És un cas semblant al d'un cotxe i un avió, els dos serveixen per al transport, però es regeixen per paràmetres diferents.

Entrant a comparar les dues versions de l'algorisme del càlcul del nucli per a codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, hi ha una consideració a fer:

- Considerem que un codi $\mathcal{C}_1$ és més lineal que un codi $\mathcal{C}_2$, quan el $k(\mathcal{C}_1)$ està més aprop del $\mathcal{C}_1$ que no pas $k(\mathcal{C}_2)$ del $\mathcal{C}_2$. Aquesta propietat és important, doncs el segon algorisme té el millor rendiment quan el codi és lineal. En canvi per al cas no lineal complet, els dos algorismes són pràcticament iguals en temps.

Recordem que la dimensió k del nucli d'un codi $\mathbb{Z}_2\mathbb{Z}_4$-additiu està comprès entre els paràmetres:

$$k \in \{\gamma + \delta, \ldots, \gamma + 2\delta\}\,.$$

per tant un codi serà completament no lineal quan $k = \gamma + \delta$ mentre que serà lineal quan $k = \gamma + 2\delta$.

Aquest fet provoca uns resultats aleatoris en les comparatives segons si un codi és més o menys lineal que un altre, sense dependre dels generadors del codi.

Un cop fetes aquestes consideracions, es pot intentar donar una certa comparació de temps entre els dos algorismes (veure Figura 6.1). En general es veu com el temps de còmput augmenta exponencialment amb una pendent més important en l'eix dels δ generadors. La millora, també augmenta exponencialment, amb unes petites variacions segons si el codi és més lineal o menys.

Capítol 7

Conclusions

En aquest projecte es presenta el desenvolupament d'un paquet d'aplicacions en l'entorn de programació matemàtica MAGMA, per al tractament dels codis anomenats $\mathbb{Z}_2\mathbb{Z}_4$-additius. Tot el paquet desenvolupat és accessible des de la web del grup CCG a `ccg.uab.es` i es pretén que sigui proposat a MAGMA com a paquet de facto en les pròximes versions.

Els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius permeten representar alguns codis binàris en general (especialment important per al cas no lineal), com a codis lineals en l'espai dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Aquest fet permetrà l'estudi de tota una sèrie de codis binaris no lineals que fins ara eren intractables. El treball desenvolupat en aquest projecte senta les bases per a futures investigacions sobre les propietats de codis i famílies de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.

7.1 Objectius Complerts

S'han assolit els objectius estimats al principi del projecte:

- S'han assolit els coneixements necessaris per a l'implementació del conjunt de funcions de la llibreria.
- S'ha desenvolupat un paquet de funcions per treballar amb codis $\mathbb{Z}_2\mathbb{Z}_4$-additius.

- S'han assolit i sobrepassat, en algun cas, els requeriments d'optimització per a càlculs d'alt cost.

- S'ha desenvolupat un manual d'ajuda i funcionament de tot el paquet desenvolupat.

- S'han provat les funcions per al control de qualitat associat al paquet.

7.2 Millores i Futures investigacions

Un cop sentades les bases per a la creació de codis $\mathbb{Z}_2\mathbb{Z}_4$-additius, es pretén que es continuï investigant i desenvolupant funcions per afegir al paquet base. En aquest sentit podem preveure dos camins clars de millora; l'optimització i el desenvolupament.

Pel que fa a l'optimització, seria interessant veure si els càlculs no polinòmics implementats en aquest projecte (bàsicament el càlcul del nucli), poden rebaixar-se de categoria i convertir-se en polinòmics. L'optimització que s'ha aconseguit a la pràctica es bona però en el pitjor dels casos la complexitat continua sent de $O(2^{\delta})$.

S'espera que en el futur es desenvolupin famílies de codis perfectes sobre estructures $\mathbb{Z}_2\mathbb{Z}_4$-additives. Els codis perfectes són aquells en que per cada paraula del codi només hi ha una possible descodificació, i per tant no pot donar lloc a ambigüitats. Les famílies de codis perfectes que es pretén desenvolupar a curt i mig termini són:

- Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius de Hadamard.

- Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius de Hamming estesos.

- Codis $\mathbb{Z}_2\mathbb{Z}_4$-additius Reed-Muller.

7.3 Conclusions Finals

Aquest ha estat un projecte complex, sobretot pel que fa al volum d'informació i conceptes teòrics que s'han hagut d'assolir. Aquesta ha estat potser la part més farragosa del projecte tant pel que fa a l'assoliment com a l'escriptura. Les bases que s'han assolit en aquesta part però, eren completament necessàries.

Un cop assolides les bases, el procés d'investigació i desenvolupament ha estat molt interessant. El fet d'haver de buscar la solució i de planificar els temps d'entrega, han donat un plus al projecte que he trobat molt enriquidor.

Espero que en el futur es puguin desenvolupar i investigar nous codis sobre la base dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius i potser, que siguin prou interessants per aplicar-los en algun projecte pràctic. El temps ho aclarirà.

Bibliografia

[1] J. J. Cannon and W. Bosma (Eds.) *Handbook of* MAGMA *Functions*, Edition 2.13, 4350 pages, 2006.

[2] P. Delsarte, "An algebraic approach to the association schemes of coding theory", *Philips Research Rep. Suppl.*, vol. 10, 1973.

[3] F.Diez, "Invariants de matrius hadamard en MAGMA", Projecte de final de carrera, Universitat Autònoma de Barcelona, 2007.

[4] C. Fernández-Córdoba, J. Pujol and M. Villanueva, "Z2Z4-linear codes: rank and nucli", submitted to *IEEE Transactions on Information Theory*, 2008

[5] "Free Sofware Fundation", `http://www.fsf.org/`.

[6] The GAP Group, *GAP – Groups, Algorithms, and Programming*, Version 4.4.10, 2007. `http://www.gap-system.org`.

[7] D. Grayson and M. Stillman, *Macaulay2: A software system devoted to supporting research in algebraic geometry and commutative algebra*, `http://www.math.uiuc.edu/Macaulay2/`.

[8] A.R. Hammons, P.V. Kumar, A.R. Calderbank, N.J.A. Sloane and P. Solé, "The $\mathbb{Z}_4$-linearity of kerdock, preparata, goethals and related codes", *IEEE Trans. on Information Theory*, vol. 40, pp. 301-319, 1994.

[9] D. S. Krotov, "$\mathbb{Z}_4$-linear Hadamard and extended perfect codes", *Proc. of the International Workshop on Coding and Cryptography*, Paris (France), Jan. 8-12, pp. 329–334, 2001.

[10] MAPLE, `http://www.maplesoft.com/`.

[11] MATHEMATICA, `http://www.wolfram.com/`.

[12] MATLAB, `http://www.mathworks.com/`.

[13] Maxima, *A GPL CAS based on DOE-MACSYMA*, `http://maxima.sourceforge.net/`.

[14] V. Ovalle "Codis Binaris no lineals en MAGMA", Projecte de final de carrera, Universitat Autònoma de Barcelona, 2008.

[15] PARI, *A computer algebra system designed for fast computations in number theory*, `http://pari.math.u-bordeaux.fr/`.

[16] J. Pujol, J. Rifà and F. I. Solov'eva "Construction of $\mathbb{Z}_4$-linear Reed-Muller codes", submitted to *IEEE Trans. on Information Theory*, 2008.

[17] V. Shoup, NTL, `http://www.shoup.net/ntl/`

[18] Singular, *A computer algebra system for polynomial computations*, `http://www.singular.uni-kl.de/`.

[19] F. I. Solov'eva "On $\mathbb{Z}_4$-linear codes with parameters of Reed-Muller codes", *Problems of Information Transmission*, vol. 43(1), pp. 26-32, 2007.

[20] W. A. Stein, *Software for Algebra and Geometry Experimentation*, `http://www.sagemath.org/`.

[21] "The software test management guide", `http://www.rulebooks.co.uk/testguide`

[22] Z.-X. Wan, *Quaternary Codes*, World Scientific, 1997.

[23] "$\mathbb{Z}_2\mathbb{Z}_4$-Additive Codes in MAGMA" `http://ccg.uab.es`

Firmat: Bernat Gastón Brasó

Bellaterra, 12 de Juny de 2008

Capítol 8

Annex 1

Resum

En aquest projecte es presenta el desenvolupament d'un paquet d'aplicacions en l'entorn de programació matemàtica MAGMA, per al tractament dels codis anomenats $\mathbb{Z}_2\mathbb{Z}_4$-additius.

Els codis $\mathbb{Z}_2\mathbb{Z}_4$-additius permeten representar alguns codis binaris, com a codis lineals en l'espai dels codis $\mathbb{Z}_2\mathbb{Z}_4$-additius. Aquest fet permetrà l'estudi de tota una sèrie de codis binaris no lineals que fins ara eren intractables.

Resumen

En este proyecto se presenta el desarrollo de un paquete de aplicaciones en el entorno de programación matemática MAGMA, para el tratamiento de codigos $\mathbb{Z}_2\mathbb{Z}_4$-additivos.

Los codigos $\mathbb{Z}_2\mathbb{Z}_4$-additivos permiten representar algunos codigos binarios, como codigos lineales en el espacio de los codigos $\mathbb{Z}_2\mathbb{Z}_4$-additivos. Este hecho permitirá el estudio de toda una serie de codigos binarios no lineales que hasta ahora eran intratables.

Abstract

In this project we present the development of an application package in the mathematical programming environment MAGMA, for the treatment of $\mathbb{Z}_2\mathbb{Z}_4$-additive codes.

$\mathbb{Z}_2\mathbb{Z}_4$-additive codes allow us to represent some binary codes, as linear codes in the $\mathbb{Z}_2\mathbb{Z}_4$-additive code space. This fact will allow us to study some binary non linear codes which were impossible to work with until now.