Università degli Studi Roma Tre

Facoltà di Scienze Matematiche Fisiche e Naturali
Corso di Laurea in Matematica

Descrizione di attacchi algebrici all'Advanced Encryption Standard

Relatore
prof. Marco Pedicini

Candidato
Mario Mento

Anno Accademico 2004-2005

Keywords: Advanced Encryption Standard, sistemi di equazioni multivariate, ideale monomiale, base di Gröbner, algoritmo di Buchberger.

Mathematics Subject Classification: 13P10, 94A60, 68W30.

Prefazione

L'argomento principale di questa trattazione sono le basi di Gröbner. Queste ultime giocano un ruolo fondamentale all'interno degli attacchi algebrici. Negli ultimi anni è cresciuta l'attenzione nei confronti di questi nuovi metodi per crittoanalizzare i cifrari, ormai in grado di resistere agli attacchi classici, tipo crittoanalisi differenziale e lineare. La forza di questi attacchi sta nello sfruttare la struttura algebrica del cifrario da crittoanalizzare. Un attacco algebrico si divide fondamentalmente in due parti: la prima parte consiste nell'esprimere il crittosistema tramite un insieme di equazioni multivariate, la seconda consiste nel risolvere il sistema ottenuto per ricavare i bits che costituiscono la chiave.

Esistono vari metodi per risolvere i sistemi legati ai cifrari, noi descriveremo sinteticamente quelli basati sul metodo della linearizzazione ed approfondiremo quelli basati sulla teoria di Gröbner.

Scopo di questa tesi è di descrivere alcuni algoritmi che servono per determinare le basi di Gröbner; per fare ciò forniremo tutte le nozioni necessarie per la teoria delle basi di Gröbner, vedremo i vantaggi che si hanno nell'utilizzare questa tecnica al fine di risolvere sistemi di equazioni polinomiali multivariate e, infine, descriveremo gli algoritmi relativi.

La tesi è organizzata come segue:

- Capitolo 1: descrive in modo dettagliato l'Advanced Encryption Standard;
- Capitolo 2: introduce il concetto di attacco algebrico, descrive i tipi di sistemi che si ottengono dall'AES e elenca i metodi, derivati dalla tecnica di linearizzazione, per risolvere tali sistemi;
- Capitolo 3: descrive dettagliatamente come si può rappresentare algebricamente l'AES;
- Capitolo 4: introduce tutta la teoria sulle basi di Gröbner e l'algoritmo di Buchberger;
- Capitolo 5: spiega, attraverso il teorema di eliminazione, perché è vantaggioso usare le basi di Gröbner per risolvere un sistema di equazioni multivariate ed enuncia il teorema di estensione e lo Shape lemma;
- Capitolo 6: descrive alcuni tra gli algoritmi più efficienti, rispetto a quello di Buchberger originale, per il calcolo delle basi di Gröbner: F4, Gebauer e Möller ed F5.

Indice

1 L'Advanced Encryption Standard **1**
1.1 Storia di AES 1
1.1.1 Requisiti per l'AES 2
1.1.2 Processo di Selezione dell'AES 4
1.1.3 Breve descrizione degli algoritmi finalisti per AES 5
1.2 Descrizione di AES 7
1.2.1 Descrizione generale di AES 7
1.2.2 Descrizione dettagliata di AES 7

2 Introduzione agli attacchi algebrici per AES **13**
2.1 AES e gli Attacchi algebrici 13
2.2 La struttura base di AES 15
2.2.1 Osservazioni sulla struttura di AES 16
2.3 Il Big Encryption System (BES) 17
2.3.1 La struttura del BES 18
2.4 Crittoanalisi algebrica di AES 19
2.4.1 Sistema quadratico multivariato del BES 19
2.5 Tecniche di attacchi potenziali 21
2.5.1 Metodi di linearizzazione 21

3 Rappresentazione polinomiale dell'AES **23**
3.1 Descrizione dell'algoritmo del Rijndael nell'anello $\boldsymbol{R}$ 23
3.1.1 Espansione della chiave 25
3.1.2 Algoritmo di cifratura del Rijndael 25
3.1.3 Algoritmo di decifratura del Rijndael 25
3.2 Relazione con la descrizione standard 26
3.2.1 Trasformazione SubByte 27
3.2.2 La trasformazione ShiftRows 28
3.2.3 Le trasformazioni MixColumns ed AddRoundKey 28
3.2.4 AES-192 ed AES-256 28
3.3 Traccia e Basi duali 29

3.4 La struttura della S-box 32

4 Basi di Gröbner **36**
4.1 Varietà affini e ideali 36
4.2 Basi di Gröbner 40
4.2.1 Ordinamento dei monomi 41
4.2.2 Algoritmo della divisione in $k[x_1, \ldots, x_n]$ 44
4.2.3 Ideali monomiali e Lemma di Dickson 46
4.2.4 Teorema delle basi di Hilbert e Basi di Gröbner 48
4.3 Algoritmo di Buchberger 51
4.3.1 Miglioramenti dell'algoritmo di Buchberger 62
4.3.2 Osservazioni sui miglioramenti dell'algoritmo di Buchberger . 66

5 Teoria di eliminazione **67**
5.1 Il problema di risolvere equazioni polinomiali 67
5.2 I teoremi di Eliminazione ed Estensione 68
5.3 Lo Shape lemma 73

6 Tecniche avanzate sulle basi di Gröbner **77**
6.1 Moduli 78
6.2 Polinomi ed algebra lineare 79
6.3 Algoritmo F4 81
6.4 L'installazione di Gebauer e Möller e l'algoritmo F5 85
6.4.1 Criterio di selezione dell'algoritmo F5 91
6.4.2 Dimostrazione del criterio dell'algoritmo F5 94

7 Conclusioni **100**

Bibliografia **104**

Capitolo 1

L'Advanced Encryption Standard

L'*Advanced Encryption Standard* (AES), conosciuto anche come Rijndael, è un algoritmo di cifratura a blocchi utilizzato come standard dal governo degli Stati Uniti d'America. Data la sua sicurezza e le sue specifiche pubbliche si presume che, in un prossimo futuro, venga utilizzato in tutto il mondo, come è successo al suo predecessore il Data Encryption Standard (DES). È stato adottato dal National Institute of Standards and Technology (NIST) e dalla US FIPS PUB 197 nel novembre del 2001, dopo 5 anni di studi e standardizzazioni. L'algoritmo è stato sviluppato da due crittografi belgi, Joan Daemen e Vincent Rijmen, che lo hanno presentato al processo di selezione per l'AES con il nome di Rijndael, nome derivato dai nomi degli inventori. Il Rijndael è un'evoluzione del primo algoritmo, chiamato Square, sviluppato da Daemen e Rijmen per SHARK. A differenza del DES, Rijndael è una rete a sostituzione e permutazione, non una rete di *Feistel*. AES è veloce, sia se sviluppato in software, sia se sviluppato in hardware, ed è relativamente semplice da implementare e richiede poca memoria. Il nuovo standard di cifratura sta sostituendo i precedenti standard e la sua diffusione continua ad aumentare.

1.1 Storia di AES

Per molti anni il DES ha rappresentato lo standard per la cifratura e l'autenticazione di documenti. Il National Institute of Standard and Technology (NIST) lo scelse come standard di cifratura nel 1977 con validità quinquennale. Fu riaffermato nel 1983, 1988 ed infine nel 1993 con una clausola che ne confermava la validità fino al Dicembre del 1998. La clausola specificava che, allo scadere di tale data, il NIST avrebbe dovuto prendere in considerazione delle alternative che offrissero un maggior grado di sicurezza. Sin dall'inizio il DES fu oggetto di molte critiche

che misero in discussione la sua forza crittografica; in riferimento alla sicurezza dello schema e a dispetto delle 256 chiavi necessarie per un attacco brutale la metà di esse, in media, ne permette la violazione. Ciò portò alla considerazione che la lunghezza della chiave era troppo piccola.

In riferimento all'utilizzo delle S-box nelle procedure di cifratura e decifratura si sospettava che queste strutture nascondessero delle trapdoor.

In vista della data di scadenza della validità del DES ed a causa di tali critiche, il NIST considerò delle alternative che offrissero un livello di sicurezza maggiore. Sebbene il triplo DES potesse essere considerato un valido sostituto, il NIST volle selezionare come nuovo standard un algoritmo di cifratura più sicuro ed efficiente. A tal fine, il 12 Settembre del 1997, il NIST indisse un concorso pubblico per la nomina dell'Advanced Encryption Standard (A.E.S.). L'obiettivo fondamentale del NIST era quello di stabilire un nuovo standard che diventasse il punto di riferimento, nel prossimo secolo, nel campo della sicurezza. Il NIST richiese ai partecipanti un completo pacchetto di documentazione che contenesse quanto segue:

1. una completa descrizione dell'algoritmo, comprendente tutte le equazioni matematiche, le tabelle, i diagrammi ed i parametri necessari all'implementazione dell'algoritmo;
2. una stima dell'efficienza computazionale, comprendente le seguenti informazioni:
   a. una descrizione della piattaforma utilizzata per generare tale stima;
   b. un'analisi delle velocità dell'algoritmo sulle suddette piattaforme;
   c. qualunque altra informazione essenziale per l'analisi dell'algoritmo;
3. un'analisi dell'algoritmo rispetto agli attacchi di crittoanalisi più conosciuti (ad esempio known o chosen plaintext);
4. un elenco dei vantaggi e dei limiti dell'algoritmo;
5. un' implementazione di riferimento in ANSI C con appropriati commenti;
6. un' implementazione ottimizzata dell'algoritmo in ANSI C e Java.

1.1.1 Requisiti per l'AES

Il NIST richiese per gli algoritmi candidati i seguenti requisiti:

1. l'algoritmo doveva implementare un cifrario a chiave simmetrica;
2. l'algoritmo doveva appartenere alla classe dei cifrari a blocchi;

3. l'algoritmo doveva supportare chiavi con taglia compresa tra 128 e 256 bit e lunghezza del testo in chiaro di 128 bit (era preferibile che l'algoritmo fosse in grado di gestire ulteriori taglie, quali ad esempio 64 e 256 bit);

4. la struttura dell'algoritmo doveva essere tale da permetterne l'implementazione su smart-card;

5. l'algoritmo doveva essere disponibile a livello mondiale, senza esclusive e royalty-free.

Ciascuno dei candidati fu sottoposto ad una serie di test ed analisi atti a valutare le caratteristiche computazionali e di sicurezza. I criteri di valutazione scaturirono dai commenti pubblici e dalle discussioni tenutesi il 15 Aprile 1997 presso il NIST.

A tal proposito si individuarono tre categorie principali:

SICUREZZA

La sicurezza rappresentò il fattore più importante della valutazione. Ogni algoritmo, per essere definito minimamente sicuro, doveva essere sottoposto alle seguenti prove:

1. la sicurezza dell'algoritmo doveva essere confrontata con quella degli altri candidati;

2. l'output dell'algoritmo doveva essere indistinguibile da una permutazione casuale del blocco in input;

COSTO

L'algoritmo candidato doveva avere una serie di attributi che ne determinassero un basso costo computazionale. Tale costo fu valutato in base ai seguenti criteri:

1. efficienza computazionale: la valutazione dell'efficienza computazionale dell'algoritmo doveva riferirsi sia all'implementazione hardware che software;

2. requisiti di memoria: si doveva considerare la grandezza del codice e la memoria richiesta.

ALGORITMO E CARATTERISTICHE DI IMPLEMENTAZIONE

La terza area di valutazione riguardò l'algoritmo e le caratteristiche implementative, quali la flessibilità, l' adattabilità software ed hardware e la semplicità.

L' algoritmo doveva:

1. gestire dimensioni della chiave e dei blocchi superiori alle taglie richieste;

2. essere implementato in modo sicuro ed efficiente in diversi ambienti;

3. essere implementato come uno stream cipher, un algoritmo di hashing e fornire servizi crittografici addizionali.

1.1.2 Processo di Selezione dell'AES

Il 2 gennaio del 1997 il NIST annunciò un programma per sviluppare e scegliere un Advanced Encryption Standard (AES) per rimpiazzare il DES. Il 12 settembre il NIST indisse un concorso pubblico per la nomina dell' AES, sollecitando la comunità crittografica mondiale per la presentazione di nuovi algoritmi crittografici.

Il 20 Agosto 1998, in California, si tenne la prima conferenza per la candidatura all' AES, durante la quale, il NIST annunciò i 15 candidati ufficiali, elencati nella seguente tabella 1.1 .

Nome dell'Algoritmo	**Autori**
Mars	IBM
RC6	RSA Laboratories
RIJDAEL	Joan Daemen,Vincent Rijmen
SERPENT	Ross Anderson,Eli Biham,Lars Knudsen
TWOFISH	B. Schneider,J. Kelsey,D. Whiting,D. Wagner, C. Hall,N., Ferguson
CAST-256	Entrust Technologies,INC.
CRYPTON	Future System,INC.
DEAL	R.Outerbridge,L.Knudsen
DFC	CNRS
E2	Nippon Telegraf and Telephone Corp.
FROG	TecApro International S.A.
HPC	L.Brown,J.Pieprzyk,J.Seberry
LOKI97	L.Brown,J.Pieprzyk,J.Seberry
MAGENTA	Detsche Telekom AG
SAFER+	Cylink Corp.

Tabella 1.1: Candidati ufficiali

Questi algoritmi furono sottoposti alla comunità crittografica mondiale e, in questa conferenza, il NIST sollecitò commenti pubblici sui candidati. Una seconda conferenza si tenne il 22 Marzo 1999 per discutere i risultati delle analisi condotte dalla comunità crittografica mondiale sugli algoritmi candidati. Il periodo dello scrutinio pubblico riguardante le recensioni iniziali si concluse il 15 Aprile del 1999. Utilizzando le analisi ed i commenti ricevuti, il NIST selezionò 5 finalisti dei 15 algoritmi presentati, quì elencati nella tabella 1.2.

Dopo l'annuncio dei finalisti, il NIST aprì formalmente il processo di valutazione pubblico accettando fino al 15 Maggio 2000 i commenti sugli algoritmi rimasti in gara. Il NIST ricercò attivamente commenti ed analisi su qualsiasi aspetto degli algoritmi candidati.

Nome dell'Algoritmo	Autori
Mars	IBM
RC6	RSA Laboratories
RIJDAEL	Joan Daemen,Vincent Rijmen
SERPENT	Ross Anderson,Eli Biham,Lars Knudsen
TWOFISH	B. Schneider,J. Kelsey,D. Whiting,D. Wagner, C. Hall,N., Ferguson

Tabella 1.2:

Il 13 e 14 Aprile 2000 a New York il NIST sponsorizzò la Third AES Candidate Conference, un forum pubblico, aperto alla discussione delle analisi sui finalisti. Gli autori degli algoritmi finalisti furono invitati ad assistere e partecipare alla discussione riguardante i commenti sui propri algoritmi.

Dopo il 15 Maggio 2000, il NIST studiò tutte le informazioni disponibili e il 2 Ottobre 2000 selezionò Rijndael quale algoritmo da proporre per l'AES.

Nel Febbraio del 2001 l' AES fu presentato come un Draft-Federal Information Processing Standard (Draft-FIPS) e pubblicato per commenti e critiche. Terminato il periodo di 90 giorni di discussione nel Maggio del 2001, dopo averlo corretto in modo appropriato in risposta ai commenti, il NIST iniziò una fase di recensione, pubblicazione e diffusione.

Dopo questa fase intermedia, il Segretario del Dipartimento del Commercio americano, il 26 Novembre 2001, ha approvato in via definitiva Rijndael quale nuovo standard AES (con FIPS PUB 197). Lo standard prevede una taglia del blocco di input di 128 bit, mentre per la chiave di cifratura prevede tre possibili lunghezze 128, 192 e 256 bit (rispettivamente AES-128, AES-192 e AES-256). Lo standard diventerà effettivo il 26 Maggio del 2002, sei mesi dopo l'approvazione.

1.1.3 Breve descrizione degli algoritmi finalisti per AES

Di seguito riportiamo, per completezza, un sommario di ognuno dei 5 finalisti in ordine alfabetico:

MARS incorpora il suo cryptographic core in una struttura nel complesso innovativa ed eterogenea. Utilizza una gran varietà d'operazioni, tra cui tecniche di rotazione dei dati di una quantità variabile determinata sia dai dati che dalla chiave segreta. Di conseguenza, non solo il MARS è efficiente in generale, ma lo è particolarmente su piattaforme che supportano efficientemente le operazioni di rotazione e moltiplicazione. Il NIST ha accettato una modifica al MARS per il secondo round (proposta dall'autore stesso) che dovrebbe migliorarne la flessibilità e l'adattabilità

ad ambienti con poca memoria, come le smart-card a basso costo. MARS è stato presentato all'AES dall'IBM (International Business Machines Corporation).

RC6 è un' algoritmo di dimensioni contenute e dovrebbe essere facile da implementare in modo compatto sia in software che in hardware; inoltre la sua semplicità dovrebbe facilitare le analisi di sicurezza cui sarà sottoposto nel secondo round, in quanto, oltretutto, ci si può basare sulle analisi eseguite sul predecessore, RC5. L'RC6 non utilizza tavole di sostituzione; la principale fonte per la sua sicurezza è la tecnica di rotazione dei dati di una quantità variabile determinata dai dati stessi. L'RC6 è veloce in generale, ma lo è particolarmente sulle piattaforme che supportano efficientemente le operazioni di rotazione e moltiplicazione. RC6 è stato proposto all'AES dai laboratori RSA.

RIJNDAEL viene eseguito in modo eccellente su tutte le piattaforme considerate. La sua schedulazione della chiave è veloce, e le sue richieste di memoria sono basse, per cui dovrebbe essere efficiente in hardware e in ambienti con poca memoria. Anche se, nel processo di selezione del primo round, gli eventuali vantaggi tratti dall'utilizzo di processori paralleli non sono stati considerati, Rijndael è in grado di usufruire dei vantaggi offerti dai processori che permettono l'esecuzione di molte istruzioni in parallelo. Rijndael è stato presentato all'AES da Joan Daemen e Vincent Rijmen.

SERPENT È ultra conservativo nei suoi margini di sicurezza; i progettisti hanno scelto di utilizzare il doppio delle iterazioni che ritenevano sicure contro gli attacchi attualmente conosciuti. Di conseguenza le performance di Serpent sono relativamente basse, comparate con quelle degli altri quattro finalisti; tuttavia questo difetto può essere attenuato dall'efficienza delle implementazioni ottimizzate utilizzando quello che gli autori chiamano la modalità "bitslice" per la quale l'algoritmo era stato progettato. Serpent dovrebbe adattarsi bene all'hardware e agli ambienti con scarsa quantità di memoria. La progettazione semplice e la scelta conservativa delle operazioni dovrebbero facilitare le ulteriori analisi. Serpent è stato presentato all'AES da Ross Anderson, Eli Biham e Lars Knudsen.

TWOFISH ha performance veloci e versatili sulla maggior parte delle piattaforme; inoltre dovrebbe avere buone prestazioni sia in hardware che in ambienti con scarsa quantità di memoria. Utilizza tavole di sostituzione variabili che dipendono dalla chiave segreta. Gli autori credono che queste tavole in genere offrano maggiore sicurezza rispetto a tavole con valori fissati. La possibilità di precalcolare queste tavole aiuta Twofish ad offrire una gran varietà di compromessi sulle performance; a seconda dei settaggi Twofish può essere ottimizzato per la velocità, per la schedulazione della chiave, per la memoria, per la dimensione del codice in software o dello spazio in hardware. Twofish è stato presentato all'AES da Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall e Niels Ferguson.

1.2 Descrizione di AES

Formalmente AES non è equivalente al Rijndael (sebbene nella pratica siano intercambiabili) dato che il Rijndael gestisce molte dimensioni di blocchi e di chiavi. Nell'AES il blocco è di dimensione fissa (128 bit) e la chiave può essere di 128, 192 o 256 bit mentre il Rijndael specifica solo che il blocco e la chiave devono essere un multiplo di 32 bit con 128 bit come minimo e 256 bit come massimo. AES opera utilizzando matrici di 4 byte chiamate stati (*State*) (il Rijndael ha un numero addizionale di colonne negli stati). Per cifrare, ogni round dell'AES (eccetto il primo e l'ultimo) esegue i seguenti quattro passaggi:

1. *SubBytes*: Sostituzione non lineare di tutti i byte che vengono rimpiazzati secondo una specifica tabella.
2. *ShiftRows*: Spostamento dei byte di un certo numero di posizioni dipendente dalla riga di appartenenza.
3. *MixColumns*: Combinazione dei byte con un operazione lineare, i byte vengono trattati una colonna per volta.
4. *AddRoundKey*: Ogni byte della tabella viene combinato con la chiave del round, la chiave del round viene calcolata dalla funzione che genera tutto l'insieme delle chiavi (key schedule).

1.2.1 Descrizione generale di AES

Diamo una descrizione generale di AES. L'algoritmo si sviluppa come segue.

1. Dato un plaintext x, inizializza $State = x$ ed esegue un operazione chiamata *AddRoundKey*, che fa lo x-or tra *RoundKey* e lo *State*.
2. Per ognuno dei primi Nr-1 Round, esegue un'operazione sullo *State* chiamata *SubBytes* usando una S-box; esegue una permutazione dello *State*: *ShiftRows*; esegue una operazione sullo *State*: *MixColumns*; esegue *AddRoundKey*.
3. Esegue *SubBytes*; esegue *ShiftRows*; esegue *AddRoundKey*.
4. Pone il ciphertext $y = State$.

1.2.2 Descrizione dettagliata di AES

Diamo, adesso, una descrizione precisa di tutte le operazioni usate in AES; descriviamo la struttura di *State* e analizziamo la costruzione del key-schedule. Tutte le operazioni in AES sono byte-oriented, e tutte le variabili usate sono formate da un numero appropriato di bytes. Il plaintext x è formato da 16 bytes, $x_0, \cdots x_{15}$. Lo *State* è rappresentato come una matrice 4×4 di bytes:

	Y															
X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	052	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	E4	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Tabella 1.3: AES S-box π_s

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

Inizialmente lo State è definito dai 16 bytes del plaintext x come segue:

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$\leftarrow$

x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

Useremo spesso la notazione esadecimale per rappresentare il contenuto di un byte. Ogni byte consiste di due cifre esadecimali.

SubByte

L'operazione *SubByte* compie una sostituzione su ogni byte dello *State* indipendentemente, usando una S-box, π_s , che è una permutazione di $\{0,1\}^8$. Per analizzare π_s rappresentiamo i byte in notazione esadecimale. Rappresentiamo π_s come una matrice 16×16 (tabella 1.3), dove le righe e le colonne sono indicizzate da cifre esadecimali. L'entrata nella riga X e la colonna Y è $\pi_s(XY)$.

La S-box di AES può essere definita algebricamente. La formulazione della S-box di AES coinvolge operazioni in un campo finito:

$$\mathbb{F}_{2^8} = \mathbb{Z}_2[x]/(x^8 + x^4 + x^3 + x + 1)$$

Sia *FildInv* l'indicatore dell'inverso moltiplicativo di un elemento del campo; *BinaryToField* converte un byte in un elemento del campo e *FieldToBinary* esegue

la conversione inversa. Questa conversione avviene nel modo più ovvio: l'elemento del campo

$$\sum_{i=0}^{7} a_i x^i$$

corrisponde al byte

$$a_7a_6a_5a_4a_3a_2a_1a_0$$

dove $a_i \in \mathbb{Z}_2, 0 \leq i \leq 7$. Così π_s è definita dall'algoritmo nel quale gli otto bits di input $a_7a_6a_5a_4a_3a_2a_1a_0$ sono sostituiti dagli otto bits di output $b_7b_6b_5b_4b_3b_2b_1b_0$.

$$\mathit{SubBytes}(a_7a_6a_5a_4a_3a_2a_1a_0\,) \rightarrow b_7b_6b_5b_4b_3b_2b_1b_0$$

Algoritmo 1.1. SubBytes$(a_7a_6a_5a_4a_3a_2a_1a_0)$

Funzioni Esterne: *FieldInv*, *BinaryToField*, *FieldToBinary*

$\mathit{BinaryToField}(a_7a_6a_5a_4a_3a_2a_1a_0) \rightarrow z$
if $z \neq 0$
then $\mathit{FieldInv}(z) \rightarrow z$
$\mathit{FieldToBinary}(z) \rightarrow (a_7a_6a_5a_4a_3a_2a_1a_0)$
$(01100011) \rightarrow (c_7c_6c_5c_4c_3c_2c_1c_0)$

Osservazione: nel seguente ciclo, i pedici sono ridotti $\pmod 8$

for $i \rightarrow 0$ **to** 7
do $(a_i + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7} + c_i) \pmod 2 \rightarrow b_i$
return $(b_7b_6b_5b_4b_3b_2b_1b_0)$

Esempio 1.1. Diamo un piccolo esempio per illustrare l'Algoritmo 1.1 dove includiamo anche la conversione in esadecimale. Supponiamo di avere il numero esadecimale 53; verifichiamo che 53 in binario è rappresentato dal byte 01010011:

$$53 \rightarrow 01010011,$$

infatti

$$53 \rightarrow 5 \cdot 16 + 3 = 83,$$

e 83 in binario è:

$$83 = 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 2^0$$

quindi 53 in notazione binaria diventa:

$$01010011$$

che rappresentato come elemento del campo diventa:

$$x^6 + x^4 + x + 1.$$

L'inverso moltiplicativo (nel campo $\mathbb{F}_{2^8}$) è:

$$x^7 + x^6 + x^3 + x.$$

Quindi in notazione binaria abbiamo:

$$(a_7a_6a_5a_4a_3a_2a_1a_0) = (11001010).$$

Poi calcoliamo:

$$\begin{aligned} b_0 &= a_0 + a_4 + a_5 + a_6 + a_7 + c_0 \mod 2 \\ &= 0 + 0 + 0 + 1 + 1 + 1 \mod 2 \\ &= 1 \\ b_1 &= a_1 + a_5 + a_6 + a_7 + a_0 + c_1 \mod 2 \\ &= 1 + 0 + 1 + 1 + 0 + 1 \mod 2 \\ &= 0, \end{aligned}$$

etc. Il risultato è:

$$(b_7b_6b_5b_4b_3b_2b_1b_0) = (11101101).$$

In notazione esadecimale 11101101 è ED. Questo calcolo può essere verificato controllando che nella tabella 1.3 l'elemento della riga 3 e della colonna 5 è proprio ED.

ShiftRows

L'operazione $ShiftRows$ agisce sullo $State$ come mostrato nel seguente diagramma:

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$\leftarrow$

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,0}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,1}$
$s_{3,3}$	$s_{3,0}$	$s_{3,1}$	$s_{3,2}$

MixColumns

L'operazione $MixColumns$ è eseguita in ognuna delle quattro colonne dello *State*; è presentata nell'algoritmo 1.2. Ogni colonna dello *State* è rimpiazzata da una nuova colonna che è costituita moltiplicando la colonna precedente per una matrice, i cui elementi sono nel campo $\mathbb{F}_{2^8}$. La funzione esterna $FieldMult$ prende in input due elementi del campo, e calcola il loro prodotto nel campo. L'addizione nel campo è semplicemente la somma modulo due (cioè lo xor delle stringhe corrispondenti). Nell'algoritmo 1.2 questa operazione è denotata con $\oplus$.

Algoritmo 1.2. MixColumn(c)

Funzioni Esterne: $FieldMult$, $BinaryToField$, $FieldToBinary$
for $i \rightarrow 1$ **to** 3
    do $BinaryToField(s_{i,c}) \rightarrow t_i$
$FieldMult(x, t_0) \oplus FieldMult(x+1, t_1) \oplus t_2 \oplus t_3 \rightarrow u_0$
$FieldMult(x, t_1) \oplus FieldMult(x+1, t_2) \oplus t_3 \oplus t_0 \rightarrow u_1$
$FieldMult(x, t_2) \oplus FieldMult(x+1, t_3) \oplus t_0 \oplus t_1 \rightarrow u_2$
$FieldMult(x, t_3) \oplus FieldMult(x+1, t_0) \oplus t_1 \oplus t_2 \rightarrow u_3$
for $i \rightarrow 0$ **to** 3
    do $s_{i,c} \rightarrow FieldToBinary(u_i)$

KeyExpansion

Rimane da descrivere come viene generato l'insieme delle chiavi (key schedule) di AES. Diamo questa descrizione per il key schedule della versione di AES a 10 round, che usa una chiave a 128-bit (la generazione delle chiavi per le versioni a 12 e 14 round sono simili a quella di 10 round, con qualche differenza poco importante). Abbiamo bisogno di chiavi per 11 round, ognuna delle quali è costituita da 16 bytes. L'algoritmo che genera le chiavi è *word-oriented*, dove una parola (word) è formata da 4 bytes o, equivalentemente, da 32 bits. Quindi ogni chiave di un round è costituita da 4 parole. L'insieme delle chiavi di tutti i round è chiamato chiave espansa (expanded-key), è formato da 44 parole ed è denotato con $w[0], \ldots, w[43]$, dove ogni $w[i]$ è una parola. La chiave espansa è costruita usando la funzione $KeyExpansion$, che è presentata nell'algoritmo 1.3.

L'input di questo algoritmo è la chiave di 128 bit, key, che è utilizzata come un vettore di bytes, $key[0], \ldots, key[15]$; l'output è il vettore delle parole, w, che abbiamo prima introdotto.
La funzione $KeyExpansion$ incorpora altre due funzioni, che sono chiamate $RotWord$ e $SubWord$. La funzione $RotWord$ esegue uno spostamento (shift) ciclico a

sinistra dei quattro bytes B_0, B_1, B_2, B_3, cioè:

$$RotWord(B_0, B_1, B_2, B_3) = (B_1, B_2, B_3, B_0)\,;$$

mentre la funzione $SubWord(B_0, B_1, B_2, B_3)$ applica la AES S-box ad ognuno dei quattro bytes B_0, B_1, B_2, B_3, cioè:

$$SubWord(B_0, B_1, B_2, B_3) = (B_0^{'}, B_1^{'}, B_2^{'}, B_3^{'})$$

dove $B_i^{'} = SubBytes(B_i)$, con $i = 0, \ldots, 3$. *Rcon* è un vettore di 10 parole, indicato con $Rcon[1], \ldots, Rcon[10]$. Tutte queste sono costanti definite in notazione esadecimale all'inizio dell'algoritmo 1.3.

Algoritmo 1.3. KeyExpansion(key)

Funzioni Esterne: $RotWord$, $SubWord$
01000000 $\rightarrow RCon[1]$
02000000 $\rightarrow RCon[2]$
04000000 $\rightarrow RCon[3]$
08000000 $\rightarrow RCon[4]$
10000000 $\rightarrow RCon[5]$
20000000 $\rightarrow RCon[6]$
40000000 $\rightarrow RCon[7]$
80000000 $\rightarrow RCon[8]$
1B000000 $\rightarrow RCon[9]$
36000000 $\rightarrow RCon[10]$
for $i \rightarrow 1$ **to** 3
    do $(key[4i], key[4i+1], key[4i+2], key[4i+3]) \rightarrow w[i]$
for $i \rightarrow 0$ **to** 43 **do**
    $w[i-1] \rightarrow temp$
    if $i \equiv 0 \pmod 4$ **then**
      $SubWord(RotWord(temp)) \oplus Rcon[i/4] \rightarrow temp$
    $w[i-4] \oplus temp \rightarrow w[i]$
return $(w[0], \ldots, w[43])$

Abbiamo, così, descritto tutte le operazioni richieste per eseguire una cifratura con AES. Per decriptare è necessario eseguire tutte le operazioni al contrario e usare il key schedule al contrario. Le funzioni $ShiftRows$, $SubBytes$ e $MixColumns$ devono essere rimpiazzate dalle loro inverse (l'operazione $AddRoundKey$ è essa stessa la sua inversa). È possibile costruire un cifrario inverso equivalente che esegue la decifratura di AES facendo una serie di operazioni (inverse) nello stesso ordine della cifratura.

Capitolo 2

Introduzione agli attacchi algebrici per AES

In questo capitolo introduciamo il concetto di attaccho algebrico e, in particolare, descriveremo in forma sintetica molti metodi proposti da vari autori per attaccare l'Advanced Encryption Standard.

Fondamentalmente un attacco algebrico ad AES si divide in due parti: la prima consiste nell'esprimere il crittosistema tramite un insieme di equazioni multivariate, la seconda consiste nel risolvere il sistema ottenuto per ricavare i bits che costituiscono la chiave. In questa tesi concentriamo la nostra attenzione sul secondo aspetto, in particolare, sulla teoria delle basi di Gröbner, che è il fondamento di molti metodi usati per risolvere i sistemi di equazioni polinomiali.

Adesso, ci occuperemo di come ottenere i sistemi che rappresentano AES, ed inoltre, dei metodi basati sulla linearizzazione per risolvere tali sistemi.

2.1 AES e gli Attacchi algebrici

Come abbiamo già visto nel capitolo 1, nel 1997 lo US National Institute of Standard and Technology (NIST) bandì una competizione, aperta a tutti, per selezionare un sostituto per il vecchio Data Encryption Standard (DES), che lavorava con blocchi da 64 bits e chiave da 56 bits. Nel 2000 il NIST annunciò che il candidato *Rijndael* era il vincitore.

Il nuovo standard di cifratura, chiamato Advanced Encryption Standard (AES), lavora con blocchi lunghi 128 bits e chiavi di lunghezze variabili: 128 bits, 192 bits e 256 bits. L'AES ha rimpiazzato il DES per proteggere informazioni "segrete ma non classificate"; inoltre, l'AES è utilizzato come standard anche nel settore privato, in particolare nel commercio, nelle banche e nelle industrie.

Il Rijndael ha una struttura semplice ed elegante. È stato progettato per re-

sistere ai *known attacks* (cioè quegli attacchi in cui l'attaccante è in possesso della stringa di plain text e della corrispondente stringa di cipher text), in particolare alla crittoanalisi differenziale e lineare.

Il Rijndael ha una struttura altamente algebrica. Le trasformazioni del round di cifratura sono basate su operazioni nel campo finito $\mathbb{F}_{2^8}$; tutto ciò ha portato ad un crescente interesse nell'aplicazione di tecniche algebriche nella crittoanalisi dei cifrari a blocchi, perché i metodi tradizionali di crittoanalisi (come la crittoanalisi differenziale e lineare) sono basati su approcci statistici. Molti cifrari moderni sono progettati per essere resistenti a questo genere di attacchi. Inoltre, proprio per la natura di questi attacchi, la loro complessità solitamente cresce esponenzialmente col numero dei round e, quindi, diventano velocemente impraticabili.

Al contrario, gli *attacchi algebrici* sfruttano la struttura algebrica intrinseca del cifrario. Gli attaccanti esprimono la trasformazione di cifratura come un insieme di equazioni polinomiali multivariate e, successivamente, tentano di risolvere tale sistema per ottenere la chiave di cifratura. Gli attacchi algebrici aprono nuove prospettive nella crittoanalisi dei cifrari a blocchi, poiché possono essere visti come un attacco strutturale al cifrario e, in caso di successo, non basta modificare il cifrario aumentando il numero di round per renderlo di nuovo sicuro a quel tipo di attacco. Ci sono, infatti, indicazioni che la complessità di tali attacchi non cresce esponenzialmente con il numero di round [**CP02**].

Mentre, in teoria, molti moderni cifrari a blocchi possono essere pienamente descritti da un sistema di equazioni polinomiali multivariate su un campo finito, nella pratica, per la maggior parte dei casi questi sistemi sono troppo complessi per essere risolti. Ci sono ancora dei cifrari di recente progettazione con una struttura altamente algebrica che, quindi, sono vulnerabili agli attacchi algebrici [**BC03**]. Nel caso di AES, Courtois e Pieprzyk hanno mostrato in [**CP02**] come è possibile esprimere le operazioni di cifratura come un largo, sparso e sovradefinito sistema di equazioni multivariate quadratiche su $\mathbb{F}_2$. Spieghiamo cosa vuol dire che le equazioni sono sparse e che il sistema è sovradefinito. Sia t il numero di monomi che compaiono nel sistema, sia d il grado di una determinata equazione del sistema, sia r il numero delle equazioni che formano il sistema, allora diciamo che le equazioni sono *sparse* se si ha che $t \ll \binom{s}{d}$ (di solito $t \approx \binom{s}{d}$), e che il sistema è *sovradefinito* se $r \gg s$. Nello stesso articolo loro proposero un metodo chiamato XSL per ottenere la chiave di cifratura per alcuni cifrari.

Nello stesso periodo Murphy e Robshaw [**MR02**] hanno mostrato come esprimere la funzione di cifratura di AES con un più semplice sistema di equazioni su $\mathbb{F}_{2^8}$. Anche se XSL è un metodo valido, quest'ultimo sistema su $\mathbb{F}_{2^8}$ sembra essere più veloce da risolvere rispetto a quello su $\mathbb{F}_2$, ed in teoria, può fornire un attacco più efficiente per la ricerca della chiave, e ciò potrebbe essere devastante per AES.

In questo capitolo diamo una descrizione sintetica dei metodi sviluppati per attaccare algebricamente AES e consideriamo qualche metodo per risolvere i cor-

rispondenti sistemi di equazioni multivariate, usando tecniche di geometria algebrica e algebra commutativa.

2.2 La struttura base di AES

Anche se nel capitolo 3 daremo una dettagliata descrizione della struttura algebrica di AES, abbiamo bisogno adesso di darne una semplice per poter descrivere il BES.

Il cifrario AES cifra blocchi di 16 byte (128 bit), usando una chiave di 16 byte, attraverso 10 round di cifratura [**MR02**].

L'input della funzione di cifratura di un round di AES può essere visto come una matrice i cui elementi sono i byte o, equivalentemente, come un vettore colonna di byte, chiamato *state*.

$$\mathbf{a} = \begin{array}{|c|c|c|c|} \hline a_{00} & a_{01} & a_{02} & a_{03} \\ \hline a_{10} & a_{11} & a_{12} & a_{13} \\ \hline a_{20} & a_{21} & a_{22} & a_{23} \\ \hline a_{30} & a_{31} & a_{32} & a_{33} \\ \hline \end{array} = (a_{00}, \ldots, a_{30}, a_{01} \ldots, a_{31}, a_{02}, \ldots, a_{33})^T$$

In AES, ogni byte è visto come un elemento del campo

$$\mathbb{F}_{2^8} = \frac{\mathbb{F}_2[x]}{\langle \mu(x) \rangle},$$

dove $\mu(x) \in \mathbb{F}_2[x]$ è il polinomio irriducibile $x^8 + x^4 + x^3 + x + 1$. Ogni round (tranne il primo e l'ultimo che sono leggermente differenti) di AES esegue le tre seguenti operazioni:

1. **S-box** Questa è l'unica trasformazione non lineare del cifrario. Il valore di ogni byte della matrice è sostituito secondo una tabella di sostituzioni (vedi tabella 1.3). Questa tabella è ottenuta combinando tre trasformazioni:

   (a) Sia w l'input, allora $x = w^{(-1)}$, dove $w^{(-1)}$ è definita come segue:

   $$w^{(-1)} = w^{254} = \begin{cases} w^{(-1)} & \text{se } w \neq 0 \\ 0 & \text{se } w = 0 \end{cases}$$

   Così l'inversione in AES è identica all'inversione standard in $\mathbb{F}_{2^8}$ per elementi del campo diversi da zero, con $0^{(-1)} = 0$.

   (b) Il valore intermedio x è considerato come un vettore in $\mathbb{F}_2$ di dimensione 8 e trasformato usando una matrice L_A, 8×8 con elementi in $\mathbb{F}_2$. Il vettore trasformato $L_A \cdot x$ è dunque considerato come un elemento di

$\mathbb{F}_{2^8}$. La matrice L_A ha la seguente espressione:

$$L_A := \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

(c) L'output della S-box è $((L_A \cdot x) + d)$, dove d è una costante appartenente ad $\mathbb{F}_{2^8}$, vedi capitolo 3.

2. **Diffusione lineare**

   (a) Ogni riga della matrice viene ruotata di un determinato numero di posizioni dei byte. Questa operazione è chiamata $ShiftRow$.

   (b) Ogni colonna $\mathbf{y}$ della matrice è considerata come un vettore di $\mathbb{F}_{2^8}^4$, ed è trasformata nella colonna $C \cdot \mathbf{y}$, dove C è una matrice 4×4 con elementi in $\mathbb{F}_{2^8}$. Questa operazione è chiamata $MixColumn$.

3. **L'addizione della chiave**

   Prima di iniziare la cifratura, la chiave originale viene espansa in 11 sottochiavi (si genera il $KeyShedule$), ognuna di 16 bytes, usate nei rispettivi round; quindi, dopo ogni diffusione lineare, ogni byte della matrice viene sommato in ($\mathbb{F}_{2^8}$) con un byte della corrispondente sottochiave.

2.2.1 Osservazioni sulla struttura di AES

In [**MR00**] Murphy e Robshaw fanno le seguenti osservazioni sulla struttura di ogni round di AES.

- La costante additiva $d \in \mathbb{F}_{2^8}$ della S-box può essere rimossa incorporandola nell'insieme delle chiavi.
- La trasformazione lineare L_A nella S-box può essere vista come una parte di una diffusione lineare più grande. Questo ci permette di dare una descrizione della funzione del round differente:
  1. *S-box*: inversione in $\mathbb{F}_{2^8}$;
  2. *Diffusione lineare*: trasformazione lineare M in $\mathbb{F}_2$;
  3. *Sottochiave*: addizione con la sottochiave del round.

- La nuova trasformazione lineare M ha una struttura particolare, descritta in [**MR00**].

2.3 Il Big Encryption System (BES)

Una delle maggiori difficoltà che si incontra nel'analisi delle propietà descritte prima per la crittoanalisi di AES è l'esistenza di operazioni su due campi differenti: $\mathbb{F}_{2^8}$ e $\mathbb{F}_2$. Per risolvere il conflitto tra queste due operazioni in AES, è stato introdotto in [**MR02**] un nuovo cifrario a blocchi iterato. Il *Big Encryption System* (BES) opera su blocchi da 128 bytes con chiavi da 128 bytes (mentre AES opera su 128 "bits"). Il BES ha una struttura algebrica molto semplice: un round del cifrario esegue le operazioni di *inversione*, *prodotto per una matrice* e *somma con la chiave*; tutte le operazioni sono sul campo $\mathbb{F}_{2^8}$.

Sia l'AES che il BES usano la matrice di byte dello stato che viene trasformata in ogni round dalle operazioni che abbiamo descritto. Gli spazi dello stato di AES e del BES sono, rispettivamente, gli spazi vettoriali $\mathbf{A} = \mathbb{F}_{2^8}^{16}$ e $\mathbf{B} = \mathbb{F}_{2^8}^{128}$.

La relazione tra l'AES ed il BES è stabilita dall'applicazione *vettore coniugato* ϕ.

Definizione 2.1 (Vettore coniugato). Sia a un elemento di $\mathbb{F}_{2^8}$, allora abbiamo:

$$\phi : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}^8$$

che è definita da

$$\tilde{\mathbf{a}} = \phi(a) = \left(a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3}, a^{2^4}, a^{2^5}, a^{2^6}, a^{2^7}\right)$$

Questa definizione può essere estesa facilmente all'applicazione vettore coniugato ϕ da $\mathbb{F}_{2^8}^n$ ad un sottoinsieme di $\mathbb{F}_{2^8}^{8n}$.

Possiamo, quindi, usare l'applicazione ϕ per inserire un elemento dello spazio dello stato di AES $\mathbf{A}$ nello spazio dello stato del BES $\mathbf{B}$. Definiamo

$$\mathbf{B_A} = \phi(\mathbf{A}) \subset \mathbf{B}$$

per inserire l'immagine dello spazio dello stato di AES nello spazio dello stato del BES. Il sottospazio $\mathbf{B_A}$ è un insieme chiuso rispetto alla somma che preserva l'inversa; infatti, il BES è definito in maniera tale che il diagramma seguente sia commutativo.

$$\begin{array}{ccccc} & \mathbf{A} & \xrightarrow{\phi} & \mathbf{B_A} & \\ & \downarrow & & \downarrow & \\ k \to & \boxed{\text{AES}} & & \boxed{\text{BES}} & \leftarrow \phi(k) \\ & \downarrow & & \downarrow & \\ & \mathbf{A} & \xleftarrow{\phi^{-1}} & \mathbf{B_A} & \end{array}$$

2.3.1 La struttura del BES

Diamo una descrizione sintetica del cifrario BES (vedi [**MR02**] per una descrizione completa). Le operazioni base su $\mathbb{F}_{2^8}$ sono le stesse di quelle dell'AES: addizione della chiave, operazioni sulle righe e le colonne e inversione della S-box.

Il problema principale deriva dall'operazione lineare su $\mathbb{F}_2$ della S-box di AES, poiché non esiste un modo semplice per rappresentarla come un prodotto matriciale; invece nel BES c'è una semplice rappresentazione matriciale di questa operazione.

Nella descrizione di AES l'operazione lineare della S-box è definita considerando $\mathbb{F}_{2^8}$ come lo spazio vettoriale $(\mathbb{F}_2)^8$. Per descrivere questa corrispondenza è usata l'applicazione $\psi : \mathbb{F}_{2^8} \to (\mathbb{F}_2)^8$; quindi l'operazione lineare di AES, che agisce componente per componente su $\mathbb{F}_2$, $f : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$ è definita come $f(a) = \psi^{-1}(L_A(\psi(a)))$, per $a \in \mathbb{F}_{2^8}$. È la necessità di avere le applicazioni ψ e ψ^{-1} che complica l'analisi algebrica di AES.

In ogni caso, f può essere rappresentata come un polinomio linearizzato su $\mathbb{F}_{2^8}$ (vedi capitolo 3):

$$f(a) = \sum_{k=0}^{7} \lambda_k a^{2^k}, \quad \text{per } a \in \mathbb{F}_{2^8}$$

dove

$$(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6, \lambda_7) = (05, 09, f9, 25, f4, 01, b5, 8f)$$

Così abbiamo che l'operazione lineare su $\mathbb{F}_2$ della S-box di AES può essere definita nel BES usando una matrice 8×8 in $\mathbb{F}_{2^8}$. Questa matrice rimpiazza l'azione dell'applicazione lineare di AES sul primo byte dell'insieme di un vettore coniugato, mentre assicura la propietà di coniugazione vettoriale dei rimanenti bytes.

La funzione di round del BES

Sia $\mathbf{b} \in \mathbf{B}$ il vettore dato in input alla funzione di round del BES e $(\mathbf{k}_B)_i \in \mathbf{B}$ le sottochiavi, allora, seguendo lo schema delle operazioni che compie AES, possiamo rappresentare la funzione di round del BES come:

$$Round_B(\mathbf{b}, (\mathbf{k}_B)_i) = M_B \cdot ((\mathbf{b}^{(-1)}) + (\mathbf{k}_B)_i$$

dove M_B è una matrice (128×128) in $\mathbb{F}_{2^8}$ che nel BES compie la diffusione lineare. Se l'input della funzione di round dell'AES è $\mathbf{a} \in \mathbf{A}$ e la sottochiave è $(\mathbf{k}_A)_i \in \mathbf{A}$, allora abbiamo:

$$Round_A(\mathbf{a}, (\mathbf{k}_A)_i) = \phi^{-1}(Round_B(\phi(\mathbf{a}), \phi((\mathbf{k}_A)_i)))$$

Riassumendo, il BES è un cifrario a blocchi di 128 bytes costituito per intero da operazioni algebriche semplici su $\mathbb{F}_{2^8}$ che, quando ristretto al sottospazio $\mathbf{B_A} = \phi(\mathbf{A})$, fornisce una descrizione alternativa di AES. Questa relazione tra i due cifrari fornisce nuovi metodi per crittoanalizzare AES.

2.4 Crittoanalisi algebrica di AES

Una volta descritta la struttura algebrica di AES, è naturale cercare di sfruttarla per la crittoanalisi del cifrario. In [**FS01**] gli autori Ferguson, Shroppel e Whitinig, descrivono come si può esprimere la funzione di cifratura di AES con una formula algebrica compatta che contiene circa 2^{50} termini. Loro riconoscono, però, la modesta utilità di questa espressione dal punto di vista crittografico; cioè, è difficile che un attaccante possa trarre vantaggio da tale espressione algebrica per portare un attacco funzionale al cifrario.

Courtois e Pieprzyk in [**CP02**] hanno trovato un sistema molto grande di equazioni quadratiche che ha come soluzione la chiave di cifratura di AES. Essi hanno considerato il fatto che l'unica componente non lineare dell'algoritmo (la S-box) è basata sulla applicazione inversa di un campo finito, ed hanno ottenuto un piccolo insieme di equazioni quadratiche multivariate (rispetto ai bits in input ed output della S-box) che descriveva completamente la trasformazione della S-box. Combinando tutte le equazioni dentro il cifrario, hanno espresso l'intera trasformazione di cifratura come un grande, sparso e sovradefinito sistema di equazioni quadratiche multivariate su $\mathbb{F}_2$ (esattamente il sistema è costituito da 8000 equazioni quadratiche con 1600 variabili, per AES-128).

Nello stesso articolo hanno proposto un metodo, chiamato *XSL* (*eXtended Sparse Linearization*), per risolvere il sistema da loro trovato. L'XSL è basato sul metodo della *linearizzazione*. Il metodo XSL cerca di trarre vantaggio dalla particolare forma del sistema che deriva da AES (sovradefinito e sparso); quindi, questo metodo sarebbe in grado di ricavare i bits che costituiscono la chiave di cifratura.

Osserviamo che il sistema dato in [**CP02**] è in $\mathbb{F}_2$, come le variabili che rappresentano i bits che compaiono durante l'esecuzione dell'algoritmo di cifratura. Come abbiamo visto, la rappresentazione di AES tramite il BES dà luogo ad un semplice sistema di equazioni su $\mathbb{F}_{2^8}$. Si può concludere che se XSL fosse un metodo valido, questo sistema sarebbe risolto in maniera più veloce rispetto al sistema originale in $\mathbb{F}_2$.

2.4.1 Sistema quadratico multivariato del BES

Ricordiamo che la funzione di round del BES è data da

$$\mathbf{b} \longmapsto M_B \cdot \mathbf{b}^{-1} + (\mathbf{k}_B)_i$$

Denotiamo il testo in chiaro con $\mathbf{p} \in \mathbf{B}$ ed il testo cifrato con $\mathbf{c} \in \mathbf{B}$, indichiamo il vettore dello stato prima della i-esima chiamata della funzione di inversione con $\mathbf{w}_i \in \mathbf{B}$ e dopo con $\mathbf{x}_i \in \mathbf{B}$, con $0 \leq i \leq 9$. Sia, inoltre, $\mathbf{k}_i$ la i-esima sottochiave di round. Allora possiamo descrivere la cifratura del BES con il seguente sistema di

equazioni:

$$\begin{array}{ll} \mathbf{w}_0 = \mathbf{p} + \mathbf{k}_0 & \\ \mathbf{x}_i = \mathbf{w}_i^{(-1)}, & \text{per } i = 0, \ldots, 9 \\ \mathbf{w}_i = M_B \mathbf{x}_{i-1} + \mathbf{k}_i, & \text{per } i = 0, \ldots, 9 \\ \mathbf{c} = M_B^* \mathbf{x}_9 + \mathbf{k}_{10} & \end{array}$$

dove M_B^* è la matrice di diffusione modificata, perché l'ultimo round del BES (ma anche di AES) non esegue l'operazione $MixColumns$.

Denotando la matrice M_B con (α) e la matrice M_B^* con (β) e rappresentando l'$(8j + m)$-esima componente di $\mathbf{x}_i$, $\mathbf{w}_i$ e $\mathbf{k}_i$ rispettivamente con $x_{i,(j,m)}$, $w_{i,(j,m)}$ e $k_{i,(j,m)}$, possiamo considerare le equazioni sopra componente per componente e, così, ottenere una collezione di equazioni quadratiche multivariate simultanee che descrivono pienamente il procedimento di cifratura del BES.

Queste sono date per $j = 0, \ldots, 15$ ed $m = 0, \ldots, 7$ (dove $m + 1$ è interpretato modulo 8) da:

$$\begin{array}{ll} 0 = w_{0,(j,m)} + p_{(j,m)} + k_{0,(j,m)} & \\ 0 = x_{i,(j,m)} w_{i,(j,m)} + 1 & \text{per } i = 0, \ldots, 9 \\ 0 = w_{i,(j,m)} + k_{i,(j,m)} + \sum_{(j',m')} \alpha_{(j,m),(j',m')} x_{i-1,(j',m')} & \text{per } i = 1, \ldots, 9 \\ 0 = c_{(j,m)} + k_{10,(j,m)} + \sum_{(j',m')} \beta_{(j,m),(j',m')} x_{9,(j',m')} & \end{array}$$

Osserviamo che abbiamo ipotizzato che non capiti l'inversione dello zero, nè nella cifratura, nè nella generazione dell'insieme delle chiavi. Questa ipotesi è vera approssimativamente per il 53% delle cifrature e l'85% delle chiavi da 128 bits [**MR02**].

Quando consideriamo la cifratura di AES inserita nella struttura del BES, otteniamo più equazioni quadratiche multivariate perché le variabili dello stato di una cifratura di AES stanno in $\mathbf{B}_A$ e possiedono la propietà di coniugazione. Possiamo, quindi, aggiungere le seguenti equazioni al precedente sistema:

$$\begin{array}{ll} 0 = x_{i,(j,m)}^2 + x_{i,(j,m+1)} & \text{per } i = 0, \ldots, 9 \\ 0 = w_{i,(j,m)}^2 + w_{i,(j,m+1)} & \text{per } i = 0, \ldots, 9 \end{array}$$

Riscriviamo queste equazioni dividendole in lineari e quadratiche multivariate:

$$\begin{array}{ll} 0 = w_{0,(j,m)} + p_{(j,m)} + k_{0,(j,m)} & \\ 0 = w_{i,(j,m)} + k_{i,(j,m)} + \sum_{(j',m')} \alpha_{(j,m),(j'.m')} x_{i-1,(j',m')} & \text{per } i = 1, \ldots, 9 \\ 0 = c_{(j,m)} + k_{10,(j,m)} + \sum_{(j',m')} \beta_{(j,m),(j'.m')} x_{9,(j',m')} & \end{array}$$

$$\begin{array}{ll} 0 = x_{i,(j,m)} w_{i,(j,m)} + 1 & \text{per } i = 0, \ldots, 9 \\ 0 = x_{i,(j,m)}^2 + x_{i,(j,m+1)} & \text{per } i = 0, \ldots, 9 \\ 0 = w_{i,(j,m)}^2 + w_{i,(j,m+1)} & \text{per } i = 0, \ldots, 9 \end{array}$$

Una cifratura di AES può essere descritta, perciò, come un sistema sovradefinito di equazioni quadratiche con 5248 equazioni su $\mathbb{F}_{2^8}$, di cui 3840 sono (estremamente sparse) equazioni quadratiche e le restanti 1408 sono equazioni lineari. Queste equazioni che rappresentano la cifratura comprendono 7808 termini, costituiti da 2560 variabili dello stato e 1408 variabili dell'insieme delle chiavi. L'insieme delle chiavi di AES può essere espresso con un sistema simile. Il sistema dell'insieme delle chiavi ha 2560 equazioni su $\mathbb{F}_{2^8}$, di cui 960 sono quadratiche e 1600 sono equazioni lineari. Queste equazioni dell'insieme delle chiavi comprendono 3308 termini costituiti da 2048 variabili, di cui 1408 sono variabili della chiave di base e 640 sono variabili ausiliari [**MR02**].

2.5 Tecniche di attacchi potenziali

Data la formulazione algebrica del BES, è chiaro che un efficiente metodo per la soluzione di questo tipo di sistemi di equazioni quadratiche multivariate dà una crittoanalisi di AES potenzialmente con poche coppie "plaintext-ciphertext". Mentre il problema di risolvere un generico sistema di equazioni multivariate di grado più grande di uno, su un sistema finito, è conosciuto come NP-completo, non è completamente improbabile che possa essere sviluppata una tecnica che sfrutta la particolare struttura algebrica dei sistemi AES e BES per diminuire la complessità. Adesso analizziamo alcuni approcci per risolvere tali sistemi.

2.5.1 Metodi di linearizzazione

Il metodo di *linearizzazione* è una tecnica ben conosciuta per risolvere grandi sistemi di equazioni polinomiali multivariate. In questo metodo consideriamo tutti i monomi nel sistema come variabili indipendenti e proviamo a risolvere il sistema con queste nuove variabili con le tecniche di algebra lineare. Per poter applicare il metodo, il numero di equazioni linearmente indipendenti nel sistema deve essere approssimativamente lo stesso del numero dei termini del sistema. Quando questa condizione non si verifica, esistono alcune tecniche che permettono di generare dellle equazioni linearmente indipendenti, per potere, così, applicare la tecnica di linearizzazione.

In [**CKPS00**] gli autori hanno proposto un algoritmo per risolvere sistemi di equazioni quadratiche multivariate chiamato *XL* (che significa eXtended Linearization). L'algorimo XL lavora nel seguente modo: se A è un sistema di m equazioni quadratiche f_i in n variabili sul campo K, e $D \in \mathbb{N}$, esegue le seguenti operazioni:

1. **Prodotto**: genera tutti i prodotti $\prod_{j=1}^{k} x_{i_j} * f_i$ con $k \leq D-2$;
2. **Linearizzazione**: considera ogni monomio di grado $\leq D$ come una nuova variabile ed esegue l'eliminazione gaussiana sul sistema ottenuto al passo 1;

3. **Risoluzione**: supponiamo che il passo 2 generi almeno un'equazione ad una variabile; l'algoritmo risolve questa equazione;

4. **Ripetizione**: semplifica le equazioni e ripete il processo per trovare i valori delle altre variabili.

In [**CKPS00**] gli autori presentano alcune stime per la complessità dell'algoritmo XL applicato ad un sistema casuale con $m \approx n$. In particolare, loro portano una prova che XL può risolvere sistemi casuali sovradefiniti di equazioni polinomiali in tempo sub-esponenziale; ma in [**CY04**] gli autori mostrano che queste previsioni sono troppo ottimistiche. In ogni caso, è opinione diffusa che l'algoritmo XL applicato ai sistemi che derivano da AES (sia su $\mathbb{F}_{2^8}$ che su $\mathbb{F}_2$) non fornisce un attacco efficiente contro il cifrario.

Dopo l'introduzione dell'algoritmo XL nel 2000, sono state proposte molte sue varianti, una delle quali, di particolare rilevanza per AES, è stata proposta da Courtois e Pieprzyk in [**CP02**].

Questo algoritmo, chiamato XSL, è appunto basato sul metodo di XL, ma sfrutta la struttura specifica delle equazioni e il fatto che sono sparse, per costruire un attacco; invece di moltiplicare le equazioni per tutti i monomi di grado minore o uguale di $D - 2$, in XSL le equazioni sono moltiplicate soltanto per dei monomi particolari (vedi [**CP02**] per una descrizione completa).

Il sistema costruito in [**CP02**] per costruire l'attacco ha 8000 equazioni quadratiche in 1600 variabili, su $\mathbb{F}_2$ (che rappresentano i bites di input ed output). In [**CP**] sono descritti due attacchi: il primo ignora l'insieme delle chiavi (key schedule) e, quindi, ha bisogno di 11 coppie di plaintext/ciphertext (per AES-128); il secondo attacco usa tutto il key schedule, ed in teoria può essere applicato con una sola coppia di plaintext/ciphertext. In [**CP02**] è fornita la complessità del secondo tipo di attacco, ed è dell'ordine di 2^{230} per AES-128.

XSL attacco al BES

Come abbiamo già osservato, il sistema su $\mathbb{F}_{2^8}$ di AES ottenuto dal BES è più semplice del sistema su $\mathbb{F}_2$ descritto in [**CP02**]. In particolare è più sparso. Questo ci fa pensare che l'attacco XSL è più indicato per il sistema del BES rispetto a quello originale di AES.

Murphy e Robshaw hanno analizzato in [**MR03**] le conseguenze di un attacco XSL contro il sistema derivato dal BES concludendo che XSL potrebbe essere un metodo valido per attaccare con successo AES-128.

Capitolo 3

Rappresentazione polinomiale dell'AES

In questo capitolo descriviamo AES in forma polinomiale [**Ros03**], in maniera più dettagliata rispetto al capitolo iniziale; infatti ogni operazione compiuta nei round dell'algoritmo viene fatta su un anello ed è, quindi, possibile rappresentare ogni funzione applicata nei round con un polinomo, o come composizione di più polinomi su un anello. Per esempio vedremo che l'S-box è rappresentata da un polinomio con coefficienti in $GF(2^8)$ (campo di Galois).

3.1 Descrizione dell'algoritmo del Rijndael nell'anello R

Sia $\mathbb{Z}_2 = \{0, 1\}$ il campo binario e consideriamo il polinomio irriducibile

$$\mu(z) := z^8 + z^4 + z^3 + z + 1 \in \mathbb{Z}_2[z].$$

Sia $\mathbb{F} = \mathbb{Z}_2[z]/\langle \mu(z) \rangle = GF(2^8)$ il campo di Galois di 2^8 elementi e consideriamo l'ideale:

$$I := \langle x^4 + 1, y^4 + 1, \mu(z) \rangle \subset \mathbb{Z}_2[x, y, z].$$

Descriveremo l'algoritmo del Rijndael attraverso una serie di manipolazioni polinomiali dentro l'anello finito

$$R := \mathbb{Z}_2[x, y, z]/I = \mathbb{F}[x, y]/\langle x^4 + 1, y^4 + 1 \rangle. \tag{3.1}$$

Da questa descrizione abbiamo che l'anello R ha contemporaneamente sia la struttura di una $\mathbb{Z}_2$-algebra finita che la struttura di una $\mathbb{F}$-algebra finita. I monomi

$$\{x^i y^j z^k : 0 \leq i, j \leq 3, 0 \leq k \leq 7\}$$

formano un $\mathbb{Z}_2$-base dell'anello (algebra) R. In particolare abbiamo $dim_{\mathbb{Z}_2} R = 128$, cioè $|R| = 2^{128}$. Nell'anello R i calcoli sono eseguiti in modo efficiente. La somma in R è fatta componente per componente e il prodotto in R è attuato attraverso il prodotto in $\mathbb{Z}_2[x, y, z]$ seguita dalla riduzione modulo l'ideale I.

Osservazione 3.1. Si può verificare che $\{x^4 + 1, y^4 + 1, \mu(z)\}$ forma una base di Gröbner ridotta dell'ideale I che è anche un ideale zero dimensionale (vedi capitolo 4). Una conseguenza è che la riduzione modulo l'ideale I è semplice.

Sia $r \in R$ un elemento e siano $r_{i,j} \in \mathbb{F}$ e $r_j \in \mathbb{F}[x]/\langle x^4 + 1\rangle$, possiamo definire r come segue:

$$r = \sum_{i=0}^{3}\sum_{j=0}^{3} r_{i,j}x^i y^j = \sum_{j=0}^{3}\left(\sum_{i=0}^{3} r_{i,j}x^i\right) y^j = \sum_{j=0}^{3} r_j y^j. \tag{3.2}$$

A livello teorico, un *crittosistema a chiave segreta* è costituito dallo spazio dei messaggi in chiaro (plain-text) M, dallo spazio dei messaggi cifrati (cipher-text) C, dallo spazio delle chiavi K con una applicazione di cifratura:

$$\epsilon : M \times K \to C$$

ed una applicazione di decifratura:

$$\delta : C \times K \to M$$

tali che $\delta(\epsilon(m, k), k) = m$, dove $m \in M$ e $k \in K$. Potrebbe essere computazionalmente impossibile calcolare la chiave $k \in K$ partendo da una seguenza di coppie di plain-text/cipher-text $(m^{(t)}, c^t = \epsilon(m^t, k))$, con $t = 1, 2, \ldots$ (*known plaintext attack*: l'attaccante possiede una stringa di plain-tex e la corrispondente stringa di cipher-tex).

Nel crittosistema AES abbiamo la possibilità di lavorare con chiavi segrete rispettivamente di 128 bits, 192 bits o 256 bits. Descriveremo il crittosistema quando $|K| = 2^{128}$. Per l'algoritmo del Rijndael definiamo

$$K = M = C = R.$$

Sarà fondamentale, per la nostra descrizione, il seguente polinomio che, vedremo, rappresenta la S-box di AES:

$$\begin{aligned}\varphi(u) :=&(z^2+1)u^{254} + (z^3+1)u^{253} + (z^7+z^6+z^5+z^4+z^3+1)u^{251}\\ &+ (z^5+z^2+1)u^{247} + (z^7+z^6+z^5+z^4+z^2)u^{239} + u^{223}\\ &+ (z^7+z^5+z^4+z^2+1)u^{191} + (z^7+z^3+z^2+z+1)u^{127}\\ &+ (z^6+z^5+z+1) \in \mathbb{F}[u].\end{aligned} \tag{3.3}$$

Supponiamo che Alice e Bob condividano una chiave segreta $k \in R$ ed Alice voglia cifrare il messaggio $m \in R$. All'inizio sia Alice che Bob devono eseguire un'espansione della chiave (key-expansion) che darà 11 elementi $k^t \in R$, con $t = 0, \ldots, 10$.

3.1.1 Espansione della chiave

Usando le notazioni dell'equazione 3.2, sia Alice che Bob calcolano ricorsivamente 11 elementi $k^t \in R$, $t = 0, \ldots, 10$, nel seguente modo:

$$\begin{aligned} &k^{(0)} := k; \\ &k_0^{(t+1)} := \left(\sum_{i=0}^{3} \varphi(k_{i,3}^{(t)}) x^i \right) x^3 + z^t + k_0^{(t)} \text{ per } t = 0, \ldots, 9; \\ &k_i^{(t+1)} := k_{i-1}^{(t+1)} + k_i^{(t)} \text{ per } i = 1, 2, 3;\ t = 0, \ldots, 9. \end{aligned}$$

3.1.2 Algoritmo di cifratura del Rijndael

Per descrivere l'algoritmo di cifratura dobbiamo definire l'elemento dell'anello:

$$\gamma := (z+1)x^3 + x^2 + x + z \in R$$

che rappresenta l'operazione di $mixColumn$ in AES. Usando la chiave del round $k^{(t)} \in R$ e partendo con il messaggio in chiaro $m \in R$, Alice calcola ricorsivamente:

$$\begin{aligned} &m^{(0)} := m + k^{(0)}; \\ &m^{(t+1)} := \gamma \sum_{i=0}^{3} \sum_{j=0}^{3} \varphi(m_{i,j}^{(t)}) x^i y^{3i+j} + k^{(t+1)} \text{ per } t = 0, \ldots, 8; \\ &c := m^{(10)} := \sum_{i=0}^{3} \sum_{j=0}^{3} \varphi(m_{i,j}^{(9)}) x^i y^{3i+j} + k^{(10)}. \end{aligned}$$

Il messaggio cifrato che Alice trasmetterà è denotato con c. Osserviamo che nel decimo round non si hanno prodotti per γ. Questo assicura che le operazioni di cifratura e di decifratura possono essere descritte formalmente dalle stesse operazioni algebriche.

3.1.3 Algoritmo di decifratura del Rijndael

Il polinomio φ introdotto nell'equazione 3.3 è una permutazione polinomiale che descrive una permutazione di elementi del campo $\mathbb{F}$ (vedremo in dettaglio la costruzione di questo polinomio nella sezione 3.4).
Prima di continuare diamo una definizione di cui abbiamo bisogno.

Definizione 3.1 (Funzioni su $GF(2^n)$). Sia f una funzione definita come segue

$$\begin{aligned} f : GF(2^n) &\to GF(2^n) \\ a &\longmapsto b = f(a) \end{aligned}$$

Tutte le funzioni su $GF(2^n)$ possono essere espresse come un polinomio su $GF(2^n)$ di grado al più $2^n - 1$:

$$f(a) = \sum_{i=0}^{2^n-1} c_i a^i. \tag{3.4}$$

Esiste un'unica permutazione polinomiale $\psi(u) \in \mathbb{F}[u]$ di grado al più 255 ($2^8 - 1 = 256 - 1$) tale che $\varphi \circ \psi = \psi \circ \varphi = id_{\mathbb{F}}$, e descriveremo come ottenere questo polinomio nella sezione 3.4. L'elemento $\gamma \in R$ è invertibile con

$$\gamma^{-1} = (z^3 + z + 1)x^3 + (z^3 + z^2 + 1)x^2 + (z^3 + 1)x + (z^3 + z^2 + z) \in R.$$

Usando l'applicazione ψ, l'elemento γ^{-1} e la chiave del round $k^{(t)}$, Bob può decifrare il messaggio m, che gli ha mandato Alice, attraverso i seguenti passaggi:

$$\begin{aligned}
c^{(0)} &:= c + k^{(10)} \\
c^{(t+1)} &:= \gamma^{-1} \sum_{i=0}^{3} \sum_{j=0}^{3} \psi(c_{i,j}^{(t)}) x^i y^{i+j} + \gamma^{-1} k^{(9-t)} \text{ per } t = 0, \ldots, 8 \\
c^{(10)} &:= \sum_{i=0}^{3} \sum_{j=0}^{3} \psi(c_{i,j}^{(9)}) x^i y^{i+j} + k^{(0)}
\end{aligned}$$

Si verifica facilmente che $m = c^{(10)}$. Notiamo che formalmente sia lo schema di cifratura che quello di decifratura seguono la stessa seguenza di trasformazioni. Il polinomio φ è semplicemente rimpiazzato da ψ, il prodotto per γ è sostituito dal prodotto per γ^{-1} e l'insieme delle chiavi è cambiato sostituendo $k^{(t)}$, per $t = 0, \ldots, 10$, con $k^{(10)}, \gamma^{-1}k^{(9)}, \ldots, \gamma^{-1}k^{(1)}, k^{(0)}$.

Osservazione 3.2. Sia la cifratura che la decifratura possono essere implementati efficientemente. In pratica i polinomi φ e ψ non sono calcolati esplicitamente, ma sono usate delle tabelle che rappresentano le permutazioni $\varphi, \psi : \mathbb{F} \to \mathbb{F}$ (vedi [**DR**, p.213]). La sostituzione degli esponenti $x^i y^j \to x^i y^{3i+j}$ non richiede nessuna operazione aritmetica e la somma della chiave di round k^{t+1} è eseguita efficientemente con le operazioni booleane di XOR. Invece potrebbero essere richieste operazioni matematiche quando si moltiplica un elemento per γ o per γ^{-1}. Se si usano operazioni algebriche, allora, in generale, la moltiplicazione per γ è leggermente più semplice di quella per γ^{-1} e quindi l'algoritmo di cifratura è più complesso di quello di decifratura.

3.2 Relazione con la descrizione standard

Nella descrizione originale dell'algoritmo del Rijndael l'anello R non era usato. Gli insiemi di elementi aventi 128 bits erano descritti da matrici 4×4, ognuna delle

quali aveva elementi di dimensione di un byte (cioè 8 bits). La descrizione assegna ad ogni elemento $r = \sum_{i=0}^{3}\sum_{j=0}^{3} r_{i,j}x^i y^j$ la matrice 4×4

$r_{0,0}$	$r_{0,1}$	$r_{0,2}$	$r_{0,3}$
$r_{1,0}$	$r_{1,1}$	$r_{1,2}$	$r_{1,3}$
$r_{2,0}$	$r_{2,1}$	$r_{2,2}$	$r_{2,3}$
$r_{3,0}$	$r_{3,1}$	$r_{3,2}$	$r_{3,3}$

dove ogni elemento $r_{i,j} \in \mathbb{F}$ rappresenta un byte. Usando uno schema specifico, l'algoritmo del Rijndael esegue le seguenti operazioni: $SubBytes$, $ShiftRows$, $MixColumns$, $AddRoundKey$. Lo schema con cui vengono eseguite le operazioni è il seguente: nel round zero si aggiunge al testo in chiaro la chiave $k^{(0)}$, quindi si esegue solo $AddRoundKey$. Dal primo al nono round si eseguono le operazioni $SubBytes$, $ShiftRows$, $MixColumns$ e $AddRoundKey$. Nel decimo round si eseguono solamente $SubBytes$, $ShiftRows$ e $AddRoundKey$. Vediamo adesso la descrizione algebrica di queste operazioni.

3.2.1 Trasformazione SubByte

In questa operazione ogni elemento $r_{i,j} \in \mathbb{F}$ è trasformato usando una permutazione φ del gruppo simmetrico di 256 elementi. La permutazione φ può essere decomposta in tre permutazioni:

$$\varphi_1 : \mathbb{F} \to \mathbb{F}, \qquad f \longmapsto \begin{cases} f^{-1} & \text{se } f \neq 0, \\ 0 & \text{se } f = 0. \end{cases} \tag{3.5}$$

$$L : \mathbb{F} \to \mathbb{F}, \qquad f \longmapsto (z^4 + z^3 + z^2 + z + 1)f \mod z^8 + 1. \tag{3.6}$$

$$\varphi_3 : \mathbb{F} \to \mathbb{F}, \qquad f \longmapsto (z^6 + z^5 + z + 1 + f). \tag{3.7}$$

La permutazione φ è definita come $\varphi := \varphi_3 \circ L \circ \varphi_1$. È possibile descrivere la permutazione φ usando un polinomio di permutazione. Per questo osserviamo che ogni permutazione di $\mathbb{F}$ può anche essere descritta tramite un'unica interpolazione polinomiale (un elemento di $\mathbb{F}[u]$) avente grado al massimo 255 (vedi [**LN**, p.28] formula di interpolazione di Lagrange). Indicheremo questo polinomio unico con $\varphi(u)$. Dal contesto sarà chiaro se interpretare φ come una permutazione o come un polinomio $\varphi(u) \in \mathbb{F}[u]$.

Questo unico polinomio di permutazione $\varphi(u)$ può essere calcolato nel seguente modo. Se $\alpha \neq 0$ allora

$$T_\alpha(u) := u \sum_{i=0}^{254} \alpha^i u^{254-i}$$

è l'unico polinomio interpolatore di Lagrange avente la propietà che

$$T_\alpha(\beta) = \begin{cases} 1 & \text{se } \alpha = \beta, \\ 0 & \text{altrimenti.} \end{cases}$$

Se $\alpha = 0$ allora $T_\alpha(u) = u^{255} + 1$ è l'unico polinomio interpolatore di Lagrange. Il polinomio $\varphi(u) \in \mathbb{F}[u]$ si può calcolare semplicemente usando l'espressione $\varphi_(u) = \sum_{\alpha \in \mathbb{F}} \varphi(\alpha) T_\alpha(u)$.

3.2.2 La trasformazione ShiftRows

In questa operazione i byte dell'i-esima riga sono spostati ciclicamente di i posizioni. Algebricamente questa operazione ha una semplice interpretazione. Consideriamo gli elementi $r = r(x, y) \in R$ dell'equazione 3.2: quindi $ShiftRows$ corrisponde alla trasformazione:

$$r = r(x, y) \longmapsto r(xy^3, y).$$

Questo si traduce nell'algoritmo di cifratura nel rimpiazzare il monomio $x^i y^i$ con il monomio $x^i y^{3i+j}$. La trasformazione inversa di $ShiftRows$, $InvShiftRows$, è $r(x, y) \longmapsto r(xy, y)$, che si traduce nel rimpiazzare il monomio $x^i y^i$ con $x^i y^{i+j}$.

3.2.3 Le trasformazioni MixColumns ed AddRoundKey

Nella trasformazione $MixColumns$ ogni colonna $r_j = \sum_{i=0}^{3} r_{i,j} x^i$ viene moltiplicata per l'elemento $\gamma = (z+1)x^3 + x^2 + x + z \in R$, che in rappresentazione esadecimale equivale a $03x^3 + 01x^2 + 01x + 02$.

La trasformazione $AddRoundKey$ aggiunge al passo t-esimo la chiave $k^{(t)}$.

3.2.4 AES-192 ed AES-256

Fino ad ora abbiamo descritto Rijndael con la misura delle chiavi e la misura del messaggio di 128 bits. Ci riferiremo a questo crittosistema indicandolo con AES-128. Nella descrizione originale [**Rijndael**] è possibile variare la dimensione sia della chiave che dei blocchi del messaggio. Nello standard adottato dal FIPS (federal information processing standards) in [**Fips**] la misura dei blocchi del messaggio è sempre di 128 bits, quindi non è variabile. In AES-192 ed AES-256 la misura della chiave segreta è rispettivamente di 192 e 256 bits. In AES-192 sono calcolati 13 elementi $k^{(t)} \in R$, $t = 0, \ldots, 12$, a partire dai 192 bits originali, e l'algoritmo del Rijndael esegue 12 round. In AES-256 invece sono calcolati 15 elementi $k^{(t)}$, con $t = 0, \ldots, 14$, a partire dai 256 bits originali e l'algoritmo è formato da 15 round. Queste sono le uniche differenze tra i tre standard.

3.3 Traccia e Basi duali

In questa sezione diamo gli strumenti necessari per poter descrivere nella sezione 3.4 la struttura della S-box di AES-128 e adottiamo la convenzione di riferirci ad una estensione finita $F = \mathbb{F}_{p^n}$ di un campo finito $K = \mathbb{F}_p$ come ad uno spazio vettoriale su K. Allora F ha dimensione n su K e, se $\{\alpha_1, \ldots, \alpha_n\}$ è una base di F su K, ogni elemento $\alpha \in F$ può essere rappresentato unicamente nella forma:

$$\alpha = c_1\alpha_1 + \cdots + c_n\alpha_n, \text{ con } c_j \in K \text{ per } 1 \leq j \leq n.$$

Cominciamo col dare la definizione di *polinomio linearizzato.*

Definizione 3.2 (Polinomio linearizzato). [**LN**, p.101] Un polinomio della forma

$$\mathcal{L}(x) = \sum_{i=0}^{n} \alpha_i x^{p^i},$$

con coefficienti nel campo $\mathbb{F}_{p^n}$, con p primo ed $n \in \mathbb{N}$, è chiamato *polinomio linearizzato* o p-polinomio sul campo $\mathbb{F}_{p^n}$

Definizione 3.3 (Traccia). Sia $x \in \mathbb{F}_{p^n} = F$, con p primo e $K = \mathbb{F}_p$. La *traccia* di x su $\mathbb{F}_p$ è definita come segue:

$$Tr_{F/K}(x) := x + x^p + x^{p^2} \cdots + x^{p^{n-1}}$$

Vedremo che:

$$\begin{aligned} Tr_{F/K} : \mathbb{F}_{p^n} &\longrightarrow \mathbb{F}_p; \\ x &\longmapsto x + x^p + \cdots + x^{p^{n-1}} \end{aligned} \tag{3.8}$$

Per semplicità indichiamo la traccia $Tr_{F/K}$ con Tr. Diamo adesso alcune propietà della traccia [**LN**, p.52].

Teorema 3.1. *Sia la traccia definita come nella definizione 3.3, allora gode delle seguenti propietà:*

(i) $Tr(x + y) = Tr(x) + Tr(y)$, *per ogni* $x, y \in \mathbb{F}_{p^n}$*;*

(ii) $Tr(cx) = cTr(x)$, *per ogni* $c \in \mathbb{F}_p, x \in \mathbb{F}_{p^n}$*;*

(iii) Tr *è una trasformazione lineare da* $\mathbb{F}_{p^n}$ *su* $\mathbb{F}_p$*;*

(iv) $Tr(a) = na$ *per ogni* $a \in \mathbb{F}_p$*;*

(v) $Tr(x^p) = Tr(x)$ *per ogni* $x \in \mathbb{F}_{p^n}$*.*

Prima di dimostrare il teorema abbiamo bisogno di dare alcuni risultati necessari per la dimostrzione.

Teorema 3.2. [**LN**, p.16] *Sia R un anello commutativo di caratteristica p, con p primo. Allora*

$$(a+b)^{p^n} = a^{p^n} + b^{p^n} \text{ e } (a-b)^{p^n} = a^{p^n} - b^{p^n}$$

per $a, b \in R$ ed $n \in \mathbb{N}$.

Dimostrazione. Usiamo

$$\binom{p}{i} = \frac{p(p-1)\cdots(p-i+1)}{1 \cdot 2 \cdot \cdots \cdot i} \equiv 0 \pmod{p},$$

per ogni $i \in \mathbb{Z}$ con $0 < i < p$. Questo segue dal fatto che $\binom{p}{i}$ è un intero e dall'osservazione che il fattore p al numeratore non può essere cancellato; quindi dalla propietà del coefficiente binomiale segue che

$$(a+b)^p = a^p + \binom{p}{1}a^{p-1}b + \cdots + \binom{p}{p-1}ab^{p-1} + b^p = a^p + b^p,$$

e per induzione su n dimostriamo la prima uguaglianza. Da ciò che abbiamo appena visto si ricava che:

$$a^{p^n} = ((a-b)+b)^{p^n} = (a-b)^{p^n} + b^{p^n},$$

e così abbiamo dimostrato anche la seconda identità. □

Lemma 3.3. [**LN**, p.45] *Se F è un campo finito con p elementi, allora ogni $a \in F$ soddisfa $a^p = a$.*

Dimostrazione. L'identità $a^p = a$ è banale per $a = 0$. Gli elementi diversi da zero di F formano un gruppo di ordine $p-1$ sotto il prodotto; quindi $a^{p-1} = 1$ per ogni $a \in F$ con $a \neq 0$, e moltiplicando per a segue $a^p = a$. □

Possiamo adesso dimostrare le propietà della traccia del teorema 3.1.

Dimostrazione. Teorema 3.1.

(i) Per $x, y \in \mathbb{F}_{p^n}$ usiamo il teorema 3.2, allora abbiamo:

$$\begin{aligned} Tr(x+y) &= x + y + (x+y)^p + \cdots + (x+y)^{p^{n-1}} \\ &= x + y + x^p + y^p + \cdots + x^{p^{n-1}} + y^{p^{n-1}} \\ &= Tr(x) + Tr(y). \end{aligned}$$

(ii) Se $c \in \mathbb{F}_p$, dal lemma 3.3, abbiamo che c^{p^j}, per $j \geq 0$. Quindi per $x \in \mathbb{F}_{p^n}$ abbiamo:

$$\begin{aligned} Tr(cx) &= cx + c^p x^p + \cdots + c^{p^{n-1}} x^{p^{n-1}} \\ &= cx + cx^p + \cdots + cx^{p^{n-1}} \\ &= cTr(x). \end{aligned}$$

(iii) Le propietà (i) ed (ii), insieme con il fatto che $Tr(x) \in \mathbb{F}_p$ per ogni $x \in \mathbb{F}_{p^n}$, ci dicono che Tr è una trasformazione lineare da $\mathbb{F}_p$ in $\mathbb{F}_{p^n}$. Per dimostrare che questa applicazione è su $\mathbb{F}_{p^n}$ è sufficiente dimostrare l'esistenza di un elemento $x \in \mathbb{F}_{p^n}$ con $Tr(x) \neq 0$. Abbiamo che $Tr(x) = 0$ se e soltanto se x è una radice del polinomio $x^{p^{n-1}} + \cdots + x^p + x \in \mathbb{F}_p[x]$ in $\mathbb{F}_{p^n}$; ma poiché questo polinomio non può avere più di p^{n-1} radici in $\mathbb{F}_{p^n}$ e $\mathbb{F}_{p^n}$ ha p^n elementi, il risultato è dimostrato.

(iv) Segue immediatamente dalla definizione di traccia e dal lemma 3.3.

(v) Per $x \in \mathbb{F}_{p^n}$ dal lemma 3.3 abbiamo $x^{p^n} = x$ e, quindi:

$$Tr(x^p) = x^p + x^{p^2} + \cdots + x^{p^n} = Tr(x).$$

□

La funzione traccia non è essa stessa una trasformazione lineare da F su K, ma serve per una descrizione di tutte le trasformazioni lineari da F su K che hanno il vantaggio di essere indipendenti dalla base scelta.

Teorema 3.4. *Sia F un'estensione finita del campo finito K, entrambi considerati come spazi vettoriali su K, allora le trasformazioni lineari da F in K sono esattamente le mappe L_β, $\beta \in F$, dove $L_\beta(\alpha) = Tr_{F/K}(\beta\alpha)$, per ogni $\alpha \in F$. Inoltre, abbiamo che $L_\beta \neq L_\gamma$, per γ e β elementi distinti di F.*

Data la base $\{\alpha_1, \ldots, \alpha_n\}$ vogliamo calcolare i coefficienti $c_j(\alpha) \in K$, con $1 \leq j \leq n$, nella rappresentazione unica

$$\alpha = c_1(\alpha)\alpha_1 + \cdots + c_n(\alpha)\alpha_n \tag{3.9}$$

di un elemento $\alpha \in F$. Osserviamo che $c_j : \alpha \longmapsto c_j(\alpha)$ è una trasformazione lineare da F in K e, così, per il teorema 3.4, devono esistere $\beta_j \in F$ tali che

$c_j(\alpha) = Tr_{F/K}(\beta_j\alpha)$ per ogni $\alpha \in F$. Ponendo $\alpha = \alpha_i$, con $1 \leq i \leq n$, vediamo che $Tr_{F/K}(\beta_j\alpha_i) = 0$ per $i \neq j$ e $Tr_{F/K}(\beta_j\alpha_i) = 1$ per $i = j$. Inoltre, $\{\beta_1, \ldots, \beta_n\}$ è ancora una base di F su K, perché se

$$d_1\beta_1 + \cdots + d_n\beta_n = 0, \text{ con } d_i \in K \text{ per } 1 \leq i \leq n,$$

allora, moltiplicando per un α_i fissato ed applicando la funzione $Tr_{F/K}$ si dimostra che $d_i = 0$.

Definizione 3.4 (Base duale). Sia K un campo finito ed F un'estensione finita di K, allora due basi $\{\alpha_1, \ldots, \alpha_n\}$ ed $\{\beta_1, \ldots, \beta_n\}$ di F su K sono dette *basi duali* se per $1 \leq i, j \leq n$ abbiamo:

$$Tr(\alpha_i\beta_j) = \begin{cases} 0 & \text{per } i \neq j \\ 1 & \text{per i=j.} \end{cases}$$

Abbiamo visto sopra che data una base $\{\alpha_1, \ldots, \alpha_n\}$ di F su K deve esistere la corrispondente base duale $\beta_1, \ldots, \beta_n$. La base duale è, infatti, unicamente determinata poiché la sua definizione implica che i coefficienti $c_j(\alpha)$ dell'equazione 3.9 sono dati da $c_j(\alpha) = Tr_{F/K}(\beta_j\alpha)$ per ogni $\alpha \in F$, e per il teorema 3.4 l'elemento β_j è unicamente determinato dalla trasformazione lineare c_j.

Il numero di basi distinte di F su K è piuttosto grande, ma ci sono due tipi speciali di basi di particolare importanza. Le prime sono le *basi polinomiali* $\{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$, costituite dalle potenze di un dato elemento α di F su K. Gli elementi α sono spesso considerati come elementi primitivi di F. Un altro tipo di base è la *base normale*, definta come segue.

Definizione 3.5 (Base normale). Sia $K = \mathbb{F}_p$ e $F = \mathbb{F}_{p^n}$, allora una base di F su K della forma $\{\alpha, \alpha^p, \ldots, \alpha^{p^{n-1}}\}$, costituita da un appropiato elemento $\alpha \in F$ e dai suoi elementi coniugati rispetto a K, è detta *base normale* di F su K.

Gli elementi coniugati sono definiti nel seguente modo.

Definizione 3.6 (Elementi coniugati). Sia $\mathbb{F}_{p^n}$ un'estensione di $\mathbb{F}_p$ e sia $\alpha \in \mathbb{F}_{p^n}$, allora gli elementi $\alpha, \alpha^p, \ldots, \alpha^{p^{n-1}}$ sono chiamati *coniugati* di α rispetto a $\mathbb{F}_p$.

3.4 La struttura della S-box

Tutte le trasformazioni sono lineari in $\mathbb{Z}_2$, eccetto per la trasformazione della S-box. È quindi molto importante descrivere bene la S-box. La permutazione φ è la composizione delle applicazioni φ_1, L, φ_3. Descriviamo, adesso, le permutazioni polinomiali di ognuna di esse. La permutazione polinomiale dell'applicazione φ_1 è data da $\varphi_1(u) = u^{254}$ (osserviamo che $254 \equiv -1 \pmod{2^8 - 1}$, quindi u^{254}

rappresenta la funzione inversa di u). La permutazione L è una applicazione lineare in $\mathbb{Z}_2$; quindi esiste un unico polinomio linearizzato $\mathcal{L}(u) = \sum_{i=0}^{7} \lambda_i u^{2^i} \in \mathbb{F}[u]$ tale che

$$\mathcal{L}(f) = L(f)$$

per ogni $f \in \mathbb{F}$. Se $\alpha_1, \ldots, \alpha_s$ è una base qualsiasi di $\mathbb{F}$ sul campo $\mathbb{Z}_2$, allora è possibile calcolare i coefficienti $\lambda_0, \lambda_1, \ldots, \lambda_7$ attraverso le equazioni

$$\mathcal{L}(\alpha_j) = \sum_{i=0}^{7} \lambda_i \alpha_j^{2^i} = \mathcal{L}(\alpha_j),\ j = 1, \ldots, 8.$$

Questo sistema di equazioni lineari può essere risolto esplicitamente. A questo scopo sia $\beta_1, \ldots, \beta_8$ la base duale di $\alpha_1, \ldots, \alpha_8$ rappresentata attraverso la relazione

$$Tr_{\mathbb{F}/\mathbb{Z}_2}(\alpha_i \beta_j) = \begin{cases} 0 & \text{per } i \neq j \\ 1 & \text{per i=j.} \end{cases}$$

Introduciamo le matrici:

$$A := \begin{pmatrix} \alpha_1 & \alpha_1^2 & \alpha_1^4 & \ldots & \alpha_1^{2^7} \\ \alpha_2 & \alpha_2^2 & \alpha_2^4 & \ldots & \alpha_2^{2^7} \\ \vdots & \vdots & & & \vdots \\ \alpha_8 & \alpha_8^2 & \alpha_8^4 & \ldots & \alpha_8^{2^7} \end{pmatrix} \quad B := \begin{pmatrix} \beta_1 & \beta_2 & \ldots & \beta_8 \\ \beta_1^2 & \beta_2^2 & \ldots & \beta_8^2 \\ \beta_1^4 & \beta_2^4 & \ldots & \beta_8^4 \\ \vdots & \vdots & & \vdots \\ \beta_1^{2^7} & \beta_2^{2^7} & \ldots & \beta_8^{2^7} \end{pmatrix}$$

Supporre che $\{\beta_1, \ldots, \beta_8\}$ sia una base duale di $\{\alpha_1, \ldots, \alpha_8\}$ significa semplicemente che $AB = I_8$. Sia S il cambio di base della trasformazione tale che:

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_8 \end{pmatrix} = S \begin{pmatrix} 1 \\ z \\ \vdots \\ z^7 \end{pmatrix}$$

e consideriamo la matrice

$$L := \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

che descrive l'applicazione lineare introdotta nell'equazione 3.6 rispetto alla base polinomiale $1, z, z^2, z^3, \ldots, z^7$.

Lemma 3.5. *I coefficienti $\lambda_0, \lambda_1, \ldots, \lambda_7$ della permutazione polinomiale $\mathcal{L}(u)$ sono dati come:*

$$\begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_7 \end{pmatrix} = BSL^tS^{-1} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_8 \end{pmatrix} \tag{3.10}$$

Dimostrazione. SL^tS^{-1} descrive il cambio di base dell'applicazione lineare L rispetto alla base $\alpha_1, \ldots, \alpha_8$. □

Al fine di calcolare esplicitamente i coefficienti $\lambda_0, \lambda_1, \ldots, \lambda_7$ possiamo lavorare con la base polinomiale $1, z, z^2, \ldots, z^7$ (in questo caso $S = I_8$). In alternativa possiamo lavorare con un base normale. Vediamo i calcoli per una base normale. Sia

$$\alpha := z^5 + 1 \in \mathbb{F}.$$

Si può verificare (usando un programma di computer algebra) che α è un elemento primitivo di $\mathbb{F}$ e che $\{\alpha_i := \alpha^{2^{i-1}} \mid i = 1, \ldots, 8\}$ formano una base normale. Ricordiamo che un elemento *primitivo* di $\mathbb{F}_p$ è un generatore del gruppo ciclico $\mathbb{F}_p^*$. La base duale di $\{\alpha_1, \ldots, \alpha_8\}$ è calcolata velocemente usando Maple come $\{\beta_j := \beta^{2^{j-1}} \mid j = 1, \ldots, 8\}$, dove $\beta = z^5 + z^4 + z^2 + 1$. È noto che la base duale di una base normale è normale anch'essa. Il cambio di base della trasformazione è calcolato in questo caso come:

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

Alla luce di ciò si calcola velocemente:

$$\begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_7 \end{pmatrix} = BSL^tS^{-1} \begin{pmatrix} \alpha \\ \alpha^2 \\ \vdots \\ \alpha^{2^7} \end{pmatrix} = \begin{pmatrix} z^2 + 1 \\ z^3 + 1 \\ z^7 + z^6 + z^5 + z^4 + z^3 + 1 \\ z^5 + z^2 + 1 \\ z^7 + z^6 + z^5 + z^4 + z^2 \\ 1 \\ z^7 + z^5 + z^4 + z^2 + 1 \\ z^7 + z^3 + z^2 + z + 1 \end{pmatrix} \tag{3.11}$$

Gli elementi λ_i trovati con questi calcoli coincidono con i coefficienti non costanti di φ introdotti nell'equazione 3.3. Per dare la forma esatta abbiamo bisogno di una descrizione polinomiale della permutazione φ_3 introdotta nella 3.7. Chiaramente il polinomio lineare $\varphi_3(u) := u + 1 + z + z^5 + z^6 \in \mathbb{F}[u]$ interpola l'applicazione affine φ_3. Concatenando le tre applicazioni polinomiali abbiamo:

$$\varphi(u) = \varphi_3 \circ \mathcal{L} \circ \varphi_1(u) = 1 + z + z^5 + z^6 + \mathcal{L}(u^{254}) \mod u^{256} + u.$$

Osserviamo che $\mathcal{L}$ ha al più otto coefficienti diversi da zero. Riducendo $\mathcal{L}(u^{254})$ con la relazione $u^{256} = u$ non cambierà e questo spiega la "sparsity" del polinomio $\varphi(u)$. Il fatto che il polinomio della permutazione sia sparso non implica che anche il polinomio inverso $\psi(u)$ lo sia. Osserviamo che:

$$\psi(u) = \varphi_1^{-1} \circ \mathcal{L}^{-1} \circ \varphi_3^{-1}(u) \mod u^{256} + u.$$

Come prima i coefficienti del polinomio $\mathcal{L}^{-1}$ sono calcolati da:

$$BS(L^{-1})^t S^{-1} \begin{pmatrix} \alpha \\ \alpha^2 \\ \vdots \\ \alpha^{2^7} \end{pmatrix} \tag{3.12}$$

Usando Maple (vedi [**Ros03**, p.9]) troviamo:

$$\begin{aligned} \mathcal{L}^{-1}(u) =& (z^6 + z^5 + z^3 + z^2 + z)u^{128} + (z^7 + z^6 + z^4 + z^3 + z + 1)u^{64} \\ &+ (z^6 + z^4 + z^3 + 1)u^{32} + (z^6 + z^5 + z^4 + z^3)u^{16} \\ &+ (z^6 + z^4 + z^3 + z)u^8 + (z^6 + z^5 + z^4 + z^3 + z^2 + z + 1)u^4 \\ &+ (z^7 + z^6 + z^5 + z^4 + z^3 + z^2 + z)u^2 + (z^2 + 1)u \in \mathbb{F}[u]. \end{aligned} \tag{3.13}$$

Combinando questo risultato con l'applicazione φ_3^{-1} abbiamo:

$$\rho(u) := \mathcal{L}^{-1}\varphi_3^{-1}(u) = \mathcal{L}^{-1}(u + \varphi_3(0)) = \mathcal{L}^{-1}(u) + \mathcal{L}^{-1}(\varphi_3(0)) = \mathcal{L}^{-1}(u) + z^2 + 1. \tag{3.14}$$

Un polinomio della forma di $\rho(u)$ è chiamato anche polinomio affine [**LN**], riflettendo il fatto che l'applicazione $\mathcal{L}^{-1}\varphi_3^{-1}$ è lineare affine su $\mathbb{Z}_2$. Componendo $\rho(u)$ con il polinomio $\varphi_1^{-1}(u) = \varphi_1(u) = u^{254}$ si ha un polinomio non sparso $\psi(u) = \rho(u)^{254}$ mod $u^{256} + u$. In [**Ros03**] è presente l'espressione completa di questo polinomio calcolata con Maple.

Capitolo 4

Basi di Gröbner

Abbiamo visto nel capitolo 2 vari metodi, basati sulla linearizzazione, per risolvere sistemi di equazioni polinomiali multivariate, ma quelli a cui siamo intererssati in questa trattazione sono fondati sulla teoria delle basi di Gröbner.

Introduciamo, adesso, tutte le nozioni necessarie per dare la definizione di base di Gröbner e descrivere i relativi algoritmi per trovarla. In questo capitolo analizzeremo solamente l'algoritmo di Buchberger che costituisce la base da cui partire per poter, poi, descrivere dei suoi miglioramenti, come l'algoritmo F4, Gebauer e Möller e l'algoritmo F5.

4.1 Varietà affini e ideali

In questa sezione introduciamo gli ideali e le varietà affini seguendo la descrizione di [**CLO92**] (cambiando qualche notazione).

Definizione 4.1 (Monomio). Un *monomio* in n variabili $x_1, \ldots, x_n$ è un prodotto della forma

$$x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdot \ldots \cdot x_n^{\alpha_n},$$

dove tutti gli esponenti stanno in $\mathbb{Z}^+$, gli interi non negativi. Il *grado totale* di un monomio è la somma di tutti gli esponenti del monomio: $\alpha_1 + \alpha_2 + \cdots + \alpha_n$. Semplifichiamo le notazioni ponendo $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{Z}^+$ e

$$x^\alpha = x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdot \ldots \cdot x_n^{\alpha_n}.$$

Indichiamo il grado totale di x^α con $|\alpha|$.

Definizione 4.2 (Polinomio). Un *polinomio* f nelle variabili $x_1, \ldots, x_n$, con coefficienti in un campo k, è una combinazione lineare finita di monomi. Scriveremo un polinomio f nella forma

$$f = \sum_\alpha c_\alpha x^\alpha, \quad c_\alpha \in k,$$

dove la somma è su un numero finito di n-ple $\alpha = (\alpha_1, \ldots, \alpha_n)$. L'insieme di tutti i polinomi nelle incognite $x_1, \ldots, x_n$, con coefficienti in k, è denotato con $k[x_1, \ldots, x_n]$. In seguito indicheremo con P l'anello di polinomi sul campo k in n variabili, $k[x_1, \ldots, x_n]$, se non indicato in altra maniera.

Useremo la seguente terminologia

Definizione 4.3. Sia $f = \sum_\alpha c_\alpha x^\alpha$ un polinomio in P.

- Chiamiamo c_α il *coefficiente* del monomio x^α nel polinomio f.
- Se $c_\alpha \neq 0$, allora chiamiamo $c_\alpha x^\alpha$ un *termine* di f.
- Il *grado totale* di f, denotato con $deg(f)$, è il massimo $|\alpha|$ tale che il coefficiente c_α è diverso da zero.
- L'insieme dei monomi in P è denotato con $T(P)$; l'insieme dei monomi di f è denotato con $T(f)$, ed è chiamato *supporto*.

Definizione 4.4. Dato un campo k e un intero positivo n, definiamo lo *spazio affine* n-dimensionale come l'insieme

$$k^n = \{(a_1, \ldots, a_n) \ : \ a_1, \ldots, a_n \in k\}.$$

Il valore di un polinomio f in $(a_1, \ldots, a_n) \in k^n$ è una funzione

$$f : k^n \to k$$

dove ogni x_i è rimpiazzata da a_i, per $1 \leq i \leq n$. La nozione di varietà è inevitabile perché in questa tesi ci occupiamo della risoluzione di sistemi di equazioni polinomiali. L'insieme di tutte le soluzioni di un sitsema di equazioni

$$f_1(x_1, \ldots, x_n) = \ldots = f_n(x_1, \ldots, x_n) = 0$$

è chiamato una varietà affine, definita come segue.

Definizione 4.5 (Varietà affine). Sia k un campo e $f_1, \ldots, f_m$ polinomi in P. Definiamo

$$\mathbf{V}(f_1, \ldots, f_m) = \{(a_1, \ldots, a_n) \in k^n \ : \ f_i(a_1, \ldots, a_n) = 0 \text{ per ogni } 1 \leq i \leq m\}.$$

Chiamiamo $\mathbf{V}(f_1, \ldots, f_m)$ la *varietà affine* definita da $f_1, \ldots, f_m$.

Esempio 4.1. Sia $k = \mathbb{R}$. Consideriamo il piano $\mathbb{R}^2$ con la varietà $\mathbf{V}(x^2 + y^2 - 1)$, che è la circonferenza di raggio 1 e centro l'origine.

Nel seguente lemma vediamo che l'intersezione e l'unione di varietà affini è ancora una varietà affine.

Lemma 4.1. *Siano* $V = \mathbf{V}(f_1, \ldots, f_s)$, $W = \mathbf{V}(g_1, \ldots, g_t) \subset k^n$ *varietà affini, allora lo sono anche le seguenti:*

$V \cap W = \mathbf{V}(f_1, \ldots, f_s, g_1, \ldots, g_t)$,

$V \cup W = \mathbf{V}(f_i g_j \ : \ 1 \leq i \leq s, 1 \leq j \leq t)$.

Dimostrazione. Vedi [**CLO92**, p.11]. □

Un sottoinsieme I di un anello P è detto *chiuso rispetto all'addizione* se:

$$a, a' \text{ appartengono ad } I \text{ allora } a + a' \text{ appartiene ad } I,$$

ed è detto *chiuso rispetto alla moltiplicazione* con elementi dell'anello se:

$$a \text{ appartiene ad } I \text{ e } r \text{ appartiene a } P \text{ allora } ar \text{ appartiene ad } I.$$

Un sottoinsieme non vuoto di un anello, che è chiuso rispetto a queste due operazioni, è detto un *ideale*.

Definizione 4.6 (Ideale). Un sottoinsieme $I \subset k[x_1, \ldots, x_n]$ è un *ideale* se soddisfa le seguenti proprietà:

(i) $0 \in I$;

(ii) se $f, g \in I$, allora $f + g \in I$;

(iii) se $f \in I$ e $h \in k[x_1, \ldots, x_n]$, allora $hf \in I$.

Definizione 4.7. Siano $f_1, \ldots, f_m$ polinomi in P. Definiamo l'ideale generato da questi polinomi come:

$$\langle f_1, \ldots, f_m \rangle = \left\{ \sum_{i=1}^{m} h_i f_i \ : \ h_1, \ldots, h_m \in P \right\}.$$

L'ideale $\langle f_1, \ldots, f_s \rangle$ ha un'interessante interpretazione in termini di equazioni polinomiali. Dati $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$, otteniamo il sistema di eqazioni

$$\begin{aligned} f_1 &= 0 \\ &\vdots \\ f_s &= 0. \end{aligned}$$

Da queste equazioni se ne possono ottenere delle altre. Per esempio, se moltiplichiamo la prima equazione per $h_1 \in k[x_1, \ldots, x_n]$, la seconda per $h_2 \in k[x_1, \ldots, x_n]$ e così via e sommiamo le equazioni risultanti, otteniamo:

$$h_1 f_1 + h_2 f_2 + \cdots + h_s f_s = 0,$$

che è una conseguenza del sistema originale. Osserviamo che il membro sinistro di questa equazione è esattamente un elemento dell'ideale $\langle f_1, \ldots, f_s \rangle$, quindi possiamo pensare a $\langle f_1, \ldots, f_s \rangle$ come formato da tutte le combinazioni di polinomi dell'equazione $f_1 = f_2 = \cdots = f_s = 0$.

Se esiste un insieme finito di polinomi in P che genera un dato ideale, allora chiamiamo questo insieme una *base* e diciamo che l'ideale è *finitamente generato*. Nella sezione 4.2 dimostreremo un teorema fondamentale dell'algebra commutativa, chiamato teorema delle basi di Hilbert. Questo teorema afferma che ogni ideale in P è finitamente generato. Osserviamo che un dato ideale può avere più basi differenti. Nella sezione 4.2 dimostreremo che è possibile scegliere un utile tipo di basi, chiamate basi di Gröbner.

Proposizione 4.2. *Se $f_1, \ldots, f_s$ e $g_1, \ldots, g_t$ sono basi dello stesso ideale in P, tali che $\langle f_1, \ldots, f_s \rangle = \langle g_1, \ldots, g_t \rangle$, allora $\mathbf{V}(f_1, \ldots, f_s) = \mathbf{V}(g_1, \ldots, g_t)$.*

Dimostrazione. Per ipotesi ogni $f \in \langle f_1, \ldots, f_s \rangle$ è anche in $\langle g_1, \ldots, g_t \rangle$ e, quindi, può essere espressa come $f = h_1 g_1 + \cdots + h_t g_t$, con $h_i \in k[x_1, \ldots, x_n]$, per ogni $1 \leq i \leq t$. Quindi, ogni $a = (a_1, \ldots, a_n) \in \mathbf{V}(g_1, \ldots, g_t)$ verifica $f(a) = 0$ ed anche $g \in \langle g_1, \ldots, g_t \rangle$ verifica $g(a) = 0$, per definizione stessa di $\langle g_1, \ldots, g_t \rangle$. Abbiamo così dimostrato che entrambe le varietà sono costituite dagli stessi punti. □

Esempio 4.2. Sia $k = \mathbb{R}$, e

$$f_1 = 2x^2 + 3y^2 - 11; \quad f_2 = x^2 - y^2 - 3; \ \text{ in } k[x, y]$$

Abbiamo che $\langle 2x^2 + 3y^2 - 11, x^2 - y^2 - 3 \rangle = \langle x^2 - 4, y^2 - 1 \rangle$, poiché:

$$2(x^2 - 4) + 3(y^2 - 1) = 2x^2 - 8 + 3y^2 - 3 = 2x^2 + 3y^2 - 11 = f_1$$

$$(x^2 - 4) - (y^2 - 1) = x^2 - y^2 - 3.$$

Da cui, applicando la proposizione 4.2, abbiamo che:

$$\mathbf{V}(f_1, f_2) = \mathbf{V}(x^2 - 4, y^2 - 1) = \{(\pm 2, \pm 1)\}$$

Così, cambiando la base dell'ideale, è più semplice determinare la varietà.

La capacità di cambiare la base di un ideale, senza incidere sulla varietà, gioca un ruolo chiave nel procedimento di risoluzione di un sistema di equazioni polinomiali. Nella sezione 4.2 introduciamo le basi di Gröbner e l'algoritmo di Buchberger, i quali risultano essere strumenti potenti per capire le varietà affini. Per studiare i polinomi che si annullano sulla stessa varietà introduciamo un nuovo oggetto algebrico.

Definizione 4.8. Sia $V \subset k^n$ una varietà affine. Poniamo allora:

$$\mathbf{I}(V) = \{f \in k[x_1, \ldots, x_n] \ : \ f(a_1, \ldots, a_n) = 0 \text{ per ogni } (a_1, \ldots, a_n) \in V\}$$

Un'osservazione importante è che $\mathbf{I}(V)$ è anche un'ideale (vedi [**CLO92**, p.32]). Può sembrare non completamente intuitivo il fatto che $\mathbf{I}(\mathbf{V}(f_1, \ldots, f_m))$ in generale non è uguale a $\langle f_1, \ldots, f_m \rangle$.

Lemma 4.3. *Se $f_1, \ldots, f_m \in P$, allora $\langle f_1, \ldots, f_m \rangle \subseteq \mathbf{I}(\mathbf{V}(f_1, \ldots, f_m))$, sebbene l'uguaglianza non si verifichi necessariamente.*

Dimostrazione. Sia $f \in \langle f_1, \ldots, f_m \rangle$, cioè $f = \sum_{i=1}^m h_i f_i$ per polinomi $h_1, \ldots h_m \in P$. Poiché $f_1, \ldots, f_m$ si annullano su $\mathbf{V}(f_1, \ldots, f_m)$, allora deve annullarsi anche $\sum_{i=1}^m h_i f_i$. Questo prova che $f \in \mathbf{V}(f_1, \ldots, f_m)$. Per la dimostrazione della seconda parte del lemma, abbiamo bisogno di un esempio in cui $\mathbf{I}(\mathbf{V}(f_1, \ldots, f_m))$ è strettamente più grande di $\langle f_1, \ldots, f_m \rangle$. Dimostriamo che l'inclusione $\langle x^2, y^2 \rangle \subset \mathbf{I}(\mathbf{V}(x^2, y^2))$ è stretta. Calcoliamo prima di tutto $\mathbf{I}(\mathbf{V}(x^2, y^2))$. Le equazioni $x^2 = y^2 = 0$ implicano che $\mathbf{V}(x^2, y^2) = \{(0, 0)\}$. Verifichiamo adesso che $\mathbf{I}(\{(0, 0)\}) = \langle x, y \rangle$. È banale verificare che ogni polinomio della forma $A(x, y)x + B(x, y)y$ si annulla nell'origine. Quindi $A(x, y)x + B(x, y)y \in \mathbf{I}(\{(0, 0)\})$. Rimane da dimostrare l'altra inclusione: supponiamo che $f = \sum_{i,j} a_{ij} x^i y^j$ si annulli nell'origine, allora $a_{00} = f(0, 0) = 0$ e, quindi:

$$f = a_{00} + \sum_{i,j \neq 0,0} a_{ij} x^i y^j = 0 + \left(\sum_{\substack{i,j \\ i>0}} a_{ij} x^{i-1} y^j \right) x + \left(\sum_{j>0} a_{0j} y^{j-1} \right) y \in \langle x, y \rangle.$$

Abbiamo dimostrato, così, che $\mathbf{I}(\{(0, 0)\}) = \langle x, y \rangle$, quindi si ha: $\mathbf{I}(\mathbf{V}(f_1, \ldots, f_m)) = \langle x, y \rangle$. Per vedere che è strettamente più grande di $\langle x^2, y^2 \rangle$, osserviamo che $x \notin \langle x^2, y^2 \rangle$ poiché per polinomi della forma $h_1(x, y)x^2 + h_2(x, y)y^2$ ogni monomio ha grado totale almeno 2. □

4.2 Basi di Gröbner

Nel paragrafo precedente, abbiamo visto che il problema di risolvere un sistema di equazioni polinomiali

$$f_1(x_1, \ldots, x_n) = \ldots = f_m(x_1, \ldots, x_n) = 0$$

è lo stesso che trovare i punti della varietà affine $\mathbf{V}(f_1, \ldots, f_m)$. Per ottenere questo risultato, introduciamo le basi di Gröbner, che risultano essere un potente strumento per descrivere gli ideali in termini di insiemi finiti di generatori. In questo paragrafo descriviamo le idee principali per trovare un tale insieme.

4.2.1 Ordinamento dei monomi

Sia P un anello di polinomi. L'insieme $T(P)$ dei monomi è importante per noi ed è possibile scegliere varie maniere di ordinare gli elementi di questo insieme.

Definizione 4.9 (Ordinamento monomiale). Un *ordinamento monomiale* su P è una relazione $>$ su $\mathbb{Z}_+^n$ o, equivalentemente, una relazione sull'insieme dei monomi x^α, con $\alpha \in \mathbb{Z}_+^n$, soddisfacente le seguenti condizioni:

(i) $>$ è un ordinamento totale su $\mathbb{Z}_+^n$;

(ii) se $\alpha > \beta$ e $\alpha, \beta, \gamma \in \mathbb{Z}_+^n$, allora $\alpha + \gamma > \beta + \gamma$;

(iii) $>$ è un *buon ordinamento* su $\mathbb{Z}_+^n$. Questo vuol dire che ogni sottoinsieme non vuoto di $\mathbb{Z}_+^n$ ha un elemento più piccolo rispetto a $>$.

Uno specifico ordinamento σ per $>$ è denotato con $>_\sigma$. La proprietà (*iii*) della definizione 4.9, il buon ordinamento, è spesso riferita come l'ordinamento ammissibile ed ha un'importante implicazione: è spesso usata per dimostrare che un particolare algoritmo che fa un uso esplicito del buon ordinamento, come l'algoritmo di Buchberger, alla fine termina.

Lemma 4.4. *Una relazione d'ordine $>$ su $\mathbb{Z}_+^n$ è un buon ordinamento se e solo se ogni sequenza decrescente in $\mathbb{Z}_+^n$*

$$\alpha(1) > \alpha(2) > \alpha(3) > \cdots$$

alla fine termina.

Dimostrazione. Vedi [**CLO92**, p.54] □

Definiamo adesso i tre ordinamenti monomiali più importanti per le nostre applicazioni.

Definizione 4.10 (Ordinamento lessicografico). Siano $\alpha = (\alpha_1, \ldots, \alpha_n)$ e $\beta = (\beta_1, \ldots, \beta_n) \in \mathbb{Z}_+^n$. Diciamo che $\alpha >_{lex} \beta$ se nella differenza vettoriale $\alpha - \beta \in \mathbb{Z}_+^n$ il primo elemento a sinistra, diverso da zero, è positivo. Scriveremo $x^\alpha >_{lex} x^\beta$ se $\alpha >_{lex} \beta$.

Esempio 4.3. $(1, 2, 0) >_{lex} (0, 3, 4)$, poiché $\alpha - \beta = (1, -1, -4)$.

Osservazione 4.1. L'ordinamento lessicografico è analogo all'ordinamento delle parole usato nei dizionari. Possiamo vedere gli elementi di una n-pla $\alpha \in \mathbb{Z}_+^n$ analogamente alle lettere che formano una parola. Le lettere sono ordinate alfabeticamente:

$$\text{a>b>} \cdots \text{>y>z.}$$

Quindi, per esempio,

$$\text{marina}>_{lex}\text{mario}$$

poiché la quinta lettera di Marina viene prima delle quinta lettera di Mario (n$>_{lex}$o)

Definizione 4.11 (Graded lex order). Siano $\alpha, \beta \in \mathbb{Z}_+^n$. Diciamo che $\alpha >_{grlex} \beta$ se

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i, \text{ oppure se } |\alpha| = |\beta| \ \alpha >_{lex} \beta.$$

Per esempio:

1. $(1,2,3) >_{grlex} (3,2,0)$, poichè $|(1,2,3)| = 6 > |(3,2,0)| = 5$.
2. $(1,2,4) >_{grlex} (1,1,5)$, poiché $|(1,2,4)| = |(1,1,5)|$, e $(1,2,4) >_{lex} (1,1,5)$.
3. Le variabili sono ordinate come nel *lex*:

$$(1,0,\ldots,0) >_{grlex} (0,1,\ldots,0) >_{grlex} \cdots >_{grlex} (0,\ldots,1)$$

oppure

$$x_1 >_{grlex} x_2 >_{grlex} \cdots >_{greex} x_n$$

Definizione 4.12 (Graded reverse lex order). Siano $\alpha, \beta \in \mathbb{Z}_+^n$. Diciamo che $\alpha >_{grevlex} \beta$ se

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i,$$

oppure se $|\alpha| = |\beta|$ e, in $\alpha - \beta \in \mathbb{Z}^n$, il primo elemento a destra, non nullo, è negativo.

Per esempio:

1. $(4,7,1) >_{grevlex} (4,2,3)$, poichè $|(4,7,1)| = 12 > |(4,2,3)| = 9$.
2. $(1,5,2) >_{grevlex} (4,1,3)$, poiché $|(1,5,2)| = |(4,1,3)|$, e $\alpha - \beta = (-3,4,-1)$.

3. Osserviamo, inoltre, che *lex* e *grevlex* danno lo stesso ordinamento sulle variabili:

$$(1, 0, \ldots, 0) >_{grevlex} (0, 1, \ldots, 0) >_{grevlex} \cdots >_{grevlex} (0, \ldots, 1)$$

oppure

$$x_1 >_{grevlex} x_2 >_{grevlex} \cdots >_{grevlex} x_n$$

4. $x^5yz >_{grevlex} x^4yz^2$, poiché entrambi i monomi hanno grado totale 7, e $x^5yz >_{lex} x^4yz^2$.

Ci sono altre definizioni comuni per termini speciali dei polinomi rispetto agli ordinamenti.

Definizione 4.13. Sia $f = \sum_\alpha a_\alpha x^\alpha$ un polinomio diverso da zero in P e sia $>$ un ordinamento monomiale.

- Il *multigrado* di f è

$$multideg(f) = max_>(\alpha \in \mathbb{Z}_+^n \ : \ a_\alpha \neq 0).$$

- Il *grado totale* di f è

$$totaldeg(f) = \sum_{1 \leq i \leq n} multideg(f)_i.$$

- Il *coefficiente principale* di f è

$$LC(f) = a_{multideg(f)} \in k.$$

- Il *monomio principale* di f è

$$LM(f) = x^{multideg(f)}.$$

- Il *termine principale* di f è

$$LT(f) = LC(f) \cdot LM(f).$$

- Il polinomio f è detto *monico* se

$$LC(f) = 1.$$

Per spiegare la precedente definizione diamo un esempio: sia $f = 4xy^2z + 4z^2 - 5x^3 + 7x^2z^2$, e indichiamo con $>$ l'ordinamento lessicografico. Allora

$$\begin{gathered} multideg(f) = (3,0,0) \\ LC(f) = -5 \\ LM(f) = x^3 \\ LT(f) = -5x^3 \end{gathered}$$

Lemma 4.5. *Siano $f, g \in k[x_1, \ldots, x_n]$ polinomi diversi da zero. Allora:*

(i) $multideg(fg) = multideg(f) + multideg(g)$*;*

(ii) *se $f + g \neq 0$, allora $multideg(f + g) \leq max(multideg(f), multideg(g))$.*

Se $multideg(f) \neq multideg(g)$, allora vale l'uguaglianza.

Osservazione 4.2. Per un sottoinsieme F di un anello polinomiale P, denotiamo con $LT(F)$ e $LM(F)$, rispettivamente, $\{LT(f) \ : \ f \in F\}$ e $\{LM(f) \ : \ f \in F\}$.

4.2.2 Algoritmo della divisione in $k[x_1, \ldots, x_n]$

Per verificare quando un elemento f è un membro di un ideale $\langle f_1, \ldots, f_m \rangle \subseteq P$ c'è bisogno di mostrare se può essere scritto come una combinazione di elementi in $\{f_1, \ldots, f_m\}$, nella forma

$$f = a_1 f_1 + \cdots + a_m f_m,$$

per $a_i \in P$. Se ciò è verfificato, allora f è nell'ideale dato. Questa verifica è fatta dall'algoritmo della divisione per il caso a più variabili, che è introdotto esplicitamente nel seguente teorema.

Teorema 4.6 (Algoritmo della divisione). *Fissiamo un ordinamento monomiale $>$ su $\mathbb{Z}_+^n$ e sia $F = (f_1, \ldots, f_m)$ una m-pla ordinata di polinomi in P. Allora ogni $f \in P$ può essere scritta come:*

$$f = a_1 f_1 + \cdots + a_m f_m + r,$$

dove $a_i, r \in P$, e, o r è zero, oppure r è una combinazione k-lineare di monomi, nessuno dei quali è divisibile da $LT(f_1), \ldots, LT(f_m)$. Chiameremo r il resto di f nella divisione per F. Inoltre, se $a_i f_i \neq 0$, allora si ha

$$multideg(f) \geq multideg(a_i f_i).$$

Dimostrazione. Vedi [**CLO92**, p.63]. Questa dimostrazione introduce l'algoritmo della divisione per l'anello polinomiale multivariato P che ritorna $a_1, \ldots, a_m$ e r del teorema e, così, ne dimostra l'esistenza. □

Definizione 4.14. L'*algoritmo della divisione* è definito come segue.
Input: $f_1, \ldots, f_m, f \in P$
Output: $a_1, \ldots, a_m, r \in P$, tali che f può essere espressa come nel teorema 4.6 (cioé il resto r, ottenuto dalla divisione di f per F).

$a_1 = 0, \ldots, a_m = 0; r = 0$
$p := f$
while $p \neq 0$ **do**
  $i := 1$
  $divisionoccurred := false$
  while $i \leq m$ **and** $divisionoccurred := false$ **do**
    if $LT(f_i)$ divide $LT(p)$ **then**
      $a_i := a_i + LT(p)/LT(f_i)$
      $p := p - (LT(p)/LT(f_i))f_i$
      $divisionoccurred := true$
    else $i := i + 1$
  if $divisionoccurred = false$ **then**
    $r := r + LT(p)$
    $p := p - LT(p)$

La variabile p rappresenta il dividendo ad ogni passo dell'algoritmo (che appunto termina quando p diventa zero), r rappresenta il resto ad ogni passo dell'algoritmo, e la variabile booleana *divisionoccurred* ci dice quando $LT(f_i)$ divide il termine principale del dividendo intermedio p. Ogni volta che entriamo nel while-loop principale, succede una delle seguenti due cose.

- (Passo di divisione) Se un $LT(f_i)$ divide $LT(p)$, allora l'algoritmo procede come nel caso ad una variabile.
- (Passo del resto) Se nessun $LT(f_i)$ divide $LT(p)$, allora l'algoritmo aggiunge $LT(p)$ al resto.

Esempio 4.4. Supponiamo di avere $f = -7y^3z - 12x^2z^4$ e di usare l'ordinamento lessicografico $x > y > z$. Vogliamo dividere f per $F = (f_1 = 4x^2z - 7y^2; f_2 = xyz^2 + 3xz^4)$ applicando l'algoritmo della divisione.
$p := f$, $LT(f_1) = 4x^2z$ divide $LT(p) = -12x^2z^4$ e, quindi: $a_1 = -3z^3$ e $p = -7y^3z - 21y^2z^3$. Adesso $LT(p) = -7y^3z$ e non è diviso da nessun termine principale di f_i, quindi $r = -7y^3z$ e $p = -21y^2z^3$, ed ha ancora termine principale non divisibile da nessun $LT(f_i)$ e quindi $r = -7y^3z - 21y^2z^3$ e $p = 0$; esco così dal while-loop principale e l'algoritmo termina. Ho ottenuto, quindi, che:$f = -3z^3 \cdot f_1 + r$.

In alcuni casi le permutazioni dell'insieme $\{f_1, \ldots, f_m\}$ possono portare a resti

differenti [**CLO92**, p.65]. Come vedremo nella sezione 4.3, questo non avviene nel caso che l'iniseme sia una base di Gröbner. Nel caso ad una variabile, l'algoritmo della divisione include un criterio sul grado del resto; ma non è così nel caso a più variabili.

4.2.3 Ideali monomiali e Lemma di Dickson

Una caratteristica importante delle basi di Gröbner è che descrivono una base finita per l'ideale generato da tutti i termini principali che compaiono nell'ideale generato da un dato insieme di polinomi. Questo ideale di termini principali è un ideale monomiale, una sottoclasse di ideali con speciali proprietà.

Definizione 4.15 (Ideale monomiale). Un ideale $I \subset k[x_1, \ldots, x_n]$ è un *ideale monomiale* se può essere generato da monomi oppure, equivalentemente, se esiste un sottoinsieme $A \subset \mathbb{Z}_+^n$ (anche infinito) tale che I è formato da tutti i polinomi che sono somme finite della forma $\sum_\alpha h_\alpha x^\alpha$, dove $h_\alpha \in k[x_1, \ldots, x_n]$. In questo caso, si scrive $I = \langle x^\alpha \ : \ \alpha \in A \rangle$.

Lemma 4.7. *Sia $I = \langle x^\alpha \ : \ \alpha \in A \rangle$ un ideale monomiale, allora un monomio x^β si trova in I se e soltanto se x^β è divisibile per x^α, per qualche $\alpha \in A$.*

Dimostrazione. Se x^β è un multiplo di x^α per qualche $\alpha \in A$, allora $x^\beta \in I$, per definizione di ideale. Viceversa, se $x^\beta \in I$, allora $x^\beta = \sum_{i=1}^m h_i x^{\alpha(i)}$, dove $h_i \in k[x_1, \ldots, x_n]$ e $\alpha(i) \in A$. Se espandiamo ogni h_i come combinazione lineare di monomi, vediamo che ogni termine del lato destro dell'equazione è divisibile per qualche $x^{\alpha(i)}$; quindi anche il lato sinistro x^β dell'equazione avrà la stessa propietà. □

Lemma 4.8. *Sia $I = \langle x^\alpha \ : \ \alpha \in A \rangle$ un ideale monomiale e sia $f \in k[x_1, \ldots, x_n]$, allora le seguenti condizioni sono equivalenti:*

(i) *$f \in I$*

(ii) *ogni termine di f stà in I, cioè $T(f) \subset I$*

(iii) *f è una combinazione k-lineare dei monomi di I*

Dimostrazione. Vedi [**CLO92**, p.69] □

Un'immediata conseguenza della propietà (iii) del lemma precedente è che un ideale monomiale è univocamente determinato dai suoi monomi. Si ha il seguente corollario.

Corollario 4.9. *Due ideali monomiali sono gli stessi se e soltanto se contengono gli stessi monomi.*

Un'importante risultato per la costruzione del teorema delle basi di Hilbert, è la proprietà che gli ideali monomiali di P sono finitamente generati. Questo è enunciato esplicitamente nel seguente teorema, chiamato lemma di Dickson.

Teorema 4.10 (Lemma di Dickson). *Un ideale monomiale $I = \langle x^\alpha : \alpha \in A\rangle \subset k[x_1, \ldots, x_n]$ può essere scritto nella forma $I = \langle x^{\alpha(1)}, \ldots, x^{\alpha(s)}\rangle$, dove $\alpha(1), \ldots, \alpha(s) \in A$. In particolare I ha una base finita.*

Dimostrazione. (Per induzione sul numero n delle variabili). Se $n = 1$, allora I è generato dai monomi x^α, dove $\alpha \in A \subset \mathbf{Z}_+$. Sia β il più piccolo elemento di $A \subset \mathbf{Z}_+$. Allora x^β divide tutti gli altri generatori, e quindi $I = \langle x^\beta\rangle$.
Assumiamo adesso che $n > 1$ e che il teorema sia vero per $n - 1$. Scriveremo le variabili come $x_1, \ldots, x_{n-1}, y$, in modo che i monomi in $k[x_1, \ldots, x_{n-1}, y]$ possano essere scritti come $x^\alpha y^m$, dove $\alpha = (\alpha_1, \ldots, \alpha_{n-1}) \in \mathbf{Z}_+^{n-1}$ e $m \in \mathbf{Z}_+$. Supponiamo che $I \subset k[x_1, \ldots, x_n]$ sia un ideale monomiale. Per trovare generatori per I, sia J l'ideale in $k[x_1, \ldots, x_{n-1}]$ generato dai monomi x^α per i quali $x^\alpha y^m \in I$ per qualche $m \geq 0$. Poiché J è un ideale monomiale in $k[x_1, \ldots, x_{n-1}]$, l'ipotesi induttiva implica che un numero finito di x^α generano J, $J = \langle x^{\alpha(1)}, \ldots, x^{\alpha(s)}\rangle$. L'ideale J può essere interpretato come la "proiezione" di I su $k[x_1, \ldots, x_{n-1}]$. Per ogni i tra 1 ed s, dalla definizione di J si ha che $x^{\alpha(i)} y^{m_i} \in I$, per qualche $m_i \geq 0$. Sia m il più grande degli m_i. Allora, per ogni k tra 0 e $m - 1$, consideriamo l'ideale $J_k \subset k[x_1, \ldots, x_{n-1}]$, generato dai monomi x^β tali che $x^\beta y^k \in I$. Si può pensare a J_k come alla "fetta" di I generata dai monomi contenenti y esattamente alla k-esima potenza. Usando, di nuovo, l'ipotesi induttiva, J_k ha un insieme finito di generatori di monomi, $J_k = \langle x^{\alpha_1(1)}, \ldots, x^{\alpha_k(s_k)}\rangle$. Asseriamo che I ègenerato da monomi che stanno nella seguente lista, sceltinel modo seguente:

da J consideriamo i seguenti monomi :	$x^{\alpha(1)} y^m, \ldots, x^{\alpha(s)} y^m$,
da J_0 consideriamo i seguenti monomi :	$x^{\alpha_0(1)}, \ldots, x^{\alpha_0(s_0)}$,
da J_1 consideriamo i seguenti monomi :	$x^{\alpha_1(1)} y, \ldots, x^{\alpha_1(s_1)} y$,
	$\vdots$
da J_{m-1} consideriamo i seguenti monomi :	$x^{\alpha_{m-1}(1)} y^{m-1}, \ldots, x^{\alpha_{m-1}(s_{m-1})} y^{m-1}$.

Per prima cosa notiamo che ogni monomio in I è divisibile per uno della lista. Per vedere quale, sia $x^\alpha y^p \in I$. Se $p \geq m$, allora $x^\alpha y^p$ è divisibile per qualcuno dei $x^{\alpha(i)} y^m$ dalla costruzione di J. Se, invece, $p \leq m - 1$, allora $x^\alpha y^p$ è divisibile per qualche $x^{\alpha_p(j)} y^p$ per costruzione di J_p. Dal lemma 4.7 segue che i monomi sopra descritti generano un ideale avente gli stessi monomi di I. Dal corollario 4.9 abbiamo che i due ideali sono gli stessi, e l'affermazione di prima è dimostrata. Per completare la dimostrazione del teorema, dobbiamo dimostrare che l'insieme finito di generatori può essere scelto da un dato insieme di generatori per l'ideale. Se riscriviamo le variabili come $x_1, \ldots, x_n$, allora il nostro ideale monomiale è $I =$

$\langle x^\alpha : \alpha \in A\rangle \subset k[x_1, \ldots, x_n]$. Dobbiamo dimostrare che I è generato da un numero finito di x^α, dove $\alpha \in A$. Da quanto visto prima, abbiamo che $I = \langle x^{\beta(1)}, \ldots, x^{\beta(s)}\rangle$ per alcuni monomi $x^{\beta(i)}$ in I. Poiché $x^{\beta(i)} \in I = \langle x^\alpha : \alpha \in A\rangle$, dal lemma 4.7 abbiamo che ogni $x^{\beta(i)}$ è divisibile per $x^{\alpha(i)}$ per qualche $\alpha(i) \in A$. A questo punto è facile verificare che $I = \langle x^{\alpha(1)}, \ldots, x^{\alpha(s)}\rangle$. □

Diamo il seguente esempio per capire meglio come lavora la dimostrazione del teorema 4.10.

Esempio 4.5. Applichiamo lo schema della dimostrazione all'ideale:

$$I = \langle x^4y^2, x^3y^4, x^2y^5\rangle.$$

Da come sono fatti gli esponenti si ha che la proiezione è $J = \langle x^2\rangle \subset k[x]$. Poiché $x^2y^5 \in I$, abbiamo che $m = 5$. Adesso abbiamo la fetta J_k, $0 \leq k \leq 4 = m - 1$, generata dai monomi conteneti y^k:

$$\begin{aligned} J_0 &= J_1 = \{0\},\\ J_2 &= J_3 = \langle x^4\rangle,\\ J_4 &= \langle x^3\rangle. \end{aligned}$$

Dalla dimostrazione del teorema 4.10 abbiamo:

$$I = \langle x^2y^5, x^4y^2, x^4y^3, x^3y^4\rangle.$$

Per decrivere un dato ideale con un insieme finito di generatori vedremo che il suo ideale monomiale di termini principali gioca un ruolo chiave.

4.2.4 Teorema delle basi di Hilbert e Basi di Gröbner

Definizione 4.16. Sia $I \subset k[x_1, \ldots, x_n]$ un ideale diverso da $\{0\}$.

(i) Denotiamo con $LT(I)$ l'insieme dei termini principali degli elementi di I:

$$LT(I) = \{cx^\alpha : \text{ esiste } f \in I \text{ con } LT(f) = cx^\alpha\}.$$

(ii) Denotiamo con $\langle LT(I)\rangle$ l'ideale generato dagli elementi di $LT(I)$.

Come conseguenza del lemma di Dickson abbiamo la seguente proposizione.

Proposizione 4.11. *Sia $I \subset k[x_1, \ldots, x_n]$ un ideale.*

(i) *$\langle LT(I)\rangle$ è un ideale monomiale.*

(ii) *Esistono $g_1, \ldots, g_m \in I$ tali che $\langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$.*

Dimostrazione. (i) I monomi principali $LM(g)$ di elementi $g \in I - \{0\}$ generano l'ideale monomiale $\langle LM(g) : g \in I - \{0\}\rangle$. Poiché $LM(g)$ e $LT(g)$ differiscono per una costante diversa da zero, l'ideale è uguale a $\langle LT(g) : g \in I - \{0\}\rangle = \langle LT(I)\rangle$. Quindi, $\langle LT(I)\rangle$ è un ideale monomiale.
(ii) Poiché $\langle LT(I)\rangle$ è generato dai monomi $LM(g)$, con $g \in I - \{0\}$, dal lemma di Dickson abbiamo che $\langle LT(I)\rangle = \langle LM(g_1), \ldots, LM(g_m)\rangle$, per un numero finito di $g_1, \ldots, g_m \in I$. Poiché $LM(g_i)$ differisce da $LT(g_i)$ soltanto per una costante non nulla, segue che $\langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$.

□

Possiamo, adesso, usare la proposizione 4.11 e l'algoritmo della divisione per dimostrare l'esistenza di un sitema finito di generatori per ogni ideale polinomiale. Includiamo la dimostrazione perché dà una traccia di come un algoritmo può verificare se un polinomio è o non è un membro di un dato ideale.

Teorema 4.12 (Teorema delle basi di Hilbert). *Ogni ideale $I \subset k[x_1, \ldots, x_n]$ ha un insieme finito di generatori; ovvero, $I = \langle g_1, \ldots, g_m\rangle$, dove $g_1, \ldots, g_m \in I$.*

Dimostrazione. Se $I = \{0\}$, prendiamo come insieme di generatori $\{0\}$ che è sicuramente finitamente generato. Se I contiene qualche polinomio non nullo, allora possiamo costruire un insieme di generatori per I come segue. Per la proposizione 4.11, esistono $g_1, \ldots, g_m \in I$ che generano $\langle LT(I)\rangle$, $\langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$. Dimostriamo che $I = \langle g_1, \ldots, g_m\rangle$. È chiaro che $\langle g_1, \ldots, g_m\rangle \subset I$, poiché ogni $g_i \in I$. Al contrario, sia $f \in I$ un polinomio, se applichiamo l'algoritmo della divisione, dalla definizione 4.14, per dividere f per $g_1, \ldots, g_m$, allora otteniamo un'espressione della forma

$$f = a_1 g_1 + \cdots + a_m g_m + r$$

dove ogni termine in r non è divisibile per nessun $LT(g_1), \ldots, LT(g_m)$. Si ha che $r = 0$; infatti, per vedere ciò, osserviamo che:

$$r = f - (a_1 g_1 + \cdots + a_m g_m) \in I.$$

Se r fosse diverso da zero, allora $LT(r) \in \langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$ e, per il lemma 4.7, abbiamo che $LT(r)$ deve essere divisibile per qualche $LT(g_i)$. Ciò contraddice la definizione di resto e, quindi, r deve essere zero. Sarà:

$$f = a_1 g_1 + \cdots + a_m g_m + 0 \in \langle g_1, \ldots, g_m\rangle,$$

che mostra, appunto, che $I \subset \langle g_1, \ldots, g_m\rangle$. Prima avevamo dimostrato l'inclusione $\langle g_1, \ldots, g_m\rangle \subset I$, che insieme a $I \subset \langle g_1, \ldots, g_m\rangle$, implica $I = \langle g_1, \ldots, g_m\rangle$. □

Osservazione 4.3. Nella dimostrazione del teorema la base $\{g_1, \ldots, g_m\}$ usata ha la proprietà che $\langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$, ma non tutte le basi di un ideale verificano questa proprietà. Definiremo questo tipo di basi speciali nella definizione 4.17.

Supponiamo, adesso, di volere verificare se una data f è un elemento dell'ideale $I = \langle g_1, \ldots, g_m\rangle$. La dimostrazione del teorema delle basi di Hilbert indica che possiamo espandere la nostra base di $\langle LT(I)\rangle$ finché siano inclusi tutti i monomi minimali; quindi, ogni polinomio $f \in I$ è ridotto a zero rispetto a questa base, poiché il resto non può essere più grande dei termini principali degli elementi della base. Per essere più precisi, esiste una base finita G di I, tale che per ogni $f \in I$, c'è un elemento della base $g \in G$ che verifica $LT(g)$ divide $LT(f)$. Alla luce di questa osservazione diamo la definizione di base di Gröbner.

Definizione 4.17 (Base di Gröbner). Dato un ordinamento monomiale, un sottoinsieme finito $G = \{g_1, \ldots, g_m\}$ di un ideale I è detto essere una *base di Gröbner* (o base standard) se

$$\langle LT(g_1), \ldots, LT(g_m)\rangle = \langle LT(I)\rangle$$

Equivalentemente (ma poco formalmente), un insieme $\{g_1, \ldots, g_m\} \subset I$ è una base di Gröbner se e soltanto se il termine principale di ogni elemento di I è divisibile da uno degli $LT(g_i)$ (questo segue dal lemma 4.7).

Esempio 4.6. Sia $I = \langle f_1, f_2\rangle \subset k[x, y]$, con $f_1 = x^3 - 2xy$ e $f_2 = x^2y - 2y^2 + x$. Supponiamo di usare l'ordinamento grlex sui monomi, allora

$$xf_2 - yf_1 = x^2.$$

Avremo $x^2 \in I$ e $x^2 \in \langle LT(I)\rangle$. Comunque, x^2 non è divisibile per $LT(f_1)$ o $LT(f_2)$, così f_1 e f_2 non possono formare una base di Gröbner di I.

Consideriamo, adesso, l'ideale $J = \langle g_1, g_2\rangle = \langle x + y, y - z\rangle$. Usando l'ordinamento lessicografico (lex) in $\mathbb{R}[x, y, z]$, dimostriamo che g_1 e g_2 formano una base di Gröbner per l'ideale J. Per far ciò bisogna mostrare che ogni termine principale in J giace nell'ideale $\langle LT(g_1), LT(g_2)\rangle = \langle x, y\rangle$, oppure, equivalentemente, che ogni termine principale è divisibile per x o y.
Ogni elemento $f \in J$ può essere scritto come:

$$f = a_1g_1 + a_2g_2 = a_1x + a_1z + a_2y - a_2z,$$

per $a_1, a_2 \in P$. Se a_1x ed a_2y non si cancellano l'uno con l'altro, allora il termine principale di f è in $\langle J\rangle$. Supponiamo, adesso, che a_1x e a_2y si cancellino a vicenda, allora i termini principali di a_1 e a_2 sono divisibili per x o per y, oppure $a_1 = a_2 = 0$. Se a_1 e a_2 non sono zero, allora $a_1z - a_2z \neq 0$ e quindi il termine principale di $a_1z - a_2z$ è divisibile per x o y.

Le basi di Gröbner furono introdotte per la prima volta intorno al 1960 da H.Hinoronaka (che le chiamò basi standard) e, indipendentemente, poco dopo da B. Buchberger nella sua tesi Ph. D.. Il nome basi di Gröbner fù coniato da Buchberger in onore del suo relatore W. Gröbner. Concludiamo questo paragrafo con due applicazioni del teoream delle basi di Hilbert. Il primo è un enunciato algebrico sugli ideali in $k[x_1, \ldots, x_n]$ che afferma: una catena crescente di ideali, cioè una sequenza crescente nidificata:

$$I_1 \subset I_2 \subset I_3 \subset \cdots,$$

ad un certo punto si stabilizza. Questa sarà una condizione necessaria per mostrare che l'algoritmo per il calcolo delle basi di Gröbner ha una fine. La seconda conseguenza è che la varietà corrispondente all'insieme di polinomi è uguale alla varietà dell'ideale generato dal'insieme di polinomi. Diamo gli enunciati formali dei precedenti risultati.

Teorema 4.13 (La condizione della catena crescente). *Sia*

$$I_1 \subset I_2 \subset I_3 \subset \cdots,$$

una catena crescente di ideali in $k[x_1, \ldots, x_n]$. *Allora esiste* $N \geq 1$ *tale che*

$$I_N = I_{N+1} = I_{N+2} = \cdots.$$

Dimostrazione. Vedi [**CLO92**, p.77] □

Definizione 4.18. Sia $I \subset P$ un ideale. Denoteremo con $\mathbf{V}(I)$ l'insieme

$$\mathbf{V}(I) = \{(a_1, \ldots, a_n) \in k^n : f(a_1, \ldots, a_n) = 0 \text{ per ogni} f \in I\}.$$

Proposizione 4.14. $\mathbf{V}(I)$ *è una varietà affine. In particolare, se* $I = \langle f_1, \ldots, f_m \rangle$, *allora* $\mathbf{V}(I) = \mathbf{V}(f_1, \ldots, f_m)$.

Dimostrazione. Vedi [**CLO92**, p.78] □

4.3 Algoritmo di Buchberger

Nel precedente paragrafo è stato mostrato che un ideale può essere descritto da un insieme finito di generatori. Dopo aver descritto un crittosistema con un insieme di equazioni, il problema è trovare gli zeri comuni del corrispondente insieme di polinomi, la varietà affine. La varietà affine è determinata dai polinomi e, equivalentemente, dall'ideale generato dai polinomi. Introduciamo in questa sezione l'algorimo di Buchberger per trovare le basi di Gröbner. Questo sistema finito di generatori speciale risulta essere molto appropriato per trovare gli zeri comuni. Le

basi di Gröbner hanno l'utile proprietà che, se applichiamo l'algoritmo della divisione ad un polinomio f ed a una base di Gröbner G, il resto r sarà unico. Questa è un'osservazione fondamentale per stabilire quando un elemento è membro di un dato ideale.

Proposizione 4.15. *Sia $G = \{g_1, \ldots, g_m\}$ una base di Gröbner per un ideale $I \subset k[x_1, \ldots, x_n]$ e sia $f \in k[x_1, \ldots, x_n]$, allora esiste un unico $r \in k[x_1, \ldots, x_n]$ con le seguenti propietà:*

(i) *i termini di r non sono divisibili per $LT(g_1), \ldots, LT(g_m)$,*

(ii) *esiste $g \in I$ tale che $f = g + r$.*

In particolare, r è il resto della divisione di f per G, non importa come gli elementi di G siano ordinati quando si applica l'algoritmo della divisione.

Dimostrazione. L'algoritmo della divisione ci dà $f = a_1g_1 + \cdots + a_mg_m + r$, dove r soddisfa la (i). Verifichiamo la (ii) ponendo $g = a_1g_1 + \cdots + a_ng_n \in I$. Questo prova l'esistenza di r.

Per provare l'unicità supponiamo che $f = g_1 + r_1 = g_2 + r_2$ soddisfi la (i) e la (ii). Allora $r_2 - r_1 = g_1 - g_2 \in I$, così ché se $r_2 \neq r_1$ allora $LT(r_2 - r_1) \in \langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_m)\rangle$. Dal lemma 4.7 segue che $LT(r_2 - r_1)$ è divisibile per qualche $LT(g_i)$. Questo è impossibile perché nessun termine di r_1, r_2 è divisibile per qualche $LT(g_1), \ldots, LT(g_m)$; allora $r_2 - r_1$ deve essere zero, e l'unicità è provata. □

Benché il resto sia unico, anche per una base di Gröbner, i quozienti a_i prodotti dall'algoritmo della divisione, $f = a_1g_1 + \cdots + a_mg_m + r$, possono cambiare se ordiniamo i generatori in maniera differente. Diamo un esempio per mostrare questo fatto.

Esempio 4.7. Nell'esempio 4.6 avevamo dimostrato che $G = \{x + z, y - z\}$ è una base di Gröbner per l'ordinamento lessicografico. Usiamo questa base per studiare l'unicità dell'algoritmo della divisione.

(a) Dividiamo xy per $x + z, y - z$.

(b) Dividiamo xy, invertendo l'ordine, per $y - z, x + z$.

(a):

$$
\begin{aligned}
&rem\,(xy, G) = \\
&rem\left(xy - \frac{xy}{x}(x + z), G\right) = \\
&rem(xy - y(x + z), G) = \\
&rem(xy - xy - yz, G) = \\
&rem(-yz, G) = \\
&rem\left(-yz + \frac{yz}{y}(y - z), G\right)) = \\
&rem(-z^2, G) = -z^2,
\end{aligned}
$$

e dall'algoritmo della divisione abbiamo che i coefficienti sono:

$$a_1 = \frac{LT(xy)}{LT(x+z)} = \frac{xy}{x} = y$$
$$a_2 = \frac{LT(-yz)}{LT(y-z)} = \frac{-yz}{y} = -z.$$

Da queste relazioni abbiamo:

$$xy = a_1 f_1 + a_2 f_2 + rem = y(x+z) - z(y-z) - z^2 = yx + yz - zy + z^2 - z^2 = xy.$$

(b):Consideriamo, adesso $G = (y-z, x+z)$

$$\begin{aligned}
&rem(xy, G) = \\
&rem\left(xy - \frac{xy}{y}(y-z), G\right) = \\
&rem(xy - x(y-z), G) = \\
&rem(xy - xy + xz, G) = \\
&rem(xz, G) = \\
&rem\left(xz - \frac{xz}{x}(x+z), G\right) = \\
&rem(xz - zx - z^2, G) = \\
&rem(-z^2, G) = -z^2
\end{aligned}$$

e dall'algoritmo della divisione abbiamo che i coefficienti sono:

$$a_1^{'} = \frac{LT(xy)}{LT(y-z)} = \frac{xy}{x} = y$$
$$a_2^{'} = \frac{LT(xz)}{LT(x+z)} = \frac{xz}{x} = z.$$

In questo caso abbiamo:

$$xy = a_1 f_1 + a_2 f_2 + rem = x(y-z) + z(x+z) - z^2 = xy,$$

e lo stesso resto (come dimostrato nella proposizione 4.15), ma i quozienti sono differenti per le due divisioni.

Come corollario, diamo il seguente criterio per vedere quando un polinomio sta nell'ideale.

Corollario 4.16. *Sia $G = \{g_1, \ldots, g_m\}$ una base di Gröbner per un ideale $I \subset k[x_1, \ldots, x_n]$, e sia $f \in k[x_1, \ldots, x_n]$, allora $f \in I$ se e soltanto se il resto della divisione di f per G è zero.*

Dimostrazione. Se il resto è zero, allora $f \in I$. Al contrario, dato $f \in I$, si ha che $f = f + 0$ soddisfa le condizioni della proposizione 4.15. Segue che zero è il resto della divisione di f per G. □

Definizione 4.19. Indicheremo con $\overline{f}^F$ il resto della divisione di f per una m-pla ordinata $F = (f_1, \ldots, f_m)$. Se F è una base di Gröbner per l'ideale $\langle f_1, \ldots, f_m \rangle$, allora possiamo considerare F come un insieme (senza nessun ordine particolare) per la proposizione 4.15 (poiché il resto r non dipende, in questo caso, dall'ordine degli elementi in F).

Nella dimostrazione del teorema delle basi di Hilbert (teorema 4.12), abbiamo osservato che, ad un certo punto, ogni polinomio f nell'ideale $I \subset k[x_1, \ldots, x_n]$ si riduce a zero modulo una base finita G. Nella costruzione di una base di Gröbner ricerchiamo elementi che hanno termini principali che generano l'ideale $\langle LT(I) \rangle$. La divisione di f per un insieme di generatori $G^{'}$, che non soddisfano questo criterio, ha come risultato un resto r che non esite in $\{LT(g) : g \in G^{'}\}$. L'idea degli S-polinomi è introdotta per trovare nuovi elementi nell'ideale, come f, che potrebbero aggiungere termini principali alla base intermedia $G^{'}$.

Definizione 4.20 (Minimo comune multiplo ed S-polinomio). Siano $f, g \in k[x_1, \ldots, x_n]$ polinomi non nulli.

(i) Se $multideg(f) = \alpha$ e $multideg(g) = \beta$, allora poniamo $\gamma = (\gamma_1, \ldots, \gamma_n)$, dove $\gamma_i = max(\alpha_i, \beta_i)$ per ogni i. Chiamiamo x^γ il *minimo comune multiplo* tra $LM(f)$ e $LM(g)$, $x^\gamma = mcm(LM(f), LM(g))$.

(ii) L'S-polinomio di f e g è la combinazione

$$S(f,g) = \frac{x^\gamma}{LT(f)} \cdot f - \frac{x^\gamma}{LT(g)} \cdot g$$

Esempio 4.8. Siano $f = x^3y^2 - x^2y^3 + x$ e $g = 3x^4y + y^2$ in $\mathbb{R}[x, y]$, con l'ordinamento $grlex$. Allora $\gamma = (4, 2)$ e

$$\begin{aligned} S(f,g) &= \frac{x^4y^2}{x^3y^2} \cdot f - \frac{x^4y^2}{3x^4y} \cdot g \\ &= x \cdot f - (1/3) \cdot y \cdot g \\ &= -x^3y^3 + x^2 - (1/3)y^3. \end{aligned}$$

Il nome S-polinomio è un'abbreviazione di polinomio di sigizia. In astronomia la parola sigizia è usata per l'allineamento di tre pianeti; quindi una sigizia allinea i termini principali di due polinomi. Il seguente lemma sarà utile per dimostrare la correttezza e la finitezza dell'algoritmo di Buchberger.

Lemma 4.17. *Supponiamo di avere una somma della forma* $\sum_{i=1}^m c_i x^{\alpha(i)} g_i$, *dove* $c_1, \ldots, c_m$ *sono costanti e* $\alpha(i) + multideg(g_i) = \delta \in \mathbb{Z}_+^n$, *quando* $c_i \neq 0$. *Se la*

somma è strettamente minore del multideg, cioé $multideg(\sum_{i=1}^{m} c_i x^{\alpha(i)} g_i) < \delta$, *allora esistono delle costanti* c_{jk} *tali che:*

$$\sum_{i=1}^{m} c_i x^{\alpha(i)} g_i = \sum_{j,k} c_{jk} x^{\delta-\gamma_{jk}} S(g_j, g_k),$$

dove $x^{\gamma_{jk}} = mcm(LM(g_j), LM(g_k))$. *Inoltre, si ha che ogni* $x^{\delta-\gamma_{jk}} S(g_j, g_k)$ *ha* $multideg < \delta$.

Dimostrazione. Sia $d_i = LC(g_i)$, così ché $c_i d_i$ è il coefficiente principale di $c_i x^{\alpha(i)} g_i$. Poiché $c_i x^{\alpha(i)} g_i$ ha multigrado δ e la loro somma ha, per ipotesi, multigrado strettamente minore di δ, segue che: $\sum_{i=1}^{m} c_i d_i = 0$.
Sia $p_i = x^{\alpha(i)} g_i / d_i$, osserviamo che, per come è stato definito, p_i ha coefficiente principale uguale ad 1. Consideriamo la somma telescopica:

$$\sum_{i=1}^{m} c_i x^{\alpha(i)} g_i = \sum_{i=1}^{m} c_i d_i p_i = c_1 d_1 (p_1 - p_2) + (c_1 d_1 + c_2 d_2)(p_2 - p_3) + \cdots$$

$$+(c_1 d_1 + \cdots + c_{m-1} d_{m-1})(p_{m-1} - p_m) + (c_1 d_1 + \cdots + c_m d_m) p_m.$$

Sia adesso $LT(g_i) = d_i x^{\beta(i)}$. Per ipotesi abbiamo che $\alpha(i) + \beta(i) = \delta$ per ogni i. Abbiamo quindi che $x^{\delta} = x^{\alpha(i)+\beta(i)} = x^{\alpha(i)} x^{\beta(i)}$ e ciò implica che $LM(g_i) = x^{\beta(i)}$ divide x^{δ}; di conseguenza anche $x^{\gamma_{jk}} = mcm(LM(g_j), LM(g_k))$ divide x^{δ}. Così, $x^{\delta-\gamma_{jk}}$ è un monomio, ed abbiamo

$$\begin{aligned} x^{\delta-\gamma_{jk}} S(g_j, g_k) &= x^{\delta-\gamma_{jk}} \left(\frac{x^{\gamma_{jk}}}{LT(g_j)} g_j - \frac{x^{\gamma_{jk}}}{LT(g_k)} g_k \right) \\ &= \frac{x^{\delta}}{d_j x^{\beta(j)}} g_j - \frac{x^{\delta}}{d_k x^{\beta(k)}} g_k \\ &= \frac{x^{\alpha(j)} g_j}{d_j} - \frac{x^{\alpha(k)} g_k}{d_k} \\ &= p_j - p_k. \end{aligned} \tag{4.1}$$

Usando questa equazione e $\sum_{i=1}^{m} c_i d_i = 0$, la serie telescopica di prima può essere scritta come:

$$\sum_{i=1}^{m} c_i x^{\alpha(i)} g_i = c_1 d_1 x^{\delta-\gamma_{12}} S(g_1, g_2) + (c_1 d_1 + c_2 d_2) x^{\delta-\gamma_{23}} S(g_2, g_3) + \cdots$$

$$+(c_1 d_1 + \cdots + c_{m-1} d_{m-1}) x^{\delta-\gamma_{m-1,m}} S(g_{m-1}, g_m),$$

che è la sommatoria scritta nell'espressione desiderata.
Poiché p_j e p_k hanno multigrado δ e coefficiente principle uguale ad 1, la differenza $p_j - p_k$ ha $multideg < \delta$. Dall'equazione 4.1 vediamo che lo stesso vale per $x^{\delta-\gamma_{jk}}S(g_j, g_k)$ e, così, il lemma è dimostrato. □

Basandosi sul lemma 4.17, Buchberger formulò il suo criterio sulle S-coppie, che permette di determinare effettivamente quando un insieme di generatori di un ideale è una base di Gröbner. Questo criterio conduce in modo naturale all'algoritmo di Buchberger.

Teorema 4.18 (Criterio di Buchbeger delle S-coppie). *Sia I un ideale polinomiale, allora una base $G = \{g_1, \ldots, g_m\}$ per I è una base di Gröbner per I se e soltanto se per tutte le coppie $i \neq j$, il resto della divisione di $S(g_i, g_j)$ per G (ordinata con un qualsiasi ordinamento) è zero, cioè $\overline{S(g_i, g_j)}^G = 0$.*

Dimostrazione. ($\Rightarrow$): Se G è una base di Gröbner, allora poiché $S(g_i, g_j) \in I$, il resto della divisione per G è zero per il corollario 4.16.
($\Leftarrow$): Sia $f \in I$ un polinomio diverso da zero. Dobbiamo dimostrare che se tutti gli S-polinomi hanno resto zero nella divisione per G allora:

$$LT(f) \in \langle LT(g_1), \ldots, LT(g_m)\rangle.$$

Prima di vedere la dimostrazione in dettaglio, vediamone la strategia.
Dato $f \in I = \langle g_1, \ldots, g_m\rangle$, ci sono polinomi $h_i \in k[x_1, \ldots, x_n]$ tali che

$$f = \sum_{i=1}^{m} h_i g_i. \tag{4.2}$$

Per il lemma 4.5 segue che

$$multideg(f) \leq max(multideg(h_i g_i)). \tag{4.3}$$

Se l'uguaglianza non compare, allora devono cancellarsi alcuni termini principali di 4.2. Il lemma 4.17 ci permette di riscrivere la 4.2 in termini di S-polinomi. Allora, la nostra ipotesi che gli S-polinomi hanno resto zero ci permetterà di rimpiazzare gli S-polinomi con espressioni in cui compaiano meno cancellazioni. Così, avremo un'espressione di f in cui compaiano meno cancellazioni di termini principali. Continuando con questo procedimento, troveremo alla fine un'espressione di f per la quale in 4.3 valga l'uguaglianza. Quindi $multideg(f) = multideg(h_i g_i)$ per qualche i, e seguirà che $LT(f)$ è divisibile per $LT(g_i)$. Questo dimostrerà che $LT(f) \in \langle LT(g_1), \ldots, LT(g_m)\rangle$, che è quello che vogliamo provare.
Diamo adesso i dettagli della dimostrazione. Data un'espressione di f come in

4.2, sia $t(i) = multideg(h_i g_i)$, e definiamo $\delta = max(t(1), \ldots, t(m))$; quindi la 4.3 diventa

$$multideg(f) \leq \delta.$$

Consideriamo, adesso, tutti i modi possibili in cui f può essere scritta nella forma 4.2. Per ognuna di tali espressioni abbiamo un possibile differente δ. Poiché l'ordinamento monomiale è un buon ordinamento, possiamo scegliere un'espressione per f tale che δ sia minimale. Mostreremo che, una volta scelto questo δ minimale, abbiamo $multideg(f) = \delta$. Allora in 4.3 compare l'uguaglianza e, come abbiamo osservato, segue che $LT(f) \in \langle LT(g_1), \ldots, LT(g_m)\rangle$.

Rimane, dunque, da dimostrare che $multideg(f) = \delta$. Lo dimostreremo per assurdo. L'uguaglianza non si verifica soltanto quando $multideg(f) < \delta$. Per isolare i termini di multigrado δ, scriviamo f nella seguente forma:

$$\begin{aligned} f &= \sum_{t(i)=\delta} h_i g_i + \sum_{t(i)<\delta} h_i g_i \\ &= \sum_{t(i)=\delta} LT(h_i) g_i + \sum_{t(i)=\delta} (h_i - LT(h_i)) g_i + \sum_{t(i)<\delta} h_i g_i. \end{aligned} \tag{4.4}$$

I monomi che compaiono nella seconda e terza sommatoria della seconda linea, hanno tutti $multideg < \delta$; quindi l'aver ipotizzato che $multideg(f) < \delta$ implica che anche la prima sommatoria ha $multideg < \delta$.

Poniamo $LT(h_i) = c_i x^{\alpha(i)}$; allora la prima sommatoria

$$\sum_{t(i)=\delta} LT(h_i) g_i = \sum_{t(i)=\delta} c_i x^{\alpha(i)} g_i$$

ha la stessa forma descritta nel lemma 4.17, poiché i termini $c_i x^{\alpha(i)} g_i$ hanno $multideg$ uguale a δ e la loro somma ha $multideg$ strettamente minore di δ, perché ipotizzato da noi. Quindi, visto che le ipotesi del lemma 4.17 sono soddisfatte, quest'ultimo implica

$$\sum_{t(i)=\delta} LT(h_i) g_i = \sum_{j,k} c_{jk} x^{\delta - \gamma_{jk}} S(g_j, g_k), \tag{4.5}$$

dove $c_{jk} \in k$ e $x^{\gamma_{jk}} = mcm(LM(g_j), LM(g_k))$.
Il prossimo passo è quello di usare l'ipotesi che il resto della divisione di $S(g_j, g_k)$ per $g_1, \ldots, g_m$ è zero. Usando l'algoritmo della divisione, questo vuol dire che ogni S-polinomio può essre scritto nella seguente forma

$$S(g_j, g_k) = \sum_{i=1}^{m} a_{ijk} g_i, \tag{4.6}$$

dove $a_{ijk} \in k[x_1, \ldots, x_n]$. L'algoritmo della divisione ci dice anche che

$$multideg(a_{ijk}g_i) \leq multideg(S(g_j, g_k)), \tag{4.7}$$

per ogni i, j, k (vedi teorema 4.6). Intuitivamente questo vuol dire che quando il resto è zero, possiamo trovare un'espressione di $S(g_j, g_k)$ in termini di G dove i termini principali non si possono tutti cancellare.
Per sfruttare questo risultato moltiplichiamo l'equazione 4.6 per $x^{\delta-\gamma_{jk}}$ e otteniamo

$$x^{\delta-\gamma_{jk}}S(g_j, g_k) = \sum_{i=1}^{m} b_{ijk}g_i,$$

dove $b_{ijk} = a_{ijk}x^{\delta-\gamma_{jk}}$. Allora 4.7 e il lemma 4.17 implicano che

$$multideg(b_{ijk}g_i) \leq multideg(x^{\delta-\gamma_{jk}}S(g_j, g_k)) < \delta. \tag{4.8}$$

Se sostituiamo l'espressione di $x^{\delta-\gamma_{jk}}S(g_j, g_k)$ nell'equazione 4.5, otteniamo

$$\begin{aligned}\sum_{t(i)=\delta} LT(h_i)g_i &= \sum_{j,k} c_{jk}x^{\delta-\gamma_{jk}}S(g_j, g_k) \\ &= \sum_{j,k} c_{jk}\left(\sum_i b_{ijk}g_i\right) = \sum_i \left(\sum_{j,k} c_{jk}b_{ijk}\right) g_i.\end{aligned}$$

Riscriviamo l'ultima sommatoria come $\sum_i \tilde{h}_i g_i$, segue dall'equazione 4.8 che

$$multideg(\tilde{h}_i g_i) < \delta$$

perché le c_{jk} sono costanti.
Come passo finale della dimostrazione, sotituiamo $\sum_{t(i)=\delta} LT(h_i)g_i = \sum_i \tilde{h}_i g_i$ nell'equazione 4.4 per ottenere

$$f = \sum_i \tilde{h}_i g_i + \sum_{t(i)=\delta} (h_i - LT(h_i))g_i + \sum_{t(i)<\delta} h_i g_i.$$

Abbiamo espresso f come una combinazione polinomiale di g_i, dove tutti i termini hanno $multideg < \delta$. Questo contraddice la minimalità di δ e completa quindi la dimostrazione del teorema. □

Questo criterio è formulato in maniera più rigida in [**BW93**], ed è usato in quest'ultima forma in [**Fau99**] e [**Fau02**]. Per questo motivo includiamo il criterio in questa forma differente. Per fare ciò dobbiamo dare alcune definizioni.

Definizione 4.21. [**BW93**, p.218] Sia $f \in k[x_1, \ldots, x_n]$ diverso da zero e $G = \{g_1, \ldots, g_m\}$ un sottoinsieme finito di $k[x_1, \ldots, x_n]$. Una rappresentazione

$$f = \sum_{i=1}^{m} b_i g_i$$

con monomi $b_i \in k[x_1, \ldots, x_n]$ è chiamata una *rappresentazione standard* di f rispetto a G se

$$max_i(LT(b_i g_i)) \leq LT(f).$$

Diciamo che c'è un *t-rappresentazione* di f rispetto a G se

$$max_i(LT(b_i g_i)) \leq t.$$

dove t è un termine nelle variabili $x_1, \ldots, x_n$. Definiamo un *termine* t, nelle variabili $x_1, \ldots, x_n$, come il prodotto di potenze della forma: $x_1^{e_1} \cdots x_n^{e_n}$, con $e_i \in \mathbb{N}$ per $i = 1, \ldots, n$. Denotiamo con $T(x_1, \ldots, x_n)$ o, piú semplicemente, con T l'insieme di tutti i termini in queste variabili.

Teorema 4.19. *Sia G un sottoinsieme finito di $k[x_1, \ldots, x_n]$, con $0 \notin G$. Supponiamo che per ogni $g_1, g_2 \in G$, $S(g_1, g_2)$ o è uguale a zero oppure ha una t-rappresentazione rispetto a G, per qualche $t < mcm(LT(g_1), LT(g_2))$, allora G è una base di Gröbner.*

Dimostrazione. Vedi [**BW93**, p.219]. □

Il criterio dato nel teorema 4.18 è molto utile perché fornisce un algoritmo per verificare quando una base è una base di Gröbner. Per esempio, consideriamo l'ideale $I = \langle y - x^2, z - x^3 \rangle$. Verifichiamo che $G = \{y - x^2, z - x^3\}$ è una base di Gröbner rispetto all'ordinamento lessicografico con $y > z > x$. Per fare ciò utilizziamo il teorema 4.18. L'unico S-polinomio che dobbiamo verificare è

$$S(y - x^2, z - x^3) = \frac{yz}{y}(y - x^2) - \frac{yz}{z}(z - x^3) = -zx^2 + yx^3.$$

Usando l'algoritmo della divisione, si trova che

$$-zx^2 + yx^3 = x^3(y - x^2) + (-x^2) \cdot (z - x^3) + 0,$$

così che $\overline{S(y - x^2, z - x^3)}^G = 0$. G è allora una base di Gröbner per I. Si può verificare che, usando l'ordinamento lessicografico con $x > y > z$, avremmo trovato che G non è una base di Gröbner per I.

Siamo pronti adesso ad enunciare il teorema che descrive l'algoritmo di Buchberger.

Teorema 4.20 (Algoritmo di Buchberger). *Sia $I = \langle f_1, \ldots, f_s \rangle \neq 0$ un ideale polinomiale, allora si può costruire una base di Gröbner per I in un numero finito di passi con il seguente algoritmo:*

input: $F = (f_1, \ldots, f_s)$
output: una base di Gröbner $G = (g_1, \ldots, g_t)$ per I, con $F \subset G$.

$G := F$
repeat
  $G' := G$
  for ogni coppia $\{p, q\}, p \neq q$ in G' **do**
    $S := \overline{S(p,q)}^{G'}$
    if $S \neq 0$ **then** $G := G \cup \{S\}$
until $G := G'$.

Dimostrazione. Diamo prima uno schema della dimostrazione.

(i) Ad ogni stadio dell'algoritmo, $G \subset I$ e vale $\langle G \rangle = I$.

(ii) Se $G = G'$, allora $\overline{S(p,q)}^{G'} = 0$ per ogni $p, q \in G$, e per il criterio di Buchberger delle S-coppie, allora G è una base di Gröbner.

(iii) L'uguaglianza $G = G'$ si verifica in un numero finito di passi, poiché gli ideali $\langle LT(G') \rangle$, formati dalle iterazioni successive del loop principale, formano una catena crescente. Per la condizione della catena crescente, questa catena di ideali deve stabilizzarsi dopo un numero finito di iterazioni e quindi $\langle LT(G) \rangle = \langle LT(G') \rangle$.

Prima dimostriamo che vale $G \subset I$ ad ogni stadio dell'algoritmo. Questo è vero all'inizio, e ogni qual volta espandiamo G, lo facciamo aggiungendo il resto $S = \overline{S(p,q)}^{G'}$, per $p, q \in G$. Così, se $G \subset I$, allora p, q e quindi anche $S(p,q)$ sono in I, e poiché stiamo dividendo per $G' \subset I$, abbiamo $G \cup \{S\} \subset I$. Osserviamo, inoltre, che G contiene anche la base data F di I, così che G è effettivamente una base di I.
L'algoritmo termina quando $G = G'$, il che vuol dire $\overline{S(p,q)}^{G} = 0$ per ogni $p, q \in G$. G è una base di Gröbner di $\langle G \rangle = I$, per il teorema 4.18.
Rimane da dimostrare che l'algoritmo termina. Abbiamo bisogno di considerare cosa accade dopo ogni passaggio attraverso il loop principale. L'insieme G è costituito da G' (il vecchio G) e da i resti, diversi da zero, degli S-polinomi degli elementi di G'. Allora

$$\langle LT(G') \rangle \subset \langle LT(G) \rangle \tag{4.9}$$

poiché $G' \subset G$. In ogni modo, se $G' \neq G$, affermiamo che $\langle LT(G') \rangle$ è strettamente più piccolo di $\langle LT(G) \rangle$. Per verificare ciò supponiamo che un resto r, diverso da

zero, di un S-polinomio è stato aggiunto a G. Poiché r è un resto della divisione per $G^{'}$, $LT(r)$ non è divisibile dai termini principali degli elementi di $G^{'}$ e, così, $LT(r) \notin \langle LT(G^{'})\rangle$. Ancora $LT(r) \in \langle LT(G)\rangle$, che prova la nostra affermazione.
Dalla 4.9 segue che gli ideali $\langle LT(G^{'})\rangle$, provenienti dalle iterazioni successive del loop, formano una catena crescente di ideali in $k[x_1, \ldots, x_n]$. Così la condizione della catena crescente (teorema 4.13) implica che, dopo un numero finito di iterazioni, la catena si stabilizzerà, così che $\langle LT(G^{'})\rangle = \langle LT(G)\rangle$ alla fine si verifica. Da quanto detto sopra abbiamo che $G^{'} = G$, così che l'algoritmo deve terminare dopo un numero finito di passi. □

Diamo un esempio per vedere come lavora l'algoritmo di Buchberger.

Esempio 4.9. Consideriamo $k[x, y]$, con l'ordinamento $grlex$, e sia $I = \langle f_1, f_2\rangle = \langle x^3 - 2xy, x^2y - 2y^2 + x\rangle$. Applichiamo l'algoritmo:
$G^{'} = (x^3 - 2xy, x^2y - 2y^2 + x)$, calcolo $S = \overline{S(x^3 - 2xy, x^2y - 2y^2 + x)}^{G^{'}}$:

$$\begin{aligned}
&S(x^3 - 2xy, x^2y - 2y^2 + x) = \\
&= \frac{mcm(x^3, x^2y)}{x^3} \cdot (x^3 - 2xy) - \frac{mcm(x^3, x^2y)}{x^2y} \cdot (x^2y - 2y^2 + x) \\
&= \frac{x^3y}{x^3} \cdot (x^3 - 2xy) - \frac{x^3y}{x^2y} \cdot (x^2y - 2y^2 + x) \\
&= y \cdot (x^3 - 2xy) - x \cdot (x^2y - 2y^2 + x) \\
&= x^3y - 2xy^2 - x^3y + 2xy^2 - x^2 \\
&= -x^2
\end{aligned}$$

ed il resto della divisione di $S(f_1, f_2) = -x^2$ per $G^{'}$ è $-x^2$ che è diverso da zero, quindi $S \neq 0$ implica $G = G \cup \{S\} = (f_1, f_2, f_3)$ dove $f_3 = -x^2$.
Adesso l'insieme candidato ad essere la base di Gröbner è $G = (f_1, f_2, f_3)$. Dobbiamo calcolare:

$$\begin{aligned}
&S(f_1, f_2) = f_3, \\
&\overline{S(f_1, f_2)}^{G^{'}} = 0, \\
&S(f_1, f_3) = (x^3 - 2xy) - (-x)(-x^2) = -2xy, \text{ma} \\
&\overline{S(f_1, f_3)}^{G^{'}} = -2xy \neq 0.
\end{aligned}$$

Dobbiamo, quindi, aggiungere $f_4 = -2xy$ al nostro insieme di generatori. Abbiamo

allora $G = (f_1, f_2, f_3, f_4)$:

$$\begin{aligned}
&\overline{S(f_1, f_2)}^{G'} = \overline{S(f_1, f_3)}^{G'} = 0,\\
&S(f_1, f_4) = y(x^3 - 2xy) - (-1/2)x^2(-2xy) = -2xy^2 = yf_4, \text{ così}\\
&\overline{S(f_1, f_4)}^{G'} = 0,\\
&S(f_2, f_3) = (x^2y - 2y^2 + x) - (-y)(-x^2) = -2y^2 + x, \text{ ma}\\
&\overline{S(f_2, f_3)}^{G'} = -2y^2 + x \neq 0.
\end{aligned}$$

Così dobbiamo aggiungere anche $f_5 = -2y^2 + x$ al nostro insieme di generatori. Posto $G = (f_1, f_2, f_3, f_4, f_5)$, facendo i calcoli otteniamo:

$$\overline{S(f_i, f_j)}^{G'} = 0, \text{ per ogni } 1 \leq i < j \leq 5.$$

Così l'algoritmo termina e abbiamo trovato che una base di Gröbner per I è:

$$\{f_1, f_2, f_3, f_4, f_5\} = \{x^3 - 2xy, x^2y - 2y^2 + x, -x^2, -2xy, -2y^2 + x\}.$$

Osserviamo che l'algoritmo usato nel precedente teorema non costituisce un modo molto pratico per il calcolo di una base di Gröbner. Notiamo (come primo miglioramento) che una volta che un resto $\overline{S(p, q)}^{G'}$ è uguale a zero, il resto rimarrà zero anche se aggiungiamo ulteriori elementi alla fine dell'insieme di generatori ordinato G'. Non ci sono, allora, ragioni per ricalcolare questi resti nel passo successivo attraverso il loop principale. A dire il vero, se aggiungiamo i nostri generatori f_j uno alla volta, gli unici resti che dobbiamo calcolare sono $\overline{S(f_i, f_j)}^{G'}$, dove $i \leq j-1$. Questo algoritmo è il primo di una famiglia di algoritmi per il calcolo delle basi di Gröbner. Più avanti introdurremo altri algoritmi per questo scopo. Molti miglioramenti dell'algoritmo di Buchberger riguardano l'ordine con il quale scegliere e ridurre le coppie critiche, che spesso è chiamata *strategia di selezione*. Altri miglioramenti, invece, riguardano i così chiamati *criteri* per evitare le coppie critiche che si ridurrebbero a zero rispetto alla base intermedia G'. L'algoritmo $F4$ è un esempio di algoritmo con una strategia di selezione migliorata, che sarà trattata nel capitolo 6.

4.3.1 Miglioramenti dell'algoritmo di Buchberger

Le basi di Gröbner, calcolate usando l'algoritmo del teorema 4.20, sono spesso più grandi del necessario. Possiamo eliminare alcuni generatori non necessari utilizzando il seguente criterio.

Lemma 4.21. *Sia G una base di Gröbner per l'ideale polinomiale I. Sia $p \in G$ un polinomio tale che $LT(p) \in \langle LT(G - \{p\})\rangle$, allora anche $G - \{p\}$ è una base di Gröbner per I.*

Dimostrazione. Sappiamo che $\langle LT(G)\rangle = \langle LT(I)\rangle$. Se $LT(p) \in \langle LT(G - \{p\})\rangle$, allora $\langle LT(G - \{p\})\rangle = \langle LT(G)\rangle$. Dalla definizione segue che $G - \{p\}$ è una base di Gröbner per I. □

Trasformando le costanti al fine di rendere tutti i coefficienti principali uguali ad 1 e rimuovendo ogni p con $LT(p) \in \langle LT(G - \{p\})\rangle$ da G, arriviamo a quello che chiameremo una base *minimale* di Gröbner.

Definizione 4.22 (Base di Gröbner minimale). Una base di Gröbner minimale per un ideale polinomiale I è una base di Gröbner G per I tale che:

(i) $LC(p) = 1$ per ogni $p \in G$,

(ii) per ogni $p \in G$, $LT(p) \notin \langle LT(G - \{p\})\rangle$.

Definizione 4.23 (Base di Gröbner ridotta). Una *base di Gröbner ridotta* per un ideale polinomiale I è una base di Gröbner G per I tale che:

(i) $LC(p) = 1$, per ogni $p \in G$;

(ii) per ogni $p \in G$, nessun monomio di p sta in $\langle LT(G - \{p\})\rangle$.

Le basi di Gröbner ridotte hanno la seguente importante propietà.

Proposizione 4.22 (Unicità delle basi di Gröbner ridotte). *Sia $I \neq 0$ un ideale polinomiale, allora, per un dato ordinamento monomiale, I ha un'unica base di Gröbner ridotta.*

Dimostrazione. Vedi [**CLO92**, p.91]. □

C'è un'altro criterio per verificare se un insieme di generatori è una base di Gröbner. Questo criterio, per esempio, è usato nella dimostrazione della correttezza dell'algoritmo $F4$. Prima di formulare il criterio abbiamo bisogno di dare la seguente definizione.

Definizione 4.24. Fissato un ordinamento monomiale e dato $G = \{g_1, \ldots, g_m\} \subset k[x_1, \ldots, x_n]$. Dato un polinomio $f \in k[x_1, \ldots, x_n]$, diciamo che f è *ridotto a zero modulo* G, e scriviamo

$$f \to_G 0$$

se f può essere scritto nella forma

$$f = a_1 g_1 + \cdots + a_m g_m,$$

tale che ogni qualvolta $a_i g_i \neq 0$, si ha

$$multideg(f) \geq multideg(a_i g_i).$$

Definizione 4.25 (Forma normale). La *forma normale* r di f rispetto a G è la chiusura riflessiva-transitiva della riduzione per gli elementi di G, $\to_G$. Questo vuol dire che, se esiste una catena di riduzioni da F in r con la divisione per elementi di G ed r non può essere ridotto da nessun altro elemento di G (rispetto a qualche ordinamento monomiale), questo r è chiamato la forma normale di f rispetto a G.

Per capire la relazione che c'è tra la definizione 4.24 e l'algoritmo della divisione diamo il seguente lemma.

Lemma 4.23. *Sia $G = \{g_1, \ldots, g_m\}$ un'insieme ordinato di elementi di $k[x_1, \ldots, x_n]$, fissiamo $f \in k[x_1, \ldots, x_n]$. Allora $\overline{f}^G = 0$ implica $f \to_G 0$, benché il contrario in generale è falso.*

Dimostrazione. Se $\overline{f}^G = 0$, allora dall'algoritmo della divisione si ha:

$$f = a_1 g_1 + \cdots + a_m g_m + 0,$$

e dal teorema 4.6, ogni qualvolta $a_i g_i \neq 0$, abbiamo:

$$multideg(f) \geq multideg(a_i g_i).$$

Questo dimostra che $f \to_G 0$. Per vedere che il contrario non sempre è vero, diamo un controesempio.
Siano $f_1 = xy + 1$ ed $f_2 = y^2 - 1 \in k[x, y]$, con l'ordinamento lessicografico. Dividendo $f = xy^2 - x$ per $G = (f_1, f_2)$, otteniamo:

$$xy^2 - x = y \cdot (xy + 1) + 0 \cdot (y^2 - 1) + (-x - y),$$

così che $\overline{f}^g = -x - y \neq 0$; ma se dividiamo per $G = (f_2, f_1)$, otteniamo:

$$xy^2 - x = 0 \cdot (xy + 1) + x \cdot (y^2 - 1),$$

e poiché $multideg(xy^2 - x) \geq multideg(x \cdot (y^2 - 1))$, segue che $f \to_G 0$. □

Come esempio per vedere in che modo la definizione 4.24 può essere usata, enunciamo una versione più generale del criterio per la basi di Gröbner 4.18.

Teorema 4.24. *Una base $G = \{g_1, \ldots, g_m\}$ per un ideale I è una base di Gröbner se e soltanto se $S(g_i, g_j) \to_G 0$ per ogni $i \neq j$.*

Dimostrazione. Nel teorema 4.18 abbiamo provato questo risultato sotto l'ipotesi che $\overline{S(g_i, g_j)}^G = 0$ per ogni $i \neq j$. Se riesaminiamo la dimostrazione, vedremo che avevamo usato

$$S(g_j, g_k) = \sum_{i=1}^{m} a_{ijk} g_i,$$

dove $multideg(a_{ijk} g_i) \leq multideg(S(g_j, g_k))$ [vedi le equazioni 4.6 e 4.7]. Questo è esattamente quello che significa $S(g_i, g_j) \to_G 0$, e quindi il teorema segue. □

Grazie al lemma 4.23 possiamo notare che il teorema 4.18 è un caso speciale del teorema 4.24. Mostriamo adesso che è sicuro che alcuni S-polinomi si riducano a zero.

Proposizione 4.25 (Secondo criterio di Buchberger). *Dato un insieme finito $G \subset k[x_1, \ldots, x_n]$, supponiamo di avere $f, g \in G$ tali che:*

$$mcm(LM(f), LM(g)) = LM(f) \cdot LM(g)$$

cioè i monomi principali di f e g sono coprimi, allora

$$S(f, g) \to_G 0.$$

Dimostrazione. Per semplicità supponiamo che f e g siano moltiplicati per appropiate costanti tali che $LC(f) = LC(g) = 1$. Scriviamo $f = LM(f) + p$ e $g = LM(g) + q$, allora, poiché $mcm(LM(f), LM(g)) = LM(f) \cdot LM(g)$, abbiamo:

$$\begin{aligned} S(f, g) &= LM(g) \cdot f - LM(f) \cdot g \\ &= (g - q) \cdot f - (f - p) \cdot g \\ &= g \cdot f - q \cdot f - f \cdot g + p \cdot g \\ &= p \cdot g - q \cdot f. \end{aligned} \tag{4.10}$$

Affermiamo che:

$$multideg(S(f, g)) = max(multideg(p \cdot g), multideg(q \cdot f)). \tag{4.11}$$

Osserviamo che 4.10 e 4.11 implicano $S(f, g) \to_G 0$, poiché $f, g \in G$. Rimane da dimostrare la 4.11; a tal fine, osserviamo che, nell'ultimo polinomio di 4.10, i monomi principali di $p \cdot g$ e $q \cdot f$ sono distinti e, quindi, non si possono cancellare tra loro. Se i monomi principali fossero gli stessi, allora avremmo:

$$LM(p) \cdot LM(g) = LM(q) \cdot LM(f),$$

ma questo è impossibile perché per ipotesi $mcm(LM(f), LM(g)) = 1$, infatti dall'ultima equazione avremmo che $LM(g)$ dovrebbe dividere $LM(q)$, che è assurdo perché $LM(g) > LM(q)$.

□

Diamo un esempio di come funziona questa proposizione. Sia $G = (yz + y, x^3 + y, z^4)$, in $k[x, y, z]$, usiamo l'ordinamento $grlex$, allora:

$$S(x^3 + y, z^4) \to_G 0,$$

per la proposizione 4.25. Usando l'algoritmo della divisione si verifica facilmente che vale:

$$S(x^3 + y, z^4) = yz^4 = (z^3 - z^2 + z - 1)(yz + y) + y,$$

così che:

$$\overline{S(x^3+y,z^4)}^G = y \neq 0.$$

Questo spiega il bisogno della definizione 4.24: la proposizione 4.25 è falsa se usiamo la nozione di resto nullo proveniente dall'algoritmo della divisione.

4.3.2 Osservazioni sui miglioramenti dell'algoritmo di Buchberger

Diamo alcune osservazioni riguardanti i miglioramenti dell'algoritmo di Buchberger.

1. Nell'algoritmo di Bucheberger se un S-polinomio viene ridotto a zero, l'algoritmo continerà a calcolare il suo resto (che già sappiamo essere zero) in ogni iterazione del loop successiva. Ma questo non è necessario che avvenga, poiché, se un S-polinomio ha resto zero nella divisione per G, tale rimarrà il resto, anche se successivamente in G aggiungo nuovi polinomi.

2. Criterio di Buchberger I [**Buch79**]: scegliamo una coppia $\{f_i, f_j\}$ tale che il minimo comune multiplo tra $LM(f_i)$ e $LM(f_j)$ sia minimo rispetto a quello di tutte le altre coppie.

3. Criterio di Buchberger II [**Buch79**]: ci sono degli S-polinomi che non è necessario che siano calcolati, infatti, grazie alla proposizione 4.25, possiamo subito vedere se si riducono a zero modulo G; se $LM(f_i)$ e $LM(f_j)$ sono coprimi allora $S(f_i, f_j)$ si riduce a zero modulo G. Quindi bisogna scegliere coppie $\{f_i, f_j\}$ tali che $mcm(LM(f_i), LM(f_j)) \neq 1$.

4. Criterio di Buchberger III [**Buch79**]: se esiste un elemento f_k della base tale che $LM(f_k)$ divide $mcm(LM(f_i), LM(f_j))$ e se gli S-polinomi $S(f_i, f_k)$ e $S(f_j, f_k)$ sono già stati considerati, allora $S(f_i, f_j)$ si riduce a zero e quindi può essere ignorato.

5. La base di Gröbner G, trovata dall'algoritmo di Buchberger, non è una base di Gröbner ridotta. Ci sono polinomi *ridondanti* che possono essere eliminati: $f_i \in G$, allora f_i è detto *ridondante* se $G - \{f_i\}$ è ancora una base di Gröbner e $\langle G \rangle = \langle G - \{f_i\} \rangle$. Se $LT(f_j)$ divide $LT(f_i)$ allora f_i può essere eliminato dalla base e $G - \{f_i\}$ è ancora una base di Gröbner. Così, ogni volta che viene aggiunto alla base intermedia un nuovo polinomio, tutti gli altri polinomi possono essere ridotti usando anche il nuovo polinomio; questo comporta la cancellazione di parecchi polinomi e il risultato è una base di Gröbner ridotta. (Vedi lemma 4.21)

Capitolo 5

Teoria di eliminazione

In questo capitolo studieremo metodi sistematici per eliminare delle variabili da sistemi di equazioni polinomiali. La strategia di base della teoria di eliminazione è data in due teoremi principali: il *teorema di eliminazione* e il *teorema di estensione*.

5.1 Il problema di risolvere equazioni polinomiali

Vogliamo vedere come è possibile applicare la tecnica delle basi di Gröbner per risolvere sistemi di equazioni polinomiali multivariate. Diamo un esempio specifico per vedere come cambiano le equazioni che costituiscono il sistema, quando le sostituiamo con una base di Gröbner.

Esempio 5.1. Consideriamo le equazioni:

$$\begin{aligned} x^2 + y^2 + z^2 &= 1 \\ x^2 + z^2 &= y \\ x &= z \end{aligned} \tag{5.1}$$

in $\mathbb{C}^3$. Queste equazioni generano l'ideale

$$I = \langle x^2 + y^2 + z^2 - 1; x^2 + z^2 - y; x - z \rangle \subset \mathbb{C}[x, y, z]$$

Vogliamo trovare tutti i punti in $\mathbf{V}(I)$. La proposizione 4.14 implica che possiamo calcolare i punti di $\mathbf{V}(I)$ usando una qualsiasi base per I. Vediamo cosa succede quando usiamo una base di Gröbner.

Calcoliamo una base di Gröbner, rispetto all'ordinamento lessicografico, con $x > y > z$. La base è data da:

$$\begin{aligned} g_1 &= x - z, \\ g_2 &= -y + 2z^2, \\ g_3 &= z^4 + (1/2)z^2 - 1/4. \end{aligned}$$

Se osserviamo attentamente questi polinomi ci accorgiamo che g_3 dipende solo dalla variabile z e, quindi, può essere risolta semplicemente. Le soluzioni sono:

$$z = \pm\frac{1}{2}\sqrt{\pm\sqrt{5} - 1}$$

Abbiamo quattro valori di z che possono essere sostituiti nell'equazione $g_2 = 0$ e $g_1 = 0$, così avremo che g_2 e g_1 dipendono, rispettivamente, da y e x. Quindi ci sono quattro soluzioni di $g_1 = g_2 = g_3 = 0$, due reali e due complesse.

Poiché $\mathbf{V}(I) = \mathbf{V}(g_1, g_2, g_3)$ per la proposizione 4.14, abbiamo trovato tutte le soluzioni del sitema originale 5.1

Attraverso questo esempio, vediamo che trovare una base di Gröbner, per un ideale rispetto all'ordinamento lessicografico, semplifica considerevolmente la forma delle equazioni. In particolare, sembra che si abbiano equazioni in cui le variabili vengano eliminate in modo successivo. Notiamo, inoltre, che l'ordine in cui vengono eliminate le variabili sembra corrispondere all'ordinamento delle stesse; infatti avevamo $x > y > z$ e, riguardando la forma della base di Gröbner trovata, vediamo che la x viene eliminata per prima, la y per seconda e così via.

Un sistema di equazioni in questa forma è facile da risolvere, specialmente quando nell'ultima equazione compare solo una variabile. Possiamo applicare le tecniche per una variabile al fine di trovare le sue radici e sostituirle nelle restanti equazioni del sistema iterando, poi, questo procedimento per le restanti equazioni.

Notiamo un'analogia tra questo metodo e il metodo delle sostituzioni successive, adoperato per risolvere un sistema lineare in forma triangolare.

5.2 I teoremi di Eliminazione ed Estensione

In questa sezione studiamo i processi di eliminazione delle variabili da un sistema di equazioni polinomiali, in particolare vediamo perché con l'ordinamento lessicografico troviamo una base di Gröbner in cui le variabili sono eliminate in modo successivo.

Per vedere in che modo funziona il processo di eliminazione, diamo un esempio simile a quello precedente.

Esempio 5.2. Vogliamo risolvere il sistema di equazioni:

$$\begin{aligned} x^2 + y + z &= 1 \\ x + y^2 + z &= 1 \\ x + y + z^2 &= 1 \end{aligned} \tag{5.2}$$

Sia I l'ideale:

$$I = \langle x^2 + y + z - 1; x + y^2 + z - 1; x + y + z^2 - 1\rangle. \tag{5.3}$$

allora una base di Gröbner per I, rispetto all'ordinamento lessicografico, è data dai seguenti quattro polinomi:

$$\begin{aligned} g_1 &= x + y + z^2 - 1 \\ g_2 &= y^2 - y - z^2 + z \\ g_3 &= 2yz^2 + z^4 - z^2 \\ g_4 &= z^6 - 4z^4 + 4z^3 - z^2 \end{aligned} \tag{5.4}$$

Sappiamo che 5.2 e 5.4 hanno le stesse soluzioni. Poiché in

$$g_4 = z^6 - 4z^4 + 4z^3 - z^2 = z^2(z-1)^2(z^2 + 2z - 1)$$

compare solo z, troviamo che i valori per cui $g_4 = 0$ sono $0, 1$ e $-1 \pm \sqrt{2}$. Sostituendo questi valori in $g_2 = y^2 - y - z^2 + z = 0$ e $g_3 = 2yz^2 + z^4 - z^2 = 0$, possiamo determinare i valori possibili di y e, alla fine, da $g_1 = x + y + z^2 - 1 = 0$, abbiamo i valori corrispondenti di x. È possibile verificare che il sistema 5.2 ha esattamente cinque soluzioni:

$$\begin{aligned} &(1, 0, 0),\ (0, 1, 0),\ (0, 0, 1), \\ &(-1 + \sqrt{2}, -1 + \sqrt{2}, -1 + \sqrt{2}), \\ &(-1 - \sqrt{2}, -1 - \sqrt{2}, -1 - \sqrt{2}). \end{aligned}$$

Ci sono due cose che permettono di trovare le soluzioni con questa tecnica.

- (*Passo di eliminazione*) Dalle equazioni iniziali troviamo

  $$g_4 = z^6 - 4z^4 + 4z^3 - z^2 = 0$$

  che coinvolge soltanto le variabili z, cioè, eliminiamo le x e le y dal sistema di equazioni.

- (*Passo di estensione*) Risolviamo la semplice equazione $g_4 = 0$ per determinare i valori di z; possiamo estendere queste soluzioni alle soluzioni delle equazioni originali.

Per vedere come funziona il passo di eliminazione, notiamo che l' osservazione che abbiamo fatto, riguardo g_4 nell'esempio precedente, può essere scritta:

$$g_4 \in I \cap \mathbb{C}[z],$$

dove I è l'ideale definito nell'equazione 5.3; infatti, $I \cap \mathbb{C}[z]$ è costituito da tutte le equazioni, derivate dal sistema originale, nelle quali sono state eliminate le variabili x e y.

La generalizzazione di questo ragionamento conduce alla seguente definizione.

Definizione 5.1 (Ideale d'eliminazione). Dato $I = \langle f_1, \ldots, f_m \rangle \subset k[x_1, \ldots, x_n]$, il k-esimo *ideale d'eliminazione* I_k è l'ideale di $k[x_{k+1}, \ldots, x_n]$ definito da

$$I_k = I \cap k[x_{k+1}, \ldots, x_n].$$

L'ideale I_k è costituito da tutte le equazioni che derivano da

$$f_1 = f_2 = \cdots = f_m = 0,$$

nelle quali sono state eliminate le variabili $x_1, \ldots, x_k$. È possibile verificare che I_k è un'ideale di $k[x_{k+1}, \ldots, x_n]$. Osserviamo che $I = I_0$ è lo 0-simo ideale di eliminazione e che ordinamenti differenti delle variabili portano a differenti ideali d'eliminazione.

Usando questa nuova definzione, possiamo vedere che eliminare le variabili $x_1, \ldots, x_k$ vuol dire trovare polinomi, diversi da zero, che stanno nel k-esimo ideale di eliminazione I_k; così, una soluzione del passo di eliminazione vuol dire trovare un procedimento sistematico per determinare elementi di I_k. Con un appropriato ordinamento sui termini , le basi di Gröbner ci permettono di far ciò immediatamente.

Teorema 5.1 (Teorema di eliminazione). [**CLO92**, p.114]
Sia $I \subset k[x_1, \ldots, x_n]$ un ideale e sia G una base di Gröbner di I rispetto all'ordinamento lessicografico, dove $x_1 > x_2 > \cdots > x_n$, allora, per ogni $0 \leq k \leq n$, l'insieme:

$$G_k = G \cap k[x_{k+1}, \ldots, x_n]$$

è una base di Gröbner del k-esimo ideale d'eliminazione I_k.

Dimostrazione. Fissiamo un intero k, tra 0 ed n, e supponiamo che $G = \{g_1, \ldots, g_m\}$. Possiamo assumere, rinumerando se necessario, $G_k = \{g_1, \ldots, g_r\}$; cioè, esattamente i primi r elementi di G stanno in $k[x_{k+1}, \ldots, x_n]$.

Dimostriamo che G_k è una base di I_k. Poiché $G_k \subset I_k$ e I_k è un ideale, abbiamo che $\langle g_1, \ldots, g_r \rangle \subset I_k$; così dobbiamo dimostrare che ogni elemento f di I_k può essere scritto come:

$$f = h_1 g_1 + \cdots + h_r g_r$$

con $h_i \in k[x_{k+1}, \ldots, x_n]$. Questo può essere verificato usando l'algoritmo della divisione; usando l'ordinamento lessicografico dividiamo f per $g_1, \ldots, g_m$ e osserviamo che:

(i) poiché $G = \{g_1, \ldots, g_m\}$ è una base di Gröbner di I e $f \in I$, il resto della divisione di f per G è zero (per il corollario 4.16);

(ii) poiché usiamo l'ordinamento lessicografico, i termini principali di $g_{r+1}, \ldots, g_m$ coinvolgono una tra le variabili $x_1, \ldots, x_k$ e, quindi, sono più grandi di ogni monomio in $f \in k[x_{k+1}, \ldots, x_n]$; così, quando applichiamo l'algoritmo della divisione, $g_{r+1}, \ldots, g_m$ non compariranno.

Da queste osservazioni segue che la divisione di f per $g_1, \ldots, g_m$ porta ad una equazione della seguente forma:

$$f = h_1 g_1 + \cdots + h_r g_r + 0 \cdot g_{r+1} + \cdots + 0 \cdot g_m + 0,$$

che implica $f \in \langle g_1, \ldots, g_r \rangle$. Questo prova che G_k è una base di I_k. Osserviamo, inoltre, che questo ragionamento mostra che, per $f \in I_k$, dividere f per G (usando l'algoritmo della divisione) è uguale a dividerla per G_k; in particolare, entrambi i resti sono zero.

Per dimostrare che G_k è una base di Gröbner basta provare che, per ogni $1 \leq i < j \leq r$, il resto della divisione di $S(g_i, g_j)$ per G_k è zero (per il teorema 4.18); ma gli S-polinomi $S(g_i, g_j)$ stanno in I_k poiché $g_i, g_j \in I_k$. Prima, abbiamo osservato che il resto della divisione di f per G_k è zero, ed è ciò che volevamo appunto provare. Questo completa la dimostrazione del teorema. □

Consideriamo l'esempio 5.2 per mostrare il funzionamento di questo teorema. L'ideale I è dato da:

$$I = \langle x^2 + y + z - 1; x + y^2 + z - 1; x + y + z^2 - 1 \rangle,$$

e la base di Gröbner è data in 5.4. Dal teorema di eliminazione segue che:

$$I_1 = I \cap \mathbb{C}[y, z] = \langle y^2 - y - z^2 + z, 2yz^2 + z^4 - z^2, z^6 - 4z^4 + 4z^3 - z^2 \rangle,$$
$$I_2 = I \cap \mathbb{C}[z] = \langle z^6 - 4z^4 + 4z^3 - z^2 \rangle.$$

Il polinomio $g_4 = z^6 - 4z^4 + 4z^3 - z^2$ non è ottenuto in modo casuale per eliminare le variabili x e y dalle equazioni originali, ma è il modo migliore per fare ciò, perché g_4 è una base e, quindi, ogni altro polinommio che elimina x e y è multiplo di g_4.

Analizziamo il passo di estensione. Supponiamo di avere un ideale $I \subset k[x_1, \ldots, x_n]$. Come già visto nel capitolo 4 la varietà affine dell'ideale I è:

$$\mathbf{V}(I) = \{(a_1, \ldots, a_n) \in k^n : f(a_1, \ldots, a_n) = 0 \text{ per ogni } f \in I\}.$$

Per descrivere i punti di $\mathbf{V}(I)$ l'idea di base è di costruire soluzioni di una coordinata per volta. Fissiamo k tra 1 e n, così abbiamo l'ideale di eliminazione I_k. Chiameremo una soluzione $(a_{k+1}, \ldots, a_n) \in \mathbf{V}(I_k)$ una *soluzione parziale* del sistema originale di equazioni. Per estendere $(a_{k+1}, \ldots, a_n)$ ad una soluzione completa in $\mathbf{V}(I)$, prima dobbiamo aggiungere un'altra coordinata alla soluzione. Questo significa trovare a_k tale che $(a_k, a_{k+1}, \ldots, a_n)$ sta nella varietà $\mathbf{V}(I_{k-1})$ del successivo ideale di eliminazione. Supponiamo che $I_{k-1} = \langle g_1, \ldots, g_r \rangle$ in $k[x_k, x_{k+1} \ldots, x_n]$, allora vogliamo trovare soluzioni $x_k = a_k$ delle equazioni:

$$g_1(x_k, a_{k+1}, \ldots, a_n) = \cdots = g_r(x_k, a_{k+1}, \ldots, a_n) = 0.$$

Qui stiamo trattando polinomi in una variabile x_k e vediamo che le a_k possono essere solamente le radici del massimo comun divisore degli r polinomi.

Il problema di base è che i polinomi di sopra possono anche non avere radici comuni, cioè, possono esistere soluzioni parziali che non si estendono a soluzioni complete. Consideriamo, per esempio, le equazioni:

$$\begin{aligned} xy &= 1 \\ xz &= 1 \end{aligned} \tag{5.5}$$

In questo caso abbiamo che $I = \langle xy - 1, xz - 1\rangle$ e una semplice applicazione del teorema di eliminazione mostra che il primo ideale di eliminazione è: $I_1 = \langle y - z\rangle$. Le soluzioni parziali sono (a, a) e queste si possono estendere a soluzioni complete $(1/a, a, a)$, eccetto la soluzione parziale $(0, 0)$.

Il problema è: quando si possono estendere le soluzioni parziali a soluzioni complete? La risposta è data dal teorema di estensione che, però, si occupa del caso in cui viene eliminata solo la prima variabile x_1: vogliamo sapere se una soluzione parziale $(a_2, \ldots, a_n) \in \mathbf{V}(I_1)$ può essere estesa ad una soluzione $a_1, \ldots, a_n \in \mathbf{V}(I)$.

Teorema 5.2 (Teorema di estensione). [**CLO92**, p. 117]
Sia $I = \langle f_1, \ldots, f_s\rangle \subset \mathbb{C}[x_1, \ldots, x_n]$ e sia I_1 il primo ideale di eliminazione di I; per ogni $1 \leq i \leq s$ scriviamo f_i nella forma

$$f_i = g_i(x_2, \ldots, x_n)x_1^{N_i} + \textit{termini nei quali } x_1 \textit{ ha grado } < N_i$$

dove $N_i \geq 0$ e $g_i \in \mathbb{C}[x_2, \ldots, x_n]$ è diverso da zero (poniamo $g_i = 0$ quando $f_i = 0$). Supponiamo di avere una soluzione parziale $(a_2, \ldots, a_n) \in \mathbf{V}(I_1)$, se $(a_2, \ldots, a_n) \notin \mathbf{V}(g_1, \ldots, g_s)$, allora esiste $a_1 \in \mathbb{C}$ tale che $(a_1, a_2, \ldots, a_n) \in \mathbf{V}(I)$.

Osserviamo che il teorema vale solo sul campo $k = \mathbb{C}$. Per vedere l'importanza del campo $\mathbb{C}$ poniamo $k = \mathbb{R}$ e consideriamo le equazioni:

$$\begin{aligned} x^2 &= y \\ x^2 &= z \end{aligned} \tag{5.6}$$

Eliminando x, abbiamo $y = z$ e la soluzione parziale è (a, a) per ogni $a \in \mathbb{R}$. Poiché i coefficienti principali di x in $x^2 - y$ e $x^2 - z$ non si annullano, il teorema di estensione garantisce che (a, a) si estende, se stiamo in $\mathbb{C}$. Su $\mathbb{R}$, invece, la situazione cambia; $x^2 = a$ non ha soluzioni reali per a negativo e, quindi, si possono estendere, a soluzioni reali, solo quelle soluzioni parziali con a positivo. Quanto detto mostra che il teorema di estensione è falso su $\mathbb{R}$.

Tornando sull'ipotesi $(a_2, \ldots, a_n) \notin \mathbf{V}(g_1, \ldots, g_s)$, osserviamo che i polinomi g_i sono i coefficienti principali, rispetto ad x_1, delle f_i; quindi $(a_2, \ldots, a_n) \notin \mathbf{V}(g_1, \ldots, g_s)$ ci dice che i coefficienti principali non si annullano simultaneamente

nella soluzione parziale. Per vedere perché questa condizione è necessaria torniamo sull'esempio 5.7; qui le equazioni

$$\begin{aligned} xy &= 1 \\ xz &= 1 \end{aligned} \tag{5.7}$$

hanno soluzioni parziali $(y, z) = (a, a)$. L'unica soluzione che non si estende è $(0, 0)$, infatti, ha i coefficienti principali y e z di x che si annullano.

Il teorema di estensione ci dice che *il passo di estensione fallisce soltanto quando i coefficienti principali si annullano simultaneamente.*

5.3 Lo Shape lemma

Nella sezione precedente abbiamo spiegato il motivo per il quale le basi di Gröbner di un ideale forniscono un valido strumento per la risoluzione dei sistemi di equazioni polinomiali, che generano l'ideale; ma il teorema di estensione è valido nel solo campo $\mathbb{C}$ e ciò ne limita l'utilizzo. Per ovviare a questo inconveniente diamo un risultato che è valido su ogni campo.

Affermiamo che, per i nostri scopi, una base di Gröbner con un ordinamento lessicografico renderà il sistema di equazioni polinomiali in una forma triangolare. Questa affermazione è enunciata in maniera formale nello *Shape lemma.* Per capire quest'ultimo, diamo alcuni risultati importanti riguardo agli ideali.

Indichiamo $k[x_1, \ldots, x_n]$ con P e consideriamo $f_1, \ldots, f_m \in P$. Inoltre, sia $\overline{k}$ la chiusura algebrica del campo k e sia $\overline{P} = \overline{k}[x_1, \ldots, x_n]$.

Definizione 5.2 (Campo perfetto). Un campo k è chiamato un *campo perfetto* se la sua caratteristica è zero, oppure, se la sua caratteristica è $p > 0$, con p tale che $k = k^p$, cioè ogni elemento ha una radice p-esima in k.

Osservazione 5.1. I campi finiti $k = GF(q)$, dove $q = p^e$ e $e > 0$, sono perfetti poiché l'applicazione $x \mapsto x^{p^{e-1}}$ fornisce radici p-esime, perché $(x^{p^{e-1}})^p = x$ per ogni $x \in k$.

Questo risulta essere importante se il sistema di equazioni corrispondente al problema crittografico descrive un insieme finito di soluzioni oppure no. L'ideale generato dai polinomi corrispondenti ad un tale sistema sarà chiamato *zero-dimensionale.* La seguente proposizione fornisce un criterio algoritmico per la finitezza.

Proposizione 5.3 (Criterio di finitezza). *Sia $>$ un ordinamento sull'insieme dei monomi $T(P)$ dell'anello polinomiale $P = \overline{k}[x_1, \ldots, x_n]$. Per un sitema di equazioni corrispondenti ad un ideale $I = \langle f_1, \ldots, f_m \rangle$, le seguenti condizioni sono equivalenti:*

(i) *il sistema di equazioni ha soltanto parecchie soluzioni finite;*

(ii) *per* $i = 1, \ldots, n$*, abbiamo che* $I \cap k[x_i] \neq \emptyset$*;*

(iii) *l'insieme dei monomi* $T(P) \setminus \{LT_>(f) : f \in I\}$ *è finito;*

(iv) *il* k*-spazio vettoriale* P/I *ha dimensione finita.*

Dimostrazione. Vedi [**KR00**, p.243] □

Osserviamo che l'algoritmo di Buchberger è in grado di verificare la condizione (*iii*) della proposizione 5.3. Inoltre, questo criterio implica che per i nostri scopi crittografici, aggiungendo le così chiamate *equazioni del campo*:

$$\{x_i^q - x_i : 1 \leq i \leq n\} \tag{5.8}$$

potremo affermare che l'ideale è zero dimensionale.

Definizione 5.3 (Ideale zero dimensionale). Un ideale $I = \langle f_1, \ldots, f_m \rangle$ in $P = k[x_1, \ldots, x_n]$ è chiamato *zero dimensionale* se soddisfa una delle condizioni equivalenti del criterio di finitezza 5.3.

Per dire qualcosa sui possibili polinomi che si trovano nell'ideale descritto da un insieme di polinomi, il seguente teorema è di fondamentale importanza. Esso afferma che un polinomio su un campo algebricamente chiuso, avente zeri comuni con i polinomi $F = \{f_1, \ldots, f_m\}$, compare elevato a qualche potenza nell'ideale generato da F.

Teorema 5.4 (Nullstellensatz di Hilbert). *Sia* k *un campo algebricamente chiuso. Se* $f, f_1, \ldots, f_m \in P$ *sono tali che* $f \in \mathbf{I}(\mathbf{V}(f_1, \ldots, f_m))$*, allora deve esistere un intero* $e \geq 1$ *tale che:*

$$f^e \in \langle f_1, \ldots, f_m \rangle$$

(*e viceversa*).

Dimostrazione. Vedi [**CLO92**, p.172] □

Definizione 5.4 (Ideale radicale). Sia $I \subset P$ un ideale. Il *radicale* di I, denotato con $\sqrt{I}$, è l'insieme

$$\{f : f^e \in I \text{ per qualche intero } e \geq 1\}.$$

Denotiamo con $\overline{k}$ la chiusura algebrica di k. Supponiamo di lavorare con un crittosistema su $k = GF(q)$, per q potenza di un primo p. Supponiamo $F = \{f_1, \ldots, f_m\} \subset \overline{k}[x_1, \ldots, x_n]$ e che le equazioni

$$\begin{aligned} y_1 &= f_1(x_1, \ldots, x_n) \\ y_2 &= f_2(x_1, \ldots, x_n) \\ &\vdots \\ y_m &= f_m(x_1, \ldots, x_n) \end{aligned}$$

descrivano le relazioni tra i caratteri in output $y_1, \ldots, y_m$ ed i caratteri del messaggio in chiaro $x_1, \ldots, x_n \in k$. Osserviamo, però, che un sottoinsieme di $x_1, \ldots, x_n$ può rappresentare i bit della chiave di un cifrario a blocchi. Poiché i bit del messaggio sono elementi di k, non siamo interessati alle possibili soluzioni che stanno in $\overline{k} \setminus k$. Perciò, aggiungendo l'insieme

$$\{x_i^q - x_i : 1 \leq i \leq n\}$$

ad F, formiamo un ideale radicale dal quale sono ancora ricavabili i bit del messaggio. Questo è dovuto al lemma di Seidenberg.

Proposizione 5.5 (Lemma di Seidenberg). *Sia k un campo, $P = k[x_1, \ldots, x_n]$ e, inoltre, sia $I \subseteq P$ un ideale zero-dimesionale. Supponiamo che, per ogni $i \in \{1, \ldots, n\}$, esista un polinomio diverso da zero $g_i \in I \cap k[x_i]$ tale che il massimo comun divisore (MCD) tra g_i e la sua derivata sia uguale ad 1, allora I è un ideale radicale.*

Dimostrazione. Vedi [**KR00**, p.250] □

Dall'aver aggiunto le equazioni del campo, devono esistere g_i, come nella proposizione precedente, che sono coprimi con le loro derivate. L'ideale I è, quindi, radicale e, grazie al criterio di finitezza (proposizione 5.3), è anche zero-dimensionale. In ogni modo, poiché $x_i^q - x_i$ si fattorizza interamente su k, la varietà corrispondente non deve contere punti $p \in V$ con coordinate in $\overline{k} \setminus k$.
Adesso siamo pronti per enunciare lo Shape lemma, il quale mostra che le basi di Gröbner con l'ordinamento lessicografico dell'ideale I, possono essere scritte in forma triangolare.

Teorema 5.6 (Lo Shape lemma). *Sia k un campo perfetto e sia $I \subseteq P$ un ideale radicale zero-dimensionale tale che le x_n coordinate dei punti in $\mathbf{V}(I)$ siano distinti. $g_n \in k[x_n]$ sia un generatore monico dell'ideale d'eliminazione $I \cap k[x_n]$ e si abbia $d = deg(g_n)$.*

(i) *La base di Gröbner ridotta dell'ideale I rispetto all'ordianamento lessicografico $x_1 > x_2 > \ldots > x_n$, è della forma*

$$\{x_1 - g_1, ..., x_{n-1} - g_{n-1}, g_n\},$$

dove $g_1, \ldots, g_n \in k[x_n]$;

(ii) *Il polinomio g_n ha d zeri distinti $a_1, \ldots, a_d \in \overline{k}$, e l'insieme degli zeri di I è*

$$\{(g_1(a_i), \ldots, g_{n-1}(a_i), a_i) : i = 1, \ldots, d\}.$$

Dimostrazione. Vedi [**KR00**, p.257] □

Le basi di Gröbner risultano essere uno dei più importanti strumenti per risolvere sistemi algebrici, ma dipendono fortemente dalla scelta dell'ordinamento: differenti ordinamenti danno vantaggi differenti. Da un punto di vista della complessità, l'ordinamento migliore è l'ordinamento *graded reverse lex*, della definizione 4.12. Il calcolo di una base di Gröbner per l'ordinamento lessicografico è spesso un problema intrattabile per esempi grandi, dove l'ordinamento *graded reverse lex* è appena in grado di ritornare una base di Gröbner; comunque, l'ordinamento lessicografico si adatta meglio al calcolo delle soluzioni dei sistemi algebrici, come visto nello Shape lemma.

Ci sono molti modi per convertire in modo efficace una base di Gröbner da un'ordinamento ad un'altro, come per esempio l'algoritmo FGLM [**FGLM93**] e l'algoritmo Gröbner Walk ([**CKM97**]). Per un ideale zero-dimensionale I, si dimostra che l'algoritmo FGLM ha una complessità polinomiale nel numero dei monomi, non nell'ideale.

Se una base di Gröbner G di un'ideale I ha una forma triangolare, in accordo con lo Shape lemma, allora questo vuol dire che abbiamo trovato un polinomio ad una variabile g_n nell'intersezione della base con l'$(n-1)$-esimo ideale d'eliminazione $G \cap I_{n-1}$. La fattorizzazione di g_n ritorna soluzioni per x_n che possono essere immesse nei polinomi dell'intersezione $G \cap I_{n-2}$. Reiterando questo processo per tutti gli ideali d'eliminazione $I_{n-3}, \ldots, I_1$, otteniamo la varietà che descrive i bit del messaggio.

Capitolo 6

Tecniche avanzate sulle basi di Gröbner

In questo capitolo analizziamo degli algoritmi per il calcolo delle basi di Gröbner, ottenuti con criteri migliorati rispetto a quelli di Buchberger. Il primo di questi è l'algoritmo F4. Questo, per evitare quante più operazioni intermedie possibili, calcola, in modo consecutivo, basi di Gröbner troncate;inoltre, rimpiazza la classica riduzione polinomiale che si trova nell'algoritmo di Buchberger con la riduzione simultanea di più polinomi. Questo potente meccanismo di riduzione è attuato da un precalcolo simbolico e dall'uso intensivo dei metodi di algebra lineare. L'algoritmo F4 è stato progettato per essere efficiente sfruttando l'ordinamento degree reverse lexicographical (DRL) [**Fau99**].

Paradossalmente se l'algoritmo di Buchberger originale è molto semplice da descrivere, diventa più difficile capire come migliorare una reale implementazione. È possibile pensare, comunque, a due miglioramenti: poiché il 90% del tempo di esecuzione dell'algoritmo è speso nel calcolo di polinomi che sono congrui a zero ([**Fau99**]), sarebbe opportuno avere un criterio più efficiente per eliminare coppie critiche inutili, perché, alla fine, congrue a zero [**Buch79**] (criteri teorici, come la proposizione 4.25, esistono ma sono computazionalmente troppo costosi). Questo aspetto cruciale del problema è sviluppato nell' algoritmo F5, introdotto in [**Fau02**], che è basato su un efficiente criterio per la scelta di coppie critiche, combinato con una strategia di selezione basata sull'algebra lineare. Questa *strategia di selezione* fu introdotta da Faugère nell'algoritmo F4 [**Fau99**]. Il secondo miglioramento è collegato alle strategie: durante il calcolo di una base di Gröbner si possono fare molte scelte (selezionare una coppia critica, scegliere un riduttore). Usualmente, una *strategia di selezione* è costituita da due parti. Prima di tutto abbiamo bisogno di avere un ordinamento con cui selezionare le coppie critiche da ridurre (Criterio di Buchberger I); per esempio, nell'algoritmo di Buchberger, questa scelta determina quali S-polinomi ridotti sono aggiunti alla base intermedia e influisce sulle nuove

riduzioni durante il corso dell'algoritmo. In secondo luogo, come nell'algoritmo della divisione, bisogna scegliere un ordinamento sugli elementi della base intermedia, la quale servirà alla riduzione dei nuovi S-polinomi.

6.1 Moduli

La definizione di un modulo è identica a quella di spazio vettoriale, eccetto per il fatto che il campo k è rimpiazzato da un anello arbitrario (considereremo anelli commutativi con unità [**BW93**]); i moduli stanno agli anelli come gli spazi vttoriali stanno ai campi.

Definizione 6.1 (R-modulo). Sia R un anello. Un R-modulo M è un gruppo abeliano additivo $(M, +)$ con un'operazione addizionale $\cdot : R \times M \to M$, chiamata *prodotto scalare*, tale che per ogni $\alpha,\ \beta \in R$, ed $a, b \in M$, le seguenti proprietà siano verificate:

(i) $\alpha \cdot (a + b) = \alpha \cdot a + \alpha \cdot b$,

(ii) $(\alpha + \beta) \cdot a = \alpha \cdot a + \beta \cdot a$,

(iii) $(\alpha\beta) \cdot a = \alpha \cdot (\beta \cdot a)$,

(iv) $1 \cdot a = a$.

Definizione 6.2 (Omomorfismo di R-moduli). Una mappa $\varphi : M \to M^{'}$ tra due R-moduli M e $M^{'}$ è un *omomorfismo di R-moduli* se per ogni $a, b \in M$ e $\alpha \in R$,

$\varphi(a + b) = \varphi(a) + \varphi(b)$

$\varphi(\alpha \cdot a) = \alpha \cdot \varphi(a)$

Un omomorfismo da M in se stesso è chiamato un *endomorfismo* di M.

Definizione 6.3 (Sottomodulo). Sia M un R-modulo e sia N un sottogruppo additivo di M. Allora N è un *sottomodulo* di M se N è chiuso sotto l'operazione di prodotto scalare. Sia $M = R[x]$ un anello polinomiale ad una variabile sull'anello R, $n \in \mathbb{N}$ e N un sottoinsieme di M, costituito da tutti i polinomi di grado minore o uguale di n; allora N è un sottomodulo di M, ma non un sottoanello dell'anello $R[x]$. Ogni volta che B è un sottoinsieme di M, deve esistere un unico sottomodulo più piccolo N di M che contiene B come sottoinsieme. Il sottomodulo N è costituito da tutte le combinazioni lineari:

$$\sum_{i=1}^{n} \alpha_i \cdot a_i, \text{ con } (\alpha_i \in R, a_i \in B)$$

N è chiamato il *sottomodulo generato da* B in M e sarà denotato con $lin(B)$. Un *sistema di generatori* per M è un sottoinsieme B di M con $lin(B) = M$. M è detto *R-modulo finitamente generato* se M ha un sistema di generatori finito. L'R-modulo M è detto *noetheriano* se ogni sottomodulo di M è finitamente generato. Un sottoinsieme B di M è detto *linearmente indipendente* se per ogni $n \in \mathbb{N}^+$, $a_1, \ldots, a_n \in B$ a due a due differenti, e $\alpha_1, \ldots, \alpha_n \in R$,

$$\sum_{i=1}^{n} \alpha_i \cdot a_i = 0 \Rightarrow \alpha_1 = \cdots = \alpha_n = 0$$

B è detta una *base* di M se B è anche un sistema di generatori per M.

Indichiamo con P l'anello di polinomi $k[x_1, \ldots, x_n]$: $P = k[x_1, \ldots, x_n]$.

Definizione 6.4 (Sigizia di termini principali). Sia $F = (f_1, \ldots, f_m) \in P^m$. Una *sigizia* sui termini principali $LT(f_1), \ldots, LT(f_m)$ è una m-pla di polinomi $S = (h_1, \ldots, h_m) \in P^m$, tali che

$$\sum_{i=1}^{m} h_i LT(f_i) = 0$$

Poniamo $S(F)$ come il sottoinsieme di P^m costituito da tutte le sigizie dei termini principali di F.

Definizione 6.5 (Modulo delle sigizie). Sia M un R-modulo e $a_1, \ldots, a_n \in M$, allora l'insieme $S(a_1, \ldots, a_n)$ di tutte le *sigizie* dell'n-pla $(a_1, \ldots, a_n) \in M^n$ è formato da tutte le n-ple $\alpha = (\alpha_1, \ldots, \alpha_n) \in R^n$ tali che:

$$\sum_{i=1}^{n} \alpha_i \cdot a_i = 0$$

È facile verificare che $S(a_1, \ldots, a_n)$ forma un sottomodulo dell'R-modulo R^n che è anche chiamato il *modulo delle sigizie* di $(a_1, \ldots, a_n)$.

6.2 Polinomi ed algebra lineare

Definizione 6.6 (Insieme delle coppie distinte). Sia $F = (f_1, \ldots, f_m)$ una m-pla di polinomi in $k[x_1, \ldots, x_n]$ e siano $\mathcal{T}(F)$ e $\mathcal{T}_\rho(F)$, rispettivamente, l'insieme delle coppie distinte di monomi in F e l'equivalente ordinato rispetto ad un ordinamento ρ. Denotiamo con s la cardinalità di $\mathcal{T}(F)$, $|\mathcal{T}(F)| = s$, che rappresenta, appunto, il numero di monomi distinti in F.

Definizione 6.7 (Applicazione di rappresentazione vettoriale). Sia $f \in k[x_1, \ldots, x_n]$ un generico polinomio scritto come

$$f = \sum_{i=1}^{s} c_i x^{\alpha_i}, \text{ con } \alpha_i \in \mathbb{Z}_+^n \text{ e } c_i \in k$$

Definiamo l'*applicazione di rappresentazione vettoriale*

$$\begin{aligned} \psi_{\mathcal{T}_\rho(F)} : & P \longrightarrow k^s \\ & f \longmapsto (c_1, \ldots, c_s) \end{aligned}$$

quindi, dato un polinomio f in P, si ha:

$$\psi_{\mathcal{T}_\rho(F)}(f) = (c_1, \ldots, c_s)$$

Possiamo definire anche la *rappresentazione matriciale* di una m-pla di polinomi F

$$\begin{aligned} \Psi_{\mathcal{T}_\rho(F)} : \quad P^m \quad &\longrightarrow \quad Mat_{m,s}(k); \\ (f_1, \ldots, f_m) &\longmapsto \begin{pmatrix} \psi_{\mathcal{T}_\rho(F)}(f_1) \\ \vdots \\ \psi_{\mathcal{T}_\rho(F)}(f_m) \end{pmatrix} \end{aligned}$$

Sarà chiaro dal contesto rispetto a quale ordinamento e quale supporto siano rappresentati i polinomi e, quindi, i pedici saranno omessi.

Diamo, adesso, una definizione che è equivalente a quella degli S-polinomi ridotti dell'algoritmo di Buchberger.

Definizione 6.8. Sia F un sottoinsieme dell'anello polinomiale $k[x_1, \ldots, x_n]$. Definiamo:

(i) l'insieme dei polinomi corrispondenti alla forma a gradini di $\Psi(F)$ è denotata con $\tilde{F}$;

(ii) denotiamo con $\tilde{F}^+$ l'insieme $\{g \in \tilde{F} : LT(g) \neq LT(F)\}$.

Le propietà elementari delle matrici a gradini sono riassunte nel seguente teorema.

Teorema 6.1. *Sia k un campo ed F un insieme finito di elementi di $k[x_1, \ldots, x_n]$, denotiamo con s la cardinalità del supporto $\mathcal{T}_\rho(F)$. Per ogni sottoinsieme $H \subseteq F$ tale che $|H| = |LT(F)|$ ed $LT(H) = LT(F)$, i vettori*

$$\psi(g) \in k^s, \text{ per } g \in \tilde{F}^+ \cup H$$

formano una base triangolare del sottospazio dello spazio vettoriale k^s generato dai vettori $\psi(f)$ per $f \in F$.

Dimostrazione. Sia $G = \tilde{F}^+ \cup H$. Tutti gli elementi g di G hanno termini principali distinti e sono combinazioni lineari di elementi di F; quindi, l'insieme $\{\psi(g) : g \in G\}$ è linearmente indipendente ed è contenuto nel sottospazio generato dai vettori corrispondenti agli elementi di F. Sia r il rango del sottospazio generato da $\psi(f)$, con $f \in F$. Abbiamo:

$$LT(G) = LT(\tilde{F}^+) \cup LT(H) = LT(\tilde{F})$$

che implica $|LT(G)| = |LT(\tilde{F})| = r$. □

Come conseguenza del teorema precedente abbiamo che l'ideale $\langle F \rangle$ è generato da $H \cup \tilde{F}^+$.

6.3 Algoritmo F4

La differenza principale tra l'algoritmo F4 e quello di Buchberger sta nel modo di ridurre gli S-polinomi. Invece di calcolare la riduzione di ogni S-polinomio singolarmente, l'algoritmo F4 sceglie un insieme di coppie critiche (a, b), con a, b nella base intermedia G' e fornisce alla funzione di riduzione i due polinomi

$$\frac{mcm(LT(a), LT(b))}{LT(a)}a, \quad \frac{mcm(LT(a), LT(b))}{LT(b)}b.$$

Supponiamo di usare la strategia di selezione normale (vedi definizione 6.12). Le coppie critiche corrispondenti al grado d sono:

$$B_d = \{(a, b) : a, b \in G' \text{ dove } totaldeg(mcm(LT(a), LT(b))) = d, a \neq b\}$$

Si ha, quindi, che il seguente insieme è dato come input alla funzione di riduzione simultanea di F4.

$$L_d = \bigcup_{(a,b)\in B_d} \left\{ \frac{mcm(LT(a), LT(b))}{LT(a)}a, \frac{mcm(LT(a), LT(b))}{LT(b)}b \right\}. \tag{6.1}$$

La riduzione in F4 usa riduttori già calcolati di una base intermedia G'. L'aggiunta di riduttori è compiuta da una funzione chiamata Simbolic Preprocessing.

Definizione 6.9 (Riduttore). Durante l'esecuzione di un algoritmo per il calcolo di una base di Gröbner, un *riduttore* r di un insieme F è un polinomio che soddisfa

$$LT(r) \in \mathcal{T}(F) \setminus LT(F).$$

Definizione 6.10 (Symbolic Preprocessing). [**Fau99**, p.67]
Il seguente algoritmo aggiunge riduttori all'insieme F rispetto alla base intermedia G'.

Input: Un insieme $F \subset k[x_1, \ldots, x_n]$ e una base intermedia G'.
Output: L'iniseme $F \cup R$ per un insieme di riduttori R
$D := LT(F)$, $R := \{\}$
while $\mathcal{T}(F \cup R) \neq D$ **do**
  Sceglie $m \in \mathcal{T}(F \cup R) \setminus D$
  Aggiunge m a D
  if $LT(m)$ é divisibile per un elemento $g \in LT(G')$ **then**
    $m' := m/LT(g)$
    Aggiunge gm' ad R.

Definizione 6.11 (RiduzioneF4). La funzione *Riduzione*F4 ritorna in output $\tilde{F}^+$, dove F è l'output della funzione Symbolic Preprocessing applicata all'insieme L_d, come definito nell'equazione 6.1, rispetto alla base intermedia G'.

Input: L'insieme L_d e una base intermedia G'
Output: L'insieme $\tilde{F}^+$
$F := SymbolicPreprocessing(L_d, G')$
$\tilde{F}$:=riduzione alla forma a gradini di F
$\tilde{F}^+ := \{g \in \tilde{F} : LT(g) \notin LT(F)\}$.

La funzione Symbolic Preprocessing è richiamata all'interno della funzione di riduzione di F4 e, così, le riduzioni vengono effettuate con i riduttori selezionati dal Symbolic Preprocessing.

Nell'algoritmo di Buchberger gli S-polinomi, che non sono ridotti a zero, servono per espandere l'ideale generato dai termini principali della base intermedia. In questo modo si ottiene una catena crescente di termini principali dell'ideale. In F4, similmente, i termini principali degli elementi di $\tilde{F}^+$ contribuiscono all'ideale generato dai termini principali della base intermedia. Tutto questo è enunciato formalmente nel seguente lemma.

Lemma 6.2. *Sia $\tilde{F}^+$ l'output della funzione di riduzione di F4 applicata all'insieme L_d rispetto alla base intermedia G', $\tilde{F}^+ = RiduzioneF4(L_d, G')$. Per ogni $f \in \tilde{F}^+$, $LT(f)$ non è un elemento di $\langle LT(G')\rangle$.*

Dimostrazione. Sia f un elemento in $\tilde{F}^+$ e sia $F = SymbolicPreprocessing(L_d, G')$. Supponiamo, per assurdo, che $LT(f) \in \langle LT(G')\rangle$. Questa ipotesi insieme con $LT(f) \in \mathcal{T}(\tilde{F}^+) \subset \mathcal{T}(\tilde{F})$ implicano che il Symbolic Preprocessing deve aggiungere un riduttore $\frac{LT(f)}{LT(g)}g$ ad F per un opportuno $g \in G'$. Questo vuol dire $LT(f) \in LT(F)$, che è una contraddizione per la definizione di $\tilde{F}^+$; quindi $LT(f)$ non è un elemento di $\langle LT(G')\rangle$. □

Il prossimo lemma serve per provare la correttezza dell'algoritmo F4.

Lemma 6.3. *Sia* $\tilde{F}^+ = RiduzioneF4(L_d, G')$*, allora si hanno le seguenti proprietà:*

(i) $\tilde{F}^+ \subset \langle G' \rangle$.

(ii) *Per ogni k-combinazione lineare,* f*, di elementi di* L_d*, la forma normale di* f *rispetto a* $\tilde{F}^+ \cup G'$ *è uguale a zero.*

Dimostrazione. (i) Ogni $f \in \tilde{F}^+$ è una combinazione lineare di elementi di L_d e riduttori R che sono entrambi sottoinsiemi di $\langle G' \rangle$.
(ii) Sia f una combinazione lineare di elementi di L_d. Supponiamo che F sia l'output della funzione Symbolic preprocessing applicata ad L_d e G'. Per costruzione, L_d è un sottoinsieme di F e, per il teorema 6.1, questi elementi sono una combinazione lineare della base triangolare $\tilde{F}^+ \cup H$, per un opportuno $H \subset F$. Gli elementi di H o sono elementi di L_d o sono della forma $x^\alpha g$, per $g \in G'$ ed $\alpha \in \mathbb{Z}_+^n$ e, quindi, f può essere scritta come:

$$f = \sum_i a_i f_i + \sum_j a_j x^{\alpha_j} g_j$$

per $f_i \in \tilde{F}^+$ e $g_j \in G'$, $a_i, a_j \in k$ e $\alpha_j \in \mathbb{Z}_+^n$. Quindi l'algoritmo della divisione dà un resto uguale a zero per un'opportuna m-pla di elementi in $\tilde{F}^+ \cup G'$ e, così, esiste una catena di riduttori che va a zero. □

Diamo una descrizione dell'algoritmo F4 e dimostriamo la sua correttezza. La strategia di selezione (rappresentata nell'algoritmo dalla funzione *Sel*) può variare come si vule: si può scegliere di selezionare tutte le coppie critiche disponibili in quel momento, oppure, per esempio, usare la strategia di selezione normale (come consiglia Faugére in [**Fau02**, p.73]).

Teorema 6.4 (Algoritmo F4). [**Fau99**, p.68] *L'algoritmo F4 calcola una base di Gröbner* G*, con* G *in* $k[x_1, \ldots, x_n]$*, di un ideale generato da* F*, tale che* $F \subset G$ *e* $\langle F \rangle = \langle G \rangle$*, in un numero finito di passi.*
Input: $F = \{f_1, \ldots, f_m\} \subset k[x_1, \ldots, x_n]$.
Output: Una base di Gröbner G per $\langle F \rangle$, che verifica $F \subset G$.
$G' := F$, $\tilde{F}_0^+ := F$, $d := 0$
$B := \{(a, b) : a, b \in G' \text{ con } a \neq b\}$
while $B \neq \emptyset$ **do**
    $d := d + 1$
    $B_d := Sel(B)$
    $B := B \setminus B_d$

$L_d := \bigcup_{(a,b)\in B_d} \left\{ \frac{mcm(LT(a),LT(b))}{LT(a)} a, \frac{mcm(LT(a),LT(b))}{LT(b)} b \right\}$
$\tilde{F}_d^+ := RiduzioneF4(L_d, G')$
for $f \in \tilde{F}_d^+$
  $B := B \cup \{(f,g) : g \in G'\}$
  $G' := G' \cup \{f\}$
  $G := G'$

Dimostrazione. La correttezza e la finitezza dell'algoritmo sono provate dalle seguenti osservazioni.

1. Dal lemma 6.3 abbiamo che durante il passo $d = d'$ dell'algoritmo, la base intermedia soddisfa:
$$G' = \bigcup_{d=1}^{d'} \tilde{F}_d^+ \subset \langle F \rangle$$

2. Il lemma 6.2 mostra che:
$$\langle LT(\tilde{F_1^+}) \rangle \subset \langle LT(\tilde{F_1^+} \cup \tilde{F_2^+}) \subset \cdots$$
è una catena crescente di ideali monomiali. La condozione di catena crescente del teorema 4.13 afferma che questa, alla fine, si stabilizza. Questo implica che il while-loop deve terminare poiché, ad un certo punto, esauriamo le coppie critiche.

3. Supponiamo che l'algoritmo termini al passo $d = d_{F4}$. Poiché ogni coppia (g_1, g_2), per $g_1, g_2 \in G = \bigcup_{d=1}^{d_{F4}} \tilde{F}_d^+$, è considerata, $S(g_1, g_2)$ si può esprimere come combinazione lineare di elemeti di G. Dal lemma 6.3 (ii) abbiamo che la loro forma normale è uguale a zero; quindi le ipotesi del criterio per le basi di Gröbner del teorema 4.24 sono soddisfatte

□

Vediamo come funziona la strategia di *selezione normale*:

Definizione 6.12 (Selezione normale). La funzione di selezione nell'algoritmo F4 è indicata con $Sel(P)$ e, nel caso della *selezione normale*, compie le seguenti operazioni:

Input: Una lista di coppie critiche P.
Output: Una lista di coppie critiche P_d
  $d := min\{deg(mcm(p)) : p \in P\}$
  $P_d := \{p \in P : deg(mcm(p)) = d\}$
Return P_d
Quindi questa funzione seleziona le coppie critiche che hanno grado totale minimale.

L'algoritmo F4 può essere migliorato utilizzando come criterio per eliminare coppie critiche inutili quello dell'installazione di Gebauer e Möller. Questo criterio può essere applicato ad ogni estensione dell'algoritmo di Buchberger, che seleziona iterativamente coppie critiche e calcola i corrispondenti S-polinomi ridotti.

Durante l'esecuzione di F4, in realtà, non vengono calcolati gli S-polinomi $S(g_i, g_j)$, ma le loro componenti:

$$\frac{x^{\gamma_{ij}}}{LT(g_i)} g_i \text{ e } \frac{x^{\gamma_{ij}}}{LT(g_j)} g_j \text{ con } x^{\gamma_{ij}} = mcm(LT(g_i), LT(g_j)),$$

che sono allocate dentro la matrice di riduzione. Il più grande dei gradi totali di questi termini corrisponde al grado totale di un S-polinomio fittizio. Questo non deve corrispondere necessariamente con il grado più grande delle coppie critiche durante il corso dell'algoritmo, poiché un S-polinomio può essere ridotto a zero, mentre le sue componenti contribuiscono al passo di riduzione. Al contrario, il grado totale di un polinomio intermedio, durante il calcolo di una base di Gröbner con l'ordinameto lessicografico, può superare il grado più grande delle coppie critiche, perché, in questo caso, un S-polinomio può ridursi ad un elemento nell'ideale di grado totale più grande.

6.4 L'installazione di Gebauer e Möller e l'algoritmo F5

L'installazione di Gebauer e Möller è stata introdotta in [**GM**] ed utilizza, per il calcolo delle basi di Gröbner, un miglioramento dei due criteri proposti da Buchberger in [**Buch85**]. Cabora, Kreuzer e Robbiano [**CKR04**] hanno mostrato come calcolare un insieme minimale di coppie critiche, concludendo che i criteri di Gebauer e Möller sono pressoché ottimali. In questa sezione introduciamo l'installazione di Gebauer e Möller e il criterio per scartare coppie critiche inutili utilizzato nell'algoritmo F5 da Faugère in [**Fau02**]. Nell'installazione di Gebauer e Möller partiamo dal fatto che le basi di Gröbner possono essere caratterizzate usando una base del loro modulo delle sigizie (vedi definizione 6.5). Diamo alcune definizioni di cui abbiamo bisogno.

Sia $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0) \in P^m$ il *vettore della base canonica*, dove l'uno è all'i-esimo posto, allora una sigizia $S \in S(F)$ può essere scritta come $S = \sum_{i=1}^{m} h_i \mathbf{e}_i$, con $h_i \in P$. Usando queste notazioni, una sigizia di un S-polinomio è definita come segue.

Definizione 6.13. Sia F una m-pla $(f_1, \ldots, f_m) \in P^m$. La sigizia S_{ij} corrispondente all'S-polinomio $S(f_i, f_j)$ è definita come:

$$S_{ij} = \frac{x^{\gamma}}{LT(f_i)} \mathbf{e}_i - \frac{x^{\gamma}}{LT(f_j)} \mathbf{e}_j, \tag{6.2}$$

dove $x^\gamma = mcm(LM(f_i), LM(f_j))$.

Un'importante propietà di $S(F)$ è che esso ha una base finita. Esiste una collezione finita di sigizie tale che ogni altra sigizia è una combinazione lineare con coefficienti polinomiali della base delle sigizie. Per dimostrare questa proprietà abbiamo bisogno di definire la nozione di *sigizia omogenea*.

Definizione 6.14 (Sigizia omogenea). Un elemento $S \in S(F)$ è *omogeneo di multigrado* α, dove $\alpha \in \mathbb{Z}^+$, se vale

$$S = (c_1 x^{\alpha(1)}, \ldots, c_m x^{\alpha(m)})$$

dove c_i è un elemento del campo k e $\alpha(i) + multideg(f_i) = \alpha$, quando $c_i \neq 0$.

Le sigizie della definizione 6.13 sono omogenee di grado γ, secondo la definizione precedente.

Lemma 6.5. *Ogni elemento di $S(F)$ può essere scritto in modo unico come somma di elementi omogenei di $S(F)$*

Dimostrazione. (Esistenza) Sia $S = (h_1, \ldots, h_m) \in S(F)$. Fissiamo un esponente $\alpha \in \mathbb{Z}^+$ e sia $h_{i\alpha}$ il termine di h_i tale che $h_{i\alpha} f_i$ ha multigrado α. Allora dobbiamo avere $\sum_{i=1}^m h_{i\alpha} LT(f_i) = 0$, poiché $h_{i\alpha} LT(f_i)$ sono termini di multigrado α nella somma $\sum_{i=1}^m h_i LT(f_i) = 0$. Si desume, quindi, che $S_\alpha = (h_{1\alpha}, \ldots, h_{m\alpha})$ è un elemento omogeneo di $S(F)$ di grado α e $S = \sum_\alpha S_\alpha$. □

La seguente proposizione afferma che le sigizie degli S-polinomi S_{ij} formano una base di tutte le sigizie sui termini principali $S(F) \subset P^m$.

Proposizione 6.6. *Data $F = (f_1, \ldots, f_m)$, ogni sigizia $S \in S(F)$ può essere scritta come:*

$$S = \sum_{i<j} u_{ij} S_{ij},$$

con $u_{ij} \in P$; quindi,

$$\{S_{ij} : 1 \leq i < j \leq m\},$$

con S_{ij} omogenei di multigrado γ, è una base omogenea di $S(F)$.

Dimostrazione. Dal lemma 6.5, possiamo assumere che S è omogenea di grado α, allora S deve avere al minimo due componenti diverse da zero, diciamo $c_i x^{\alpha(i)}$ e $c_j x^{\alpha(j)}$, dove $i < j$. Avremo $\alpha(i) + multideg(f_i) = \alpha(j) + multideg(f_j) = \alpha$, ciò implica che $x^\gamma = mcm(LM(f_i), LM(f_j))$ divide x^α. Poiché

$$S_{ij} = \frac{x^\gamma}{LT(f_i)} \mathbf{e}_i - \frac{x^\gamma}{LT(f_j)} \mathbf{e}_j$$

un semplice calcolo mostra che la i-esima componente di

$$S - c_i LC(f_i) x^{\alpha-\gamma} S_{ij}$$

deve essere zero, e l'unica altra componente toccata è la j-esima. Segue che da S abbiamo prodotto una sigizia omogenea con meno componenti diverse da zero. Continuando in questo modo, possiamo scrivere S come una combinazione lineare delle S_{ij}. □

Un'interessante osservazione è che non abbiamo bisogno sempre di tutte le S_{ij} per generare le sigizie in $S(F)$. Vediamo ciò con un esempio.

Esempio 6.1. Sia $F = (x^2y^2 + z, xy^2 - y, x^2y + yz)$, ed usiamo l'ordinamento lessicografico in $k[x, y, z]$, allora le tre sigizie corrispondenti agli S-polinomi, sono:

$$\begin{aligned} S_{12} &= (1, -x, 0) \\ S_{13} &= (1, 0, -y) \\ S_{23} &= (0, x, -y). \end{aligned}$$

Possiamo vedere che $S_{23} = S_{13} - S_{12}$; quindi, S_{23} è ridondante, nel senso che può essere espressa come combinazione lineare delle altre due sigizie. (In questo caso, i coefficienti sono costanti; in altri casi più generali, è possibile trovare relazioni tra le sigizie con elementi polinomiali). Nel questo caso, $\{S_{12}, S_{13}\}$ forma una base per le sigizie.

Diamo adesso un criterio algoritmico per trovare le basi di Gröbner.

Teorema 6.7. *Sia $G = (g_1, \ldots, g_t) \in P^t$. Una base $G = (g_1, \ldots, g_t)$ per un ideale I è una base di Gröbner se e soltanto se per ogni elemento $S = (h_1, \ldots, h_t)$ in una base omogenea per le sigizie $S(G)$ abbiamo:*

$$S \cdot G = \sum_{i=1}^{t} h_i g_i \longrightarrow_G 0$$

Dimostrazione. Useremo la strategia (e le notazioni) della dimostrazione del teorema 4.18. Partiamo con $f = \sum_{i=1}^{t} h_i g_i$. Sia $m(i) = multideg(h_i g_i)$ minimale tra tutti i modi possibili di scrivere f in termini di $g_1, \ldots, g_t$ e sia $\delta = max(m(i))$. Come prima, abbiamo bisogno di mostrare che $multideg(f) < \delta$ porta ad una contraddizione. Dall'equazione 4.4, sappiamo che $multideg(f) < \delta$ implica che $\sum_{m(i)=\delta} LT(h_i) g_i$ ha multigrado strettamente più piccolo di δ. Questo vuol dire che $\sum_{m(i)=\delta} LT(h_i) LT(g_i) = 0$, così che:

$$S = \sum_{m(i)=\delta} LT(h_i) \mathbf{e}_i$$

è una sigizia in $S(G)$. Osserviamo anche che S è omogeneo di grado δ. Le nostre ipotesi ci forniscono una base omogenea $S_1, \ldots, S_m$ di $S(G)$, con la propietà che $S_j \cdot G \to_G 0$ per ogni j. Possiamo scrivere S nella forma

$$S = u_1 S_1 + \cdots + u_m S_m. \tag{6.3}$$

Scrivendo le u_j come somme di termini ed, espandendo, vediamo che l'equazione 6.3 esprime S come una somma di sigizie omogenee. Poiché S è omogenea di multigrado δ, segue, dall'unicità del lemma 6.5, che possiamo scartare tutte le sigizie di multigrado diverso da δ. Quindi nell'equazione 6.3 possiamo assumere che, per ogni j, si abbia:

$$u_j = 0; \text{ oppure } u_j S_j \text{ è omogeneo di multigrado } \delta.$$

Supponiamo che S_j abbia multigrado γ_j. Se $u_j \neq 0$, allora segue che u_j può essere scritta nella forma $u_j = c_j x^{\delta - \gamma_j}$ per qualche $c_j \in k$. Così, la 6.3 può essere scritta

$$S = \sum_j c_j x^{\delta - \gamma_j} S_j$$

dove la somma è su quei j tali che $u_j \neq 0$. Moltiplicando (utilizzando l'opportuna operazione di prodotto tra elementi di P^t) ambo i membri per G, otteniamo:

$$\sum_{m(i)=\delta} LT(h_i) g_i = S \cdot G = \sum_j c_j x^{\delta - \gamma_j} S_j \cdot G \tag{6.4}$$

Per ipotesi, $S_j \cdot G \to_G 0$, cioè

$$S_j \cdot G = \sum_{i=1}^{t} a_{ij} g_i \tag{6.5}$$

dove

$$multideg(a_{ij} g_i) \leq multideg(S_j \cdot G) \tag{6.6}$$

per ogni i, j. Osserviamo che le equazioni 6.4, 6.5, 6.6 sono simili alle corrispondenti 4.5, 4.6, 4.7; infatti il resto della dimostrazione del teorema è uguale a quello che avevamo nella dimostrazione del teorema 4.18. L'unico dettaglio che dovremmo verificare è che $x^{\delta - \gamma_j} S_j \cdot G$ ha multigrado minore di δ. Verificato ciò, il teorema è dimostrato. □

Se usiamo la base $\{S_{ij}\}$ per le sigizie $S(G)$, allora, è possibile verificare che i polinomi $S_{ij} \cdot G$ corrispondono esattamente agli S-polinomi $S(g_i, g_j)$, che compaiono durante lo svolgimento dell'algoritmo di Buchberger.

Per poter utilizzare il teorema 6.7 al meglio, abbiamo bisogno di conoscere come costruire basi omogenee più piccole di $S(G)$. Adesso mostreremo che, partendo dalla base $\{S_{ij} : i < j\}$, esiste un modo sistematico per dire quando un elemento può essere omesso.

Proposizione 6.8. *Data $G \subset (g_1, \ldots, g_m) \in P^m$, supponiamo di avere un sottoinsieme $L \subset \{S_{ij} : 1 \leq i < j \leq m\}$ che è una base di $S(G)$. In aggiunta, supponiamo di avere elementi distinti g_i, g_j e $g_k \in G$ tali che:*

$$LT(g_k) \text{ divide } mcm(LT(g_i), LT(g_j))$$

Se $S_{ij}, S_{jk} \in L$, allora $L \setminus \{S_{ij}\}$ è ancora una base di $S(G)$. (Osserviamo che se $i > j$, poniamo $S_{ij} = S_{ji}$.)

Dimostrazione. Senza ledere la generalità, assumiamo che $i < j < k$. Poniamo $x^{\gamma_{ij}} = mcm(LM(g_i), LM(g_j))$, e siano $x^{\gamma_{ik}}$ e $x^{\gamma_{jk}}$ definite in modo analogo. Sia $x^{\gamma_{ijk}} = mcm(LM(g_i), LM(g_j), LM(g_k))$. Le nostre ipotesi implicano che $x^{\gamma_{ik}}$ e $x^{\gamma_{jk}}$ dividono entrambe $x^{\gamma_{ij}}$. Dobbiamo dimostrare che:

$$S_{ij} = \frac{x^{\gamma_{ij}}}{x^{\gamma_{ik}}} S_{ik} - \frac{x^{\gamma_{ij}}}{x^{\gamma_{jk}}} S_{jk}$$

Ci sono $m(m-1)/2$ sigizie S_{ij}. Se c'è qualche dipendenza in questi generatori di $S(G)$, allora esistono sigizie di ordine più alto nel modulo $P^{m(m-1)/2}$ che le cancellano. Per creare una base canonica per questo modulo delle sigizie, le $m(m-1)/2$ sigizie sono ordinate con $<_1$, che è definito nel seguente modo:

$$S_{ab} <_1 S_{cd} \Leftrightarrow x^{\gamma_{ab}} < x^{\gamma_{cd}} \text{ o } (x^{\gamma_{ab}} = x^{\gamma_{cd}}, b \leq d, \text{ dove } b = d \Rightarrow a < c).$$

Usando questo ordinamento, denoteremo l'i-esimo vettore unità in $P^{m(m-1)/2}$ non più con $\mathbf{e}_i$ ma con $\mathbf{e}_{ab}$, se S_{ab} è l'i-esima sigizia in questo ordinamento.
Il modulo delle sigizie

$$S^{(2)}(G) = \left\{ \sum_{\substack{i,j=1 \\ i<j}}^{m} h_{ij}\mathbf{e}_{ij} \in P^{m(m-1)/2} : \sum_{\substack{i,j=1 \\ i<j}}^{m} h_{ij} S_{ij} = 0 \right\}$$

ha la base $L^{(2)} = \{S_{ijk} : 1 \leq i < j < k \leq m\}$ con

$$S_{ijk} = \frac{x^{\gamma_{ijk}}}{x^{\gamma_{ij}}}\mathbf{e}_{ij} - \frac{x^{\gamma_{ijk}}}{x^{\gamma_{ik}}}\mathbf{e}_{ik} + \frac{x^{\gamma_{ijk}}}{x^{\gamma_{jk}}}\mathbf{e}_{jk}$$

Gli elementi S_{ijk} sono omogenei di multigrado γ_{ijk} secondo la definizione 6.14. Le nostre ipotesi implicano che S_{ijk} e S_{ij} sono omogenei di multigrado uguale $\gamma_{ij} = \gamma_{ijk}$. In questo caso, S_{ij} può essere espressa in termini delle sigizie S_{ik} e S_{jk}, poiché S_{ijk} è un elemento del secondo modulo delle sigizie $S^{(2)}(G)$; quindi,

$$S_{ij} = \frac{x^{\gamma_{ijk}}}{x^{\gamma_{ik}}} S_{ik} + \frac{x^{\gamma_{ijk}}}{x^{\gamma_{jk}}} S_{jk}$$

Questo ci permette di rimuovere S_{ij} da L e l'insieme $L \setminus \{S_{ij}\}$ continua a generare $S(G)$, poiché in ogni rappresentazione della base di un $S \in S(G)$, S_{ij} può essere rimpiazzato da S_{ik} e S_{jk}. □

L'installazione di Gebauer e Möller è un miglioramento dell'algoritmo di Buchberger. Un vantaggio di questo metodo è che ogni algoritmo per le basi di Gröbner, che seleziona coppie critiche in una maniera simile a quella dell'algoritmo di Buchberger, può utilizzare questo criterio di selezione (vedi [**GM**]).

Teorema 6.9 (Installazione di Gebauer e Möller). *Sia $I = \langle f_1, \ldots, f_m \rangle \neq \{0\}$ un'ideale polinomiale, allora una base di Gröbner per I può essere costruita in un numero finito di passi dal seguente algoritmo.*
Input: $F = \{f_1, \ldots, f_m\}$
Output: Una base di Gröbner G per $\langle f_1, \ldots, f_m \rangle$.
$G := \{f_1\}$
$D := \{\}$
for $t := 2$ **to** m
  $UpdatePairs(D, t)$
  $G := G \cup \{f_t\}$
  $r := m$
while esistono $(i, j) \in D$ **repeat**
  $h := \overline{S(f_i, f_j)}^G$
  if $h \neq 0$ **then**
    $f_{r+1} := h$
    $D := UpdatePairs(D, r + 1)$
    $G := G \cup \{f_{r+1}\}$
    $r := r + 1$
  $D := D \setminus \{(i, j)\}$
return G

La procedura UpdatePairs è definita come segue.

Input: Un insiema di coppie D e un intero positivo t.
Output: Un insieme aggiornato di coppie critiche D, tale che $\{S_{ij} : (i, j) \in D\}$ insieme con qualche S_{ij}, con $1 \leq i < j \leq t$ e $LT(f_i), LT(f_j) = x^{\gamma_{ij}}$, forma una base del modulo delle sigizie

$$\left\{ (g_1, \ldots, g_t) \in P^t : \sum_{i=1}^{t} g_i LT(f_i) = 0 \right\}.$$

1. Cancelliamo tutte le coppie in D che soddisfano

$$x^{\gamma_{ij}} = x^{\gamma_{ijt}} \text{ e } x^{\gamma_{it}} \neq x^{\gamma_{ij}} \neq x^{\gamma_{jt}};$$

2. Denotiamo l'insieme delle coppie restanti con D';

3. Sia $D1 := \{(i, t) : 1 \leq i < t\}$;

4. Cancelliamo in $D1$ ogni coppia (i,j) per le quali esiste una coppia $(j,t) \in D1$, tale che
$$x^{\gamma_{jt}} \mid x^{\gamma_{it}} \text{ e } x^{\gamma_{jt}} \neq x^{\gamma_{it}};$$

5. In ogni sottoinsieme non vuoto $\{(j,t) : x^{\gamma_{jt}} = \tau\} \subset D1'$, con monomi $\tau \in T(P)$, fissiamo un elemento (i,j) che soddisfa
$$LT(f_i)LT(f_t) = x^{\gamma_{it}}$$
oppure se una tale coppia (i,t) non esiste, ne fissiamo una arbitraria (i,t). Cancelliamo gli altri elementi di $\{(j,t) : x^{\gamma_{jt}} = \tau\}$ in $D1'$. Infine cancelliamo in $D1'$ tutte le (i,t) con
$$LT(f_i)LT(f_t) = x^{\gamma_{it}}$$
e denotiamo di nuovo con $D1'$ questo sottoinsieme di $D1'$ ottenuto alla fine;

6. **return** $D := D1' \cup D'$.

Dimostrazione. Vedi [**GM**, p.283]. □

Supponiamo di voler calcolare una base di Gröbner dell'insieme $\{f_1, \ldots, f_m\}$. Come nell'algoritmo di Buchberger, se abbiamo una coppia critica che non si riduce a zero, l'S-polinomio ridotto viene chiamato f_{m+1}, ed è aggiunto all'insieme originale. Questo procedimento viene ripetuto e gli S-polinomi sono indicizzati con l'ordine in cui sono aggiunti alla base iniziale. L'installazione di Gebauer e Möller invece di provare tutte le combinazioni delle coppie (i,j), per $1 \leq i < j$, esegue una selezione di tutte le possibili combinazioni.

6.4.1 Criterio di selezione dell'algoritmo F5

Analizziamo adesso il criterio di selezione dell'algoritmo F5. L'autore, Faugère, afferma in [**Fau02**], che se la sequenza data in input è una sequenza regolare allora l'algoritmo non genera coppie critiche inutili.

Definizione 6.15. Sia P un anello polinomiale e $I = \langle f_1, \ldots, f_m \rangle$ un ideale in P.

(i) Un ideale I è chiamato *proprio* se è diverso da P, [**BW93**, p.25]

(ii) Un elemento $f \in P$ è chiamato un *non-zerodivisore* se $fg = 0$ implica $g = 0$, per ogni $g \in P$

(iii) Una sequenza di elementi $f_1, \ldots, f_m \in P$ è chiamata una *sequenza regolare* se l'ideale I è proprio e l'immagine di f_i è un non-zerodivisore in $P/\langle f_1, \ldots, f_{i-1} \rangle$, per $i = 1, \ldots, m$.

Diamo un esempio di sequenza regolare per spiegare la definizione.

Esempio 6.2. Sia P l'anello polinomiale $\mathbb{Z}_2[x_1, x_2, x_3, x_4]$, e siano:

$$\begin{aligned} f_1 &= x_2x_4 - x_3^2, \\ f_2 &= x_1x_4 - x_2x_3, \\ f_3 &= x_1x_3 - x_2^2. \end{aligned}$$

L'ideale $I = \langle f_1, f_2, f_3\rangle$ è proprio. Ogni $f_i, f_j, i, j \in \{1, 2, 3\}$ con $i \neq j$, sono coprimi, così $f_i g = 0$ in $P/\langle f_j\rangle$ per $g \in P$ implica $g = 0$. Quindi ogni coppia di polinomi f_i, f_j, forma una sequenza regolare in P; tuttavia, $x_3 * f_3 = 0$ e $x_4 * f_3 = 0$ in $P/\langle f_1, f_2\rangle$ con x_3, x_4 non nell'ideale $\langle f_1, f_2\rangle$, quindi f_1, f_2, f_3 non è una sequenza regolare in P.

Per analizzare il criterio usato nell'algoritmo F5 abbiamo bisogno di estendere l'ordinamento originale $<$ ad un nuovo ordinamento $<_{P^m}$, per il modulo P^m su un anello polinomiale P. Indichiamo con $\mathbf{e}_i$ sempre il vettore canonico i-esimo unità $(0, \ldots, 0, 1, 0, \ldots, 0)$ in P^m.

Definizione 6.16. Per due elementi del modulo $H = \sum_{i=j}^{m} h_i\mathbf{e}_i$ ed $H' = \sum_{i=j'}^{m} h'_i\mathbf{e}_i$ in P^m, con $h_j, h_{j'}$ diversi da zero, in P, l'ordinamento dei termini del modulo è definito come segue:

$$H <_{P^m} H' \Leftrightarrow j > j' \text{ oppure } (j = j' \text{ e } LT(h_j) < LT(h'_{j'})).$$

In questa sezione omettiamo alcune definizioni date da Faugère in [**Fau02**] ed enunciamo il criterio di selezione delle coppie critiche in maniera differente dall'autore, per renderlo uniforme con il resto della trattazione fin qui data.

Una volta definito un ordinamento sugli elementi del modulo possiamo parlare di termini principali del modulo.

Definizione 6.17. Il *termine principale del modulo* di un elemento $H = \sum_{i=j}^{m} h_i\mathbf{e}_i$, con $h_j \in P$ diversi da zero, è definito come:

$$LMT(H) = LT(h_j)\mathbf{e}_j$$

Diamo, adesso, due definizioni essenziali per l'algoritmo F5, quelle di firma e di indice di un polinomio.

Definizione 6.18 (Firma ed indice di un polinomio). Durante il calcolo di una base di Gröbner di una m-pla $F = (f_1, \ldots, f_m)$, eseguita con l'algoritmo F5, la *firma* di un polinomio f, $\mathcal{S}(f)$, è uguale al termine principale del modulo del più piccolo elemento $H = \sum_{i=1}^{m} h_i\mathbf{e}_i$ che soddisfa

$$LT(H \cdot F) = LT(\sum_{i=1}^{m} h_i f_i) = LT(f). \tag{6.7}$$

Quindi $\mathcal{S}(f)$ ha la forma $t\mathbf{e}_j$, con il termine $t \in P$ e l'intero $j \in \{1, \ldots, m\}$.
L'*indice* di un polinomio f, è l'indice del vettore canonico unità presente nella firma, cioè, se $\mathcal{S}(f) = t\mathbf{e}_j$, allora $index(f) = j$.

Definizione 6.19 (t-rappresentazione). Sia P un sottoinsieme finito di $k[x_1, \ldots, x_n]$, sia $f \in k[x_1, \ldots, x_n]$, con $f \neq 0$, e $t \in T$. Supponiamo che

$$f = \sum_{i=1}^{k} m_i p_i,$$

con monomi $m_i \in k[x_1, \ldots, x_n]$, $m_i \neq 0$, e $p_i \in P$, non necessariamente distinti a coppie, allora diciamo che questa è una *t-rappresentazione* di f rispetto a P se

$$max\{LT(m_i p_i) \mid 1 \leq i \leq k\} \leq t.$$

Riformuliamo il criterio F5 nel seguente teorema. Il terzo criterio è un indebolimento del teorema 4.19. Per permettere questo indebolimento è usato il secondo criterio. Se il secondo criterio vale per un elemento g nella base di Gröbner G, allora g è chiamato *ammissibile* [**Fau02**, p.77].

Teorema 6.10 (Criterio dell'algoritmo F5). *Sia $F = \{f_1, \ldots, f_m\}$ e sia $G = \{g_1, \ldots, g_{m_G}\} \subset P$ tale che genera l'ideale I. Definiamo $x^{\gamma_{ij}} = mcm(LM(g_i), LM(g_j))$, per $i, j \in \{1, \ldots, m_G\}$.*
L'insieme G è una base di Gröbner se si verificano i seguenti criteri:

(i) *$F \subset G$;*

(ii) *Per ogni $g \in G$, esiste un elemento del modulo $H = \sum_{i=1}^{m} h_i \mathbf{e}_i \in P^m$, con*

$$H \cdot F = \sum_{i=1}^{m} h_i f_i = g_i,$$

tale che $LMT(H)$ è uguale ad $S(g)$;

(iii) *L'S-polinomio $S(g_i, g_j)$ è zero oppure ha una t-rappresentazione $\sum_{l=1}^{m} b_l f_l$, con*

1. $t < x^{\gamma_{ij}}$,
2. $\mathcal{S}(t) \leq_{P^m} \mathcal{S}(\frac{x^{\gamma_{ij}}}{LT(g_i)} g_i)$ *ed* $\mathcal{S}(t) \leq_{P^m} \mathcal{S}(\frac{x^{\gamma_{ij}}}{LT(g_j)} g_j)$,
3. $\mathcal{S}(b_l f_l) \leq_{P^m} \mathcal{S}(S(g_i, g_j))$, *per* $1 \leq l \leq m$, *per ogni coppia* (i, j) *che soddisfa*:
   - $\mathcal{S}(g_j) <_{P^m} \mathcal{S}(g_i)$,

- se $\mathcal{S}(\frac{x^{\gamma_{ij}}}{LT(g_i)}g_i)) = t_i\mathbf{e}_{i'}$ *e* $\mathcal{S}(\frac{x^{\gamma_{ij}}}{LT(g_j)}g_j)) = t_j\mathbf{e}_{j'}$, *allora* t_i e t_j *non sono divisibili, rispettivamente, da elementi di* $\{LT(f) : f \in \langle f_{i'+1}, \ldots, f_m\rangle\}$ *e* $\{LT(f) : f \in \langle f_{j'+1}, \ldots, f_m\rangle\}$.

Dimostrazione. Vedi la sezione 6.4.2. □

L'algoritmo F5 prende in input una m-pla $F = (f_1, \ldots, f_m)$ di polinomi ed incrementalmente calcola una base di Gröbner intermedia G_i per ogni insieme $\{f_i, \ldots, f_m\}$ da $i = m$ fino ad $i = 1$. Durante l'i-esimo round dell'algoritmo, vengono presi come input la base G_{i+1} ed il polinomio f_i e viene calcolata la base G_i. Mentre calcola l'i-esima base intermedia G_i, l'algoritmo seleziona coppie critiche in ordine crescente. Una determinata coppia (i, j) viene scartata se uno dei due termini t_i, t_j presenti nella firma corrispondente (vedi teorema 6.10, terzo criterio) è riducibile rispetto a G_{i+1}. Inoltre, l'algoritmo preserva la proprietà che gli elementi della nuova base trovata siano ammissibili e prende traccia delle loro firme.

6.4.2 Dimostrazione del criterio dell'algoritmo F5

Per poter dimostrare questo criterio abbiamo bisogno di usare notazioni diverse, incentrate sulla "rappresentazione di un polinomio" e, per fare ciò, useremo quelle che Faugère fornisce in [**Fau02**]. Ridiamo alcune definizioni equivalenti a quelle già usate in precedenza ed altre non ancora mai viste. Sia $(f_1, \ldots, f_m)$ una m-pla di polinomi che sta in P^m e I l'ideale generato da $(f_1, \ldots, f_m)$.

Definizione 6.20. Sia v la funzione di valutazione definita nel seguente modo:

$$\begin{aligned} v : P^m &\longrightarrow P \\ \mathbf{g} &\longmapsto \sum_{i=1}^{m} f_i g_i \end{aligned} \tag{6.8}$$

dove $\mathbf{g} = (g_1, \ldots, g_m)$. Abbiamo che $v(\mathbf{e}_i) = f_i$, e $\mathbf{g} = \sum_{i=1}^m g_i\mathbf{e}_i$.

Definizione 6.21 (Sigizia). Una m-pla $\mathbf{g} = (g_1, \ldots, g_m)$ è chiamata *sigizia* se $v(\mathbf{g}) = 0$, cioè $v(\mathbf{g}) = v(\sum_{i=1}^m g_i\mathbf{e}_i) = \sum_{i=1}^m g_i v(\mathbf{e}_i) = \sum_{i=1}^m g_i f_i = 0$. Quindi $(g_1, \ldots, g_m)$ è una sigizia se $\sum_{i=1}^m g_i f_i = 0$. Chiameremo le sigizie $\mathbf{s}_{i,j} = f_j\mathbf{e}_i - f_i\mathbf{e}_j$ *sigizie principali*. Indichiamo con $Psyz$ il modulo generato dalle sigizie principali.

Definizione 6.22 (Indice). Per ogni $\mathbf{g} \in P^m$ esiste un indice i tale che $\mathbf{g} = \sum_{k=i}^m g_k\mathbf{e}_k$ con $g_i \neq 0$. Questo i sarà indicato come l'*indice* di $\mathbf{g}$, $index(\mathbf{g})$.

Definizione 6.23 (Termine principale di g). Rispetto al nuovo ordinamento $<_{P^m}$, della definizione 6.16, abbiamo $LT(\mathbf{g}) = LT(g_i)\mathbf{e}_i$.

Questa definizione è equivalente alla definizione di termine principale del modulo 6.17.

Definizione 6.24 (Grado di g). Definiamo il *grado* di $\mathbf{g} = \sum_{i=1}^{m} g_i \mathbf{e}_i$, $deg(\mathbf{g})$ come

$$max\{deg(g_i) + deg(f_i), \text{ per } i = 1, \ldots, m\}.$$

Definizione 6.25 (Insieme degli indici dei polinomi di I). Sia $\mathbf{T}_i = \{t\mathbf{e}_i \mid t \in T\}$, così che $LT(\mathbf{g}) \in \mathbf{T}_i$. Allora $\mathbf{T} = \cup_{i=1}^{m} \mathbf{T}_i$ è l'*insieme degli indici di tutti i polinomi* dell'ideale I.

Se definiamo l'insieme delle m-ple $\mathbf{g}$ tali che i termini principali di $\sum_{i=1}^{m} g_i f_i$ siano uguali a t, come $W(t) = \{\mathbf{g} \in P^m \mid LT(v(\mathbf{g})) = t\}$, per $t \in T$, questo certamente può contenere più di un elemento, quindi al fine di sceglierne uno soltanto, diamo la seguente proposizione.

Proposizione 6.11. *Sia ω definita nel seguente modo:*

$$\begin{aligned} \omega : T &\longrightarrow P^m \\ t &\longmapsto min_{<_{P^m}} W(t) \end{aligned}$$

Se $(t_1, t_2) \in T(I) \times T(I)$, allora $LT(\omega(t_1)) \neq LT(\omega(t_2))$ se $t_1 \neq t_2$.

Corollario 6.12. *Per ogni polinomio $p \in I$, definiamo $v_1(p) = LT(\omega(LT(p)))$. Se p_1 e p_2 sono due polinomi in I, con termini principali diversi $LT(p_1) \neq LT(p_2)$, allora $v_1(p_1) \neq v_1(p_2)$.*

Nell'algoritmo F5, $v_1(p)$ rappresenta la *firma* del polinomio p; essa è unica e non dipende dall'ordine con cui vengono fatti i calcoli. Abbiamo bisogno di inserire questo dato all'interno della rappresentazione dei polinomi. Matematicamente la rappresentazione dei polinomi sarà $R = \mathbf{T} \times P$. Se $r = (t\mathbf{e}_i, f) \in R$ allora definiamo:

$$\begin{aligned} poly(r) &= f \in P \\ \mathcal{S}(r) &= t\mathbf{e}_i \in \mathbf{T} \\ index(r) &= i \in \mathbb{N} \end{aligned}$$

Durante l'esecuzione dell'algoritmo la propietà $v_1(poly(r)) = \mathcal{S}(r)$ si conserva.

Definizione 6.26. Diciamo che $r \in R$ è *ammissibile* se esiste $\mathbf{g} \in v^{-1}(poly(r))$ tale che $LT(\mathbf{g}) = \mathcal{S}(r)$.

Siano $0 \neq \lambda \in k$, $v \in T$, $\mathbf{t} = \omega\mathbf{e}_k \in \mathbf{T}$ ed $r = (u\mathbf{e}_i, p) \in R$, definiamo $\lambda r = (u\mathbf{e}_i, \lambda p)$, $v\mathbf{t} = (v\omega)\mathbf{e}_k$ e $vr = (uv\mathbf{e}_i, vp)$. Non definiamo una somma. Estendiamo inoltre alcune definizioni ad elementi di R:

$$\begin{aligned} &\text{per } r \in R LT(r) = LT(poly(r)), \\ &\text{per } r \in R LC(r) = LC(poly(r)), \\ &\text{per } r \in R \text{ e } G \subset P, NF(r, G) = NF(\mathcal{S}(r), NF(poly(r), G)) \end{aligned}$$

dove $NF(poly(r), G)$ è la forma normale del polinomio rappresentato da r modulo G. Diamo una definizione di t-rappresentazione di un polinomio equivalente alla definizione 6.19, solo che adesso invece di trovarci su $k[x_1, \ldots, x_n]$ siamo su R e invece di lavorare con i polinomi, lavoriamo con le rappresentazioni di polinomi $r \in R$.

Definizione 6.27 (t-rappresentazione). Sia $\mathcal{P}$ un sottoinsieme finito di R e $r, t \in R$. Se

$$poly(r) = \sum_{p \in \mathcal{P}} m_p p, m_p \in P,$$

allora diciamo che è una *t-rappresentazione* di r rispetto a $\mathcal{P}$ se

$$LT(t) \geq LT(mp) \text{ ed } \mathcal{S}(r) \geq \mathcal{S}(m_p p),$$

per ogni $p \in \mathcal{P}$. Questa propietà sarà indicata come $f = \mathcal{O}_{\mathcal{P}}(t)$. Useremo la notazione $f = o_{\mathcal{P}}(t)$ se esiste $t^{'} \in R$ tale che $\mathcal{S}(t^{'}) \leq \mathcal{S}(t)$ ed $LT(t^{'}) < LT(t)$ tale che $f = \mathcal{O}_{\mathcal{P}}(t^{'})$.

Definizione 6.28. Diciamo che $r \in R$ è *normalizzato* se $\mathcal{S}(r) = t\mathbf{e}_k$ e t non è top-riducibile da $\langle f_{k+1}, \ldots, f_m \rangle$. Diciamo che $(u, r) \in T \times R$ è normalizzato se ur lo è. Diciamo che una coppia $(r_i, r_j) \in R^2$ è normalizzata se $\mathcal{S}(r_j) <_{P^m} \mathcal{S}(r_i)$, (u_i, r_i) e (u_j, r_j) sono normalizzate, dove

$$u_i = \frac{mcm(LT(r_i), LT(r_j))}{LT(r_i)} \text{ e } u_j = \frac{mcm(LT(r_i), LT(r_j))}{LT(r_j)}.$$

A questo punto siamo in grado di rienunciare il teorema 6.10 con le nuove notazioni

Teorema 6.13. *Sia $F = (f_1, \ldots, f_m)$ una lista di polinomi. Sia $G = (r_1, \ldots, r_{n_G}) \in R^{n_G}$ tale che:*

(i) *$F \subset poly(G)$. Sia $g_i = poly(r_i)$ e $G_1 = (g_1, \ldots, g_{n_G})$.*

(ii) *Tutti gli r_i sono ammissibili, con $i = 1, \ldots, n_G$.*

(iii) *Per ogni $(i, j) \in \{1, \ldots, n_G\}$, tali che le coppie (r_i, r_j) siano normalizzate allora l'S-polinomio $S(g_i, g_j) = o_{G_1}(u_i r_i)$ o zero, dove*

$$u_i = \frac{mcm(LT(g_i), LT(g_j))}{LT(r_i)}.$$

Allora G_1 è una base di Gröbner per I.

Dimostrazione. Sia f un elemento di $I = \langle G_1 \rangle$. Definiamo

$$\mathcal{V} = \left\{ (\mathbf{s}, \sigma) \in P^{n_G} \times \mathcal{S}_n \mid \sum_{i=1}^{n_G} s_i g_{\sigma(i)} = f \text{ e } \mathcal{S}(s_1 r_{\sigma(1)} \geq \mathcal{S}(s_2 r_{\sigma(2)} \geq \cdots \right\}.$$

Definiamo un nuovo ordinamento $(\mathbf{s}, \sigma) <_1 (\mathbf{s}', \sigma')$. Usiamo la notazione $\overline{v} = (\mathcal{S}(s_1 r_{\sigma(1)}), \mathcal{S}(s_2 r_{\sigma(2)}), \ldots)$ e $\overline{v}' = (\mathcal{S}(s_1' r_{\sigma'(1)}), \mathcal{S}(s_2' r_{\sigma'(2)}), \ldots)$. Definiamo $(\mathbf{s}, \sigma) <_1 ((\mathbf{s}', \sigma'))$ se una delle seguenti condizioni si verifica:

(i) $\overline{v} <_{lex} \overline{v}'$

(ii) $\overline{v} = \overline{v}'$ e $max_i LT(s_i g_{\sigma(i)}) < max_i LT(s_i' g_{\sigma'(i)})$

(iii) $\overline{v} = \overline{v}'$ e $t = max_i LT(s_i g_{\sigma(i)}) = max_i LT(s_i' g_{\sigma'(i)})$ e $\#\{i \mid LT(s_i g_{\sigma(i)}) = t\} < \#\{i \mid LT(s_i' g_{\sigma'(i)}) = t\}$.

Sia $\mathbf{s} = min_{<_1} \mathcal{V}$. Supponiamo che σ sia l'identità (riordinando G). Sia $t = max_i LT(s_i g_i)$ ed $\mathcal{I} = \{i \mid LT(s_i g_i) = t\}$, $r = \#\mathcal{I}$. Supponiamo per assurdo che $t > LT(f)$; necessariamente $r \geq 2$. Supponiamo che esista i per il quale (s_i, r_i) non è normalizzato; ciò vuol dire che $\mathcal{S}(r_i) = u\mathbf{e}_k$ ed $LT(s_i)u \in LT(\langle f_{k+1}, \ldots, f_m \rangle)$. Poiché r_i è ammissibile, possiamo scrivere:

$$g_i = \sum_{j=k}^{m} \omega_i f_i, \text{ tale che } LT(\omega_k) = u,$$

$$s_i \omega_k = r + \sum_{\substack{r \in G \\ \mathcal{S}(r) <_{Pm} \mathbf{e}_k}}^{m} \lambda_j poly(g),$$

con $LT(r) < LT(s_i \omega_k)$ ed $LT(\lambda_j poly(g)) \leq LT(u_k u)$. Allora

$$\begin{aligned} f &= \sum_{j \neq i} s_j g_j + s_i \omega_k g_k + \sum_{j=k+1}^{m} s_i \omega_j g_j \\ &= \sum_{j \neq i} s_j g_j + r g_k + \sum_{\substack{r \in G \\ \mathcal{S}(r) <_{Pm} \mathbf{e}_k}} g_k \lambda_j poly(g) + \sum_{j=k+1}^{m} s_i \omega_j g_j. \end{aligned}$$

Questa espressione è $<_1 \mathbf{s}$ e questa è una contraddizione. Dunque tutte le coppie (s_i, r_i) sono normalizzate. Definiamo i seguenti insiemi

$$\omega = max\{\mathcal{S}(s_i, r_i) \mid i \in \mathcal{I}\} \text{ e}$$
$$\mathcal{J} = \{i \in \mathcal{I} \mid \mathcal{S}(s_i, r_i) = \omega\}.$$

Se la cardinalità di $\mathcal{J}$ è maggiore di uno, poiché le r_i sono ammissibili allora per ogni $i \in \mathcal{J}$ abbiamo

$$r_i = \sum_{j=j_0}^{m} \omega_{i,j} f_j, \text{ con } LT(\omega_{i,j_0})\mathbf{e}_{j_0} = \omega.$$

Possiamo scrivere f nel seguente modo:

$$f = \sum_{i<min\mathcal{I}} s_i g_i + \left(\sum_{i\in\mathcal{J}} s_i \omega_{i,j_0} \right) g_{j_0} + \sum_{i\in\mathcal{J}} \sum_{j=j_0+1}^{m} \omega_{i,j} g_j + \sum_{i>max\mathcal{I}} s_i g_i$$

così troviamo un'altra espressione di f che è minore secondo $<_1$ di $\mathbf{s}$. Di conseguenza la cardinalità di $\mathcal{J}$ è uno e sia $k \in \mathcal{J}$ ed $l \in \mathcal{I} \setminus \{k\}$. Per costruzione abbiamo che $\mathcal{S}(s_l r_l) <_{P^m} \mathcal{S}(s_k r_k)$. Scriviamo f come segue:

$$f = s_k g_k - \frac{LC(s_k)}{LC(s_l)} s_l g_l + \left[1 + \frac{LC(s_k)}{LC(s_l)} \right] s_l g_l + \sum_{i\neq k,l} s_i g_i.$$

Siano $m_k = LM(s_k)$ e $m_l = \frac{LC(s_k)}{LC(s_l)} LM(s_l)$ ed $s_i^{'} = s_i - LM(s_i)$. Quindi $t = LT(m_k g_k) = LT(m_l g_l)$, e come conseguenza abbiamo che il minimo comune multiplo tra $LT(g_k)$ ed $LT(g_l)$ divide t, cioè:

$$\begin{aligned} &\text{se } \tau_{k,l} := mcm(LT(g_k), LT(g_l)), \text{ allora} \\ &\quad \tau_{k,l} \text{ divide } t, \end{aligned}$$

equivale a dire che:

$$m_k g_k - m_l g_l = \frac{LC(s_k) t}{\tau_{k,l}} spol(g_k, g_l).$$

Poiché (s_k, g_k) e (s_l, g_l) sono normalizzate deduciamo che (g_k, g_l) è normalizzata, così che

$$\begin{aligned} m_k g_k - m_l g_l &= \frac{t}{\tau_{k,l}} o_G(u_k r_k) \\ &= o_G(s_k r_k), \text{ dove } u_k = \frac{\tau_{k,l}}{LT(r_k)}. \end{aligned}$$

Avremo quindi:

$$f = o_G(s_k r_k) + s_k^{'} g_k - \frac{LC(s_k)}{LC(s_l)} s_l^{'} g_l + \alpha s_l g_l + \sum_{i\neq k,l} s_i g_i,$$

dove

$$s_i^{'} = s_i - LM(s_i)\ (LT(s_i^{'}) < LT(s_i)) \text{ e}$$
$$\alpha = 1 + \frac{LC(s_k)}{LC(s_l)} \in k.$$

Questa è una nuova espressione di f che è minore ($<_1$) di $\mathbf{s}$. Siamo arrivati ad una contraddizione e quindi $t \leq LT(f)$. Possiamo ridurre f con un elemento di G_1; cioè $NF(f, G_1) = 0$. Questo vuol dire che G_1 è una base di Gröbner per I (vedi definizione 6.29). □

Per completeza diamo la definizione di base di Gröbner usata da Faugère in questo articolo, che corrisponde a quella del [**BW93**, p.207].

Definizione 6.29. Un sottoinsieme G di $k[x_1, \ldots, x_n]$ è detto *base di Gröbner* se è un insieme finito, $0 \neq G$, e G soddisfa la condizione

$$NF(f, G) = 0, \text{ per ogni } f \in \langle G \rangle.$$

L'algoritmo F5 sembra lavorare con un tempo esponenziale. In [**BFS03**] gli autori presentano dei risultati riguardanti la complessità legata alla risoluzione di sistemi sovradefiniti su $\mathbb{F}_2$, usando le basi di Gröbner. In particolare loro esaminano l'algoritmo F5, concludendo che la complessità totale per il calcolo delle basi di Gröbner per $m(n)$ equazioni quadratiche in n variabili segue il seguente andamento: se $m \sim Nn$ con N costante, il calcolo è esponenziale, se $n \ll m \ll m^2$ il calcolo è sub-esponenziale e se $m \sim Nn^2$, con N costante, la complessità è polinomiale. Quindi, in quest'ultimo caso, il metodo delle basi di Gröbner porterebbe ad un attacco efficace al cifrario.

Capitolo 7

Conclusioni

Diamo alcuni risultati sulla complessità del calcolo delle basi di Gröbner per un sistema di equazioni polinomiali che deriva da AES.

La complessità della maggior parte degli algoritmi usati per il calcolo delle basi di Gröbner di un ideale è strettamante legata al grado totale dei polinomi intermedi che sono generati durante l'algoritmo. Nel caso peggiore sappiamo ([**CMR**]) che l'algoritmo di Buchberger lavora con un tempo esponenziale doppio, quindi, dato un sistema generico, il comportamento degli algoritmi per trovare le basi di Gröbner sembra troppo complicato.

Osserviamo che il sistema generato dal BES ha una struttura molto regolare; può essere considerato come un sistema di equazioni iterato con sottosistemi simili ripetuti per ogni round. Si può usare la trasformazione $\mathbf{x} \mapsto \mathbf{x}^{254}$ come l'inversione della S-box per eliminare alcune variabili. Inoltre, poiché il sistema comprende le equazioni che mettono in relazione ogni variabile con la sua coniugata, si ha la seguente semplice proposizione ([**CMR**]):

Proposizione 7.1. *Il massimo grado dei polinomi che compaiono all'interno del calcolo di una base di Gröbner per un sistema generato dal BES con n variabili è al massimo n.*

Questo è un limite superiore, e quindi ci si aspetta che in pratica i gradi dei polinomi intermedi siano minori.

Inoltre ci sono delle tecniche comuni usate in crittoanalisi che possono essere affiancate ai metodi di computer-algebra che abbiamo descritto; per esempio, è possibile adattare una tecnica conosciuta come *meet in the middle* che consiste nel considerare due sistemi con metà della dimensione del sistema originale. Invece di risolvere l'intero sistema di equazioni ottenuto da n round, possiamo provare a risolvere due sottosistemi con $n/2$ round, considerando i bits delll'output del round $n/2$ (che è anche l'input del round $n/2+1$) come variabili. Scegliendo un appropiato ordinamento monomiale otteniamo due insiemi di equazioni (ognuno ricoprente

metà delle operazioni di cifratura) che mettono in relazione queste variabili con le sottochiavi del round. Questi due sistemi possono essere combinati con qualche altra equazione relativa alle sottochiavi del round. Questo fornisce un terzo sistema più piccolo che può essere risolto per trovare la chiave di cifratura; così è possibile ridurre la complessità dell'attacco.

È importante notare che in pratica l'attaccante non è interessato a scoprire l'intera soluzione del sistema, ma soltanto le variabili relative alla chiave di cifratura originale. Per i fini crittografici sarebbe sufficiente scoprire solo qualche bits che costituisce la chiave per rendere poco sicuro il cifrario.

In conclusione è possibile usare una combinazione di tecniche crittoanalitiche ed algebriche (basi di Gröbner) per mettere in pratica un attacco con successo, senza necessariamente risolvere l'intero sistema.

Tutto ciò giustifica l'uso e lo sviluppo dei metodi legati alla teoria di Gröbner al fine di crittoanalizzare l'Advanced Encryption Standard.

Indice analitico

Algoritmo
    della divisione, 44, 45
    F4, 83
Applicazione di rappresentazione vettoriale, 80
Attacchi algebrici, 14

Base
    duale, 32
    normale, 32
    polinomiale, 32
Base di Gröbner, 50
    minimale, 63
    ridotta, 63

Campo perfetto, 73
Coefficiente principale, 43
Criterio
    dell'algoritmo F5, 93
    secondo di Buchberger, 65
Crittosistema a chiave segreta, 24

Elementi
    coniugati, 32
    primitivi, 34
Elemento ridondante, 66
Equazioni sparse, 14

Forma normale, 64

Grado totale, 43
    di un monomio, 36
    di un polinomio, 37

Ideale, 38
    d'eliminazione, 70
    monomiale, 46
    proprio, 91
    radicale, 74
    zero dimensionale, 74
Insieme delle coppie distinte di monomi, 79
Installazione di Gebauer e Möller, 90

KeyExpansion, 12

Lemma di Seindenberg, 75

Meet in the middle, 100
Minimo comune multiplo, 54
MixColumns, 11
Modulo delle sigizie, 79
Monomio
    in n variabili, 36
    principale, 43
Multigrado, 43

Omomorfismo di R-moduli, 78
Ordinamento
    lessicografico, 41
    monomiale, 41

Polinomio, 36
    firma di un, 92
    indice di un, 92
    linearizzato, 29
    monico, 43

R-modulo, 78
    noetheriano, 79

Rappresentazione standard, 59
Riduttore, 81

S-polinomio, 54
Selezione normale, 84
Shape lemma, 75
ShiftRows, 10
Sigizia, 94
    di termini principali, 79
    omogenea, 86
    principale, 94
Sistema
    di generatori per un R-modulo, 79
    sovradefinito, 14
Sottomodulo, 78
Spazio affine, 37
Strategia di selezione, 77
SubBytes, 9

t-rappresentazione, 93, 96
Teorema
    delle basi di Hilbert, 49
    di Eliminazione, 70
    di Estensione, 72
    Nullstellensatz di Hilbert, 74
Termine principale, 43
Traccia, 29

Varietà affine, 37
Vettore
    coniugato, 17
    della base canonica, 85

Bibliografia

[**BC03**] A. Biryukov, C. De Canniere. *Block Ciphers and Systems of Quadratic Equations*, in FSE 2003.

[**BFS03**] Magali Bardet, Jean-Charles Faugére, Bruno Salvy. *Complexity of Gröbner basis computation for Semi-regular Overdetermined sequences over* $\mathbb{F}_2$ *with solution in* $\mathbb{F}_2$. Unitè de recherche INRIA Lorraine. Rapport de recherche n. 5049, Decembre 2003.

[**Buch65**] B.A. Buchberger, *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringers nach einem nulldimensionalen Polynomideal*, PhD thesis, Innsbruck, 1965.

[**Buch79**] B.A. Buchberger, *Criterion for detecting unnecessary reductions in the costruction of Gröbner basis*, Proc. EUROSAM 79, Lecture Notes in Computer Science, vol. 72 Springer, Berlin, 1979, pp. 3-21.

[**Buch85**] B.A. Buchberger, *Gröbner bases: an algorithmic method in polynomial ideal theory*, in: Reidel (Ed.), Recent Trends in Multidimensional System Theory, Bose, 1985.

[**BW93**] T. Becker, V. Weispfenning, *Gröbner Bases, A Computational Approach to Commutative Algebra,* Springer-Verlag, 1993.

[**CLO92**] D. Cox, J. Little, and D. Oshea, *Ideals, Varieties and Algorithms,* Springer-Verlag, 1992.

[**CMR**] Carlos Cid, Sean Murphy, Matthew Robshaw. *Computational and Algebraic Aspects of the Advanced Encryption Standard*. Information Security Group, Royal Holloway, University of London.

[**CKM97**] S. Collart, M. Kalkbrener, D. Mall. *Converting Bases with the Gröbner Walk*, Journal of Symbolic Computation 24 (1997), no. 3, p.465-469.

[**CKPS00**] Nicolas Courtois, Alexander Klimov, Jacques Patarin, Adi Shamir. *Efficient Algorithms for Solving Overdefined System of Multivariate Polynomial Equations*. In Eurocrypt 2000, p. 392-407. Springer, 2000.

[**CKR04**] M. Caboara, M. Kreuzer, e L. Robbiano, *Efficiently Computing Minimal Sets of Critical Pairs*, Preprint submitted to Journal of Symbolic Computation (2004)

[**CP**] Nicolas Courtois, Josef Pieprzyk. *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*, Cryptology ePrint Archive, Report 2002/044.

[**CP02**] Nicolas Courtois, Josef Pieprzyk. *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*, In Yulian Zheng, editor, Advances in Cryptology - AIACRYPT 2002, volume 2501 of Lectures Notes in Computer Science, p. 267-287. Springer, 2002.

[**CY04**] Jiun-Ming Chen, Bo-Yin Yang. *Theorethical Analysis of XL over Small Fields*. In Proceedings of the 9th Australasian Conference on Information Security and Privacy, 2004.

[**DR**] J. Daemen, V. Rijmen, *The Design of Rijndael*, Springer 2002.

[**Fau99**] J.C. Faugère, *A New Efficient Algorithm for Computing Gröbner Bases (F4)*, Journal of Pure and Applied Algebra 139 (1999), 61-88.

[**Fau02**] J.C. Faugère, *A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F5)*, Proceedings of the 2002 International Symposium on Symbolic and Algebraic Manipulation, 2002.

[**FGLM93**] J.C. Faugère, P. Gianni, D. Lazard, T. Mora. *Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering*, Journal of Symbolic Computation 16 (1993), p. 329-344.

[**FS01**] N. Ferguson, R. Shroppel, D. Whitinig, *A simple algebraic representation of Rijndael*. In Proceedings of Selected Areas in Cryptography, p.103-111. Springer-Verlag, 2001.

[**Fips**] Federal information processing standards publication 197, advanced encryption standard, November 2001. Avaiable at http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[**GM**] R. Gebauer, H.M. Möller, *On an Installation of Buchberger's Algorithm*, Journal of Symbolic Computation 6 (1988) p. 275-286.

[**GMNRT**] A. Giovini, T. Mora, G. Niesi, L. Robbiano, C. Traverso, *One sugar cube, please, or Selection Strategies in the Buchberger Algorithm*, in: S.M. Watt (Ed.), Proc. 1991 Internat. Symp. on Symbolic and Algebraic Computation, ISSAC' 91, ACM, New York, 1991.

[**KR00**] M. Kreuzer e L. Robbiano, *Computational Commutative Algebra 1*, Springer, 2000.

[**KR04**] M. Kreuzer e L. Robbiano, *Computational Commutative Algebra 2*, Springer, 2004.

[**Laz**] D. Lazard, *Gaussian elimination and resolution of systems of algebraic equations*, in: Proc. EUROCAL 83, Lecture Notes in Computer Science, vol. 162, Springer, Berlin, 1983, p. 146-157.

[**LN**] R. Lidl, H. Niederreiter *Introduction to Finite Fields and their Application*, Cambridge university press, Revised edition 1994.

[**MR00**] S. Murphy, M. Robshaw. *New Observation on Rijndael*, August 2000. Nist AES website.

[**MR02**] S. Murphy, M. Robshaw. *Essential Algebric Structure Within the AES.* In M. Yung, editor, Advances in Cryptology - CRYPTO 2002, volume 2442 of LNCS,p. 1-16. Springer-Verlag, 2002.

[**MR03**] S. Murphy, M. Robshaw. *Comments on the Security of the AES and the XSL-Technique.* Electronic Letters, 39:26-38, 2003.

[**Rijndael**] J. Daemen, V. Rijmen, *AES Proposal: Rijndael*, september 1999. AES algorithm submission.

[**Ros03**] J. Rosenthal *A Polynomial Description of the Rijndael Advanced Encryption Standard*, (2003).

[**Stin02**] D.R. Stinson *Cryptography, Theory and Practice*, Chapman & Hall/CRC 2002