MT5462 Advanced Cipher Systems

Mark Wildon, `mark.wildon@rhul.ac.uk`

Administration:

- Please take the first installment of the notes.
- All handouts will be put on Moodle marked **M.Sc.**.
- **Lectures:** Monday 5pm (ALT3), Friday 11am (McCrea 2-01), Friday 4pm (BLT2).
- **Extra lecture for MT5462:** Thursday 1pm (MFoxSem)
- **Office hours in McCrea LGF 0-25:** Tuesday 3.30pm, Wednesday 11am, Thursday 11.30am (until 12.30pm) or by appointment
- **Relevant seminar:** The Information Security Group Seminar is at 11am Thursdays. To subscribe to the mailing list go to: `www.lists.rhul.ac.uk/mailman/listinfo/isg-research-seminar`.

§1 Revision of fields and polynomials

Definition 1.1

A *field* is a set of elements $\mathbb{F}$ with two operations, $+$ (addition) and $\times$ (multiplication), and two special elements $0, 1 \in \mathbb{F}$ such that $0 \neq 1$ and

(1) $a + b = b + a$ for all $a, b \in \mathbb{F}$;

(2) $0 + a = a + 0 = a$ for all $a \in \mathbb{F}$;

(3) for all $a \in \mathbb{F}$ there exists $b \in \mathbb{F}$ such that $a + b = 0$;

(4) $a + (b + c) = (a + b) + c$ for all $a, b, c \in \mathbb{F}$;

(5) $a \times b = b \times a$ for all $a, b \in \mathbb{F}$;

(6) $1 \times a = a \times 1 = a$ for all $a \in \mathbb{F}$;

(7) for all non-zero $a \in \mathbb{F}$ there exists $b \in \mathbb{F}$ such that $a \times b = 1$;

(8) $a \times (b \times c) = (a \times b) \times c$ for all $a, b, c \in \mathbb{F}$;

(9) $a \times (b + c) = a \times b + a \times c$ for all $a, b, c \in \mathbb{F}$.

If $\mathbb{F}$ is finite, then we define its *order* to be its number of elements.

Exercise: Show, from the field axioms, that if $x \in \mathbb{F}$, then x has a unique additive inverse, and that if $x \neq 0$ then x has a unique multiplicative inverse. Show also that if $\mathbb{F}$ is a field then $a \times 0 = 0$ for all $a \in \mathbb{F}$.

Exercise: Show from the field axioms that if $\mathbb{F}$ is a field and a, $b \in \mathbb{F}$ are such that $ab = 0$, then either $a = 0$ or $b = 0$.

Theorem 1.2

Let p be a prime. The set $\mathbb{F}_p = \{0, 1, \ldots, p - 1\}$ with addition and multiplication defined modulo p is a finite field of order p.

Example 1.3

The addition and multiplication tables for the finite field $\mathbb{F}_4 = \{0, 1, \alpha, 1+\alpha\}$ of order 4 are

$+$	0	1	α	$1+\alpha$
0	0	1	α	$1+\alpha$
1	1	0	$1+\alpha$	α
α	α	$1+\alpha$	0	1
$1+\alpha$	$1+\alpha$	α	1	0

$\times$	1	α	$1+\alpha$
1	1	α	$1+\alpha$
α	α	$1+\alpha$	1
$1+\alpha$	$1+\alpha$	1	α

Definition 1.4

If $f(x) = a_0 + a_1 x + a_2 + \cdots + a_m x^m$ where $a_m \neq 0$, then we say that m is the *degree* of the polynomial f, and write $\deg f = m$. The degree of the zero polynomial is, by convention, -1. We say that a_0 is the *constant term* and a_m is the *leading term*.

Lemma 1.5 (Division algorithm)

Let $\mathbb{F}$ be a field, let $g(x) \in \mathbb{F}[x]$ be a non-zero polynomial and let $g(x) \in \mathbb{F}[x]$. There exist polynomials $s(x), r(x) \in \mathbb{F}[x]$ such that

$$f(x) = s(x)g(x) + r(x)$$

and either $r(x) = 0$ or $\deg r(x) < \deg g(x)$.

We say that $s(x)$ is the *quotient* and $r(x)$ is the *remainder* when $f(x)$ is divided by $g(x)$. Lemma 1.5 will not be proved in lectures. The important thing is that you can compute the quotient and remainder. In Mathematica: `PolynomialQuotientRemainder`, using `Modulus -> p` for finite fields.

Lemma 1.7

Let $\mathbb{F}$ be a field.

(i) *If $f \in \mathbb{F}[x]$ has $a \in \mathbb{F}$ as a root, i.e. $f(a) = 0$, then there is a polynomial $g \in \mathbb{F}[x]$ such that $f(x) = (x - a)g(x)$.*

(ii) *If $f \in \mathbb{F}[x]$ has degree $m \in \mathbb{N}_0$ then f has at most m distinct roots in $\mathbb{F}$.*

(iii) *Suppose that $f, g \in \mathbb{F}[x]$ are non-zero polynomials such that $\deg f, \deg g < t$. If there exist distinct $c_1, \ldots, c_t \in \mathbb{F}$ such that $f(c_i) = g(c_i)$ for each $i \in \{1, \ldots, t\}$ then $f = g$.*

Lemma 1.7

Let $\mathbb{F}$ be a field.

(i) *If $f \in \mathbb{F}[x]$ has $a \in \mathbb{F}$ as a root, i.e. $f(a) = 0$, then there is a polynomial $g \in \mathbb{F}[x]$ such that $f(x) = (x - a)g(x)$.*

(ii) *If $f \in \mathbb{F}[x]$ has degree $m \in \mathbb{N}_0$ then f has at most m distinct roots in $\mathbb{F}$.*

(iii) *Suppose that $f, g \in \mathbb{F}[x]$ are non-zero polynomials such that $\deg f, \deg g < t$. If there exist distinct $c_1, \ldots, c_t \in \mathbb{F}$ such that $f(c_i) = g(c_i)$ for each $i \in \{1, \ldots, t\}$ then $f = g$.*

Part (iii) is the critical result. It says, for instance, that a linear polynomial is determined by any two of its values: when $\mathbb{F}$ is the real numbers $\mathbb{R}$ this should be intuitive—there is a unique line through any two distinct points. Similarly a quadratic is determined by any three of its values, and so on.

Conversely, given t values, there is a polynomial of degree at most t taking these values at any t distinct specified points. This has a nice constructive proof.

Lemma 1.8 (Polynomial interpolation)

Let $\mathbb{F}$ be a field. Let

$$c_1, c_2, \ldots, c_t \in \mathbb{F}$$

be distinct and let $y_1, y_2, \ldots, y_t \in \mathbb{F}$. The unique polynomial $f(x) \in \mathbb{F}[x]$ of degree $< t$ such that $f(c_i) = y_i$ for all i is

$$f(x) = \sum_{i=1}^{t} y_i \frac{\prod_{j \neq i}(x - c_j)}{\prod_{j \neq i}(c_i - c_j)}.$$

§2: Shamir's Secret Sharing Scheme

Example 2.1

Ten people want to know their mean salary. But none is willing to reveal her salary s_i to the others, or to a 'Trusted Third Party'. Instead Person 1 chooses a large number M. She remembers M, and whispers $M + s_1$ to Person 2. Then Person 2 whispers $M + s_1 + s_2$ to Person 3, and so on, until finally Person 10 whispers $M + s_1 + s_2 + \cdots + s_{10}$ to Person 1. Person 1 then subtracts M and can tell everyone the mean $(s_1 + s_2 + \cdots + s_{10})/10$.

Exercise 2.3

In the two person version of the scheme, Person 1 can deduce Person 2's salary from $M + s_1 + s_2$ by subtracting $M + s_1$. Is this a defect in the scheme?

Definition 2.4

Let p be a prime and let $s \in \mathbb{F}_p$. Let $n \in \mathbb{N}$, $t \in \mathbb{N}$ be such that $t \leq n < p$. Let $c_1, \ldots, c_n \in \mathbb{F}_p$ be distinct non-zero elements. In the *Shamir scheme* with n people and *threshold* t, Trevor chooses at random $a_1, \ldots, a_{t-1} \in \mathbb{F}_p$ and constructs the polynomial

$$f(x) = s + a_1 x + \cdots + a_{t-1} x^{t-1}$$

with constant term s. Trevor then issues the *share* $f(c_i)$ to Person i.

Example 2.5

Suppose that $n = 5$ and $t = 3$. Take $p = 7$ and $c_i = i$ for each $i \in \{1, 2, 3, 4, 5\}$. We suppose that $s = 5$. Trevor chooses $a_1, a_2 \in \mathbb{F}_7$ at random, getting $a_1 = 6$ and $a_2 = 1$. Therefore $f(x) = 5 + 6x + x^2$ and the share of Person i is $f(c_i)$, for each $i \in \{1, 2, 3, 4, 5\}$, so

$$\big(f(1), f(2), f(3), f(4), f(5)\big) = (5, 0, 4, 3, 4).$$

Exercise 2.6

Suppose that Person 1, with share $f(1) = 5$, and Person 2, with share $f(2) = 0$, cooperate in an attempt to discover s. Show that for each $z \in \mathbb{F}_7$ there exists a unique polynomial $f_z(x)$ such that $\deg f \leq 2$ and $f(0) = z$, $f_z(1) = 5$ and $f_z(2) = 0$.

Theorem 2.7

In a Shamir scheme with n people, threshold t and secret s, any t people can determine s but any $t-1$ people can learn nothing about s.

Lemma 1.7

Let $\mathbb{F}$ be a field.

(i) *If $f \in \mathbb{F}[x]$ has $a \in \mathbb{F}$ as a root, i.e. $f(a) = 0$, then there is a polynomial $g \in \mathbb{F}[x]$ such that $f(x) = (x - a)g(x)$.*

(ii) *If $f \in \mathbb{F}[x]$ has degree $m \in \mathbb{N}_0$ then f has at most m distinct roots in $\mathbb{F}$.*

(iii) *Suppose that $f, g \in \mathbb{F}[x]$ are non-zero polynomials such that $\deg f, \deg g < t$. If there exist distinct $c_1, \ldots, c_t \in \mathbb{F}$ such that $f(c_i) = g(c_i)$ for each $i \in \{1, \ldots, t\}$ then $f = g$.*

Lemma 1.8 (Polynomial interpolation)

Let $\mathbb{F}$ be a field. Let $c_1, c_2, \ldots, c_t \in \mathbb{F}$ be distinct and let $y_1, y_2, \ldots, y_t \in \mathbb{F}$. The **unique** *polynomial $f(x) \in \mathbb{F}[x]$ of degree $< t$ such that $f(c_i) = y_i$ for all i is*

$$f(x) = \sum_{i=1}^{t} y_i \frac{\prod_{j \neq i}(x - c_j)}{\prod_{j \neq i}(c_i - c_j)}.$$

Exercise 2.8

Suppose Trevor shares $s \in \mathbb{F}_p$ across n computers using the Shamir scheme with threshold t. He chooses the first t computers. They are instructed to exchange their shares; then each computes s and sends it to Trevor. Unfortunately Malcolm has compromised computer 1. Show that Malcolm can both learn s and trick Trevor into thinking his secret is any chosen $s' \in \mathbb{F}_p$.

Example 2.9

The root key for DNSSEC, part of web of trust that guarantees an IP connection really is to the claimed end-point, and not Malcolm doing a Man-in-the-Middle attack, is protected by a secret sharing scheme with $n = 7$ and $t = 5$: search for 'Schneier DNSSEC'.

Exercise 2.10

Take the Shamir scheme with threshold t and evaluation points $1, \ldots, n \in \mathbb{F}_p$ where $p > n$. Trevor has shared two large numbers r and s across n cloud computers, using polynomials f and g so that the shares are $\bigl(f(1), \ldots, f(n)\bigr)$ and $\bigl(g(1), \ldots, g(n)\bigr)$.

(a) How can Trevor secret share $r + s \bmod p$?

(b) How can Trevor secret share $rs \bmod p$? [*Hint: several steps are needed.*]

Note that all the computation has to be done on the cloud!

Remark 2.11

The Reed–Solomon code associated to the parameters p, n, t and the field elements $c_1, c_2, \ldots, c_n$ is the length n code over $\mathbb{F}_p$ with codewords all possible n-tuples

$$\big\{\big(f(c_1), f(c_2), \ldots, f(c_n)\big) : f \in \mathbb{F}_p[x],\ \deg f \leq t-1\big\}.$$

It will be studied in MT5461. By Theorem 2.7, each codeword is determined by any t of its positions. Thus two codewords agreeing in $n-t+1$ positions are equal: this shows the Reed–Solomon code has minimum distance at least $n-t+1$.

Remark 2.11

The Reed–Solomon code associated to the parameters p, n, t and the field elements $c_1, c_2, \ldots, c_n$ is the length n code over $\mathbb{F}_p$ with codewords all possible n-tuples

$$\{(f(c_1), f(c_2), \ldots, f(c_n)) : f \in \mathbb{F}_p[x],\ \deg f \leq t-1\}.$$

It will be studied in MT5461. By Theorem 2.7, each codeword is determined by any t of its positions. Thus two codewords agreeing in $n-t+1$ positions are equal: this shows the Reed–Solomon code has minimum distance at least $n-t+1$.

For simplicity we have worked over a finite field of prime order in this section. Reed–Solomon codes and the Shamir secret sharing scheme generalize in the obvious way to arbitrary finite fields. For example, the Reed–Solomon codes used on compact discs have alphabet the finite field $\mathbb{F}_{2^8}$.

§3 Introduction to Boolean Functions

Recall that $\mathbb{F}_2 = \{0, 1\}$ is the finite field of size 2 whose elements are the *bits* 0 and 1. As usual, $+$ denotes addition in $\mathbb{F}_2$ or in $\mathbb{F}_2^n$.

Definition 3.1

Let $n \in \mathbb{N}$. An n-variable *boolean function* is a function $\mathbb{F}_2^n \to \mathbb{F}_2$.

For example, $f(x, y, z) = xyz + x$ is a Boolean function of the three variables x, y and z, such that $f(1, 0, 0) = 0 + 1 = 1$ and $f(1, 1, 1) = 1 + 1 = 0$. We shall see that Boolean functions are very useful for describing the primitive building blocks of modern stream and block ciphers.

Exercise 3.2

What is a simpler form for $x^2y + xz + z + z^2$?

Exercise 3.3

Let $\mathrm{maj}(x, y, z) = xy + yz + zx$ where, as usual, the coefficients are in $\mathbb{F}_2$. Show that

$$\mathrm{maj}(x, y, z) = \begin{cases} 0 & \text{if at most one of } x, y, z \text{ is } 1 \\ 1 & \text{if at least two of } x, y, z \text{ are } 1. \end{cases}$$

We call $\mathrm{maj} : \mathbb{F}_2^3 \to \mathbb{F}_2$ the *majority vote function*. It is a 3-variable Boolean function.

A modern block cipher has plaintexts and ciphertexts $\mathbb{F}_2^n$ for some fixed n. The encryption functions are typically defined by composing carefully chosen cryptographic primitives over a number of *rounds*.

Example 3.4

(1) Each round of the widely used block cipher AES is of the form $(x, k) \mapsto G(x) + k$ where $+$ is addition in $\mathbb{F}_2^{128}$, $x \in \mathbb{F}_2^{128}$ is the input to the round (derived ultimately from the plaintext) and $k \in \mathbb{F}_2^{128}$ is a 'round key' derived from the key.

The most important cryptographic primitive in the function $G : \mathbb{F}_2^{128} \to \mathbb{F}_2^{128}$ is inversion in the finite field $\mathbb{F}_{2^8}$. The inversion function is highly non-linear and hard to attack. Just for fun, the 255 values of the boolean function sending 0 to 0 and a non-zero x to the bit in position 0 of x^{-1} are shown below, for one natural order on $\mathbb{F}_{2^8}$.

0110101101100111000111010110100000011101100100000100110001011111
1011111110110111101000110000101100111001011111111111010000001010
10100100101110100001000000101010100110100000001000011110110011001
1011000111101000010111000101100111010011001110011100001010101010.

(2) In the block cipher SPECK proposed by NSA in June 2013, the non-linear primitive is modular addition in $\mathbb{Z}/2^m\mathbb{Z}$. As a 'toy' version we take $m = 8$; in practice m is at least 16 and usually 64. Identify $\mathbb{F}_2^8$ with $\mathbb{Z}/2^8\mathbb{Z}$ by writing numbers in their binary form, as on the preliminary problem sheet. For instance, $13 \in \mathbb{Z}/2^8\mathbb{Z}$ has binary form $0000\,1101$ (the space is just for readability) and

$$1010\,1010 \boxplus 0000\,1111 = 1011\,1001$$
$$1000\,0001 \boxplus 1000\,0001 = 0000\,0010$$

corresponding to $170 + 15 = 185$ mod 256 and $129 + 129 = 2$ mod 256. Modular addition is a convenient operation because it is very fast on a computer, but it has some cryptographic weaknesses. In SPECK it is combined with other functions in a way that appears to give a very strong and fast cipher.

One sign that modular addition is weak is that the low numbered bits are 'close to' linear functions. We make this precise in §6 on linear cryptanalysis. For example

$$\begin{aligned}(\ldots, x_2, x_1, x_0) \boxplus (\ldots, y_2, y_1, y_0)\\ = (\ldots, x_2 + y_2 + c_2, x_1 + y_1 + x_0 y_0, x_0 + y_0)\end{aligned}$$

where c_2 is the carry into position 2, defined using the majority vote function by $c_2 = \mathrm{maj}(x_1, y_1, x_0 y_0)$. Unless both x_0 and y_0 are 1, bit 1 is $x_1 + y_1$, a linear function of $(\ldots, x_2, x_1, x_0)$ and $(\ldots, y_2, y_1, y_0)$. By Exercise 4.4, output bit 2 is given by the more complicated polynomial

$$x_2 + y_2 + x_1 y_1 + x_0 x_1 y_0 + x_0 y_0 y_1.$$

This formula can be used for part of Question 5 on Problem Sheet 3: it is the algebraic normal form of the boolean function for bit 2 in modular addition.

A boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ can be defined by its *truth table*, which records for each $x \in \mathbb{F}_2^n$ its image $f(x)$. For example, the boolean functions $\mathbb{F}_2^2 \to \mathbb{F}_2$ of addition and multiplication are shown below:

x	y	$x + y$	xy	$x \wedge y$	$x \vee y$	$x \implies y$
0	0	0	0	F	F	
0	1	1	0	F	T	
1	0	1	0	F	T	
1	1	0	1	T	T	

It is often useful to think of 0 as false and 1 as true. Then xy corresponds to $x \wedge y$, the logical 'and' of x and y, as shown above. The logical 'or' of x and y is denoted $x \vee y$.

A boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ can be defined by its *truth table*, which records for each $x \in \mathbb{F}_2^n$ its image $f(x)$. For example, the boolean functions $\mathbb{F}_2^2 \to \mathbb{F}_2$ of addition and multiplication are shown below:

x	y	$x+y$	xy	$x \wedge y$	$x \vee y$	$x \implies y$
0	0	0	0	F	F	
0	1	1	0	F	T	
1	0	1	0	F	T	
1	1	0	1	T	T	

It is often useful to think of 0 as false and 1 as true. Then xy corresponds to $x \wedge y$, the logical 'and' of x and y, as shown above. The logical 'or' of x and y is denoted $x \vee y$.

Exercise 3.5

Use the true/false interpretation to complete the columns for $x \implies y$. Could you convince a sceptical friend that false statement imply true statements?

Example 3.6

The Toffoli function is a 3-variable boolean function important in quantum computing. It can be defined by

$$\mathrm{toffoli}(x_0, x_1, x_2) = \begin{cases} x_0 & \text{if } x_1 x_2 = 0 \\ \overline{x_0} & \text{if } x_1 x_2 = 1. \end{cases}$$

Here $\overline{x}$ denotes the bitflip of x, defined by $\overline{0} = 1$ and $\overline{1} = 0$. (You will have seen this if you did the Preliminary Problem Sheet.) In the true/false interpretation $\overline{F} = T$ and $\overline{T} = F$.

	x_2	x_1	x_0	$\mathrm{maj}(x_0, x_1, x_2)$	$\mathrm{toffoli}(x_0, x_1, x_2)$	$f_{\{0\}}$	$f_{\{0,2\}}$
$\varnothing$	0	0	0	0	0	0	0
$\{0\}$	0	0	1	0	1	1	0
$\{1\}$	0	1	0	0	0	0	0
$\{0,1\}$	0	1	1	1	1	0	0
$\{2\}$	1	0	0	0	0	0	0
$\{0,2\}$	1	0	1	1	1	0	1
$\{1,2\}$	1	1	0	1	1	0	0
$\{0,1,2\}$	1	1	1	1	0	0	0

	x_2	x_1	x_0	$\mathrm{maj}(x_0, x_1, x_2)$	$\mathrm{toffoli}(x_0, x_1, x_2)$	$f_{\{0\}}$	$f_{\{0,2\}}$
$\varnothing$	0	0	0	0	0	0	0
$\{0\}$	0	0	1	0	1	1	0
$\{1\}$	0	1	0	0	0	0	0
$\{0, 1\}$	0	1	1	1	1	0	0
$\{2\}$	1	0	0	0	0	0	0
$\{0, 2\}$	1	0	1	1	1	0	1
$\{1, 2\}$	1	1	0	1	1	0	0
$\{0, 1, 2\}$	1	1	1	1	0	0	0

The sets on the left record which variables are true. For example, the majority vote function is true on the rows labelled by the sets of sizes 2 and 3, namely, $\{0, 1\}, \{0, 2\}, \{1, 2\}, \{1, 2, 3\}$, and false on the other rows.

Given a subset J of $\{0, \ldots, n-1\}$ we define $f_J : \mathbb{F}_2^n \to \mathbb{F}_2$ by

$$f_J(x) = \bigwedge_{j \in J} x_j \wedge \bigwedge_{j \notin J} \overline{x}_j.$$

In words, f_J is the n-variable boolean function whose truth table has a unique 1 (or true) in the row labelled J. For instance $f_{\{0\}}(x_0, x_1, x_2) = x_0 \wedge \overline{x}_1 \wedge \overline{x}_2$ and $f_{\{0,2\}}(x_0, x_1, x_2) = x_0 \wedge \overline{x}_1 \wedge x_2$ are shown above.

Exercise 3.7

(i) For what set J do we have

$$\mathrm{toffoli} = f_{\{0\}} \vee f_{\{0,1\}} \vee f_{\{0,2\}} \vee f_J?$$

(ii) Express the majority vote function in the form above.

(iii) Find a way to complete the right-hand side in

$$\mathrm{maj}(x) = (x_0 \wedge x_1 \wedge \overline{x}_2) \vee (x_0 \wedge \overline{x}_1 \wedge x_2) \vee (\overline{x}_0 \wedge x_1 \wedge x_2) \vee (\ldots).$$

Recall that [**Typo in printed notes: $i \in J$ should be $j \in J$**]

$$f_J(x) = \bigwedge_{j \in J} x_j \wedge \bigwedge_{j \notin J} \overline{x}_j.$$

	x_2	x_1	x_0	$\mathrm{maj}(x_0, x_1, x_2)$	$\mathrm{toffoli}(x_0, x_1, x_2)$	$f_{\{0\}}$	$f_{\{0,2\}}$
$\varnothing$	0	0	0	0	0	0	0
$\{0\}$	0	0	1	0	1	1	0
$\{1\}$	0	1	0	0	0	0	0
$\{0,1\}$	0	1	1	1	1	0	0
$\{2\}$	1	0	0	0	0	0	0
$\{0,2\}$	1	0	1	1	1	0	1
$\{1,2\}$	1	1	0	1	1	0	0
$\{0,1,2\}$	1	1	1	1	0	0	0

We saw in Exercise 3.7 that

(a) $\mathrm{toffoli} = f_{\{0\}} \vee f_{\{0,1\}} \vee f_{\{0,2\}} \vee f_{\{1,2\}}$;

(b) $\mathrm{maj} = f_{\{0,1\}} \vee f_{\{0,2\}} \vee f_{\{1,2\}} \vee f_{\{1,2,3\}}$;

(c) $\mathrm{maj}(x_0, x_1, x_2) = (x_0 \wedge x_1 \wedge \overline{x}_2) \vee (x_0 \wedge \overline{x}_1 \wedge x_2) \vee (\overline{x}_0 \wedge x_1 \wedge x_2) \vee (x_0 \wedge x_1 \wedge x_2)$.

How would you express the boolean function $g(x_0, x_1, x_2)$ that is true if and only if $x_0 = x_1 = x_2$ as a disjunction ($\bigvee$) of the f_J?

Theorem 3.8 (Disjunctive Normal Form)

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a boolean function.

(i) *Suppose that the truth table of f has 1 in the rows labelled by the sets J for $J \in \mathcal{T}$. Then*

$$f = \bigvee_{J \in \mathcal{T}} f_J.$$

(ii) *If $\mathcal{T} \neq \mathcal{T}'$ then $\bigvee_{J \in \mathcal{T}} f_J \neq \bigvee_{J \in \mathcal{T}'} f_J$.*

This theorem says that every boolean function f has a unique *disjunctive normal form* $\bigvee_{J \in \mathcal{T}} f_J$, for a suitable set $\mathcal{T}$.

Corollary 3.9

There are 2^{2^n} n-variable boolean functions.

Exercise 3.10

By Corollary 3.9, there are 16 truth tables of 2-variable boolean functions. Using the true/false notation, the 8 for which $f(F, F) = F$ are shown below. What is a suitable label for the rightmost column? What are the disjunctive normal forms of these 8 functions? What is a concise way to specify the remaining 8 functions?

	x_1	x_0	$x_0 \vee x_1$	x_0	x_1	$x_0 + x_1$	$x_0 \wedge x_1$	$x_0 \wedge \overline{x}_1$	$\overline{x}_0 \wedge x_1$	??
$\varnothing$	F	F	F	F	F	F	F	F	F	F
$\{0\}$	F	T	T	T	F	T	F	T	F	F
$\{1\}$	T	F	T	F	T	T	F	F	T	F
$\{0,1\}$	T	T	T	T	T	F	T	F	F	F

In $\mathbb{F}_2$ we have $0^2 = 0$ and $1^2 = 1$. Therefore the Boolean functions $f(x_1) = x_1^2$ and $f(x_1) = x_1$ are equal. Hence, as seen in Exercise 3.2, multivariable polynomials over $\mathbb{F}_2$ do not need squares or higher powers of the variables. Similarly, since $2x_1 = 0$, the only coefficients needed are the bits 0 and 1. For instance, $x_0 + x_0x_2^2x_3^3 + x_0^2 + x_2x_3$ is the same Boolean function as $x_2x_3 + x_0x_2x_3$.

Given $I \subseteq \{0, 1, \ldots, n-1\}$, let

$$x_I = \prod_{i \in I} x_i.$$

We say the x_I are *boolean monomials*. By definition (or convention if you prefer), $x_\varnothing = 1$. For example, $x_{\{1,2\}} = x_1x_2$. It is one of the three boolean monomial summands of

$$\mathrm{maj}(x_0, x_1, x_2) = x_0x_1 + x_1x_2 + x_2x_0.$$

The functions f_J so useful for proving Theorem 3.8 have a particularly simple form as polynomials:

$$f_J(x) = \prod_{j \in J} x_j \prod_{j \notin J} \overline{x}_j.$$

Exercise 3.11

Define the 3-variable Boolean function

$$g(x_0, x_1, x_2) = \begin{cases} 1 & \text{if } x_0 = x_1 = x_2 \\ 0 & \text{otherwise.} \end{cases}$$

Express g as sum of boolean monomials. The negation of g is defined by $\overline{g} = \overline{g(x)}$. What is $\overline{g}$ as a sum of boolean monomials?

Similarly you can use the truth table on page 10 to express the Toffoli function and its negation as a sum of boolean monomials.

It is only a small generalization of Exercise 3.11 to prove the following theorem.

Theorem 3.12

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be an n-variable Boolean function.

(a) *There exist unique coefficients $b_J \in \{0,1\}$, one for each $J \subseteq \{0,1,\ldots,n-1\}$ such that*
$$f = \sum_{I \subseteq \{0,1,\ldots,n\}} b_J f_J.$$

(b) *There exist unique coefficients $c_I \in \{0,1\}$, one for each $I \subseteq \{0,1,\ldots,n-1\}$, such that*
$$f = \sum_{I \subseteq \{0,1,\ldots,n-1\}} c_I x_I.$$

It is only a small generalization of Exercise 3.11 to prove the following theorem.

Theorem 3.12

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be an n-variable Boolean function.

(a) *There exist unique coefficients $b_J \in \{0, 1\}$, one for each $J \subseteq \{0, 1, \ldots, n-1\}$ such that*
$$f = \sum_{I \subseteq \{0,1,\ldots,n\}} b_J f_J.$$

(b) *There exist unique coefficients $c_I \in \{0, 1\}$, one for each $I \subseteq \{0, 1, \ldots, n-1\}$, such that*
$$f = \sum_{I \subseteq \{0,1,\ldots,n-1\}} c_I x_I.$$

The expression for f in (b) is called the *algebraic normal form* of f.

As shorthand, we write $[x_I]f$ for the coefficient of x_I in the boolean function f. Thus $f = \sum_{I \subseteq \{1,\ldots,n\}} ([x_I]f) x_I$ is the algebraic normal form of f.

Exercise 3.13

Let $f(x, y, z) = 1 + x + xz + yz + xyz$ and let

$$g(x, y, z) = f(0, y, z) + f(1, y, z)$$

and let

$$\begin{aligned} h(x, y, z) &= g(x, 0, z) + g(x, 1, z) \\ &= f(0, 0, z) + f(1, 0, z) + f(0, 1, z) + f(1, 1, z) \end{aligned}$$

Find the algebraic normal form of g and h. What is the connection between $g(0, 0, 0)$ and $h(0, 0, 0)$ and $[x]f$, $[xy]f$? How would you find $[xz]f$ and $[xyz]f$ by this method?

Exercise 3.13

Let $f(x,y,z) = 1 + x + xz + yz + xyz$ and let

$$g(x,y,z) = f(0,y,z) + f(1,y,z)$$

and let

$$\begin{aligned} h(x,y,z) &= g(x,0,z) + g(x,1,z) \\ &= f(0,0,z) + f(1,0,z) + f(0,1,z) + f(1,1,z) \end{aligned}$$

Find the algebraic normal form of g and h. What is the connection between $g(0,0,0)$ and $h(0,0,0)$ and $[x]f$, $[xy]f$? How would you find $[xz]f$ and $[xyz]f$ by this method?

Proposition 3.14

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be an n-variable Boolean function. Then

$$[x_I]f = \sum f(z_0, \ldots, z_{n-1})$$

where the sum is over all $z_0, \ldots, z_{n-1} \in \{0,1\}$ such that $\{j : z_j = 1\} \subseteq I$.

Coulter McDowell Lecture 2019

- Prof. Jeffrey Vaaler (University of Texas at Austin)
  Minkowski's convex body theorem and some of its applications
- Tuesday 5th November 6.15pm
- Windsor Building Auditorium, 6.15pm

Public lecture, suitable for A-level students. Refreshments afterwards. Stefanie Gerke (and I) will be around to say hello.

Pure Mathematics Seminar

- Prof. Kevin Buzzard (Imperial College)
  The future of mathematics?
- Wednesday 6th November 2pm
- Munro Fox Lecture Room

Kevin is leading a team of M.Sc. students to formalize mathematics using a computer theorem prover. From his abstract

> *I personally believe that Lean is part of what will become a paradigm shift in the way humans do mathematics, and that people who do not switch will ultimately be left behind.*

§4 The Discrete Fourier Transform

Given $x \in \mathbb{F}_2$ we define $(-1)^x$ by regarding x as an ordinary integer. Thus $(-1)^0 = 1$ and $(-1)^1 = -1$. Given an n-variable boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ we define $(-1)^f : \mathbb{F}_2^n \to \{-1, 1\}$ by $(-1)^f(x) = (-1)^{f(x)}$.

Definition 4.1

Let $f, g : \mathbb{F}_2^n \to \mathbb{F}$ be Boolean functions. We define the *correlation* between f and g by

$$\mathrm{corr}(f, g) = \frac{1}{2^n} \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)} (-1)^{g(x)}.$$

The summand $(-1)^{f(x)}(-1)^{g(x)}$ is 1 when $f(x) = g(x)$ and -1 when $f(x) = -g(x)$. Hence

$$\mathrm{corr}(f, g) = \frac{c_{\mathrm{same}} - c_{\mathrm{diff}}}{2^n}$$

where

$$c_{\mathrm{same}} = \left|\{x \in \mathbb{F}_2^n : f(x) = g(x)\}\right|, \; c_{\mathrm{diff}} = \left|\{x \in \mathbb{F}_2^n : f(x) \neq g(x)\}\right|.$$

Given $T \subseteq \{0, 1, \ldots, n-1\}$, define $L_T : \mathbb{F}_2^n \to \mathbb{F}_2$ by

$$L_T(x) = \sum_{t \in T} x_t.$$

For example, $L_{\{i\}}(x_0, x_1, \ldots, x_{n-1}) = x_i$ returns the entry in position i and $L_\varnothing(x) = 0$ is the zero function.

Exercise 4.2

Find all the linear 3-variable boolean functions. Which 3-variable boolean functions are uncorrelated with the zero function?

Lemma 4.3

The linear functions $\mathbb{F}_2^n \to \mathbb{F}$ are precisely the $L_T : \mathbb{F}_2^n \to \mathbb{F}_2$ for $T \subseteq \{0, 1, \ldots, n-1\}$. If $S, T \subseteq \{0, 1, \ldots, n-1\}$ then

$$\mathrm{corr}(L_S, L_T) = \begin{cases} 1 & \text{if } S = T \\ 0 & \text{otherwise.} \end{cases}$$

Recall that if $T \subseteq \{0, 1, \ldots, n-1\}$ then $L_T : \mathbb{F}_2^n \to \mathbb{F}_2$ is the linear n-variable boolean function defined by

$$L_T(x) = \sum_{t \in T} x_t.$$

Example 4.4

Let $\mathrm{maj} : \mathbb{F}_2^3 \to \mathbb{F}_2$ be the majority vote function from Exercise . We have [**corrected off-by-one error**]

$$\mathrm{corr}(\mathrm{maj}, L_T) = \begin{cases} \frac{1}{2} & \text{if } T = \{0\}\{1\}, \{2\} \\ -\frac{1}{2} & \text{if } T = \{0, 1, 2\} \\ 0 & \text{otherwise.} \end{cases}$$

We define an inner product on the vector space W of functions $\mathbb{F}_2^n \to \mathbb{R}$ by

$$\langle \theta, \phi \rangle = \frac{1}{2^n} \sum_{x \in 2^n} \theta(x)\phi(x).$$

If f and g are n-variable boolean functions then

$$\langle (-1)^f, (-1)^g \rangle = \mathrm{corr}(f, g).$$

Exercise 4.5

(i) Let $\theta \in W$. Check that, as required for an inner product, $\langle \theta, \theta \rangle \geq 0$ and that $\langle \theta, \theta \rangle = 0$ if and only if $\theta(x) = 0$ for all $x \in \mathbb{F}_2^n$.

(ii) Show that if $n = 2$ then W is 4-dimensional. What is $\dim W$ in general?

We define an inner product on the vector space W of functions $\mathbb{F}_2^n \to \mathbb{R}$ by

$$\langle \theta, \phi \rangle = \frac{1}{2^n} \sum_{x \in 2^n} \theta(x)\phi(x).$$

If f and g are n-variable boolean functions then

$$\langle (-1)^f, (-1)^g \rangle = \mathrm{corr}(f, g).$$

Exercise 4.5

(i) Let $\theta \in W$. Check that, as required for an inner product, $\langle \theta, \theta \rangle \geq 0$ and that $\langle \theta, \theta \rangle = 0$ if and only if $\theta(x) = 0$ for all $x \in \mathbb{F}_2^n$.

(ii) Show that if $n = 2$ then W is 4-dimensional. What is $\dim W$ in general?

Writing functions $f \in W$ like columns of truth tables $\begin{pmatrix} f(00) \\ f(01) \\ f(10) \\ f(11) \end{pmatrix}$, we have

$$(-1)^{L_\varnothing} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \text{and} \quad (-1)^{L_{\{1\}}} = \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} \quad \text{and so on.}$$

Reminder of Inner Product Spaces

- Any orthonormal set is linearly independent: for instance, with three orthonormal vectors u, v, w, if $\alpha u + \beta v + \gamma w = 0$ then taking the inner product with u we get

$$0 = \langle 0, u \rangle = \langle \alpha u + \beta v + \gamma w, u \rangle = \alpha.$$

- If $x = \alpha u + \beta v + \gamma w$ where u, v, w are orthonormal then $\langle x, x \rangle = \alpha^2 + \beta^2 + \gamma^2$.

The inner product on the vector space W of functions $\mathbb{F}_2^n \to \mathbb{R}$ is defined by

$$\langle \theta, \phi \rangle = \frac{1}{2^n} \sum_{x \in 2^n} \theta(x)\phi(x).$$

We saw that $\langle (-1)^f, (-1)^g \rangle = \mathrm{corr}(f, g)$ for n-variable boolean functions f and g.

Theorem 4.6 (Discrete Fourier Transform)

(a) *The functions $(-1)^{L_T}$ for $T \subseteq \{0, 1, \ldots, n-1\}$ are an orthonormal basis for the vector space W of functions $\mathbb{F}_2^n \to \mathbb{R}$.*

(b) *Let $\theta : \mathbb{F}_2^n \to \mathbb{R}$. Then*

$$\theta = \sum_{T \subseteq \{0,1,\ldots,n-1\}} \langle \theta, (-1)^{L_T} \rangle (-1)^{L_T}.$$

(c) *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. Then*

$$(-1)^f = \sum_{T \subseteq \{0,1,\ldots,n-1\}} \mathrm{corr}(f, L_T)(-1)^{L_T}.$$

Corollary 4.7

Let f be an n-variable boolean function. Then

$$\sum_{T \subseteq \{0,1,\ldots,n-1\}} \mathrm{corr}(f, L_T)^2 = 1.$$

Since there are 2^n linear functions (corresponding to the 2^n subsets of $\{0, 1, \ldots, n-1\}$), it follows that any n-variable boolean function f has a squared correlation of at least $1/2^n$.

Example 4.8

(1) Let $f(x_0, x_1, x_2) = x_0 x_1 x_2$. We have $\mathrm{corr}(f, L_\varnothing) = \frac{3}{4}$, $\mathrm{corr}(f, L_{\{0\}}) = \frac{1}{4}$, $\mathrm{corr}(f, L_{\{0,1\}}) = -\frac{1}{4}$ and $\mathrm{corr}(f, L_{\{0,1,2\}}) = \frac{1}{4}$. By Theorem 4.6(c) and symmetry, the Discrete Fourier Transform of f is

$$(-1)^f = \frac{3}{4} + \frac{1}{4} \sum_{\substack{T \subseteq \{0,1,2\} \\ T \neq \varnothing}} (-1)^{|T|-1} (-1)^{L_T}.$$

We will check Parseval's Theorem holds.

Example 4.8 [continued]

(2) *Exercise:* Consider the 2-variable boolean function $f(x_0, x_1) = x_0 x_1$. Find its correlations with the four linear functions $L_\varnothing(x_0, x_1) = 1$, $L_{\{0\}}(x_0, x_1) = x_0$, $L_{\{1\}}(x_0, x_1) = x_1$, $L_{\{0,1\}}(x_0, x_1) = x_1 + x_2$ and deduce that

$$(-1)^{x_0 x_1} = \tfrac{1}{2}(-1)^{L_\varnothing} + \tfrac{1}{2}(-1)^{L_{\{0\}}} + \tfrac{1}{2}(-1)^{L_{\{1\}}} - \tfrac{1}{2}(-1)^{L_{\{0,1\}}}$$

(3) Let $b(x_0, x_1, y_0, y_1) = x_0 y_0 + x_1 y_1$. We shall use MATHEMATICA to show that $\mathrm{corr}(b, L_T) = \pm\frac{1}{4}$ for every $T \subseteq \{0, 1, 2, 3\}$. By the remark following Corollary 4.7, this function achieves the cryptographic ideal of having all correlations as small (in absolute value) as possible.

Bent Functions

An n-variable boolean function such as b where the correlations all have absolute value $1/\sqrt{2^n}$ is called a *bent function*. Since correlations are rational numbers, bent functions exist only for even n. Many different constructions have been found and applied in cryptography.

Piling-Up Lemma

Lemma 4.9 (Piling-up Lemma)

Let f be an m-variable boolean function of $x_0, \ldots, x_{m-1}$ and let g be an n-variable boolean function of $y_0, \ldots, y_{n-1}$. Define $f + g$ by

$$(f+g)(x_0, \ldots, x_{m-1}, y_0, \ldots, y_{n-1}) = f(x_0, \ldots, x_{m-1}) + g(y_0, \ldots, y_{n-1}).$$

Given $S \subseteq \{0, \ldots, m-1\}$ and $T \subseteq \{0, \ldots, n-1\}$, let $L_{(S,T)}(x, y) = L_S(x) + L_T(y)$. The $L_{(S,T)}$ are all linear functions of the $m + n$ variables and

$$\mathrm{corr}(f + g, L_{(S,T)}) = \mathrm{corr}(f, L_S)\,\mathrm{corr}(g, L_T).$$

For instance the Piling-up Lemma implies that $x_0 y_0 + \cdots + x_{m-1} y_{m-1}$ is a bent function for all m, generalizing Example 4.8.

§5 The Berlekamp–Massey Algorithm

Example 5.1

By Question 4 on Sheet 5, the sum u of the keystreams of the LFSR with taps $\{3,4\}$ and width 4 and the LFSR with taps $\{2,3\}$ and width 3, using keys 0001 and 001, has period 105.

$$u_i = (\underset{0}{0}, \underset{1}{0}, \underset{2}{1}, \underset{3}{1}, \underset{4}{1}, \underset{5}{1}, \underset{6}{0}, \underset{7}{1}, \underset{8}{0}, \underset{9}{0}, \underset{0}{0}, \underset{1}{0}, \underset{2}{0}, \underset{3}{0}, \underset{4}{1}, \underset{5}{0}, \underset{6}{1}, \underset{7}{0}, \underset{8}{0}, \underset{9}{1}, \ldots)$$

The output of the Berlekamp–Massey algorithm applied to the first n terms $u_0 \ldots u_{n-1}$ for $n \geq 6$ is below. No change for $n = 7, 8, 12$.

n	width	feedback polynomial	taps	m
6	3	$1 + z$	$\{1\}$	2
9	4	$1 + z + z^4$	$\{1, 4\}$	6
10	6	$1 + z + z^3$	$\{1, 3\}$	9
11	6	$1 + z^2 + z^3 + z^5$	$\{2, 3, 5\}$	9
≥ 13	7	$1 + z^2 + z^4 + z^5 + z^7$	$\{2, 4, 5, 7\}$	12

Example 5.1 [continued]

For instance, the first 10 terms $u_0 u_1 \dots u_9$ are generated by the LFSR of width 6 with feedback polynomial $1 + z + z^3$; its taps are $\{1, 3\}$. Taking as the key $u_0 u_1 u_2 u_3 u_4 u_5 = 001111$, the first 30 terms of the keystream are:

$$\begin{aligned} k_i =&(0,0,1,1,1,1,0,1,0,0,1,1,1,0,1,0,0,1,1,1,\dots) \\ u_i =&(0,0,1,1,1,1,0,1,0,0,0,0,0,0,1,0,1,0,0,1,\dots) \\ &\;\;0\;\;1\;\;2\;\;3\;\;4\;\;5\;\;6\;\;7\;\;8\;\;9\;\;0\;\;1\;\;2\;\;3\;\;4\;\;5\;\;6\;\;7\;\;8\;\;9 \end{aligned}$$

Since $k_{10} \neq u_{10}$, running the Berlekamp–Massey algorithm on the first 11 bits $u_0 \dots u_9 u_{10}$ gives a different LFSR. (The width stays as 6, but the taps change to $\{2, 3, 5\}$.) The new LFSR generates $u_0 \dots u_9 u_{10} u_{11}$, so is also correct for the first 12 bits. This is why there is no change for $n = 12$.

For all $n \geq 13$ the output of the algorithm is the LFSR of width 7 and feedback polynomial $1 + z^2 + z^4 + z^5 + z^7$; as suggested on the problem sheet, this may also be found by the method of annihilators.

Preliminaries

Fix throughout a binary stream

$$u_0 u_1 u_2 \dots .$$

Let $U_n(z) = u_0 + u_1 z + \cdots + u_{n-1} z^{n-1}$ be the polynomial recording the first n terms. Recall from §1 that the degree of a non-zero polynomial $h(z)$ is its highest power of z.

Lemma 5.3

The word $u_0 u_1 \dots u_{n-1}$ is the output of the LFSR with width ℓ and taps T if and only if $U_n(z) g_{T_n}(z) = h(z) + z^n r(z)$ for some polynomials $h(z)$ and $r(z)$ with $\deg h < \ell$.

Example of Lemma 5.3

Example 5.4

Let $u = (0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0) = u_0 \dots u_{12}$ be the first 13 entries of the keystream in Example 5.1. The first 12 entries $u_0 \dots u_{11}$ are generated by the LFSR of width 6 with taps $\{2, 3, 5\}$. Correspondingly, by the 'if' direction of Lemma 5.3,

$$
\begin{aligned}
(z^2 + z^3 + z^4 + z^5 + z^7)&g_{\{2,3,5\}}(z) \\
&= (z^2 + z^3 + z^4 + z^5 + z^7)(1 + z^2 + z^3 + z^5) \\
&= z^2 + z^3 + z^5 + z^{12} \\
&= h(z) + z^{12} r(z)
\end{aligned}
$$

where $h(z) = z^2 + z^3 + z^5$ and $r(z) = 1$. This equation also shows that the 'only if' direction fails to hold when $n = 13$ since z^{12} is not of the form $z^{13} r(z)$. Correspondingly, by the 'only if' direction of Lemma 5.3, the LFSR generates $(0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, \mathbf{1})$ rather than u.

At step n of the Berlekamp–Massey algorithm we have two LFSRs:

- An LFSR F_m of width ℓ_m with taps T_m, generating
$$u_0 u_1 \ldots u_{m-1} \overline{u}_m \ldots .$$
- An LFSR F_n of width ℓ_n with taps T_n, where $n > m$, generating
$$u_0 u_1 \ldots u_{m-1} u_m \ldots u_{n-1}.$$

Thus F_m is correct for the first m positions, and then wrong, since it generates $\overline{u}_m$ rather than u_m. If F_n generates $u_0 u_1 \ldots u_{m-1} u_m \ldots u_{n-1} u_n$ then case (a) applies and the algorithm returns F_n. The next proposition deals with case (b), when F_n outputs $\overline{u}_n$ rather than u_n.

Proposition 5.5

With the notation above, suppose that the LFSR F_n generates $u_0 u_1 \ldots u_{n-1} \overline{u}_n$. The LFSR with feedback polynomial

$$z^{n-m} g_{T_m}(z) + g_{T_n}(z)$$

and width $\max(n - m + \ell_m, \ell_n)$ *generates* $u_0 u_1 \ldots u_{n-1} u_n$.

Example 5.6

Take the keystream $k_0 k_1 \dots k_9$ of length 10 shown below:

$$\underset{0}{(1},\underset{1}{1},\underset{2}{1},\underset{3}{0},\underset{4}{1},\underset{5}{0},\underset{6}{1},\underset{7}{0},\underset{8}{0},\underset{9}{0)}.$$

The LFSR F_6 of width $\ell_6 = 3$ and taps $T_6 = \{1, 3\}$ generates the keystream

$$\underset{0}{(1},\underset{1}{1},\underset{2}{1},\underset{3}{0},\underset{4}{1},\underset{5}{0},\underset{6}{0},\underset{7}{1},\underset{8}{1},\underset{9}{1)}.$$

The LFSR F_7 of width $\ell_7 = 4$ and taps $T_7 = \{1, 4\}$ generates the keystream

$$\underset{0}{(1},\underset{1}{1},\underset{2}{1},\underset{3}{0},\underset{4}{1},\underset{5}{0},\underset{6}{1},\underset{7}{1},\underset{8}{0},\underset{9}{0)}.$$

Note that F_7 is wrong in position 7.

Example 5.6 [continued]

Using Proposition 5.5, taking $m = 6$ and $n = 7$ we compute

$$\begin{aligned} z^{n-m} g_{T_m} + g_{T_n}(z) &= z^{7-6} g_{\{1,3\}}(z) + g_{\{1,4\}}(z) \\ &= z(1 + z + z^3) + (1 + z + z^4) \\ &= 1 + z^2. \end{aligned}$$

This is the feedback polynomial of the LFSR F_8 with taps $T_8 = \{2\}$ and width $\ell_8 = n - m + \ell_m = 7 - 6 + 3 = 4$. As expected this generates

$$(\underset{0}{1}, \underset{1}{1}, \underset{2}{1}, \underset{3}{0}, \underset{4}{1}, \underset{5}{0}, \underset{6}{1}, \underset{7}{0}, \underset{8}{1}, \underset{9}{0}).$$

correct for the first 8 positions. (And then wrong for u_8.) Although the only tap in $\{2\}$ is 2, we still have to take the width of F_8 to be 4 (or more), to get the first 8 positions correct.

Continuing Example 5.6

Exercise 5.7

Continuing from the example, apply Proposition 5.5 taking $n = 8$, $m = 6$, and F_8 and F_6 as in Example 5.6. You should get the LFSR F_9 with taps $\{3, 5\}$ generating

$$\underset{0}{(1}, \underset{1}{1}, \underset{2}{1}, \underset{3}{0}, \underset{4}{1}, \underset{5}{0}, \underset{6}{1}, \underset{7}{0}, \underset{8}{0}, \underset{9}{0}).$$

which is the full keystream. The width is now $8 - 6 + 3 = 5$; since 5 is a tap, this is the minimum possible width for these taps.

Berlekamp–Massey algorithm

Let c be least such that $u_c \neq 0$. The algorithm defines LFSRs $F_c, F_{c+1}, \ldots$ so that each F_n has width ℓ_n and taps T_n and generates the first n positions of the keystream: $u_0, \ldots, u_{n-1}$.

- [Initialization] Set $T_c = \varnothing$, $\ell_c = 0$, $T_{c+1} = \varnothing$ and $\ell_{c+1} = c + 1$. Set $m = c$.
- [Step] We have an LFSR F_n with taps T_n of width ℓ_n generating $u_0, \ldots, u_{n-1}$ and an LFSR F_m generating $u_0, \ldots, u_{m-1}, \overline{u}_m$.
  (a) If F_n generates $u_0, \ldots, u_{n-1}, u_n$ then set $T_{n+1} = T_n$, $\ell_{n+1} = \ell_n$. This defines F_{n+1} with $F_{n+1} = F_n$. Keep m as it is.
  (b) If F_n generates $u_0, \ldots, u_{n-1}, \overline{u}_n$, calculate

$$g(z) = z^{n-m} g_{T_m}(z) + g_{T_n}(z)$$

  where, as usual, g_{T_m} and g_{T_n} are the feedback polynomials. Define T_{n+1} so that $g(z) = 1 + \sum_{t \in T_{n+1}} z^t$. Set

$$\ell_{n+1} = \max(\ell_n, n + 1 - \ell_n).$$

  If $\ell_{n+1} > \ell_n$, update m to n, otherwise keep m as it is.

Thus m changes if and only if the width increases in step (b).

Example 5.8

We apply the Berlekamp–Massey algorithm to the keystream $(1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1)$ from Example 5.6 extended by one extra bit $u_{10} = 1$. After initialization we have $T_0 = \varnothing$, $\ell_0 = 0$, $T_1 = \varnothing$, $\ell_1 = 1$. Case (a) applies in each step n for $n \in \{2, 4, 5, 9\}$. The table below shows the steps when case (b) applies.

n	T_n	ℓ_n	m	T_m	$n-m$	T_{n+1}	ℓ_{n+1}
1	$\varnothing$	1	0	$\varnothing$	1	$\{1\}$	1
3	$\{1\}$	1	0	$\varnothing$	3 **[corr.]**	$\{1,3\}$	3
6	$\{1,3\}$	3	3	$\{1\}$	3	$\{1,4\}$	4
7	$\{1,4\}$	4	6	$\{1,3\}$	1	$\{2\}$	4
8	$\{2\}$	4	6	$\{1,3\}$	2	$\{3,5\}$	5
10	$\{3,5\}$	5	8	$\{2\}$	2	$\{2,3,4,5\}$	6

Exercise on Example 5.8

- Run the algorithm starting with step 1, in which you should define $T_2 = \{1\}$, and finishing with step 6, in which you should define $T_7 = \{1, 4\}$.
- Then check that steps 7 and 8 of the algorithm are exactly what we did in Example 5.6 and Exercise 5.7.
- At step 9 you should find that case (a) applies; check that step 10 finishes with the LFSR F_{11} of width $\ell_{11} = 6$ and taps $T_{11} = \{2, 3, 4, 5\}$, generating $u_0 u_1 \dots u_{10}$.

Berlekamp–Massey theorem

To prove that the LFSRs defined by running the Berlekamp–Massey algorithm have minimal possible width we need the following lemma. The proof is not obvious, but if you think 'what can I possibly do using Lemma 5.3' you should find the main idea.

Lemma 5.9

Let $n \geq \ell$. If an LFSR F of width ℓ generates the keystream $(u_0, u_1, \ldots, u_{n-1}, b)$ of length $n+1$ then any LFSR F' generating the keystream $(u_0, u_1, \ldots, u_{n-1}, \overline{b})$ has width ℓ' where $\ell' \geq n + 1 - \ell$.

Lemma 5.3

The word $u_0 u_1 \ldots u_{n-1}$ is the output of the LFSR with width ℓ and taps T if and only if $U_n(z) g_{T_n}(z) = h(z) + z^n r(z)$ for some polynomials $h(z)$ and $r(z)$ with $\deg h < \ell$.

Berlekamp–Massey theorem

To prove that the LFSRs defined by running the Berlekamp–Massey algorithm have minimal possible width we need the following lemma. The proof is not obvious, but if you think 'what can I possibly do using Lemma 5.3' you should find the main idea.

Lemma 5.9

Let $n \geq \ell$. If an LFSR F of width ℓ generates the keystream $(u_0, u_1, \ldots, u_{n-1}, b)$ of length $n+1$ then any LFSR F' generating the keystream $(u_0, u_1, \ldots, u_{n-1}, \overline{b})$ has width ℓ' where $\ell' \geq n + 1 - \ell$.

Recall that step n of the Berlekamp–Massey algorithm returns an LFSR F_{n+1} with taps T_{n+1} and width ℓ_{n+1} generating $u_0 \ldots u_{n-1} u_n$.

Theorem 5.10

With the notation above, $\max T_{n+1} \leq \ell_{n+1}$. *Moreover ℓ_{n+1} is the least width of any LFSR generating $u_0, \ldots, u_{n-1}, u_n$.*

Linear Complexity

The *linear complexity* of a word $u_0 u_1 \dots u_{n-1}$ is the minimal width of an LFSR that generates it. Modern stream ciphers aim to generate keystreams with high linear complexity. For example, take the m-quadratic stream cipher from Example 8.5. If $m = 1$ the keystream $u_0 u_1 \dots u_{29}$ for $k = 10101$ and $k' = 101010$ is

$$(1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1).$$

The table below shows the linear complexity of the first n bits of the keystream for small n and m.

$m\backslash n$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	2	2	2	2	5	5	5	5	5	5	5	5	5
2	0	2	2	2	2	2	5	5	5	5	5	5	5	5	5
3	0	0	0	4	4	4	4	4	4	6	6	6	6	6	6
4	0	0	0	0	0	7	7	7	7	7	7	7	7	7	8
5	0	0	0	0	5	5	5	5	5	5	5	7	7	7	8

For $n = 5$ the linear complexity is about $n/2$: this is the expected linear complexity of a random sequence of bits.

§6 Linear cryptanalysis

Example 6.1

Let $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ be the S-box in the Q-block cipher (see Example 9.5 in the main notes), defined by

$$S\big((x_0, x_1, x_2, x_3)\big) = (x_2, x_3, x_0 + x_1x_2, x_1 + x_2x_3).$$

(a) Suppose we look at position 0 of the output by considering $L_{\{0\}} \circ S : \mathbb{F}_2^4 \to \mathbb{F}_2$. We have

$$\begin{aligned}(L_{\{0\}} \circ S)\big((x_0, x_1, x_2, x_3)\big) &= L_{\{0\}}(x_2, x_3, x_0 + x_1x_2, x_1 + x_2x_3) \\ &= x_2 \\ &= L_{\{2\}}\big((x_0, x_1, x_2, x_3)\big).\end{aligned}$$

Hence $L_{\{0\}} \circ S = L_{\{2\}}$. By Lemma 4.3,

$$\operatorname{corr}(L_{\{0\}} \circ S, L_T) = \begin{cases} 1 & \text{if } T = \{2\} \\ 0 & \text{otherwise.} \end{cases}$$

Example 6.1 [continued]

(b) Instead if we look at position 2, the relevant Boolean function is $L_{\{2\}} \circ S$, for which $L_{\{2\}} \circ S\big((x_0, x_1, x_2, x_3)\big) = x_0 + x_1 x_2$.
Exercise: show that

$$\operatorname{corr}(L_{\{2\}} \circ S, L_T) = \begin{cases} \frac{1}{2} & \text{if } T = \{0\}, \{0,1\}, \{0,2\} \\ -\frac{1}{2} & \text{if } T = \{0,1,2\} \\ 0 & \text{otherwise} \end{cases} .$$

Example 6.2

For $k \in \mathbb{F}_2^{12}$ let $e_k : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ be the Q-block cipher, as defined in Example 8.4. Then $e_k\big((v, w)\big) = (v', w')$ where

$$v' = w + S\big(v + S(w + k^{(1)}) + k^{(2)}\big).$$

Recall that $k^{(1)} = (k_0, k_1, k_2, k_3)$ and $k^{(2)} = (k_4, k_5, k_6, k_7)$. Example 6.1 suggests considering $\mathrm{corr}(L_{\{0\}} \circ e_k, L_{\{2\}})$. We have

$$\begin{aligned}(L_{\{0\}} \circ e_k)\big((v, w)\big) &= L_{\{0\}}\big((v', w')\big) = v'_0 \\ L_{\{2\}}\big((v, w)\big) &= v_2.\end{aligned}$$

Exercise: using that $k_0^{(1)} = k_0$, $k_1^{(1)} = k_1$, $k_2^{(1)} = k_2$ and $k_2^{(2)} = k_6$, check that

$$v'_0 = v_2 + (w_1 + k_1)(w_2 + k_2) + k_0 + k_6.$$

Example 6.2 [continued]

By definition

$$\begin{aligned}\operatorname{corr}(L_{\{0\}} \circ e_k, L_{\{2\}}) &= \frac{1}{2^8} \sum_{(v,w)\in\mathbb{F}_2^8} (-1)^{v_2+(w_1+k_1)(w_2+k_2)+k_0+k_6}(-1)^{v_2} \\ &= \frac{1}{2^8}(-1)^{k_0+k_6} \sum_{(v,w)\in\mathbb{F}_2^8} (-1)^{(w_1+k_1)(w_2+k_2)} \\ &= (-1)^{k_0+k_6}\frac{1}{2^2} \sum_{w_1,w_2\in\mathbb{F}_2} (-1)^{(w_1+k_1)(w_2+k_2)}\end{aligned}$$

where the third line follows because the summand for (v, w) is the same for all 2^6 pairs with the same w_1 and w_2. In $\sum_{w_1,w_2\in\mathbb{F}_2}(-1)^{(w_1+k_1)(w_2+k_2)}$, the values of k_1 and k_2 are irrelevant.

Example 6.2 [continued]

By definition

$$\begin{aligned}
\operatorname{corr}(L_{\{0\}} \circ e_k, L_{\{2\}}) &= \frac{1}{2^8} \sum_{(v,w)\in\mathbb{F}_2^8} (-1)^{v_2+(w_1+k_1)(w_2+k_2)+k_0+k_6}(-1)^{v_2} \\
&= \frac{1}{2^8}(-1)^{k_0+k_6} \sum_{(v,w)\in\mathbb{F}_2^8} (-1)^{(w_1+k_1)(w_2+k_2)} \\
&= (-1)^{k_0+k_6}\frac{1}{2^2} \sum_{w_1,w_2\in\mathbb{F}_2} (-1)^{(w_1+k_1)(w_2+k_2)}
\end{aligned}$$

where the third line follows because the summand for (v, w) is the same for all 2^6 pairs with the same w_1 and w_2. In $\sum_{w_1,w_2\in\mathbb{F}_2}(-1)^{(w_1+k_1)(w_2+k_2)}$, the values of k_1 and k_2 are irrelevant. For instance, if both are 0 we average $(-1)^{w_1w_2}$ over all four $(w_1, w_2) \in \mathbb{F}_2^2$ to get $\frac{1}{2}$; if both are 1 we average $(-1)^{(w_1+1)(w_2+1)}$, seeing the same summands in a different order, and still getting $\frac{1}{2}$. Hence $\frac{1}{2^2}\sum_{w_1,w_2\in\mathbb{F}_2}(-1)^{(w_1+k_1)(w_2+k_2)} = \frac{1}{2}$ and

$$\operatorname{corr}(L_{\{0\}} \circ e_k, L_{\{2\}}) = \tfrac{1}{2}(-1)^{k_0+k_6}$$

Attack on the Q-block cipher

We can estimate this correlation from a collection of plaintext/ciphertext pairs $(v, w), (v', w')$ by computing $(-1)^{v'_0+v_2}$ for each pair. The mean should be close to $\frac{1}{2}(-1)^{k_0+k_6}$, and the sign then tells us $k_0 + k_6$. There are similar high correlations of $\frac{1}{2}$ for output bit 1. Using these one learns k_2 and k_3 as well as $k_1 + k_7$.

Exercise 6.3

Given $k_0 + k_6, k_1 + k_7, k_1, k_2, k_3$, how many possibilities are there for the key in the Q-block cipher?

Attack on the Q-block cipher

We can estimate this correlation from a collection of plaintext/ciphertext pairs $(v, w), (v', w')$ by computing $(-1)^{v'_0+v_2}$ for each pair. The mean should be close to $\frac{1}{2}(-1)^{k_0+k_6}$, and the sign then tells us $k_0 + k_6$. There are similar high correlations of $\frac{1}{2}$ for output bit 1. Using these one learns k_2 and k_3 as well as $k_1 + k_7$.

Exercise 6.3

Given $k_0 + k_6, k_1 + k_7, k_1, k_2, k_3$, how many possibilities are there for the key in the Q-block cipher?

The attack by differential cryptanalysis required chosen plaintexts. The attack by linear cryptanalysis works with any observed collection of plaintext/ciphertext pairs. It is therefore more widely applicable, as well as more powerful.

How to Find High Correlations

In the attack on the Q-Block Cipher we saw that the correlation depended on the key only by a sign. This is because key addition, as is almost universally the case for block ciphers, was done in $\mathbb{F}_2^n$.

Lemma 6.4

Fix $k \in \mathbb{F}_2^n$. Define $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ by $F(x) = x + k$. Then

$$\operatorname{corr}(L_S \circ F, L_T) = \begin{cases} (-1)^{L_S(k)} & \text{if } S = T \\ 0 & \text{if } S \neq T. \end{cases}$$

How to Find High Correlations

In the attack on the Q-Block Cipher we saw that the correlation depended on the key only by a sign. This is because key addition, as is almost universally the case for block ciphers, was done in $\mathbb{F}_2^n$.

Lemma 6.4

Fix $k \in \mathbb{F}_2^n$. Define $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ by $F(x) = x + k$. Then

$$\mathrm{corr}(L_S \circ F, L_T) = \begin{cases} (-1)^{L_S(k)} & \textit{if } S = T \\ 0 & \textit{if } S \neq T. \end{cases}$$

Another very useful result gives correlations through the composition of two functions.

Proposition 6.5

Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be functions. For $S, T \subseteq \{0, 1, \ldots, n-1\}$,

$$\mathrm{corr}(L_S \circ G \circ F, L_T) = \sum_{U \subseteq \{0,1,\ldots,n-1\}} \mathrm{corr}(L_S \circ G, L_U)\, \mathrm{corr}(L_U \circ F, L_T).$$

Example 6.6

(1) Take $G(x_0, x_1) = (x_0, x_0x_1)$. The matrix of correlations, with rows and columns labelled $\varnothing$, $\{0\}$, $\{1\}$, $\{0,1\}$ is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

(2) By Lemma 6.4, the matrix for $(x_0, x_1) \mapsto (x_0 + 1, x_1)$ is diagonal, with entries $1, -1, 1, 1$.

(3) Hence $H(x_0, x_1) = (x_0 + 1, x_0x_1 + x_1) = (\overline{x}_0, \overline{x}_0x_1)$ has correlation matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{pmatrix}.$$

Application of Proposition 6.5 to Q-block cipher

Let $F : \mathbb{F}_2^3 \to \mathbb{F}_2^3$ be the S-box in the 3 bit version of the Q-block cipher, so $F\big((x_0, x_1, x_2)\big) = (x_1, x_2, x_0 + x_1x_2)$. The matrix below shows the correlations,

$$\begin{pmatrix}
1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\
\cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & -\frac{1}{2} \\
\cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & -\frac{1}{2} & \cdot & \frac{1}{2} \\
\cdot & \frac{1}{2} & \cdot & -\frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} \\
\cdot & -\frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2}
\end{pmatrix}$$

using $\cdot$ for a 0 correlation, with subsets ordered

$$\varnothing, \{0\}, \{1\}, \{0,1\}, \{2\}, \{0,2\}, \{1,2\}, \{0,1,2\}.$$

For example the first four rows show that tapping in positions $\varnothing$, $\{0\}$, $\{1\}$, or $\{0,1\}$ gives a linear function.

Application of Proposition 6.5 to Q-block cipher

Let $F : \mathbb{F}_2^3 \to \mathbb{F}_2^3$ be the S-box in the 3 bit version of the Q-block cipher, so $F\big((x_0, x_1, x_2)\big) = (x_1, x_2, x_0 + x_1x_2)$. The matrix below shows the correlations,

$$\begin{pmatrix} 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & -\frac{1}{2} \\ \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & -\frac{1}{2} & \cdot & \frac{1}{2} \\ \cdot & \frac{1}{2} & \cdot & -\frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} \\ \cdot & -\frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} & \cdot & \frac{1}{2} \end{pmatrix}$$

using $\cdot$ for a 0 correlation, with subsets ordered

$$\varnothing, \{0\}, \{1\}, \{0,1\}, \{2\}, \{0,2\}, \{1,2\}, \{0,1,2\}.$$

By taking powers of this matrix we can compute correlations through any power of F. In the lecture we will use MATHEMATICA to find the order of the (normal) four bit version of F.

Problem Sheet 8, Question 5

(5) Let $\mathcal{P} = \mathcal{C} = \mathbb{F}_2^8$. Consider the cryptosystem with keys $(k, k') \in \mathbb{F}_2^8 \times \mathbb{F}_2^8$ and encryption functions defined by

$$e_{(k,k')}(x) = P(x + k) + k',$$

where P is the pseudo-inversion function from AES.

(a) Find $e_{(k,k')}^{-1}(z)$ for $z \in \mathbb{F}_2^8$.

(b) In a difference attack on this cryptosystem, the attacker takes $\boldsymbol{\Delta} = 1000\,0000$ corresponding to $1 \in \mathbb{F}_{2^8}$ and chooses $x \in \mathbb{F}_2^8$. She uses her black box to calculate $z = e_{(k,k')}(x)$ and $z_{\boldsymbol{\Delta}} = e_{(k,k')}(x_{\Delta})$, and finds $\boldsymbol{\Gamma} = z + z_{\boldsymbol{\Delta}}$. Suppose that $\boldsymbol{\Gamma} \neq 1000\,0000$. Show, using Lemma 10.8, that she can find $\{k, k + \boldsymbol{\Delta}\}$.

(c) Find all possible keys (k, k') in terms of $\boldsymbol{\Gamma}$.