Math 5251 Cyclic Redundancy Checks (Chap. 5)

(= fancier parity checks for error-detection, no correction)

Here the course takes an algebraic turn (like Math 4281, 5285-5286) treating $\Sigma = \{0,1\}$ as actual numbers, namely ...

§5.1 $\mathbb{F}_2 = GF(2) = \mathbb{Z}/2 = \mathbb{Z}/2\mathbb{Z}$ = integers mod 2

In the integers $\mathbb{Z}$, we know rules like

even + even = even   even . even = even
even + odd = odd   even . odd = even
odd + odd = even   odd . odd = odd

which we can codify in a system with 2 "numbers"

$\{ 0, 1 \}$
"evens" "odds"

+	0	1
0	0	1
1	1	0

×	0	1
0	0	0
1	0	1

It's called $\mathbb{F}_2$ = the field with 2 elements

$= GF(2)$ = Galois field with 2 elements

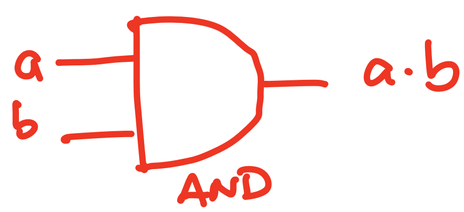
$= \mathbb{Z}/2 = \mathbb{Z}/2\mathbb{Z}$ = integers modulo 2

i.e., after $+, \times$ take remainders $\{0,1\}$ on division by 2

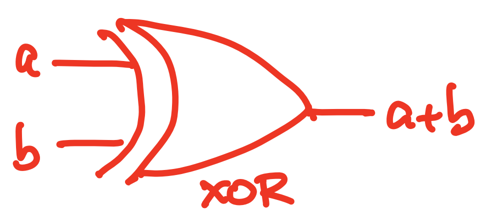
<u>Q</u>: How do we subtract in $\mathbb{F}_2$, e.g. who is -0? -1?
How do we divide a/b?

REMARK: In electrical engineering implementations interpreting $\{\underbrace{0}_{\text{FALSE}}, \underbrace{1}_{\text{TRUE}}\}$, they build/use logic gates:

$\times$ = AND



$+$ = XOR
"exclusive OR"



What we will really work with are ...

§ 5.2 $\mathbb{F}_2[x] :=$ polynomials in x with coefficients in $\mathbb{F}_2$

Remember $\left\{\begin{array}{c}\text{adding / subtracting} \\ \text{multiplying} \\ \text{dividing}\end{array}\right\}$ polynomials

with $\mathbb{R}$ coefficients ?

e.g.

$$
\begin{array}{r|llll}
 & 3x^2 & & -3 & \\
\hline
x^3+x+1 & 3x^5 & & -2x^2 & +7 \\
 & 3x^5 & +3x^3 & +3x^2 & \\
\hline
 & & -3x^3 & -5x^2 & +7 \\
 & & -3x^3 & & -3x-3 \\
\hline
 & & & -5x^2 & +3x+10
\end{array}
$$

stop here since degree is less than x^3+x+1

$$\Rightarrow \underbrace{3x^5-2x^2+7}_{f(x)} = \underbrace{(3x^2-3)}_{q(x)}\underbrace{(x^3+x+1)}_{g(x)} + \underbrace{(-5x^2+3x+10)}_{r(x)}$$

$$f(x) \quad = \quad q(x) \quad g(x) \quad + \quad r(x)$$

---

quotient

$$
\begin{array}{r|l}
 & q(x) \\
\hline
g(x) & f(x) \\
 & \vdots \\
\hline
 & r(x)
\end{array}
$$

remainder

$\Rightarrow$ One can write

$$f(x) = q(x) \cdot g(x) + r(x)$$

with $\deg(r) < \deg(g)$

uniquely, in fact.

(proof later)

We can similarly do this in

$\mathbb{F}_2[x] :=$ polynomials in x with $\mathbb{F}_2$ coefficients.

EXAMPLE

$$\begin{array}{r|l} & x^2 \qquad\quad +1 \\ \hline x^3+x+1 & x^5 \qquad\quad + x^2 + 1 \\ & x^5 + x^3 + x^2 \\ \hline & \qquad x^3 \qquad\quad + 1 \\ & \qquad x^3 \quad + x + 1 \\ \hline & \qquad\qquad x \end{array}$$

deg=3

deg=1; stop

$$\begin{array}{r|l} & q(x) \\ \hline g(x) & f(x) \\ & \vdots \\ \hline & r(x) \end{array}$$

$f(x) = q(x) \cdot g(x) + r(x)$
$\deg(r) < \deg(g)$

FASTER NOTATION:

	x^5	x^4	x^3	x^2	x	1
$x^5+x^2+1 \rightsquigarrow$	1	0	0	1	0	1
$x^3+x+1 \rightsquigarrow$			1	0	1	1

$$\begin{array}{r|l} & \quad 101 \rightsquigarrow x^2+1 \\ \hline 1011 & 100101 \\ & 1011 \\ \hline & \quad 1001 \\ & \quad 1011 \\ \hline & \qquad 10 \rightsquigarrow x \end{array}$$

We'll see later why $q(x), r(x)$ are unique if

$f(x) = q(x) \cdot g(x) + r(x)$ in $\mathbb{F}_2[x]$ (or $\mathbb{R}[x]$, $\mathbb{Q}[x]$, ...)
$\deg(r) < \deg(g)$

ACTIVE LEARNING:

$$f_1(x) = x^4 + x^2$$

$$f_2(x) = x^4 + x^2 + 1$$

(a) In $\mathbb{F}_2[x]$, divide $f_1(x), f_2(x)$ by

$$g_1(x) = x$$

$$g_2(x) = x+1$$

(b) How can one spot quickly whether

x divides $f(x)$ in $\mathbb{F}_2[x]$?

$x+1$ divides $f(x)$ in $\mathbb{F}_2[x]$?

(c) How does the answer to (b) relate to plugging in $x=0$, $x=1$, that is, evaluating $f(0)$, $f(1)$ in $\mathbb{F}_2$

§ 5.3 Cyclic redundancy checks (CRC's)

= an error-detection scheme where sender & receiver

1st pick a generator polynomial $g(x) \in \mathbb{F}_2[x]$.
(and we'll see some choices are better!)

2nd sender agrees to send messages as bit strings whose corresponding polynomial $d(x) \in \mathbb{F}_2[x]$ is always divisible by $g(x)$, by tacking on $\deg(g)$ extra bits at the end.

3rd the noisy channel transmits coefficients of some corrupted $\tilde{d}(x)$ instead of $d(x)$.

4th receiver computes the remainder $e(x)$ upon dividing $\tilde{d}(x)$ by $g(x)$;

reports $\begin{cases} \text{no error} & \text{if } e(x)=0, \\ \text{error} & \text{if } e(x)\neq 0. \end{cases}$

EXAMPLE We agree on $g(x) = x^3 + x + 1$ in $\mathbb{F}_2[x]$ $\leftrightarrow$ 1011

as generator polynomial.

I want to send you the information 10101,

so I must pick $\overset{7}{1}\overset{6}{0}\overset{5}{1}\overset{4}{0}\overset{3}{1}\,\underbrace{\overset{2}{a}\,\overset{1}{b}\,\overset{0}{c}}_{\text{3 extra bits, since } \deg(g)=3}$ to send

arranging that $f(x) = x^7 + x^5 + x^3 + ax^2 + bx + c$ is divisible by $g(x)$:

$$
\begin{array}{r|cccccccc}
 & & & & 1 & 0 & 0 & 1 & 1 \\
\hline
1011 & 1 & 0 & 1 & 0 & 1 & a & b & c \\
 & 1 & 0 & 1 & 1 & & & & \\
\hline
 & & & & 1 & 1 & a & b & c \\
 & & & & 1 & 0 & 1 & 1 & \\
\hline
 & & & & & 1 & a+1 & b+1 & c \\
 & & & & & 1 & 0 & 1 & 1 \\
\hline
 & & & & & & a+1 & b & c+1
\end{array}
$$

I want this to be 0, so pick $a=1, b=0, c=1$

and send $\boxed{10101101}$ $\leftrightarrow d(x) = x^7 + x^5 + x^3 + x^2 + 1$

If you receive $d(x)$ as $\tilde{d}(x) = 10101101$,

you compute

$$
\begin{array}{r}
10011 \\
1011 \,\big)\, \overline{10101101} \\
\vdots \\
\hline
000 = e(x)
\end{array}
$$

and are happy; no error.

If you receive $\tilde{d}(x)$ as $10\overset{0}{\cancel{1}}01101$, you compute

called a 1-bit error

$$
\begin{array}{r}
10100 \\
1011 \,\big)\, \overline{10001101} \\
1011 \\
\hline
1111 \\
1011 \\
\hline
1000 \\
1011 \\
\hline
111 = e(x) \neq 0
\end{array}
$$

ERROR - retransmit!

If you receive $\tilde{d}(x)$ as $10\overset{0}{\cancel{1}}011\overset{1}{\cancel{0}}1$, you compute

called a 2-bit or burst error, 4 bits apart

$$
\begin{array}{r}
10110 \\
1011 \,\big)\, \overline{10001111} \\
1011 \\
\hline
1111 \\
1011 \\
\hline
1001 \\
1011 \\
\hline
101 = e(x) \neq 0
\end{array}
$$

ERROR - retransmit!

ACTIVE LEARNING

(a) What happens if you receive $\tilde{d}(x)$ as $\overset{0}{\cancel{1}}010110\overset{0}{\cancel{1}}$?

(b) Can you explain why 1-bit errors are always detected by this CRC with $g(x) = x^3 + x + 1 \leftrightarrow 1011$?

We can analyze the errors undetected by the CRC $g(x)$ once we know a fact from Chap. 10: in $\mathbb{F}_2[x]$ and much more generally, one has uniqueness for the quotient, remainder $q(x), r(x)$ in this sense:

here
$$\begin{array}{r} q(x) \\ g(x)\,\overline{)\,f(x)} \\ \vdots \\ \overline{r(x)} \end{array}$$

$$\text{if } f(x) = q_1(x)\cdot g(x) + r_1(x) = q_2(x)\cdot g(x) + r_2(x)$$

with $\deg(r_i) < \deg(g)$ for $i = 1,2$

then $r_1(x) = r_2(x)$ and $q_1(x) = q_2(x)$.

In particular, $g(x)$ divides $f(x) \iff r(x) = 0$

NOTATION: $g(x) \mid f(x)$

COROLLARY: If $d(x)$ is sent, but $\tilde{d}(x) \neq d(x)$ received, the CRC with generator $g(x)$ misses the error

$$\iff g(x) \mid \tilde{d}(x) - d(x) \text{ in } \mathbb{F}_2[x]$$

proof: Write $d(x) = q(x) \cdot g(x)$ where $q(x) \in \mathbb{F}_2[x]$; possible since $d(x)$ was sent that way by CRC rules.

Then $g(x)$ misses the error

$$\iff \text{remainder } e(x) = 0 \text{ in } \begin{array}{r} \tilde{q}(x) \\ g(x) \overline{\smash{\big)}\ \tilde{d}(x)} \\ \vdots \\ \hline e(x) = 0 \end{array}$$

uniqueness of remainder

$$\iff \tilde{d}(x) = \tilde{q}(x) \cdot g(x) \quad \text{for some } \tilde{q}(x) \in \mathbb{F}_2[x]$$

$$\iff \tilde{d}(x) - d(x) = \tilde{q}(x)\, g(x) - q(x)\, g(x)$$
$$= \left(\tilde{q}(x) - q(x)\right) g(x)$$
for some $\tilde{q}(x)$

$$\iff g(x) \mid \tilde{d}(x) - d(x).\ \blacksquare$$

COROLLARY Assume $g(x) \in \mathbb{F}_2[x]$ has $\deg(g) \geq 1$ and nonzero constant term, that is

$$g(x) = 1 + a_1 x + a_2 x^2 + \dots + a_{r-1} x^{r-1} + x^r \quad \text{with } r \geq 1.$$

Then when used to generate a CRC,

(a) $g(x)$ never misses 1-bit errors,

(b) $g(x)$ also catches every 2-bit error until they are at least N_0 bits apart where $N_0 :=$ smallest N for which $g(x) \mid x^N + 1$.

EXAMPLES

(1) $g(x) = x^3 + x + 1$ catches all 1-bit errors and all 2-bit errors up to 6 bits apart,

since $x^3 + x + 1 \nmid \left.\begin{array}{l} x+1, \\ x^2+1, \\ x^3+1, \\ x^4+1, \\ x^5+1, \\ x^6+1 \end{array}\right\}$ easy to check

but $x^3 + x + 1 \mid x^7 + 1$; $N_0 = 7$.

(2) We can later easily produce small $g(x)$ doing much better,

e.g. $x^{15} + x + 1$ has $N_0 = 2^{15} - 1 = 32767$

(3) Note that when we use a CRC with generator $g(x) = x+1$, this is the same as our old parity check bit scheme:

$$b_1 b_2 \cdots b_\ell \longmapsto b_1 b_2 \cdots b_\ell b_{\ell+1}$$

where $b_{\ell+1} = b_1 + b_2 + \cdots + b_\ell$ in $\mathbb{F}_2$

$$= \begin{cases} 0 & \text{if } \sum_{i=1}^{\ell} b_i \text{ even} \\ 1 & \text{if } \sum_{i=1}^{\ell} b_i \text{ odd} \end{cases}$$

Since $g(x) = x+1$ has nonzero constant term and $\deg(g) = 1 \geq 1$,

it detects all 1-bit errors.

But it has $N_0 = 1$, and misses all 2-bit errors, since

$$x+1 \mid x^N + 1 = (x+1)(x^{N-1} + x^{N-2} + \cdots + x^2 + x + 1)$$

in $\mathbb{F}_2[x]$ $\quad \forall N \geq 1.$

proof: A 1-bit error means $\tilde{d}(x)-d(x)=x^n$ for some n, and we claim $g(x)$ can't divide x^n:

given $h(x)\in\mathbb{F}_2[x]$ with highest power x^M and smallest power x^m

so $h(x)=x^m+a_{m+1}x^{m+1}+\dots+a_{M-1}x^{M-1}+x^M$,

one finds $g(x)h(x)=$

$$(1+a_1x+\dots+a_{r-1}x^{r-1}+x^r)(x^m+a_{m+1}x^{m+1}+\dots+a_{M-1}x^{M-1}+x^M)=$$

$$x^m+\left(\begin{array}{c}\text{terms involving}\\ x^{m+1},x^{m+2},\dots,x^{M+r-1}\end{array}\right)+x^{M+r}$$

which can't equal $x^n = 0+0\cdot x^1+0\cdot x^2+\dots+0\cdot x^{n-1}+x^n$.

A 2-bit error N bits apart means $\tilde{d}(x)-d(x)=x^n+x^{n+N}$
$=x^n(x^N+1)$

for some n, and we claim

$$g(x)\mid x^n(x^N+1)\Rightarrow g(x)\mid x^N+1:$$

If $x^n+x^{n+N}=g(x)h(x)$ with some h written as above,

$$=x^m+\left(\begin{array}{c}\text{terms involving}\\ x^{m+1},x^{m+2},\dots,x^{M+r-1}\end{array}\right)+x^{M+r}$$

then this forces $m=n$, so one can cancel x^n from both $h(x)$ and x^n+x^{n+N}, giving

$$1+x^N=g(x)\hat{h}(x), \text{ i.e. } g(x)\mid x^N+1. \quad \blacksquare$$