Binary Arithmetic: counting with ones

Robin McLean

28 February 2015

Weighing in ounces

Suppose we have one of each of the following weights:

32oz, 16oz, 8oz, 4oz, 2oz and 1oz.

Can we weigh any whole number of ounces?

e.g. 43oz = 32oz + 8oz + 2oz + 1oz.

Are there any other ways of making 43oz with our given weights?

What about other weights from 0oz to 63oz?

Arithmetic base 2

For ordinary arithmetic we use base 10 with columns (100s, 10s and units) that are powers of 10 and digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. We could use base 2, working with columns that are powers of 2. Corresponding to digits, we use bits 0 and 1. For example:

$$\begin{array}{cccccccc} & & 32 & 16 & 8 & 4 & 2 & 1 \\ \cline{3-8} 43 & = & 1 & 0 & 1 & 0 & 1 & 1 \end{array}$$

Roman numerals

M	1,000
D	500
C	100
L	50
X	10
V	5
I	1

Can you interpret these Roman numerals?

MDCCCCLXXXXVIIII

MCMXCIX

MIM

Roman numerals are posh, but useless. The representation of a number may not be unique and the system is not based on powers.

1	1	17	10001
2	10	18	10010
3	11	19	10011
4	100	20	10100
5	101	21	10101
6	110	22	10110
7	111	23	10111
8	1000	24	11000
9	1001	25	11001
10	1010	26	11010
11	1011	27	11011
12	1100	28	11100
13	1101	29	11101
14	1110	30	11110
15	1111	31	11111
16	10000	32	100000

Digits and successive remainders in base ten

Suppose we divide 4,357 repeatedly by 10:

$10)\underline{4357}$
$10)\underline{\ \ 435}$ rem 7
$10)\underline{\ \ \ \ 43}$ rem 5
$10)\underline{\ \ \ \ \ \ 4}$ rem 3
$10)\underline{\ \ \ \ \ \ 0}$ rem 4

successive remainders (read upwards) give the digits of our original number. This is because

$$\begin{aligned} 4357 &= 435x10 + 7 = \{[43x10] + 5\}x10 + 7 \\ &= \{[(4x10 + 3)x10] + 5\}x10 + 7 \\ &= 4x10^3 + 3x10^2 + 5x10 + 7. \end{aligned}$$

Converting from decimal to binary

When we divide a number repeatedly by 2, successive remainders (read upwards) give its digits (now bits) in base 2.

For example, to convert 75_{10} to base 2:

$2)\underline{75}$

$2)\underline{37}$ rem 1

$2)\underline{18}$ rem 1

$2)\underline{\ 9}$ rem 0

$2)\underline{\ 4}$ rem 1

$2)\underline{\ 2}$ rem 0

$2)\underline{\ 1}$ rem 0

$2)\underline{\ 0}$ rem 1

so $75_{10} = 1001011_2$.

A friend claims that you don't need to know all your tables, provided that you can multiply and divide by 2. Moreover you needn't bother with the remainders! This sounds suspicious, but your friend shows you an example:

To work out 13 x 27, put the numbers in two columns. In the first column, produce more numbers by dividing by 2 (ignoring remainders), Stop when you reach 1. In the second column, produce more numbers by doubling. Finally, add up those numbers in the second column (marked with an asterisk) that are opposite odd numbers in the first column.

$$\begin{array}{rcl} 13 & & 27 \ast \\ 6 & & 54 \\ 3 & & 108 \ast \\ 1 & & \underline{216} \ast \\ 13x27 & = & 351 \end{array}$$

Puzzle

Does your friend's method always work? Try it with some other numbers.

If you think that it won't always work, try to produce a counterexample (an example in which it doesn't work).

If you think that it does always work, try to show why .

Binary arithmetic

Addition is easy - think $1 + 1$ is 0 down and 1 to carry.

Carrying in **subtraction** is slightly trickier, but subtraction can always be done by complementary addition (described last month by Elliott Tjia).

Multiplication is delightful - no hard tables. No OFSTED inspectors said that children didn't know their 1-times table!

Division is also very easy. At each stage in a long division sum, the number you are dividing by either doesn't go (so enter 0 in the answer and bring down the next bit) or does go (necessarily only once).

Bicimals

In base ten, fractions are often handled as decimals, with columns representing $\frac{1}{10}$ths, $\frac{1}{100}$ths, $\frac{1}{1000}$ths,… to the right of the decimal point. In binary arithmetic we have columns for $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots$ to the right of a binary point. To express $\frac{1}{3}$ as a bicimal, we could divide 1 by 11:

$$\begin{array}{l}
\qquad \underline{0.010101\ldots} \\
11)1.000000\ldots \\
\qquad\ \ \underline{11} \\
\qquad\quad 100 \\
\qquad\qquad \underline{11}\ldots
\end{array}$$

An alternative method is shown on the next slide.

$$\begin{aligned}
\frac{1}{3} &= \frac{1}{4} + \frac{1}{12} = \frac{1}{4} + \frac{1}{4}x\frac{1}{3} \\
&= \frac{1}{4} + \frac{1}{4}\left(\frac{1}{4} + \frac{1}{4}x\frac{1}{3}\right) = \frac{1}{4} + \frac{1}{16} + \frac{1}{16}x\frac{1}{3} \\
&= \frac{1}{4} + \frac{1}{16} + \frac{1}{16}\left(\frac{1}{4} + \frac{1}{4}x\frac{1}{3}\right) = \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{64}x\frac{1}{3} \\
&= \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256} + \ldots = 0.01010101\ldots_2
\end{aligned}$$

An error-correcting code

Suppose that we wish to transmit messages (in the form of 0s and 1s) with less than 12 bits, and that our message may contain a single error. Let us use the 1st, 2nd, 4th and 8th bits of the message as checks, leaving the 3rd, 5th, 6th, 7th, 9th, ...15th bits for the message itself. Our plan of campaign is as follows.

(1) Calculate the check bits x_1, x_2, x_4 and x_8.

(2) Transmit the entire message including check bits. Call this $x_1, x_2, x_3, \ldots x_{15}$.

(3) Call the entire received message $y_1, y_2, y_3, \ldots, y_{15}$. Interrogate different subsets of this message and use the idea of our number cards to find where the error lies.

Recall that, for our length of message, each number card would consist of all the numbers from one of the rows below:

1	3	5	7	9	11	13	15
2	3	6	7	10	11	14	15
4	5	6	7	12	13	14	15
8	9	10	11	12	13	14	15

Choose the check bits x_1, x_2, x_4 and x_8 so that each of the following sums is even:

$$\begin{aligned}
&x_1 + x_3 + x_5 + x_7 + x_9 + x_{11} + x_{13} + x_{15}\\
&x_2 + x_3 + x_6 + x_7 + x_{10} + x_{11} + x_{14} + x_{15}\\
&x_4 + x_5 + x_6 + x_7 + x_{12} + x_{13} + x_{14} + x_{15}\\
&x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15}.
\end{aligned}$$

Example

Suppose that we wish to send the message M below. We calculate the check bits (in bold) and transmit the complete string T. Suppose that the string of bits R is received. The entries in this row are $y_1, y_2, \ldots y_{15}$. Note that one error (underlined) has occurred.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M			0		1	1	0		0	1	0	1	1	1	0
T	**0**	**1**	0	**1**	1	1	0	**0**	0	1	0	1	1	1	0
R	**0**	**1**	0	**1**	1	1	$\underline{1}$	**0**	0	1	0	1	1	1	0

$y_1 + y_3 + y_5 + y_7 + y_9 + y_{11} + y_{13} + y_{15}$ is odd,
$y_2 + y_3 + y_6 + y_7 + y_{10} + y_{11} + y_{14} + y_{15}$ is odd,
$y_4 + y_5 + y_6 + y_7 + y_{12} + y_{13} + y_{14} + y_{15}$ is odd, and
$y_8 + y_9 + y_{10} + y_{11} + y_{12} + y_{13} + y_{14} + y_{15}$ is even, so the recived message, R, has an error in position $1 + 2 + 4 = 7$. This can now be corrected.

Bases within bases

Given the binary representation of a number, it is easy to find its representation in any of the bases 4, 8, 16, . . . simply by taking the binary columns 2 (or 3 or 4 . . .) at a time, For example

$$\begin{array}{cccc|cc|cccccc} & & 32 & 16 & 8 & 4 & 2 & 1 & & 16 & 4 & 1 \\ \hline 45_{10} & = & 1 & 0 & 1 & 1 & 0 & 1 & = & 2 & 3 & 1 \end{array}$$

and

$$\begin{array}{ccccc|ccc|ccc} & & 32 & 16 & 8 & 4 & 2 & 1 & & 8 & 1 \\ \hline 45_{10} & = & 1 & 0 & 1 & 1 & 0 & 1 & = & 5 & 5 \end{array}$$

In the same way, we can think of arithmetic base 100 or 1,000 . . . embedded in base 10. For example 5,371 can be considered as 53 in the 100s column and 71 in the units column. This allows you to work to more decimal places than your calculator can display.

What is 12345678 x 87654321?

10^{12}	10^8	10^4	1	
		1234	5678	
		8765	4321	x
		2453	4638	
	533	2114		
	4976	7670		
1081	6010			
1082	1520	2237	4638	

My calculator gives $\sqrt{2}$ as 1.4142136 so, if

$$\sqrt{2} = 1.4142136 + x,$$

then x is numerically less than $5x10^{-8}$ and $x^2 < 2.5x10^{-15}$. Squaring gives

$$2 = 1.4142136^2 + 2.8284272x + x^2.$$

Using my calculator to multiply in base 10^4 shows that

$$1.4142136^2 = 2.00000010642496.$$

So $x \approx -\frac{1.0642496x10^{-7}}{2.8284272} \approx -3.762690x10^{-8}$ to 14 d.p.

$$\begin{aligned}\text{Thus} \quad \sqrt{2} &= 1.41421360000000 - 0.00000003762690 \\ &= 1.41421356237310 \quad \text{to 14 decimal places.}\end{aligned}$$