MTH303

Section 1.2: Binary Numbers

R.Touma

These lecture slides are not enough to understand the topics of the course; they could be used along with the textbook

Sequences and Series

Definition:

An infinite geometric series is a series of the form

$$\sum_{n=1}^{\infty} cr^{n-1} = c + cr + cr^2 + ... + cr^n + ...$$

where $c \neq 0$ is the first term and $r \neq 0$ is the ratio.

Theorem (Geometric Series)

The geometric series has the following properties:

$$\begin{cases} \text{If } |r| < 1, & \text{then } \sum_{n=1}^{\infty} cr^{n-1} = \frac{c}{1-r} \\ \text{If } |r| > 1, & \text{then } \sum_{n=1}^{\infty} cr^{n-1} \text{diverges.} \end{cases}$$

Proof

The sum of the first n terms of a geometric series is:

$$\begin{aligned} S_n &= c + cr^1 + cr^2 + cr^3 + ... + cr^{n-1} \\ &= c\frac{1-r^n}{1-r} \end{aligned}$$

if $|r| < 1, \lim_{n\to\infty} r^n = 0$ and $S = \lim_{n\to\infty} S_n = \frac{c}{1-r}$.
if $|r| > 1, \lim_{n\to\infty} r^n = \infty$ and the series diverges.
Remark:
When rational numbers are expressed in decimal form, it is often the case that infinitely many digits are required.
Example $\frac{1}{3} = 0.3333 = 0.\overline{3}$. Here $\overline{3}$ means that 3 is repeated for ever.

$$\begin{aligned} 0.\overline{3} &= 0.3 + 0.03 + 0.003 + ... \\ &= 3(\frac{1}{10})^1 + 3(\frac{1}{10})^2 + 3(\frac{1}{10})^3 + ... \end{aligned}$$

is the sum of a converging geometric series with first term $\dfrac{3}{10}$ and ratio $r = \dfrac{1}{10}$.

The geometric series converges to $\dfrac{3/10}{1-1/10} = \dfrac{1}{3}$

Binary Fractions

Recall

The binary number $101|_2$ is equal to 5 in base 10.

In fact $101|_2 = 1 + 0 \times 2^1 + 1 \times 2^2$. In general the binary number $a_n a_{n-1} a_{n-2} \cdots a_1 a_0|_2$ is equivalent to the number

$$a_0 + a_1 \times 2 + a_2 \times 2^2 + a_3 \times 2^3 + \cdots a_n \times 2^n$$

in base 10.

Binary fractions can be expressed as sums involving negative powers of 2. If R is a real number, $0 < R < 1$, then there exist digits $d_1, d_2, ...$ so that:

$$R = d_1 \times 2^{-1} + d_2 \times 2^{-2} + d_3 \times 2^{-3} + ... + d_n \times 2^{-n} + ...,$$

with $d_j \in \{0, 1\}$.

R is expressed in the binary fraction notation:

$$0.d_1 d_2 d_3 ... d_n ...\text{two}$$

Example:

The fraction $\frac{7}{10}$ can be expressed as 0.7 in base 10, while its binary representation requires infinitely many digits $\frac{7}{10} = 0.1\overline{0110}_{\text{two}}$.

Algorithm for finding the base 2 representation

Let R denote a real number that lies between 0 and 1 ($0 < R < 1$). We want to determine the digits $d_1, d_2, ...d_n, ...$ such that

$$R = d_1 \times 2^{-1} + d_2 \times 2^{-2} + d_3 \times 2^{-3} + ... + d_n \times 2^{-n} + ...$$

We multiply both sides of the last equation by 2, we obtain:

$$2R = d_1 + (d_2 \times 2^{-1} + d_3 \times 2^{-2} + ... + d_n \times 2^{-n+1} + ...)$$

The quantity in parentheses on the right side is positive and is less than 1.
Therefore d_1 is the integer part of $2R$, denoted $d_1 = int(2R)$. Let F_1 denote the fractional part of $2R$.
$F_1 = frac(2R) = d_2 \times 2^{-1} + d_3 \times 2^{-2} + ... + d_n \times 2^{-n+1} + ...$.
Again we multiply by 2 both sides of the equality; we obtain:

$$2F_1 = d_2 + d_3 \times 2^{-1} + ... + d_n \times 2^{-n+2} + ...$$

We set $d_2 = int(2F_1)$ and $F_2 = frac(2F1)$. The process is continued ad

infinitum, and two sequences $\{d_k\}$ and $\{F_k\}$ are generated ($d_k = int(2F_{k-1})$ and $F_k = frac(2F_{k-1})$ where $d_1 = int(2R)$ and $F_1 = frac(2R)$).

The binary decimal representation of R is then given by the convergent series $R = \sum_{d=1}^{\infty} d_j 2^{-j}$.

Example:

The binary decimal representation of $\frac{7}{10}$ can be found as follows:

Let $R = \frac{7}{10} = 0.7$

$$\begin{aligned} 2R = 1.4, &\quad d_1 = int(1.4) = 1, &\quad F_1 = frac(1.4) = 0.4 \\ 2F_1 = 0.8, &\quad d_2 = int(0.8) = 0, &\quad F_2 = frac(0.8) = 0.8 \\ 2F_2 = 1.6, &\quad d_3 = int(1.6) = 1, &\quad F_3 = frac(1.6) = 0.6 \\ 2F_3 = 1.2, &\quad d_4 = int(1.2) = 1, &\quad F_4 = frac(1.2) = 0.2 \end{aligned}$$

$$2F_4 = 0.4, \quad d_5 = int(0.4) = 0, \quad F_5 = frac(0.4) = 0.4$$
$$2F_5 = 0.8, \quad d_6 = int(0.8) = 0, \quad F_6 = frac(0.8) = 0.8$$
$$2F_6 = 1.6, \quad d_7 = int(1.6) = 1, \quad F_7 = frac(1.6) = 0.6$$

Note that $2F_2 = 1.6 = 2F_6$ so that $d_k = d_{k+4}$ and $F_k = F_{k+4}$ will occur for $k = 2, 3, 4, ...$
Thus $\frac{7}{10} = 0.7 = 0.1\overline{0110}_{\text{two}}$.

Example:
Find the base 10 rational number that the binary number $0.\overline{01}_{\text{two}}$ represents.

$$\begin{aligned} 0.\overline{01}_{\text{two}} &= 0 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} + 0 \times 2^{-5} + 1 \times 2^{-6} + ... \\ &= \sum_{k=1}^{\infty} (2^{-2})^k = \frac{a}{1-r} \\ &= \frac{2^{-2}}{1-2^{-2}} = \frac{\frac{1}{4}}{1-\frac{1}{4}} = \frac{1}{3} \end{aligned}$$

Binary shifting

If a rational number that is equivalent to an infinite repeating binary expansion is to be found, then a shift in the digits can be helpful. Let

$$S = 0.00000\overline{11000}_{\text{two}}.$$

Multiplying both sides of the last equation by $2^5 = 32$ will shift the binary point five places to the right; we obtain:

$$32S = 0.\overline{11000}_{\text{two}}.$$

Similarly, multiplying S by $2^{10} = 1024$ will shift the binary point 10 places to the right

$$1024S = 11000.\overline{11000}_{\text{two}}.$$

If we compute the difference $1024S - 32S$, we obtain:

$$1024S - 32S \quad = \quad 11000_{\text{two}}$$

Thus

$$\begin{aligned} 992S &= 24, \quad (\text{since} 11000_{\text{two}} = 24) \\ S &= 8/33 \end{aligned}$$

Scientific notation

A standard way to represent a real number, called scientific notation is obtained by shifting the decimal point and supplying the appropriate power of 10. For example:

$$\begin{aligned} 0.0000747 &= 7.47 \times 10^{-5} \\ 31.4159265 &= 3.14159265 \times 10^{1} \\ 9,700,000,000 &= 9.7 \times 10^{9} \end{aligned}$$

Machine numbers

Computers use normalized floating-point binary representation for real numbers. This means that the mathematical quantity x, is not actually stored in the

computer, but instead the computer stores a binary approximation to x:

$$x \approx \pm q \times 2^n.$$

q is the $mantissa$ and it is a finite binary expression that satisfies $1/2 \leq q < 1$. n is an integer, it is called the exponent. Due to physical restrictions such as the number of bits (for storage), only a small subset of the real number is used in a computer. The number of binary digits is restricted in both the numbers q and n. For example, consider the numbers of the form:

$$0.d_1d_2d_3d_4{}_{\text{two}} \times 2^n$$

where $d_1 = 1$, and d_2, d_3, d_4 are either 0 or 1 and $n \in \{-3, -2, -1, 0, 1, 2, 3, 4\}$. There are 8 choices for the mantissa and 8 choices for the exponent; this produces a set of 64 numbers:

$$\{0.1000_{\text{two}} \times 2^{-3}, 0.1001_{\text{two}} \times 2^{-3}, ..., 0.1110_{\text{two}} \times 2^4, 0.1111_{\text{two}} \times 2^4\}$$

Notice here that when the mantissa and the exponent are restricted, the computer

has a limited number of values it chooses to approximate real numbers x.

Example:

What would happen if a computer had a 4-bit mantissa and was restricted to perform the computation $(\frac{1}{10} + \frac{1}{5}) + \frac{1}{6}$?

Assume that the computer rounds all real numbers to the closest number in table 1.13

$$\begin{aligned} \frac{1}{10} &\approx 0.1101_{\text{two}} \times 2^{-3} &&= 0.01101_{\text{two}} \times 2^{-2} \\ \frac{1}{5} &\approx 0.1101_{\text{two}} \times 2^{-2} &&= 0.1101_{\text{two}} \times 2^{-2} \\ \frac{3}{10} & && 1.00111_{\text{two}} \times 2^{-2} \end{aligned}$$

The computer rounds the number $1.00111_{\text{two}} \times 2^{-2}$; it stores the number

$0.1010_{\text{two}} \times 2^{-1}$. The next step is:

$$\begin{aligned} \frac{3}{10} &\approx 0.1010_{\text{two}} \times 2^{-1} &&= 0.1010_{\text{two}} \times 2^{-1} \\ \frac{1}{6} &\approx 0.1011_{\text{two}} \times 2^{-2} &&= 0.01011_{\text{two}} \times 2^{-1} \\ \frac{7}{15} & && 0.11111_{\text{two}} \times 2^{-1} \end{aligned}$$

The computer rounds the number $0.11111_{\text{two}} \times 2^{-1}$; it stores the number $0.1000_{\text{two}} \times 2^{0}$. The computer's solution to the addition problem is:

$$\frac{7}{15} \approx 0.1000_{\text{two}} \times 2^{0}.$$

The error the computer makes is:

$$\frac{7}{15} - 0.1000_{\text{two}} \times 2^{0} \approx 0.466667 - 0.5 \approx 0.033333$$

Computer accuracy, floating points numbers

To store numbers accurately, computer must use at least 24-bits binary mantissa which translates to seven decimal places. When a 32 bits mantissa is used, numbers with 9 decimal places can be placed.

Computers use two modes for representing numbers: an integer mode and a floating point mode. The first mode is used when performing computations that are known to be integer valued. The second mode is used for scientific and engineering applications.

Computer that use 32 bits to represent single precision real number use 8 bits for the exponent and 24 bits for the mantissa. They can represent real numbers with magnitude in the range of 2.938736×10^{-39} and 1.701412×10^{38}.

If the computer has 64 bits double precision real numbers, it might use 11 bits for the exponent and 53 bits for the mantissa. They can represent real numbers with magnitude in the range of $5.562684646268003 \times 10^{-309}$ and $8.988465674311580 \times 10^{307}$. with about 16 decimal digits of numerical precision.

Assignment

Prepare the following problems:

Page 23, ♯ 2-a, 2-c, 3-a, 3-c, 4-a, 6-a, 6-b, 7-a, 9-a, 13-a, 13-b.