COMP1521 Week 7

Welcome!!!
Please sit down we shall start at 11:05am

Overview

- Two’s complement
- Floating-point numbers in binary

Two’s Complement

- How we represent negative numbers in Binary
- Negative numbers are determined by whether or not the most significant bit is set to 1

Example:

-13 = 0b1101

Twos complement formula

Convert from decimal to binary

1. Write out the binary of the positive number
2. Flip all the bits
3. Add 1

Convert from binary to decimal

1. Minus 1
2. Flip all the bits
3. Convert to decimal as if it was positive
4. Add a minus sign in front of your result

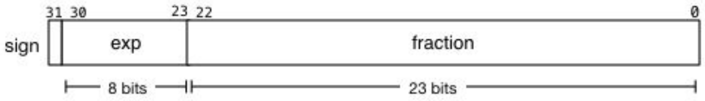
Why: https://youtu.be/4qH4unVtJkE

n	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2^n	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
bit																

IEEE-754 standard for floating-point numbers in binary (binary to decimal)

Floating-point Representation

Single-precision floating-point numbers that follow the IEEE 754 standard have an internal structure that looks like this:



The value is determined as:

$$-1^{\mathrm{sign}} \times (1 + \mathrm{frac}) \times 2^{\mathrm{exp}-127}$$

The 32 bits in each `float` are used as follows:

`sign` is a single bit that indicates the number's sign. If set to 0, the number is positive; if set to 1, the number is negative.

`exp` is an unsigned 8-bit value (giving a range of $[0 \cdots 255]$) which is interpreted as a value in the range $[-127 \cdots 128]$ by subtracting 127 (the "bias") from the stored 8-bit value. It gives a multiplier for the fraction part (i.e., $2^{\mathrm{exp}-127}$).

`frac` is a value in the range $[0 \cdots 1]$, determined using positional notation:

$$\frac{\mathrm{bit}_{22}}{2^1} + \frac{\mathrm{bit}_{21}}{2^2} + \frac{\mathrm{bit}_{20}}{2^3} + \cdots + \frac{\mathrm{bit}_{2}}{2^{21}} + \frac{\mathrm{bit}_{1}}{2^{22}} + \frac{\mathrm{bit}_{0}}{2^{23}}$$

The overall value of the floating-point value is determined by adding 1 to the fraction: we assume that the "fraction" part is actually a value in the range $[1 \cdots 2]$, but save bits by not explicitly storing the leading 1 bit.

8 bit Floating Point Example

- sign bit → 1 bit
- exponent bits → 3 bits (bias = 3)
- fraction bits → 4 bits

Example:

Convert 01000110

IEEE-754 standard for floating-point numbers in binary (decimal to binary)

1. Express the decimal using this equation
   a. num = (1+frac)x2^n
   b. (1 + frac) = k/(largest 2^n that is smaller than k)
   c. 0.375

Special Floating Point Numbers

- 0 → exponent bits are 0
- Infinity → exponent bits are all 1’s fraction is 0
- NaN → exponent bits are all 1’s fraction is not 0

Code from tut

shorturl.at/kuFHW

Feedback form

Code: Dinosaur