Exercise Session

Week 05

Adel Gavranović
agavranovic@student.ethz.ch

Today's Overview

polybox for session material   Mail to TA

Special

- From now on: 90min sessions, no break, earlier lunch
- Please fill out the evaluation survey

Today's Topics

Comments on last [code]expert Exercises

- I'll usually be done with correcting on Sunday morning. Please make sure to have a look at my feedback before the next exercise session
- Give you variables descriptive names
- Use more comments
- Use tabs to indent code
- `unsigned int` representation $\neq$ `int` representation (check Handout02.pdf on the polybox)

Question or Comments re: Exercises?

Learning Objectives Checklist

Now I...

- ☐ have refreshed my memory on how ”two’s complement representation” works
- ☐ know how floating point numbers are stored
- ☐ can convert a non-integer number into its binary representation
- ☐ can compute the elements of the set $F(\beta, p, e_{min}, e_{max})$
- ☐ can perform arithmetic operations in the set $F(\beta, p, e_{min}, e_{max})$
- ☐ know the floating point guidelines and the reasoning behind them

Recap: Binary Representation ...but which one?

Math. Decimal	Math. Binary	`int`	`unsigned int`
42_{10}	101010_2	`101010`	`101010`
-42_{10}	-101010_2	`1010110`	nope

`int` stores numbers with *two's complement representation*.

(Technically, the rest of the binary-representation-numbers would be filled up (to the left) with either `1`'s or `0`'s depending on the type/representation because an `int` stores numbers in a fixed amount of space, but we're not covering this, so don't worry about that.)

Binary Arithmetic

Task

Perform the following steps:

1. Convert the integer numbers `a = 4` and `b = 7` into their binary representation (not two's complement)
2. Add them in their binary representation
3. Convert the result into decimal

Solution

$a = 4_{10} = 100_2$
$b = 7_{10} = 111_2$
$100_2 + 111_2 = 1011_2$
$1011_2 = 11_{10}$

Expressions

Task

Evaluate the following expressions:

1. `5 < 4 < 1`
2. `true > false`

Solution 1

```
5 < 4 < 1
(5 < 4) < 1
false < 1
0 < 1
true
```

Solution 2

```
true > false
1 > 0
true
```

Questions?

Binary Representation

binary	1	1	1	1	.	1	1	1
decimal	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}	2^{-3}
	8	4	2	1	.	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$

Tasks

1. 11.01_2 in decimal
2. 101.1_2 in decimal
3. 7.125_{10} in binary
4. 4.375_{10} in binary
5. 1.1 in binary

Solutions

1. $2 + 1 + 0 + \frac{1}{4} = 3.25_{10}$
2. $4 + 0 + 1 + \frac{1}{2} = 5.5_{10}$
3. 111.001_2
4. 100.011_2
5. $1.1_{10} = 1.000110..._2$

Binary Representation

My way of transforming decimal into binary

1. calculate the numbers before the decimal point
2. write that number_2 down. now look at the number_{10}.rest
3. can you substract $\frac{1}{2^n}$ from number_{10}?
   if possible, subtract $\frac{1}{2^n}$ from the number_{10} and add a 1 at the end of your number_2
   else add a 0 at the end of your number_2
4. if $\text{number}_{10} = 0$, stop and check your solution
   else, `n++;` and go to 3. again
5. check solution again and remember $2^{-2} \neq \frac{1}{2}$ (common mistake)

Normalized Floating Point Systems

$F^*(\beta, p, e_{min}, e_{max})$

$*$	normalized ($b_0 \neq 0$)
$\beta \geq 2$	base
$p \geq 1$	precision (number of places)
e_{min}	smallest possible exponent
e_{max}	largest possible exponent

...describes numbers of the form:

$$\pm d_0.d_1 d_2 d_3 \ldots d_{p-1} \cdot \beta^e$$

$d_i \in \{0, \ldots, b-1\}$
$d_0 \neq 0$
$e \in [e_{min}, e_{max}]$

Questions?

Exercise

Exercise

Are the following numbers in the set $F^*(2, 4, -2, 2)$?

$0.000 \cdot 2^1 = 0_{10}$
$1.000 \cdot 2^1 = 2_{10}$
$1.001 \cdot 2^{-1} = 0.5625_{10}$
$1.0001 \cdot 2^{-1} = 0.53125_{10}$
$1.111 \cdot 2^{-2} = 0.46875_{10}$
$1.111 \cdot 2^5 = 60_{10}$

Solutions

is in F^*

$1.000 \cdot 2^1 = 2_{10}$
$1.001 \cdot 2^{-1} = 0.5625_{10}$
$1.111 \cdot 2^{-2} = 0.46875_{10}$

is not in F^*

$0.000 \cdot 2^1$ not ”normalizable”
$1.0001 \cdot 2^{-1}$ $5 > p = 4$
$1.111 \cdot 2^5$ $5 \notin [-2, 2]$

Questions?

Exercises

Exercise

State the following numbers in $F^*(2, 4, -2, 2)$ in decimal

1. the largest number
2. the smallest number
3. the smallest non-negative number

How many numbers are in $F^*(2, 4, -2, 2)$?

Solution

largest:	$1.111 \cdot 2^2 = 7.5_{10}$
smallest:	$-1.111 \cdot 2^2 = -7.5_{10}$
smallest > 0:	$1.000 \cdot 2^{-2} = 0.25_{10}$

Normalized Floating Point Systems

Solution

For a fixed exponent there are three digits we can vary freely, and for each number also the negative number is in the set, thus resulting in $2 \cdot 2^3 = 16$ numbers per exponent. There are 5 possible exponents, thus resulting in $5 \cdot 16 = 80$ numbers. Notice that in normalized number systems we cannot count some numbers twice, as we've seen in the lecture that the representation of a number is unique.

Trick

For given $F^*(\beta, p, e_{min}, e_{max})$:

Largest:	$1.11\ldots1 \cdot 2^{e_{max}}$
Smallest:	$-$Largest
Smallest > 0:	$1.00\ldots0 \cdot 2^{e_{min}}$

Questions?

Arithmetic in F^*

Adding floats

1. Bring both numbers to the same exponent
2. Add the significand as binary numbers
3. Re-normalize the sum
4. Round if necessary

Example

$F^*(2, 6, -2, 3)$
$1.125_{10} + 9.25_{10}$
$1.001_2 + 1001.01_2$ (already same exponent ($\cdot 2^0$))
1010.011 (add them like you did in primary school)
$1.010011 \cdot 2^3$ Re-normalizing, adjust e and "cut" for p
$1.01010 \cdot 2^3$ Rounding, like decimal: 1 up, 0 down, and carry
$1.01010_2 \cdot 2^3_2 = 1010.10_2 = 10.5_{10} \neq 10.375_{10}$

Why 10.5 and not 10.375?

Simply because the exact number 10.375 *can't* be represented in the F^* we were given. The nearest number that *is* in the set F^* is 10.5. This is why floats can sometimes be dangerous and we must follow the *floating point guidelines*.

(By the way: It's not 10.25, because we're rounding up in this case, even if the difference to from both 10.25 and 10.5 to 10.375 is 0.125.)

Exercise

Exercise

Add $1.001 \cdot 2^{-1} = 0.5625_{10}$
and $1.111 \cdot 2^{-2} = 0.46875_{10}$
in $F^*(2, 4, -2, 2)$.

Solution

1. Bring both to same exponent, say -1
   $1.001 \cdot 2^{-1} + 0.1111 \cdot 2^{-1}$
2. Add them as binary numbers: $10.0001 \cdot 2^{-1}$
3. Re-normalize: $1.00001 \cdot 2^{0}$
4. Round: $1.000 \cdot 2^{0} = 1_{10} \neq 1.03125_{10}$

Floating Point guidelines

go to Floating_Point_Guidelines.pdf
+ maybe a short live demo on less ugly code

How to Compare Floating Point Numbers

go to Comparing_FP.pdf

in short: don't check for equality,
check for "being-within-tolerance"

```
// Example of "equality"-check function for floats
bool equal(double x, double y, double tol){
   double diff = x - y;
   if(diff < 0){
      diff *= -1;
   }
   return (diff < tol);
}
```

Exam Question

YOUR EXAM WILL BE DIFFERENT. DONT'T FORGET THAT

Geben Sie ein möglichst knappes normalisiertes Fliesskommazahlensystem an, mit welchem sich die folgenden dezimalen Werte gerade noch genau darstellen lassen: jede Verkleinerung von p, $e_{\max}$ oder $-e_{\min}$ muss dazu führen, dass mindestens eine Zahl nicht mehr dargestellt werden kann.
Hinweis: p zählt auch die führende Ziffer.
Tipp: Schreiben Sie sich die normalisierte Binärzahldarstellung der Werte auf, wenn sie für Sie nicht offensichtlich ist.

Provide a smallest possible normalized floating point number system that can still represent the following values exactly: any decrease of the numbers p, $e_{\max}$ or $-e_{\min}$ must imply that at least one of the numbers cannot be represented any more.
Hint: p does also count the leading digit.
Tip: Write down the normalized binary representation of the values, if it is not obvious for you.

Werte / *Values*: $2.25,\ \frac{1}{8},\ 0.5,\ 16.5,\ 2^3$

$F^*(\beta, p, e_{\min}, e_{\max})$ mit / *with*

$\beta = 2 \quad , p = \qquad , e_{\min} = \qquad , e_{\max} = \qquad .$

Exam Question

YOUR EXAM WILL BE DIFFERENT. DONT'T FORGET THAT

Geben Sie ein möglichst knappes normalisiertes Fliesskommazahlensystem an, mit welchem sich die folgenden dezimalen Werte gerade noch genau darstellen lassen: jede Verkleinerung von p, $e_{\max}$ oder $-e_{\min}$ muss dazu führen, dass mindestens eine Zahl nicht mehr dargestellt werden kann.
Hinweis: p zählt auch die führende Ziffer.
Tipp: Schreiben Sie sich die normalisierte Binärzahldarstellung der Werte auf, wenn sie für Sie nicht offensichtlich ist.

Provide a smallest possible normalized floating point number system that can still represent the following values exactly: any decrease of the numbers p, $e_{\max}$ or $-e_{\min}$ must imply that at least one of the numbers cannot be represented any more.
Hint: p does also count the leading digit.
Tip: Write down the normalized binary representation of the values, if it is not obvious for you.

Werte / *Values*: $2.25,\ \frac{1}{8},\ 0.5,\ 16.5,\ 2^3$

$F^*(\beta, p, e_{\min}, e_{\max})$ mit / *with*

$\beta = 2$, $p = 6$, $e_{\min} = -3$, $e_{\max} = 4$.

Questions?