Float Imprecision Identifier

Michael Tegegn

BACKGROUND: Floating points Numbers

- Most natural way of representing numbers
- Modern computers use IEEE 754 floating point standard
- Different formats:
  - Single precision (float32)
  - Double precision (float64)
  - Extended precision (80 bit floating point)
  - …

MOTIVATION: Floating point analysis

“Floating point numbers are like piles of sand; every time you move them around, you lose a little sand and pick up a little dirt.” — Brian Kernighan and P. J. Plauger

- Floating point numbers are unreliable
- Imprecision propagates through arithmetic computation
- Why not use the format with the highest precision?
  - Memory and Time cost

Floating Point Imprecision Identifier

- Language:
  - SmallVM + Floating point Support
- Floating point operations:
  - Arithmetic + Casting + Comparison
- Value:

  | Integer Interval

  | Float Interval
- 32 bit and 64 bit floats only

Cont…

- For floating point values:
  - *FloatInterval of f32Interval * f64Interval*
  - Keep track of both 32-bit representation and 64-bit representation
- Example: 1.1 + 2.1 = (3.2(float32), 3.2(float64))
- Similar widening and narrowing operators used

Precise Analysis

- What to do when we have to choose between the two values?
- Casting to Int?
  - Always use 64 bit float value
- Filtering?
  - Always use 64 bit float value

Tool

- Let people choose tolerable precision loss
  - precision loss threshold
- precision loss > threshold? Report
- Run: ./analyser <threshold> <*.ll file>

example.c

```c
1  int main() {
2    float a = 1234.56789;
3    float b = 56789.1234;
4    int c = 100;
5    float d = a * b;
6    float e = 1.0;
7    int y = 0;
8    for (int i = 0; i < 10000; i ++) {
9      e = (float) i * (float) i;
10   }
11   return 0;
12 }
```

- # Example Run 1: ./analyzer 0.001 ./test/example.ll
- FP Imprecision @ example.c:main:5:15,
  %mul = (32[70110032.000000, 70110032.000000],
  64[70110029.152527, 70110029.152527]),
  diff = 2.847473
- FP Imprecision @ example.c:main:9:19,
  %mul2 = (32[0.000000, 99980000.000000],
  64[0.000000, 99980001.000000]),
  diff = 1.000000

- # Example Run 2: ./analyzer 1.000 ./test/example.ll
- FP Imprecision @ example.c:main:5:15,
  %mul = (32[70110032.000000, 70110032.000000],
  64[70110029.152527, 70110029.152527]),
  diff = 2.847473

Limitations and potential

- Limitations:
  - Built on the assumption that only big values can have the highest discrepancy
  - Some numbers are more difficult to represent
- Potential application:
  - Grow tool to check for imprecision of every IEEE floating point format
  - Use to compare floating point safety of two semantically similar programs

Thank you!

Questions?