CMPSCI 630 Systems **Spring 2023**

Lecture 8

Lecturer: Emery Berger *Scribe: Allison Poh*

8.1 Project Details

Project Title: OUROBOROS (named after snake eating its own tail)

- Python makes it easy to see byte code: use dis (disassembler) module
- In below example, we may think that if we see foo, we can just replace foo with 42. But this is incorrect since python is a dynamic language.

```
def foo(n):
    return 42
```

- Other dynamics: can change the way operators work (e.g., +); in C++ this is called operator overloading
- Take a best effort approach: try to write compiler optimizations; rather than looking for every single one, find ones that will not destroy everything
- IR = low level representation (usually looks like a control flow graph)
  - Static Single Assignment (SSA) is a widely used IR that introduced new variables (e.g., x → x_prime) and essentially turned imperative language into funcitonal language.
  - For this project, we will build a src-to-src compiler (i.e., python code → do something → python code)
- Visitor pattern: ast_functionDef(...)
  - Have a class that, by default, does nothing for nodes
  - When we have a class that's special, then the visitor will visit it
  - For this project, we have to use this to process ASTs (or we could alternatively write a function that walks the tree)
- First step: write simple visitors just to get the feel for the code (start small and get accustomed to this way of programming)
- Project may seem straightforward but there are a lot of gotchas. Juan created over 100 tests, so make a lot of tests and share tests with each other in the class

8.2 FDIV Bug

- Dividing a float in a certain range with another float in another range caused a substantial error.
- This error was baked into a chip, leading to a recall of intell processors (there are known errors all the time, but they aren't nearly as bad as this one).
- Led to verification:
  - 1 bit / 1 bit requires 4 tests
  - 32 bits / 32 bits requires 2^64 tests
  - 64 bits / 64 bits requires 2^128 tests
  - Exhausive testing does not work for large state spaces

8.3 Verification

- Formal verification: prove using math (someones PhD dissertation was on formal verification of FDIV bug)
  - "Testing can only reveal the presence of bugs but not their absence" (not exactly true, but justifies formal verificaiton)
- Model checking: the in-between of exhaustive testing and formal verification
  - The standard approach is to perform automatic testing up to size n, collapsing identical states.
- What does TicTok do? What is the formal specificaiton of TickTok?
  - What does it mean to be correct? Well, correct with respect to its specification. But is the specificaiotn correct? Is the meta specificaiton correct? ...
  - Formal verificaiton is hard since it's hard to specify things.
- quicksort example:
  - Requirements of quicksort: needs to terminate and, if the input is a multiset, it needs to output a sequence
  - Fairly simple specification: $\forall i, j$ where $i \geq 0, i \leq n, j \geq 0, j \leq n \therefore i \leq j : \text{output}[i] \leq \text{output}[j]$
  - But what if sorting strings? What does $\leq$ mean then? And what about empty string? Lowercase vs. uppercase? ... Specification becomes long and, once specification is long, it's just another "identical program" that is hard to read.
- The Spec Problem: the specification is not useful as it is not "clearly" correct
- The Oracle Problem: where's the truth? if no oracle, then no truth and specification could be wrong
- Total correctness: the program = the specification
- Partial correctness: the program properties are a subset of the specification (e.g., program never divides by 0)

8.4 EPIC

- EPIC: Explicitly Parallel Instruction Computing; created by Intel and HP (originally called VLIW for Very Long Instruction Word)
- If all the below are independent, then they can run in parallel:

```
ADD
ADD
MULT
SUB
```

- Only make a big chunk if you know that everything can be run in parallel
- This shifts the responsibility of parallelism to the compiler ... But there's a big problem as this can't be done (e.g., 18 car lane example)
- The plan was to create a new chip, Itanium, that uses EPIC. The chip was an unbelievable failure and was nicknamed Itanic.
  - Major compatibility problems since it used a totally different instruction set
  - This is impossible in C because of aliasing of pointers (i.e., pointers p and q can point to the same object)
  - Cominatorial/state space explosion