Chopped Symbolic Execution

- **Problem**: heavily-branching uninteresting code might hinder deep exploration
- **Solution**: mark code, create state snapshot, skip code, recover snapshot and execute *relevant* branches of skipped code if side-effects influence current state
- David will present his work later

Trabish, Mattavelli, Rinetzky, Cadar: *Chopped Symbolic Execution*, ICSE 2018
Floating-Point Arithmetic

- **Problem**: missing floating point support in KLEE
- **Solution**: implement it—twice

JIT Fuzzing Solver

- **Problem**: traditional constraint-solving is often slow and boring
- **Solution**: take a set of constraints, translate them into a program, if input traverses only true branches it represents satisfying assignment, use fuzzer to find these inputs

Liew, Donaldson, Cadar (on-going work)
Quality of Symbolic Executors

- **Problem**: many people use symbolic executors to test software, only few people test symbolic executors
- **Solution**: combine program generation with differential testing for symbolic executors

Array Constraint Optimisations

- **Problem:** high solving time for constraints involving large arrays
- **Solution:** use semantics-preserving constraint transformations to improve solving time

Perry, Mattavelli, Zhang, Cadar: *Accelerating Array Constraints in Symbolic Execution*, ISSTA 2017
Program Transformations

- **Problem**: path explosion and high solving time
- **Solution**: use program transformations to improve solving time and to aid exploration

Binary-level Symbolic Execution

- **Problem:** KLEE executes LLVM bitcode
- **Solution:** add support for native binaries

Busse, Cadar (on-going work)