Progess Report 3

Xie Li

September 15, 2020

Overview of the Progress

- Introduction to KLEE and static analyzer of AbsInt
- A summary of the tools: capabilities and algorithms.
- A survey paper.
- Currently working on designing data structures for later development.

Tool Survey: KLEE

Usage:

```
llvm-gcc --emit-llvm -c code.c -o code.bc
klee --max-time 2 --sym-args 1 10 10--sym-files 2
2000 --max-fail 1 code.bc
```

Algorithm and Techniques:

- Symbolic Execution.
- Maintaining of path condition of the path.
- Classify dangerous operation and when bug identified use SMT solver to find concrete value.

Application:

KLEE was applied and reach a coverage of 81% of GNU `Coreutil`. It found totally 10 errors and even some bugs in heavily-tested code.

Tool Survey: Astree of AbsInt

Toolset: Check C code for runtime error: Astree.
Check code guideline: RuleChecker.
Compiling: CompCert.
Check Stack usage: StackAnalyzer .
Analyze execution time: aiT, TimeWeaver.

Tool Survey: ASTREE

ABSINT focuses on non-functional program errors.

Capability: Check

- Division by zero
- Out-of-bounds array indexing.
- erroreous pointer manipulation and dereferencing
- interger and floating-point arithmetic overflow
- read uninitialized variables
- data races
- inconsistent locking
- violation of user-given assertions
- unreachable code

Summary of Tools

- CBMC: verifies memory safety (array bounds and safte use of pointers), check for exceptions. Algorithm used: bounded model checking.
- NuSMV: a symbolic model checker utilizing BDD library and able to model and check.
- CPAchecker: a static analyzing tool capable of doing data-flow analysis and automatic testing. Algorithm used: CPA algorithm which integrate several static analysis algorithm based on abstract intepretation, symbolic execution for automatic testing and CEGAR loop for refining.
- Infer: A static analysis tool used mainly for finding bugs for programs that manipulating heaps and memory. Algorithm: inference of separation logic and invariant synthesis using shape analysis, incorrectness logic inference.
- KLEE: A testing tool based on llvm use symbolic execution for automatic testing.
- Astree: A static analyzer detecting runtime error. Algorithm: Abstract intepretation.

Functionalities of Tools

Toolname	AI	CE	BMC	SMC	SE	Conc	SA
CBMC		×	×		×		
CPAchecker	×	×	×(dep.)		×		×
Infer							×
KLEE				×			
Astree	×			×		×	

Survey Paper

A Survey of Automated Techniques for Formal Software Verification

	Tool name	Tool developer	Symbolic analysis	Abstraction	Counterexample	BMC	Concurrency	Languages
II	ASTRÉE	École Normale Supérieure	×	×				C (subset)
	CODESONAR	Grammatech Inc.	×	×				C, C++, ADA
	PolySpace	PolySpace Technologies	×	×			×	C, C++, ADA, UML
	PREVENT	Coverity	×	×			×	C, C++, Java
III	BLAST	UC Berkeley/EPF Lausanne	×	×	×		×	C
	F-SOFT (abs)	NEC	×	×	×			C
	Java PathFind.	NASA	×		×	×	×	Java
	MAGIC	Carnegie Mellon University	×	×	×		×[1]	C
	SATABS	Oxford University	×	×	×		×	C, C++, SpecC, SystemC
	SLAM	Microsoft	×	×	×		×	C
	SPIN	Bell Labs[2]			×	×	×	PROMELA, C[3]
	ZING	Microsoft Research			×	×	×	ZING (object oriented)
IV	CBMC	CMU/Oxford University	×		×	×		C, C++, SpecC, SystemC
	F-SOFT (bmc)	NEC	×		×	×		C
	EXE	Stanford University	×		×	×		C
	SATURN	Stanford University	×		×	×		C

LLVM Pass & LLVM C++ API

Compiling of llvm pass:

```
 clang ‘llvm-config --cxxflags‘ -Wl,-znodelete
-fno-rtti -fPIC -shared [Passname].cpp -o
[Passname].so ‘llvm-config --ldflags‘  opt -load
./[Passname].so -[Pass] [filename].ll
```

Compiling of llvm api:

```
 c++ IRReaderTest.cpp ‘llvm-config --cxxflags
--ldflags --libs core‘ -o a
```

Later Work

- Design of the abstract data structure.
- Building the gap between LLVM data structure and our own data structure.