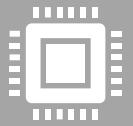
Alive2: Bounded Translation Validation for LLVM

Nuno Lopes
Juneyoung Lee, Chung-Kil Hur
Zhengyang Liu, John Regehr

Microsoft Research
Seoul National University
University of Utah

Bugs in LLVM

Simple implementation bugs



Complex architectural flaws

What's the impact of a different semantics? What breaks?

How to guarantee a fix is complete across millions of LoC?

Specification document (LangRef) is huge and not always clear

Impact of compiler bugs is critical (security incl)

We need semantics for verification

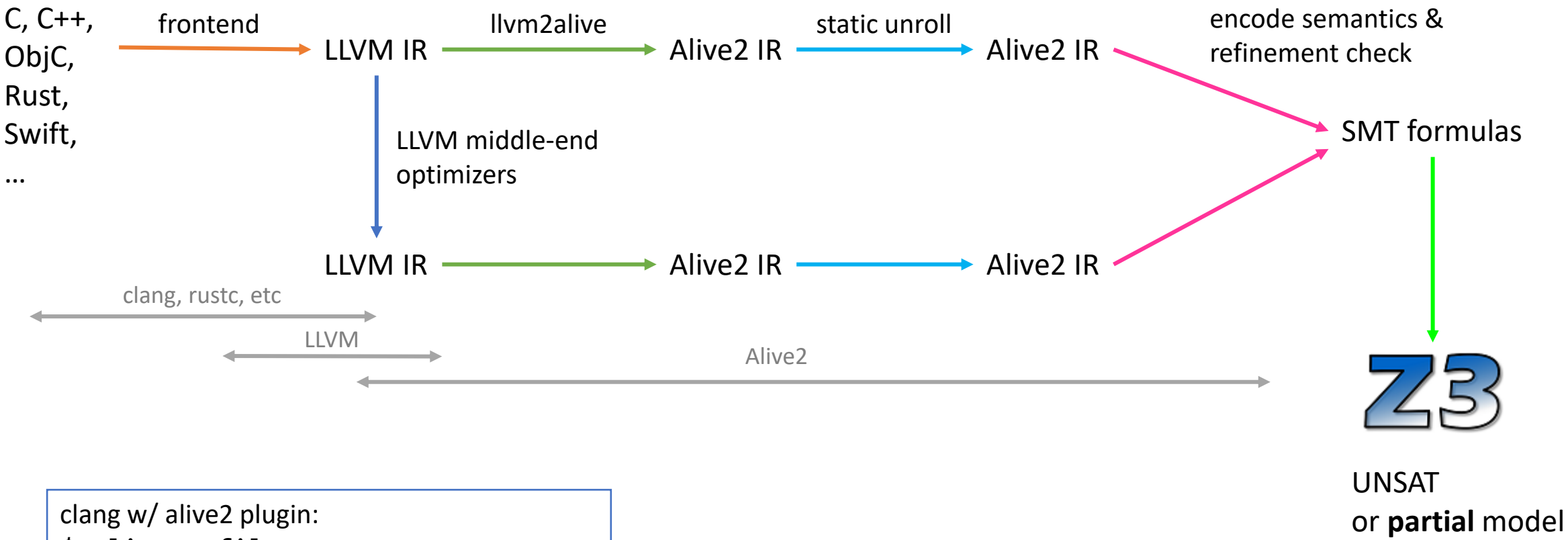
```
select %c, %a, %b
```

	UB if c poison + conditional poison	UB if c poison + poison if either a/b poison	Conditional poison + non-det choice if c poison	Conditional poison + poison if c poison	Poison if any of a/b/c poison
control-flow → select	✓		✓	✓	
select → control-flow	✓	✓			
select → arithmetic		✓			✓
select removal	✓	✓		✓	✓
select hoisting	✓	✓	✓		
easy movement			✓	✓	✓

Which one is the best and why?

Which one LLVM uses?

Simple Integration with LLVM



clang w/ alive2 plugin:

```
$ alivecc file.c
```

opt plugin:

```
$ opt -tv -instcombine -tv file.ll
```

Continuous verification

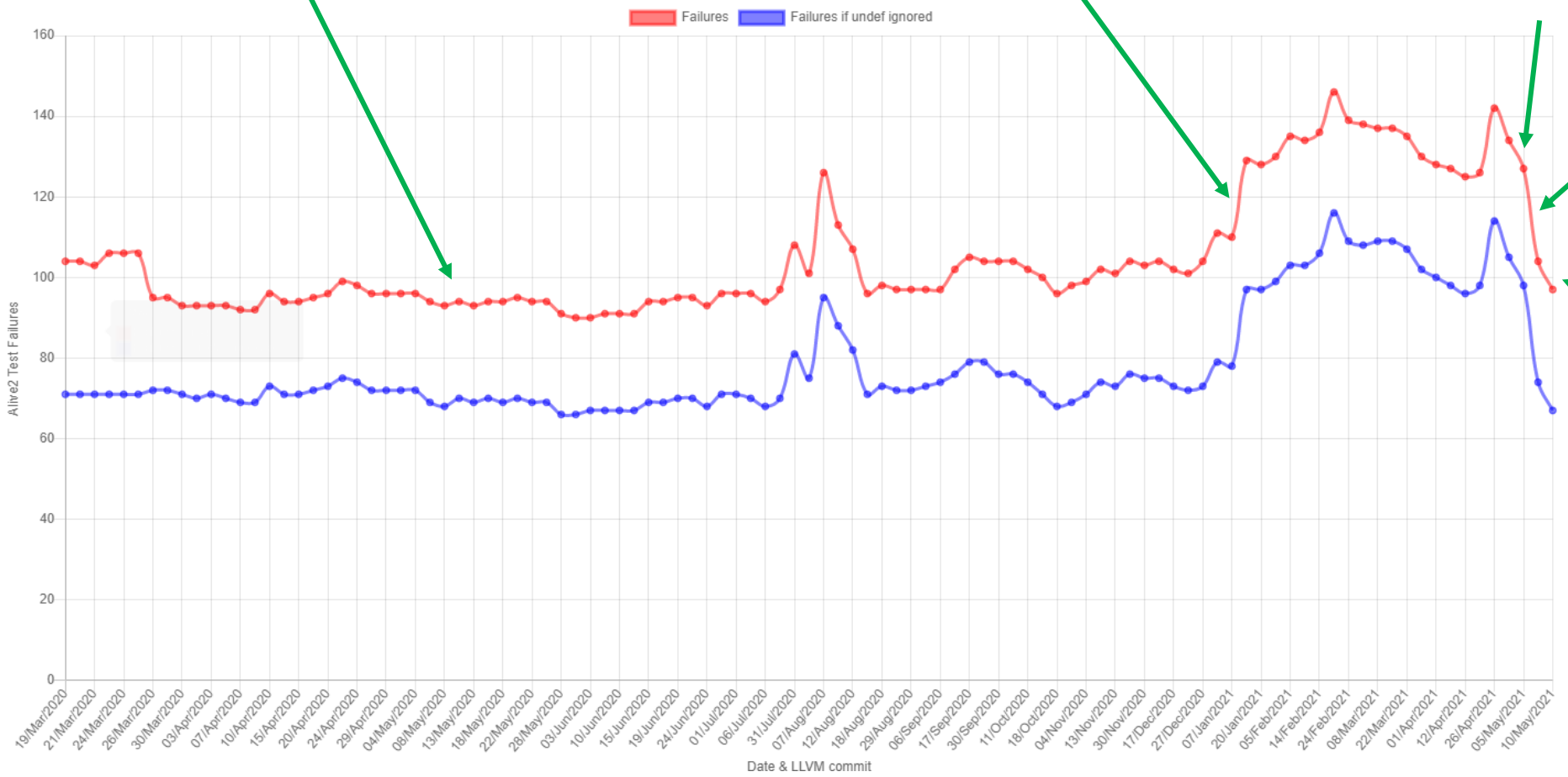
Alive2 adds support for more LLVM features
Finds new bugs in LLVM; fixed at same pace

LLVM adds new unit tests
for select issues

Fix SimplifyCFG bug

Fix long-standing
InstCombine bug re
select instruction

Fix regression in Alive2
when passing null
pointers as arguments to
function calls



Alive2

- Ensure LLVM adheres to **a** specification
- We reported 54 bugs in LLVM so far
- Adding support for an LLVM feature usually uncovers some bugs in LLVM
- Actively used by LLVM developers

- Requires zero changes to LLVM
- Fully automatic
- Easy to use with clang/opt plugins
- Very low false-positive rate

https://alive2.llvm.org

https://github.com/AliveToolkit/alive2