NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE | College of Computing and Data Science

Verified VCG and Verified Compiler for Dafny

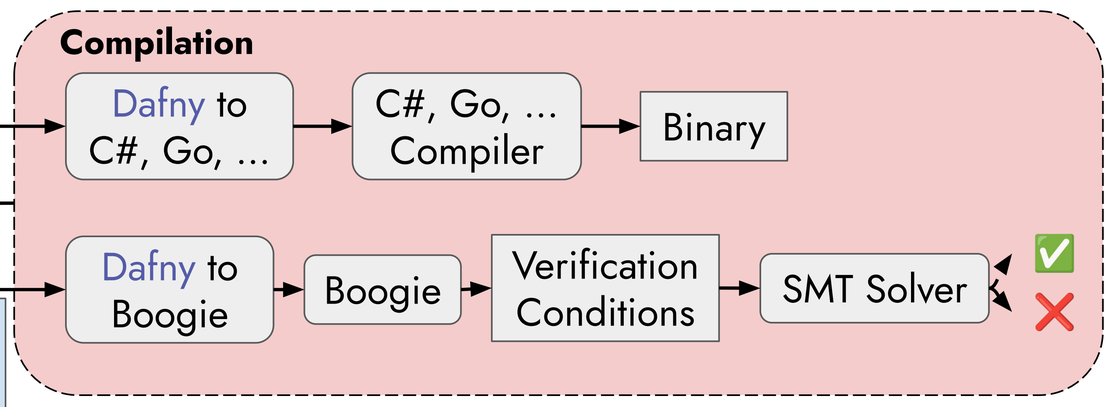
Presented By: Daniel NEZAMABADI

Supervisors: Asst. Prof. Yong Kiam TAN / Prof. Magnus MYREEN (Chalmers)

How does Dafny verify this method and produce machine code?

```
method McCarthy(n: int) returns (r: int)
  ensures r == if n <= 100 then 91 else n - 10
  decreases 111 - n
{
  if n <= 100 {
    var tmp := McCarthy(n + 11);
    r := McCarthy(tmp);
  } else {
    r := n - 10;
  }
}
```

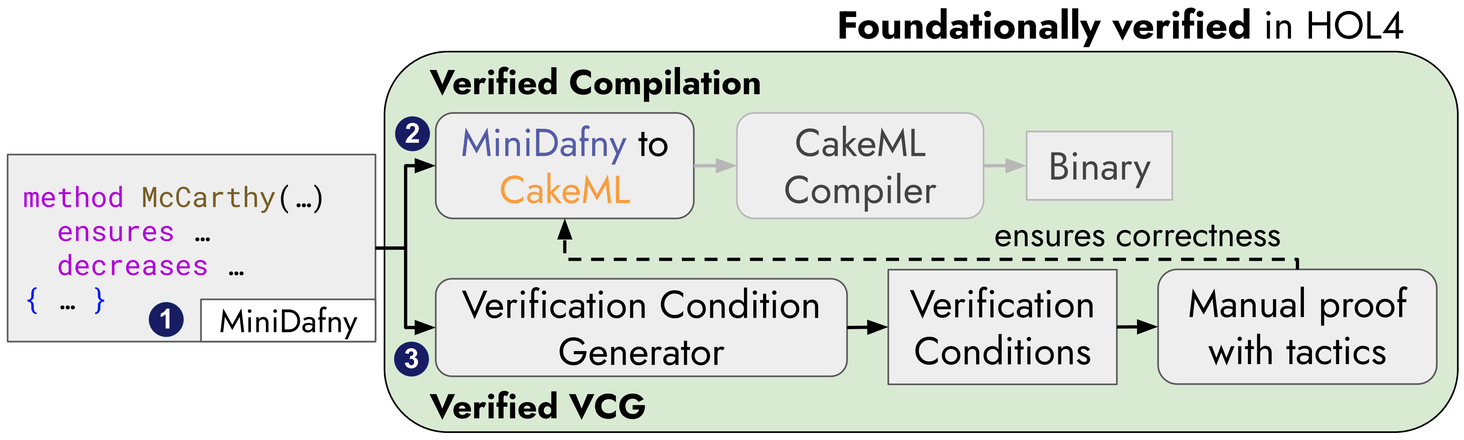
Nested recursion can be a challenge for verifiers: need to establish *ensures* for the first call to prove *decreases* for the second call.



Problem: A lot of non-trivial code must be trusted, and past work has identified (soundness) bugs.

How does our work improve Dafny's trust story?

Answer: Reduce the trusted base through end-to-end formal verification of the pipeline in a proof assistant (HOL4)

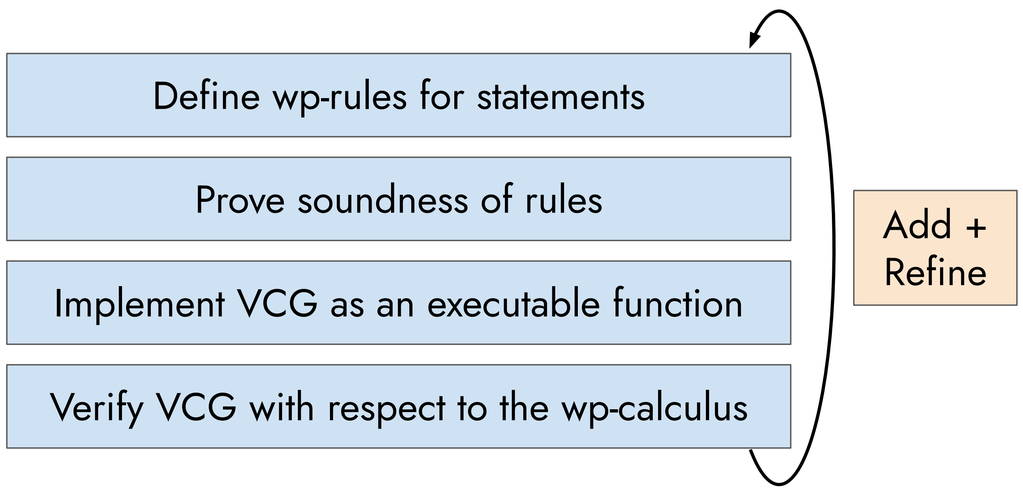


Grayed out parts existed before this project.

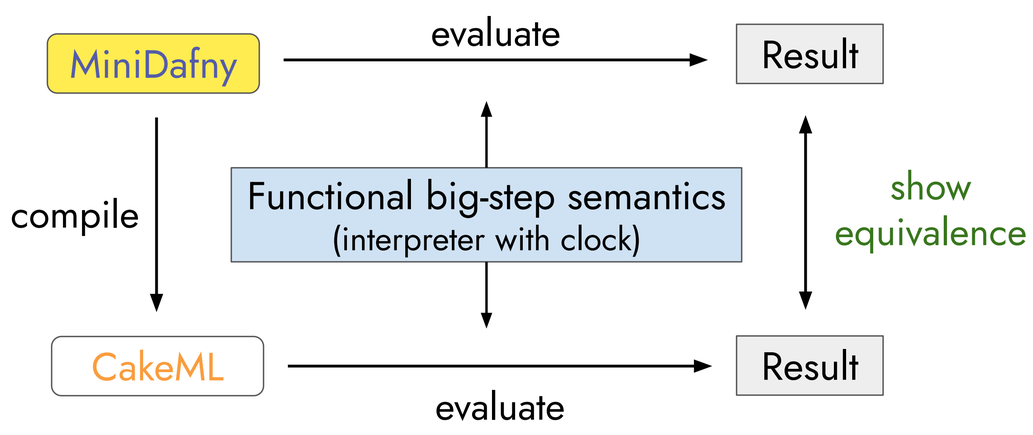
Our Contributions:

1. Functional big-step semantics modeling a non-trivial subset of Dafny (MiniDafny)
   - Complex control flow: mutual recursion, while loops, early returns
   - Imperative features: variables, arrays
2. Verified compiler from MiniDafny to CakeML, then to machine code (x64, ARM8)
3. Verified weakest precondition (wp) calculus and verification condition generator (VCG)

How do we show VCG correctness?



How do we show compiler correctness?



Putting It All Together

```
compile mccarthy = inr mccarthy_cml ∧ … ⇒
  AppReturns (INT n) (… [mccarthy_cml] …)
    (INT (if n <= 100 then 91 else n - 10))
```

Application of our compiler

Hoare Triple

Theorem: "For any integer *n*, the compiled McCarthy method returns 91 if *n* is at most 100, and *n* - 10 otherwise."

By building on the verified CakeML compiler, our *guarantees extend down to machine code*.

Future work:

- automated proofs via SMT solvers
- more supported Dafny features
- deeper integration with CakeML ecosystem

Code | Paper | Talk