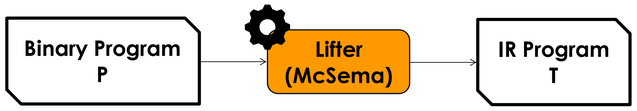
A Scalable Validation of Binary Lifters, PLDI'20

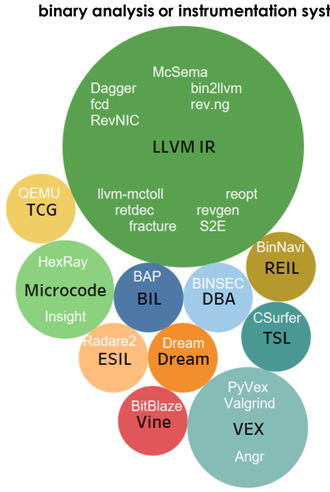
github.com/sdasgup3/validating-binary-decompilation

Binary Lifting



Lifting Pivotal in Binary Analysis

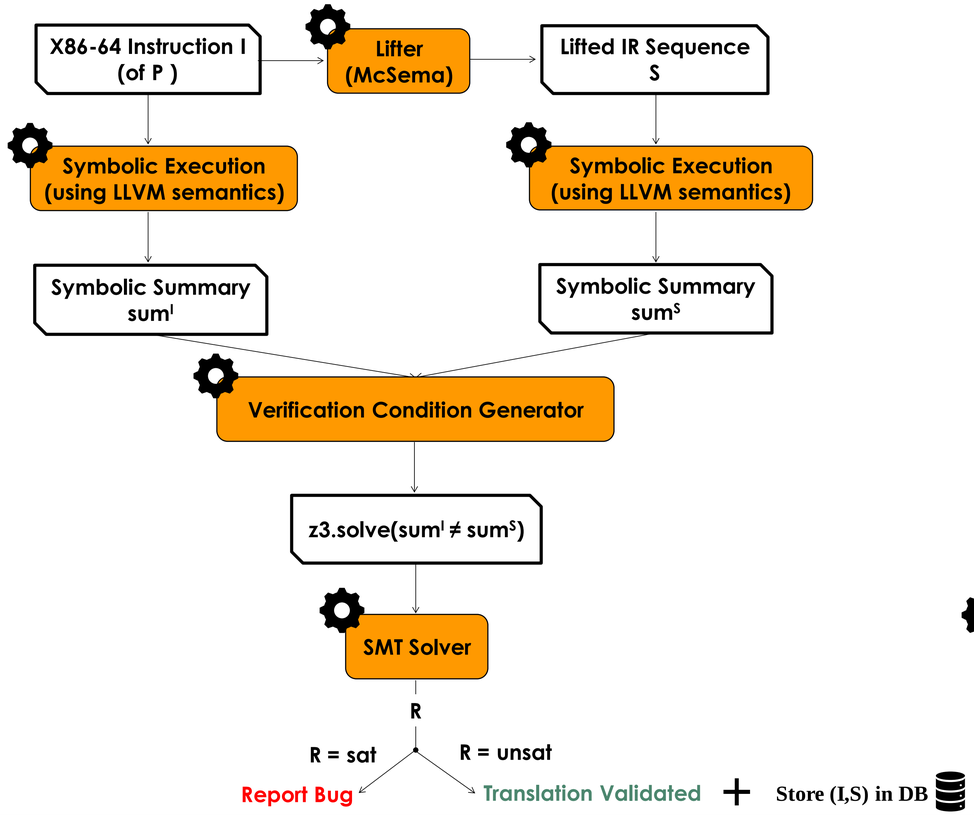
† Each circle represents an IR used by the lifter embedded in one or more binary analysis or instrumentation systems



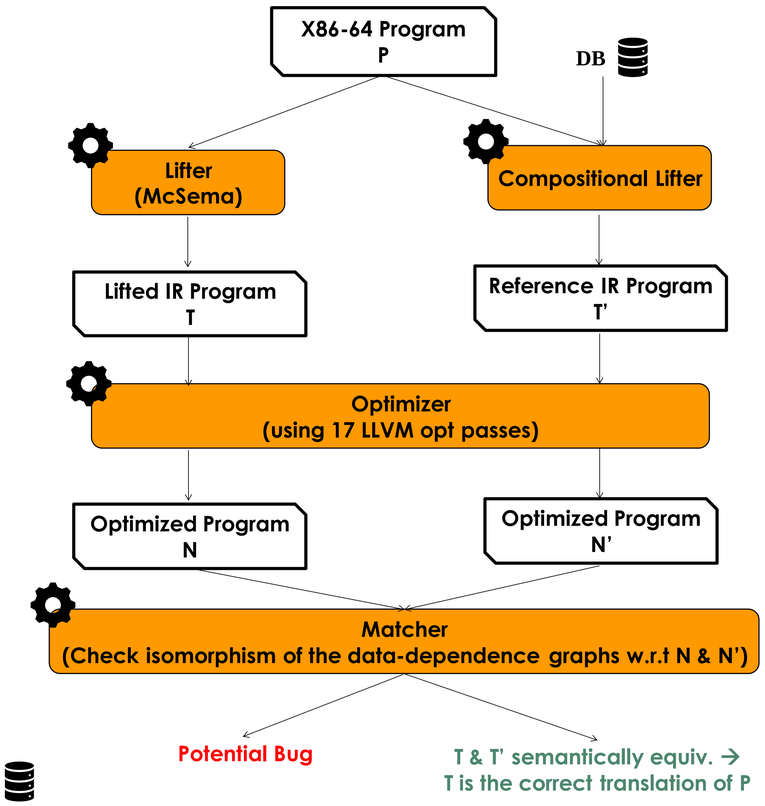
Faithful lifting makes binary analysis results reliable

Our Approach

Single Instruction Translation validation (SITV)



Program Level Validation (PLV)



Challenges

Manual encoding of the effects of binary instructions is hard

- Vast number of instructions in modern ISAs
- Standard manuals are often ambiguous exhibiting inconsistent behaviors of variants

Develop formal & informal techniques to validate the correctness of binary lifting from P to T

Evaluation & Results

SITV

- Evaluated SITV on 1349 out of 3736 x86-64 instruction variants. Excluded ones are un-supported either by McSema or by LLVM semantics
- Found 29 solver failures; all acknowledged as bugs by McSema developers

PLV

- Evaluated PLV on 2348 LLVM single-source benchmark functions
- Matcher proved the correctness of translations with 93% success rate; Remaining 7% are manually inspected as false alarms

Autotuning based Matcher

Observation

- Changing the order of normalizer passes affects the matching results
- Not all 17 opt. passes are needed for every pair of functions

Intuition

To frame the problem of selecting the optimal normalizing pass sequence, for each function pair, as an application of program autotuning

Improved Results

False alarm rate reduced from 7% to 4%

Find us on PLDI Slack: @Sandeep Dasgupta