Software Engineering
Programming Languages

universität ulm

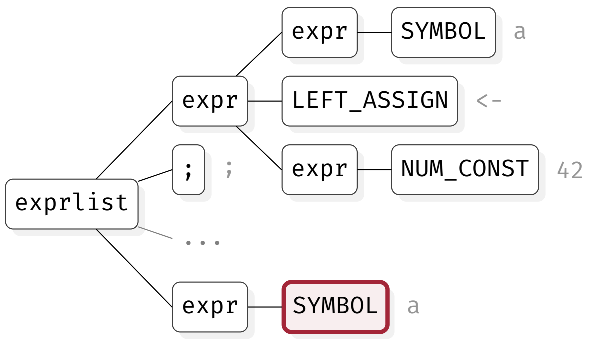
Static Dataflow Analysis of R Programs

Florian Sihler, Matthias Tichy

1) Parse

We rely on the R parser to convert the input files into an abstract syntax tree (AST).

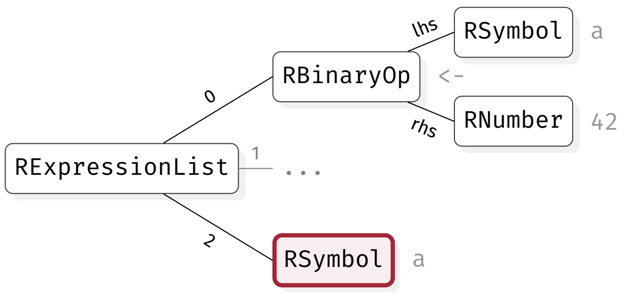
The figure on the right illustrates the AST for the R code “`a <- 42; b <- 3;` **`a`**”.



2) Normalize

Subsequently, we create a uniform and version-independent representation of the AST, abstracting away from the intricacies of the R language.

Currently, we support R versions 3.6 – 4.4.1.

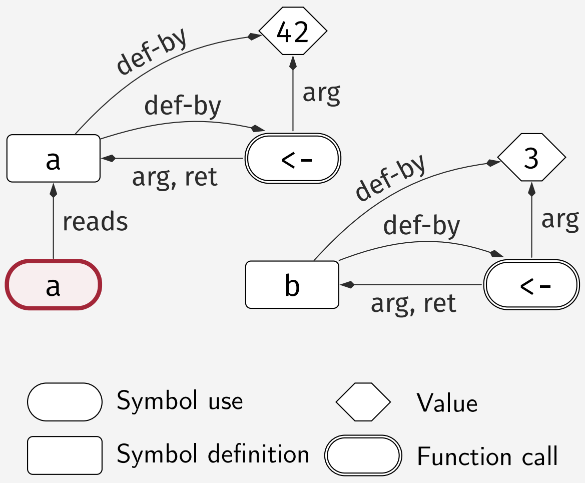


3) Dataflow Analysis

The analysis uses a syntax-guided approach in the form of a stateful fold over the normalized AST, intertwining dataflow and control flow analysis. Fold handlers are dispatched dynamically using an abstract interpretation of the R environment (mapping symbols to their definitions).

We support side-effects, higher-order functions, sourced files, and more of R’s dynamic nature.

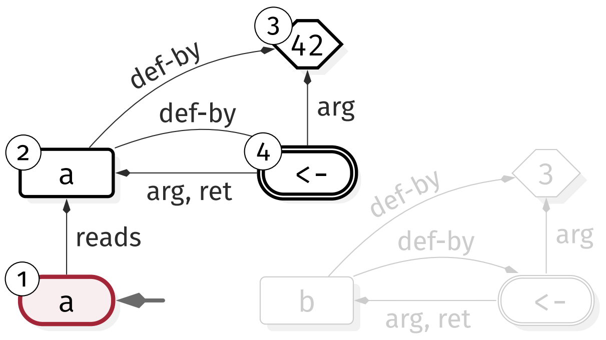
The figure on the right illustrates the incremental construction of the dataflow graph, desugaring `<-` into a function call (mirroring R’s semantics).



4) Backward Program Slicing

A slice for a program point is a subset of the program containing only the parts that may have an influence on the computation at that point. On the dataflow graph, this reduces to a reachability problem.

In practice, slicing for points of interest achieves an average reduction of 87.3 % in lines of code.



5) Reconstruction

As a final step, we use the nodes selected by the backward program slicing to reconstruct an executable R program.

```
a <- 42
a
```

flowR
A program slicer and dataflow analyzer for the R programming language.

Available for Visual Studio Code and RStudio.

87.3% average reduction in LOC when slicing for points of interest

average time to analyze an R program (without caching) **473 ms**

96.1% identical results with automated input-output equivalence testing

4103 real-world research artifacts

October 30, 2024 Institute of Software Engineering and Programming Languages Ulm University