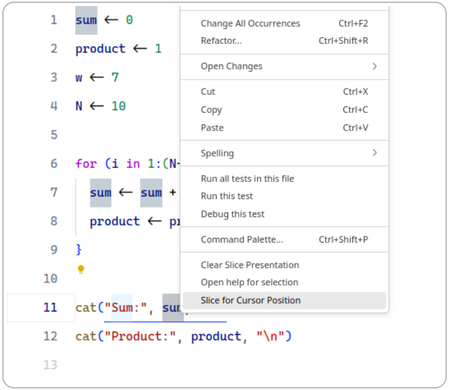
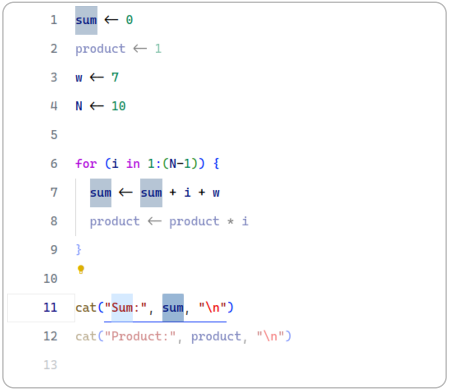


```
sum ← 0
product ← 1
w ← 7
N ← 10

for (i in 1:(N-
  sum ← sum +
  product ← pı
}

cat("Sum:", sur
cat("Product:", product, "\n")
```



```
sum ← 0
product ← 1
w ← 7
N ← 10

for (i in 1:(N-1)) {
  sum ← sum + i + w
  product ← product * i
}

cat("Sum:", sum, "\n")
cat("Product:", product, "\n")
```

# *flowR*: A Program Slicer for the R Programming Language

deRSE ’24 | Ulm University | Florian Sihler and Prof. Matthias Tichy | March 6, 2024

Software Engineering
Programming Languages

# The R Programming Language

# The R Programming Language

- R is mainly designed for statistical computing[1]

[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]

[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]

[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]

[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains

[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains
- Several problems in practice[2, 4]

[4] Wonsil et al., “Reproducibility as a Service” (2023, Software: Practice and Experience)
[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
    - Heavily used in research (e.g., social science)[2]
    - Ranks 6th on PYPL[3]
    - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains
- Several problems in practice[2, 4]
    - Replication

[4] Wonsil et al., “Reproducibility as a Service” (2023, Software: Practice and Experience)
[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains
- Several problems in practice[2, 4]
  - Replication
  - Program comprehension

[4] Wonsil et al., “Reproducibility as a Service” (2023, Software: Practice and Experience)
[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains
- Several problems in practice[2, 4]
  - Replication
  - Program comprehension
  - Missing tool support

[4] Wonsil et al., “Reproducibility as a Service” (2023, Software: Practice and Experience)
[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# The R Programming Language

- R is mainly designed for statistical computing[1]
  - Heavily used in research (e.g., social science)[2]
  - Ranks 6th on PYPL[3]
  - Over 20 000 packages on CRAN[1]
- Most users are from non-computer-science domains
- Several problems in practice[2, 4]
  - Replication
  - Program comprehension
  - Missing tool support
- Demand for software engineering practices[5]

[5] Thimbleby, “Improving Science That Uses Code” (2023, Oxford University Press)
[4] Wonsil et al., “Reproducibility as a Service” (2023, Software: Practice and Experience)
[3] https://pypl.github.io/ [archived]
[2] Trisovic et al., “A Large-Scale Study on Research Code Quality and Execution” (2022, Nature Publishing Group)
[1] https://cran.r-project.org/

# R Scripts

**We analyzed 4 083 R-files**[7]

[7] Sihler et al., “On the Anatomy of Real-World R Code for Static Analysis” (2024, MSR)

# R Scripts

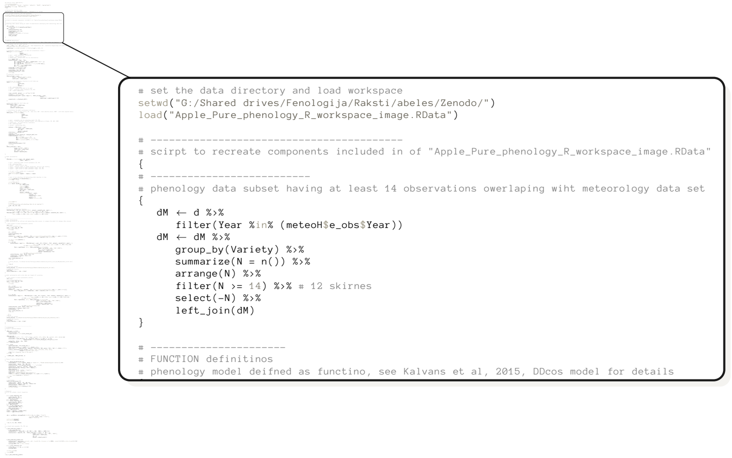
**We analyzed 4 083 R-files**[7]

[7] Sihler et al., “On the Anatomy of Real-World R Code for Static Analysis” (2024, MSR)

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts

## We analyzed 4 083 R-files[7]

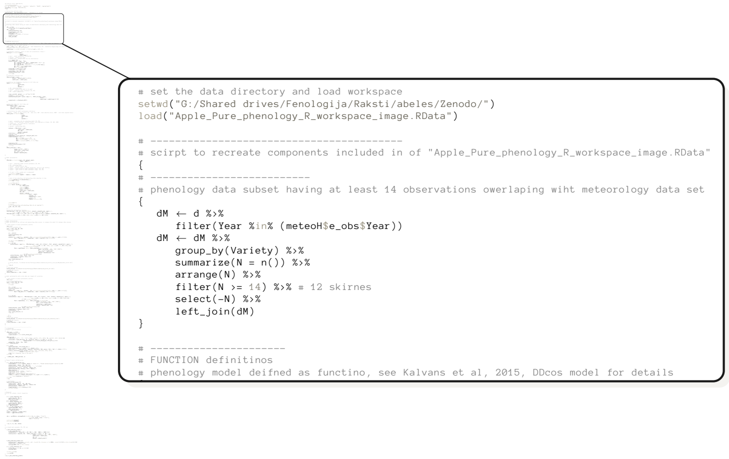


```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# -----------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations owerlaping wiht meteorology data set
{
   dM ← d %>%
      filter(Year %in% (meteoH$e_obs$Year))
   dM ← dM %>%
      group_by(Variety) %>%
      summarize(N = n()) %>%
      arrange(N) %>%
      filter(N >= 14) %>% # 12 skirnes
      select(-N) %>%
      left_join(dM)
}

# ----------------------
# FUNCTION definitinos
# phenology model deifned as functino, see Kalvans et al, 2015, DDcos model for details
```

[7] Sihler et al., “On the Anatomy of Real-World R Code for Static Analysis” (2024, MSR)

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

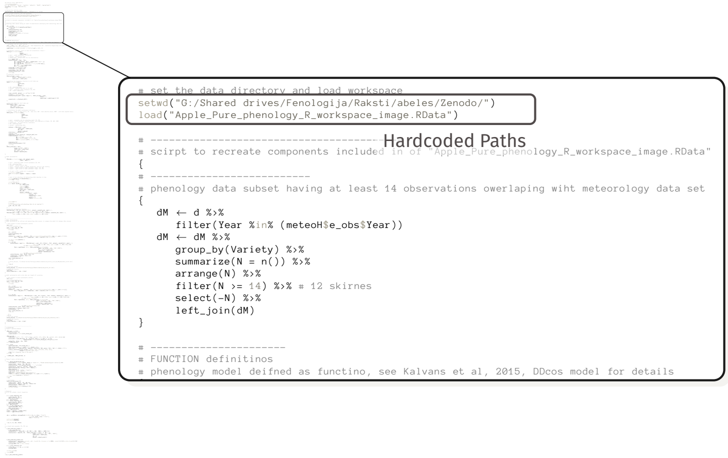
# R Scripts **Fail to Replicate**



```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# ------------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations owerlaping wiht meteorology data set
{
   dM ← d %>%
      filter(Year %in% (meteoH$e_obs$Year))
   dM ← dM %>%
      group_by(Variety) %>%
      summarize(N = n()) %>%
      arrange(N) %>%
      filter(N >= 14) %>% # 12 skirnes
      select(-N) %>%
      left_join(dM)
}

# ----------------------
# FUNCTION definitinos
# phenology model deifned as functino, see Kalvans et al, 2015, DDcos model for details
```

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

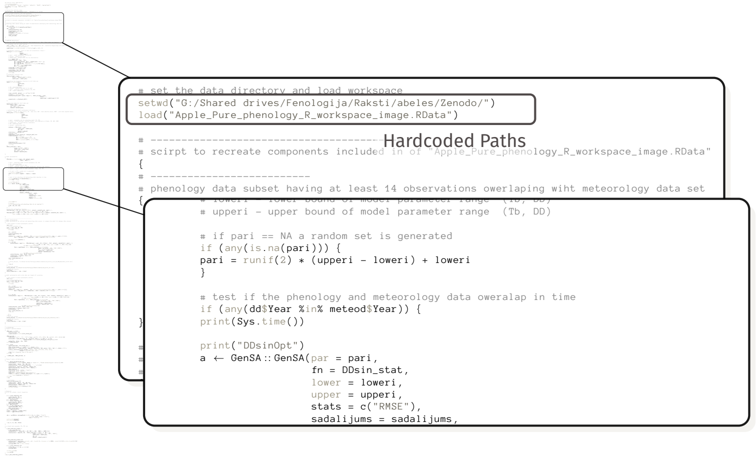
# R Scripts Fail to Replicate



```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# -------------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations overlaping wiht meteorology data set
{
   dM ← d %>%
      filter(Year %in% (meteoH$e_obs$Year))
   dM ← dM %>%
      group_by(Variety) %>%
      summarize(N = n()) %>%
      arrange(N) %>%
      filter(N >= 14) %>% # 12 skirnes
      select(-N) %>%
      left_join(dM)
}

# ----------------------
# FUNCTION definitinos
# phenology model deifned as functino, see Kalvans et al, 2015, DDcos model for details
```

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts **Fail to Replicate**



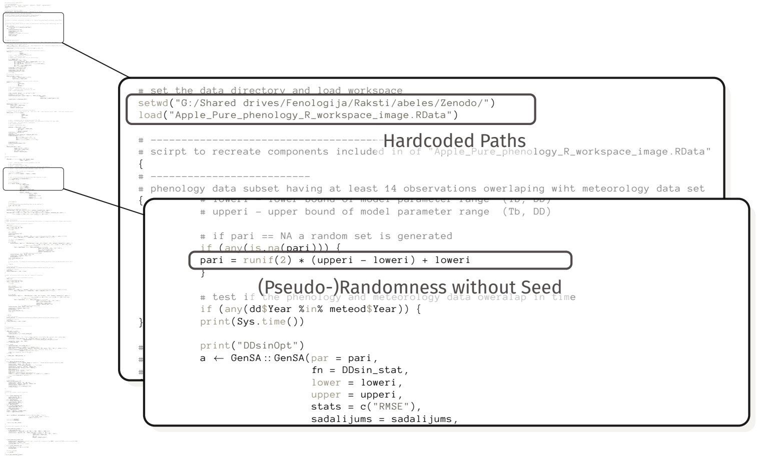
```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# ----------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations overlaping wiht meteorology data set
{
```

Hardcoded Paths

```
        # loweri - lower bound of model parameter range  (Tb, DD)
        # upperi - upper bound of model parameter range  (Tb, DD)

        # if pari == NA a random set is generated
        if (any(is.na(pari))) {
        pari = runif(2) * (upperi - loweri) + loweri
        }

        # test if the phenology and meteorology data oweralap in time
        if (any(dd$Year %in% meteod$Year)) {
        print(Sys.time())

        print("DDsinOpt")
        a ← GenSA::GenSA(par = pari,
                         fn = DDsin_stat,
                         lower = loweri,
                         upper = upperi,
                         stats = c("RMSE"),
                         sadalijums = sadalijums,
```

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts **Fail to Replicate**



```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# ----------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations owerlaping wiht meteorology data set
{
```

Hardcoded Paths

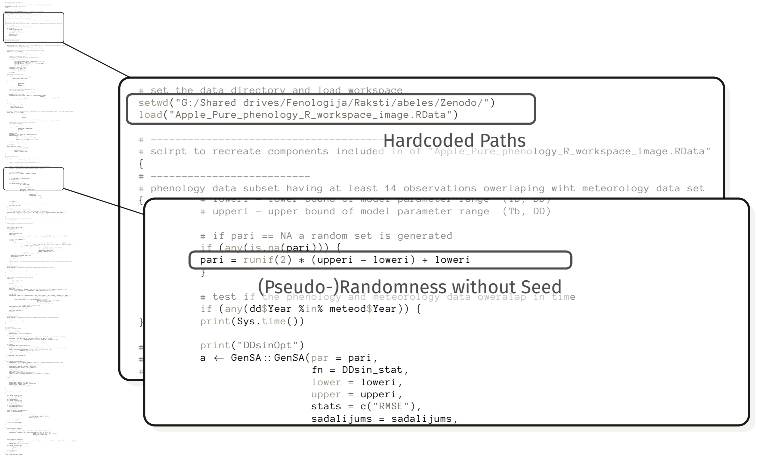
```
    # loweri - lower bound of model parameter range  (Tb, DD)
    # upperi - upper bound of model parameter range  (Tb, DD)

    # if pari == NA a random set is generated
    if (any(is.na(pari))) {
    pari = runif(2) * (upperi - loweri) + loweri
    }

    # test if the phenology and meteorology data overalap in time
    if (any(dd$Year %in% meteod$Year)) {
    print(Sys.time())

    print("DDsinOpt")
    a ← GenSA::GenSA(par = pari,
                     fn = DDsin_stat,
                     lower = loweri,
                     upper = upperi,
                     stats = c("RMSE"),
                     sadalijums = sadalijums,
```

(Pseudo-)Randomness without Seed

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts Fail to Replicate

**74 % even fail to complete![2]**



```
# set the data directory and load workspace
setwd("G:/Shared drives/Fenologija/Raksti/abeles/Zenodo/")
load("Apple_Pure_phenology_R_workspace_image.RData")

# ----------------------------------------
# scirpt to recreate components included in of "Apple_Pure_phenology_R_workspace_image.RData"
{
# --------------------------
# phenology data subset having at least 14 observations overlaping wiht meteorology data set
{
```

Hardcoded Paths

```
    # loweri - lower bound of model parameter range  (Tb, DD)
    # upperi - upper bound of model parameter range  (Tb, DD)

    # if pari == NA a random set is generated
    if (any(is.na(pari))) {
    pari = runif(2) * (upperi - loweri) + loweri
    }

    # test if the phenology and meteorology data overalap in time
    if (any(dd$Year %in% meteod$Year)) {
    print(Sys.time())

    print("DDsinOpt")
    a ← GenSA::GenSA(par = pari,
                      fn = DDsin_stat,
                      lower = loweri,
                      upper = upperi,
                      stats = c("RMSE"),
                      sadalijums = sadalijums,
```

(Pseudo-)Randomness without Seed

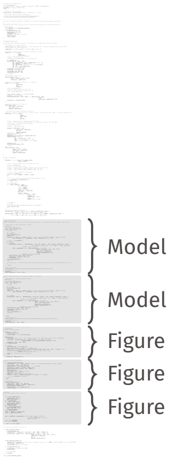
[2] Trisovic et al., "A Large-Scale Study on Research Code Quality and Execution" (2022, Nature Publishing Group)

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts Do Too Much

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts Do Too Much



Model

Model

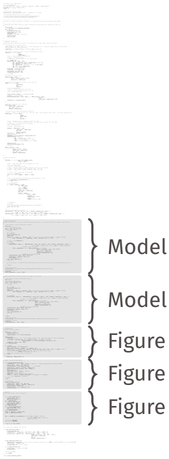
Figure

Figure

Figure

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

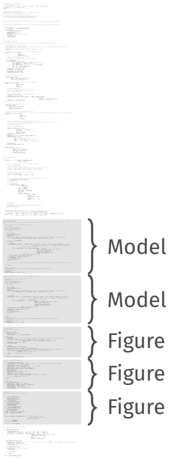
# R Scripts Do Too Much



- Several analyses in one script

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

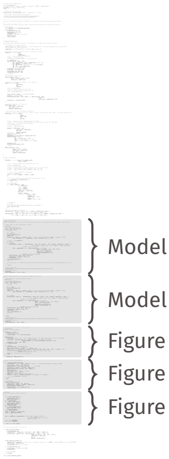
# R Scripts Do Too Much



- Several analyses in one script
- Hard to comprehend

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts **Do Too Much**



- Several analyses in one script
- Hard to comprehend
- Hard to extract/re-use parts

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# R Scripts Are Hard to Analyze

# R Scripts Are Hard to Analyze

```
[8] mod.proj_wc ← ecospat.ESM.Projection(ESM.modeling.output=mod,
        new.env=eval(parse(text = paste("env_",period,"_wc",sep=""))))
```

[8] Ma et al., *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations* (2021, Zenodo) [L. 135f]

# R Scripts Are Hard to Analyze

```
[8] mod.proj_wc ← ecospat.ESM.Projection(ESM.modeling.output=mod,
        new.env=eval(parse(text = paste("env_",period,"_wc",sep=""))))
```

- String-based code evaluation

[8] Ma et al., *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations* (2021, Zenodo) [L. 135f]

# R Scripts Are Hard to Analyze

```
[8] mod.proj_wc ← ecospat.ESM.Projection(ESM.modeling.output=mod,
        new.env=eval(parse(text = paste("env_",period,"_wc",sep=""))))
```

- String-based code evaluation

```
[9] pull.cat ← function(x) {
       bins ← 5
       increments ← (range(x)[2] - range(x)[1])/(bins - 1)
       to_return ← seq(range(x)[1], range(x)[2], increments)
       return(to_return)
    }

    up.cat ← function(new_bins) {
       up_bins = new_bins
       body(pull.cat)[[2]] ↞ substitute(bins ← up_bins)
    }
```

[9] Robertson, *Social hierarchy reveals thermoregulatory trade- offs in response to repeated stressors* (2020, Zenodo) [L. 68ff]
[8] Ma et al., *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations* (2021, Zenodo) [L. 135f]

# R Scripts Are Hard to Analyze

```
[8] mod.proj_wc ← ecospat.ESM.Projection(ESM.modeling.output=mod,
          new.env=eval(parse(text = paste("env_",period,"_wc",sep=""))))
```

- String-based code evaluation

```
[9] pull.cat ← function(x) {
       bins ← 5
       increments ← (range(x)[2] - range(x)[1])/(bins - 1)
       to_return ← seq(range(x)[1], range(x)[2], increments)
       return(to_return)
    }

    up.cat ← function(new_bins) {
       up_bins = new_bins
       body(pull.cat)[[2]] ↞ substitute(bins ← up_bins)
    }
```

- Self-modifying code

[9] Robertson, *Social hierarchy reveals thermoregulatory trade- offs in response to repeated stressors* (2020, Zenodo) [L. 68ff]
[8] Ma et al., *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations* (2021, Zenodo) [L. 135f]

# R Scripts Are Hard to Analyze

```
[8] mod.proj_wc ← ecospat.ESM.Projection(ESM.modeling.output=mod,
          new.env=eval(parse(text = paste("env_",period,"_wc",sep=""))))
```

- String-based code evaluation

```
[9] pull.cat ← function(x) {
       bins ← up_bins # (e.g., 6)
       increments ← (range(x)[2] - range(x)[1])/(bins - 1)
       to_return ← seq(range(x)[1], range(x)[2], increments)
       return(to_return)
    }

    up.cat ← function(new_bins) {
       up_bins = new_bins
       body(pull.cat)[[2]] ↞ substitute(bins ← up_bins)
    }
```

- Self-modifying code

[9] Robertson, *Social hierarchy reveals thermoregulatory trade- offs in response to repeated stressors* (2020, Zenodo) [L. 68ff]
[8] Ma et al., *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations* (2021, Zenodo) [L. 135f]

# R Misses Sophisticated Analysis Tools

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)

- R language server github.com/REditorSupport

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - Reference tracing & Refactorings (rename)

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)
- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - Reference tracing & Refactorings (rename)
- {lintr} github.com/r-lib/lintr

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
    - Syntax-highlighting and auto-completion
    - Refactorings (rename, extract functions and variables)
- R language server github.com/REditorSupport
    - Syntax-highlighting and auto-completion
    - Reference tracing & Refactorings (rename)
- {lintr} github.com/r-lib/lintr
    - Style & syntax errors
    - Potential semantic errors

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - Reference tracing & Refactorings (rename)

- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - Potential semantic errors

- {CodeDepends} github.com/duncantl/CodeDepends

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - Refactorings (rename, extract functions and variables)
- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - Reference tracing & Refactorings (rename)
- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - Potential semantic errors
- {CodeDepends} github.com/duncantl/CodeDepends
  - Dependency analysis
  - Creation of call-graphs

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - ~~Refactorings (rename, extract functions and variables)~~ ← Often wrong (simple heuristics)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - Reference tracing & Refactorings (rename)

- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - Potential semantic errors

- {CodeDepends} github.com/duncantl/CodeDepends
  - Dependency analysis
  - Creation of call-graphs

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - ~~Refactorings (rename, extract functions and variables)~~ ← Often wrong (simple heuristics)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - ~~Reference tracing & Refactorings (rename)~~ ← Often wrong (XPath-Expressions)

- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - Potential semantic errors

- {CodeDepends} github.com/duncantl/CodeDepends
  - Dependency analysis
  - Creation of call-graphs

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - ~~Refactorings (rename, extract functions and variables)~~ ← Often wrong (simple heuristics)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - ~~Reference tracing & Refactorings (rename)~~ ← Often wrong (XPath-Expressions)

- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - ~~Potential semantic errors~~

  ↑ XPath-Expressions, packages

- {CodeDepends} github.com/duncantl/CodeDepends
  - Dependency analysis
  - Creation of call-graphs

# R Misses Sophisticated Analysis Tools

- RStudio IDE posit.co
  - Syntax-highlighting and auto-completion
  - ~~Refactorings (rename, extract functions and variables)~~ ← Often wrong (simple heuristics)

- R language server github.com/REditorSupport
  - Syntax-highlighting and auto-completion
  - ~~Reference tracing & Refactorings (rename)~~ ← Often wrong (XPath-Expressions)

- {lintr} github.com/r-lib/lintr
  - Style & syntax errors
  - ~~Potential semantic errors~~

    ↑ XPath-Expressions, packages

- {CodeDepends} github.com/duncantl/CodeDepends
  - ~~Dependency analysis~~ ← Only top scope
  - Creation of call-graphs

# R Scripts …

# R Scripts …

1. fail to replicate

# R Scripts . . .

1. fail to replicate
2. do too much

# R Scripts …

1. fail to replicate
2. do too much
3. are hard to analyze

# R Scripts …

1. fail to replicate
2. do too much
3. are hard to analyze
4. are not well supported by tools

# R Scripts . . .

1. fail to replicate
2. do too much
3. are hard to analyze
4. are not well supported by tools

Better Software, Better Research

# R Scripts …

1. fail to replicate
2. **do too much**
3. are hard to analyze
4. are not well supported by tools

Better Software, Better Research

# The Goal of flowR

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# The Goal of flowR

- Interested in a single figure

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# The Goal of flowR

- Interested in a single figure

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# The Goal of flowR

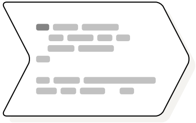
- Interested in a single figure
- ≈ *70 %* reduction

[6] Drudze et al., *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019"* (2021, Zenodo)

# The Architecture

[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

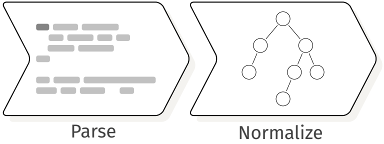
# The Architecture



Parse

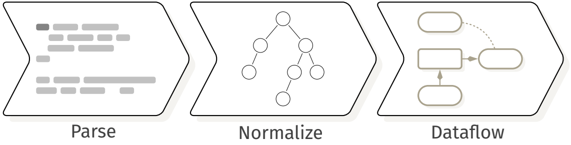
[10] Sihler, "Constructing a static program slicer for R programs" (2023, Ulm University)
[11] Weiser, "Program Slicing" (1984, IEEE Transactions on Software Engineering)

# The Architecture



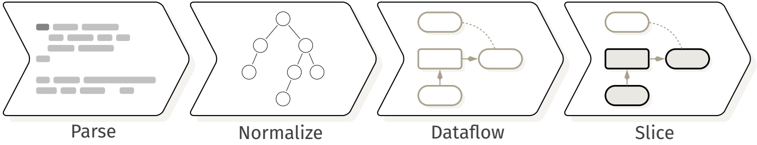
[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

# The Architecture



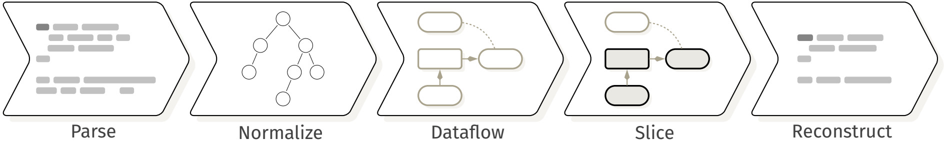
[10] Sihler, "Constructing a static program slicer for R programs" (2023, Ulm University)
[11] Weiser, "Program Slicing" (1984, IEEE Transactions on Software Engineering)

# The Architecture



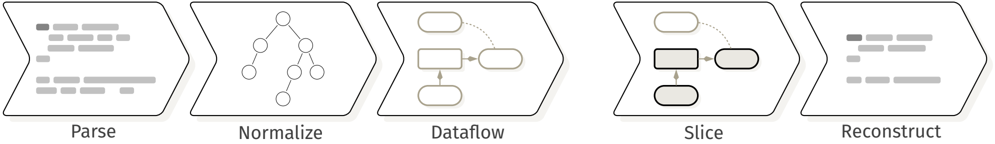
[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

# The Architecture



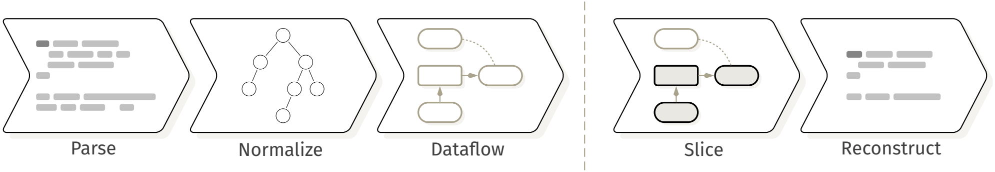
[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

# The Architecture



[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

# The Architecture



[10] Sihler, “Constructing a static program slicer for R programs” (2023, Ulm University)
[11] Weiser, “Program Slicing” (1984, IEEE Transactions on Software Engineering)

# Visual Studio Code Integration

# Visual Studio Code Integration

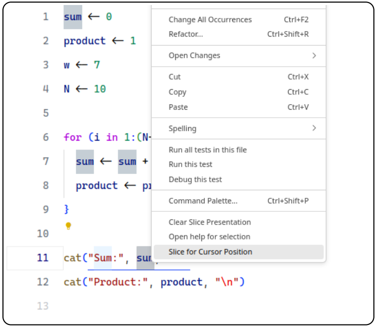
Rudimentary VSCode integration at:

github.com/Code-Inspect/vscode-flowr

# Visual Studio Code Integration

Rudimentary VSCode integration at:

github.com/Code-Inspect/vscode-flowr

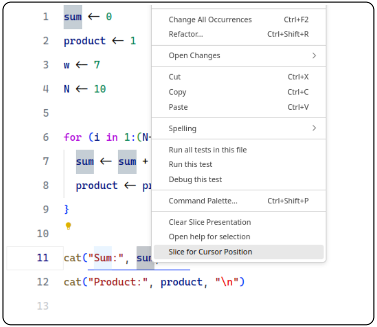


```
sum ← 0
product ← 1
w ← 7
N ← 10

for (i in 1:(N-
  sum ← sum +
  product ← pr
}

cat("Sum:", sum
cat("Product:", product, "\n")

```

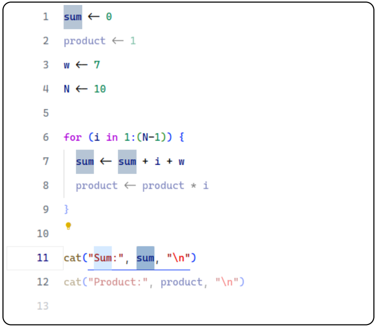
# Visual Studio Code Integration

Rudimentary VSCode integration at:

github.com/Code-Inspect/vscode-flowr



```
sum ← 0
product ← 1
w ← 7
N ← 10

for (i in 1:(N-
  sum ← sum +
  product ← pr
}

cat("Sum:", sur
cat("Product:", product, "\n")
```



```
sum ← 0
product ← 1
w ← 7
N ← 10

for (i in 1:(N-1)) {
  sum ← sum + i + w
  product ← product * i
}

cat("Sum:", sum, "\n")
cat("Product:", product, "\n")
```

# Using flowR

- github.com/Code-Inspect/flowr
- github.com/Code-Inspect/vscode-flowr
- hub.docker.com/r/eagleoutice/flowr
- npmjs.com/package/@eagleoutice/flowr

# Using flowR

github.com/Code-Inspect/flowr

github.com/Code-Inspect/vscode-flowr

hub.docker.com/r/eagleoutice/flowr

npmjs.com/package/@eagleoutice/flowr

### Server

```
{
  "type": "request-file-analysis",
  "id":        "1",
  "filetoken": "123",
  "content":   "x←1;␣x*y"
}

{
  "type": "request-slice",
  "id":        "2",
  "filetoken": "123",
  "criterion": ["1@x"]
}
```

# Using flowR

github.com/Code-Inspect/flowr

github.com/Code-Inspect/vscode-flowr

hub.docker.com/r/eagleoutice/flowr

npmjs.com/package/@eagleoutice/flowr

## Server

```
{
  "type": "request-file-analysis",
  "id":         "1",
  "filetoken": "123",
  "content":   "x←1;␣x*y"
}

{
  "type": "request-slice",
  "id":         "2",
  "filetoken": "123",
  "criterion": ["1@x"]
}
```

## Library

```
const s = new SteppingSlicer({
  shell, tokenMap,
  request:
     requestFromInput("x←1; x*y"),
  criterion: ['1@x'],
})
const slice =
   await s.allRemainingSteps()
```

# Using flowR

- github.com/Code-Inspect/flowr
- github.com/Code-Inspect/vscode-flowr
- hub.docker.com/r/eagleoutice/flowr
- npmjs.com/package/@eagleoutice/flowr

## Server

```
{
  "type": "request-file-analysis",
  "id":         "1",
  "filetoken": "123",
  "content":    "x←1;␣x*y"
}

{
  "type": "request-slice",
  "id":         "2",
  "filetoken": "123",
  "criterion": ["1@x"]
}
```

## Library

```
const s = new SteppingSlicer({
  shell, tokenMap,
  request:
     requestFromInput("x←1; x*y"),
  criterion: ['1@x'],
})
const slice =
   await s.allRemainingSteps()
```

## REPL

```
R> :parse "x←1; x*y"
exprlist
⊢ expr
|  ⊢ expr
|  |  └ SYMBOL "x" (1:1)
[...]

R> :dataflow* "x ← 1; x * y"
https://mermaid.live/edit#base64:eyJj...
```

# Using flowR

github.com/Code-Inspect/flowr

github.com/Code-Inspect/vscode-flowr

hub.docker.com/r/eagleoutice/flowr

npmjs.com/package/@eagleoutice/flowr

## Server

```
{
  "type": "request-file-analysis",
  "id":         "1",
  "filetoken": "123",
  "content":   "x←1;␣x*y"
}

{
  "type": "request-slice",
  "id":         "2",
  "filetoken": "123",
  "criterion": ["1@x"]
}
```

## Library

```
const s = new SteppingSlicer({
  shell, tokenMap,
  request:
     requestFromInput("x←1; x*y"),
  criterion: ['1@x'],
})
const slice =
   await s.allRemainingSteps()
```

## REPL

```
R> :parse "x←1; x*y"
exprlist
⊢ expr
| ⊢ expr
| |  └ SYMBOL "x" (1:1)
[...]

R> :dataflow* "x ← 1; x * y"
https://mermaid.live/edit#base64:eyJj...
```

```
docker run -it --rm eagleoutice/flowr
```

# Appendix

# The R Code Static Analysis Landscape

# The R Code Static Analysis Landscape

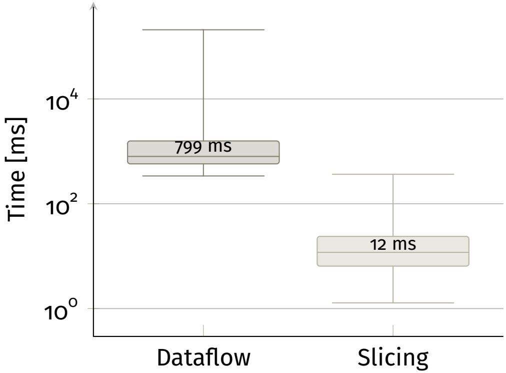
| | goal | method | impl. lang. | op. assignments | func. assignments | value trace (a.i., …) | controlflow | non-std. eval. | special operators | function calls | libraries | quotation | reflection | side effects | static scope | dynamic scope | type inference | pointer analysis | external files | pre-processors | hooks | FFI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [12] {CodeDepends} | static analysis | AST visitor | R | ● | ○ | | | ○ | ○ | ○ | ○ | | | ○ | ● | | | | | | | |
| [13] {codetools} | static analysis | AST visitor | R | ● | ○ | ○ | ○ | ○ | ○ | ○ | | ○ | | | ● | | | | | | | |
| [14] {checkglobals} | missing libs. | AST visitor | R, C | ● | ○ | | | ○ | ○ | ○ | ○ | ○ | | | ○ | | | | | | | ○ |
| ~~[15] {rstatic}~~ | ~~static analysis~~ | ~~AST visitor~~ | ~~R~~ | ● | | ○ | ● | | ○ | | | | | | ● | | | | | | | |
| [16] {CodeAnalysis} | static analysis | AST visitor | R | ● | ○ | ○ | ● | ○ | ○ | ○ | ○ | ○ | | | ● | | ○ | | ○ | | | |
| ~~[17] {RTypeInference}~~ | ~~type inference~~ | ~~AST visitor~~ | ~~R~~ | ● | | ○ | ● | | ○ | | | | | | ● | | ● | | | | | |
| [18] {pkgstats} | package insight | ctags & gtags | R, C++ | ○ | | | | | ○ | ○ | | | | | ○ | | | | | | | |
| [19] {globals} | distributed env. | AST visitor | R | ● | | ○ | ○ | ○ | ○ | ○ | | ○ | | | ● | | | | | | | |
| ~~[20] {Rclean}~~ | ~~debug/refactor~~ | ~~PDG traversal~~ | ~~R~~ | ● | ○ | | | ○ | ○ | ○ | ○ | | | | ● | | | | | | | |
| [21] {lintr} | linting | XPath, visitor | R | ● | ○ | | ○ | | ○ | | ○ | | | | ● | | | | | | | |
| [22] {SimilaR} | plagiarism | PDG, visitor | R, C++ | ○ | | | ○ | | | ○ | | | | | ● | | | | | | | |
| [23] {rco} | optimization | AST visitor | R | ● | | ● | ○ | | | | | | | | ● | | | | | | | |
| [24] {cyclocomp} | code complexity | AST visitor | R | | | | ○ | | | | | | | | ● | | | | | | | |
| [25] {flow} | visualize, debug | AST visitor | R, C | ○ | | | ● | | ○ | | ○ | ○ | | | | | | | ○ | | | |
| [26] {PaRe} | code review | Regex | R | ○ | | | ○ | | | ○ | | | | | | | | | | | | |
| ~~[27] {dfgraph}~~ | ~~static analysis~~ | ~~AST visitor~~ | ~~R~~ | ○ | | | ○ | | | ○ | | | | | | | | | | | | |
| ~~[28] {rflowgraph}~~ | ~~call graph~~ | ~~AST visitor~~ | ~~R~~ | | | | | | | ○ | ○ | | | | | | | | | | | |
| [29] {languageserver} | editor support | XPath, visitor | R, C | ● | ○ | | | | | ○ | ○ | | | | ● | | | | | | | |
| [30] RStudio | editor support | AST visitor | Java, C++, TS, … | ● | ○ | | | | | ○ | ○ | | | | ● | | | | | | | |
| ~~[31] ROSA~~ | ~~optimization~~ | ~~visitor~~ | ~~C++, R~~ | ● | | | ○ | | | ○ | | | ○ | | ● | | ● | ○ | | | | |
| ~~[32] Random~~ | ~~abstract int.~~ | ~~trace & visitor~~ | ~~R~~ | ○ | ○ | ○ | ○ | | | | ○ | | | | ○ | | | | | | | |
| [4] RaaS | reproducibility | AST visitor | Python, R | ● | ○ | | | ○ | ○ | ○ | ○ | | | | ● | | | | | | | |
| [33] GNU R | execute R | bytecode | C, Fortran, R | ● | ○ | ● | ○ | ○ | ● | ○ | | ○ | | | ● | | | | | | | |
| ~~[34] FastR~~ | ~~execute R~~ | ~~AST visitor~~ | ~~Java, C, R, …~~ | ● | | ○ | | | ○ | ○ | ○ | | | | ● | | | | | | | |
| [35] Ř | execute R | SSA, bytecode | C++, R, C, … | ● | ● | ● | ● | ○ | ● | ○ | | ○ | | | ● | ○ | ● | | | | | |
| [36] renjin | execute R | SSA, CFG | R, Java, C, … | ● | ○ | ○ | ● | | ○ | ○ | | | | ○ | ● | | ● | | | | | |
| ~~[37] pqR~~ | ~~execute R~~ | ~~bytecode~~ | ~~C, R, …~~ | ● | ○ | ● | ○ | ○ | ● | ○ | | ○ | | | ● | | | | | | | |
| ~~[38] MRO~~ | ~~execute R~~ | ~~bytecode~~ | ~~R, C, Fortran, …~~ | ● | ○ | ○ | ○ | | | | | | | | | | | | | | | |
| ~~[39] RCC~~ | ~~transpile C~~ | ~~CFG, bytecode~~ | ~~C/C++, Fortran, R, …~~ | ● | | ● | ○ | ○ | | ○ | | | | ○ | ● | | | | | | | |

# Performance Measurements

- We generated *every* possible variable of interest

# Performance Measurements

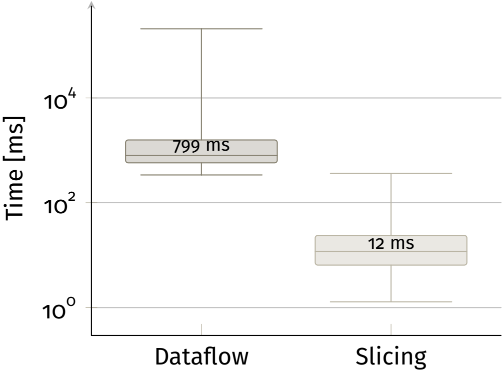
- We generated *every* possible variable of interest



99th percentile

# Performance Measurements

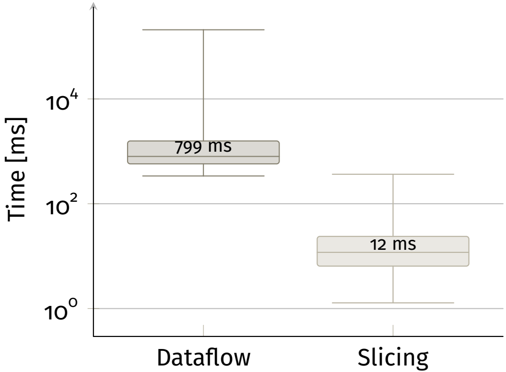
- We generated *every* possible variable of interest
- Dataflow results can be cached

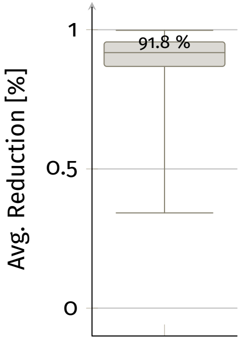


99th percentile

# Performance Measurements

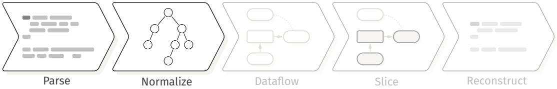
- We generated *every* possible variable of interest
- Dataflow results can be cached





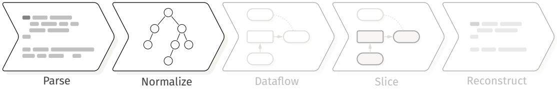
99th percentile

# Parse & Normalize

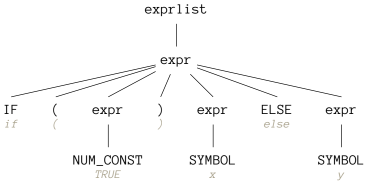


[40] R Core Team, *R Language Definition* (2023)

# Parse & Normalize

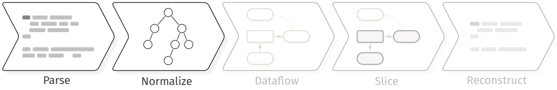


```
parse(text="if(TRUE) x else y")
```

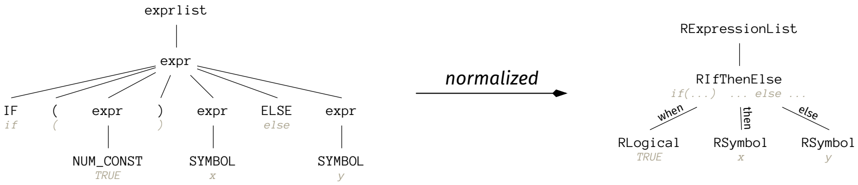


[40] R Core Team, *R Language Definition* (2023)

# Parse & Normalize

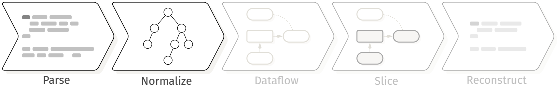


```
parse(text="if(TRUE) x else y")
```

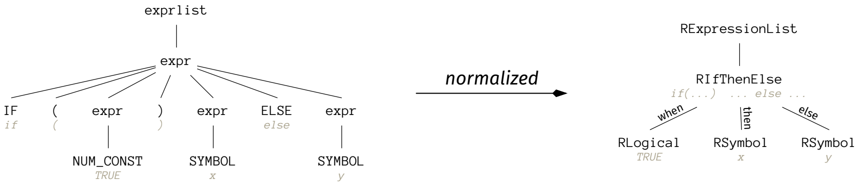


[40] R Core Team, *R Language Definition* (2023)

# Parse & Normalize



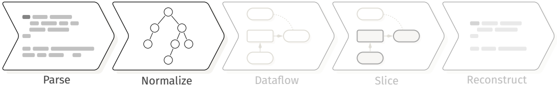
```
parse(text="if(TRUE) x else y")
```



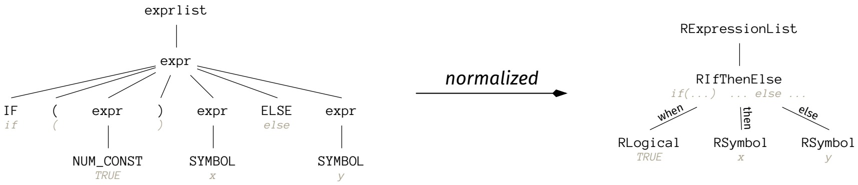
- Normalizing constants, namespacing, operators, …

[40] R Core Team, *R Language Definition* (2023)

# Parse & Normalize



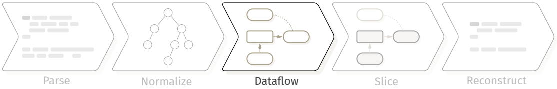
```
parse(text="if(TRUE) x else y")
```



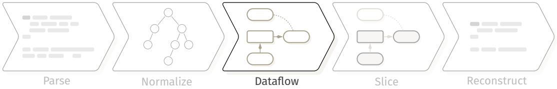
- Normalizing constants, namespacing, operators, ...
- We use the “R language definition”[40] as a basis

[40] R Core Team, *R Language Definition* (2023)

# Dataflow



# Dataflow



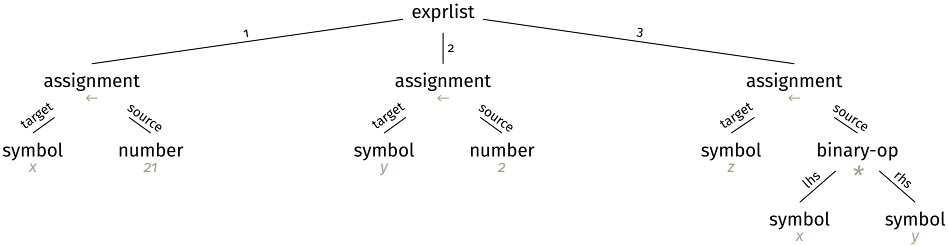
$x_0 \leftarrow 21$

$y_0 \leftarrow 2$

$z_0 \leftarrow x_1 * y_1$

# Dataflow

$x_0 \leftarrow 21$
$y_0 \leftarrow 2$
$z_0 \leftarrow x_1 * y_1$

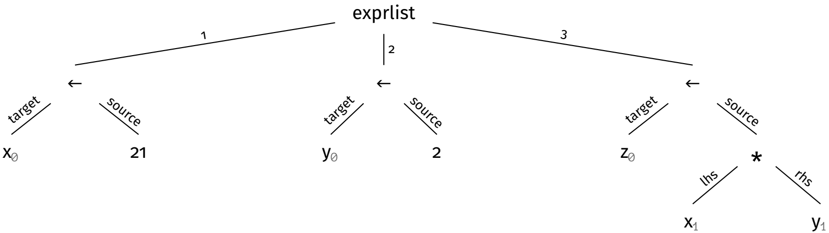


# Dataflow

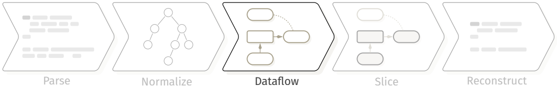
$x_{\emptyset} \leftarrow 21$

$y_{\emptyset} \leftarrow 2$

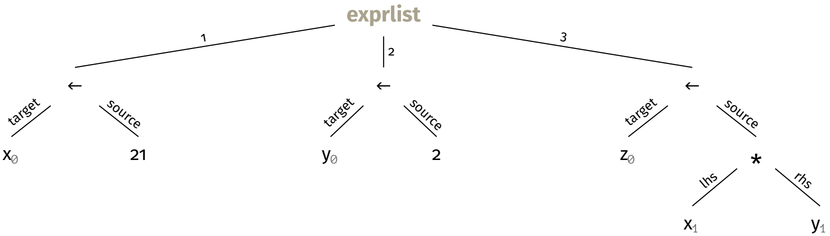
$z_{\emptyset} \leftarrow x_{1} * y_{1}$



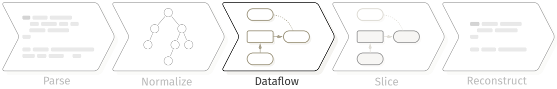
# Dataflow



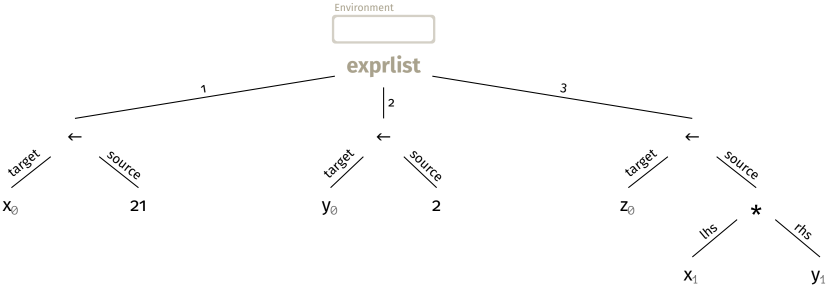
$$x_{\emptyset} \leftarrow 21$$
$$y_{\emptyset} \leftarrow 2$$
$$z_{\emptyset} \leftarrow x_{1} * y_{1}$$



# Dataflow



$x_0 \leftarrow 21$
$y_0 \leftarrow 2$
$z_0 \leftarrow x_1 * y_1$

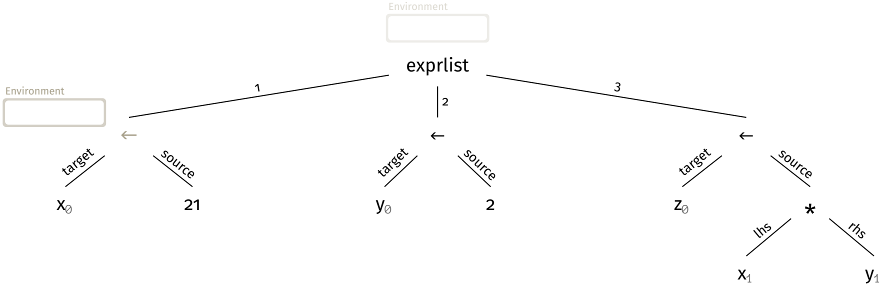


# Dataflow

$x_{\emptyset} \leftarrow 21$

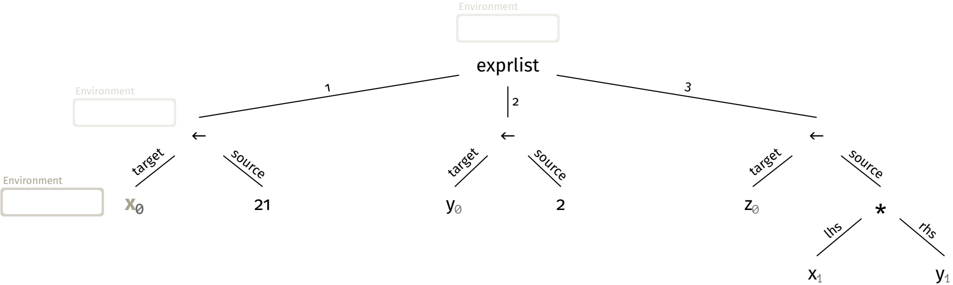
$y_{\emptyset} \leftarrow 2$

$z_{\emptyset} \leftarrow x_{1} * y_{1}$



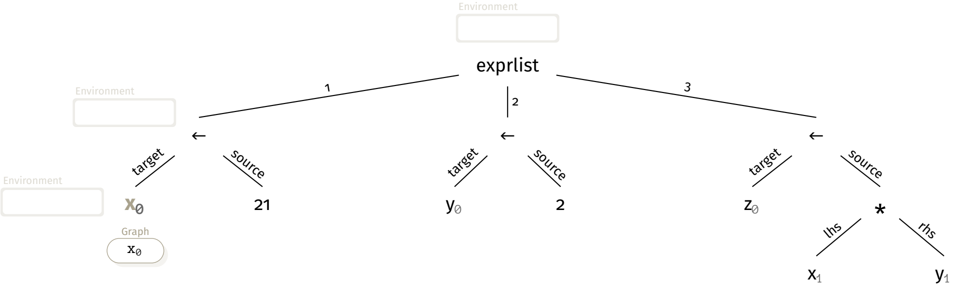
# Dataflow

$x_0 \leftarrow 21$
$y_0 \leftarrow 2$
$z_0 \leftarrow x_1 * y_1$

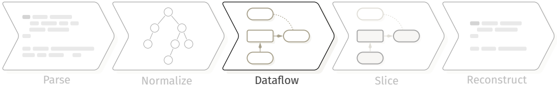


# Dataflow

$$x_0 \leftarrow 21$$
$$y_0 \leftarrow 2$$
$$z_0 \leftarrow x_1 * y_1$$



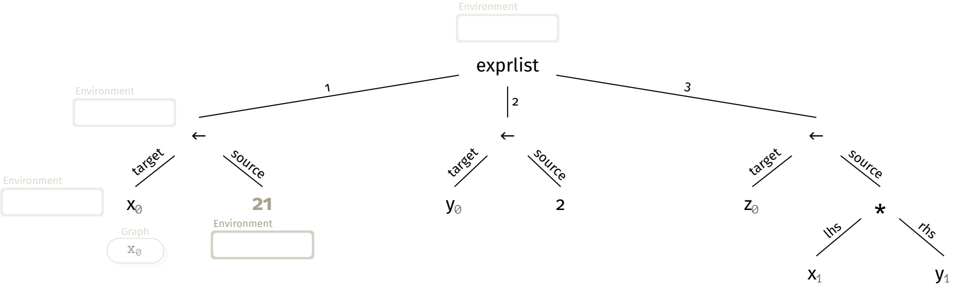
# Dataflow



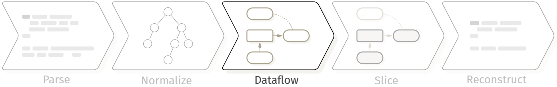
$x_0 \leftarrow 21$

$y_0 \leftarrow 2$

$z_0 \leftarrow x_1 * y_1$



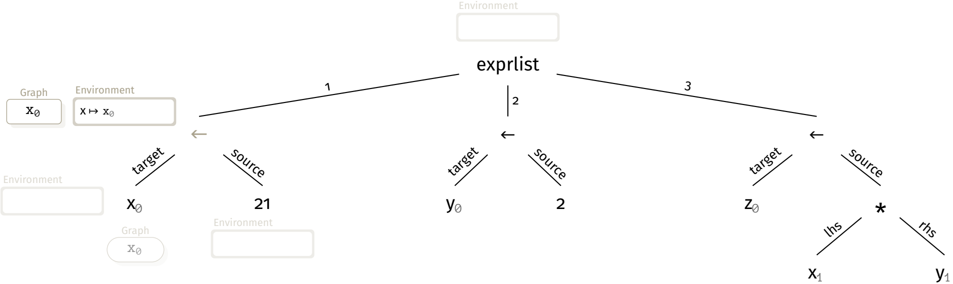
# Dataflow



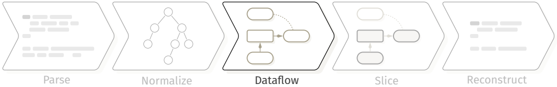
$x_0 \leftarrow 21$

$y_0 \leftarrow 2$

$z_0 \leftarrow x_1 * y_1$



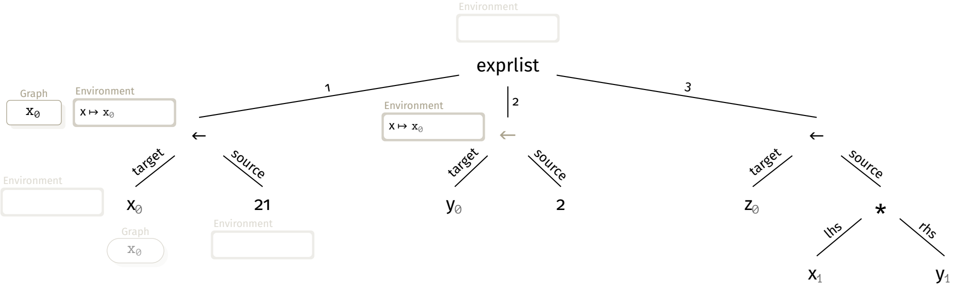
# Dataflow



$x_{\emptyset} \leftarrow 21$

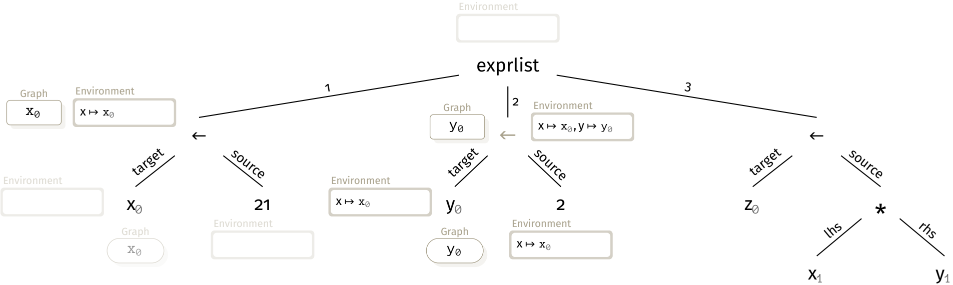
$y_{\emptyset} \leftarrow 2$

$z_{\emptyset} \leftarrow x_{1} * y_{1}$

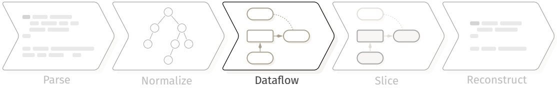


# Dataflow

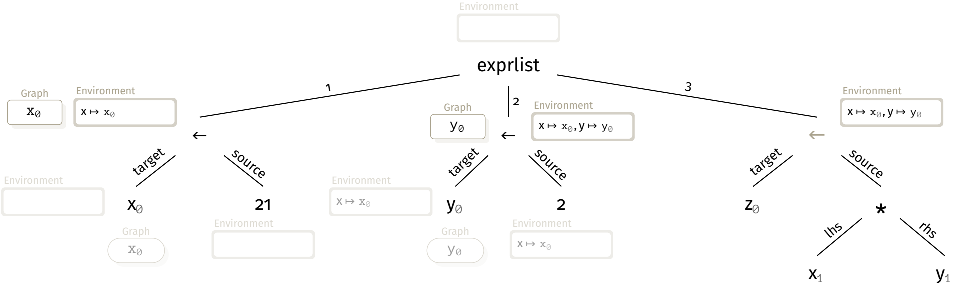
$$x_0 \leftarrow 21$$
$$y_0 \leftarrow 2$$
$$z_0 \leftarrow x_1 * y_1$$

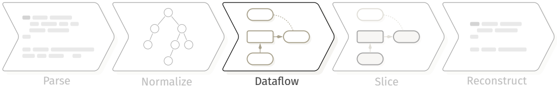


# Dataflow



$$x_{\emptyset} \leftarrow 21$$
$$y_{\emptyset} \leftarrow 2$$
$$z_{\emptyset} \leftarrow x_{1} * y_{1}$$



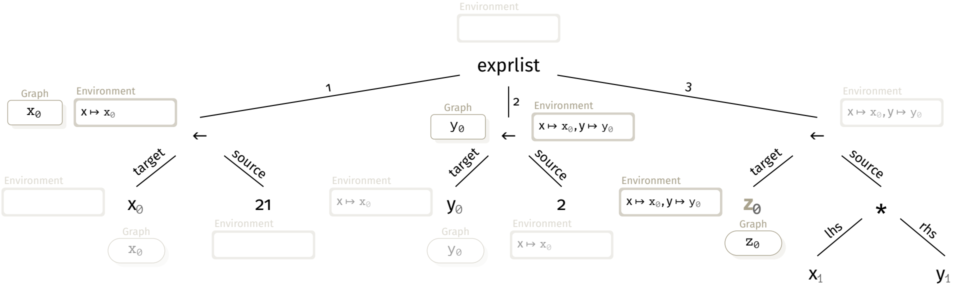


# Dataflow

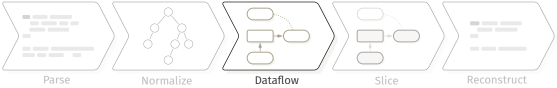
$x_{\emptyset} \leftarrow 21$

$y_{\emptyset} \leftarrow 2$

$z_{\emptyset} \leftarrow x_{1} * y_{1}$



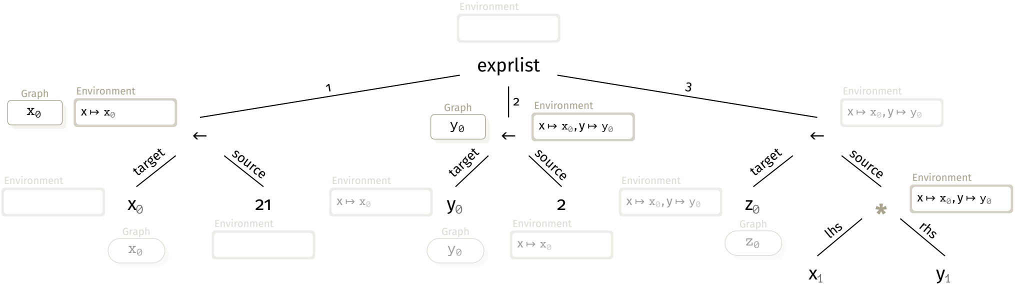
# Dataflow



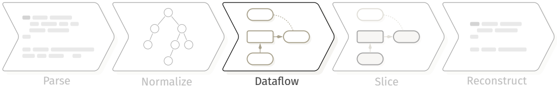
$x_{\emptyset} \leftarrow 21$

$y_{\emptyset} \leftarrow 2$

$z_{\emptyset} \leftarrow x_{1} * y_{1}$



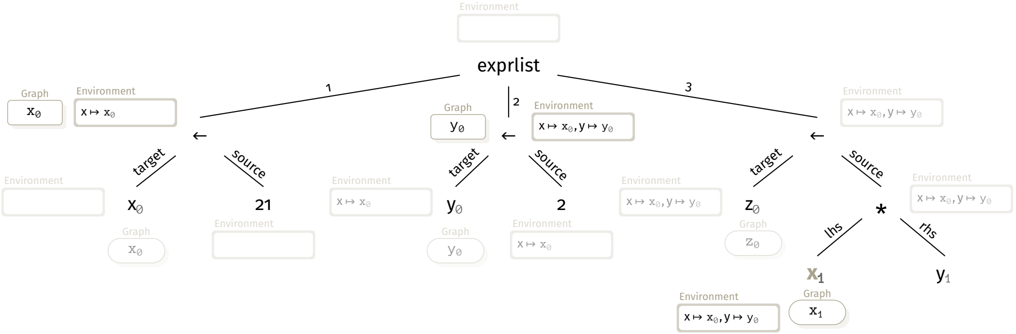
# Dataflow



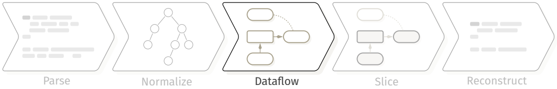
$x_\emptyset \leftarrow 21$

$y_\emptyset \leftarrow 2$

$z_\emptyset \leftarrow x_1 * y_1$



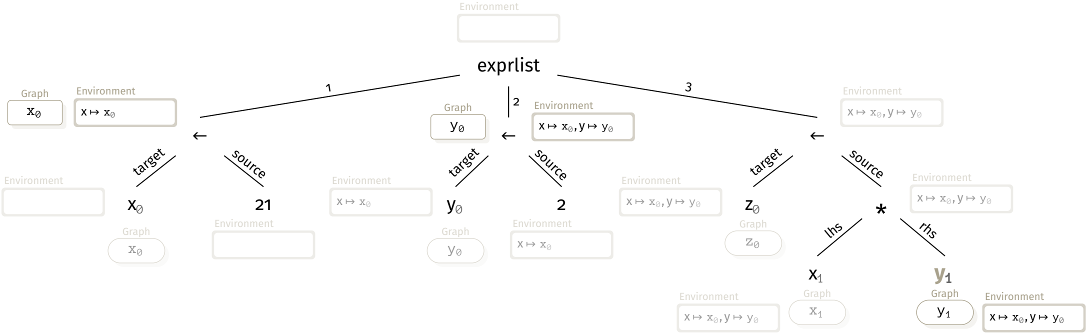
# Dataflow



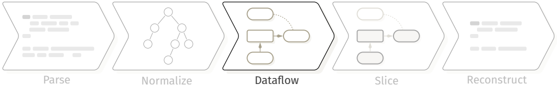
$x_0 \leftarrow 21$

$y_0 \leftarrow 2$

$z_0 \leftarrow x_1 * y_1$



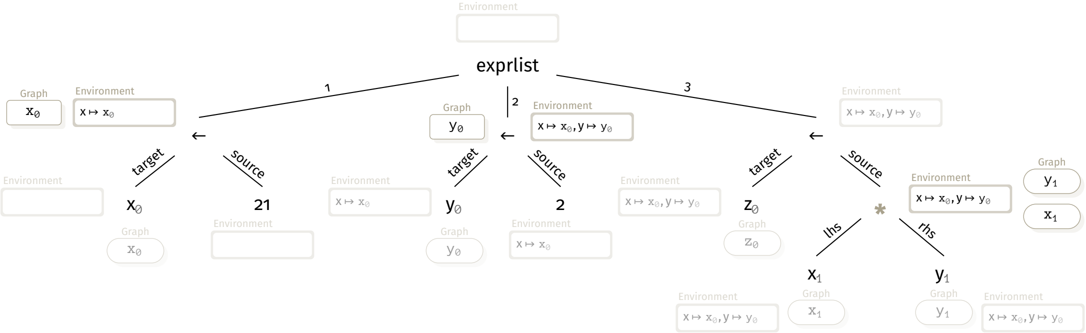
# Dataflow



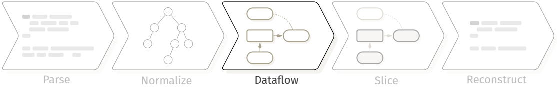
$x_{\emptyset} \leftarrow 21$

$y_{\emptyset} \leftarrow 2$

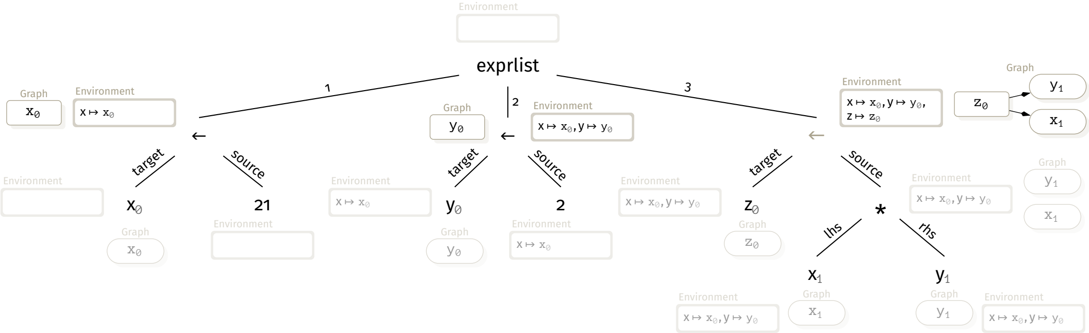
$z_{\emptyset} \leftarrow x_{1} * y_{1}$



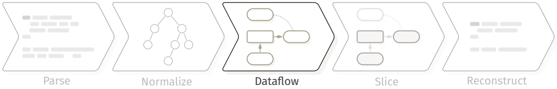
# Dataflow



$x_0 \leftarrow 21$
$y_0 \leftarrow 2$
$z_0 \leftarrow x_1 * y_1$



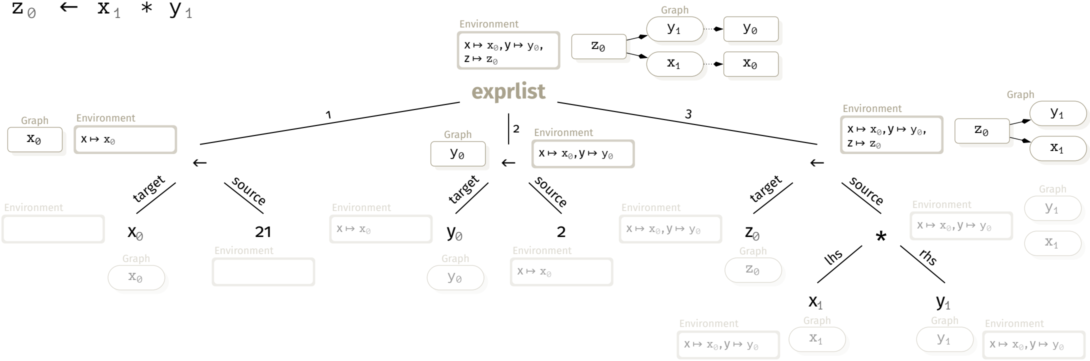
# Dataflow



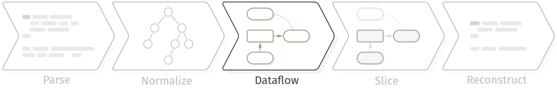
$x_0 \leftarrow 21$

$y_0 \leftarrow 2$

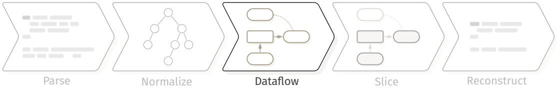
$z_0 \leftarrow x_1 * y_1$



# Resulting Dataflow

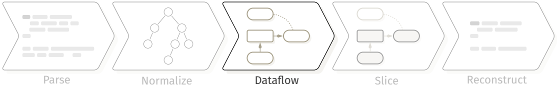


# Resulting Dataflow



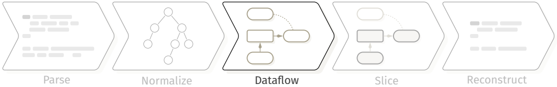
```
a ← 3
a ← x * m

if(m > 3) {
  a ← 5
}

b ← a + c
```

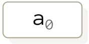
# Resulting Dataflow



```
$a_0 \leftarrow 3$
$a_1 \leftarrow x_0 * m_0$

if($m_1 > 3$) {
  $a_2 \leftarrow 5$
}

$b_0 \leftarrow a_3 + c_0$
```

# Resulting Dataflow

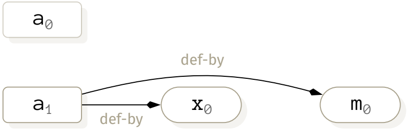




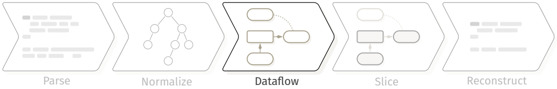
```
> a_0 ← 3
  a_1 ← x_0 * m_0

  if(m_1 > 3) {
    a_2 ← 5
  }

  b_0 ← a_3 + c_0
```

# Resulting Dataflow

```
  a_0 ← 3
> a_1 ← x_0 * m_0

  if(m_1 > 3) {
    a_2 ← 5
  }

  b_0 ← a_3 + c_0
```



# Resulting Dataflow



$a_0 \leftarrow 3$

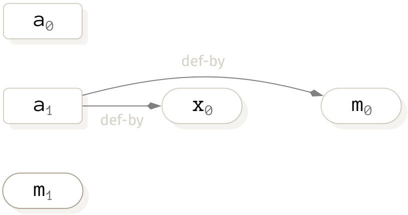
$a_1 \leftarrow x_0 * m_0$

$\rangle$ $\texttt{if}(m_1 > 3)$ {

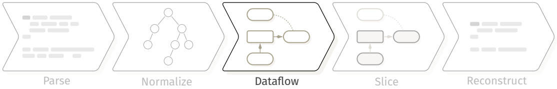
  $a_2 \leftarrow 5$

}

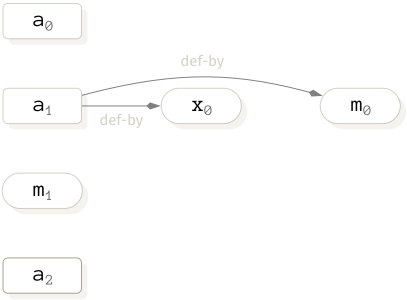
$b_0 \leftarrow a_3 + c_0$



# Resulting Dataflow

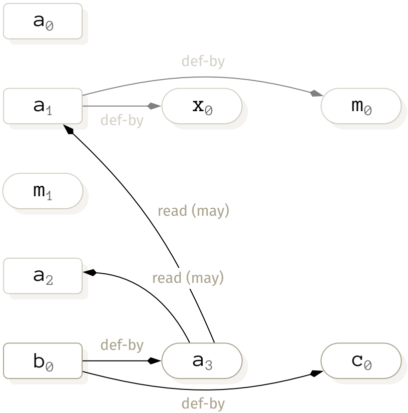


```
  a_0 ← 3
  a_1 ← x_0 * m_0

  if(m_1 > 3) {
›   a_2 ← 5
  }

  b_0 ← a_3 + c_0
```

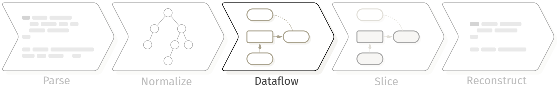


# Resulting Dataflow

```
 a_0 ← 3
 a_1 ← x_0 * m_0

 if(m_1 > 3) {
   a_2 ← 5
 }

> b_0 ← a_3 + c_0
```



# Resulting Dataflow



$a_0 \leftarrow 3$

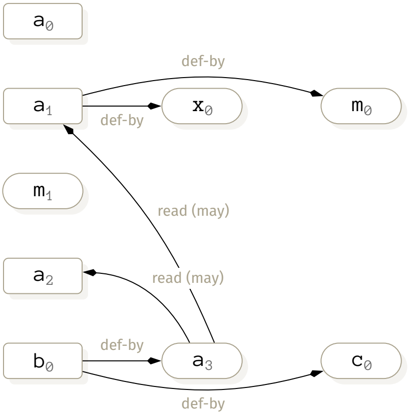
$a_1 \leftarrow x_0 * m_0$

$\texttt{if}(m_1 > 3)\ \{$

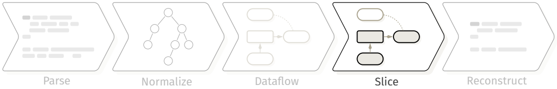
$\quad a_2 \leftarrow 5$

$\}$

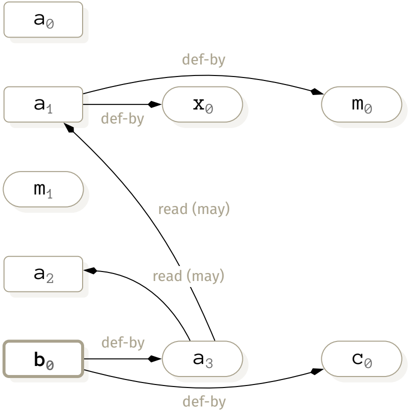
$b_0 \leftarrow a_3 + c_0$



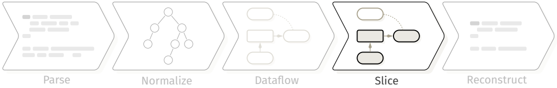
# Slicing, I



```
a_0 ← 3
a_1 ← x_0 * m_0

if(m_1 > 3) {
  a_2 ← 5
}

b_0 ← a_3 + c_0
```



# Slicing, I



$\mathtt{a}_0 \leftarrow 3$

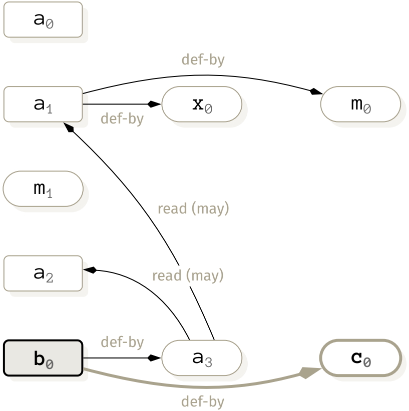
$\mathtt{a}_1 \leftarrow \mathtt{x}_0 * \mathtt{m}_0$

$\mathtt{if}(\mathtt{m}_1 > 3)\ \{$

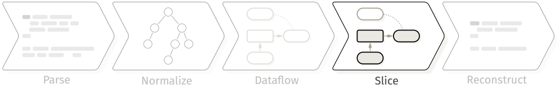
$\quad \mathtt{a}_2 \leftarrow 5$

$\}$

$\underline{\mathtt{b}}_0 \leftarrow \mathtt{a}_3 + \mathtt{c}_0$



# Slicing, I



$a_0 \leftarrow 3$

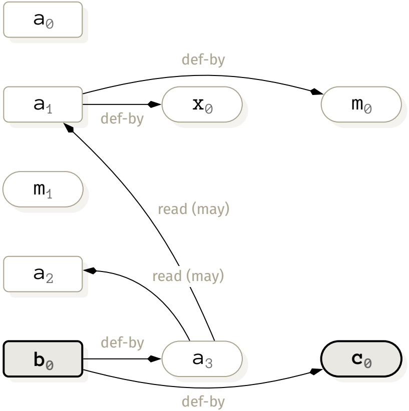
$a_1 \leftarrow x_0 * m_0$

$\mathtt{if}(m_1 > 3)\ \{$

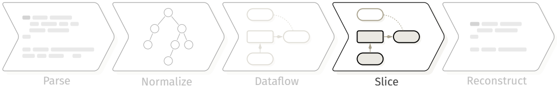
$\quad a_2 \leftarrow 5$

$\}$

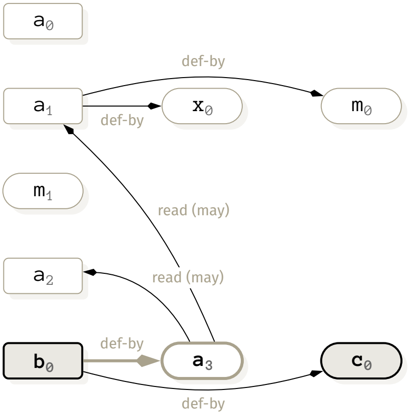
$\underline{b}_0 \leftarrow a_3 + c_0$



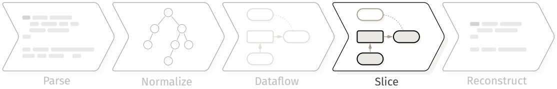
# Slicing, I



```
a_0 ← 3
a_1 ← x_0 * m_0

if(m_1 > 3) {
  a_2 ← 5
}

b_0 ← a_3 + c_0
```



# Slicing, I



$a_0 \leftarrow 3$

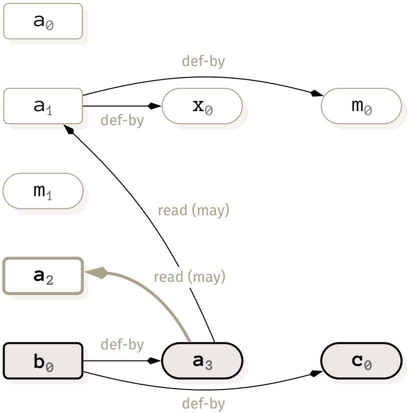
$a_1 \leftarrow x_0 * m_0$

$\texttt{if}(m_1 > 3)\ \{$

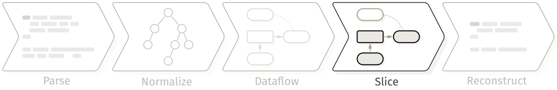
$\quad a_2 \leftarrow 5$

$\}$

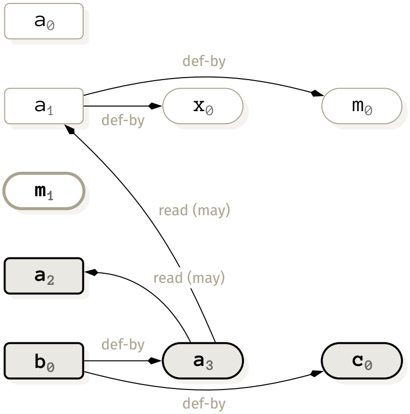
$\underline{b}_0 \leftarrow a_3 + c_0$



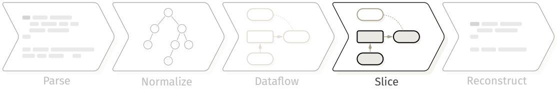
# Slicing, I



```
a_0 ← 3
a_1 ← x_0 * m_0

if(m_1 > 3) {
   a_2 ← 5
}

b_0 ← a_3 + c_0
```



# Slicing, I



$a_0 \leftarrow 3$

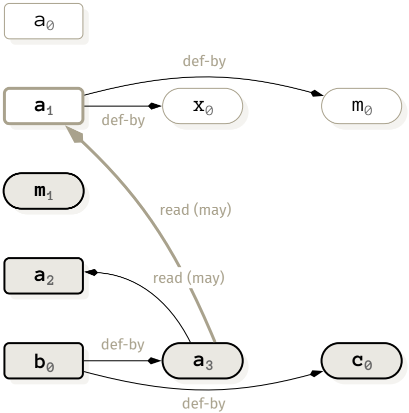
$a_1 \leftarrow x_0 * m_0$

$\text{if}(m_1 > 3)\ \{$

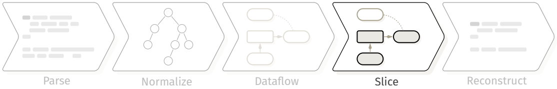
$\quad a_2 \leftarrow 5$

$\}$

$\underline{b}_0 \leftarrow a_3 + c_0$



# Slicing, I



$a_0 \leftarrow 3$

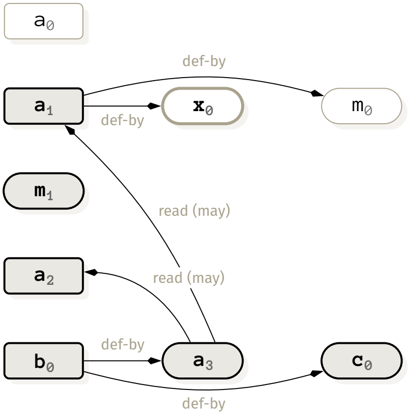
$a_1 \leftarrow x_0 * m_0$

$\texttt{if}(m_1 > 3)\ \{$

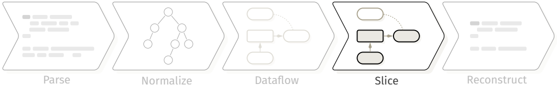
$\quad a_2 \leftarrow 5$

$\}$

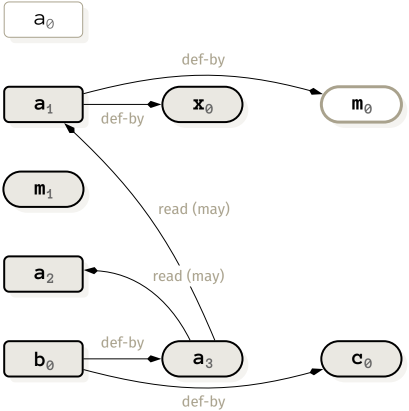
$\underline{b}_0 \leftarrow a_3 + c_0$



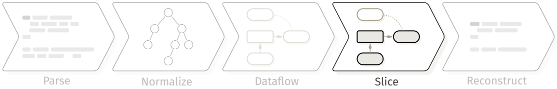
# Slicing, I



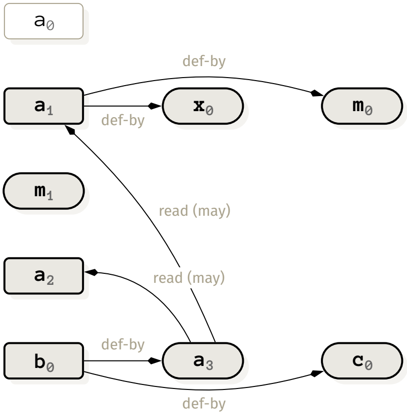
```
a_0 ← 3
a_1 ← x_0 * m_0

if(m_1 > 3) {
  a_2 ← 5
}

b_0 ← a_3 + c_0
```



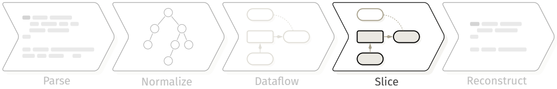
# Slicing, I



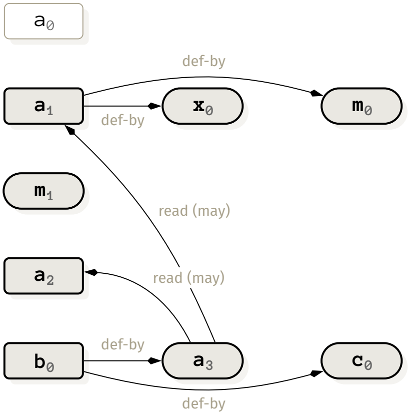
```
a_0 ← 3
a_1 ← x_0 * m_0

if(m_1 > 3) {
  a_2 ← 5
}

b_0 ← a_3 + c_0
```



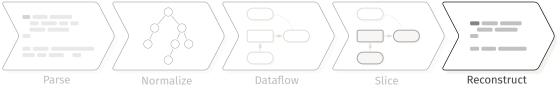
# Slicing, I



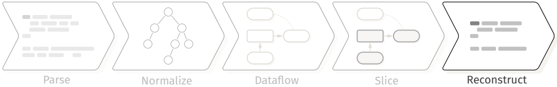
```
$a_0 \leftarrow 3$
$a_1 \leftarrow x_0 * m_0$

if($m_1$ > 3) {
  $a_2 \leftarrow 5$
}

$\underline{b}_0 \leftarrow a_3 + c_0$
```



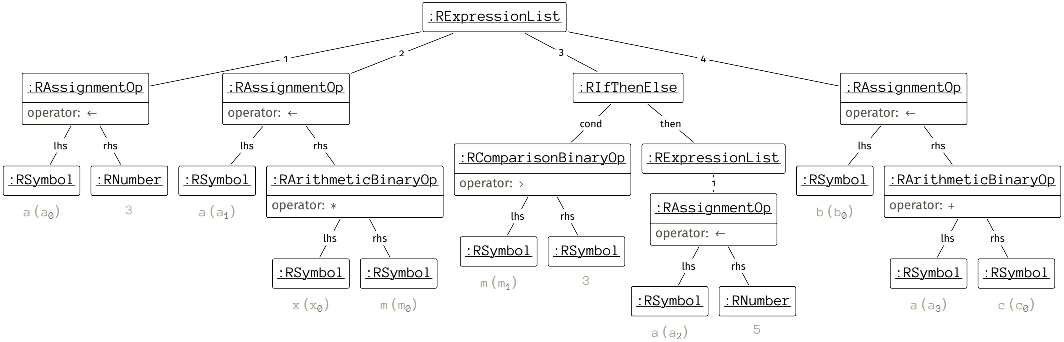
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

# Slicing, II

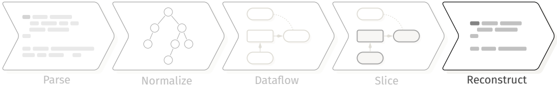


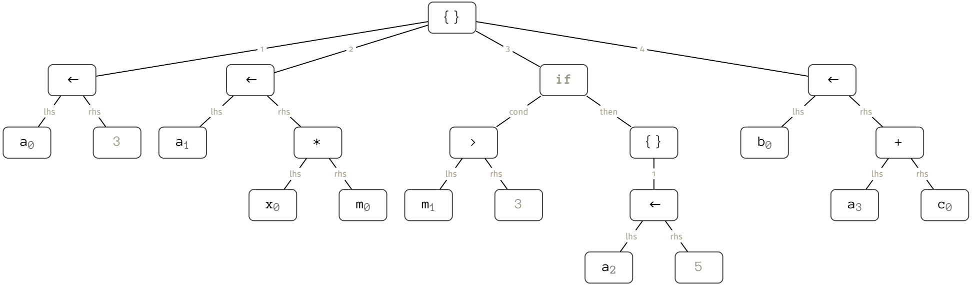
Parse
Normalize
Dataflow
Slice
Reconstruct



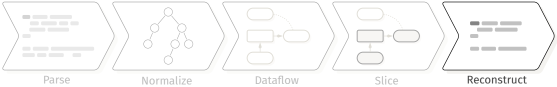
:RExpressionList
1
2
3
4
:RAssignmentOp
operator: ←
lhs
rhs
:RSymbol
a (a_0)
:RNumber
3
:RAssignmentOp
operator: ←
lhs
rhs
:RSymbol
a (a_1)
:RArithmeticBinaryOp
operator: ∗
lhs
rhs
:RSymbol
x (x_0)
:RSymbol
m (m_0)
:RIfThenElse
cond
then
:RComparisonBinaryOp
operator: >
lhs
rhs
:RSymbol
m (m_1)
:RSymbol
3
:RExpressionList
1
:RAssignmentOp
operator: ←
lhs
rhs
:RSymbol
a (a_2)
:RNumber
5
:RAssignmentOp
operator: ←
lhs
rhs
:RSymbol
b (b_0)
:RArithmeticBinaryOp
operator: +
lhs
rhs
:RSymbol
a (a_3)
:RSymbol
c (c_0)

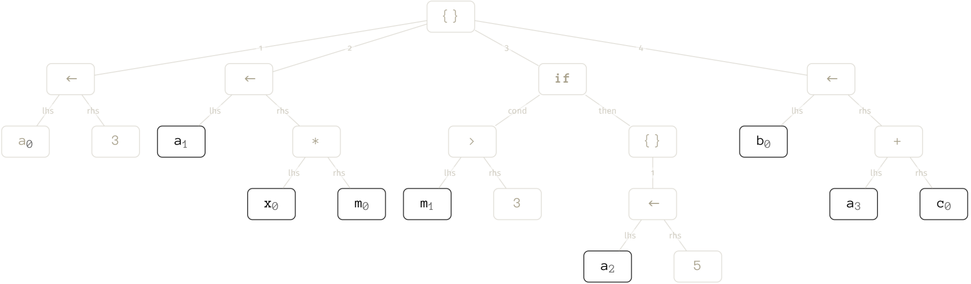
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

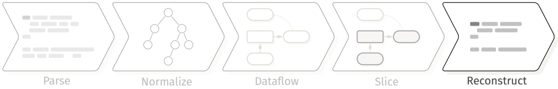

{ }
1
2
3
4
←
lhs
rhs
$a_0$
3
←
lhs
rhs
$a_1$
*
lhs
rhs
$x_0$
$m_0$
if
cond
then
>
lhs
rhs
$m_1$
3
{ }
1
←
lhs
rhs
$a_2$
5
←
lhs
rhs
$b_0$
+
lhs
rhs
$a_3$
$c_0$

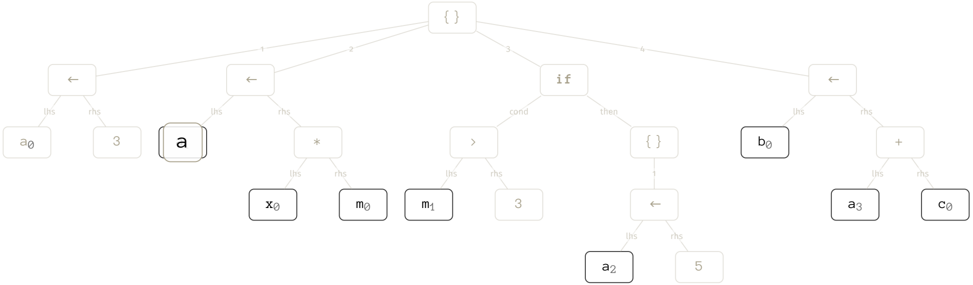
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

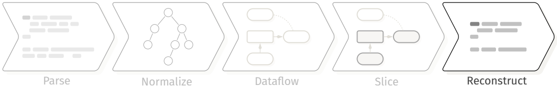

{ }
1
2
3
4
←
←
if
←
lhs
rhs
lhs
rhs
cond
then
lhs
rhs
a∅
3
a1
*
>
{ }
b∅
+
lhs
rhs
lhs
rhs
1
lhs
rhs
x∅
m∅
m1
3
←
a3
c∅
lhs
rhs
a2
5

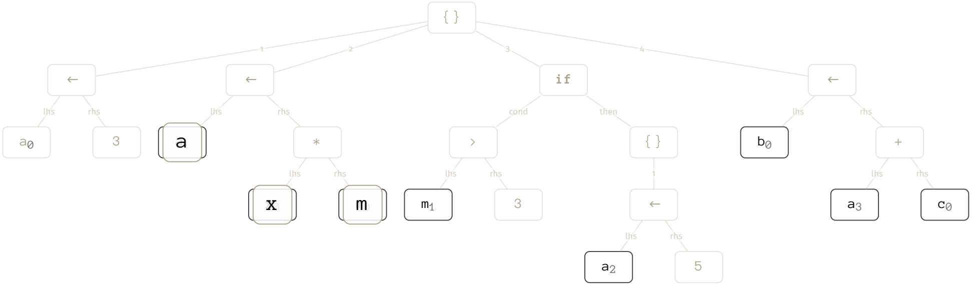
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

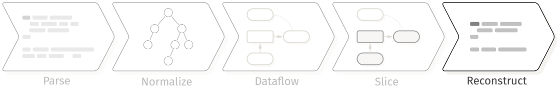

{ }
1
2
3
4
←
←
if
←
lhs
rhs
lhs
rhs
cond
then
lhs
rhs
$a_\emptyset$
3
a
*
>
{ }
$b_\emptyset$
+
lhs
rhs
lhs
rhs
1
lhs
rhs
$x_\emptyset$
$m_\emptyset$
$m_1$
3
←
$a_3$
$c_\emptyset$
lhs
rhs
$a_2$
5

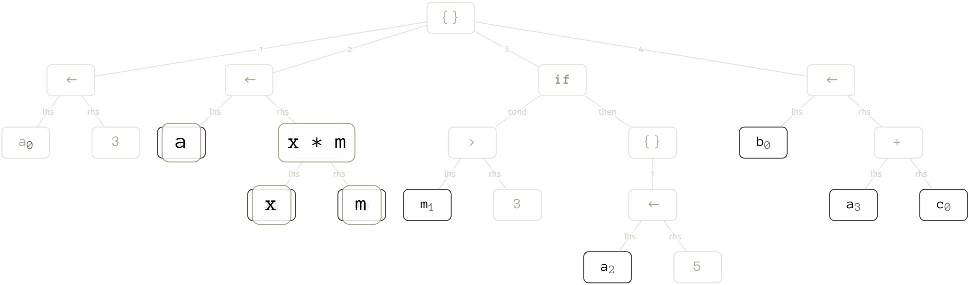
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

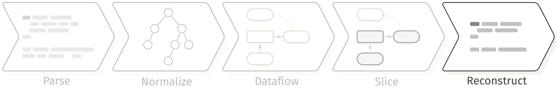

{ }
1
2
3
4
←
←
if
←
lhs
rhs
lhs
rhs
cond
then
lhs
rhs
$a_\emptyset$
3
a
*
>
{ }
$b_\emptyset$
+
lhs
rhs
lhs
rhs
1
lhs
rhs
x
m
$m_1$
3
←
$a_3$
$c_\emptyset$
lhs
rhs
$a_2$
5

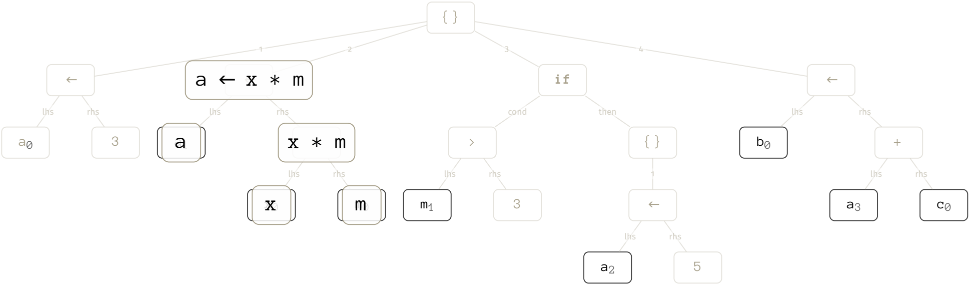
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct

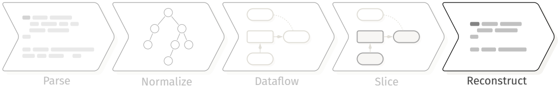

{ }
1
2
3
4
←
←
if
←
lhs
rhs
a∅
3
a
x * m
x
m
cond
then
>
m1
3
{ }
1
←
a2
5
b∅
+
a3
c∅

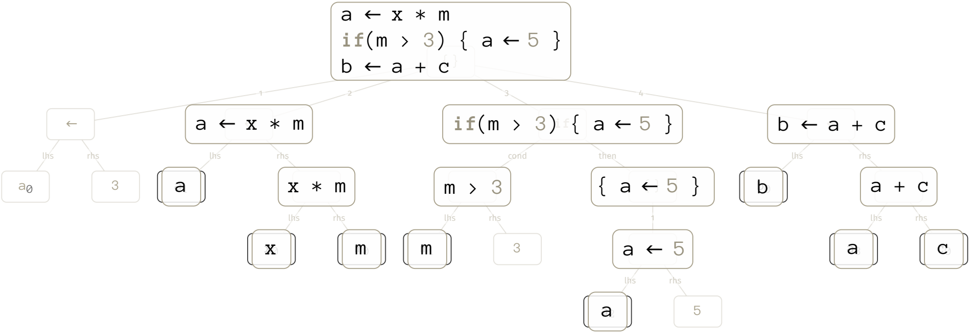
# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct


{ }
1
2
3
4
←
a ← x * m
if
←
lhs
rhs
a∅
3
a
x * m
cond
then
>
{ }
b∅
+
x
m
m1
3
←
a3
c∅
a2
5

# Slicing, II


Parse
Normalize
Dataflow
Slice
Reconstruct


a ← x * m
if(m > 3) { a ← 5 }
b ← a + c
1
2
3
4
←
lhs
rhs
a₀
3
a ← x * m
lhs
rhs
a
x * m
lhs
rhs
x
m
if(m > 3) { a ← 5 }
cond
then
m > 3
lhs
rhs
m
3
{ a ← 5 }
1
a ← 5
lhs
rhs
a
5
b ← a + c
lhs
rhs
b
a + c
lhs
rhs
a
c

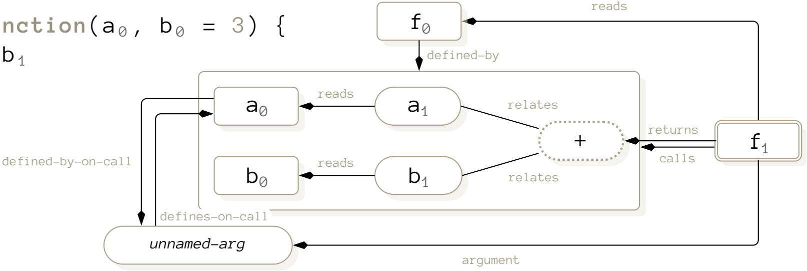
# There Is More…

# There Is More…

```
f_0 ← function(a_0, b_0 = 3) {
   a_1 + b_1
}
f_1(39)
```

# There Is More…

```
f_0 ← function(a_0, b_0 = 3) {
    a_1 + b_1
}
f_1(39)
```



# Definition-Retrieval

```
paste(
   "(*|descendant-or-self::exprlist/*)[self::FUNCTION or self::OP-LAMBDA]/
       following-sibling::SYMBOL_FORMALS[text() = '{token_quote}' and @line1 <= {
       row}]",
   "(*|descendant-or-self::exprlist/*)[LEFT_ASSIGN[preceding-sibling::expr[count
       (*)=1]/SYMBOL[text() = '{token_quote}' and @line1 <= {row}] and following-
       sibling::expr[@start > {start} or @end < {end}]]]",
   "(*|descendant-or-self::exprlist/*)[RIGHT_ASSIGN[following-sibling::expr[count
       (*)=1]/SYMBOL[text() = '{token_quote}' and @line1 <= {row}] and preceding-
       sibling::expr[@start > {start} or @end < {end}]]]",
   "(*|descendant-or-self::exprlist/*)[EQ_ASSIGN[preceding-sibling::expr[count(*)=
       1]/SYMBOL[text() = '{token_quote}' and @line1 <= {row}] and following-
       sibling::expr[@start > {start} or @end < {end}]]]",
   "forcond/SYMBOL[text() = '{token_quote}' and @line1 <= {row}]",
sep = "|")
```

# References I

[1] *The Comprehensive R Archive Network — cran.r-project.org.* 2024

[2] Ana Trisovic et al. “A Large-Scale Study on Research Code Quality and Execution”. 2022

[3] *PYPL — PopularitY of Programming Language index.* 2024

[4] Joseph Wonsil et al. “Reproducibility as a Service”. 2023

[5] Harold Thimbleby. “Improving Science That Uses Code”. 2023

[6] Inese Drudze et al. *Apple phenology data set and R script, related to publication "Full flowering phenology of apple tree (Malus domestica) in Pūre orchard, Latvia from 1959 to 2019".* June 2021

[7] Florian Sihler et al. “On the Anatomy of Real-World R Code for Static Analysis”. 2024

[8] Liang Ma et al. *Predicting range shifts of pikas (Mammalia, Ochotonidae) in China under scenarios incorporating land-use change, climate change, and dispersal limitations.* Aug. 2021

[9] Joshua Robertson. *Social hierarchy reveals thermoregulatory trade- offs in response to repeated stressors.* Oct. 2020

[10] Florian Sihler. “Constructing a static program slicer for R programs”. 2023

# References II

[11] Mark Weiser. “Program Slicing”. July 1984

[12] Duncan Lang et al. *CodeDepends. Analysis of R Code for Reproducible Research and Code Comprehension.* 2018

[13] Luke Tierney. *codetools: Code Analysis Tools for R.* 2023

[14] Joris Chau. *checkglobals: Static Analysis of R-Code Dependencies.* 2023

[15] Nick Ulle and Duncan Temple Lang. *rstatic: Low-level Static Analysis Tools for R Code.* 2019

[16] Duncan Lang et al. *CodeAnalysis. Tools for static analysis of R code.* 2023

[17] Nick Ulle and Duncan Temple Lang. *RTypeInference: Infer Types of Inputs and Outputs for R Expressions.* 2021

[18] Mark Padgham. *pkgstats.* 2021

[19] Henrik Bengtsson. *globals: Identify Global Objects in R Expressions.* 2022

[20] Matthew Lau. *Rclean: A Tool for Writing Cleaner, More Transparent Code.* 2022

[21] Jim Hester et al. *lintr: A ’Linter’ for R Code.* 2023

[22] Maciej Bartoszuk and Marek Gagolewski. *SimilaR: R Source Code Similarity Evaluation.* 2020

# References III

[23] Juan Cruz Rodriguez. *rco: The R Code Optimizer.* 2021

[24] Gabor Csardi. *cyclocomp: Cyclomatic Complexity of R Code.* 2023

[25] Antoine Fabri. *flow: View and Browse Code Using Flow Diagrams.* 2023

[26] Maarten van Kessel. *PaRe: A Way to Perform Code Review or QA on Other Packages.* 2023

[27] Dan Kary. *dfgraph: Visualize R Code with Data Flow Graphs.*

[28] Evan Patterson. *The algebra and machine representation of statistical models.* 2020

[29] Randy Lai. *languageserver: Language Server Protocol.* 2023

[30] Posit team. *RStudio: Integrated Development Environment for R.* 2023

[31] Rathijit Sen et al. *ROSA: R Optimizations with Static Analysis.* 2017

[32] Gianluca Amato and Francesca Scozzari. "Random: R-Based Analyzer for Numerical Domains". 2012

[33] R Core Team. *R: A Language and Environment for Statistical Computing.* 2023

[34] Tomas Kalibera et al. "A fast abstract syntax tree interpreter for R". 2014

[35] Olivier Flückiger et al. "Sampling optimized code for type feedback". 2020

# References IV

[36] Alexander Bertram. “Renjin: A new r interpreter built on the jvm”. 2013

[37] Radford M Neal. “Speed Improvements in pqR: Current Status and Future Plans”. 2014

[38] *Microsoft R Open Source.* 2019

[39] John Garvin. *RCC: A compiler for the R language for statistical computing.* 2004

[40] R Core Team. *R Language Definition.* 2023