

# Cut to the Core



## Automated Feature Extraction in R Using Program Slicing

FOSD '24 | Ulm University | **Ruben Dunkel**, Florian Sihler, Thomas Thüm and Matthias Tichy | April, 2024

## Real-World R Code

[1] 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)

# Real-World R Code



- Very long and complex
- Partial data availability (reproducibility problem)
- Take long to run

[1] 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)

# Real-World R Code

1 } Load

2 } Model

3 } Model

4 } Figure

5 } Figure

6 } Figure

- Very long and complex
- Partial data availability (reproducibility problem)
- Take long to run

[1] 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)

# Real-World R Code

1 } Load

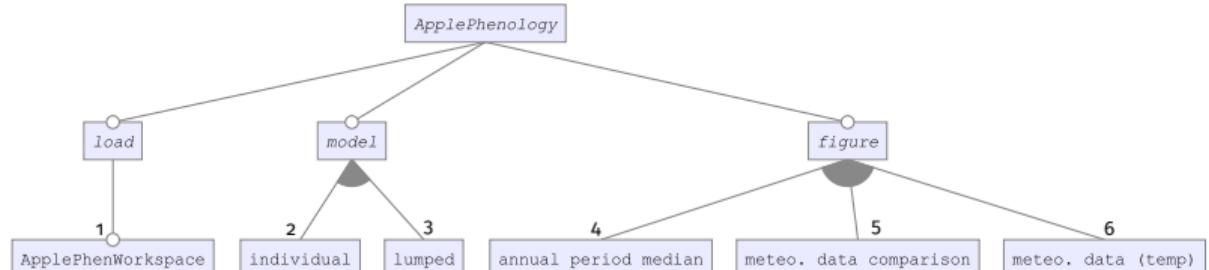
2 } Model

3 } Model

4 } Figure

5 } Figure

6 } Figure



Made with ❤ and variability.dev

[1] 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 Plan

# The Plan

```
sum  ← 0
prod ← 1
n    ← 10

for (i in 1:(n-1)) {
  prod ← prod * i
  sum  ← sum + i
}

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

# The Plan

```
sum  ← 0
prod ← 1
n    ← 10

for (i in 1:(n-1)) {
  prod ← prod * i
  sum  ← sum + 2
}

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

# The Plan

```
sum  ← 0
prod ← 1
n    ← 10

for (i in 1:(n-1)) {
  prod ← prod * i
  sum  ← sum + 2
}

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



Dataflow-  
Analysis

# The Plan

```
sum  ← 0
prod ← 1
n ← 10

for (i in 1:(n-1)) {
    prod ← prod * i
    sum  ← sum + 2
}

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

(simplified dataflow)

Dataflow-  
Analysis

# The Plan

```
sum ← 0
prod ← 1
n ← 13

for (i ← 1 to n-1) {
    prod ← prod * i
    sum ← sum + 2
}

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

(simplified dataflow)

Dataflow-  
Analysis

# The Plan

```
sum ← 0
prod ← 1
n ← 13

for (i in 1..(n-1)) {
    prod ← prod * i
    sum ← sum + 2
}

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

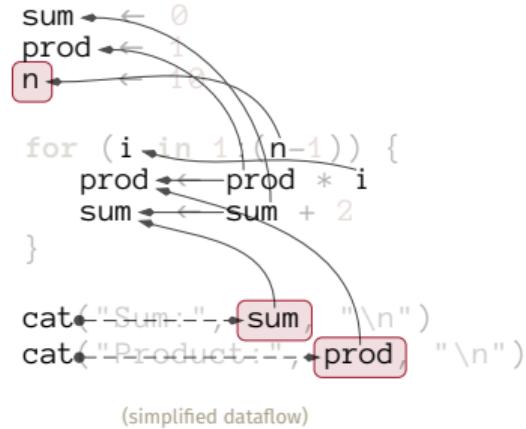
(simplified dataflow)

The diagram illustrates the dataflow for the variables sum and prod. It shows the initial values: sum is set to 0 and prod is set to 1. Inside the loop, prod is updated by multiplying its current value by i, and sum is updated by adding 2 to its current value. Finally, both sum and prod are outputted using the cat function.

Dataflow-  
Analysis

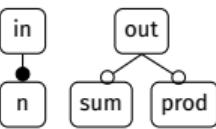
Collect in-  
and outputs

# The Plan

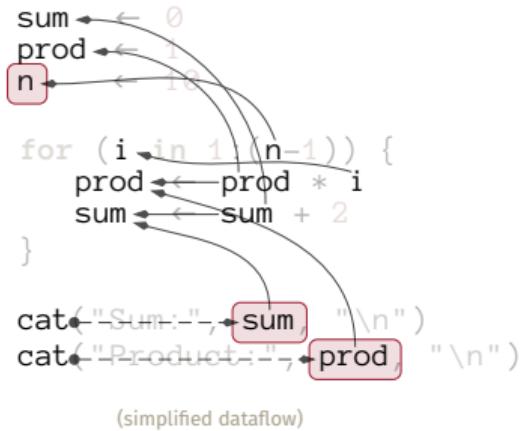


Dataflow-  
Analysis

Collect in-  
and outputs

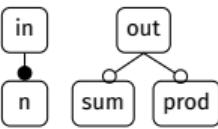


# The Plan

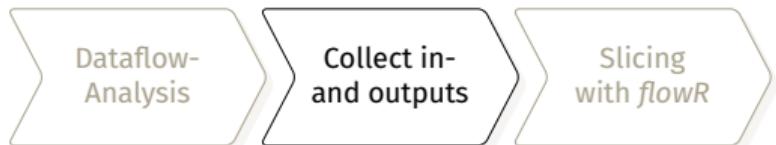
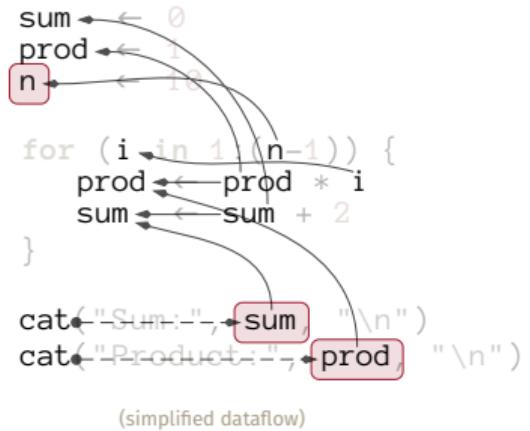


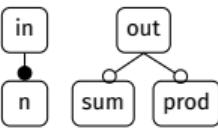
Dataflow-Analysis

Collect in-  
and outputs

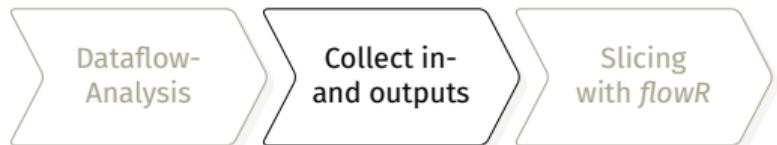
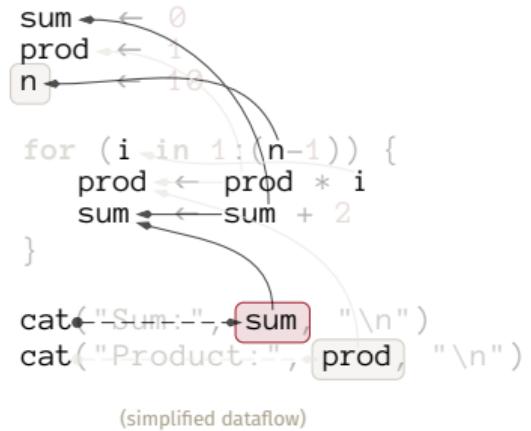


# The Plan

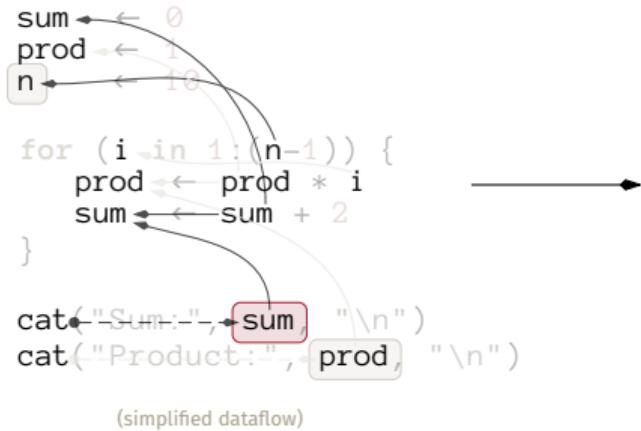




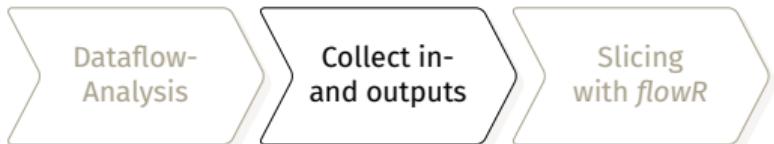
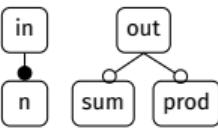
# The Plan



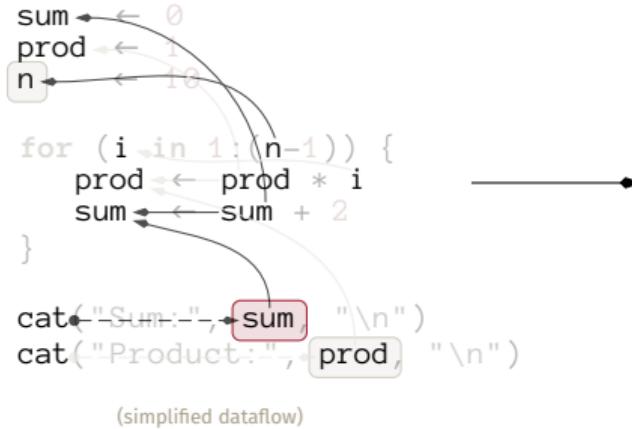
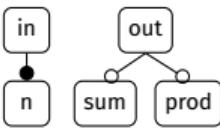
# The Plan



```
sum ← 0  
prod ← 1  
n ← 10  
  
for (i in 1:(n-1)) {  
    prod ← prod * i  
    sum ← sum + 2  
}  
  
cat("Sum:", sum, "\n")  
cat("Product:", prod, "\n")
```



# The Plan



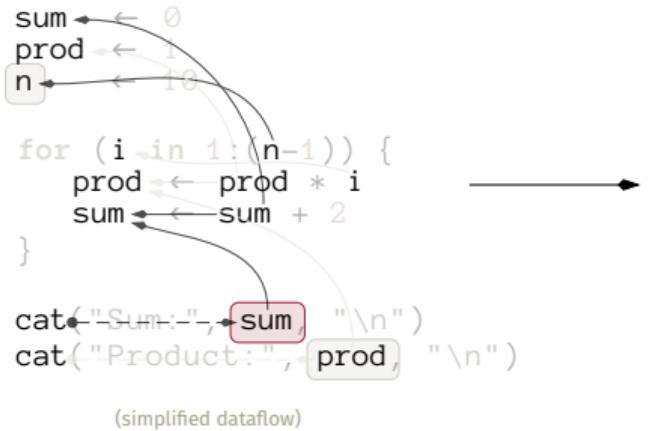
```
sum ← 0
prod ← 1
n ← 10

for (i in 1:(n-1)) {
    prod ← prod * i
    sum ← sum + 2
}

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



# The Plan



```
sum ← 0  
prod ← 1  
n ← 10
```

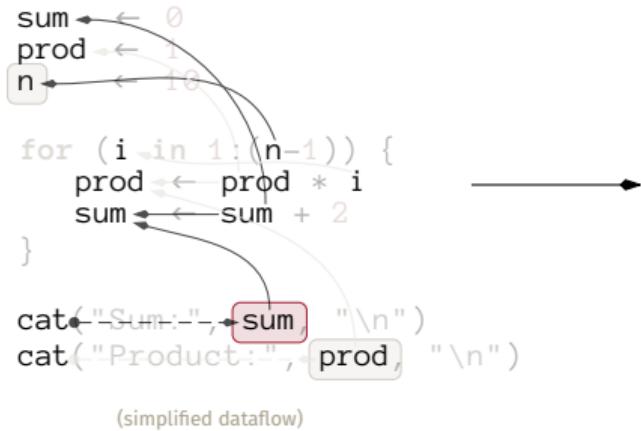
```
for (i in 1:(n-1)) {  
    prod ← prod * i  
    sum ← sum + 2  
}  
  
cat("Sum:", sum, "\n")  
cat("Product:", prod, "\n")
```

(sum)  
(prod)  
(core)  
(core)  
(prod)  
(sum)  
(core)  
(sum)  
(prod)

(sum)  
(prod)  
(core)  
(core)  
(prod)  
(sum)  
(core)  
(sum)  
(prod)



# The Plan



```
sum ← 0
prod ← 1
n ← 10

for (i in 1:(n-1)) {
    prod ← prod * i
    sum ← sum + 2
}

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

(sum)  
(prod)  
(core)

(core)  
(prod)  
(sum)  
(core)

(sum)  
(prod)

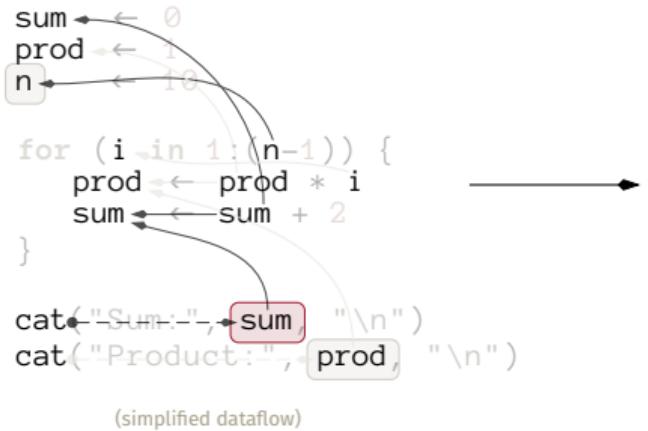
(sum)  
(prod)  
(core)

(core)  
(prod)  
(sum)  
(core)

(sum)  
(prod)



# The Plan



```
sum ← 0  
prod ← 1  
n ← 10
```

(sum)  
(prod)  
(core)

```
for (i in 1:(n-1)) {  
    prod ← prod * i  
    sum ← sum + 2  
}
```

(core)  
(prod)  
(sum)  
(core)

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

(sum)  
(prod)



# Open Questions



1. Who already worked with R?
2. What is your experience with code-extraction?
3. What is good/important related work?

# Open Questions

Dataflow-  
Analysis

Collect in-  
and outputs

Slicing  
with *flowR*

Slices to  
Proposed  
Features

Allow  
Manual  
Adjustments

Add  
Preprocessor  
Directives

1. Who already worked with R?
2. What is your experience with code-extraction?
3. What is good/important related work?



[ruben.dunkel@uni-ulm.de](mailto:ruben.dunkel@uni-ulm.de)

## References

- [1] 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