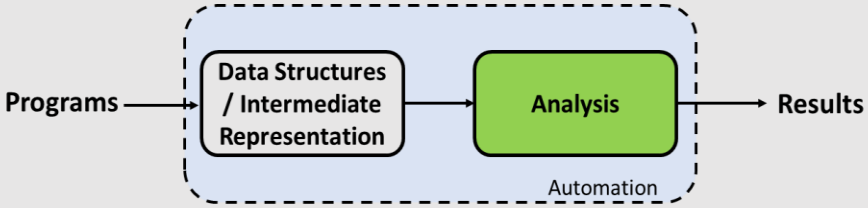
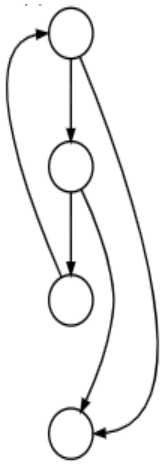
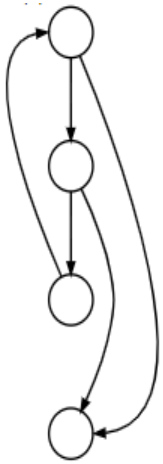




Program Analysis

Venkatesh Vinayakarao

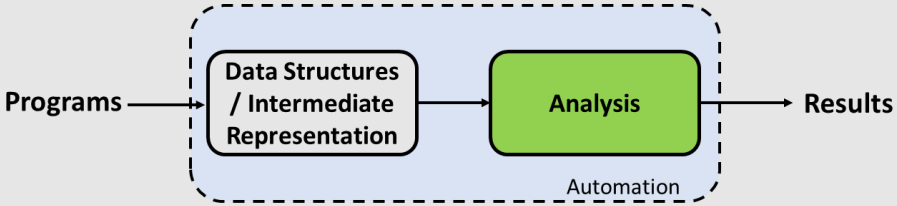
venkateshv@cmi.ac.in
Mar – Apr, 2018
Chennai Mathematical Institute



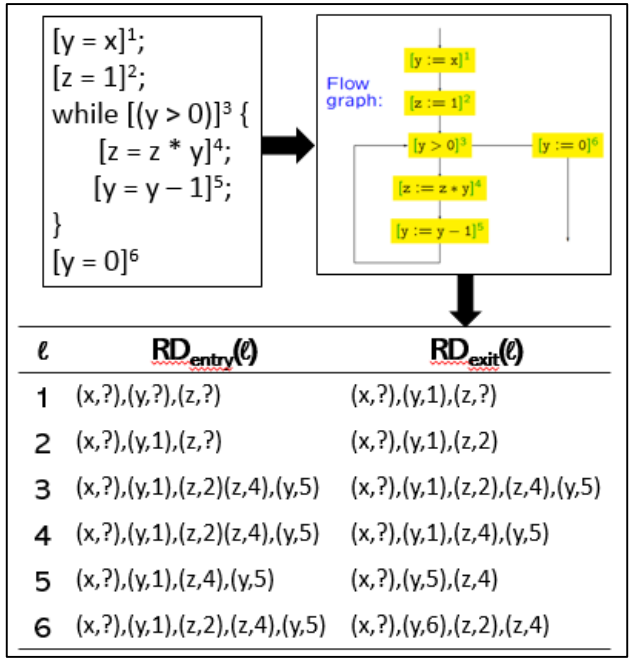
"The supply of grand challenges … shows little sign of drying up."

– Harman and O'Hearn in "Opportunities and Open Problems for Static and Dynamic Program Analysis", Madrid, Spain, 2018.

Quick Review

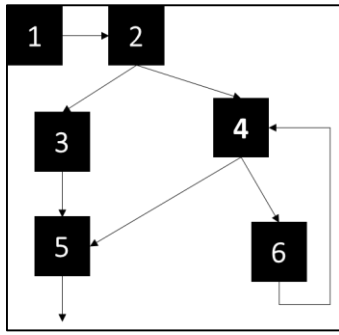


Types
- Static
- Dynamic
- Hybrid



ℓ	$RD_{entry}(\ell)$	$RD_{exit}(\ell)$
1	(x,?),(y,?),(z,?)	(x,?),(y,1),(z,?)
2	(x,?),(y,1),(z,?)	(x,?),(y,1),(z,2)
3	(x,?),(y,1),(z,2)(z,4),(y,5)	(x,?),(y,1),(z,2),(z,4),(y,5)
4	(x,?),(y,1),(z,2)(z,4),(y,5)	(x,?),(y,1),(z,4),(y,5)
5	(x,?),(y,1),(z,4),(y,5)	(x,?),(y,5),(z,4)
6	(x,?),(y,1),(z,2),(z,4),(y,5)	(x,?),(y,6),(z,2),(z,4)

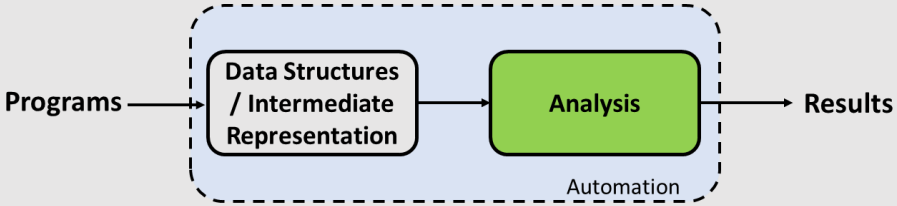
Static Analysis

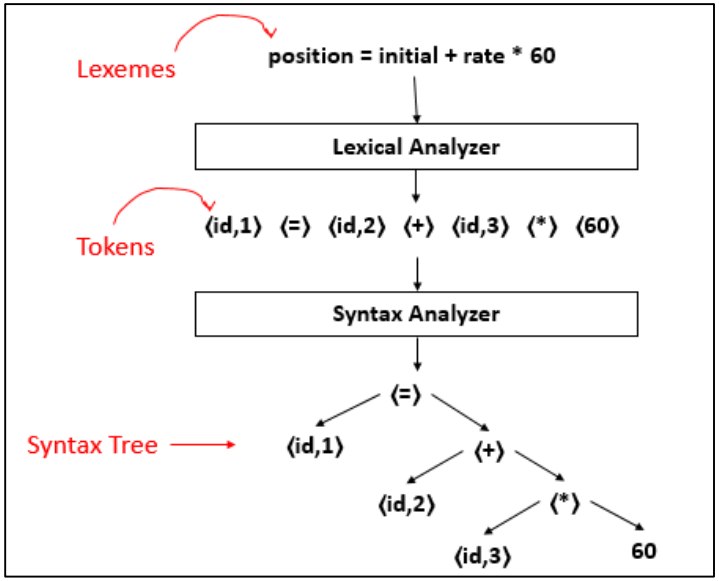


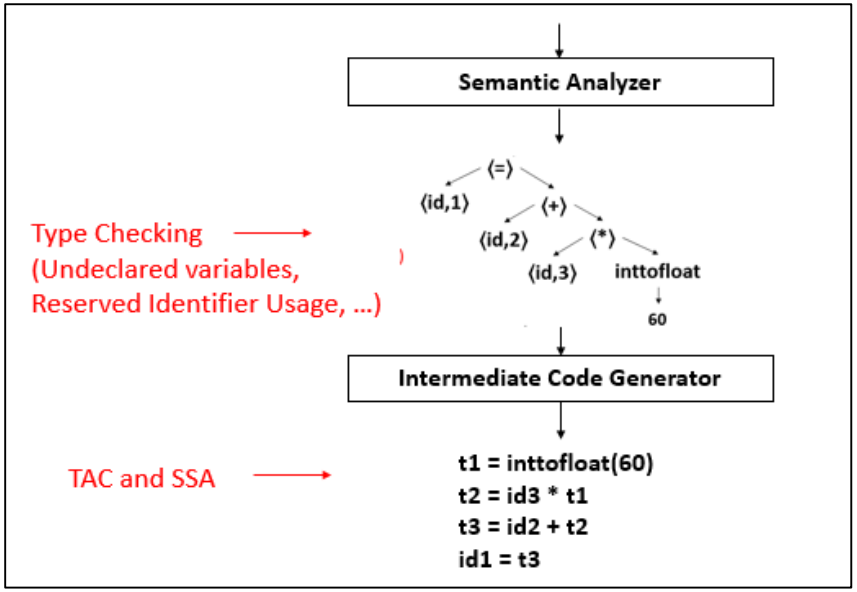
Path	Frequency
1 2 3 5	100
1 2 4 6 4 5	2000
1 2 4 6 4 6 4 5	10

Dynamic Analysis

Quick Review





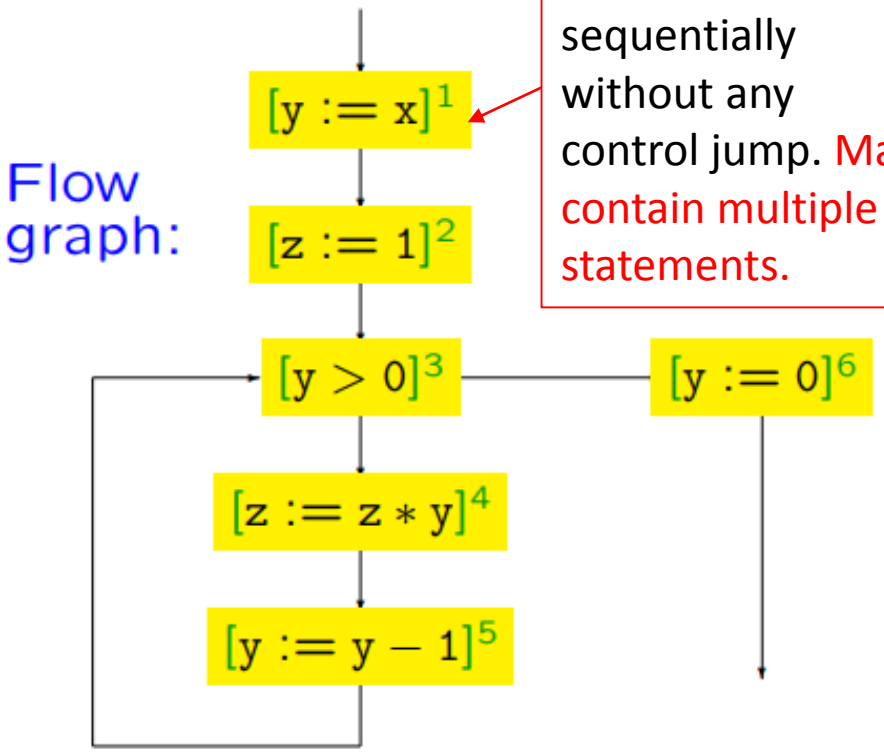


Agenda

- Program Representations
  - Basic Blocks
  - Control Flow Graphs
  - Static Single Assignment
  - Three-Address-Code
- Hands-On Session on Traversing a CFG
- More Program Representations

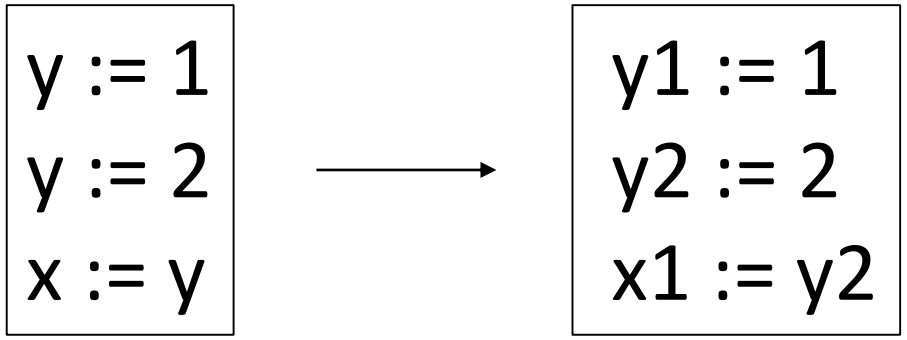
Basic Blocks and Control Flow Graphs

```
y = x;
z = 1;
while (y > 0) {
      z = z * y;
      y = y – 1;
}
y = 0
```



Basic Blocks are set of statements that execute sequentially without any control jump. May contain multiple statements.

Static Single-Assignment (SSA)



Notice that y1 is never used.
The line [y := 1] can be removed.

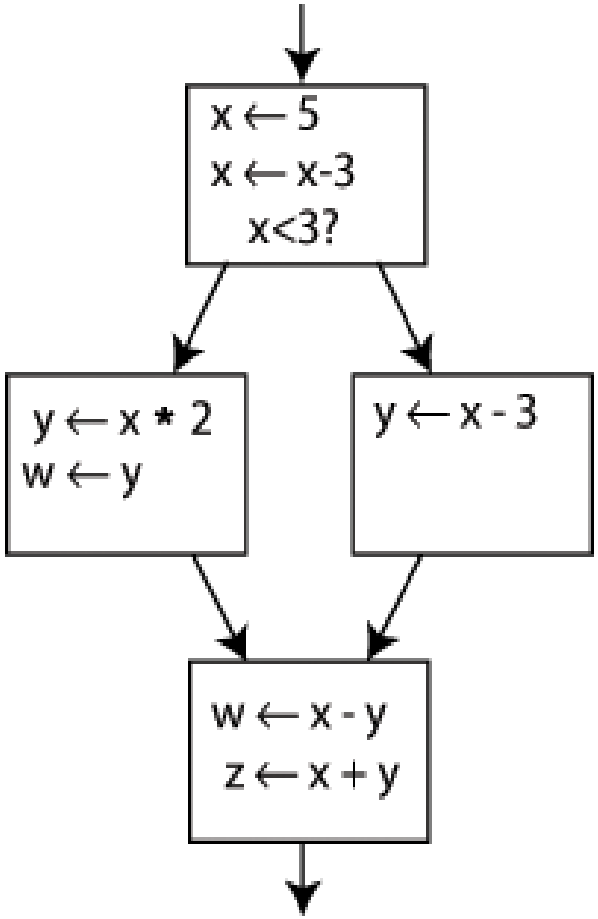
SSA

```
x := 5;
x := x – 3;
if (x < 3) {
    y := x * 2;
    w := y;
} else {
    y := x – 3;
}
w := x – y;
z  := x + y;
```

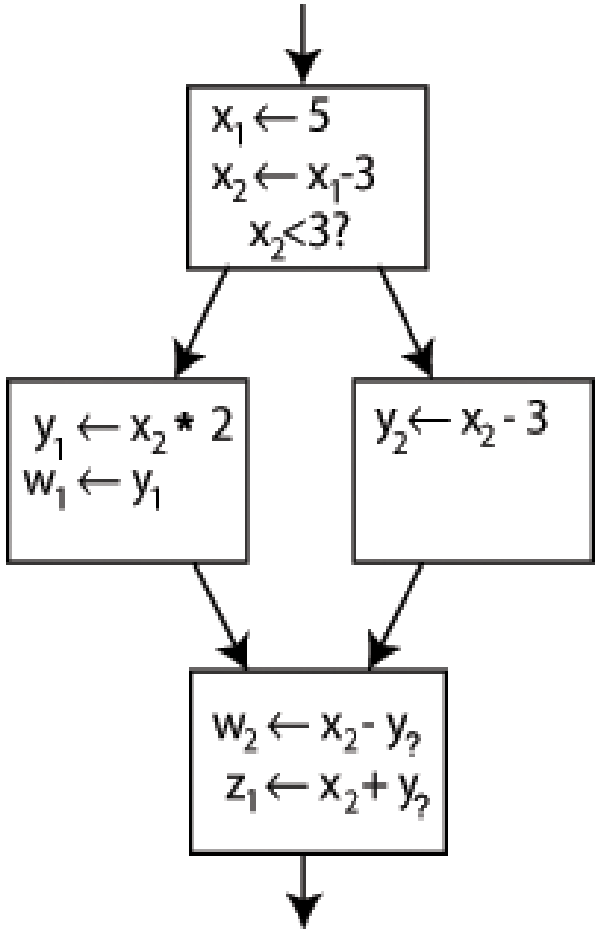
How to deal with if…else.. ?

Control Flow Graphs

```
x := 5;
x := x – 3;
if (x < 3) {
      y := x * 2;
      w := y;
} else {
      y := x – 3;
}
w := x – y;
z  := x + y;
```

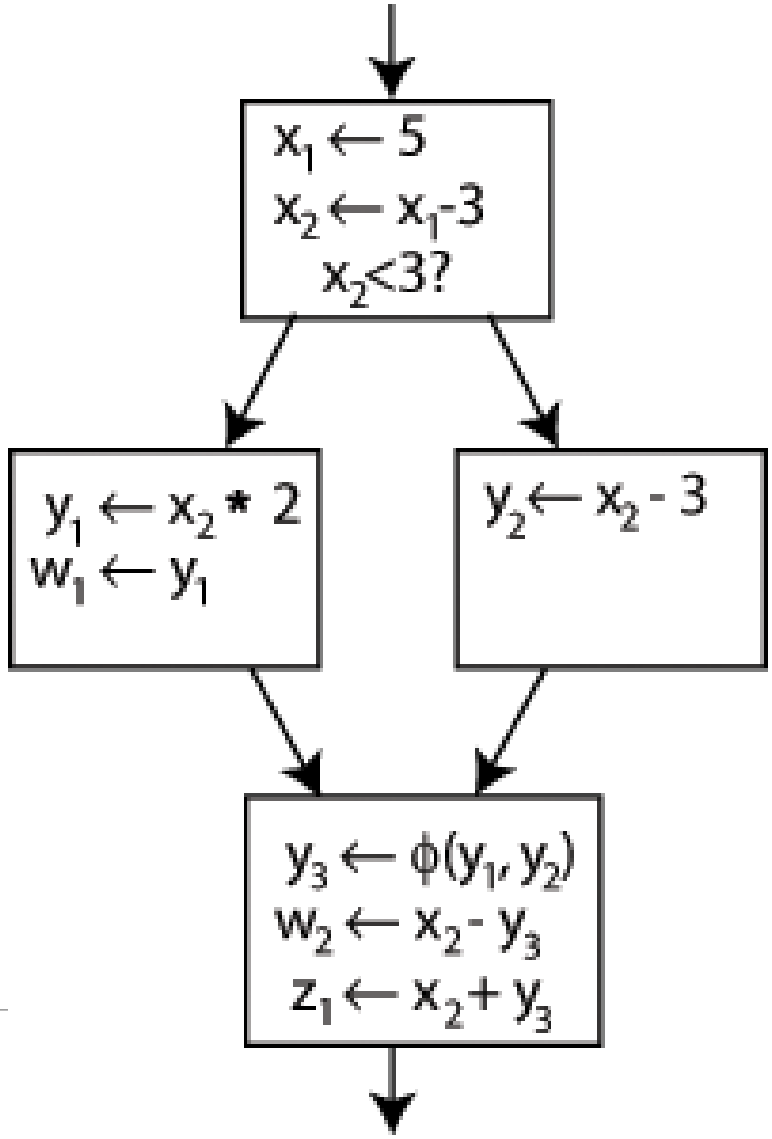


CFG



SSA Form

SSA Example

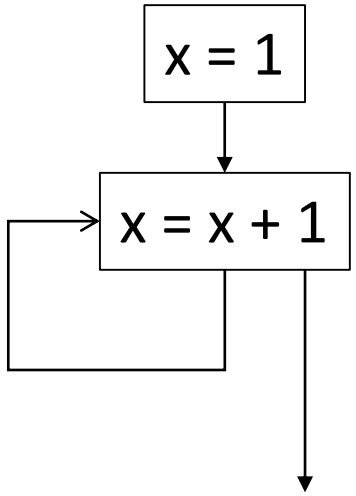


Example taken from Wikipedia

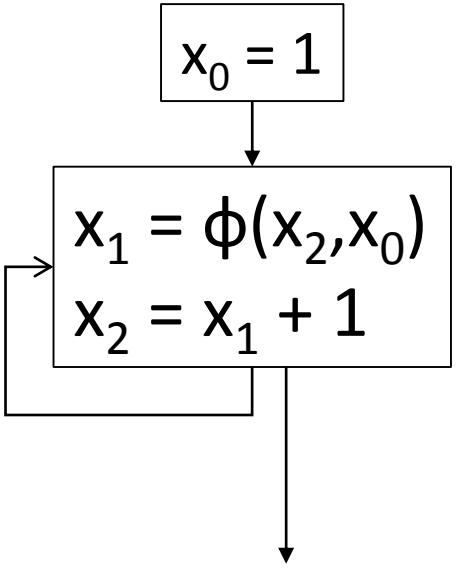
Loops and SSA

```
x = 1;
while (true) {
    x = x + 1;
}
```

Source Code

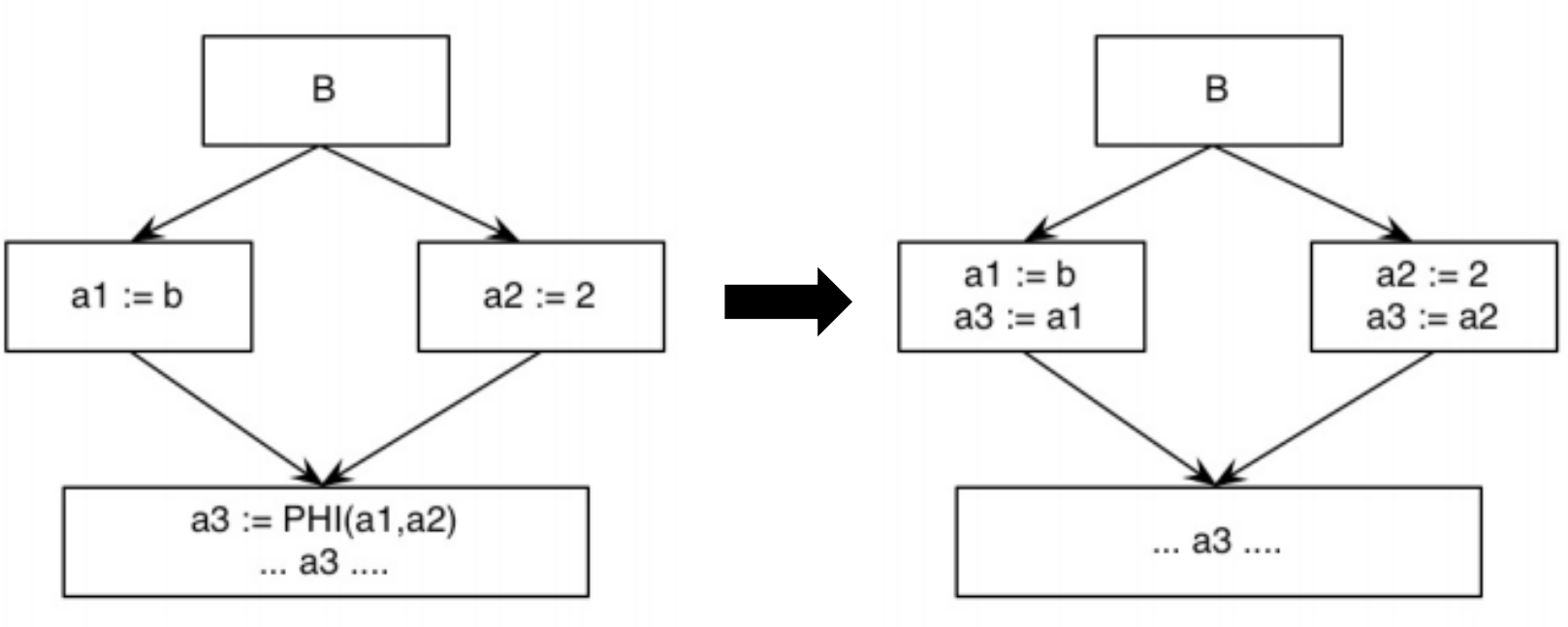


Control Flow Graph Representation



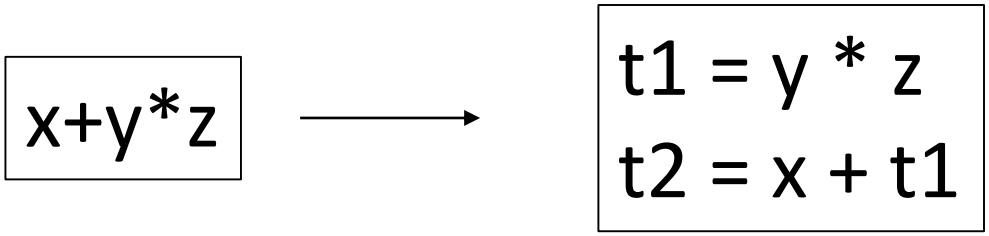
Graph Representation with SSA Form

Getting Out of SSA



Three-Address Code

At most one operator on RHS.



Section 6.2 of the Dragon Book

Example

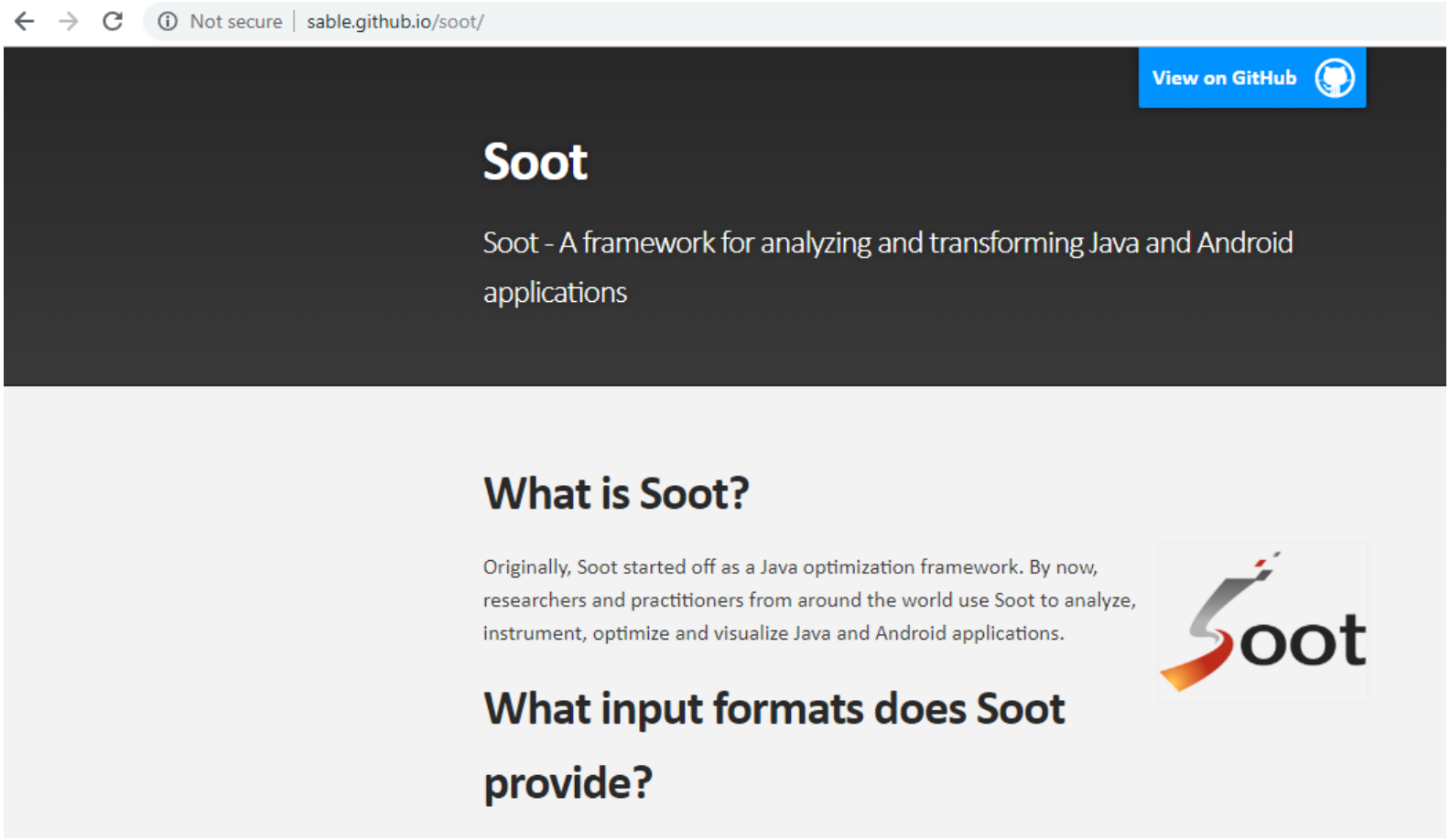
```
a = b * -c + b * -c
```

```
t1 = -c
t2 = b * t1
t3 = -c
t4 = b * t3
t5 = t2 + t4
a = t5
```

Playing with the Control Flow

Hands-On Session with Soot Framework

Soot


Not secure | sable.github.io/soot/
View on GitHub
Soot
Soot - A framework for analyzing and transforming Java and Android applications
What is Soot?
Originally, Soot started off as a Java optimization framework. By now, researchers and practitioners from around the world use Soot to analyze, instrument, optimize and visualize Java and Android applications.
Soot
What input formats does Soot provide?

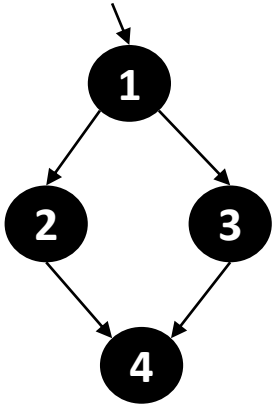
More Information

- Eclipse JDT
  - A good starting point is https://www.eclipse.org/articles/article.php?file=Article-JavaCodeManipulation_AST/index.html
  - For more information, visit https://www.eclipse.org/jdt/core/
- Soot
  - Soot Survivor's Guide - http://www.brics.dk/SootGuide/
  - Soot Tutorials - https://github.com/Sable/soot/wiki/Tutorials
  - Joe Palmer's Introductory Slides on https://www.eecis.udel.edu/~pollock/471/compiler-slides/SOOT.ppt

Dominators

- Node d of a flow graph dominates node n (written as d dom n)
  - if **every** path from the entry node of the flow graph to n goes through d.

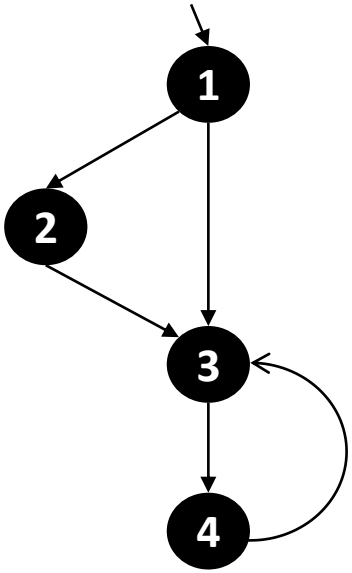
Does 1 dom 2? Yes. Every path from the entry node to 2 goes through 1.



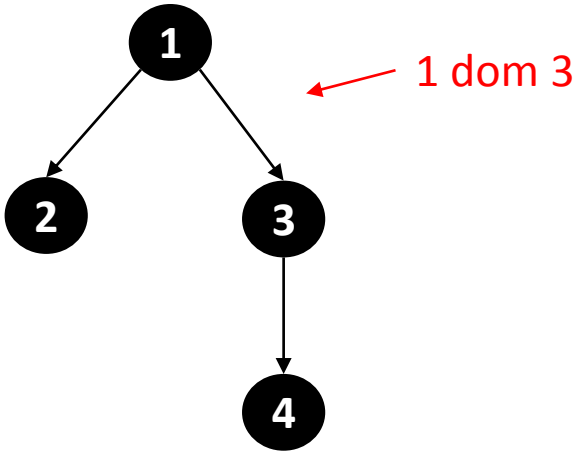
Does 2 dom 4? No. Entry-1-3-4 path does not go through 2.

Does 1 dom 4? Yes.

Dominator Tree



A Flow Graph



Dominator Tree

Dominance Tree is useful in deciding where to place the phi functions, in program slicing, etc.