

Overview of Static Analysis Research @KU

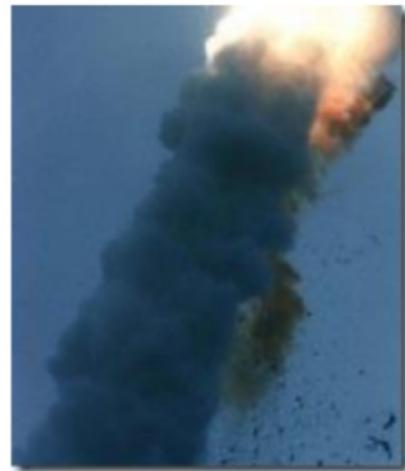
Hakjoo Oh
Programming Research Laboratory
Korea University

Motivation: Unsafe Softwares

Motivation: Unsafe Softwares

Ariane 5 Explosion

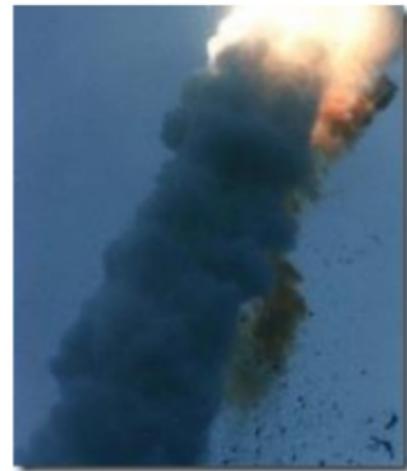
- Cost
 - \$500,000,000
- Disaster
 - ESA's Ariane 5 unmanned rocket was intentionally destroyed seconds after launch on its maiden flight
 - Also destroyed was its cargo of four scientific satellites
- Cause
 - When the guidance system tried to convert the sideways rocket velocity from 64-bits to 16-bits format, an overflow error resulted
 - When the system shut down, control passed to an **identical** redundant unit...



Motivation: Unsafe Softwares

Ariane 5 Explosion

- Cost
 - \$500,000,000
- Disaster
 - ESA's Ariane 5 unmanned rocket was intentionally destroyed seconds after launch on its maiden flight
 - Also destroyed was its cargo of four scientific satellites
- Cause
 - When the guidance system tried to convert the sideways rocket velocity from 64-bits to 16-bits format, an overflow error resulted
 - When the system shut down, control passed to an **identical** redundant unit...



Mars Polar Lander Crash

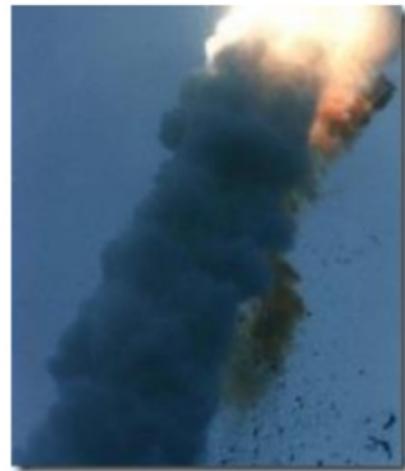
- Cost
 - \$125,000,000
- Disaster
 - After a 286-day journey from Earth, the Mars Climate Orbiter fell too far into Mars' atmosphere, causing it to crash
- Cause
 - The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA



Motivation: Unsafe Softwares

Ariane 5 Explosion

- Cost
 - \$500,000,000
- Disaster
 - ESA's Ariane 5 unmanned rocket was intentionally destroyed seconds after launch on its maiden flight
 - Also destroyed was its cargo of four scientific satellites
- Cause
 - When the guidance system tried to convert the sideways rocket velocity from 64-bits to 16-bits format, an overflow error resulted
 - When the system shut down, control passed to an **identical** redundant unit...



Mars Polar Lander Crash

- Cost
 - \$125,000,000
- Disaster
 - After a 286-day journey from Earth, the Mars Climate Orbiter fell too far into Mars' atmosphere, causing it to crash
- Cause
 - The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA



'/' 응용 프로그램에 서버 오류가 있습니다.

인덱스가 배열 범위를 벗어났습니다.

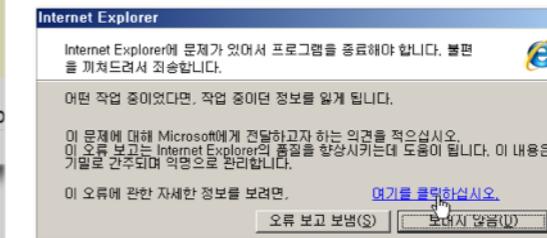
설명: 현재 웹 요청을 실행하는 동안 처리되지 않은 예외가 발생했습니다. 스택 추적을 발생한 위치에 대한 자세한 정보를 확인하십시오.

예외 정보: System.IndexOutOfRangeException: 인덱스가 배열 범위를 벗어났습니다.

소스 오류:

```
줄 192:         {  
줄 193:             new_link_aid = Regex.Split(rel_article_list[i].  
줄 194:             mco  
줄 195:         }  
줄 196:     }
```

소스 파일: d:\WEB\mnews.jtb



Motivation: Unsafe Softwares

Ariane 5 Explosion

- Cost
 - \$500,000,000
- Disaster
 - ESA's Ariane 5 unmanned rocket was intentionally destroyed seconds after launch on its maiden flight
 - Also destroyed was its cargo of four scientific satellites
- Cause
 - When the guidance system tried to convert the sideways rocket velocity from 64-bits to 16-bits format, an overflow error resulted
 - When the system shut down, control passed to an identical redundant unit...



Mars Polar Lander Crash

- Cost
 - \$125,000,000
- Disaster
 - After a 286-day journey from Earth, the Mars Climate Orbiter fell too far into Mars' atmosphere, causing it to crash
- Cause
 - The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA



'/' 응용 프로그램에 서버 오류가 있습니다.

인덱스가 배열 범위를 벗어났습니다.

설명: 현재 웹 요청을 실행하는 동안 처리되지 않은 예외가 발생했습니다. 스택 추적을 발생한 위치에 대한 자세한 정보를 확인하십시오.

예외 정보: System.IndexOutOfRangeException: 인덱스가 배열 범위를 벗어났습니다.

소스 오류:

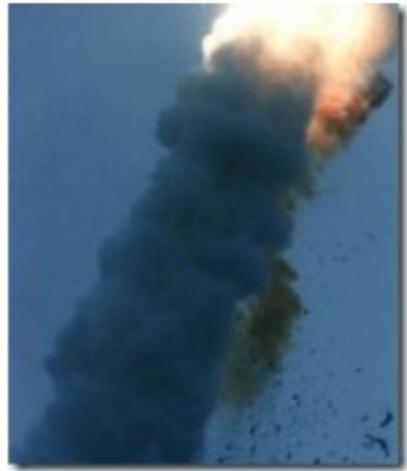
```
줄 192:         {  
줄 193:             new_link_aid = Regex.Split(rel_article_list[i].  
줄 194:             m[
```



Motivation: Unsafe Softwares

Ariane 5 Explosion

- Cost
 - \$500,000,000
- Disaster
 - ESA's Ariane 5 unmanned rocket was intentionally destroyed seconds after launch on its maiden flight
 - Also destroyed was its cargo of four scientific satellites
- Cause
 - When the guidance system tried to convert the sideways rocket velocity from 64-bits to 16-bits format, an overflow error resulted
 - When the system shut down, control passed to an identical redundant unit...



Mars Polar Lander Crash

- Cost
 - \$125,000,000
- Disaster
 - After a 286-day journey from Earth, the Mars Climate Orbiter fell too far into Mars' atmosphere, causing it to crash
- Cause
 - The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA



'/' 응용 프로그램에 서버 오류가 있습니다.

인덱스가 배열 범위를 벗어났습니다.

설명: 현재 웹 요청을 실행하는 동안 처리되지 않은 예외가 발생했습니다. 스택 추적을 발생한 위치에 대한 자세한 정보를 확인하십시오.

예외 정보: System.IndexOutOfRangeException: 인덱스가 배열 범위를 벗어났습니다.

소스 오류:

```
줄 192:         {  
줄 193:             new_link_aid = Regex.Split(rel_article_list[i].  
줄 194:             m{
```



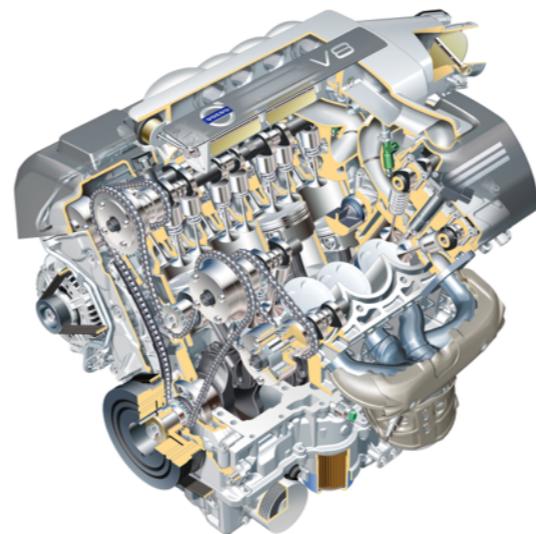
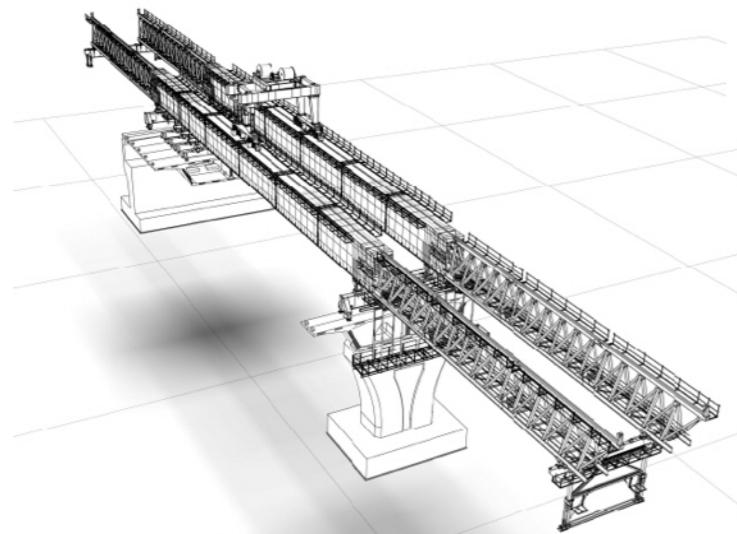
과학
과학일반

'나로호 불발' 압력측정 소프트웨어 오류가 원인



The Fundamental Reason

- Will our engineered artifact behave as intended?



Current Technology for Safe SW

Manual, ad-hoc, postmortem:

code review, testing, simulation, debugging, etc

Our Mission

Technology for “Software MRI”



Example: Sparrow



The Early Bird

- Detect memory errors in C programs
 - e.g., buffer-overrun, memory leak, null-dereference, etc
- Features (vs. testing)
 - Full **automation**
 - Find bugs **early**
 - **All bugs** found (ensured by theory)

```
16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input ();
19
20 if (((KeySym)(keysym) >= 0xFF91) && ((KeySym)(keysym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             curfinal[keysym] = 1;
33         }
34     else
35     {
36         curfinal[keysym] = 2;
37     }
38 }
39 if (keysym < 10)
40 {
41     unparseputc(curfinal[keysym], pty);
42 }
43 }
```

```
16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input ();
19
20 if (((KeySym)(keysym) >= 0xFF91) && ((KeySym)(keysym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             safe → curfinal[keysym] = 1;
33         }
34     }
35     else
36     {
37         buffer-overrun → curfinal[keysym] = 2;
38     }
39     if (keysym < 10)
40     {
41         unparseputc(curfinal[keysym], pty);
42     }
43 }
```

Sparrow automatically
pinpoints the buffer-overrun bug

```
16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input();
19
20 if (((KeySym)(keysym) >= 0xFF91) && ((KeySym)(keysym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             safe curfinal[keysym] = 1;
33         }
34     }
35     else
36     {
37         buffer-overrun curfinal[keysym] = 2;
38     }
39     if (keysym < 10)
40     {
41         unparseputc(curfinal[keysym], pty);
42     }
43 }
```

curfinal: buffer of size 10

Sparrow automatically
pinpoints the buffer-overrun bug

```
16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input();
19
20 if (((KeySym)(keysym) >= 0xFF90 & keysym: any integer &ym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             safe curfinal[keysym] = 1;
33         }
34     else
35     {
36         buffer-overrun curfinal[keysym] = 2;
37     }
38     if (keysym < 10)
39     {
40         unparseputc(curfinal[keysym], pty);
41     }
42 }
43 }  
safe
```

curfinal: buffer of size 10

Sparrow automatically
pinpoints the buffer-overrun bug

```

16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input();
19
20 if (((KeySym)(keysym) >= 0xFF90 & keysym: any integer &ym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)           keysym: [0,15]
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             safe curfinal[keysym] = 1;      Sparrow automatically
33         }
34         else
35         {
36             buffer-overrun curfinal[keysym] = 2;    pinpoints the buffer-overrun bug
37         }
38     }
39     if (keysym < 10)
40     {
41         unparseputc(curfinal[keysym], pty);
42     }
43 }

```

curfinal: buffer of size 10

keysym: [0,15]

safe

buffer-overrun

curfinal[keysym] = 2;

safe

```

16 static char *curfinal = "HDACB  FE";
17
18 keysym = read_from_input();
19
20 if (((KeySym)(keysym) >= 0xFF90 & keysym: any integer &ym) <= 0xFF94))
21 {
22     unparseputc((char)(keysym-0xFF91 + 'P'), pty);
23     key = 1;
24 }
25 else if (keysym >= 0)
26 {
27     if (keysym < 16)           keysym: [0,15]
28     {
29         if (read_from_input())
30         {
31             if (keysym >= 10) return;
32             safe curfinal[keysym] = 1;      Sparrow automatically
33         }
34         else
35         {
36             buffer-overrun curfinal[keysym] = 2;    pinpoints the buffer-overrun bug
37             if (keysym < 10)                      curfinal:[10,10]
38             {
39                 if (keysym < 10)                  keysym:[10,15]
40                 {
41                     unparseputc(curfinal[keysym], pty);
42                 }
43             }
44         }
45     }
46 }

```

curfinal: buffer of size 10

keysym: [0,15]

Sparrow automatically
pinpoints the buffer-overrun bug

curfinal:[10,10]
keysym:[10,15]

buffer-overrun

curfinal[keysym] = 2;

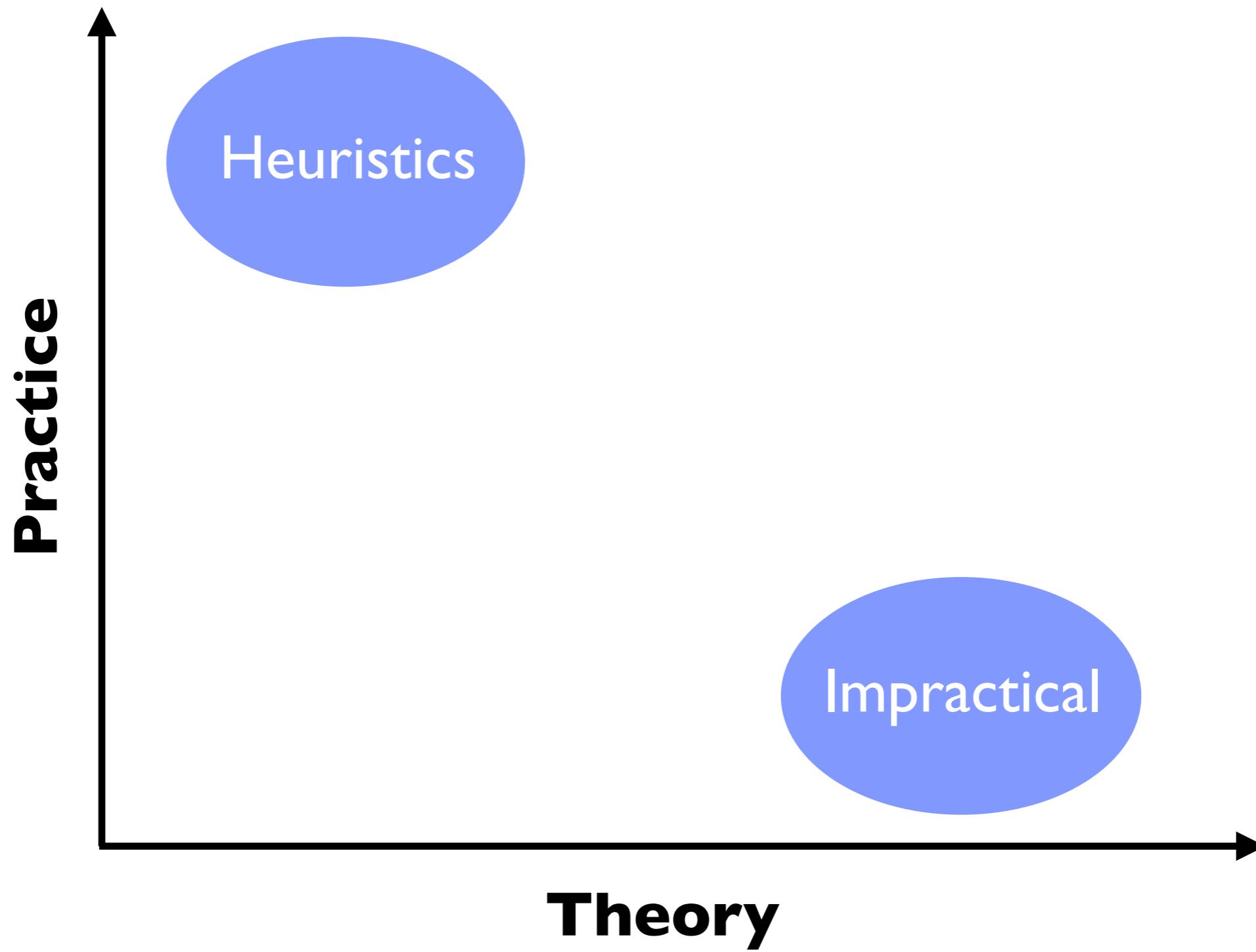
safe

Static Program Analysis

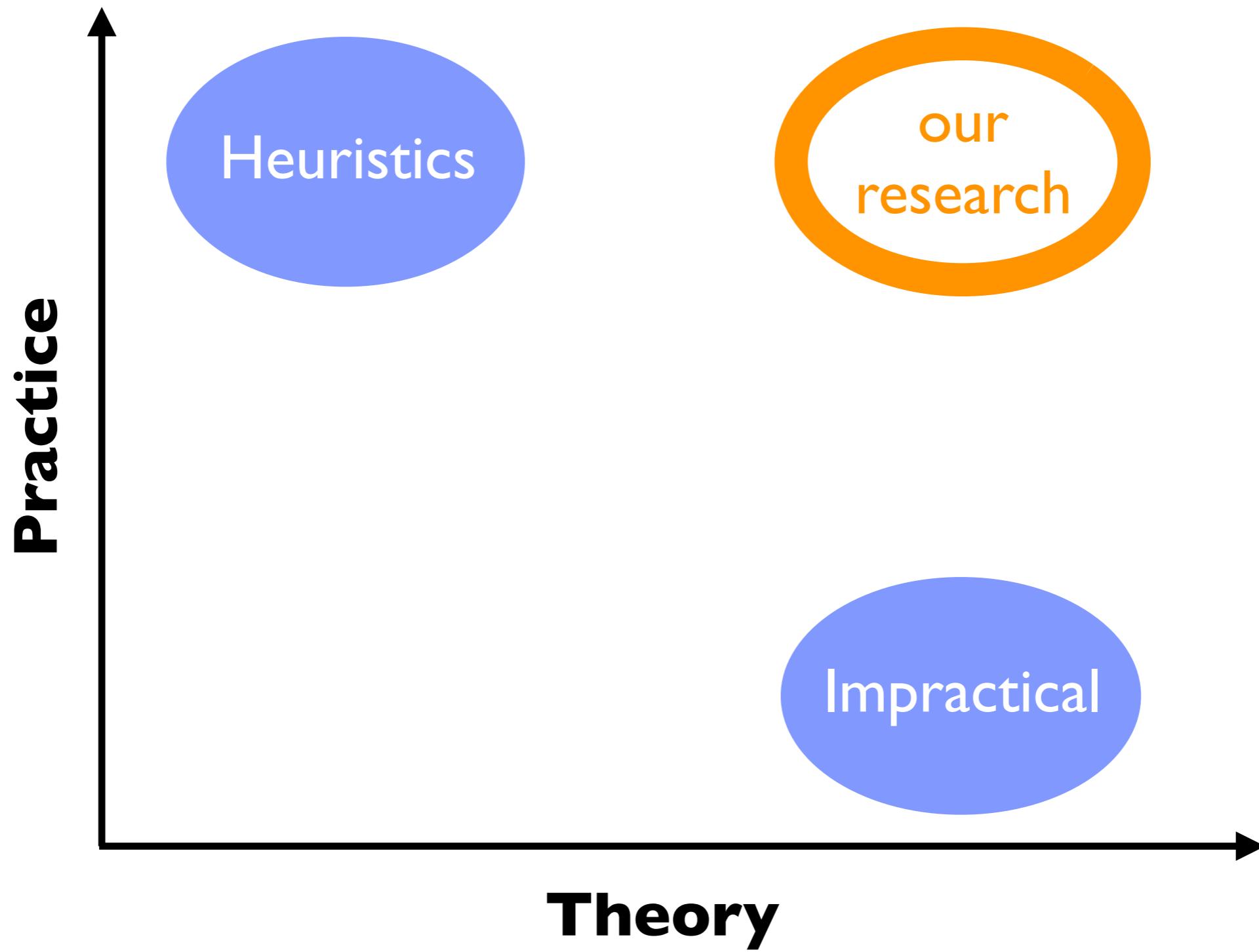
- Predict SW behavior statically and automatically
 - **static**: before execution, before sell / embed
 - **automatic**: sw is analyzed by sw (“static analyzers”)
 - **systematic**: based on foundational theory (Abstract Interpretation)

Our Research

Direction



Direction

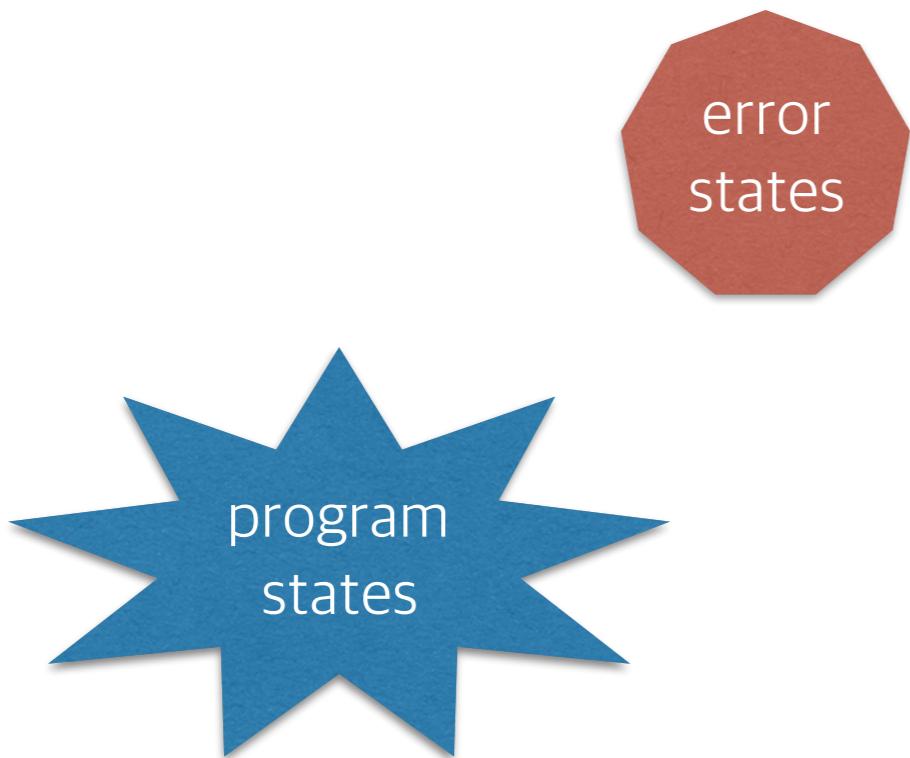


The Contribution

Achieved **sound**, **precise**, and **scalable** static analysis

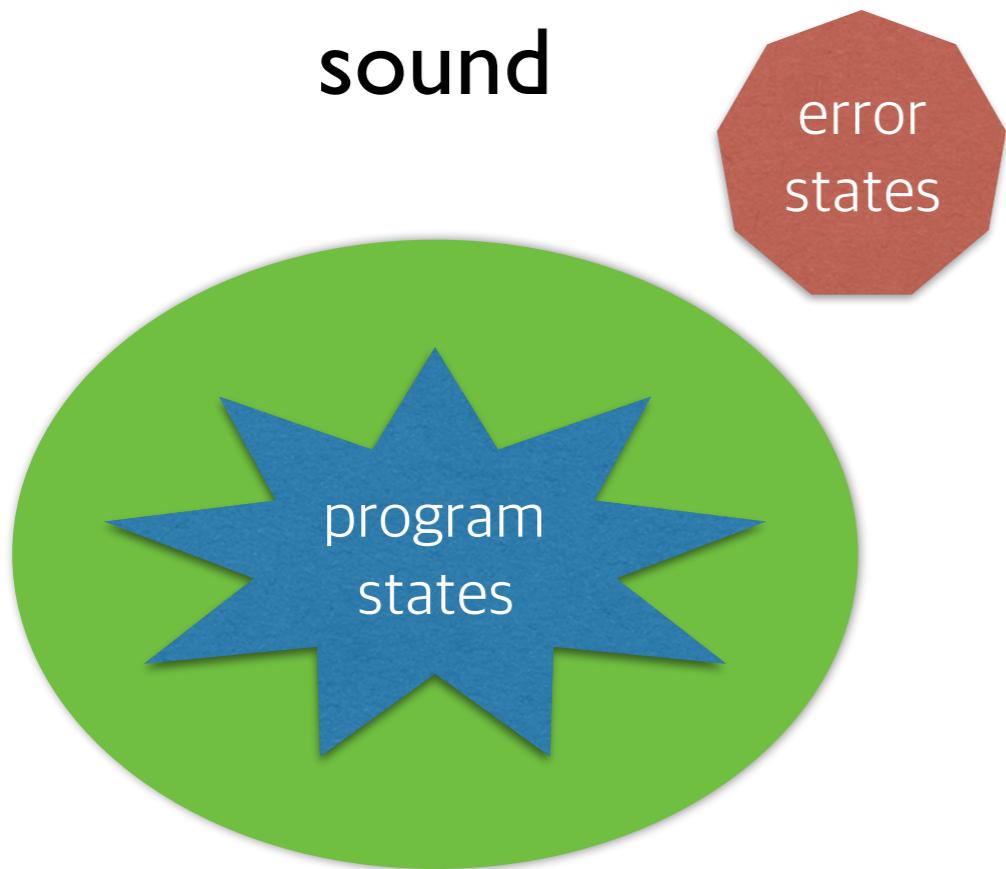
(1) Soundness

Find all bugs / verify absence



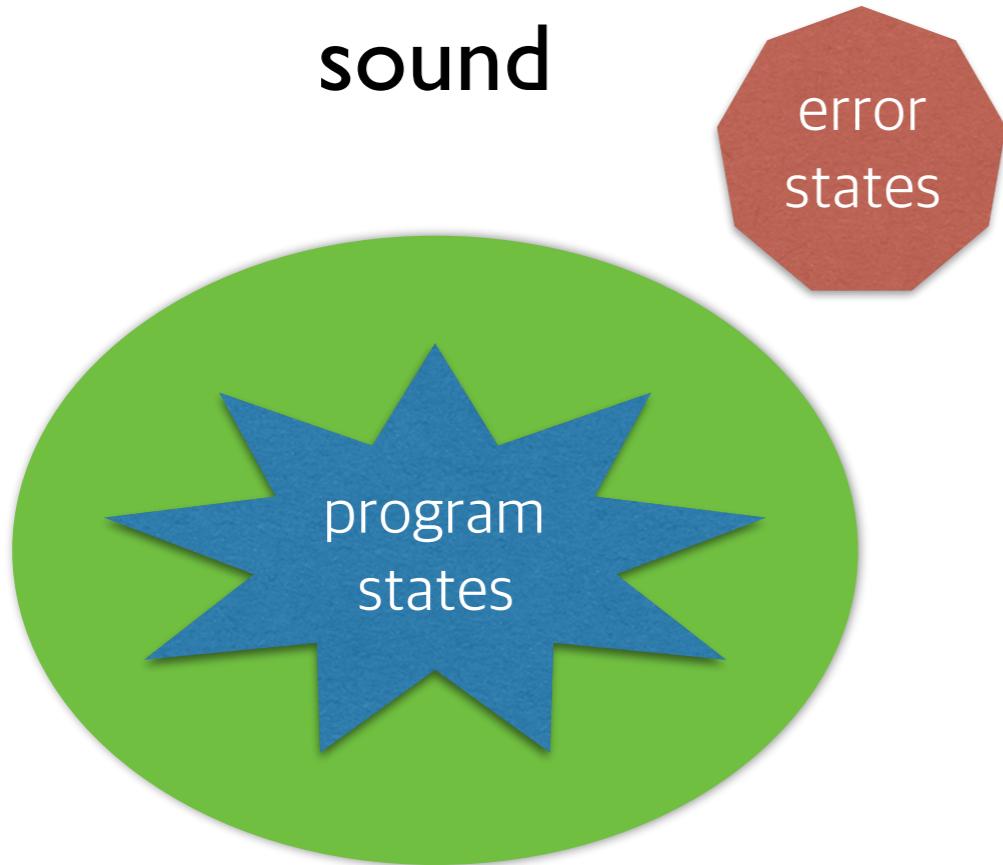
(1) Soundness

Find all bugs / verify absence

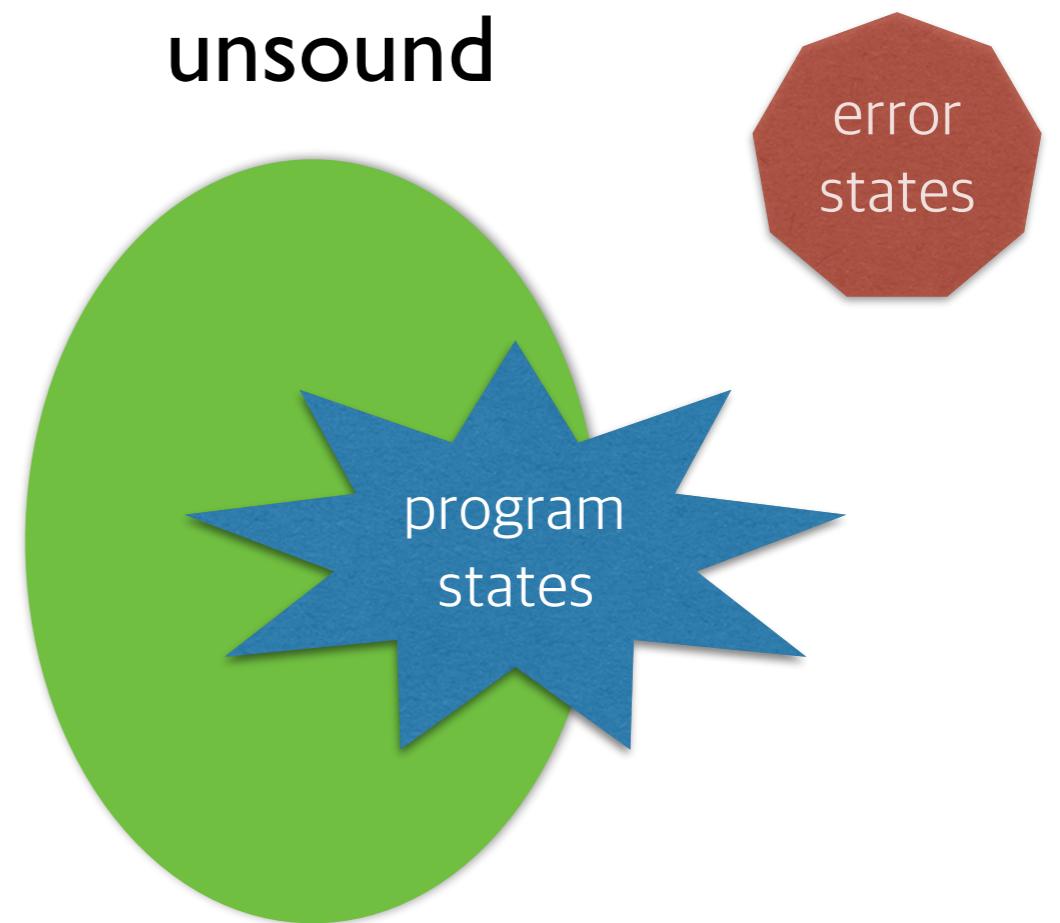


(1) Soundness

Find all bugs / verify absence



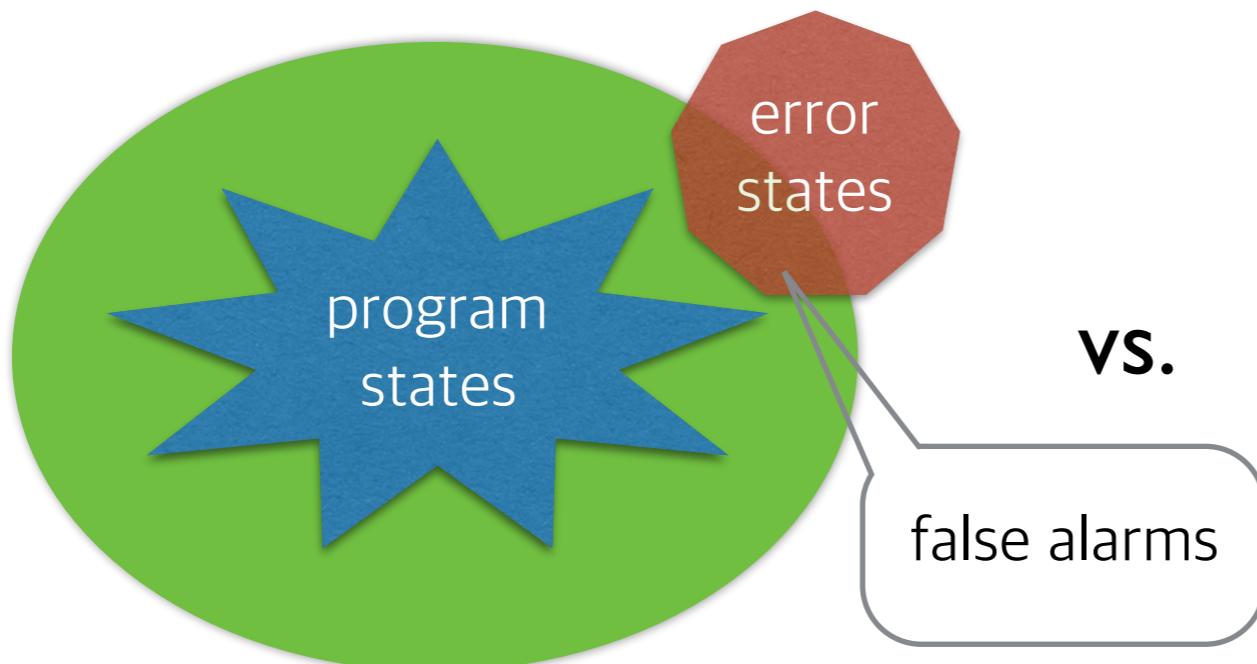
vs.



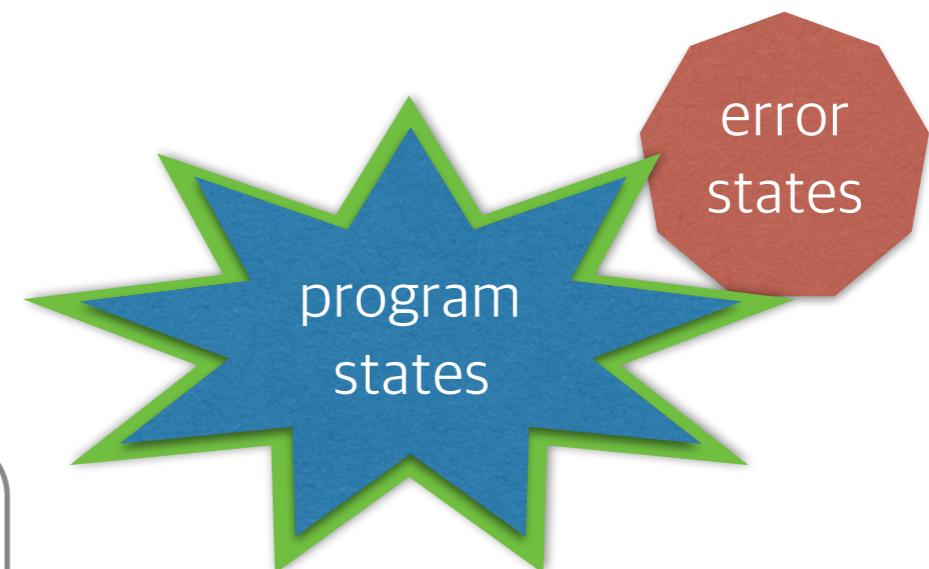
(2) Precision

Few false alarms

imprecise



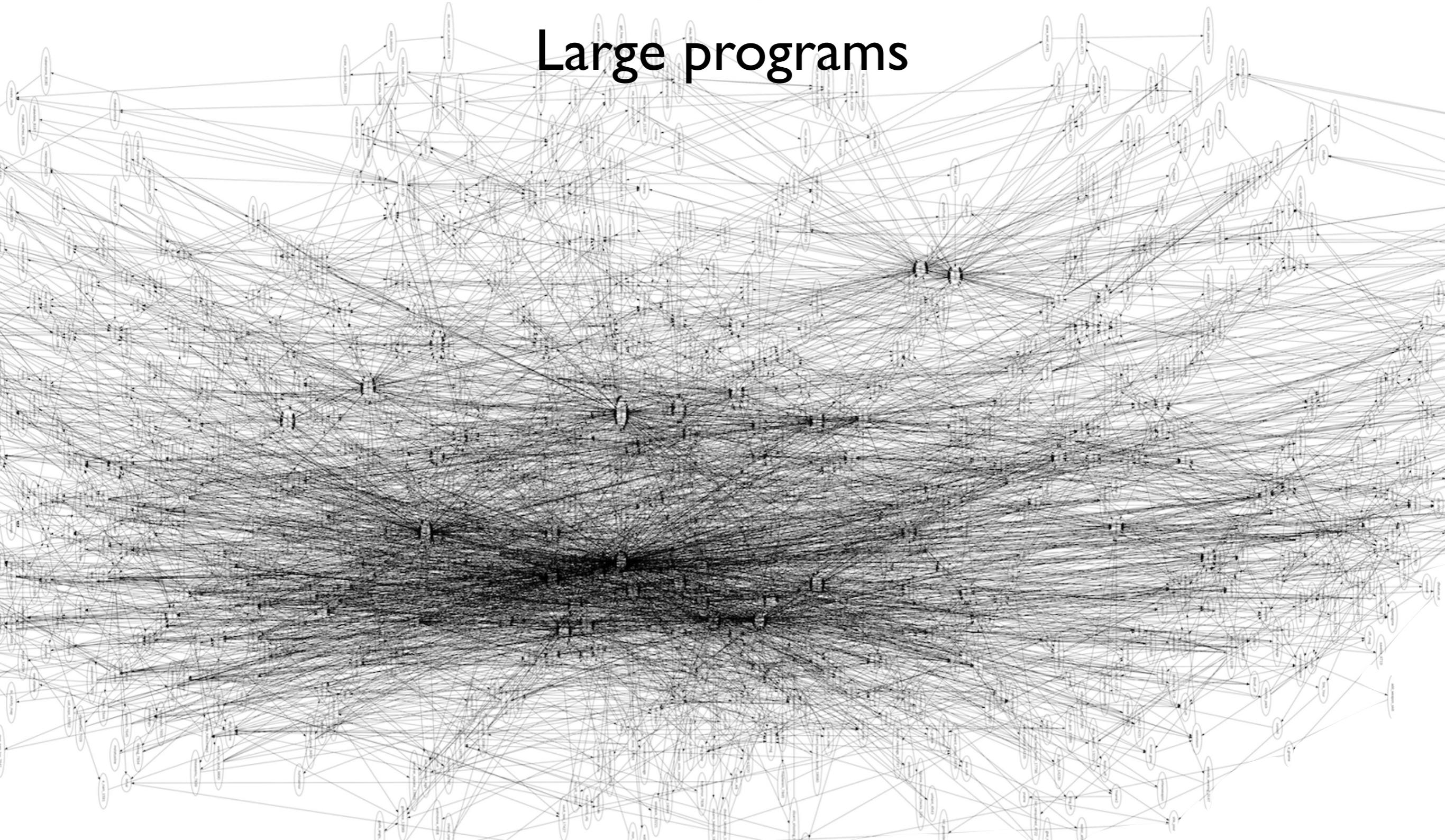
precise



(3) Scalability

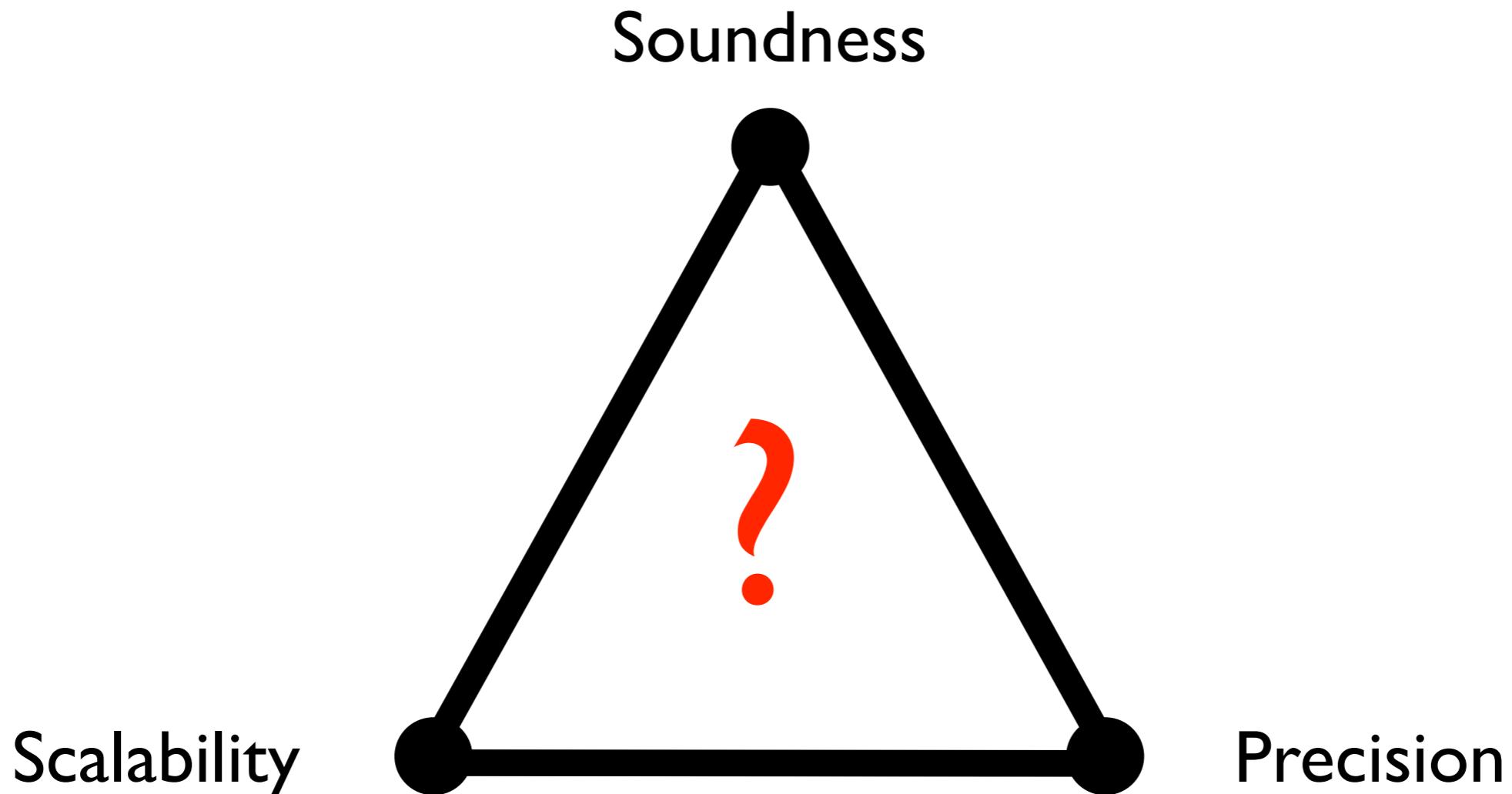
nethack-3.3.0 (211KLoC)

Large programs



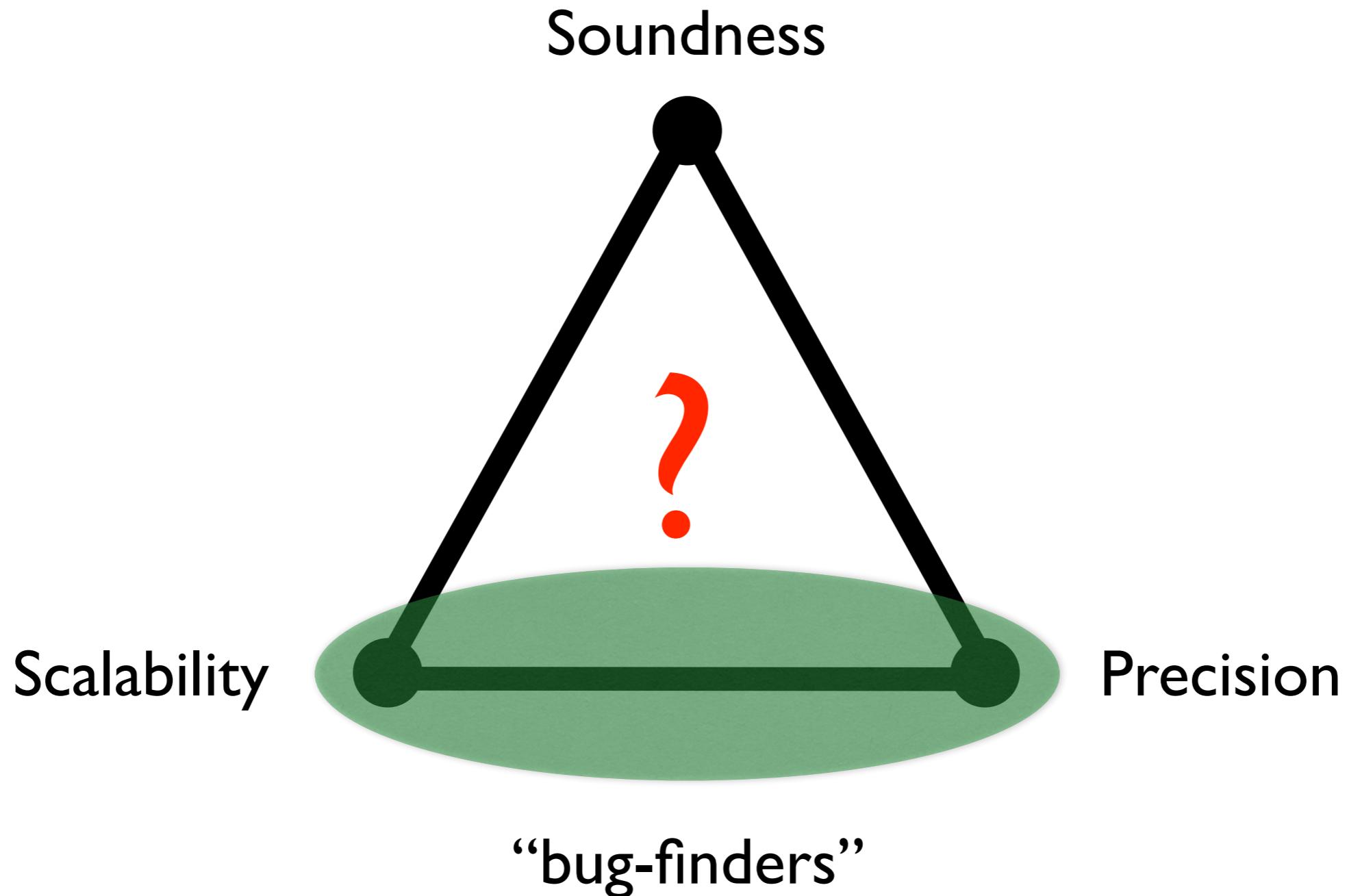
Before
our work

Common Sense: Infeasible



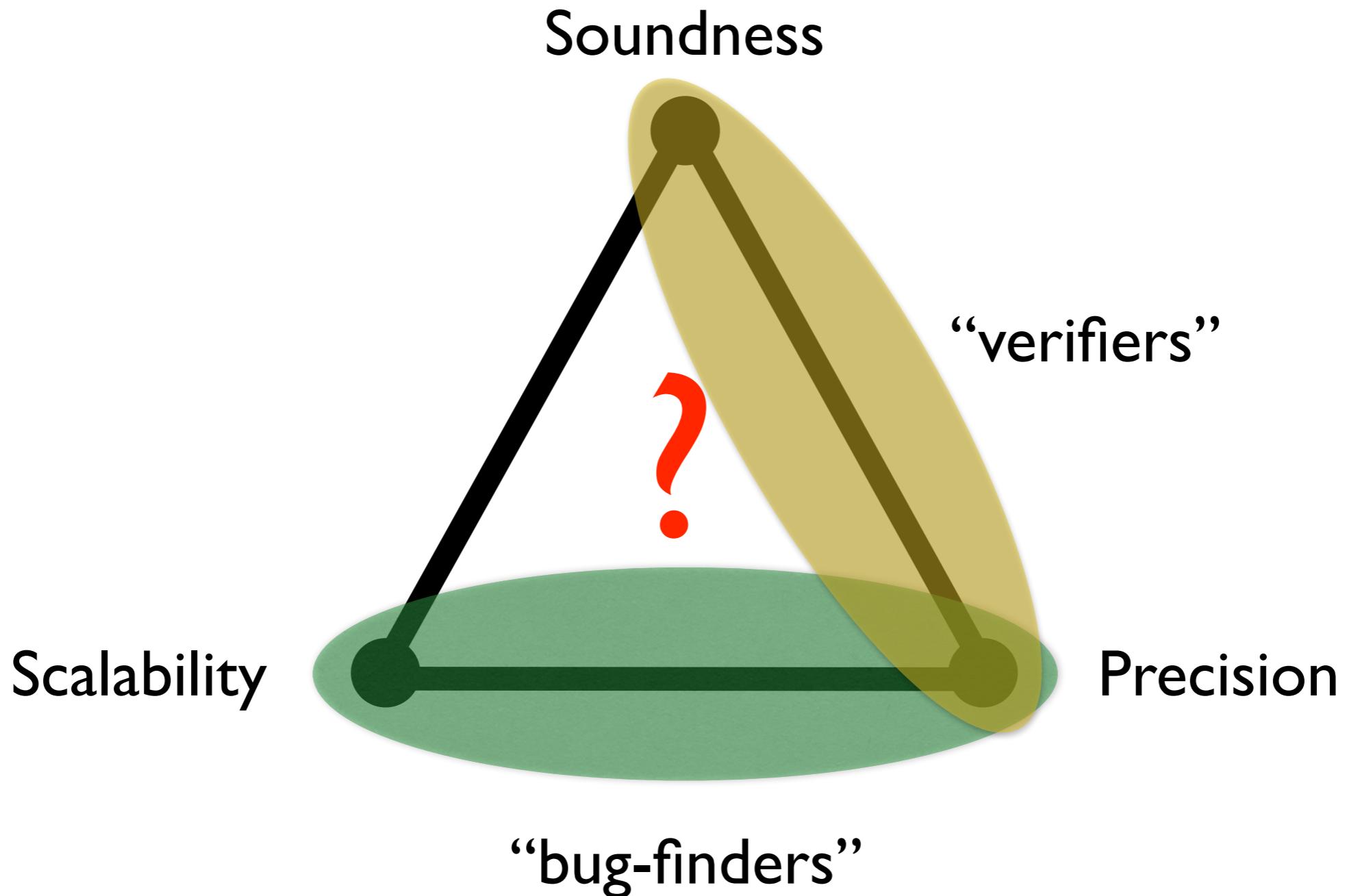
Before
our work

Common Sense: Infeasible



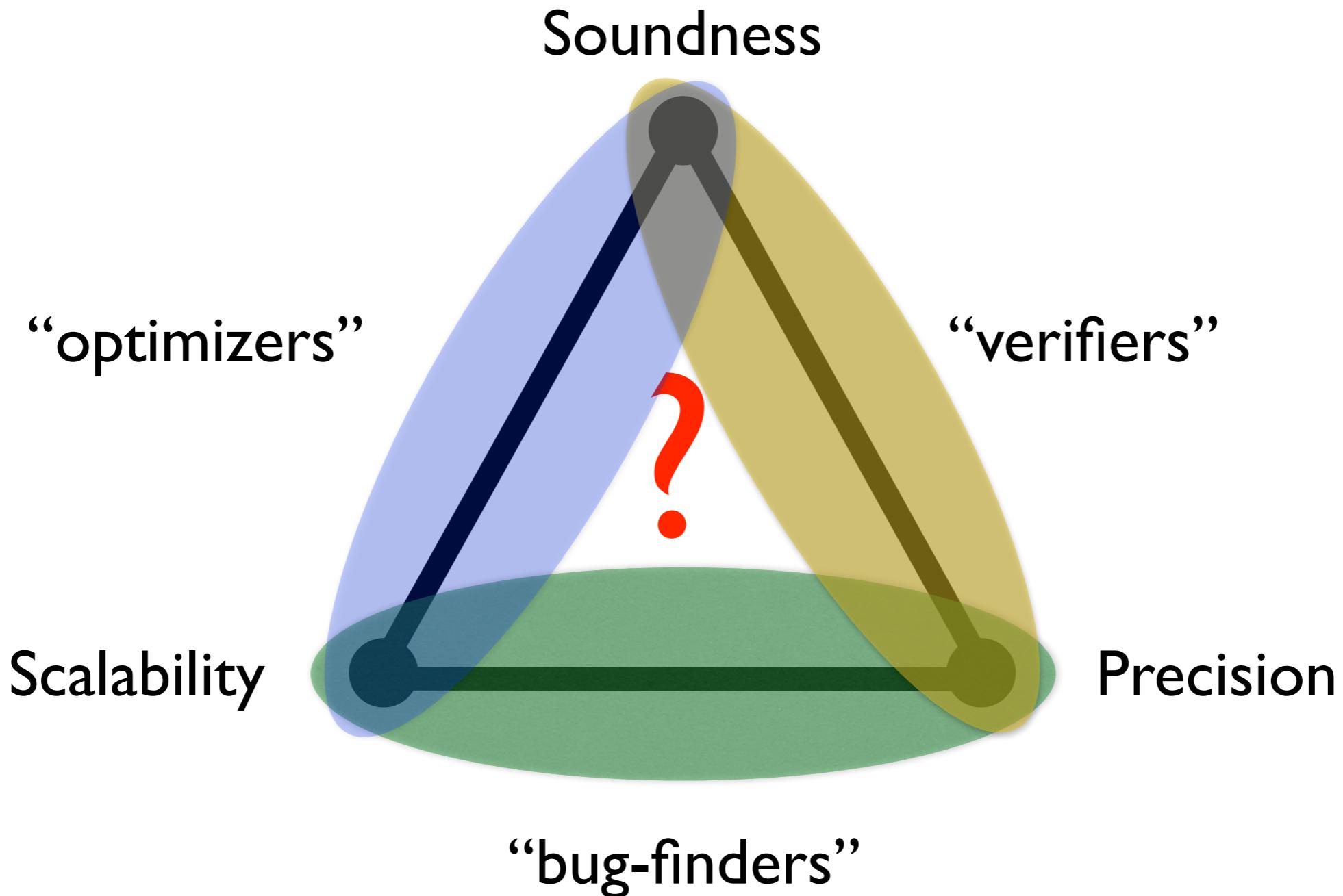
Before
our work

Common Sense: Infeasible



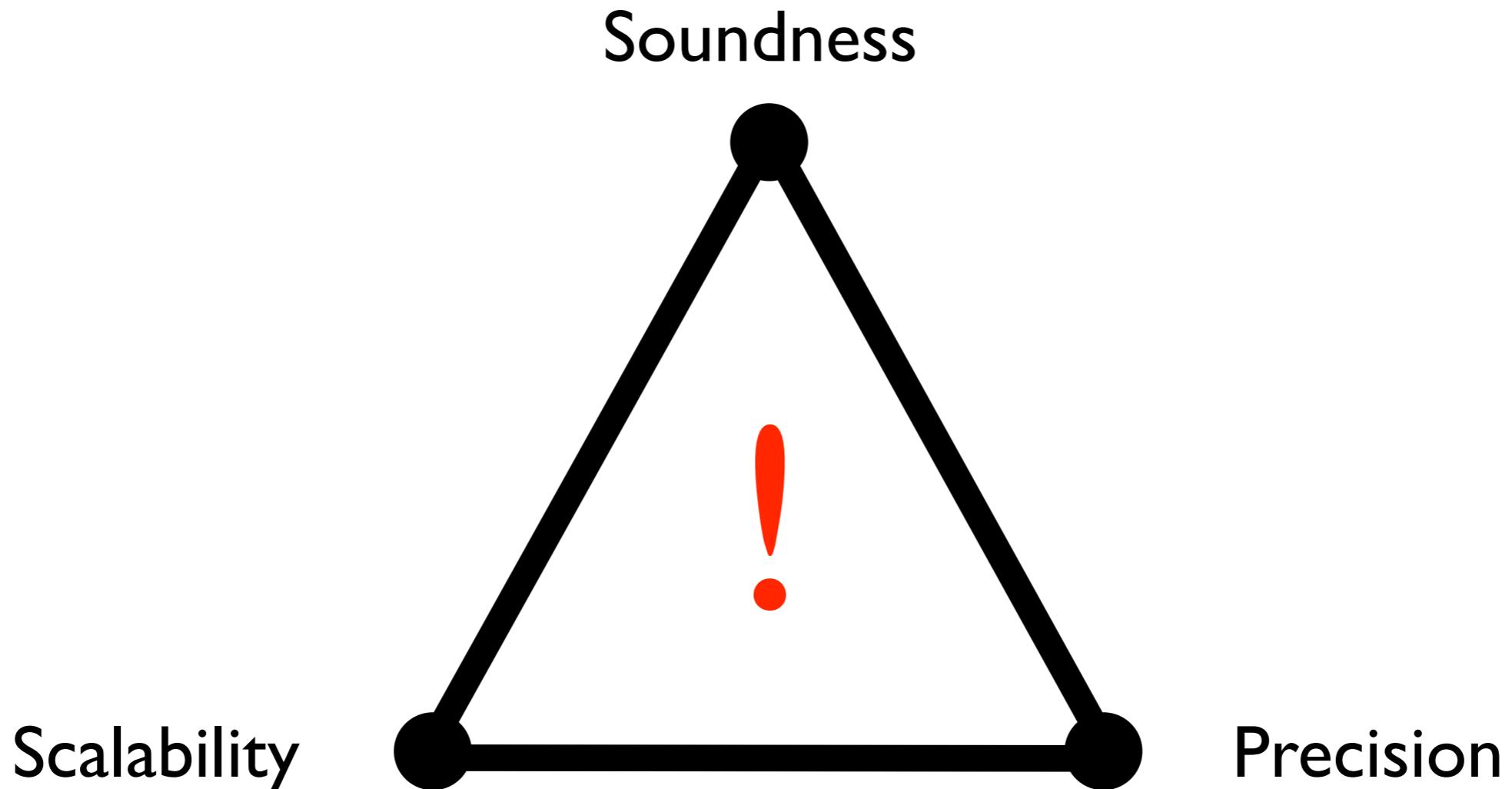
Before
our work

Common Sense: Infeasible



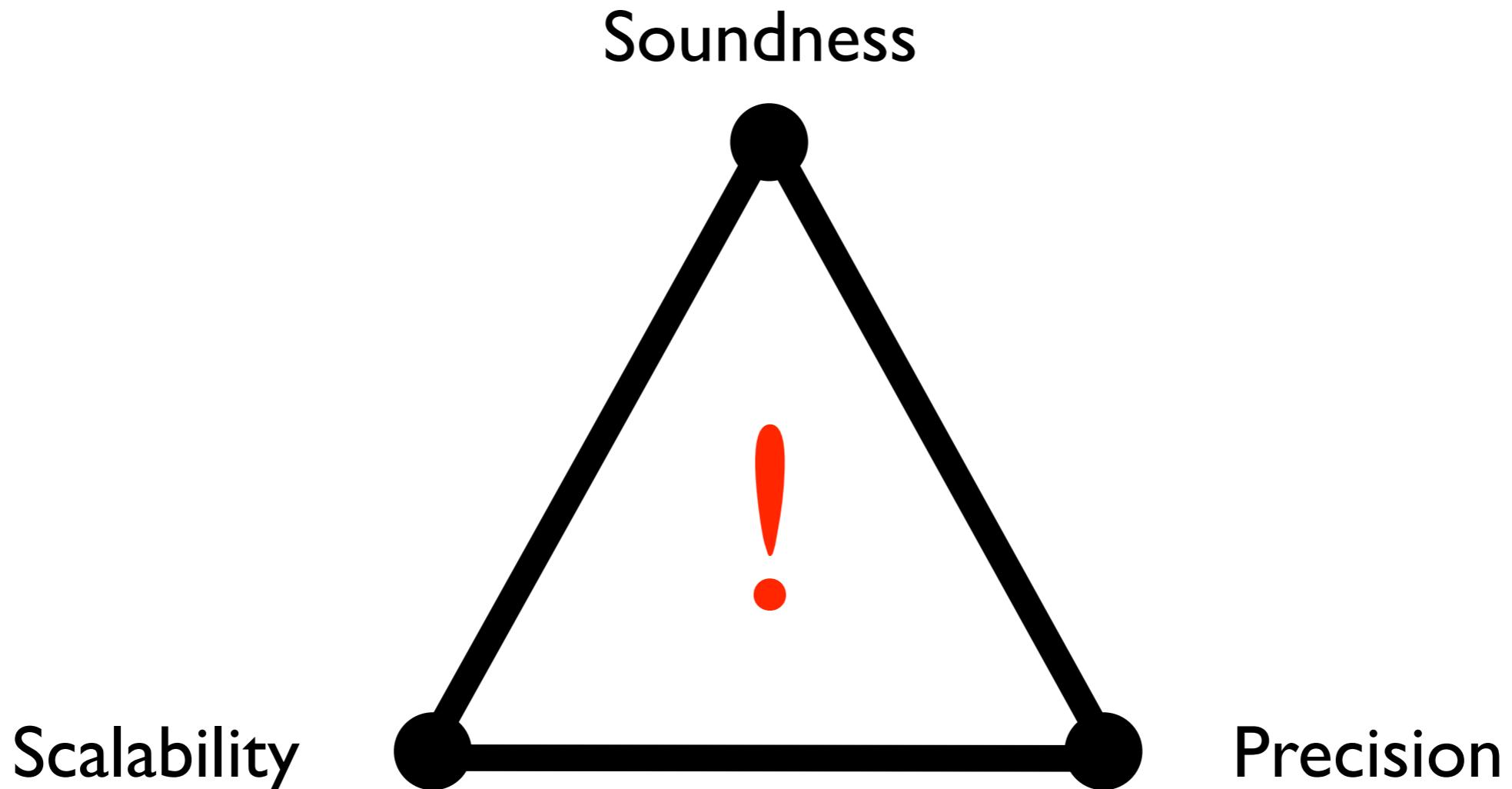
After
our work

Not Any More



After
our work

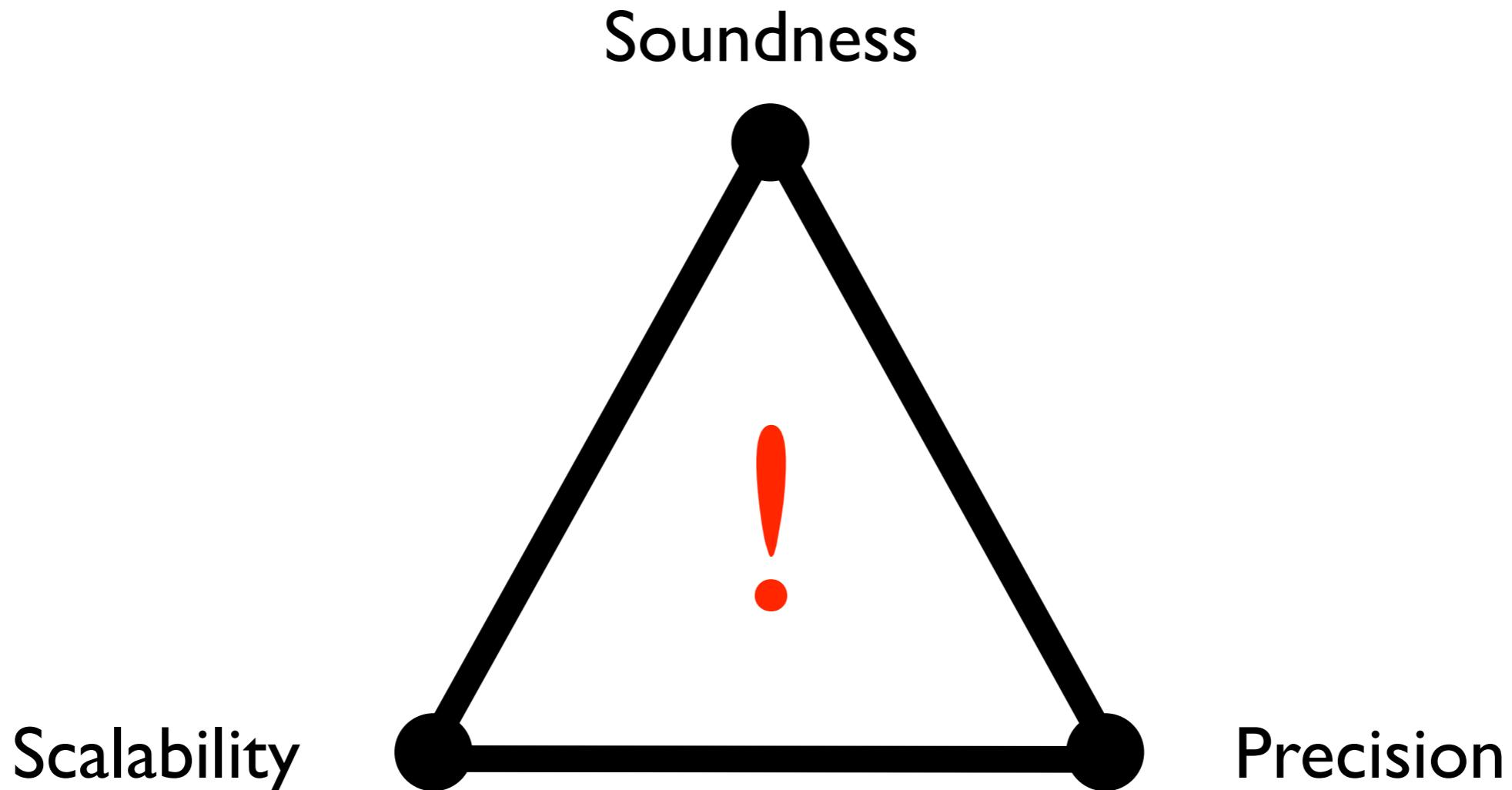
Not Any More



**General Sparse
Analysis Framework
[PLDI'12]**

After
our work

Not Any More



**General Sparse
Analysis Framework
[PLDI'12]**

**Selective X-Sensitivity
Approach
[PLDI'14, OOPSLA'15]**

Significance

- Cracked down the common sense that sound, precise, and scalable static analysis is infeasible
- Publication:
 - General Sparse Analysis Framework
 - **ACM PLDI 2012** (top conference in programming languages)
 - **ACM TOPLAS 2014** (top journal in programming languages)
 - Selective X-Sensitivity Approach
 - **ACM PLDI 2014** (top conference in programming languages)
 - **ACM OOPSLA 2015** (top conference in programming languages)
 - **ACM TOPLAS 2015** (top journal in programming languages)

Motivation

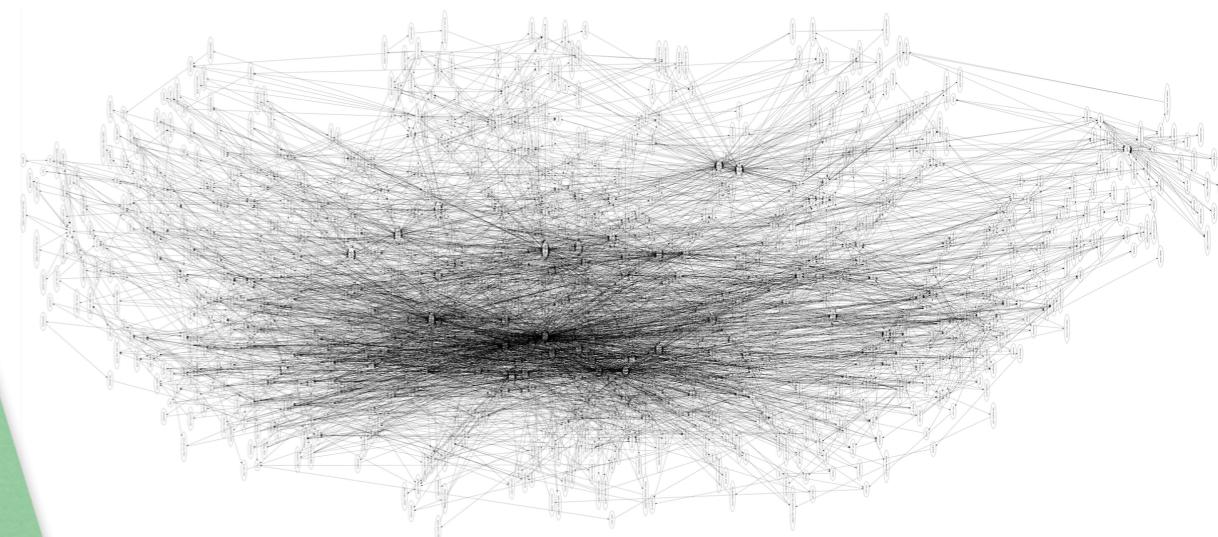
- In 2007, commercialized 
 - memory-bug-finding tool for full C
 - sound in design, unsound yet scalable in reality
- Realistic workbench available
 - “let’s try to achieve sound, precise, yet scalable version”

The Challenge in Reality



(2007, sound-&-global version)

Soundness



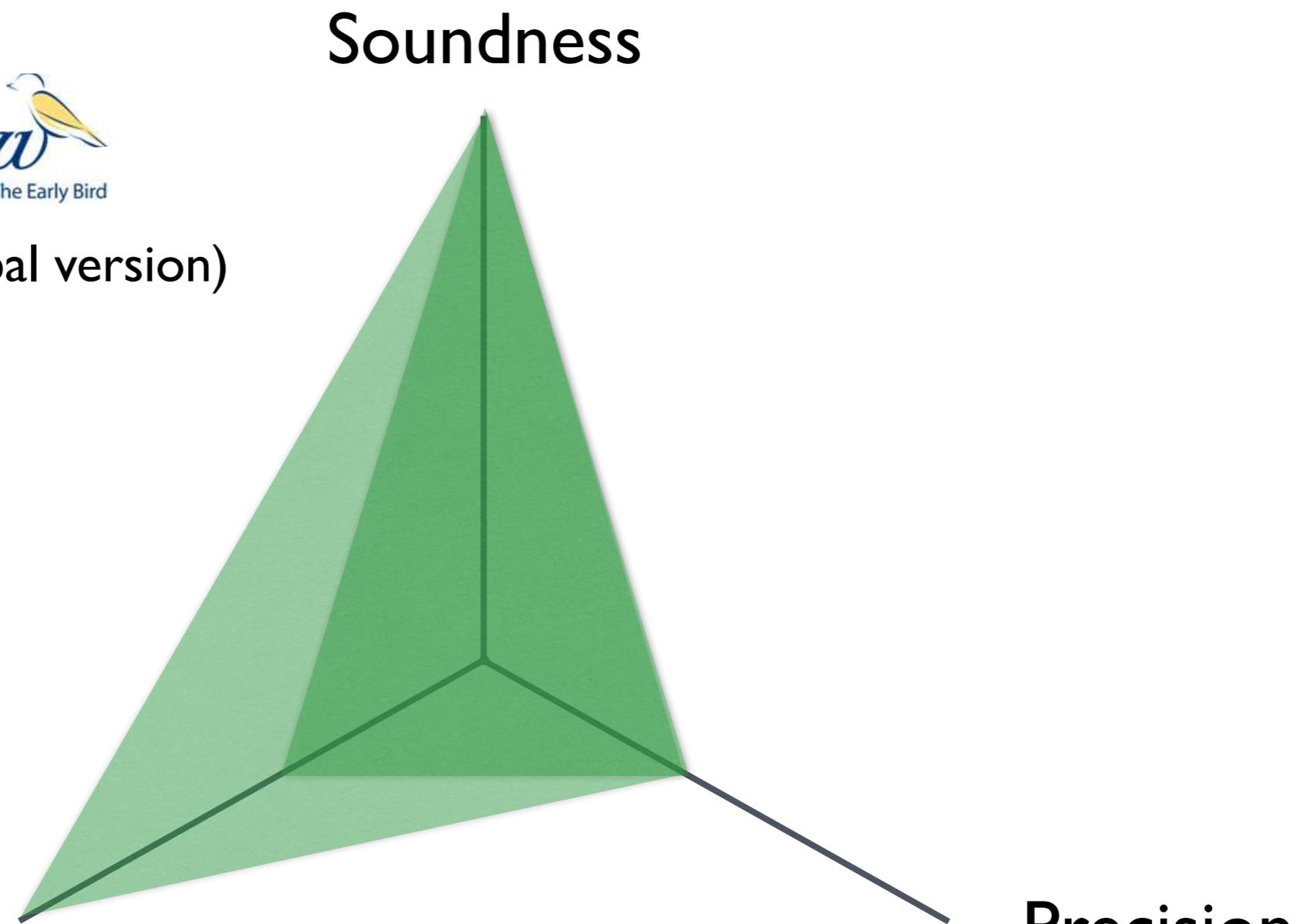
Scalability

Precision

The First Goal: Scalability



(2012, sound-&-global version)

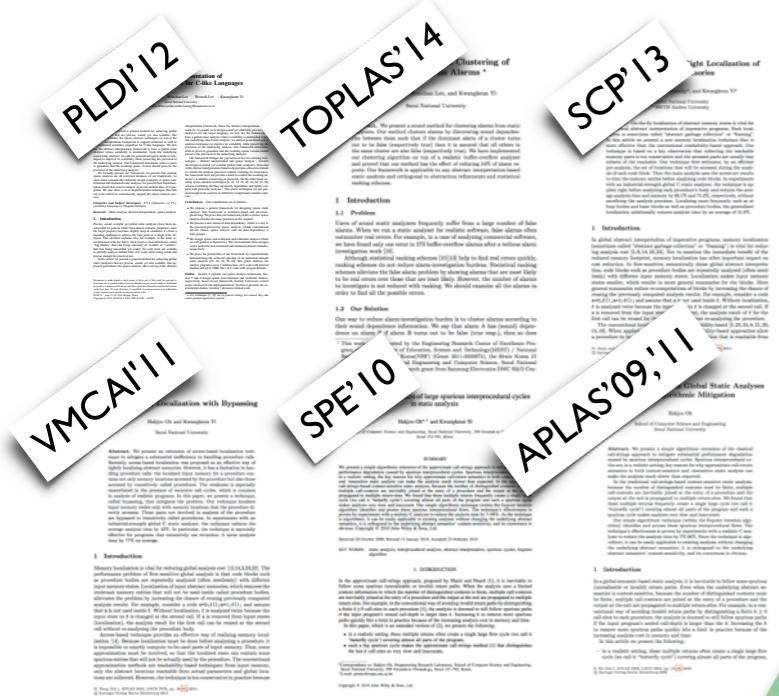


Scalability
General Sparse
Analysis Framework
[PLDI'12]

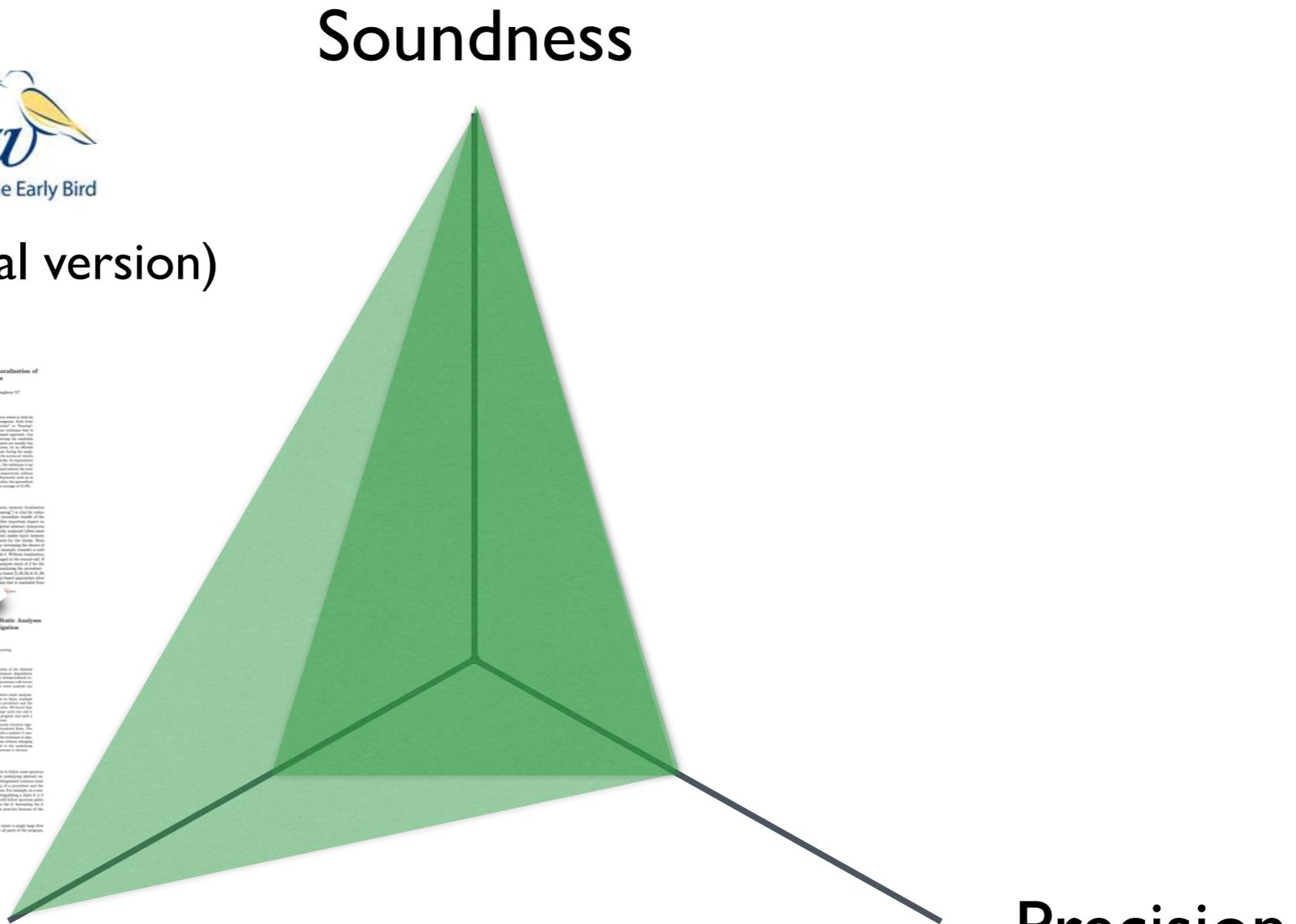
The First Goal: Scalability



(2012, sound-&-global version)



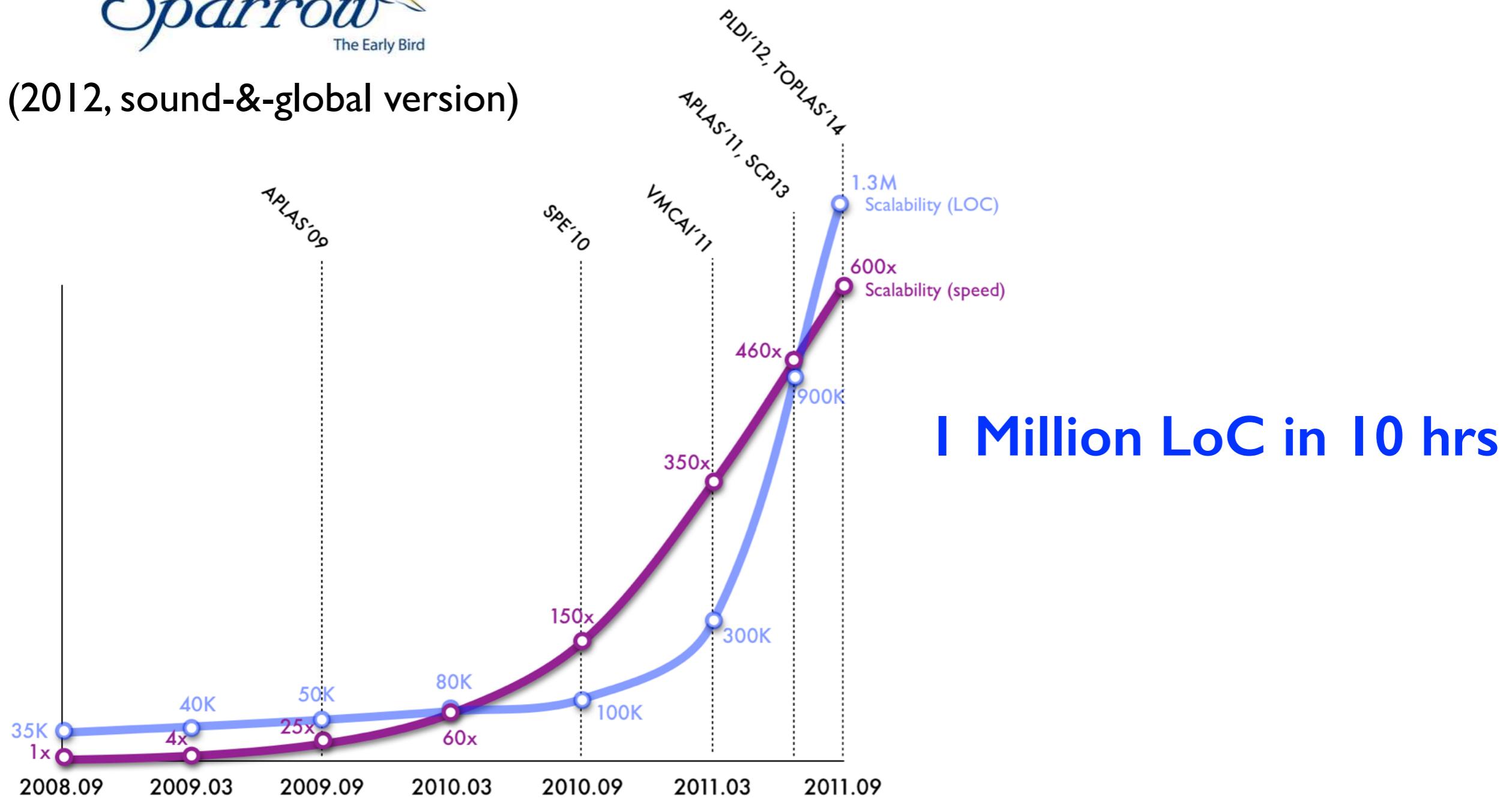
Scalability
General Sparse
Analysis Framework
[PLDI'12]



Scalability Improvement

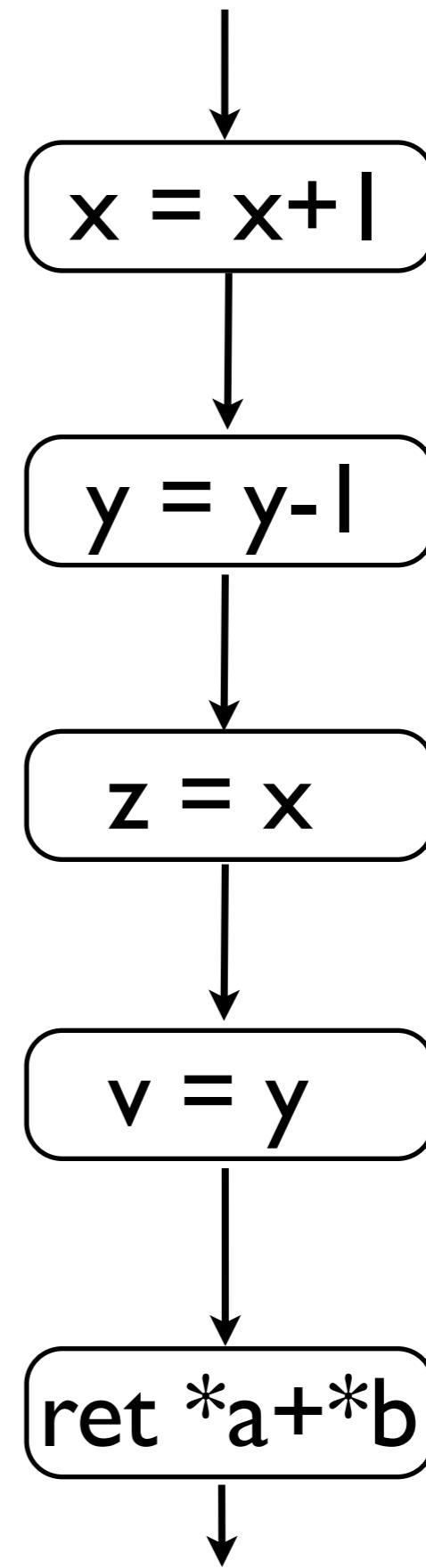


(2012, sound-&-global version)



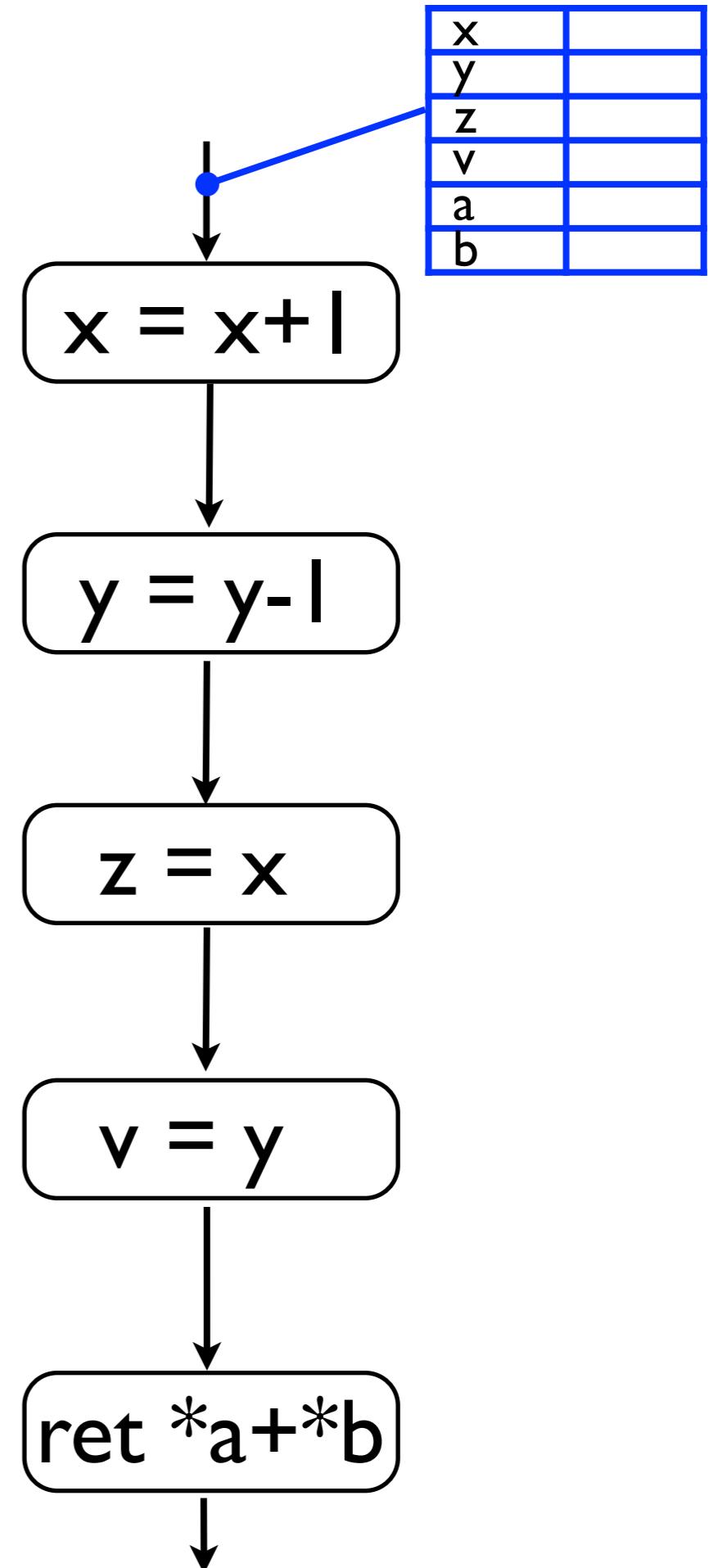
Key: General Sparse Analysis

“Right Part at Right Moment”



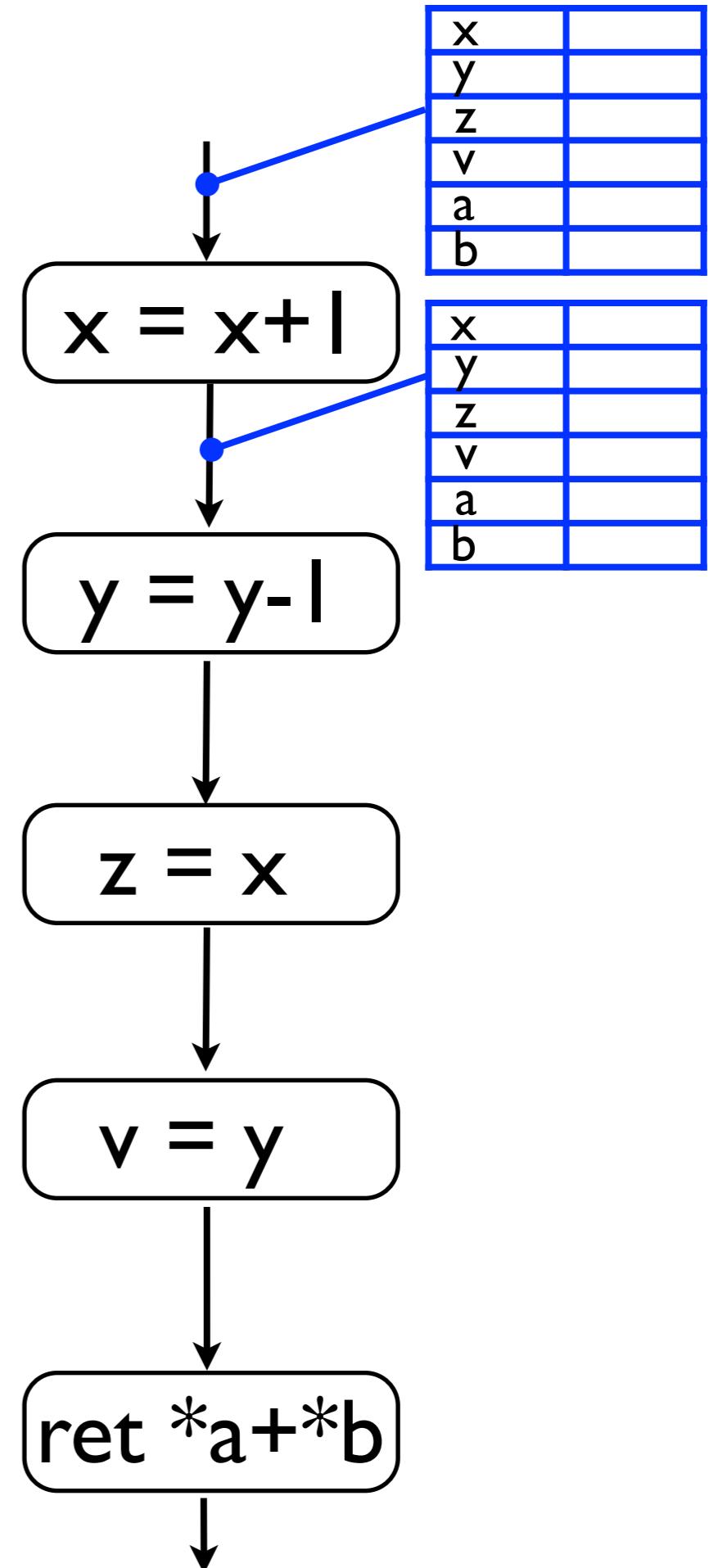
Key: General Sparse Analysis

“Right Part at Right Moment”



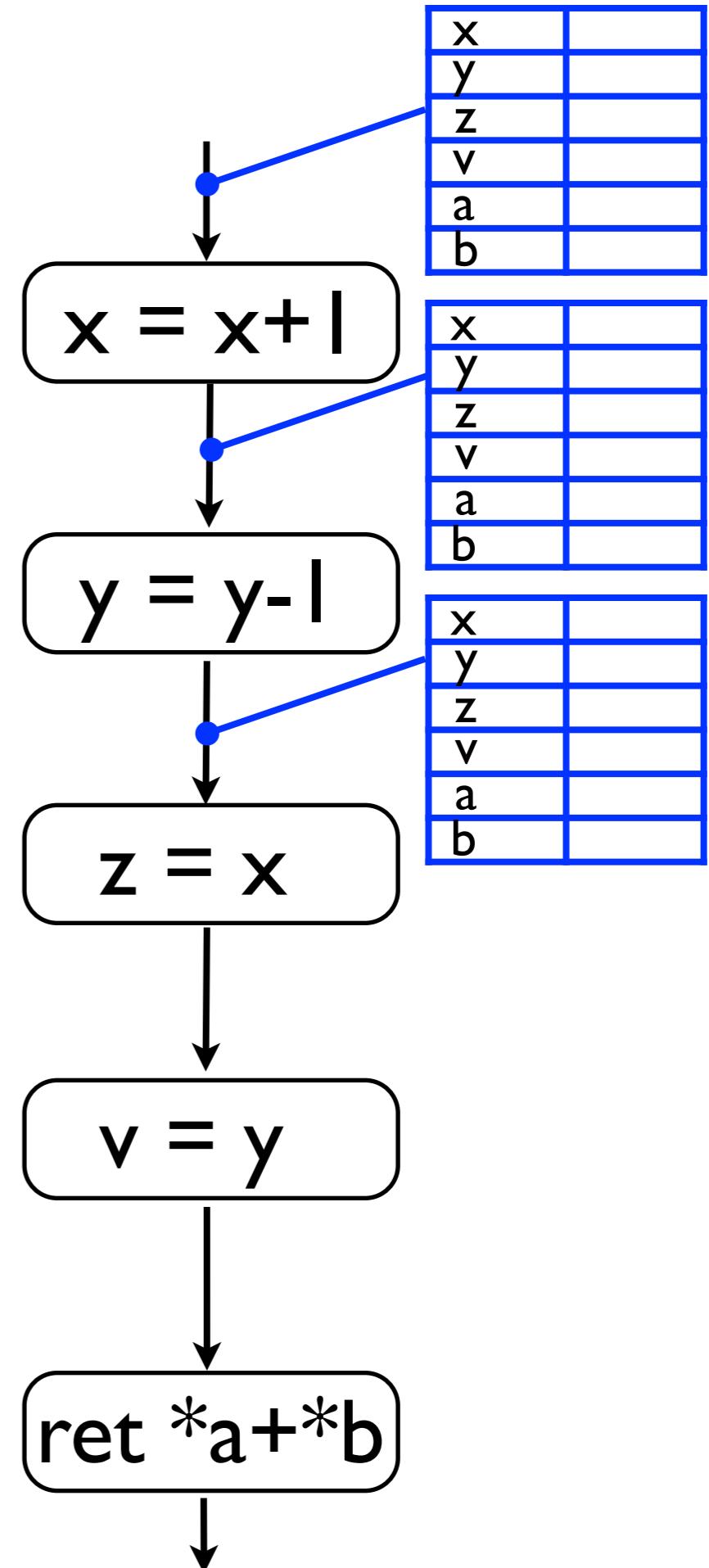
Key: General Sparse Analysis

“Right Part at Right Moment”



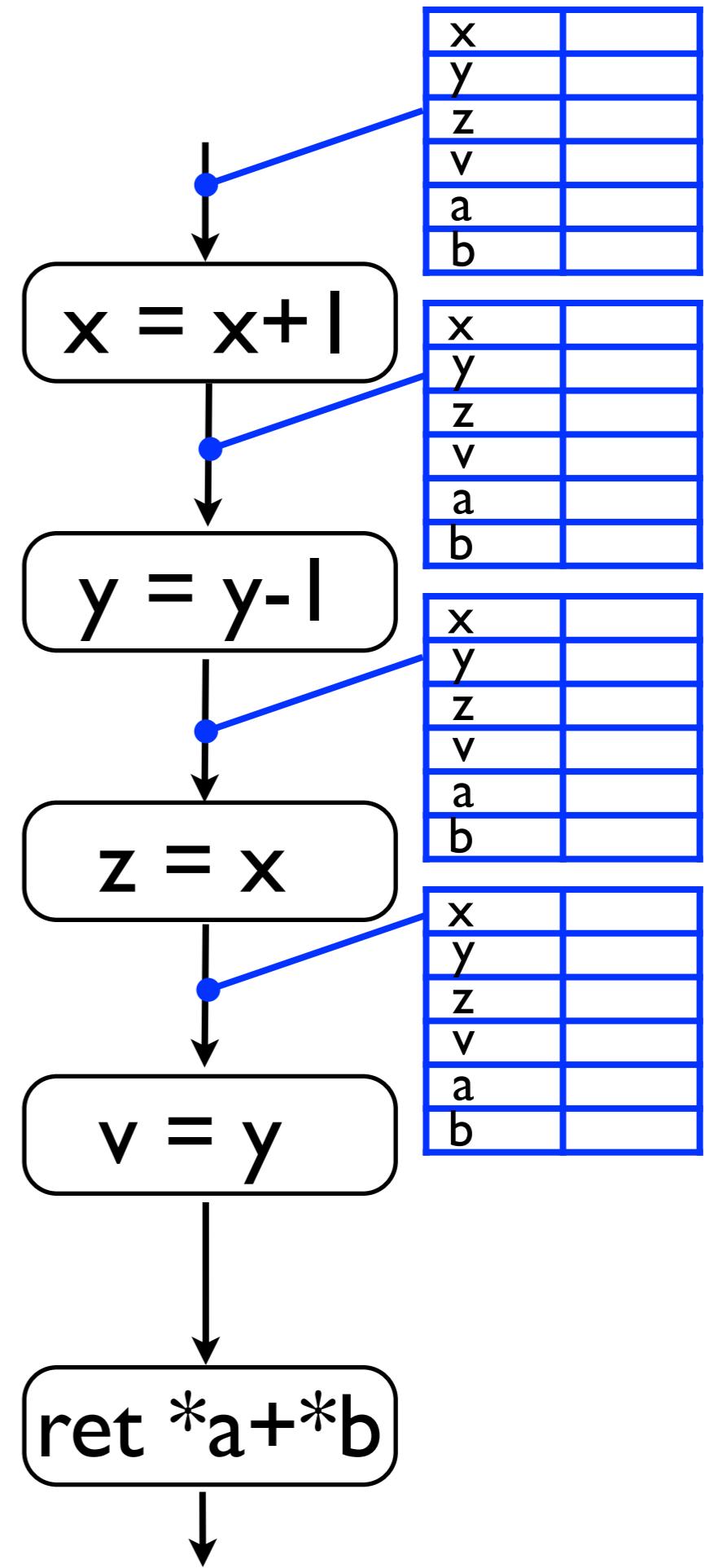
Key: General Sparse Analysis

“Right Part at Right Moment”



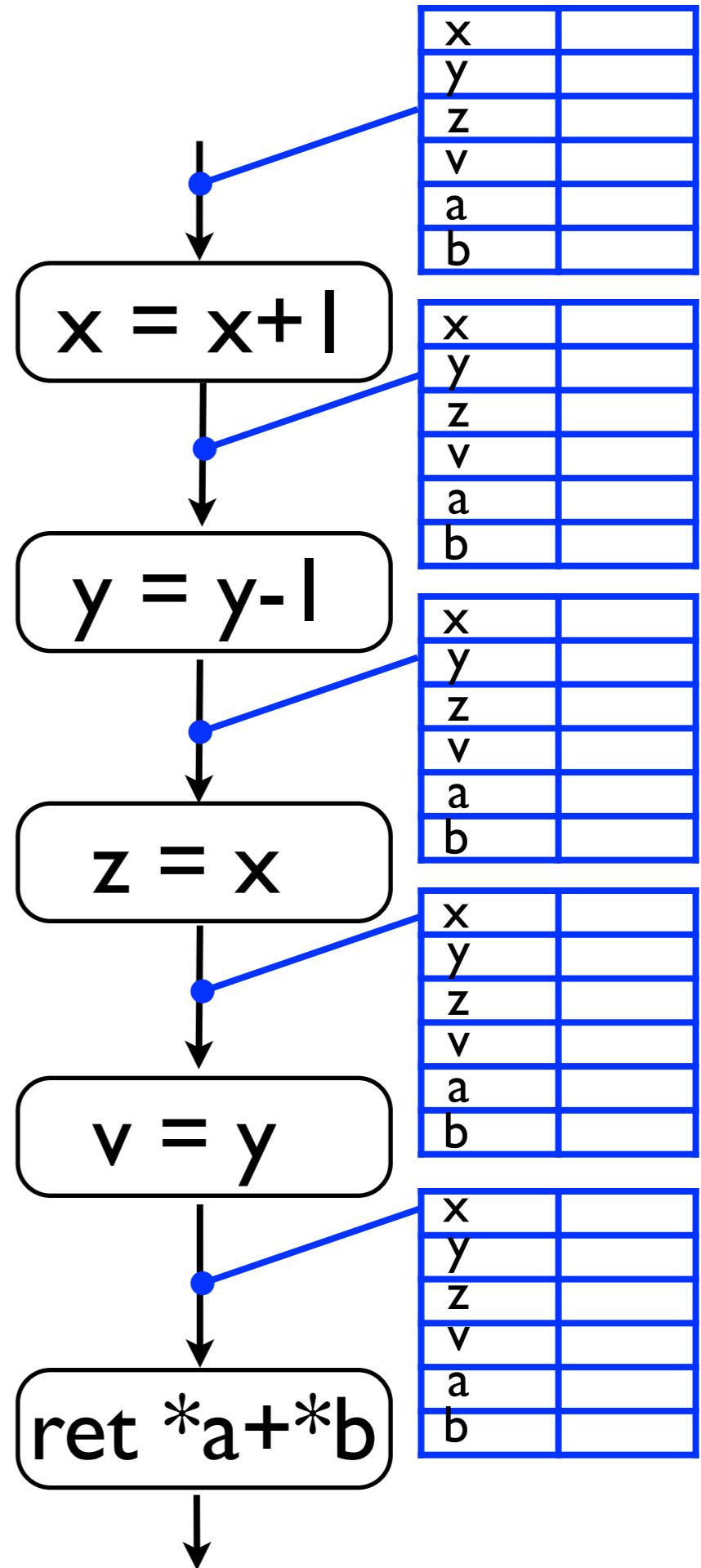
Key: General Sparse Analysis

“Right Part at Right Moment”



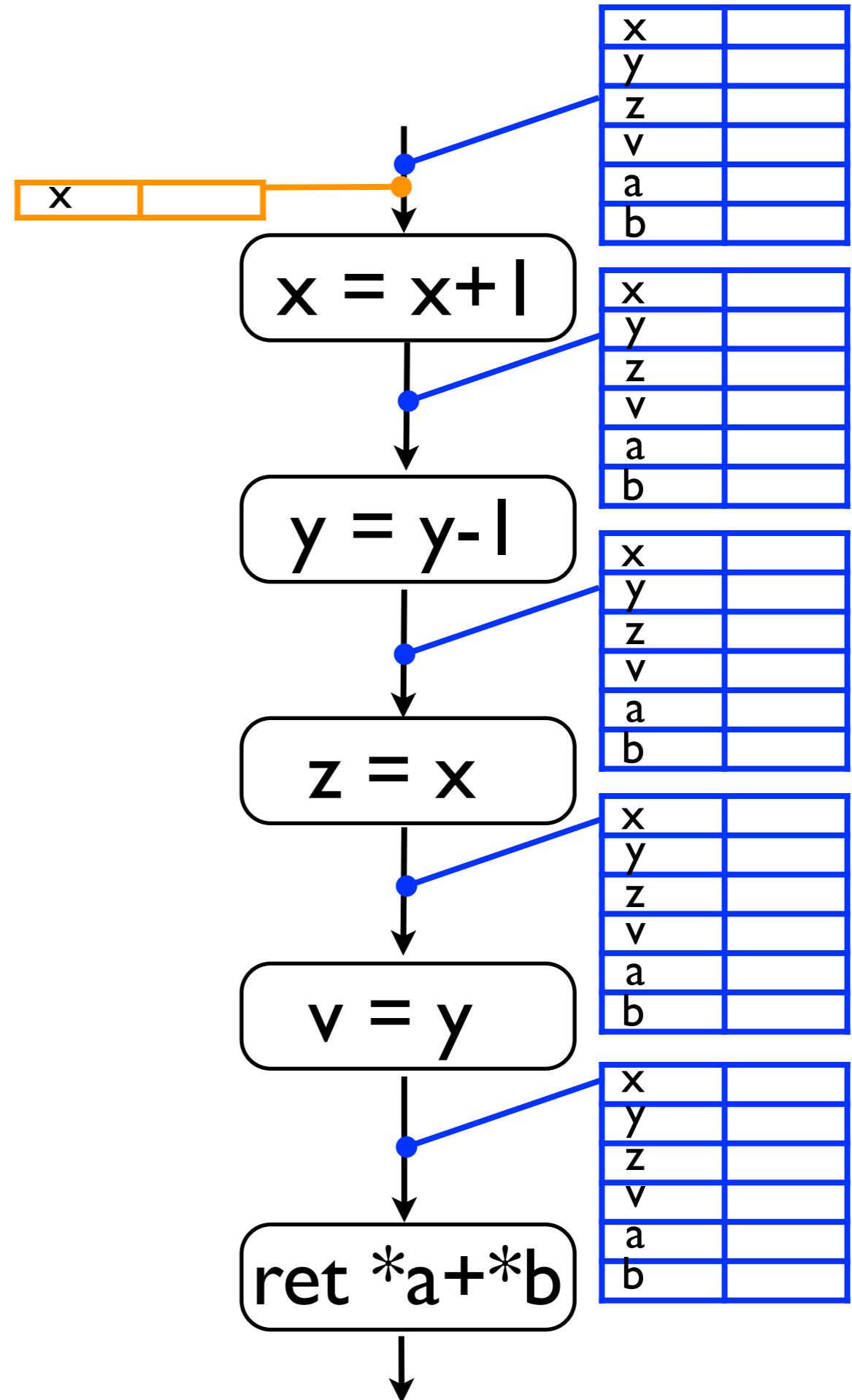
Key: General Sparse Analysis

“Right Part at Right Moment”



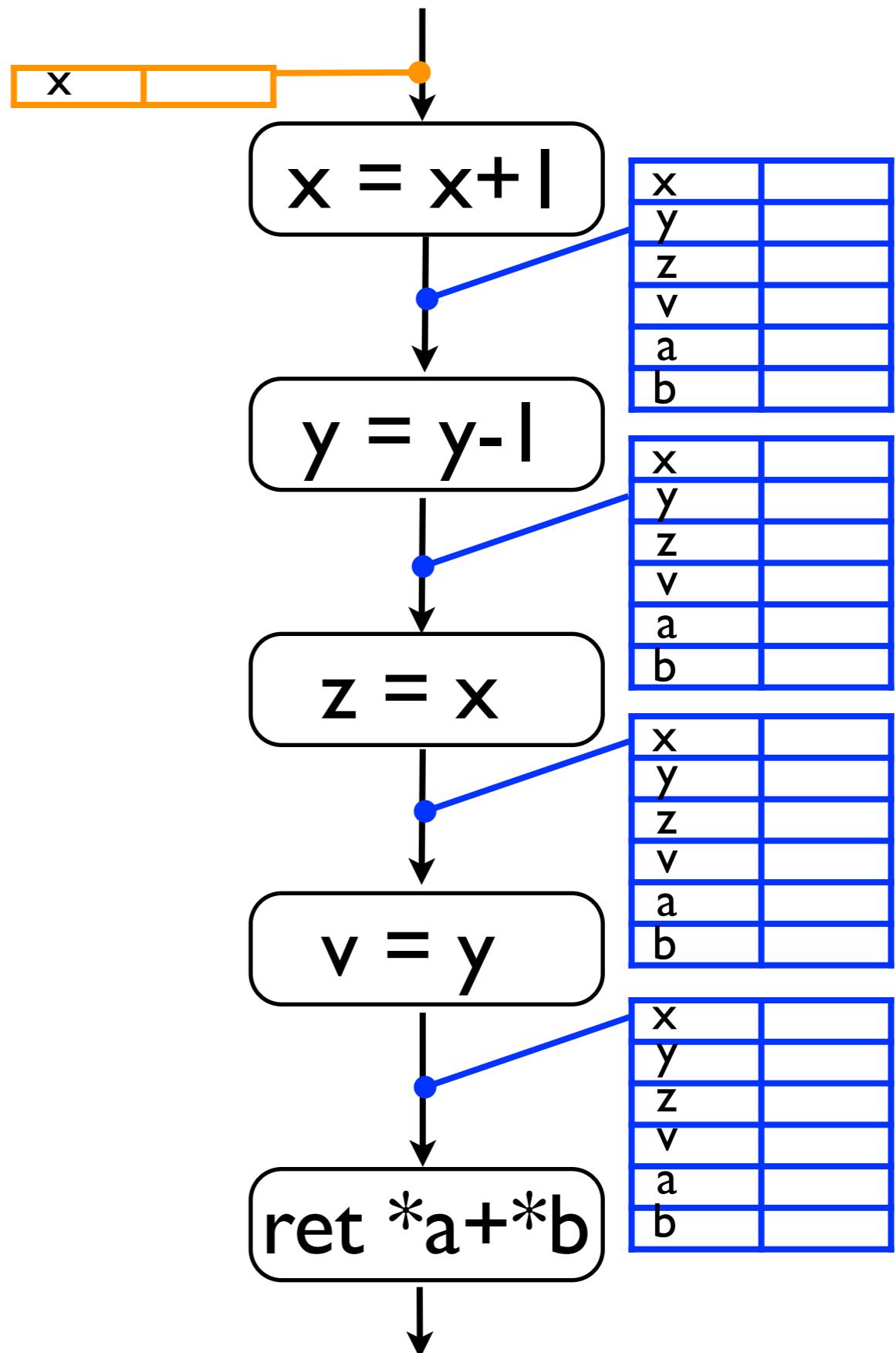
Key: General Sparse Analysis

“Right Part at Right Moment”



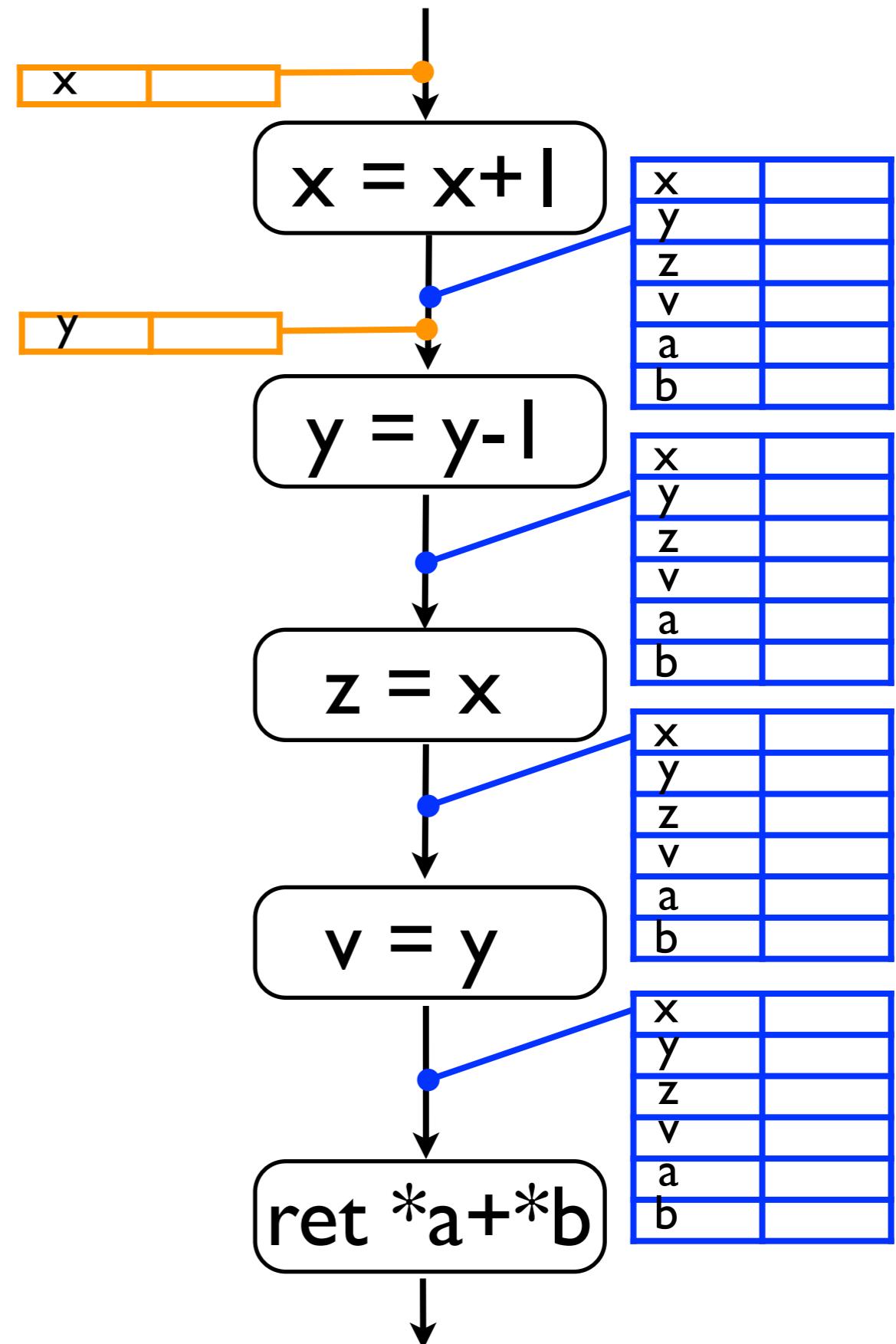
Key: General Sparse Analysis

“Right Part at Right Moment”



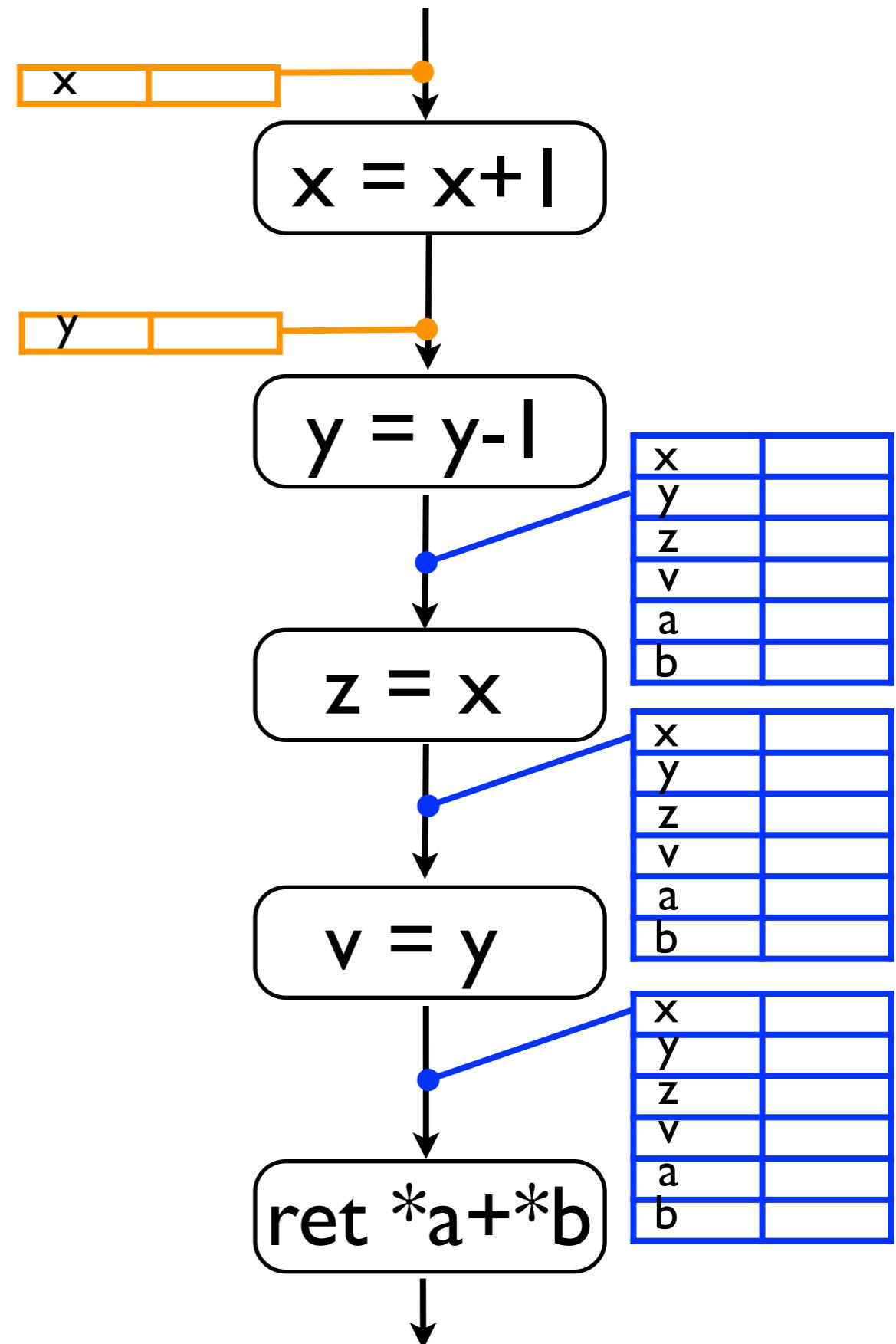
Key: General Sparse Analysis

“Right Part at Right Moment”



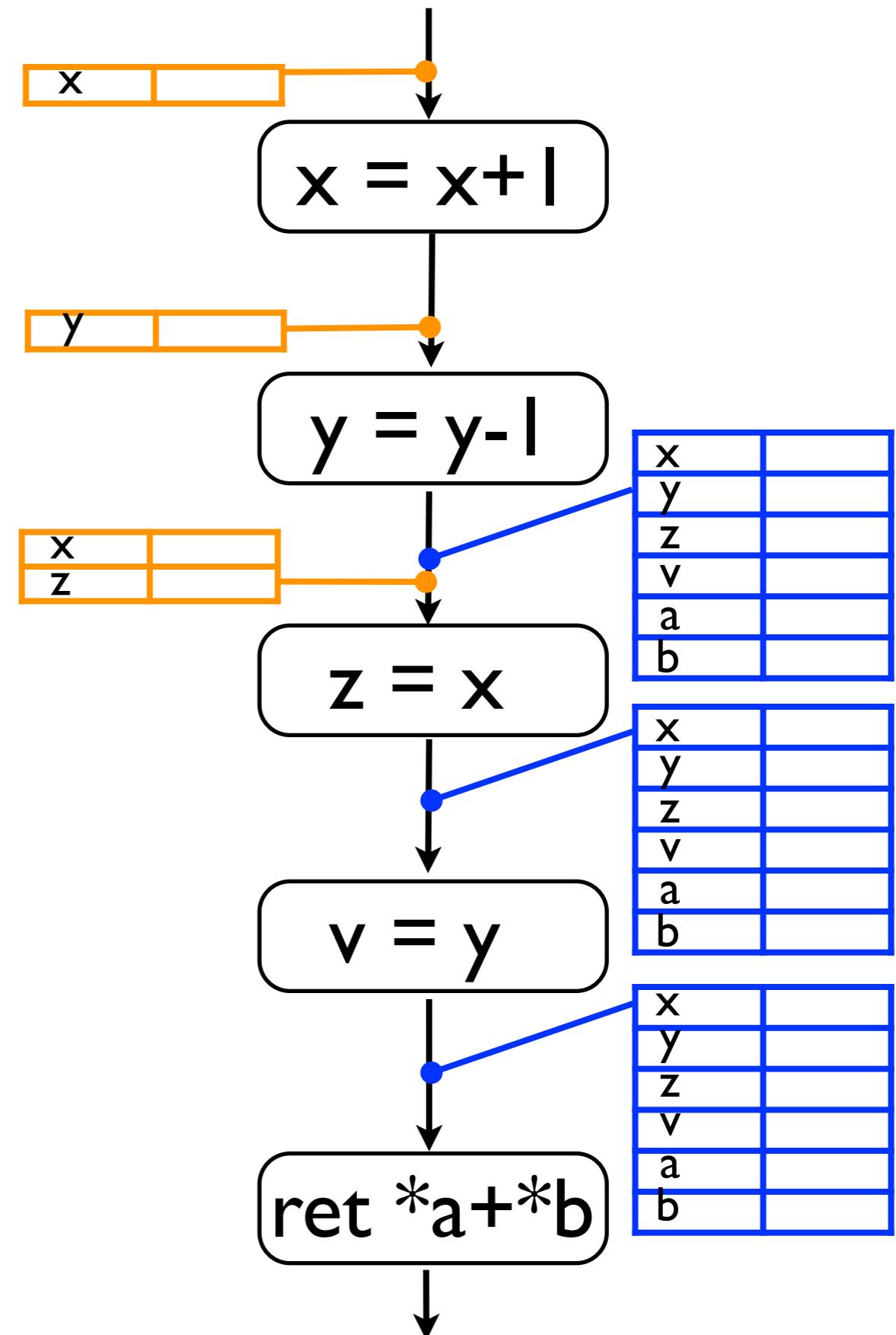
Key: General Sparse Analysis

“Right Part at Right Moment”



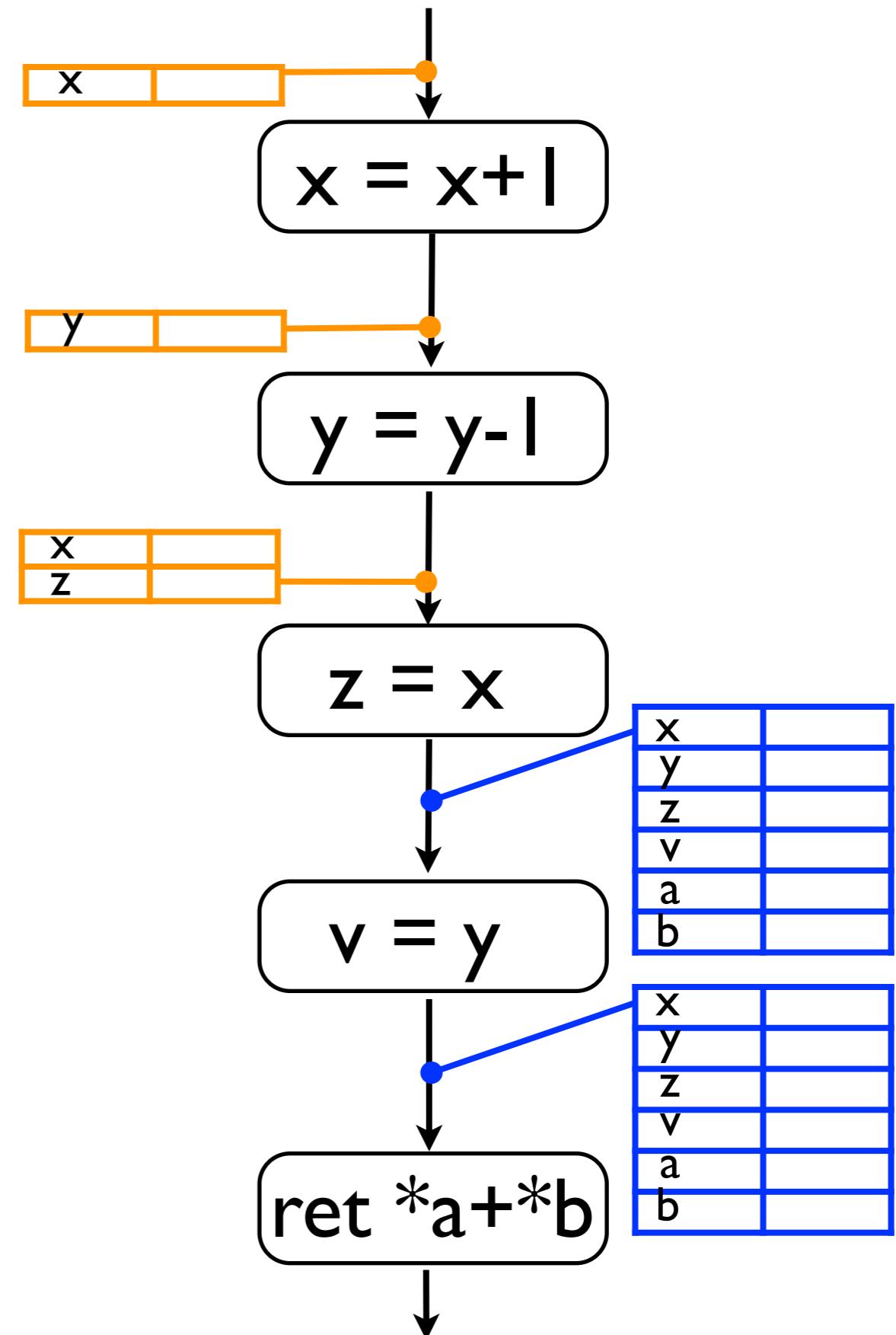
Key: General Sparse Analysis

“Right Part at Right Moment”



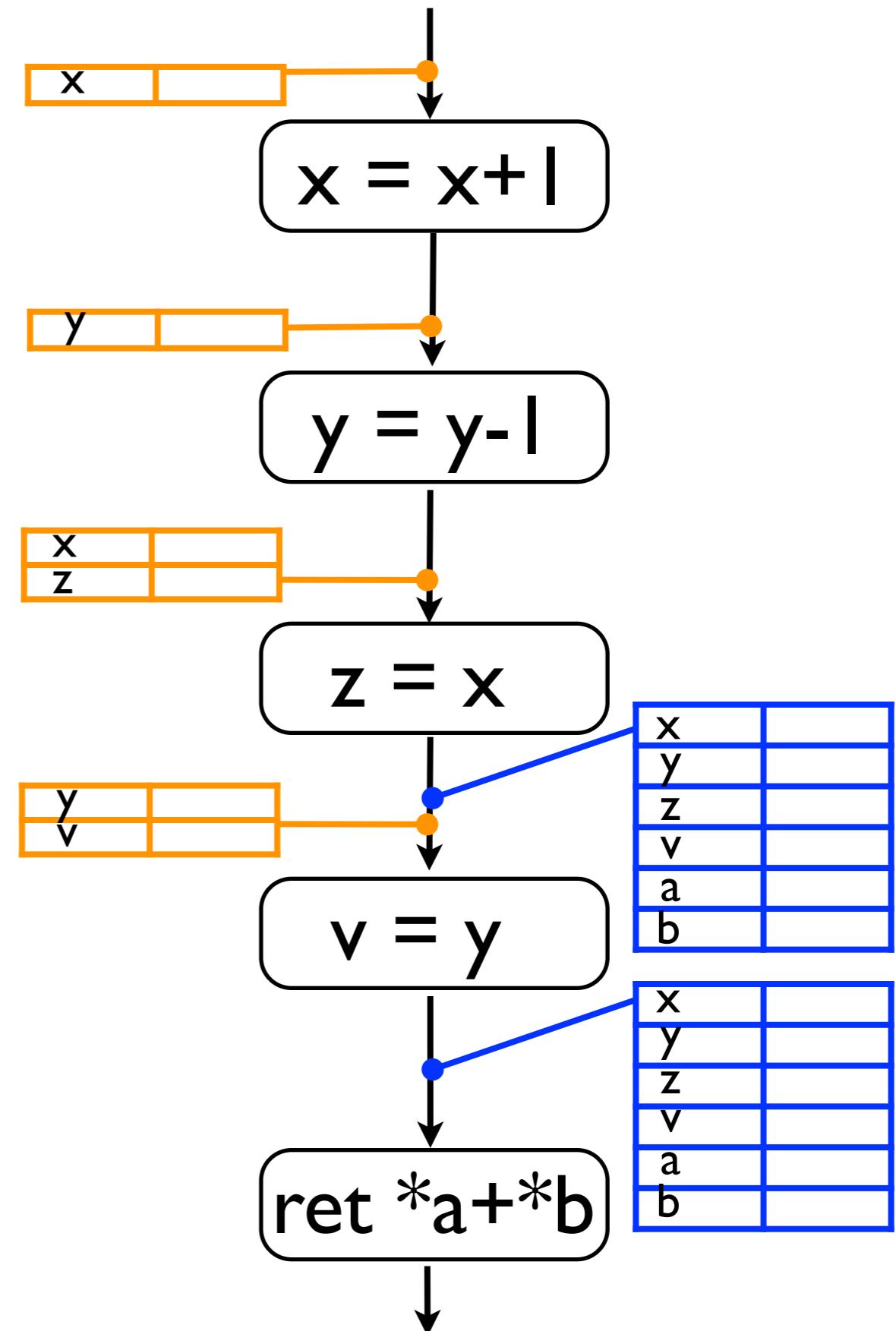
Key: General Sparse Analysis

“Right Part at Right Moment”



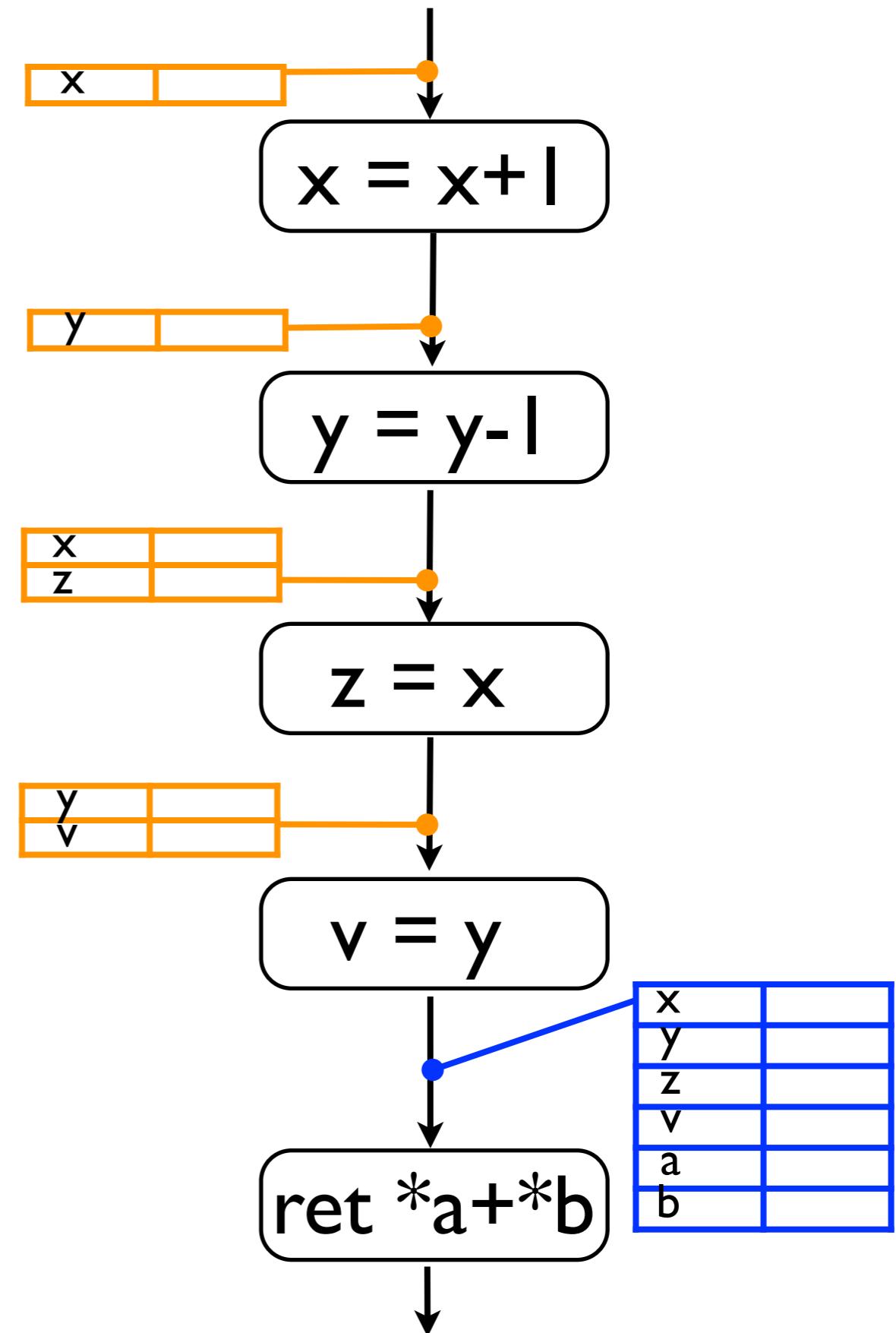
Key: General Sparse Analysis

“Right Part at Right Moment”



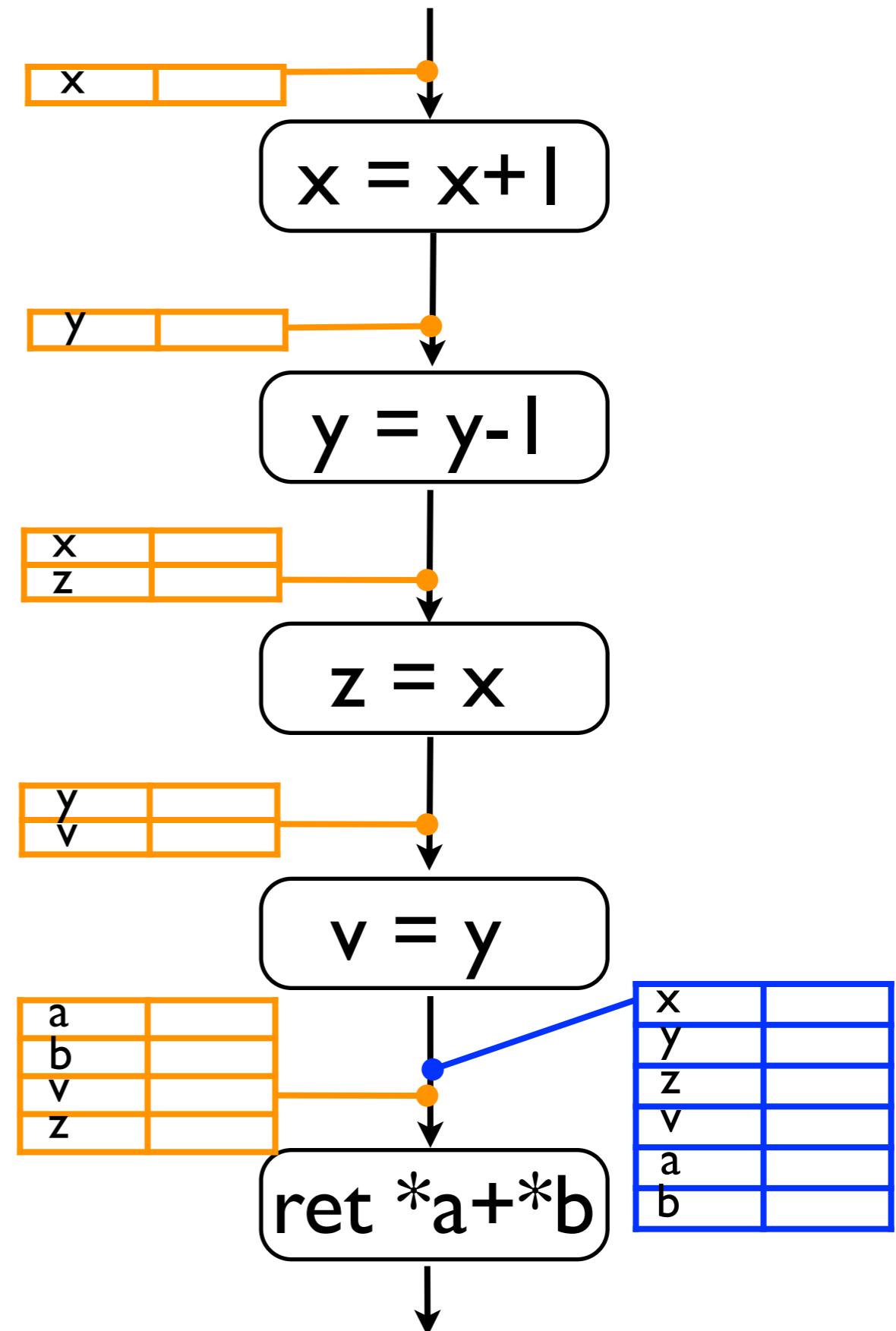
Key: General Sparse Analysis

“Right Part at Right Moment”



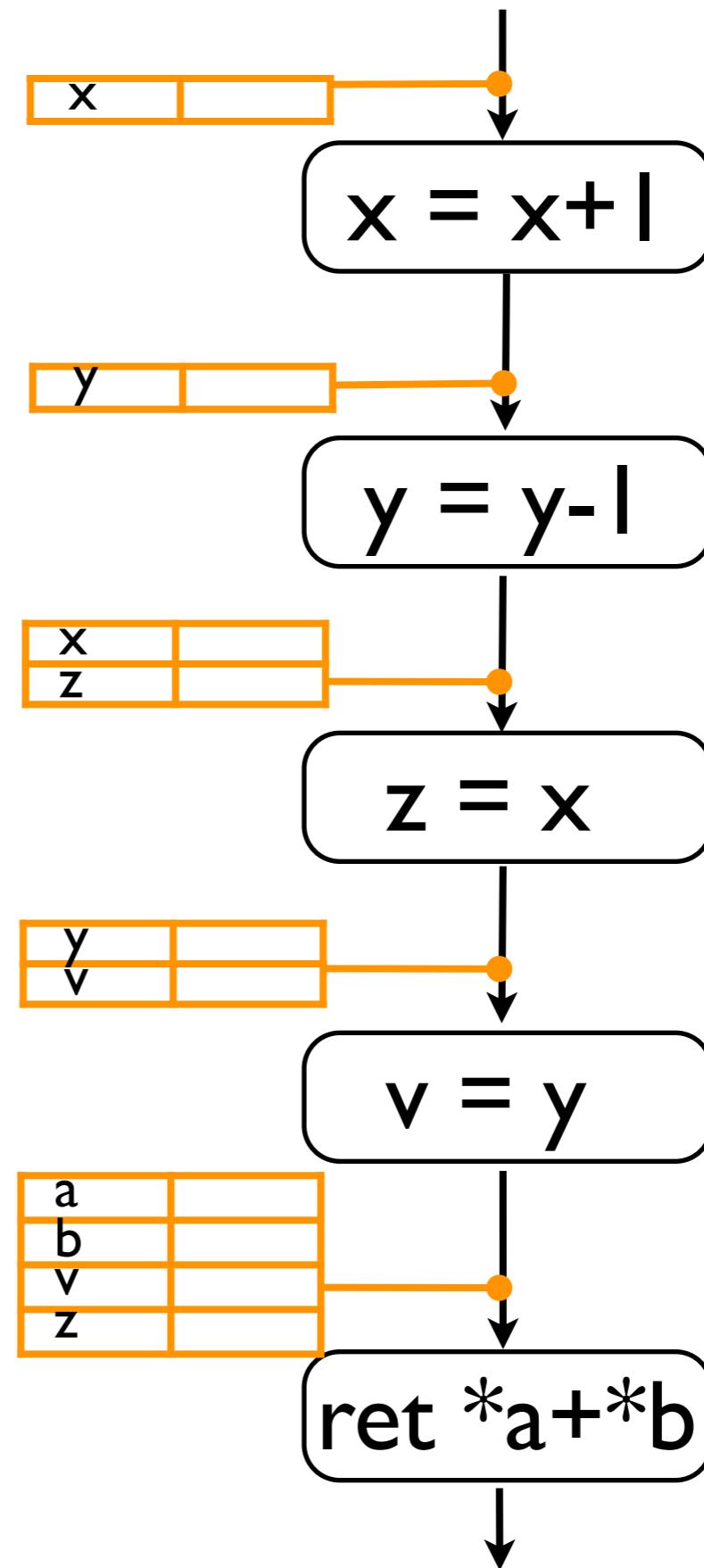
Key: General Sparse Analysis

“Right Part at Right Moment”



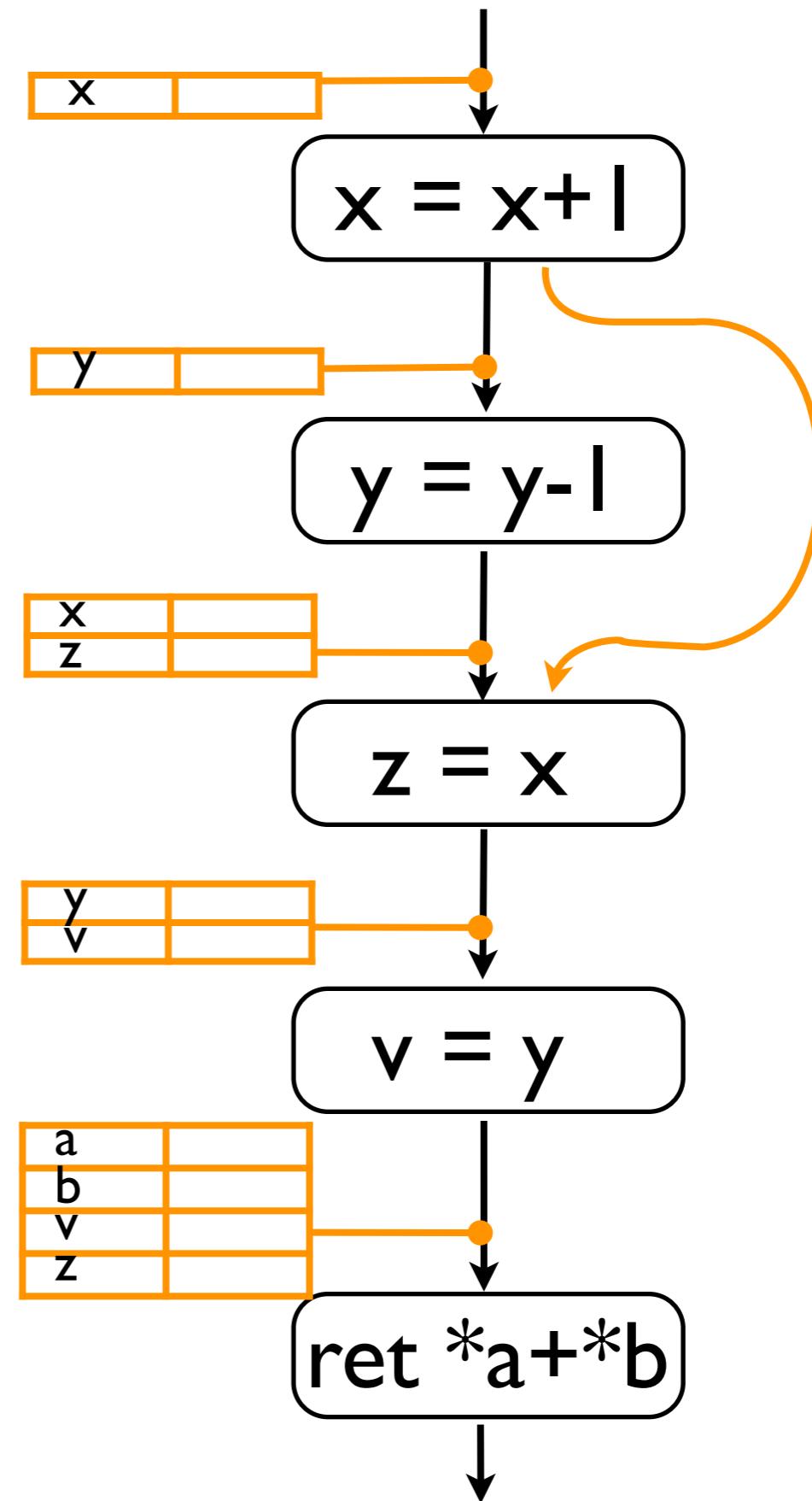
Key: General Sparse Analysis

“Right Part at Right Moment”



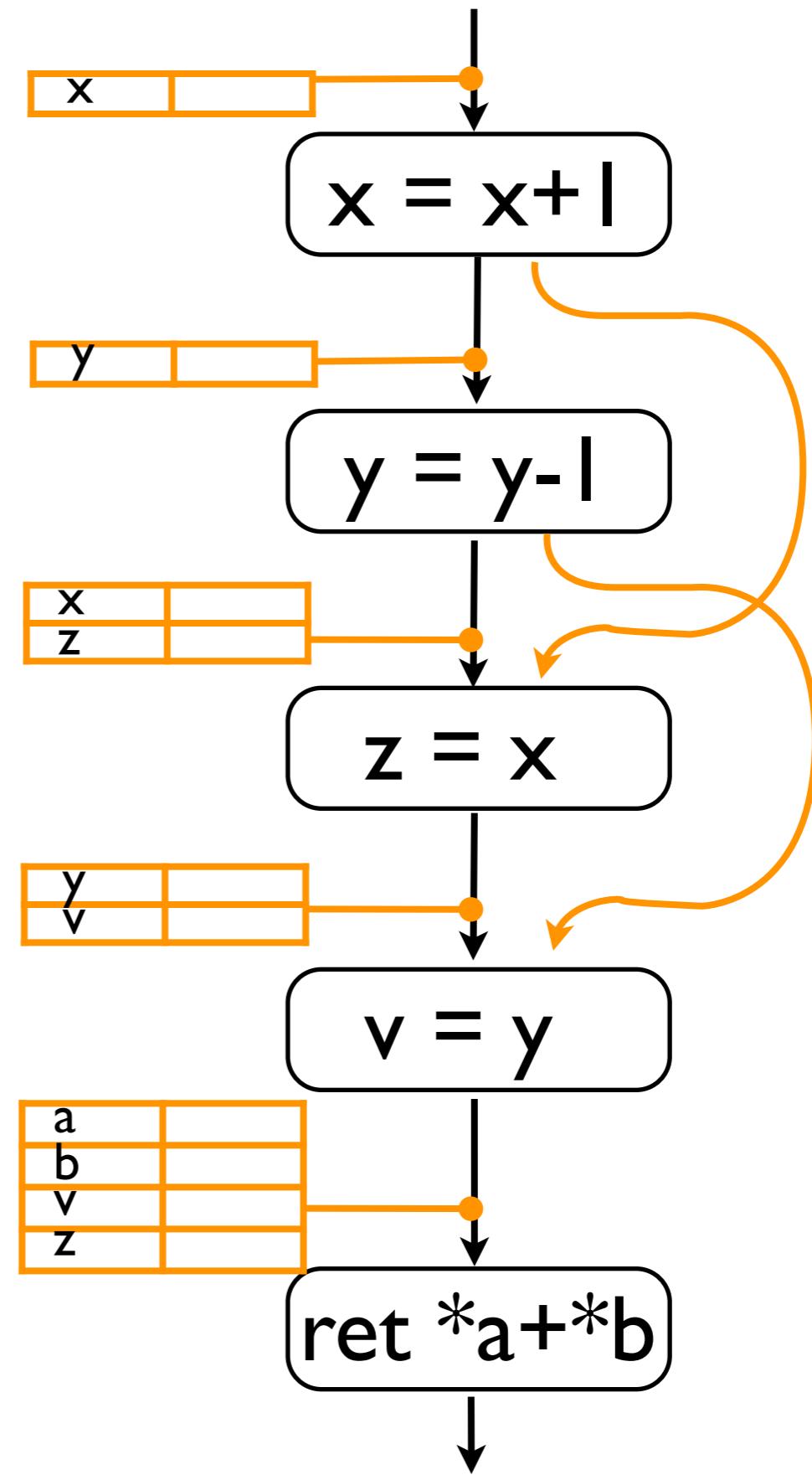
Key: General Sparse Analysis

“Right Part at Right Moment”



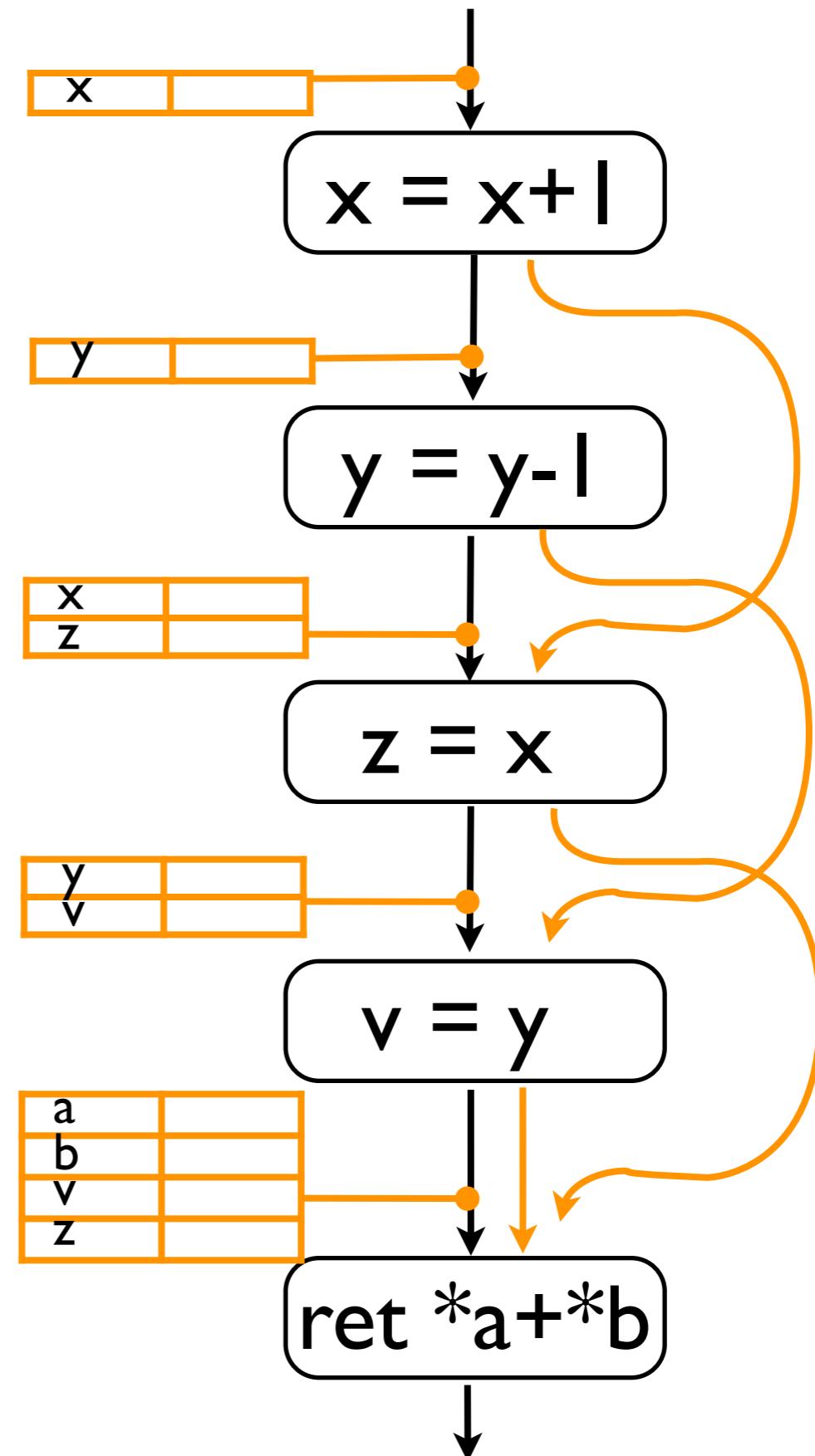
Key: General Sparse Analysis

“Right Part at Right Moment”



Key: General Sparse Analysis

“Right Part at Right Moment”



General Sparse Analysis Framework

Theorem. (preservation of soundness and precision)

$$\hat{F} : \hat{D} \rightarrow \hat{D} \xrightarrow{\text{sparsify}} \hat{F}_s : \hat{D} \rightarrow \hat{D}$$

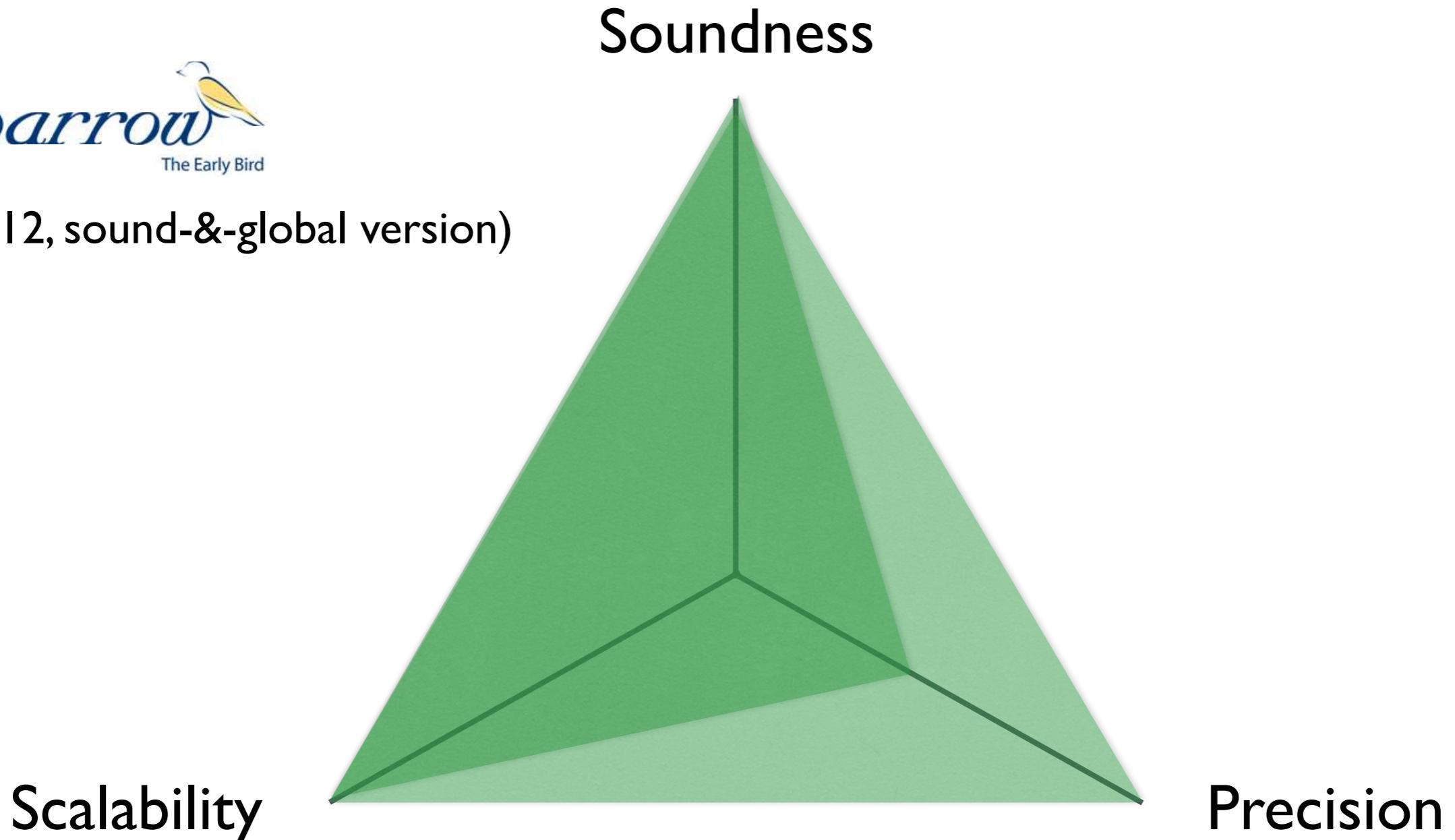
$$\text{fix } \hat{F} = \text{fix } \hat{F}_s$$

“An important strength is that the **theoretical result** is **very general** ... The result should be **highly influential** on future work in sparse analysis.” (from PLDI reviews)

The Second Goal: Precision



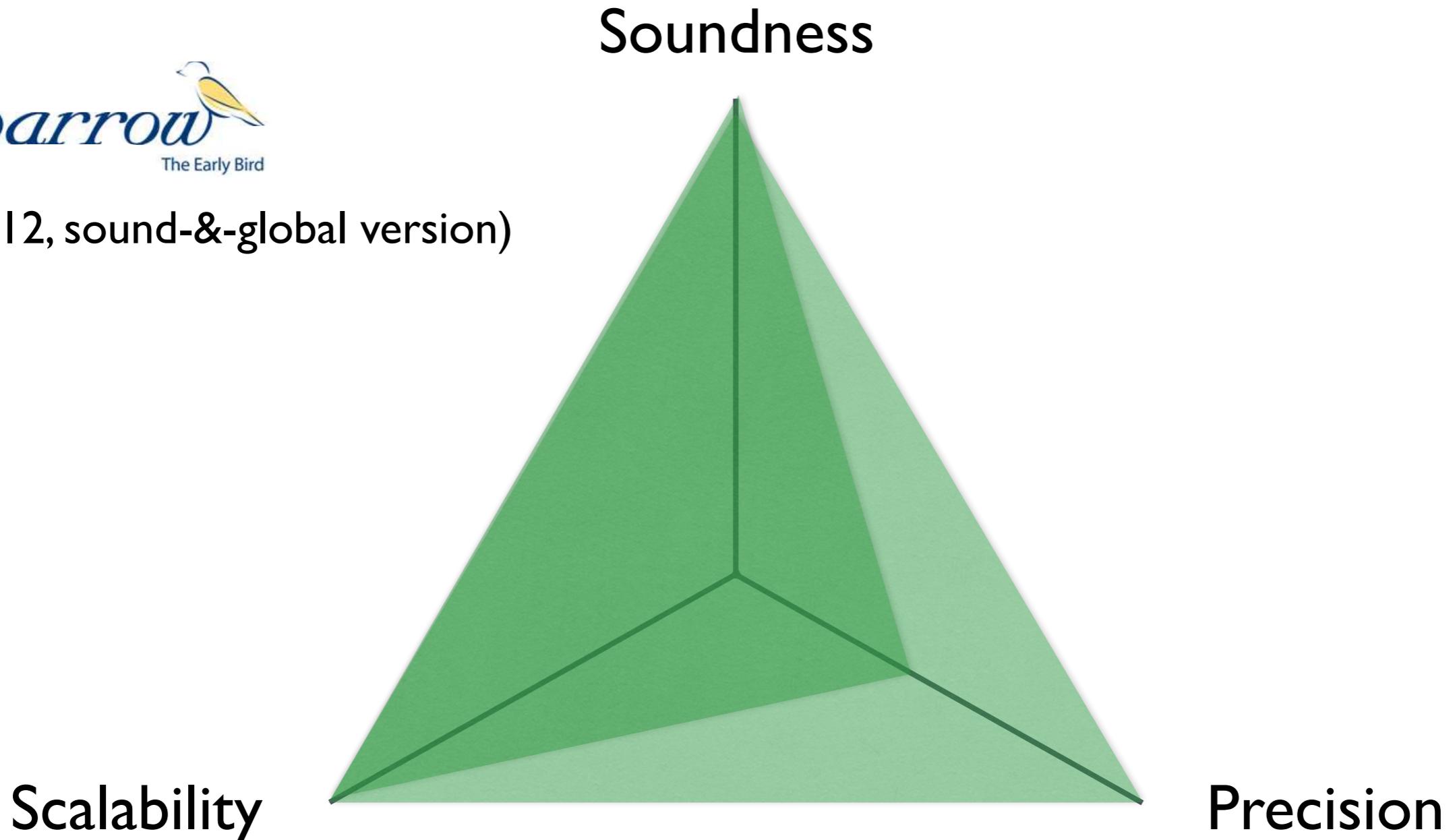
(2012, sound-&-global version)



The Second Goal: Precision

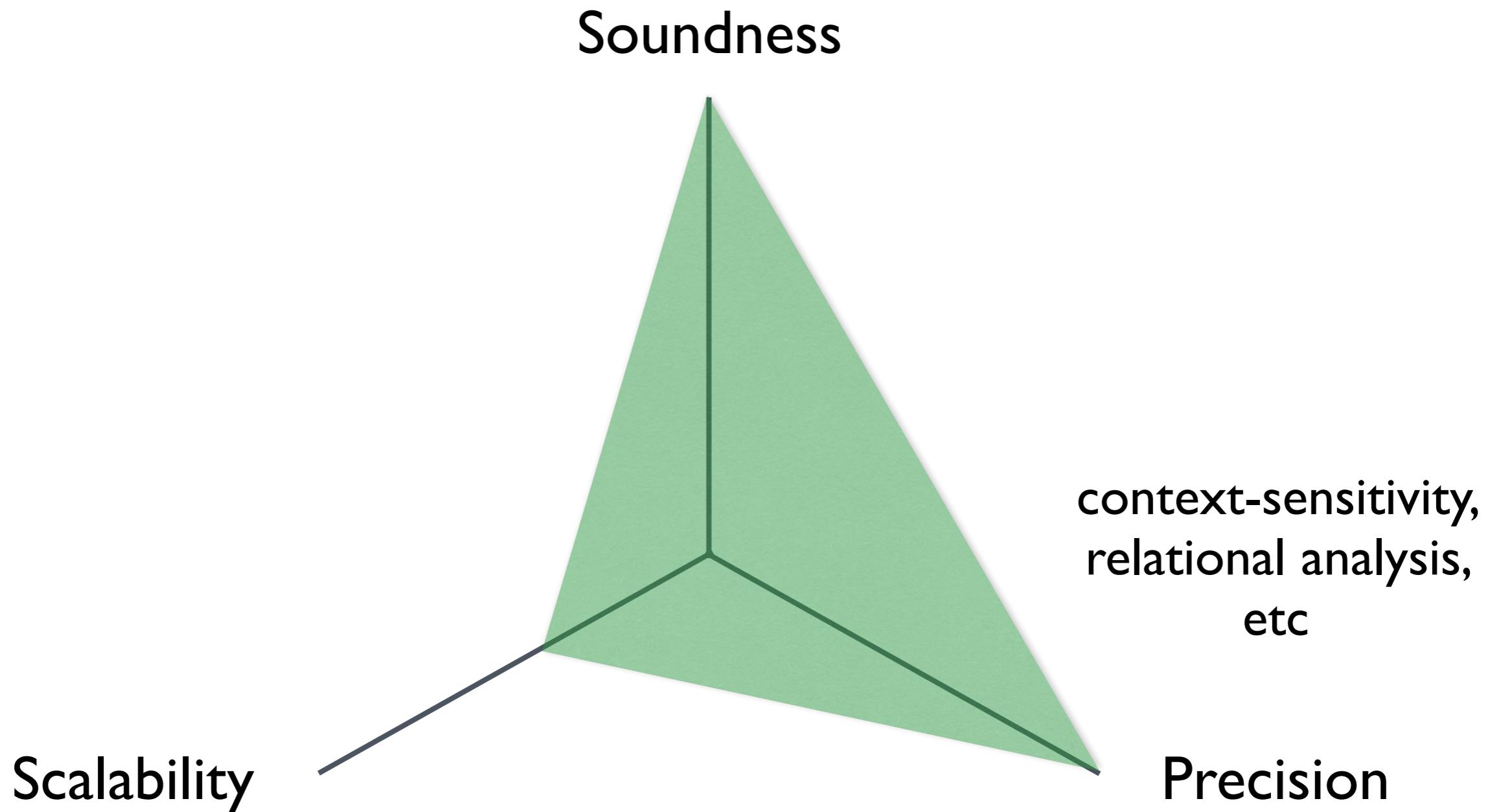


(2012, sound-&-global version)



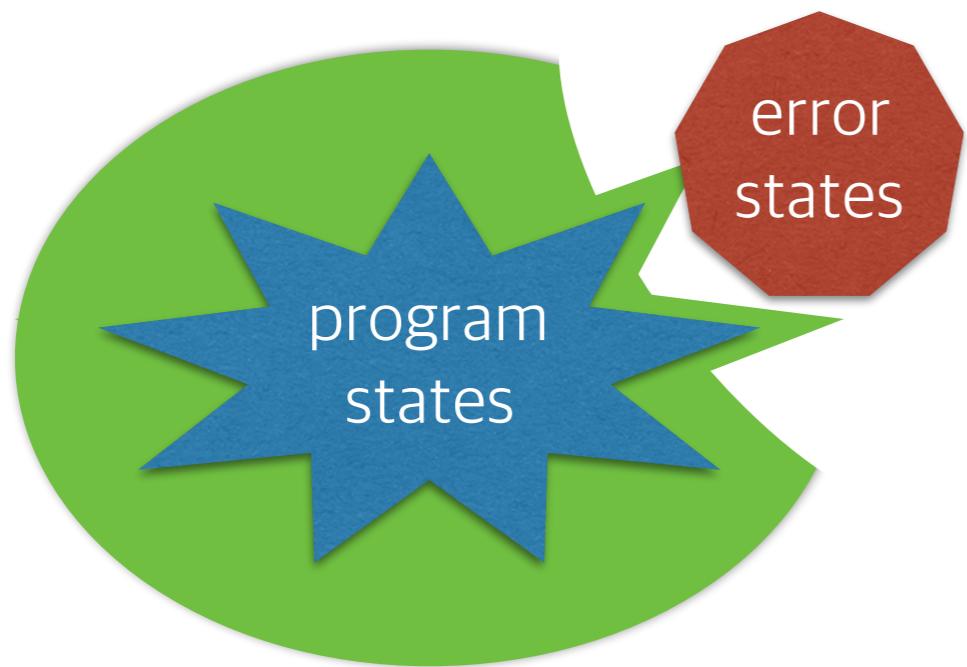
Challenge: Can we achieve it without scalability loss?

cf) Existing Techniques

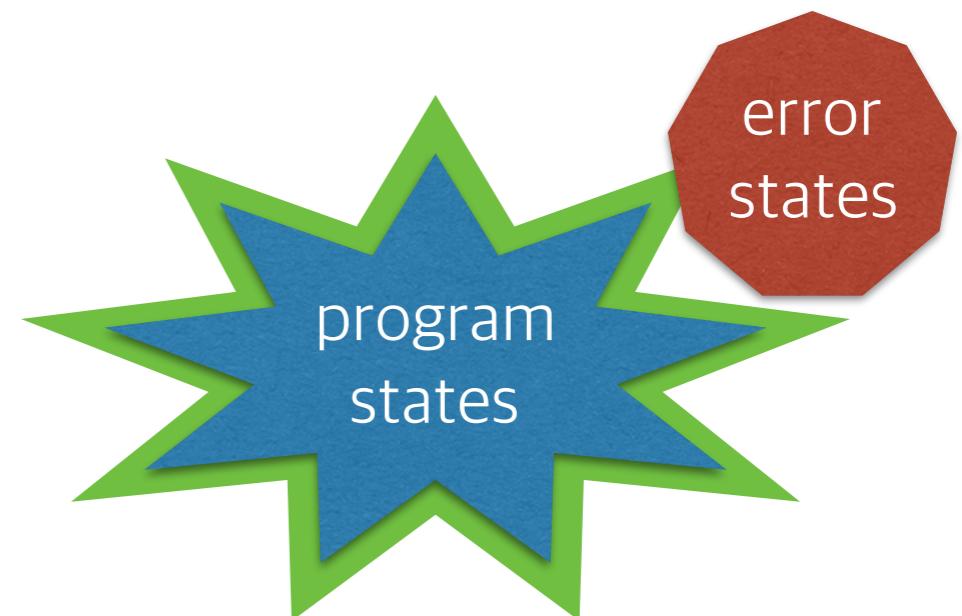


Selective X-Sensitivity Approach

- **Key Idea:** Improve precision only when it matters



vs.

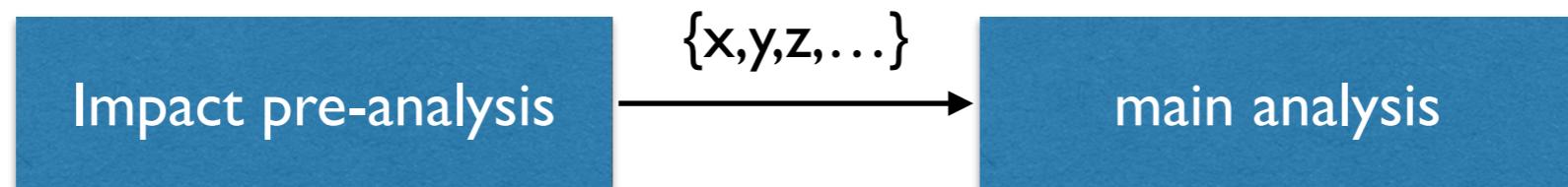


ours

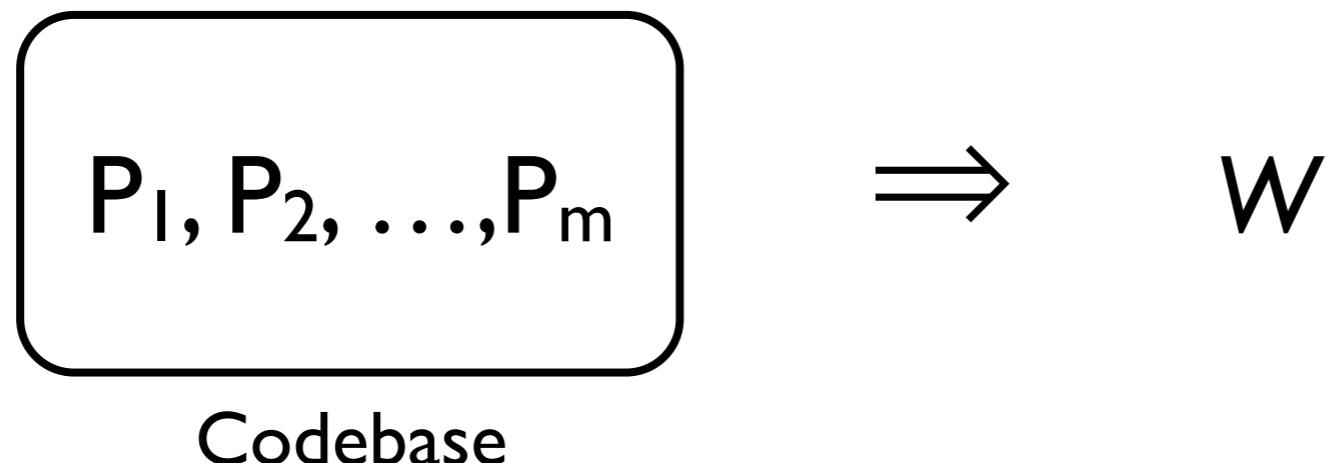
existing techniques

Selection Strategy

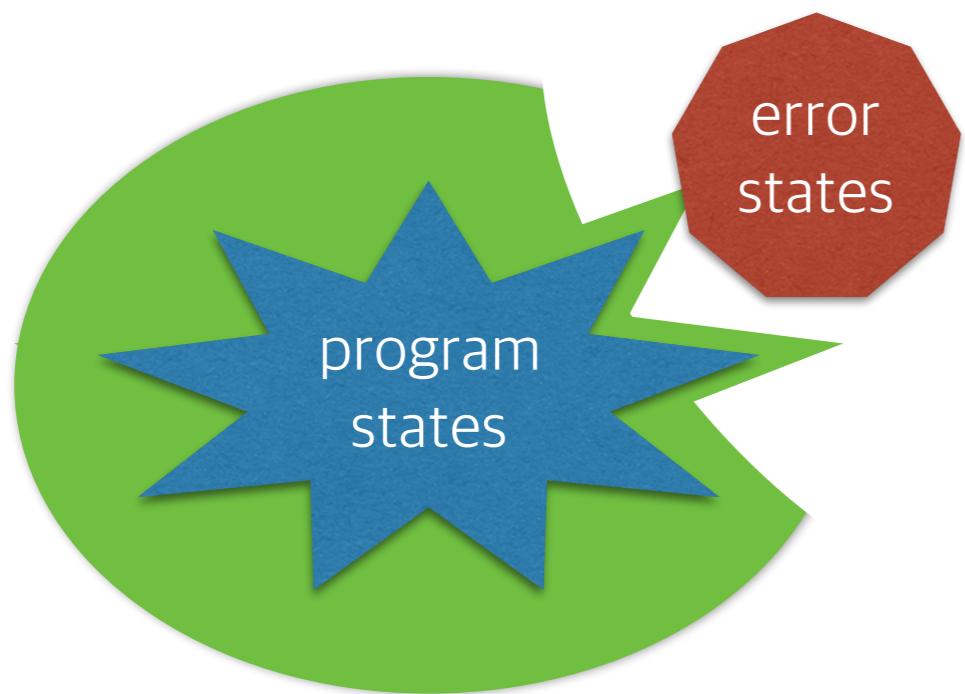
- Impact pre-analysis [PLDI'14]:



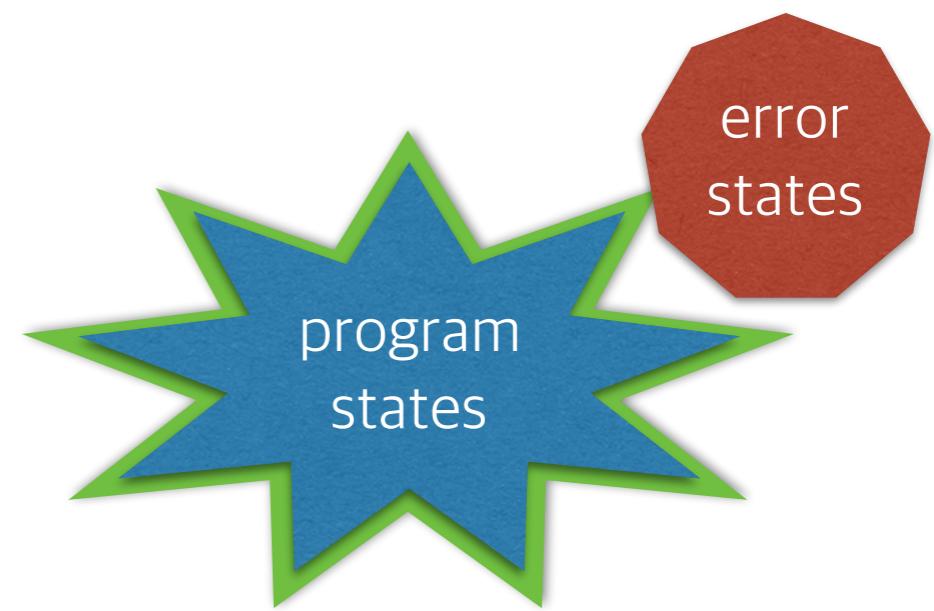
- Learning from codebase [OOPSLA'15]:



Effectiveness



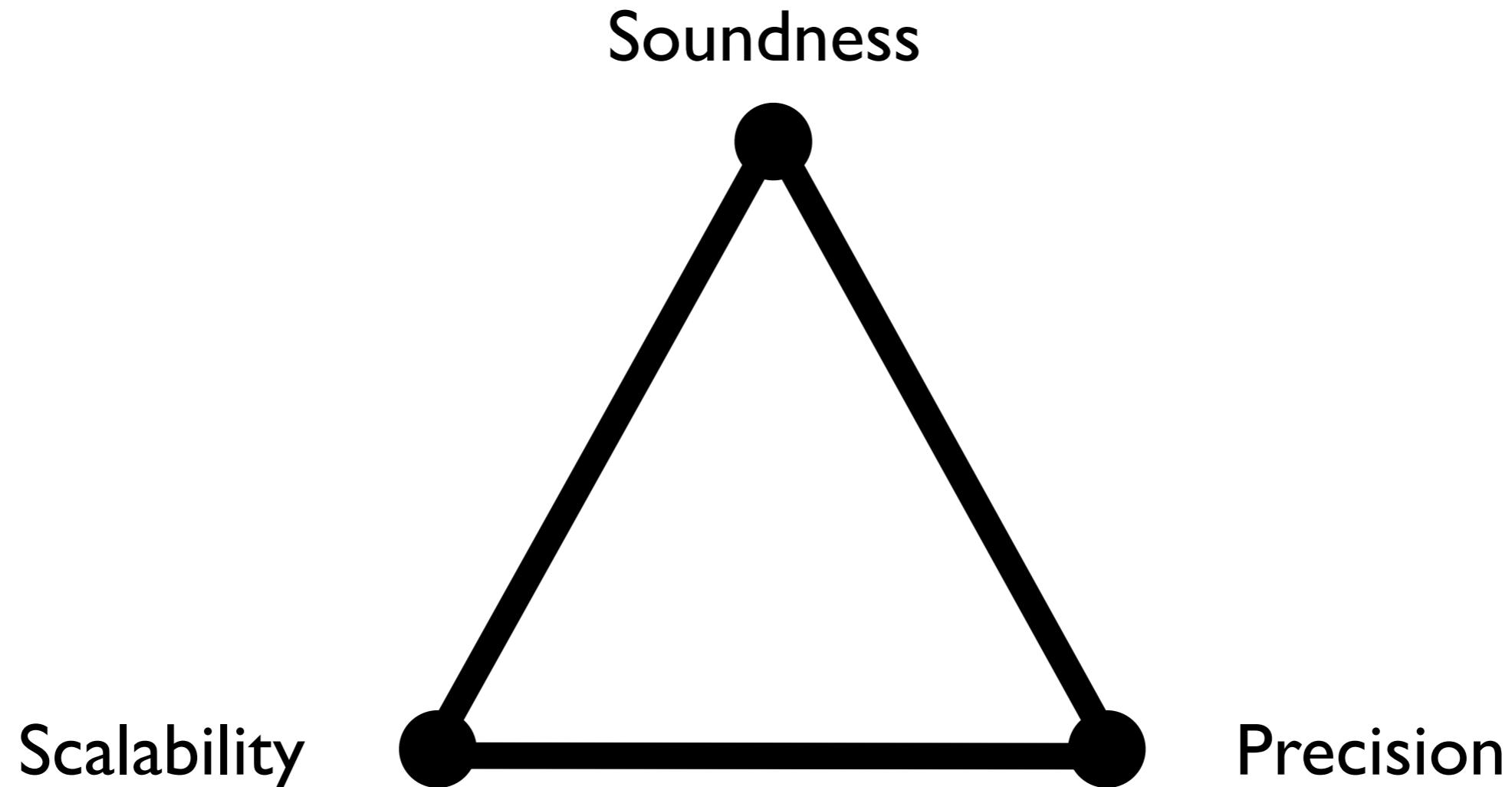
vs.



+25% / -25%

+25% / -1300%

Enabled Powerful Static Analysis



**General Sparse
Analysis Framework**
[PLDI'12]

**Selective X-Sensitivity
Framework**
[PLDI'14]

Static Analysis for Verification

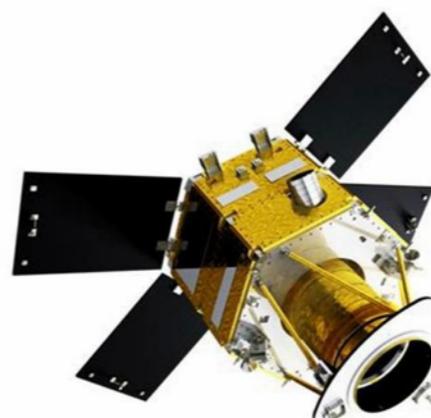
Safety-critical softwares



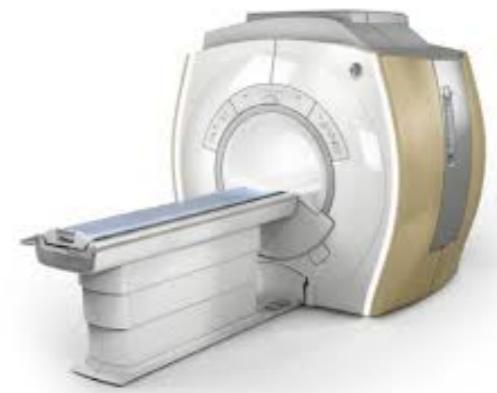
Red Hawk



Hubo



Deimos-2

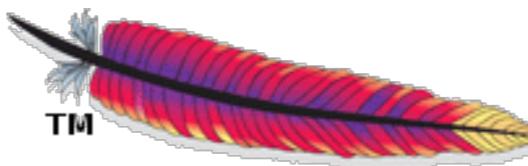


KAIST PET-MRI

- Static verifier for flight SW
- Static verifier for robot SW
- Static verifier for satellite SW, etc

Static Analysis for Security

Security-critical softwares



Massive Security Bug In OpenSSL Could Affect A Huge Chunk Of The Internet

Posted Apr 7, 2014 by Greg Kumparak (@grg)

122 Like 14k Share 1,152 Tweet 1,490



I saw a t-shirt one time. "I'm a bomb disposal technician," it read. "If you see me running, to keep up."

The same sort of idea can be applied to net security: when all the net security people you know are freaking out, it's probably an okay time to worry.

This afternoon, many of the net security people I know are freaking out. A very serious bug in OpenSSL — a cryptographic library that

Sendmail disasters

These are the most serious sendmail security and reliability problems through sendmail 8.8.7 in 1997. Unattributed quotes here are from All

- Security verifiers for OpenSSL, Apache, Sendmail, etc

Static Analysis for Modern Computing Platforms

- Mobile
- Cloud
- Parallel
- Wearable
- ...

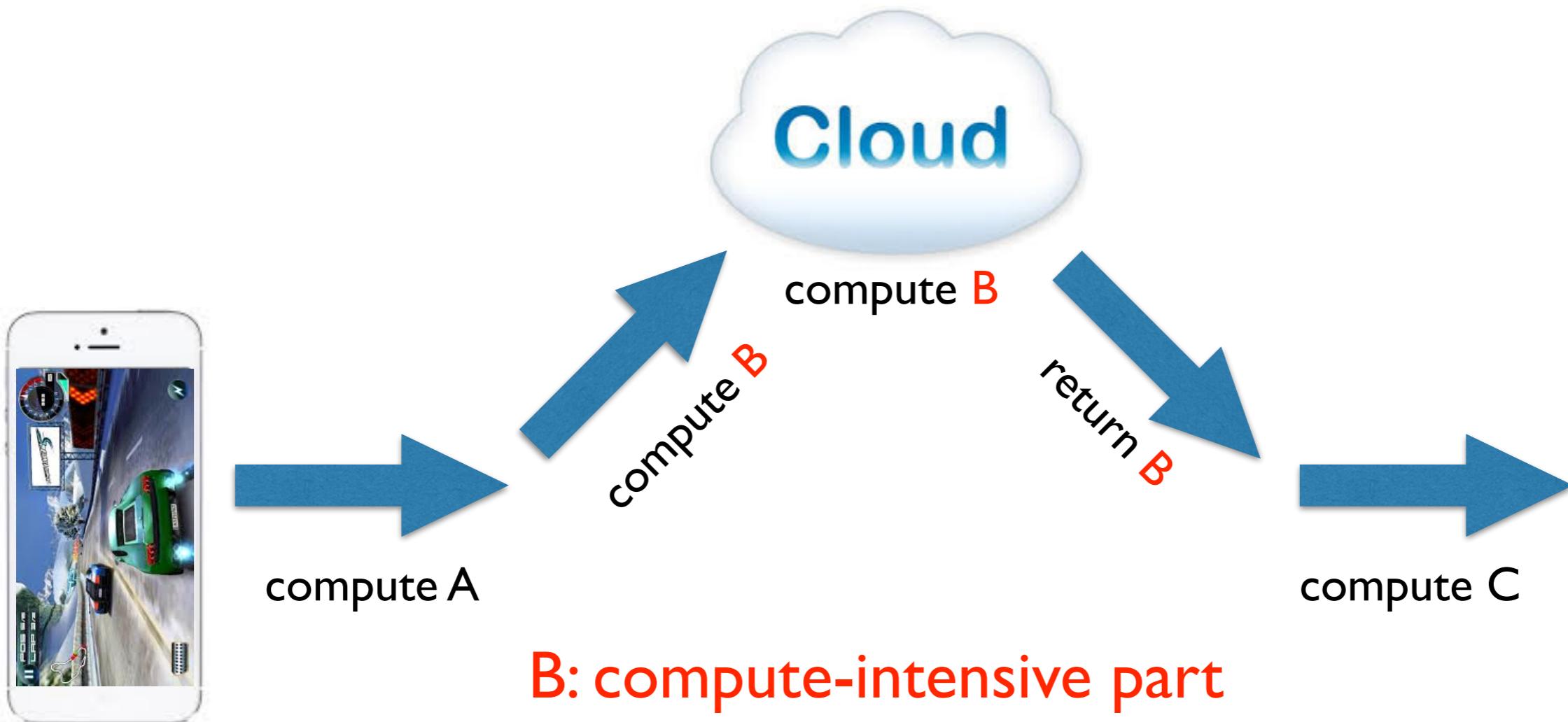


- New Software challenges:
e.g., reliability, energy-efficiency, security, ...

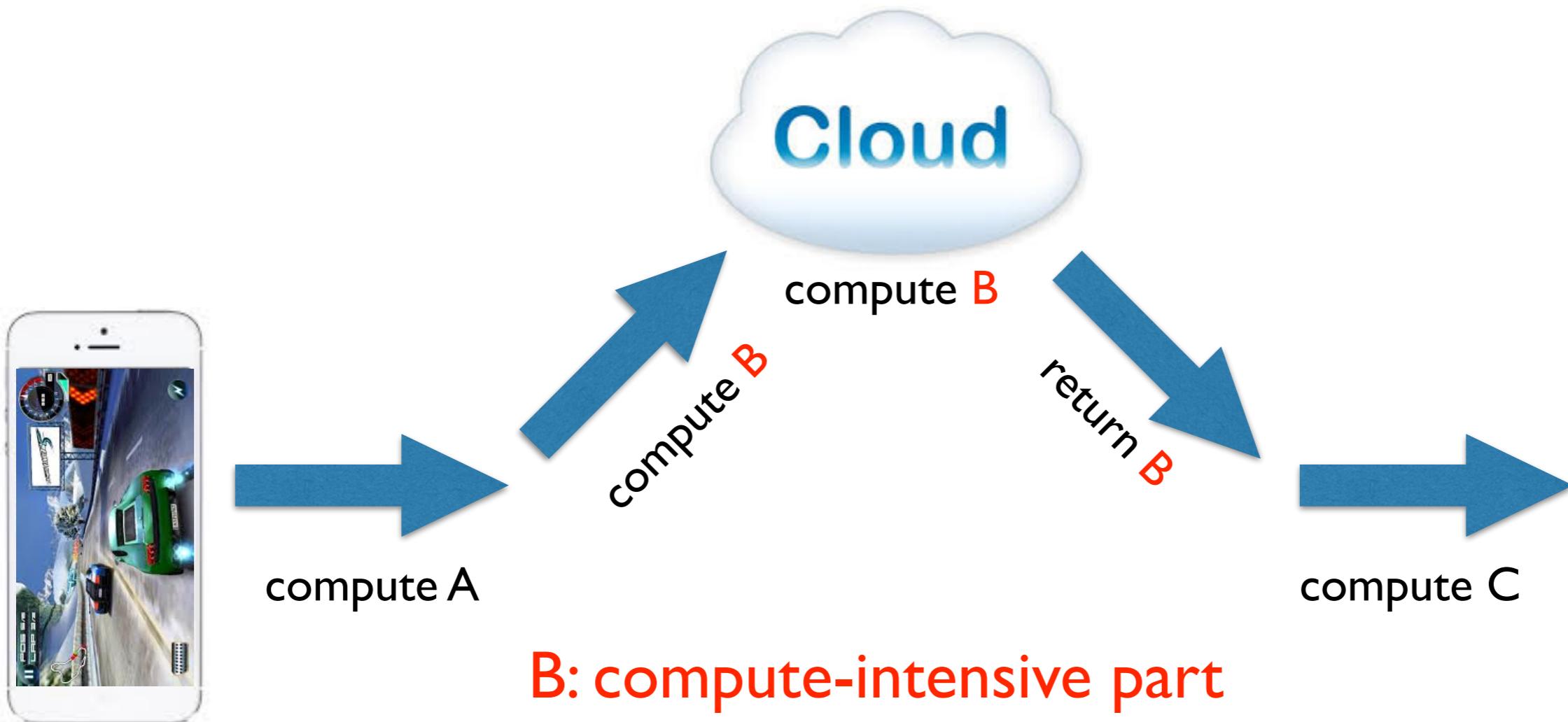
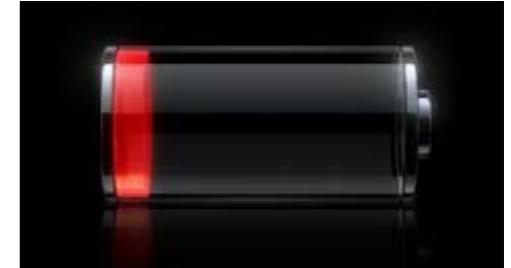
Static Analysis for Mobile Computing



Static Analysis for Mobile Computing



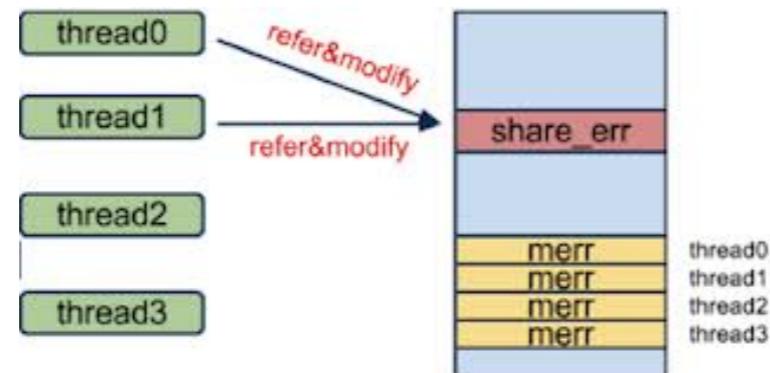
Static Analysis for Mobile Computing



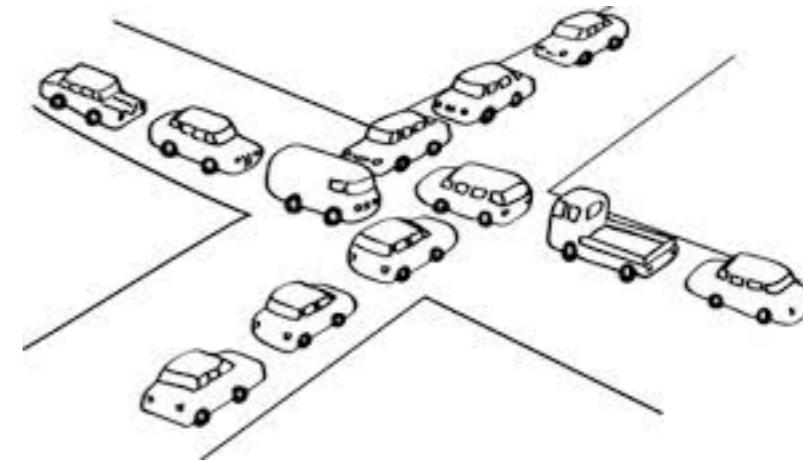
Plan: Static analysis to estimate power consumption

Static Analysis for Parallel Computing

Concurrency bugs



data races



dead locks

Programming system for

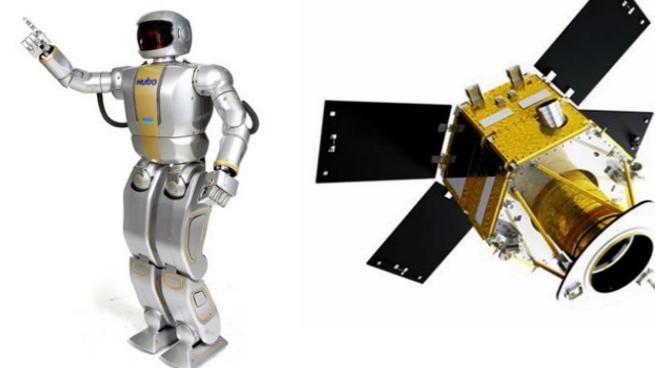
- detection of concurrency bugs
- repair of concurrency bugs

Many others (SE, Network, etc)

Mobile / Cloud / Parallel Computing



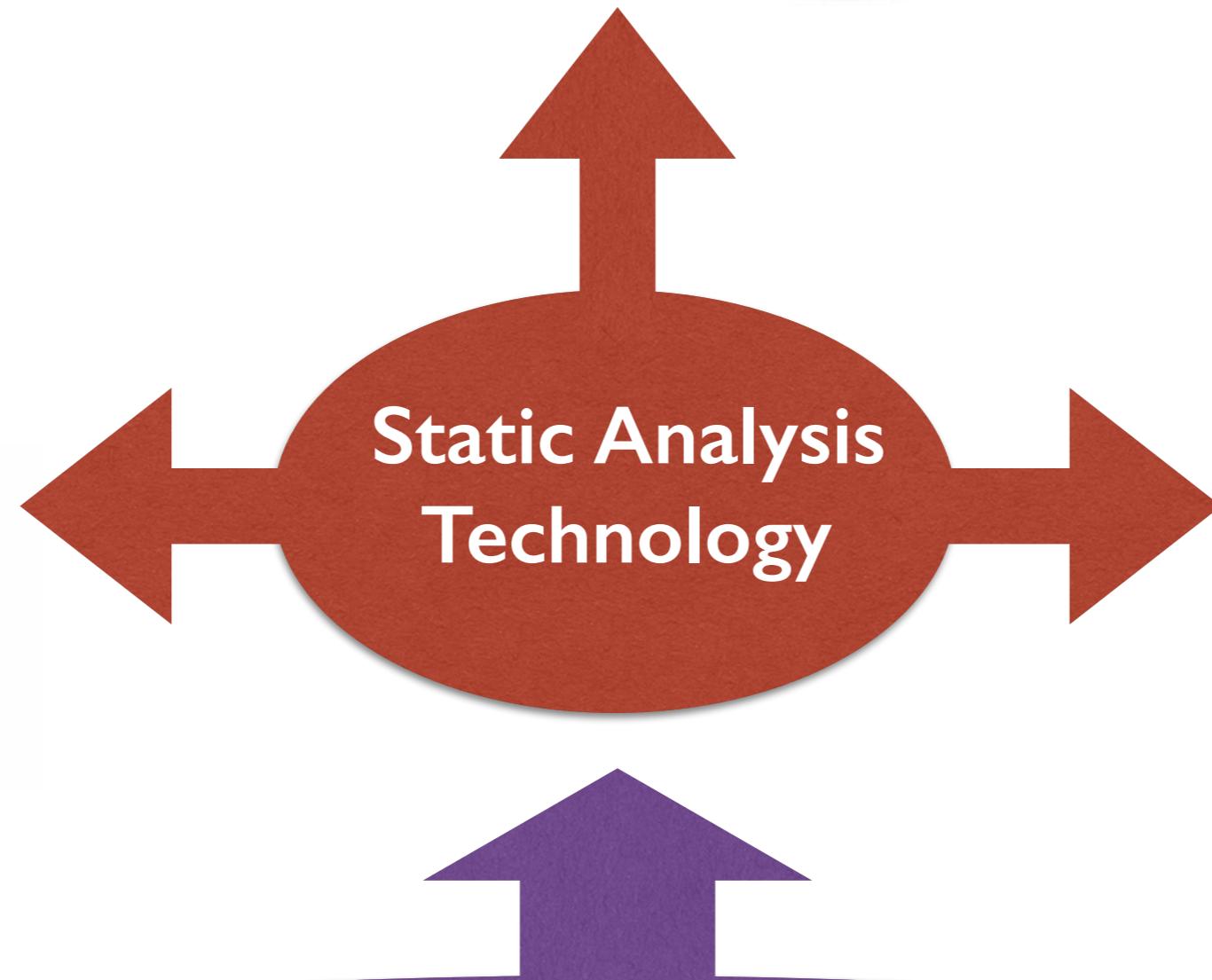
SW Verification



SW Security



Static Analysis
Technology



Programming Languages Theories



4: Research Internship Positions in Program Analysis @ Google

We have a number of research internship openings in 2015. Internships could be full-time or part-time, depending on the candidate's availability and interest. Possible topics include:

- Dynamic symbolic execution
- Refinement-based alias analysis
- Distributed static analysis of large applications
- Identification of vulnerabilities in Java through static analysis
- Concurrent data-flow analysis
- Refining flow-insensitive analyses
- Ideal candidates would have strong research background and solid (C++) programming skills.

Google

facebook.

- D
- I
- C
- R
- Ideal ca

Internship Positions in Program Analysis @ Google

research internship openings in 2015. Internships could



Infer

A tool to detect bugs in Android and iOS apps before they ship

Google

facebook.



- Design
- Identify
- Create
- Review
- Ideal candidate

A tool to do

Internship Positions in Program Analysis @ Google

internship openings in 2015. Internships could

Static Code Analysis

potential bugs—in the source code of a project with the static analyzer built into Xcode. Source code may have subtle errors that the compiler and manifest themselves only at runtime, when they could be difficult to identify and fix.

Steps

1. Choose Product > Analyze.
2. In the issue navigator, select an analyzer message.
3. In the source editor, click the corresponding message.
4. Use the pop-up menu in the analysis results bar above the edit area to study the flow path of the flaw.
5. Edit the code to fix the flaw.

The video shows the process of looking at a flaw in the source file `SKTText.m`.

The screenshot shows the Xcode interface with the title bar "gate Editor Product Window Help" and the window title "Sketch". The status bar at the bottom right shows "Clean Sketch: Succeeded | 2:45 PM" and "No issues". The left sidebar shows a file tree with files like `SKTAppDelegate.h`, `SKTAppDelegate.m`, `SKTDocument.h`, etc., and `SKTText.m` is selected. The main editor pane displays the `SKTText.m` file. A blue circle highlights a specific line of code:

```
// This kind of graphic shouldn't appear opaque just be
[textView setDrawsBackground:NO];
```

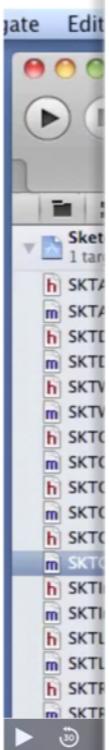
 A tooltip appears over the code: "This has been handy for debugging text editing via". The code continues with:

```
/*
[textView setBackgroundColor:[NSColor greenColor]];
[ textView setDrawsBackground:YES];
*/
// Start off with the all of the text selected.
[textView setSelectedRange:NSMakeRange(0, [contents len
// Specify that the text view should grow and shrink to
// removed, but only in the vertical direction. With t
// be large enough to show an extra line fragment but
// able to see just-typed text on the screen. Sending
// view without also sending -setMinSize: or -setMaxSi
// default minimum and maximum sizes of a text view ar
// at initialization time.
[textView setMinSize:NSMakeSize(bounds.size.width, 0.0)
[textView setMaxSize:NSMakeSize(bounds.size.width, supe
)];
[textView setVerticallyResizable:YES];
```



- Detection
- Identification
- Classification
- Remediation
- Ideal candidate

A tool to detect



Internship Positions in Program Analysis @ Google

internship openings in 2015. Internships could



1. Detect and handle navigation
2. Identify malicious code
3. In the source editor
4. Use the pop-up report
5. Edit the code to fix it

The video shows

Products Solutions Mandiant

Home > Products > Mobile Security

Mobile Security

Mobile Threat Preve

Detect and prevent cyber attacks that spy on, profile, or use mobile devices

Malicious apps compromise mobile security to access private information, such as contact lists and calendar details. They also use mobile device features, such as cameras and microphones, to spy, profile users, or conduct cyber attacks.

FireEye Mobile Security (Mobile Threat Prevention) detects and prevents these mobile threats and provides visibility into mobile security trends across the enterprise. FireEye Mobile Threat Prevention also integrates with industry leading mobile device management (MDM) providers.

Research Program

- Undergraduate research interns
- Graduate students for pursuing master and phd courses
- Related researches:



Massachusetts
Institute of
Technology

Stanford