Analyzing Million Lines of C: A General Sparse Analysis Framework

Kwangkeun Yi

Programming Research Laboratory
Seoul National University

04/23/2012 @ CSAIL MIT

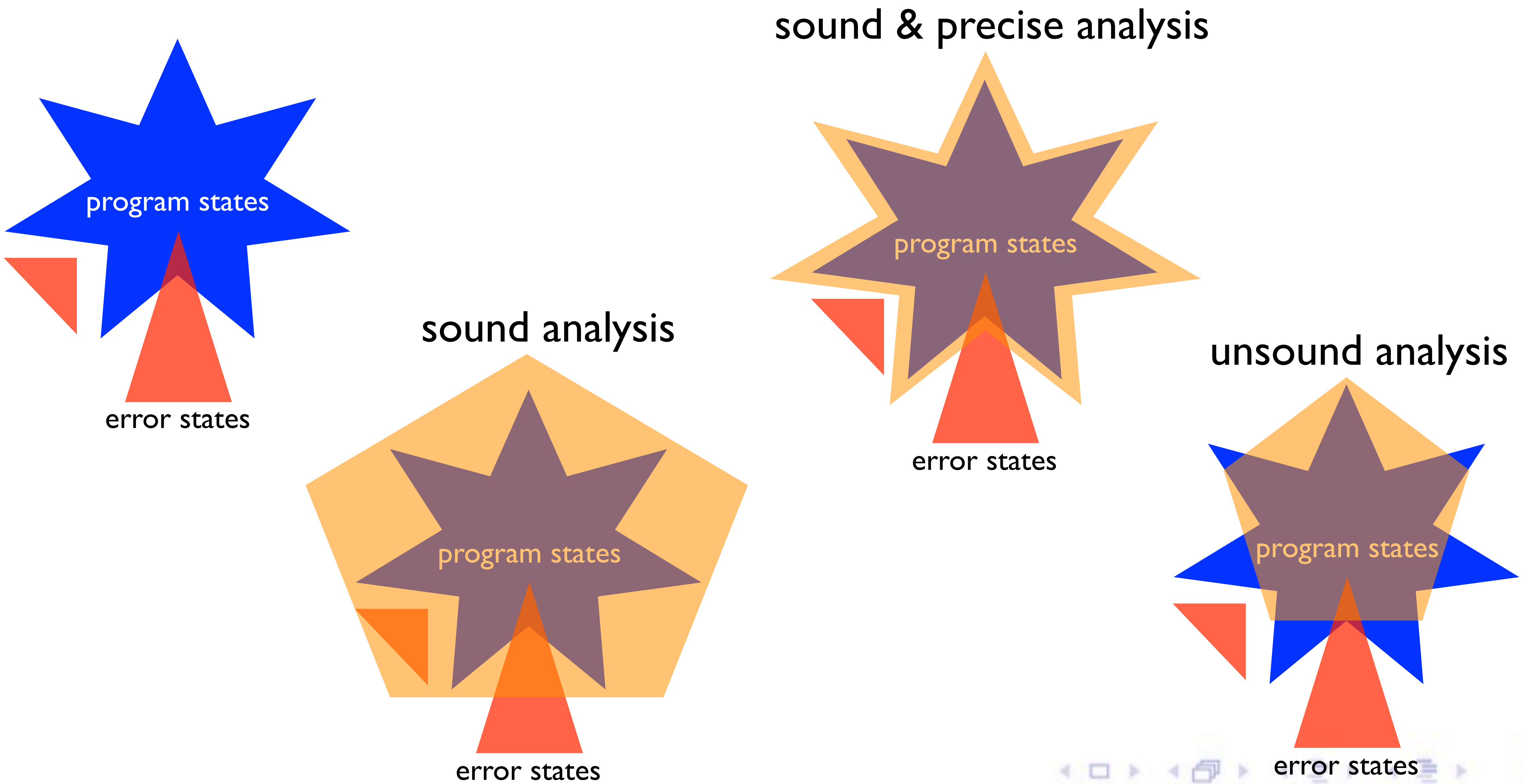
Co-work with Hakjoo Oh, Kihong Heo, Wonchan Lee, Wooseok Lee
[PLDI'12, VMCAI'11, APLAS'11,· · ·]

Static Analysis: Sound, Unsound, Useful

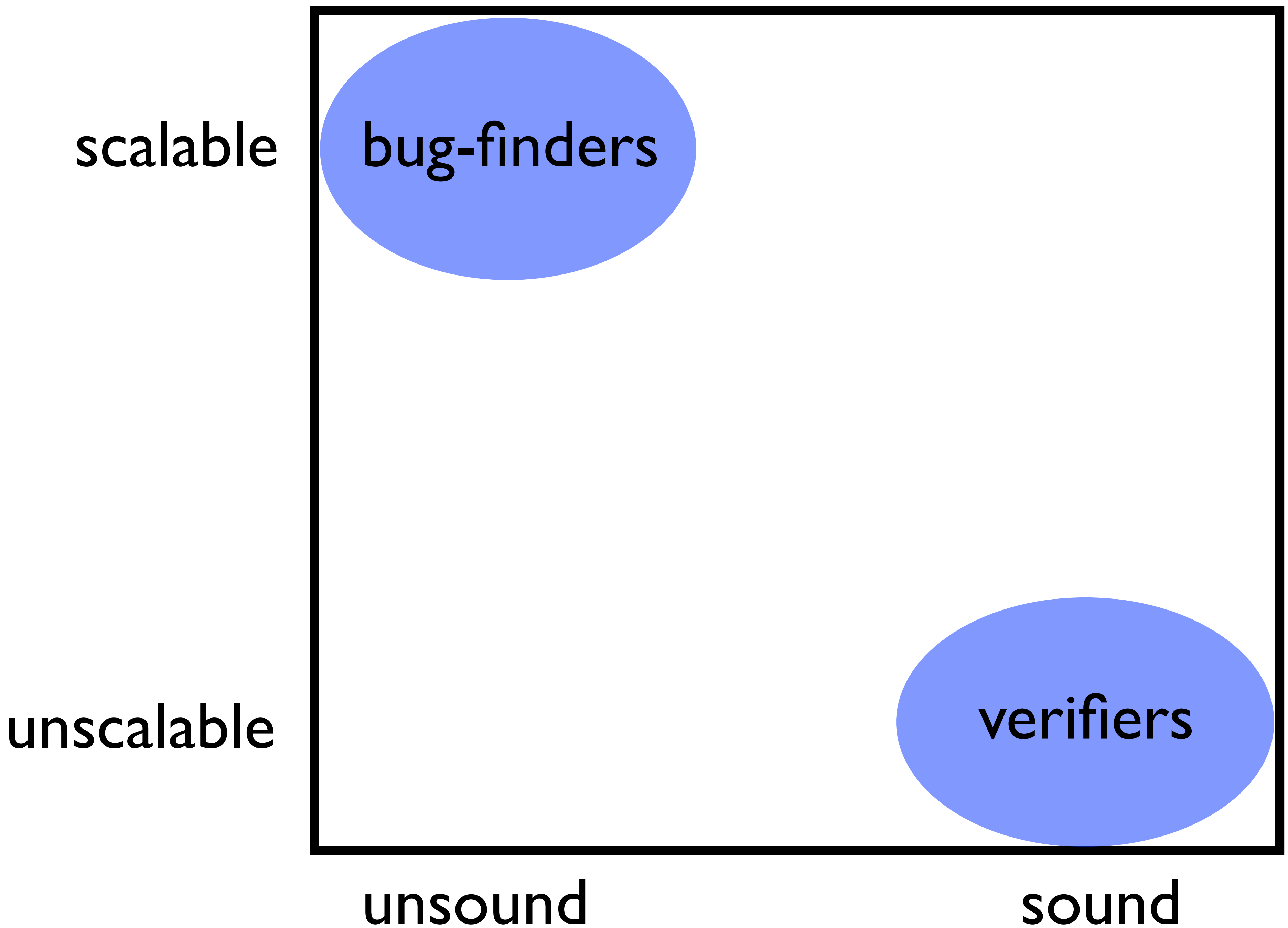
Automatic sound estimation of sw behaviors before execution

- under many names

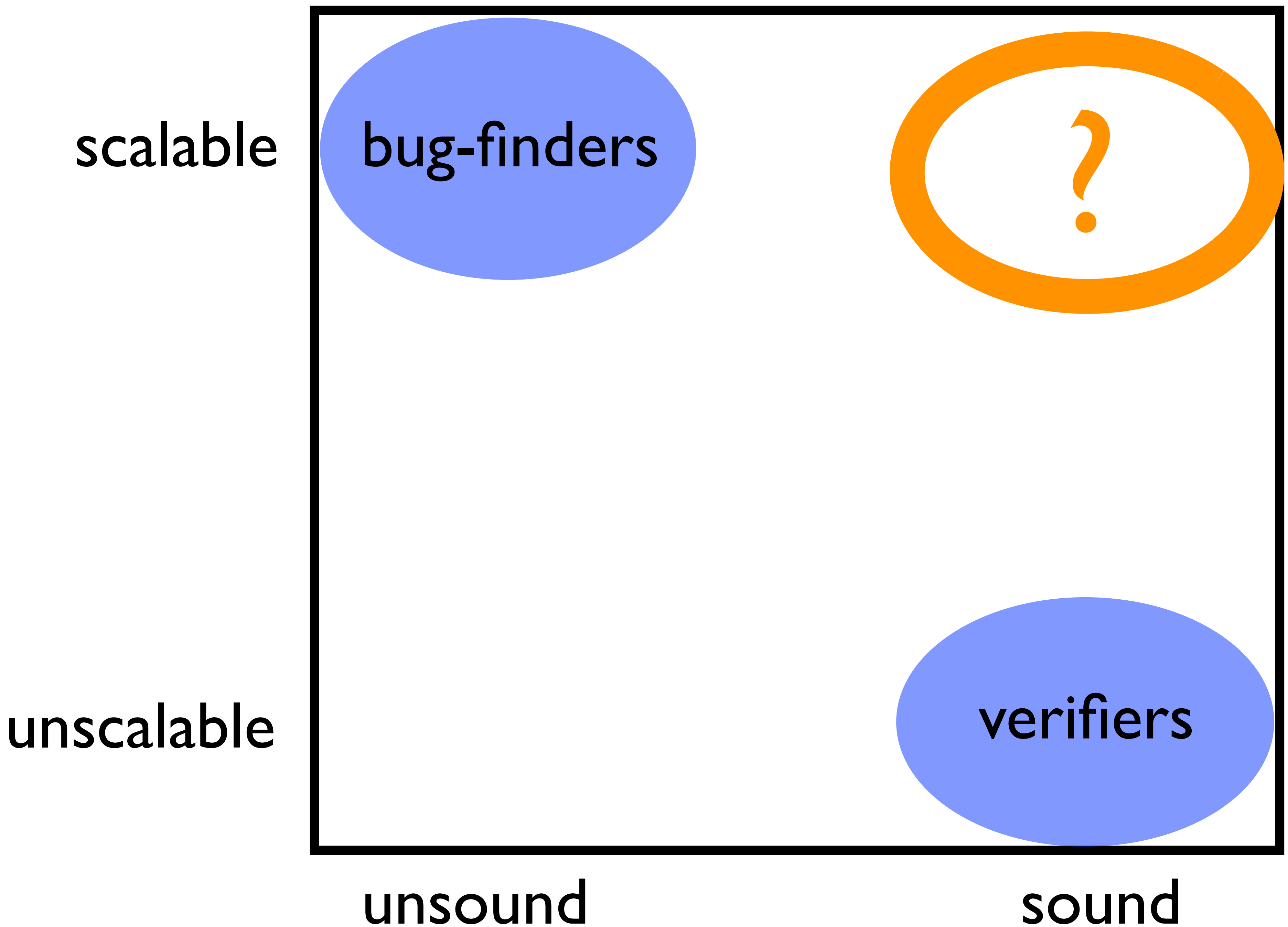
theory	"abstract interpretation"
pl, se, veri.	"type system", "model checking", "theorem proving"
cmplr	"data-flow analysis", etc.



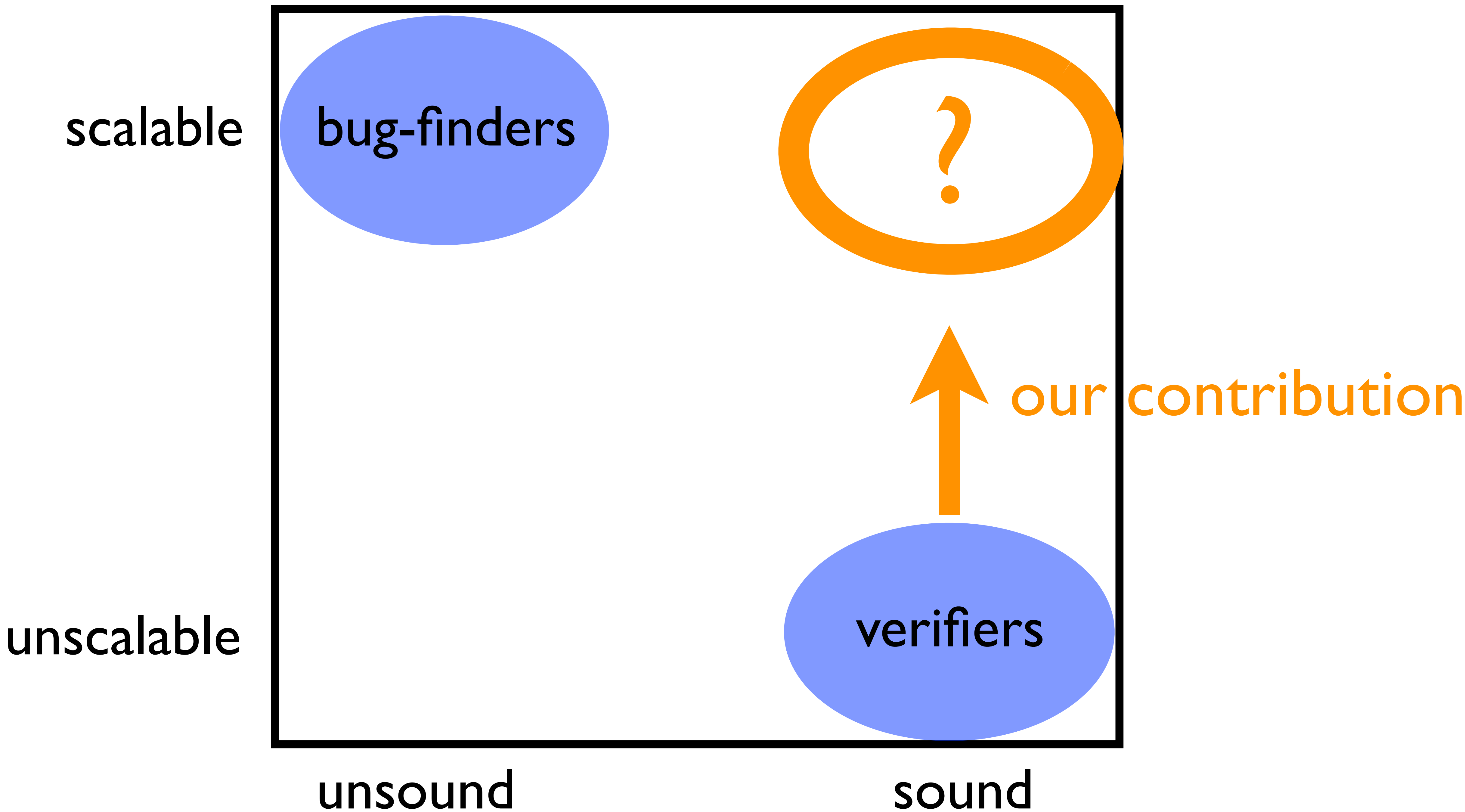
Dichotomy: "bug-finders" vs. "verifiers"



Dichotomy: “bug-finders” vs. “verifiers”



Dichotomy: "bug-finders" vs. "verifiers"

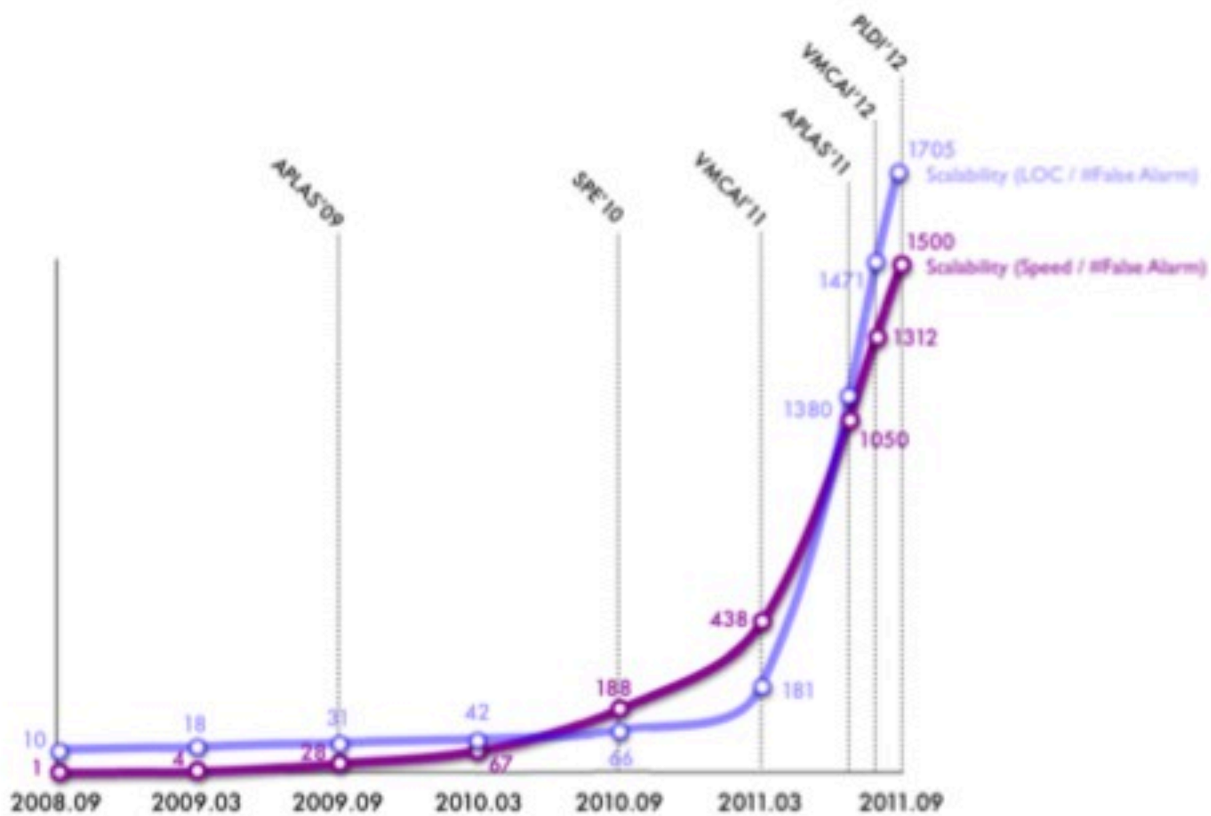


Our Story

- In 2007, we commercialized Sparrow The Early Bird
  - sound in design, unsound yet scalable in reality
  - memory-bug-finding tool for full C, non domain-specific
- Realistic workbench available
  - "let's try to scale-up its sound & global version"

Our Scalability Improvement

Sparrow The Early Bird sound-&-global version



- < 1.4M in 10hrs with intervals
- < 0.14M in 20hrs with octagons

This Talk

- How we achieved
  - sound design of Sparrow
  - spatial & temporal localizations
- Sparse analysis framework
  - general for AI-based analyzers for C-like languages
  - precision-preserving

"An important strength is that the theoretical result is very general. It could be applied to many other analyses. PLDI papers have been accepted that were simply instances of this framework." (from PLDI reviews)

Static Analysis Example: Abstract Equations

```
     x = readInt;
1:
     while (x ≤ 99)
2:
          x++;
3:
     end
4:
```

Capture the dynamics by abstract equations; solve; reason.

$$
\begin{aligned}
x_1 &= [-\infty, +\infty] \ or \ x_3 \\
x_2 &= x_1 \ and \ [-\infty, 99] \\
x_3 &= x_2 \dot{+} 1 \\
x_4 &= x_1 \ and \ [100, +\infty]
\end{aligned}
$$

How to Design Sound Static Analyses?

Abstract Interpretation [CousotCousot]: a powerful design theory

- How to derive correct yet arbitrarily precise equations?
  - Non-obvious: ptrs, heap, exns, high-order ftns, etc.

```
x = readInt;
while (x ≤ 99)
        x++;
end
```

$$\overset{\text{how?}}{\Longrightarrow} \quad \begin{array}{lcl} x_1 & = & [-\infty, +\infty] \ \mathit{or} \ x_3 \\ x_2 & = & x_1 \ \mathit{and} \ [-\infty, 99] \\ x_3 & = & x_2 \dot{+} 1 \\ x_4 & = & x_1 \ \mathit{and} \ [100, +\infty] \end{array}$$

  - Define an abstract semantics function $\hat{F}$ s.t. $\cdots$

- How to solve the equations in a finite time?

$$\begin{array}{lcl} x_1 & = & [-\infty, +\infty] \ \mathit{or} \ x_3 \\ x_2 & = & x_1 \ \mathit{and} \ [-\infty, 99] \\ x_3 & = & x_2 \dot{+} 1 \\ x_4 & = & x_1 \ \mathit{and} \ [100, \infty] \end{array} \quad \overset{\text{how?}}{\Longrightarrow} \quad \begin{array}{lcl} x_1 & = & [-\infty, +\infty] \\ x_2 & = & [-\infty, 99] \\ x_3 & = & [-\infty, 100] \\ x_4 & = & [100, +\infty] \end{array}$$

  - Fixpoint iterations for an upperbound of $\mathit{fix}\hat{F}$

Design of Sparrow

The Early Bird

- Designed in the *abstract interpretation* framework
- To find memory safety violations in C
  - buffer-overrun, memory leak, null deref., etc.
  - flow-sensitive values analysis for int & ptrs (static + dynamic)
  - for the full set of C

Program

$\langle \mathbb{C}, \hookrightarrow \rangle$

- $\mathbb{C}$: set of program points
- $\hookrightarrow \subseteq \mathbb{C} \times \mathbb{C}$: control flow relation

$c' \hookrightarrow c$ (c is the next command to c')

Commands

$$lv := e \mid lv := \mathsf{alloc}(a) \mid \mathsf{assume}(x < e) \mid \mathsf{call}(f_x, e) \mid \mathsf{return}_f$$

$$\begin{array}{llll} \text{expression} & e & \rightarrow & n \mid e \texttt{ + } e \mid lv \mid \&lv \\ \text{l-value} & lv & \rightarrow & x \mid \texttt{*}e \mid e\texttt{[}e\texttt{]} \mid e\texttt{.}x \\ \text{allocation} & a & \rightarrow & \texttt{[}e\texttt{]}_l \mid \texttt{\{}x\texttt{\}}_l \end{array}$$

Abstract Semantics

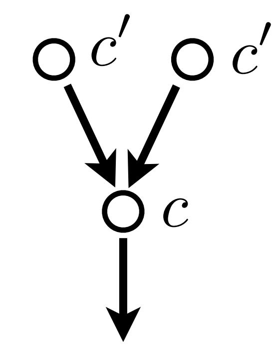
- One abstract state $\in \hat{\mathbb{S}}$ that subsumes all reachable states at each program point

$$
\begin{aligned}
[\![\hat{P}]\!] \in \mathbb{C} \to \hat{\mathbb{S}} &= \mathit{fix}\,\hat{F} \\
\hat{\mathbb{S}} &= \hat{\mathbb{L}} \to \hat{\mathbb{V}}
\end{aligned}
$$

$$
\begin{aligned}
\hat{\mathbb{L}} &= \mathit{Var} + \mathit{AllocSite} + \mathit{AllocSite} \times \mathit{FieldName} \\
\hat{\mathbb{V}} &= \hat{\mathbb{Z}} \times 2^{\hat{\mathbb{L}}} \times 2^{\mathit{AllocSite} \times \hat{\mathbb{Z}} \times \hat{\mathbb{Z}}} \times 2^{\mathit{AllocSite} \times 2^{\mathit{FieldName}}} \\
\hat{\mathbb{Z}} &= \{[l, u] \mid l, u \in \mathbb{Z} \cup \{-\infty, +\infty\} \wedge l \leq u\} \cup \{\bot\}
\end{aligned}
$$

- Abstract semantic function

$$
\begin{aligned}
\hat{F} &\in (\mathbb{C} \to \hat{\mathbb{S}}) \to (\mathbb{C} \to \hat{\mathbb{S}}) \\
\hat{F}(\hat{X}) &= \lambda c \in \mathbb{C}. \hat{f}_c\Big(\bigsqcup_{c' \hookrightarrow c} \hat{X}(c')\Big)
\end{aligned}
$$



$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$: abstract semantics at point c

Computing $\mathit{fix}\hat{F} = \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$

$$\hat{F}(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{c' \hookrightarrow c} \hat{X}(c')).$$

$\hat{X}, \hat{X}' \in \mathbb{C} \to \hat{\mathbb{S}}$
$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$

$\hat{X} := \hat{X}' := \lambda c.\bot$
repeat
$\quad \hat{X}' := \hat{X}$
$\quad$ **for all** $c \in \mathbb{C}$ **do**
$\quad\quad \hat{X}(c) := \hat{f}_c(\bigsqcup_{c' \hookrightarrow c} X(c'))$
until $\hat{X} \sqsubseteq \hat{X}'$

Naive fixpoint algorithm

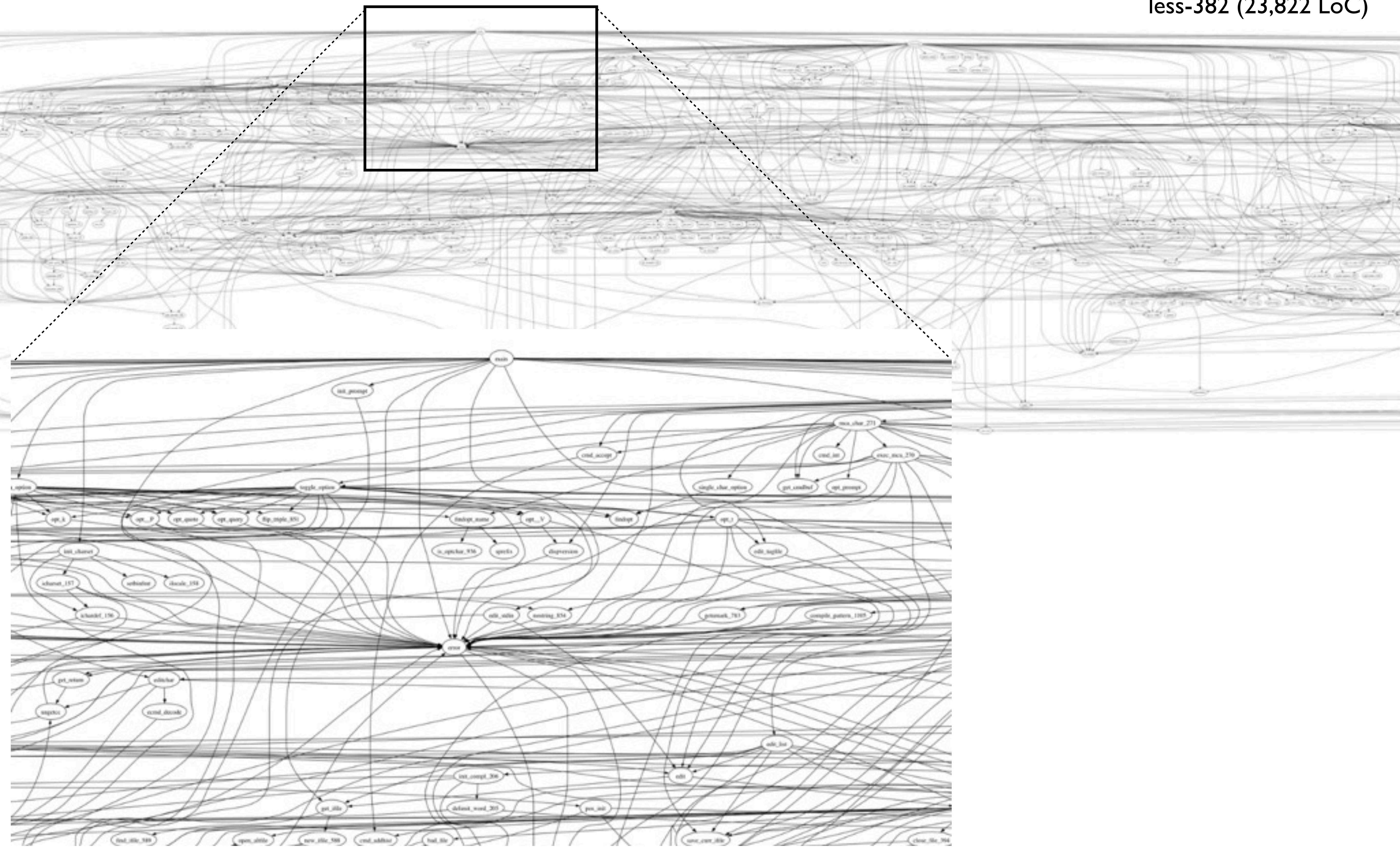
$W \in \mathit{Worklist} = 2^{\mathbb{C}}$
$\hat{X} \in \mathbb{C} \to \hat{\mathbb{S}}$
$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$

$W := \mathbb{C}$
$\hat{X} := \lambda c.\bot$
repeat
$\quad c :=$ choose(W)
$\quad \hat{s} := \hat{f}_c(\bigsqcup_{c' \hookrightarrow c} X(c'))$
$\quad$ **if** $\hat{s} \not\sqsubseteq \hat{X}(c)$
$\quad\quad W := W \cup \{c' \in \mathbb{C} \mid c \hookrightarrow c'\}$
$\quad\quad \hat{X}(c) := \hat{X}(c) \sqcup \hat{s}$
until $W = \emptyset$

Worklist algorithm

The Algorithms Too Weak To Scale

less-382 (23,822 LoC)

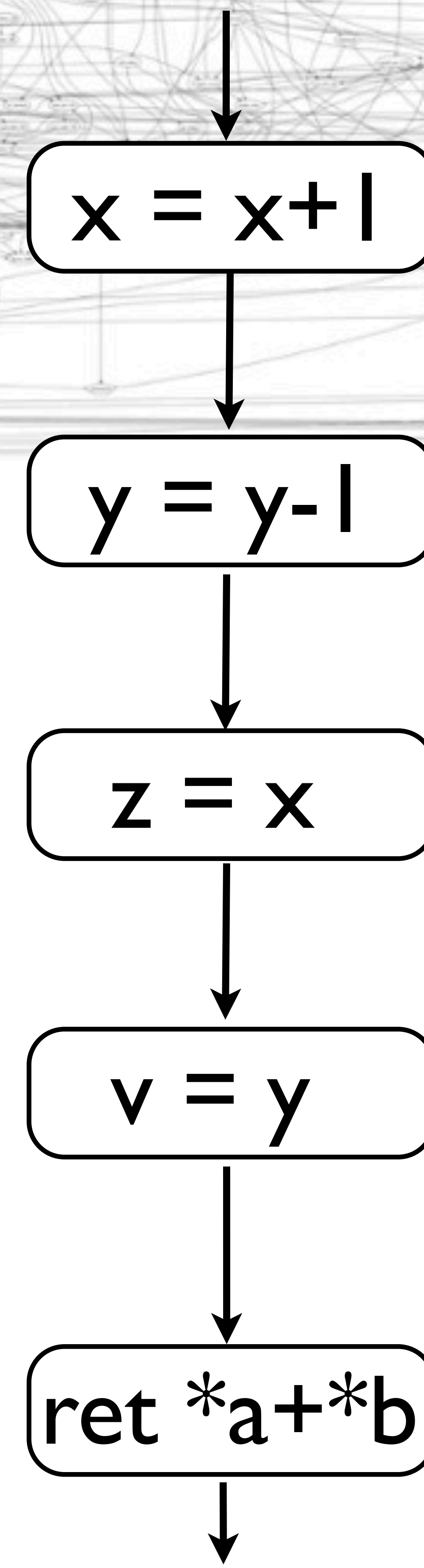


Improving Scalability

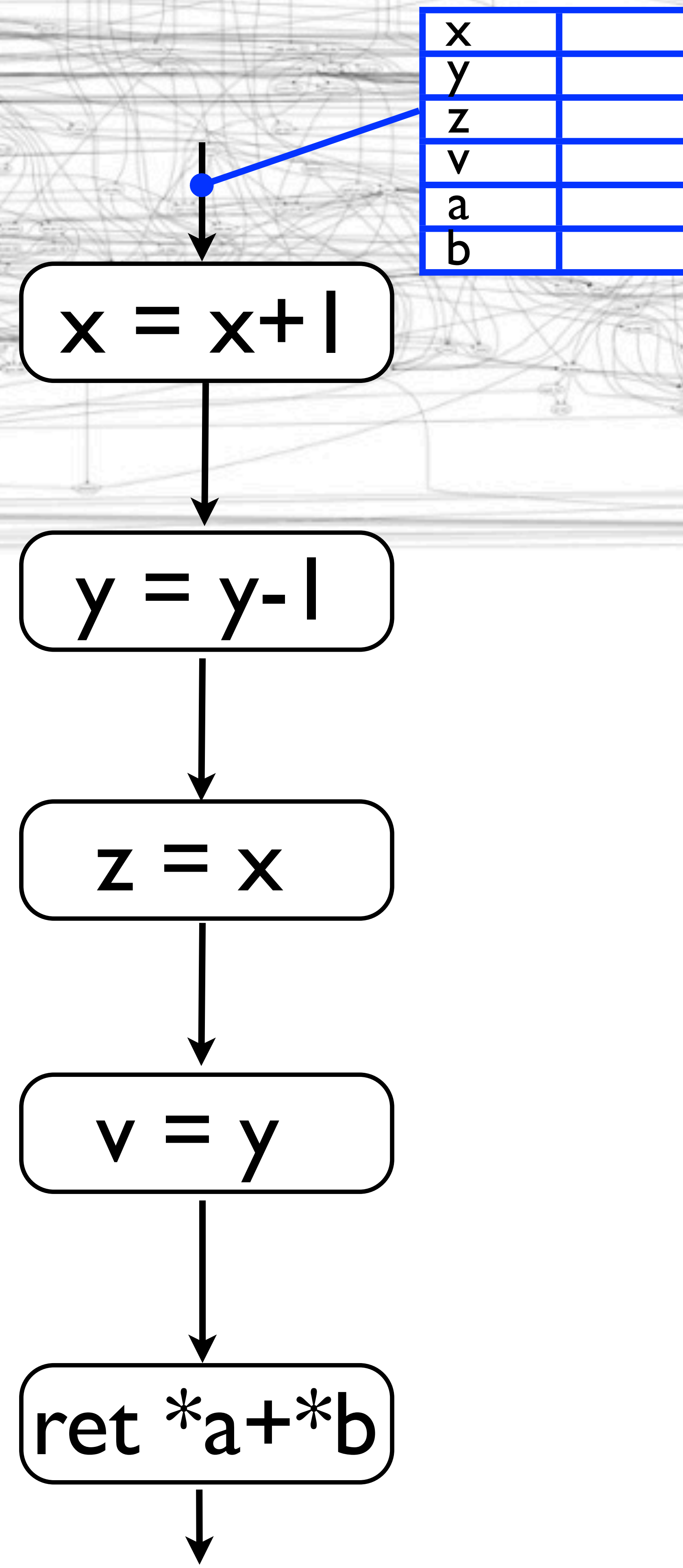
Key Idea: Localization

"Right Part at Right Moment"

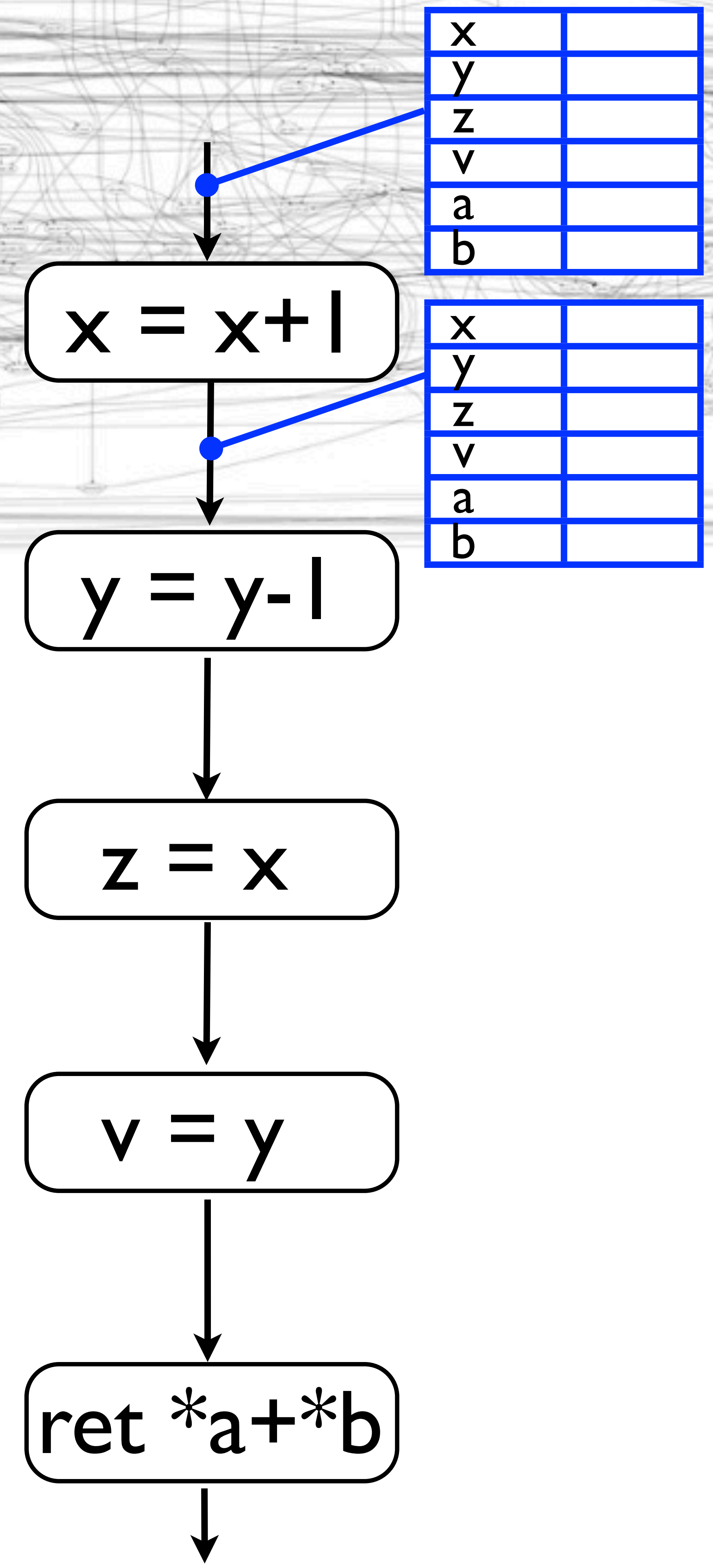
Spatial & Temporal Localizations



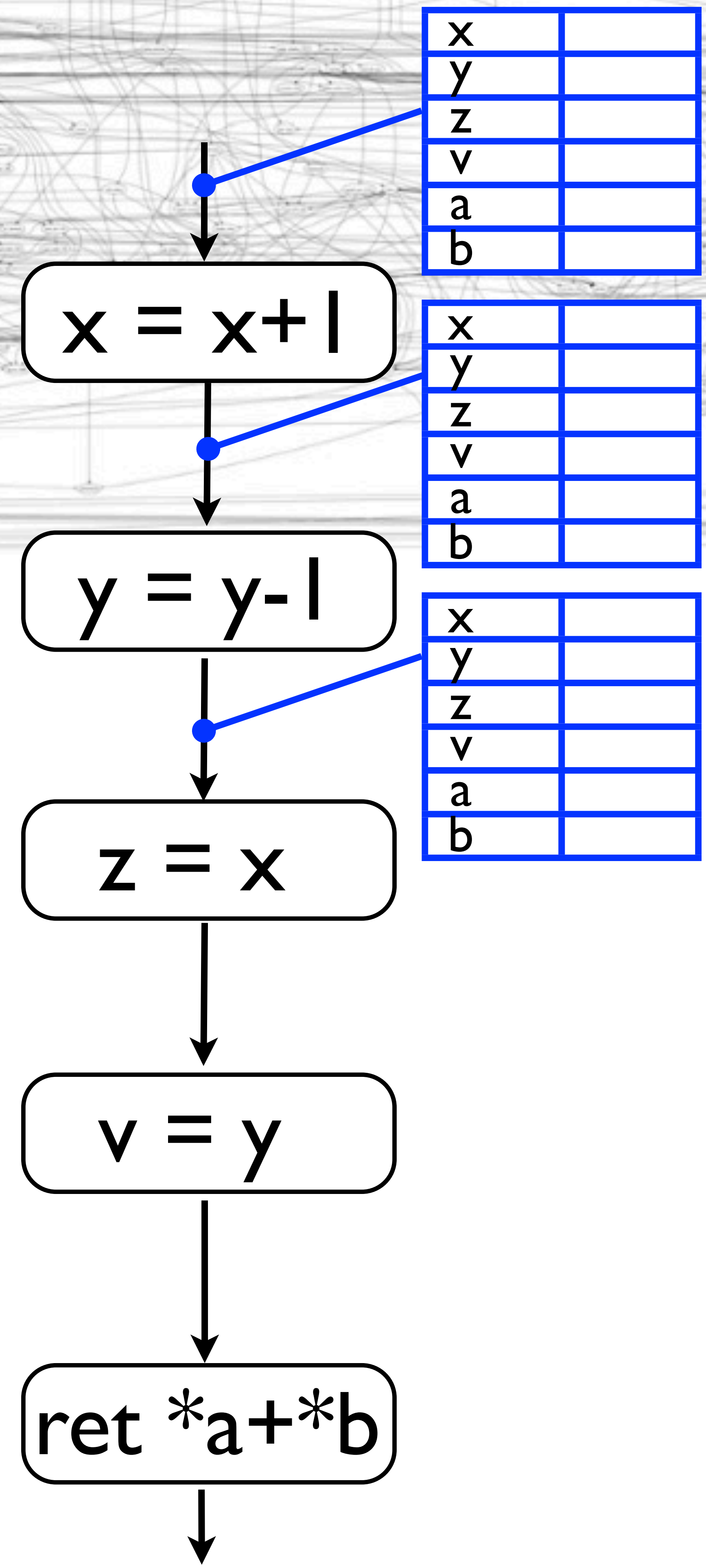
Spatial & Temporal Localizations



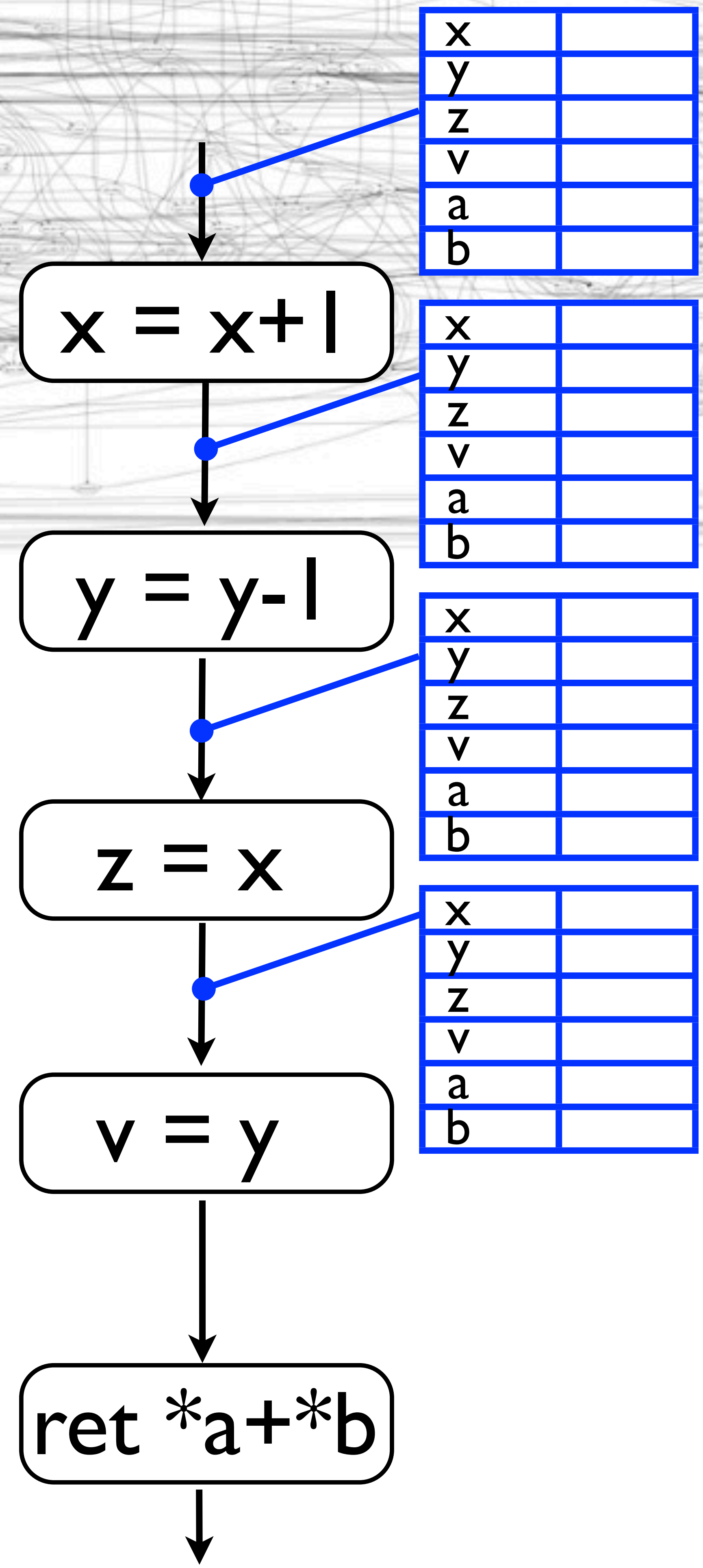
Spatial & Temporal Localizations



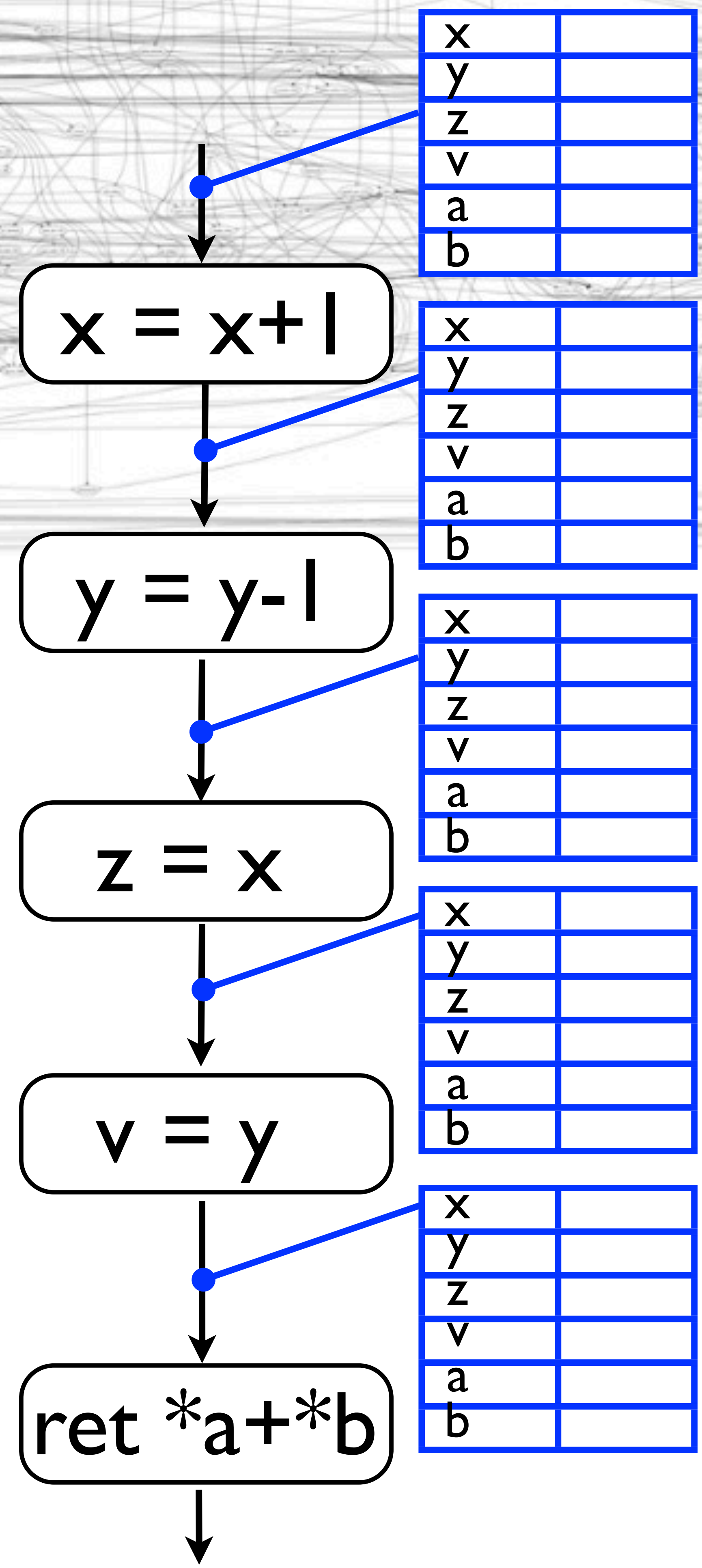
Spatial & Temporal Localizations



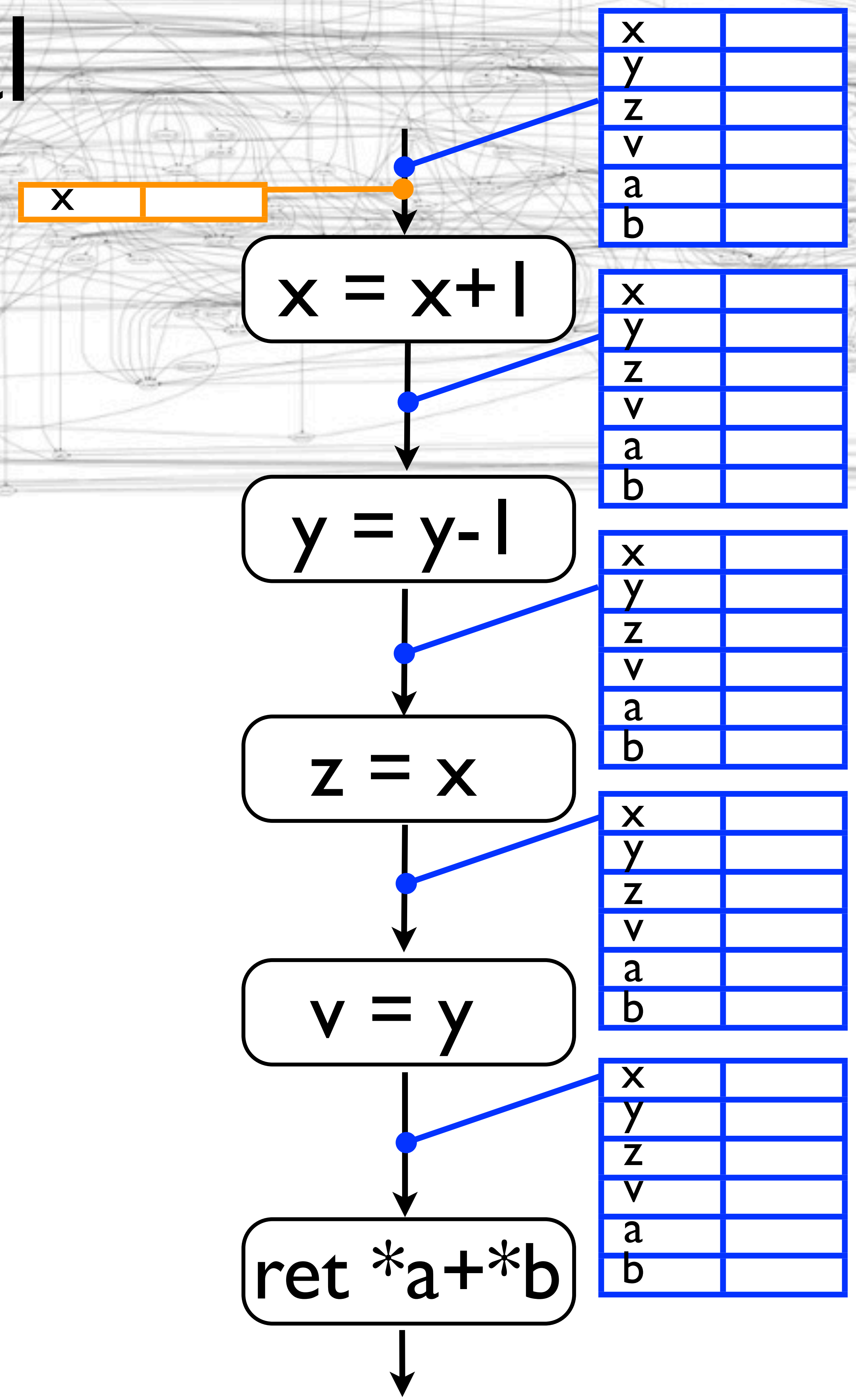
Spatial & Temporal Localizations



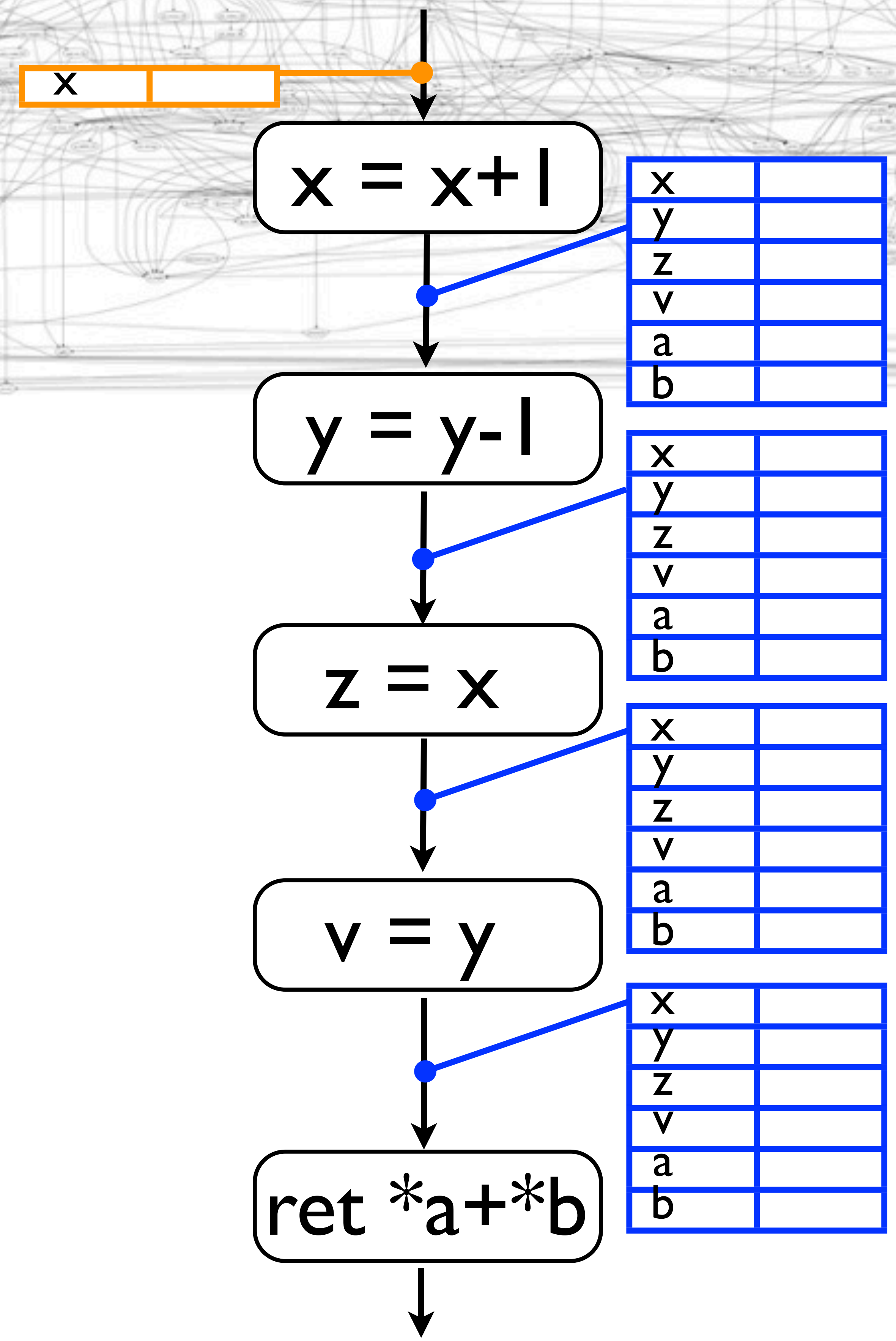
Spatial & Temporal Localizations



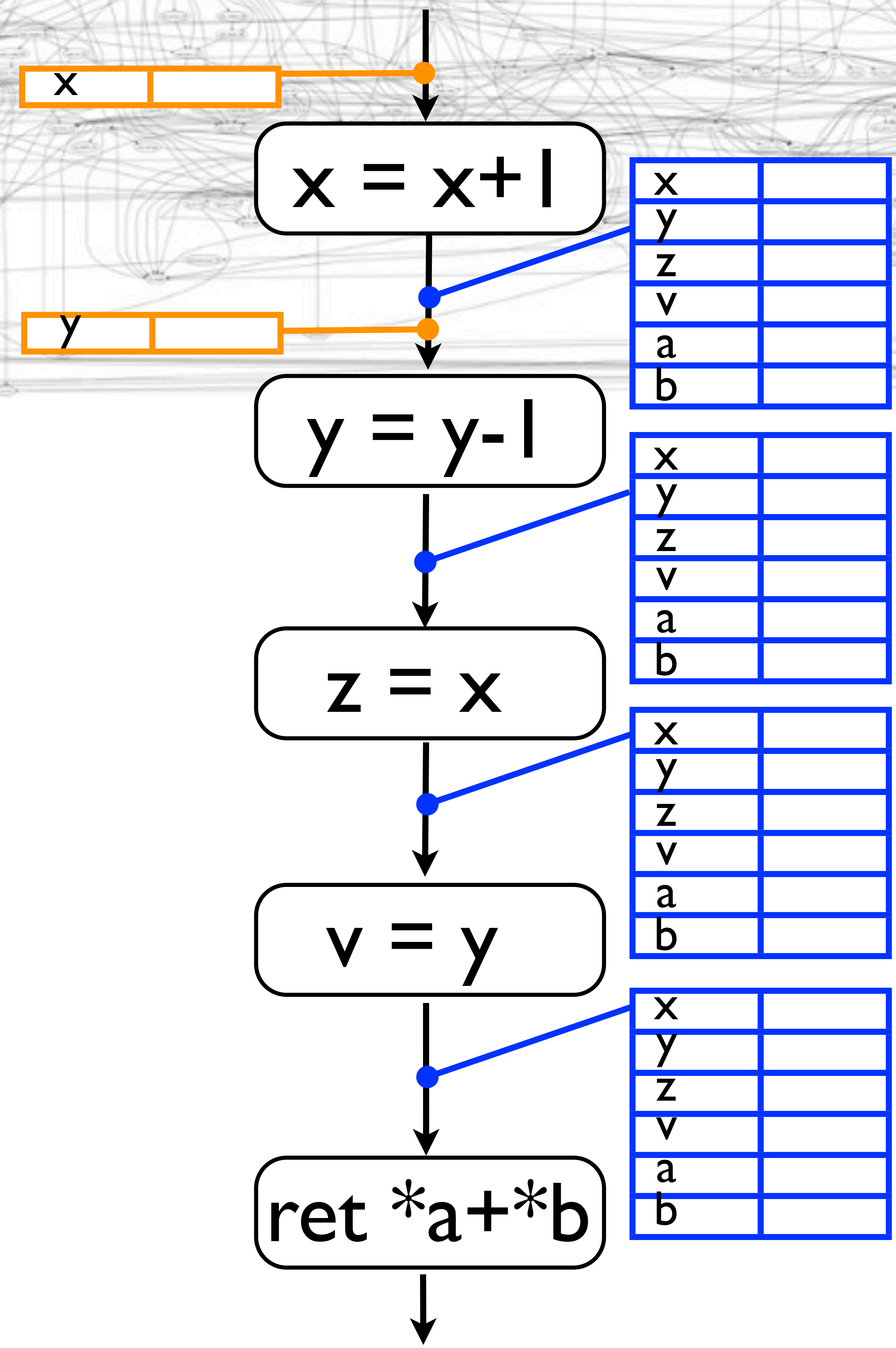
Spatial & Temporal Localizations



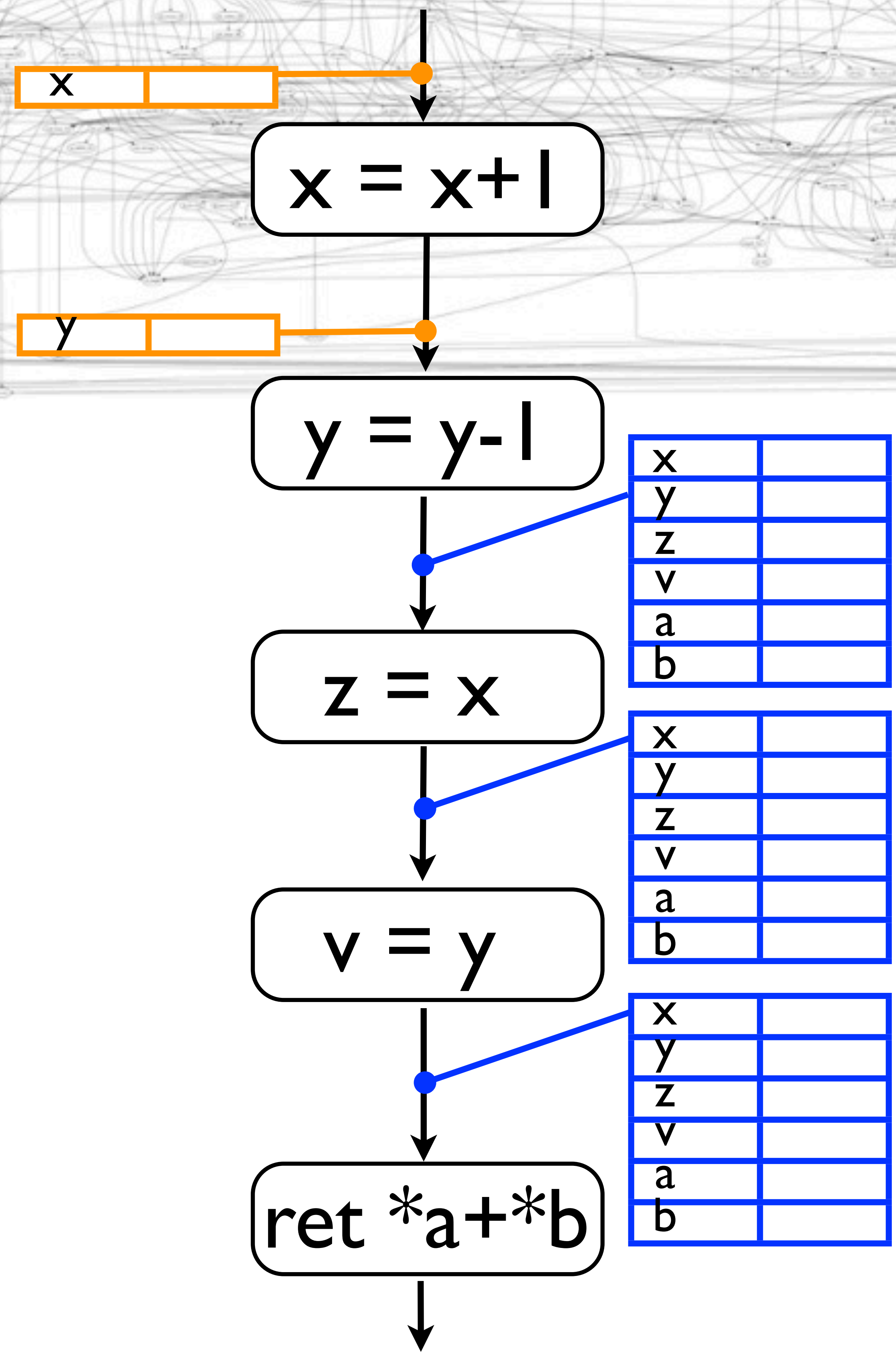
Spatial & Temporal Localizations



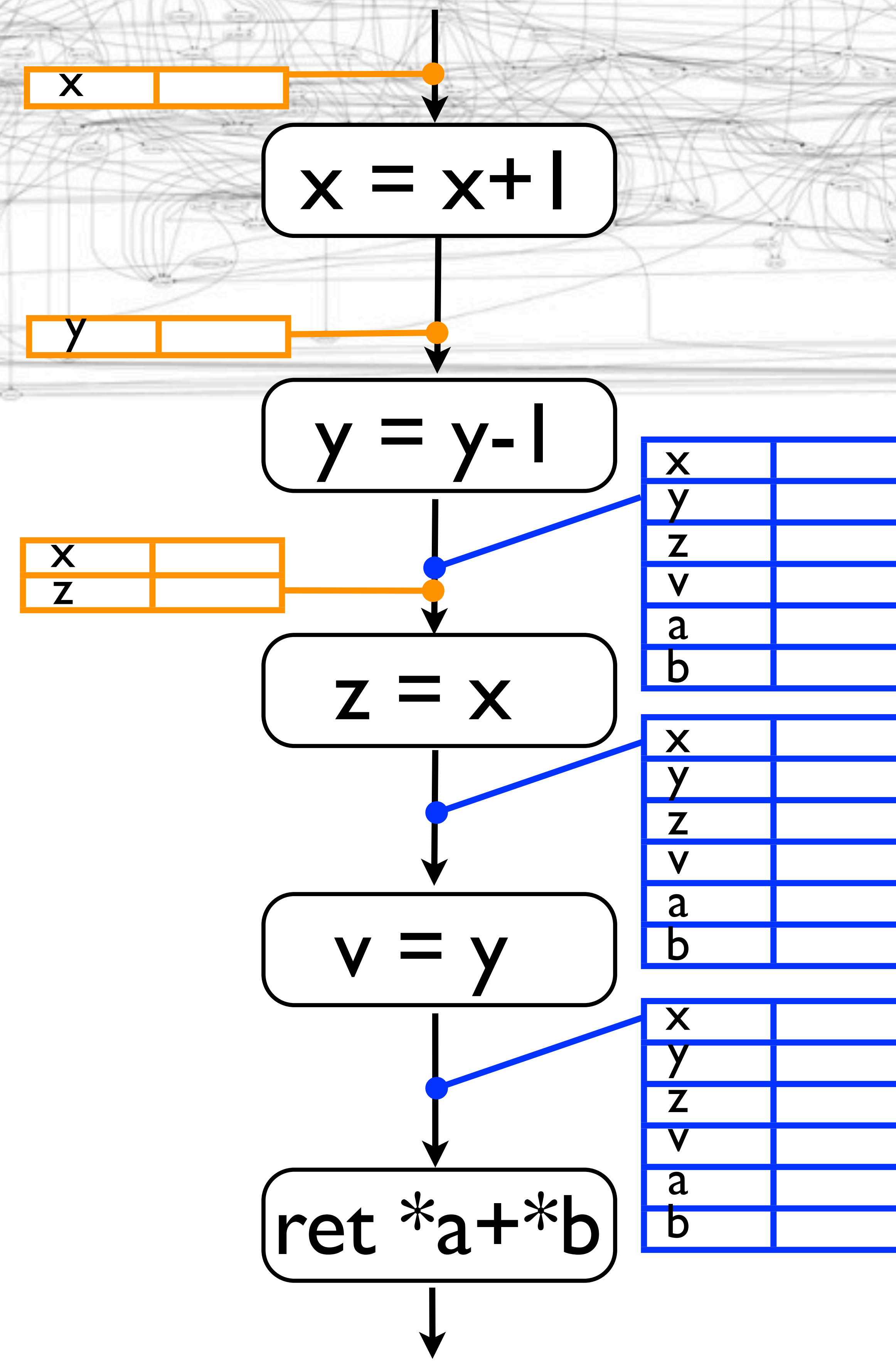
Spatial & Temporal Localizations



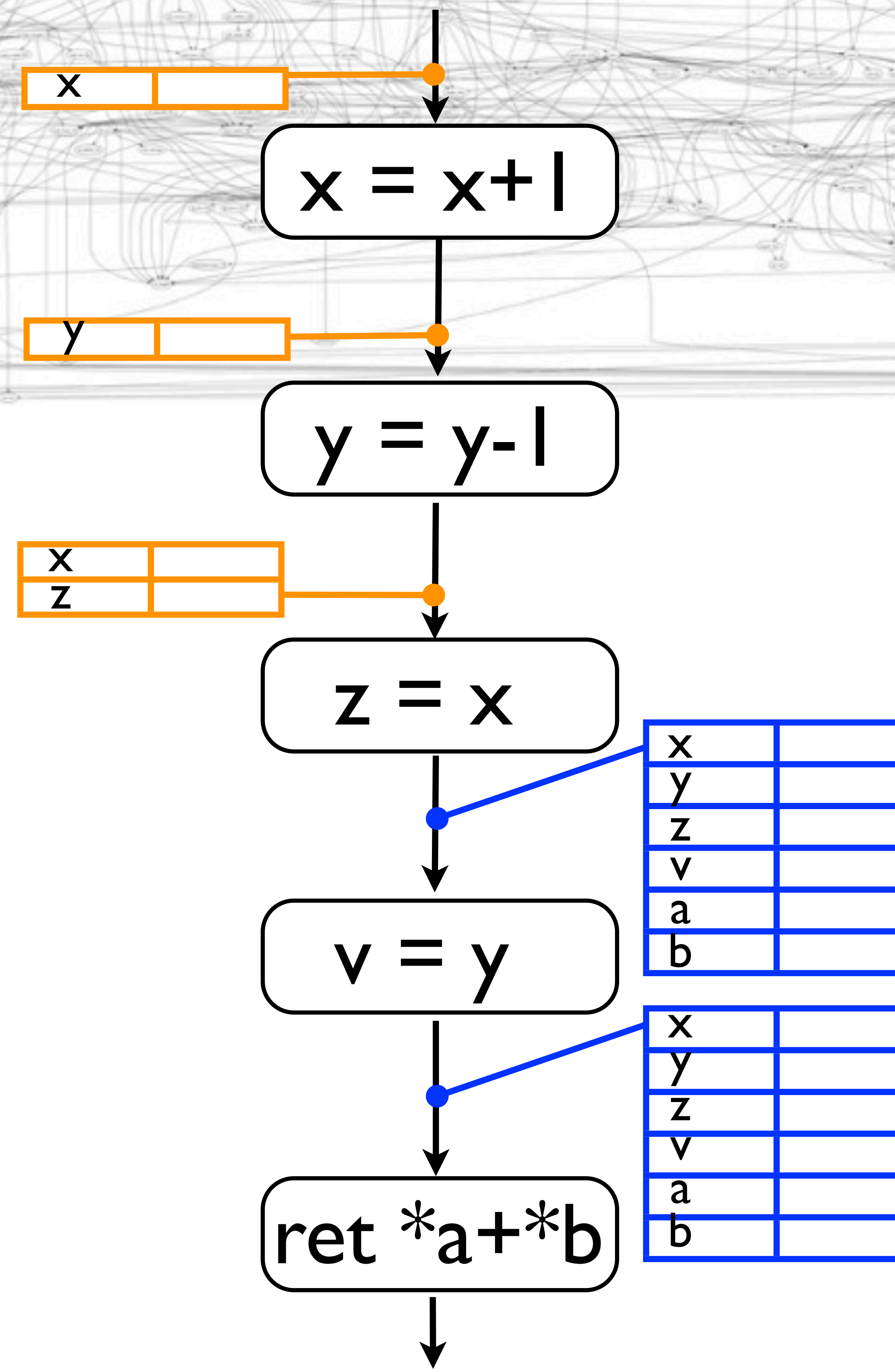
Spatial & Temporal Localizations



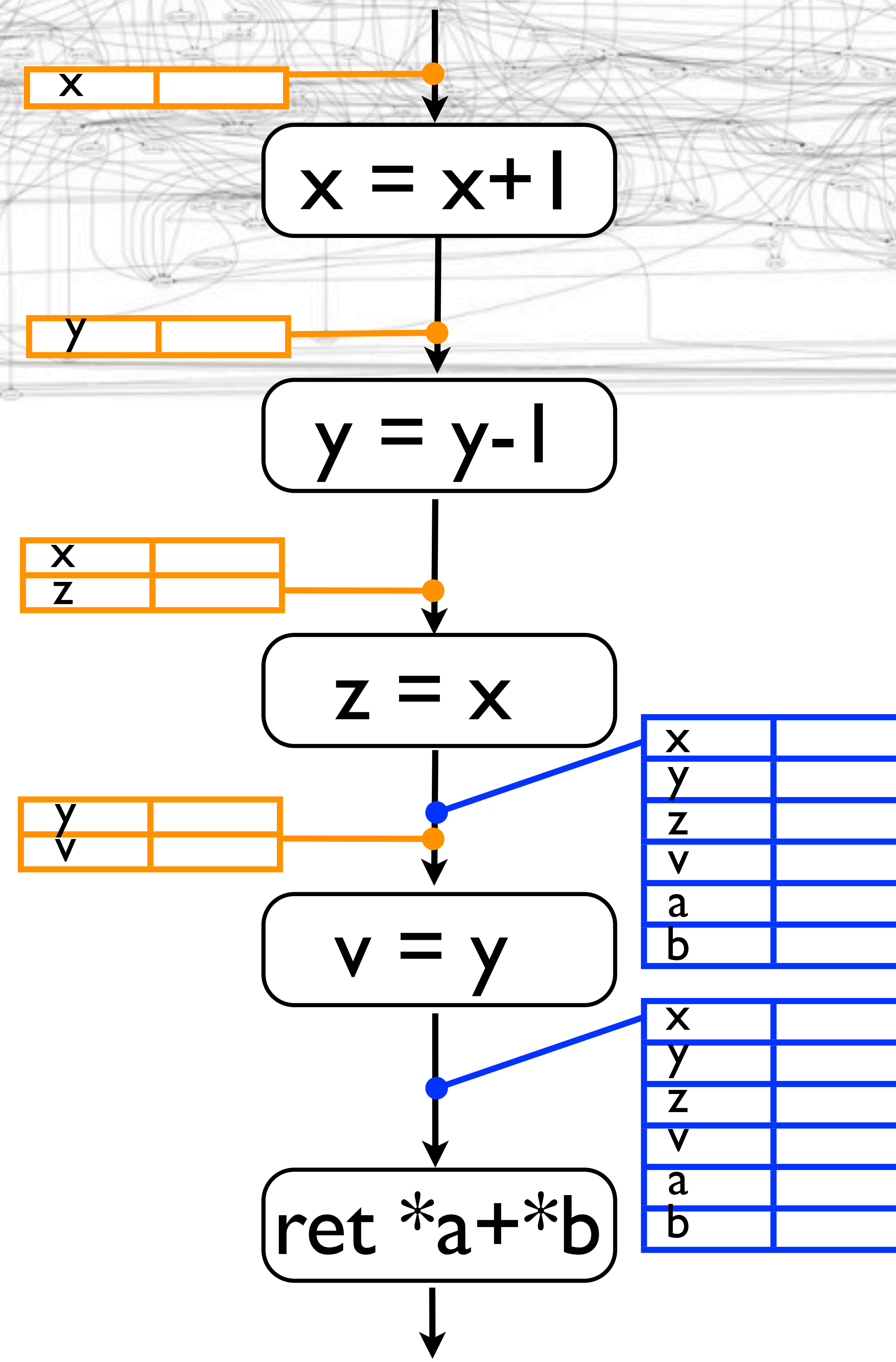
Spatial & Temporal Localizations



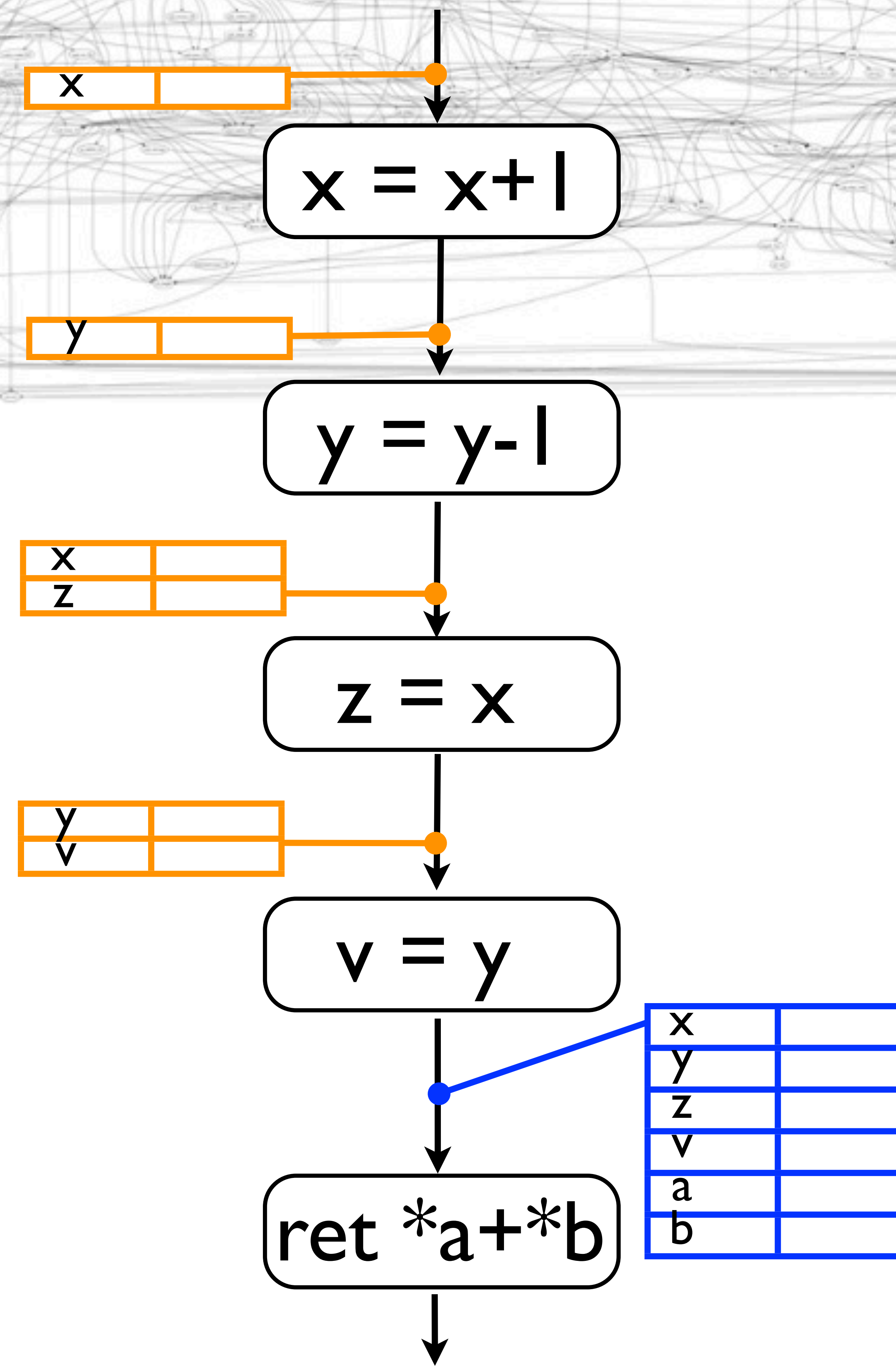
Spatial & Temporal Localizations



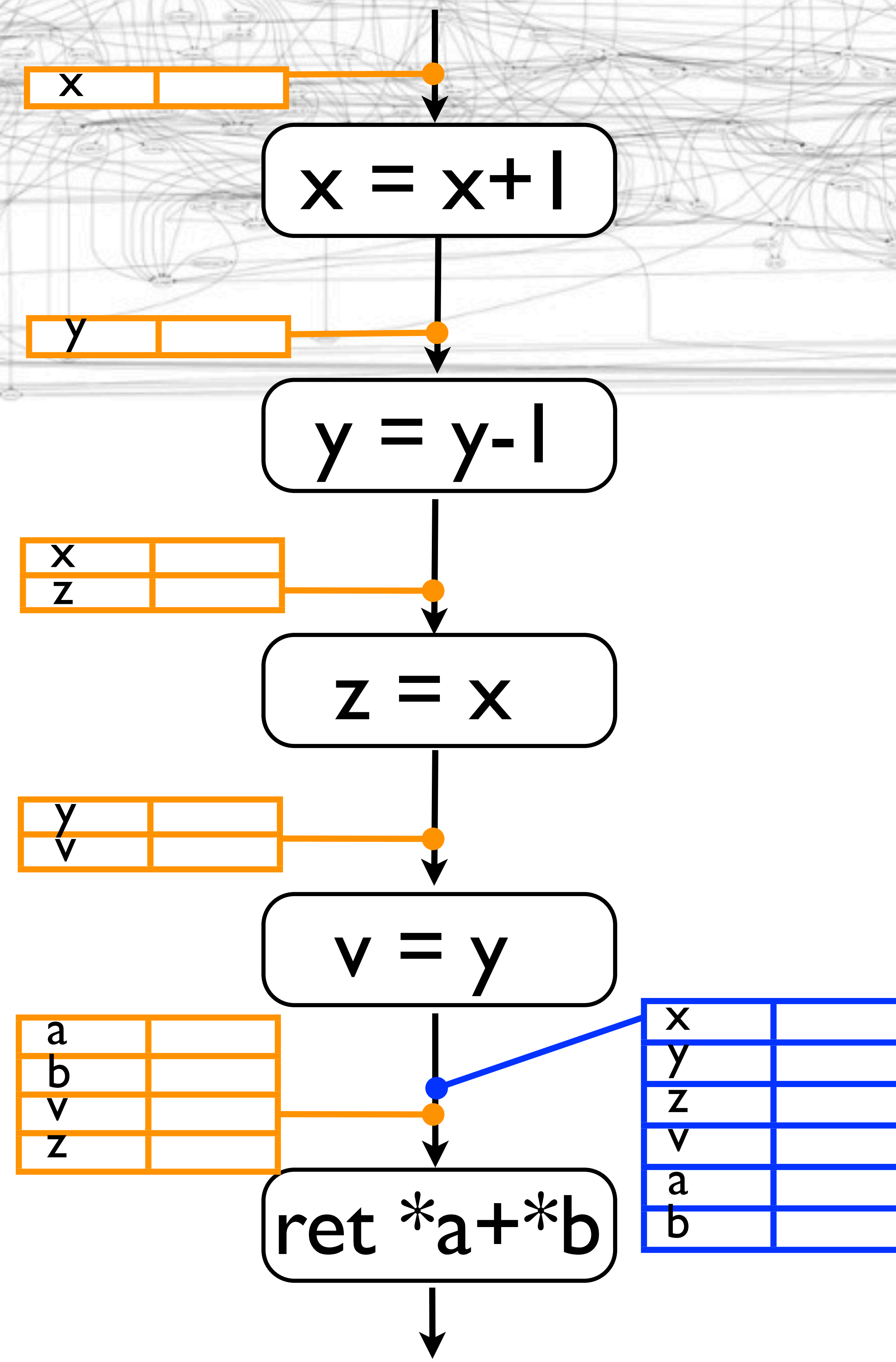
Spatial & Temporal Localizations



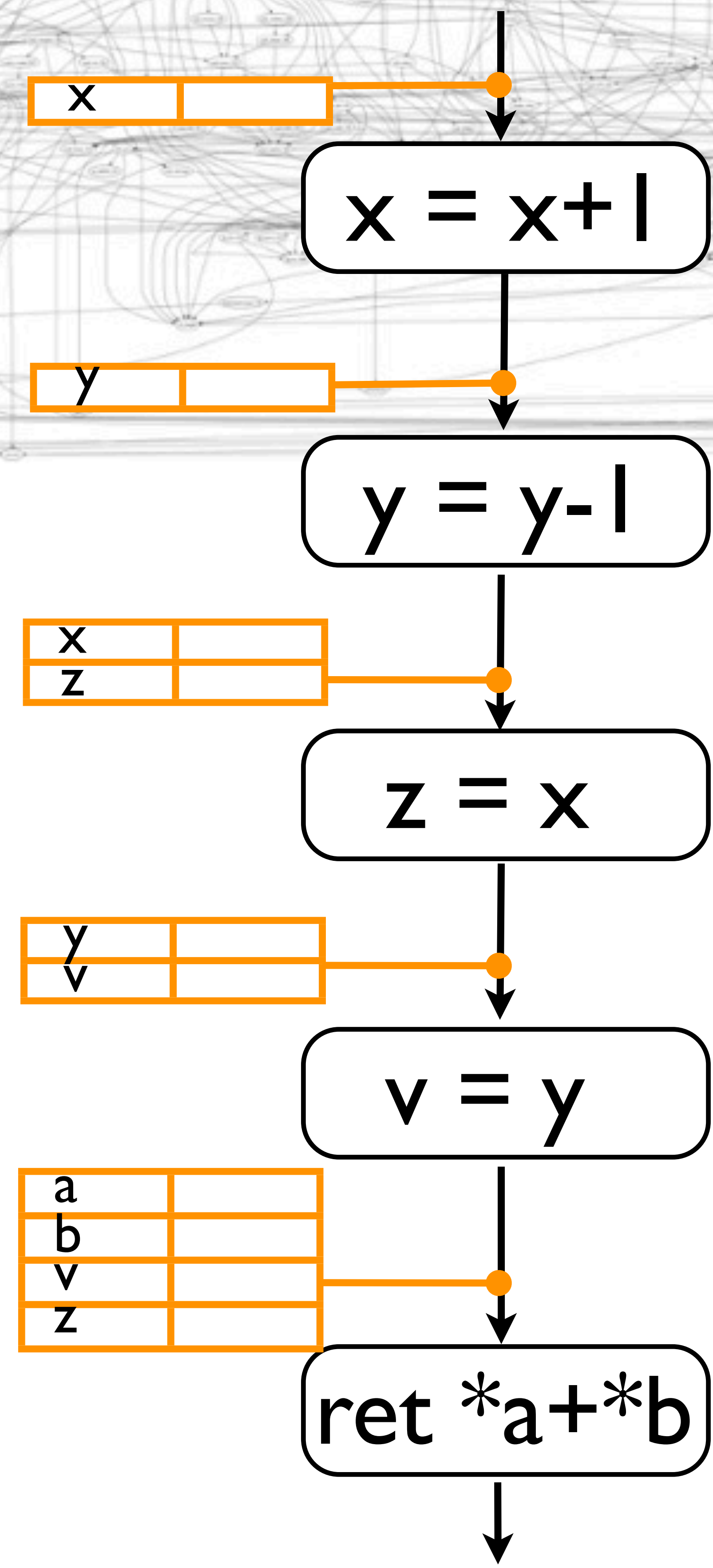
Spatial & Temporal Localizations



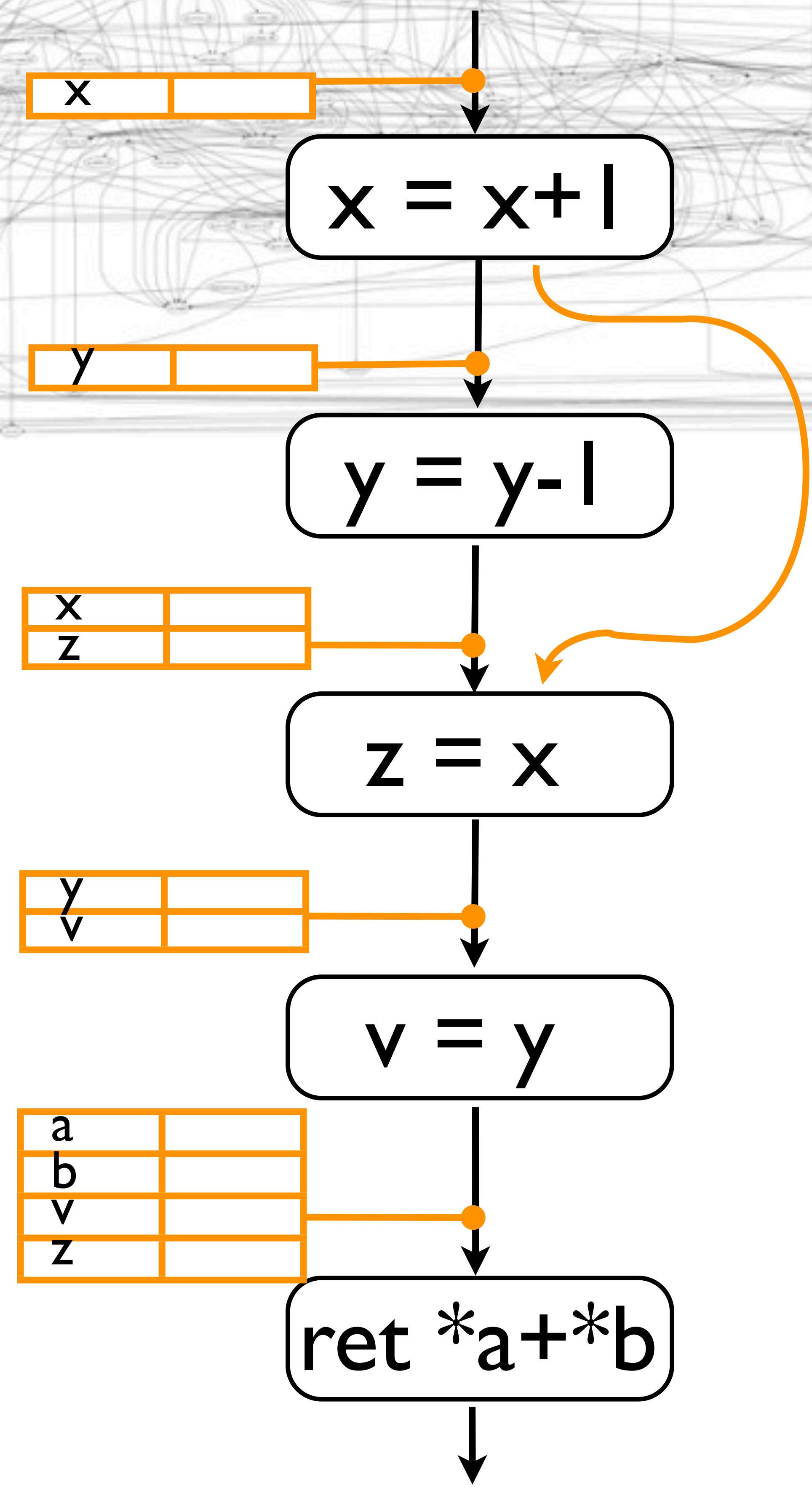
Spatial & Temporal Localizations



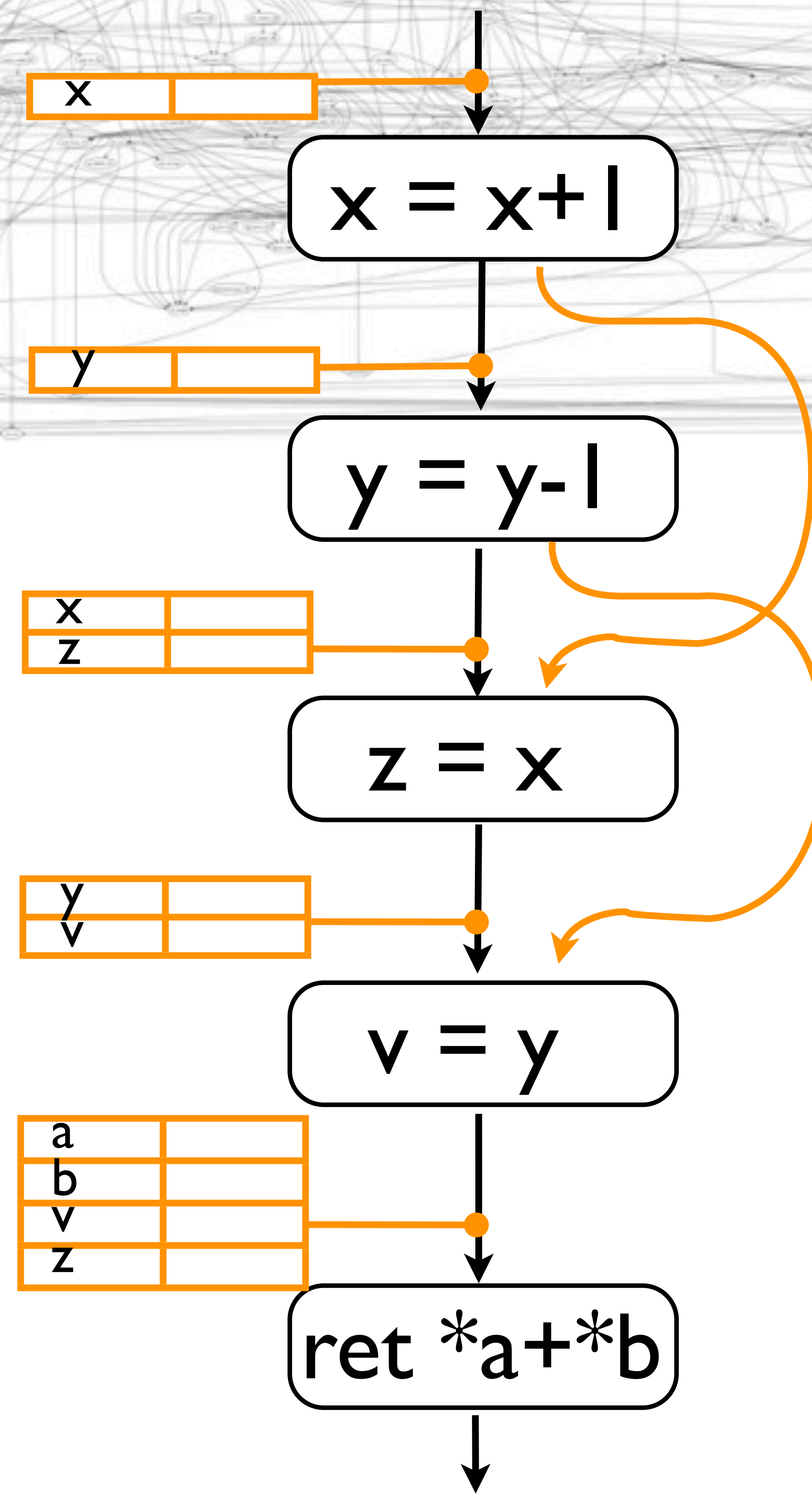
Spatial & Temporal Localizations



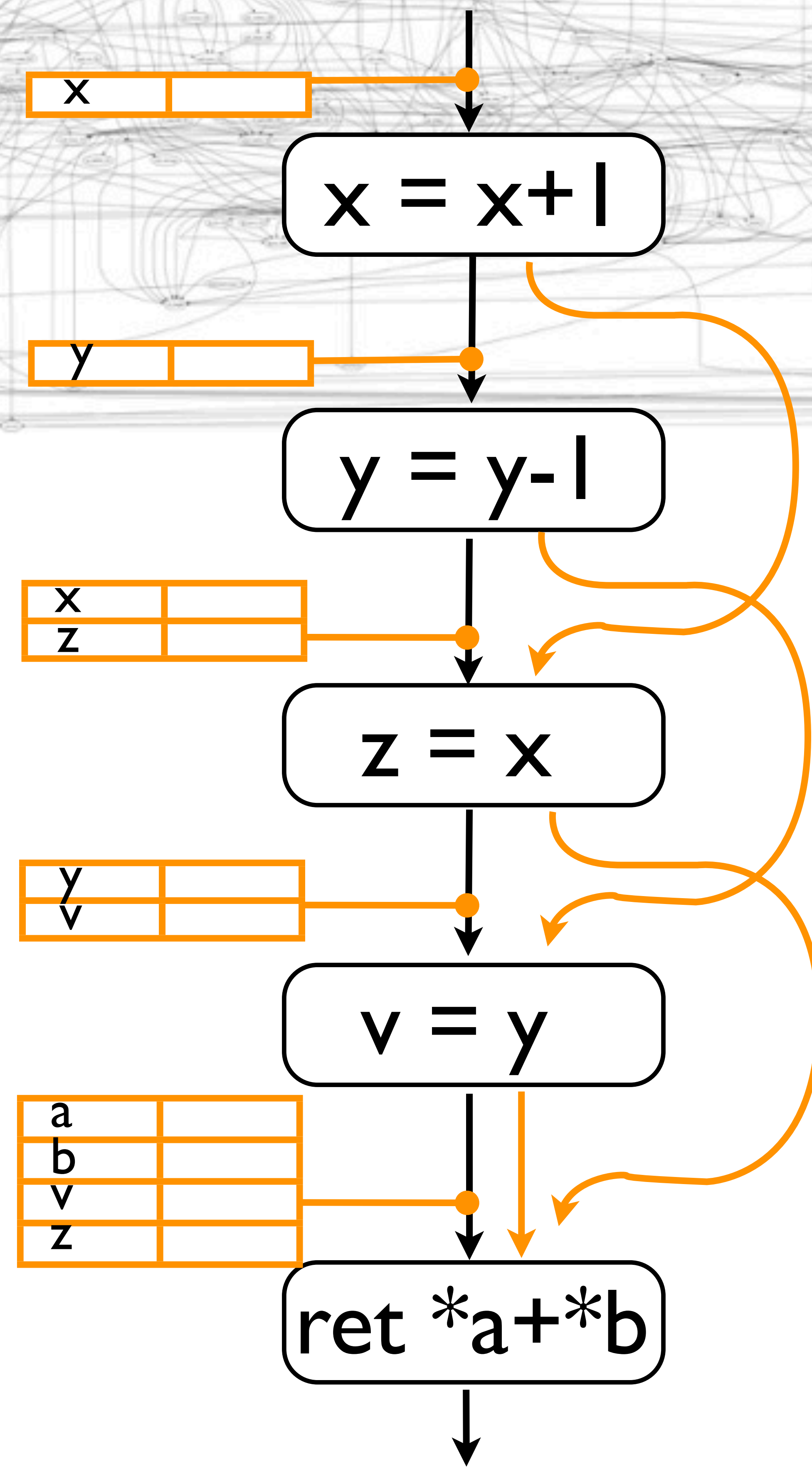
Spatial & Temporal Localizations



Spatial & Temporal Localizations



Spatial & Temporal Localizations



Spatial & Temporal Localizations

$$\hat{X}, \hat{X}' \in \mathbb{C} \to \hat{\mathbb{S}}$$
$$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$$

$$\hat{X} := \hat{X}' := \lambda c.\bot$$

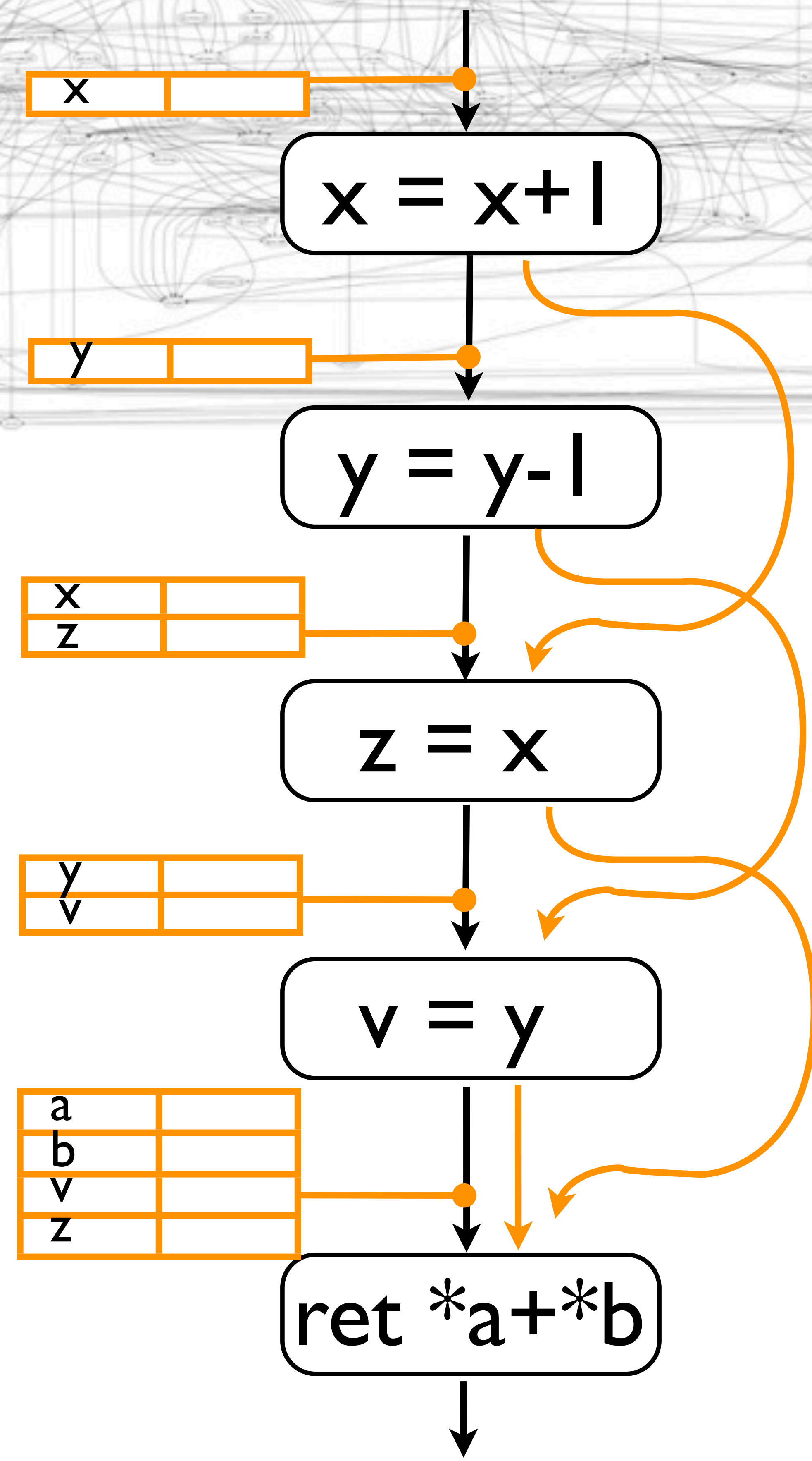
repeat

$$\hat{X}' := \hat{X}$$

for all $c \in \mathbb{C}$ **do**

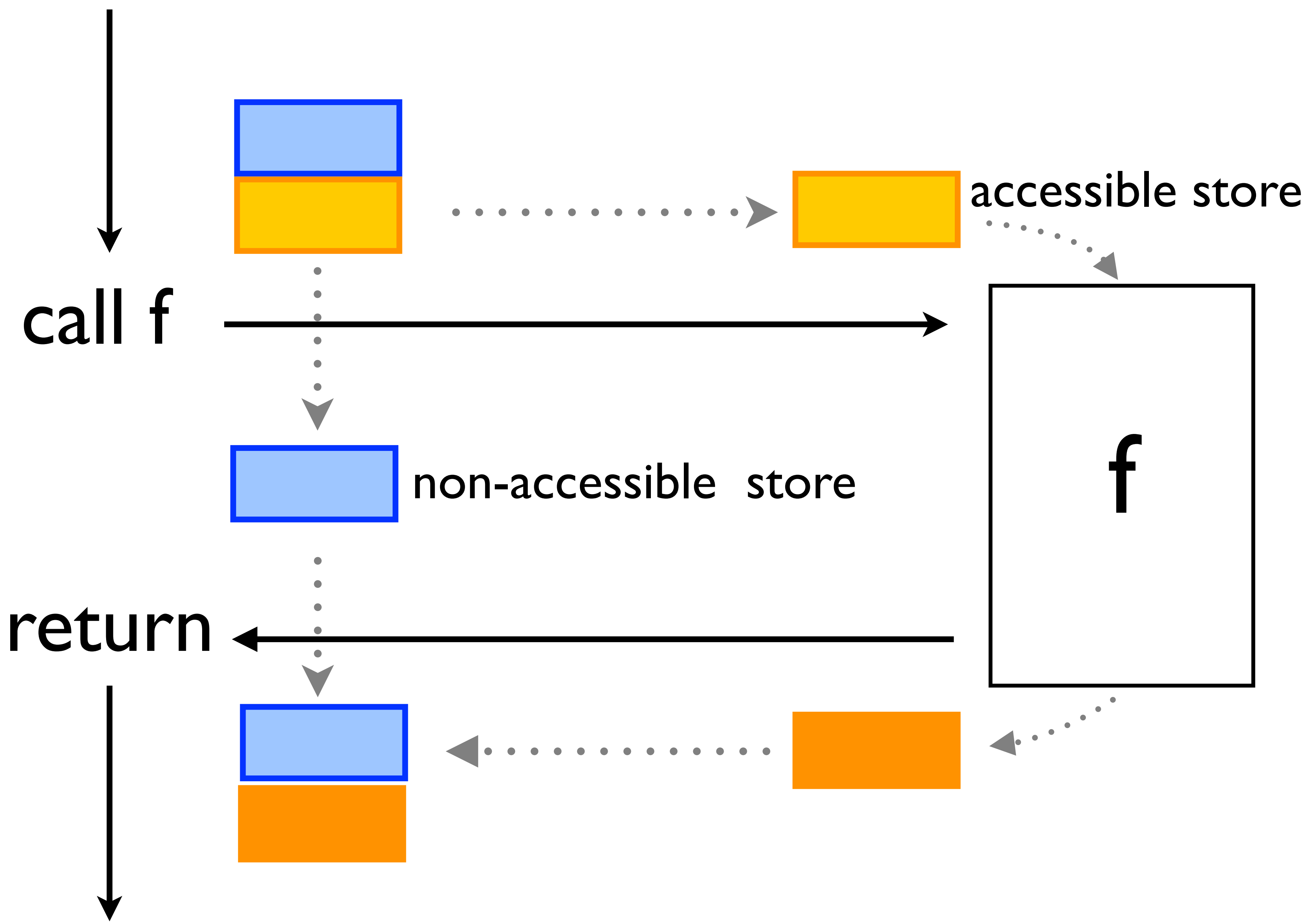
$$\hat{X}(c) := \hat{f}_c(\bigsqcup_{\boxed{c' \hookrightarrow c}} \boxed{\hat{X}(c')})$$

until $\hat{X} \sqsubseteq \hat{X}'$

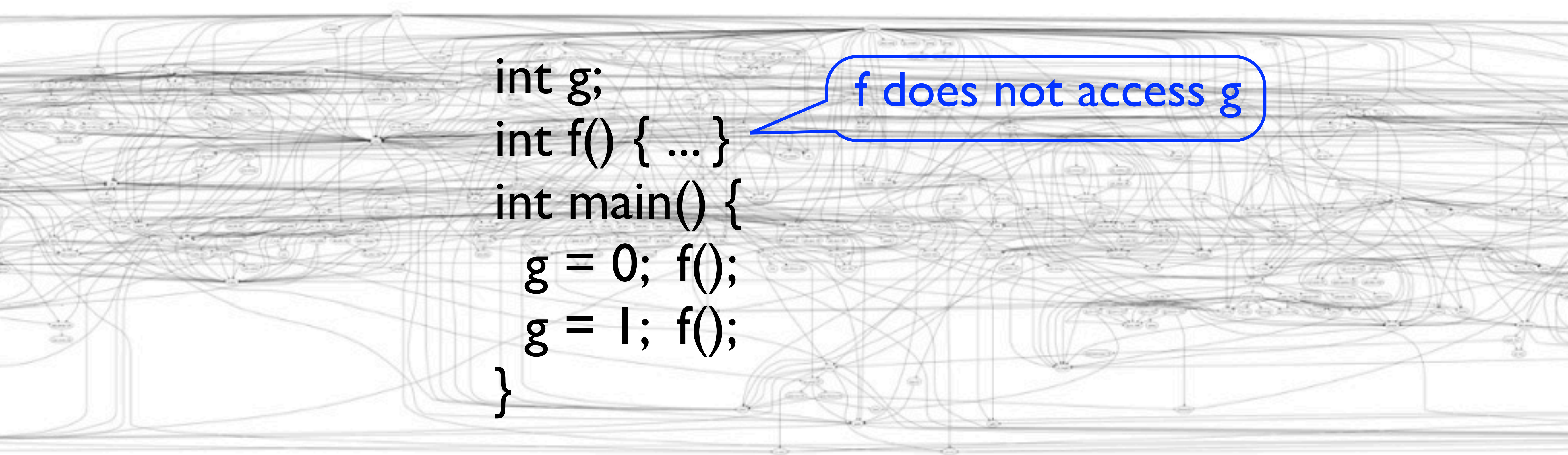


Spatial Localization

Spatial Localization
("framing", "abstract gc")



Vital in Analysis Practice



```
int g;
int f() { ... }
int main() {
   g = 0;  f();
   g = 1;  f();
}
```

f does not access g

On average, 755 re-analysis per procedure

But Existing Approach is Too Conservative

huge room for localizations than reachability-based technique

Program	LOC	accessed memory / reachable memory
spell-1.0	2,213	5 / 453 (1.1%)
barcode-0.96	4,460	19 / 1175 (1.6%)
httptunnel-3.3	6,174	10 / 673 (1.5%)
gzip-1.2.4a	7,327	22 / 1002 (2.2%)
jwhois-3.0.1	9,344	28 / 830 (3.4%)
parser	10,900	75 / 1787 (4.2%)
bc-1.06	13,093	24 / 824 (2.9%)
less-290	18,449	86 / 1546 (5.6%)

average : only 4%

Hurdle: Accessed Locations Before Analysis?

- Yes, by yet another analysis
- The pre-analysis must be quick
- The pre-analysis must be safe
  - over-estimating the accessed abstract locs

Our Pre-analysis

For Safely Estimating the Accessed Abstract Locations

- one further abstraction
- correct design

$$\mathbb{C} \to \hat{\mathbb{S}} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{\mathbb{S}}$$

- abstract semantic function: flow-insensitive

$$\hat{F}_p = \lambda \hat{s}.(\bigsqcup_{c \in \mathbb{C}} \hat{f}_c(\hat{s}))$$

Implement in the Abstract Semantics for Call Cmd

$$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$$

$$\hat{f}_c(\hat{s}) = \begin{cases} \hat{s}[\hat{\mathcal{L}}(lv)(\hat{s}) \overset{w}{\mapsto} \hat{\mathcal{V}}(e)(\hat{s})] & \mathsf{cmd}(c) = lv := e \\ \hat{s}[\hat{\mathcal{L}}(lv)(\hat{s}) \overset{w}{\mapsto} \langle \bot, \bot, \{\langle l, [0,0], \hat{\mathcal{V}}(e)(\hat{s}).1\rangle\}, \bot\rangle] & \mathsf{cmd}(c) = lv := \mathsf{alloc}(\texttt{[}e\texttt{]}_l) \\ \hat{s}[\hat{\mathcal{L}}(lv)(\hat{s}) \overset{w}{\mapsto} \langle \bot, \bot, \bot, \{\langle l, \{x\}\rangle\}\rangle] & \mathsf{cmd}(c) = lv := \mathsf{alloc}(\texttt{\{}x\texttt{\}}_l) \\ \hat{s}[x \mapsto \langle \hat{s}(x).1 \sqcap [-\infty, \mathsf{u}(\hat{\mathcal{V}}(e)(\hat{s}).1)], \hat{s}(x).2, \hat{s}(x).3, \hat{s}(x).4\rangle] & \mathsf{cmd}(c) = \mathsf{assume}(x < e) \\ \hat{s}[x \mapsto \hat{\mathcal{V}}(e)(\hat{s})]\big|_{\mathsf{access}(f)} & \mathsf{cmd}(c) = \mathsf{call}(f_x, e) \\ \hat{s} & \mathsf{cmd}(c) = \mathsf{return}_f \end{cases}$$

$$\mathsf{access}(f) = \bigcup_{g \in \mathsf{callees}(f)} \left(\bigcup_{c \in \mathsf{control}(g)} \mathcal{A}(c)(\hat{s}_{pre})\right)$$

Performance of sound & global Sparrow

The Early Bird

Programs	LOC	Interval$_{vanilla}$		Interval$_{base}$		Spd$\uparrow_1$	Mem$\downarrow_1$	Interval$_{sparse}$						Spd$\uparrow_2$	Mem$\downarrow_2$
		Time	Mem	Time	Mem			Dep	Fix	Total	Mem	$\hat{D}(c)$	$\hat{U}(c)$		
gzip-1.2.4a	7K	772	240	14	65	55 x	73 %	2	1	3	63	2.4	2.5	5 x	3 %
bc-1.06	13K	1,270	276	96	126	13 x	54 %	4	3	7	75	4.6	4.9	14 x	40 %
tar-1.13	20K	12,947	881	338	177	38 x	80 %	6	2	8	93	2.9	2.9	42 x	47 %
less-382	23K	9,561	1,113	1,211	378	8 x	66 %	27	6	33	127	11.9	11.9	37 x	66 %
make-3.76.1	27K	24,240	1,391	1,893	443	13 x	68 %	16	5	21	114	5.8	5.8	90 x	74 %
wget-1.9	35K	44,092	2,546	1,214	378	36 x	85 %	8	3	11	85	2.4	2.4	110 x	78 %
screen-4.0.2	45K	∞	N/A	31,324	3,996	N/A	N/A	724	43	767	303	53.0	54.0	41 x	92 %
a2ps-4.14	64K	∞	N/A	3,200	1,392	N/A	N/A	31	9	40	353	2.6	2.8	80 x	75 %
bash-2.05a	105K	∞	N/A	1,683	1,386	N/A	N/A	45	22	67	220	3.0	3.0	25 x	84 %
lsh-2.0.4	111K	∞	N/A	45,522	5,266	N/A	N/A	391	80	471	577	21.1	21.2	97 x	89 %
sendmail-8.13.6	130K	∞	N/A	∞	N/A	N/A	N/A	517	227	744	678	20.7	20.7	N/A	N/A
nethack-3.3.0	211K	∞	N/A	∞	N/A	N/A	N/A	14,126	2,247	16,373	5,298	72.4	72.4	N/A	N/A
vim60	227K	∞	N/A	∞	N/A	N/A	N/A	17,518	6,280	23,798	5,190	180.2	180.3	N/A	N/A
emacs-22.1	399K	∞	N/A	∞	N/A	N/A	N/A	29,552	8,278	37,830	7,795	285.3	285.5	N/A	N/A
python-2.5.1	435K	∞	N/A	∞	N/A	N/A	N/A	9,677	1,362	11,039	5,535	108.1	108.1	N/A	N/A
linux-3.0	710K	∞	N/A	∞	N/A	N/A	N/A	26,669	6,949	33,618	20,529	76.2	74.8	N/A	N/A
gimp-2.6	959K	∞	N/A	∞	N/A	N/A	N/A	3,751	123	3,874	3,602	4.1	3.9	N/A	N/A
ghostscript-9.00	1,363K	∞	N/A	∞	N/A	N/A	N/A	14,116	698	14,814	6,384	9.7	9.7	N/A	N/A

none

spatial localization

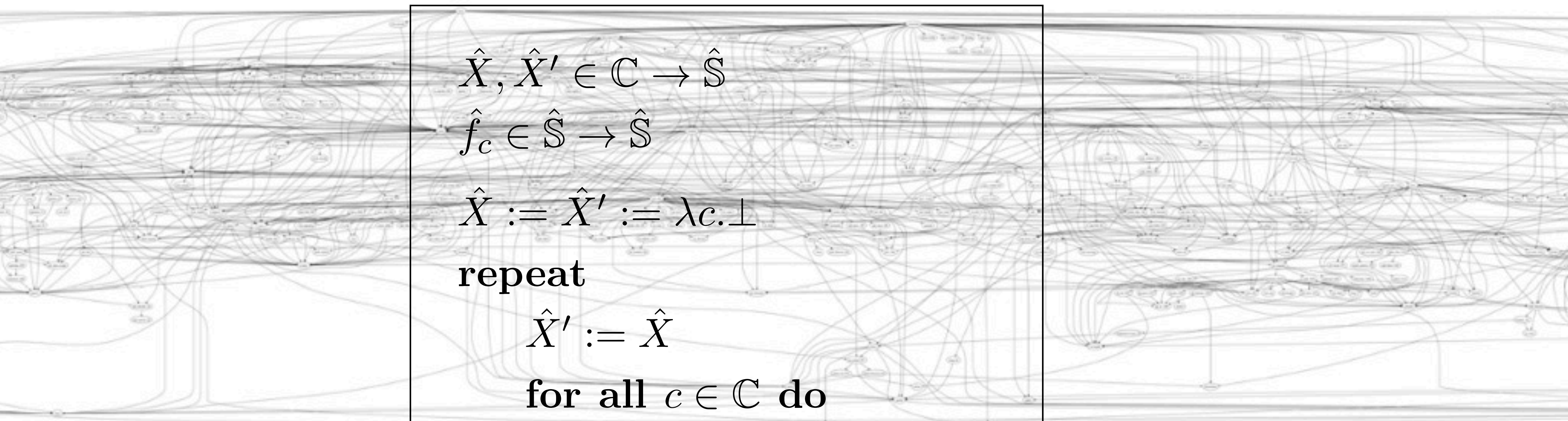
spatial+temporal localization

Temporal Localization

(and spatial localization automatically follows)

Temporal Localization

- Don't blindly follow the control flow of pgm text
- Follow the dependency of statement semantics
  - from definition points directly to their use points



$\hat{X}, \hat{X}' \in \mathbb{C} \to \hat{\mathbb{S}}$

$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$

$\hat{X} := \hat{X}' := \lambda c.\bot$

repeat

$\quad \hat{X}' := \hat{X}$

$\quad$ **for all** $c \in \mathbb{C}$ **do**

$\quad\quad \hat{X}(c) := \hat{f}_c(\bigsqcup_{\boxed{c' \hookrightarrow c}} \hat{X}(c'))$

until $\hat{X} \sqsubseteq \hat{X}'$

Temporal Localization

- Don't blindly follow the control flow of pgm text
- Follow the dependency of statement semantics
  - from definition points directly to their use points

$$\hat{X}, \hat{X}' \in \mathbb{C} \to \hat{\mathbb{S}}$$
$$\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$$

$$\hat{X} := \hat{X}' := \lambda c.\bot$$

repeat

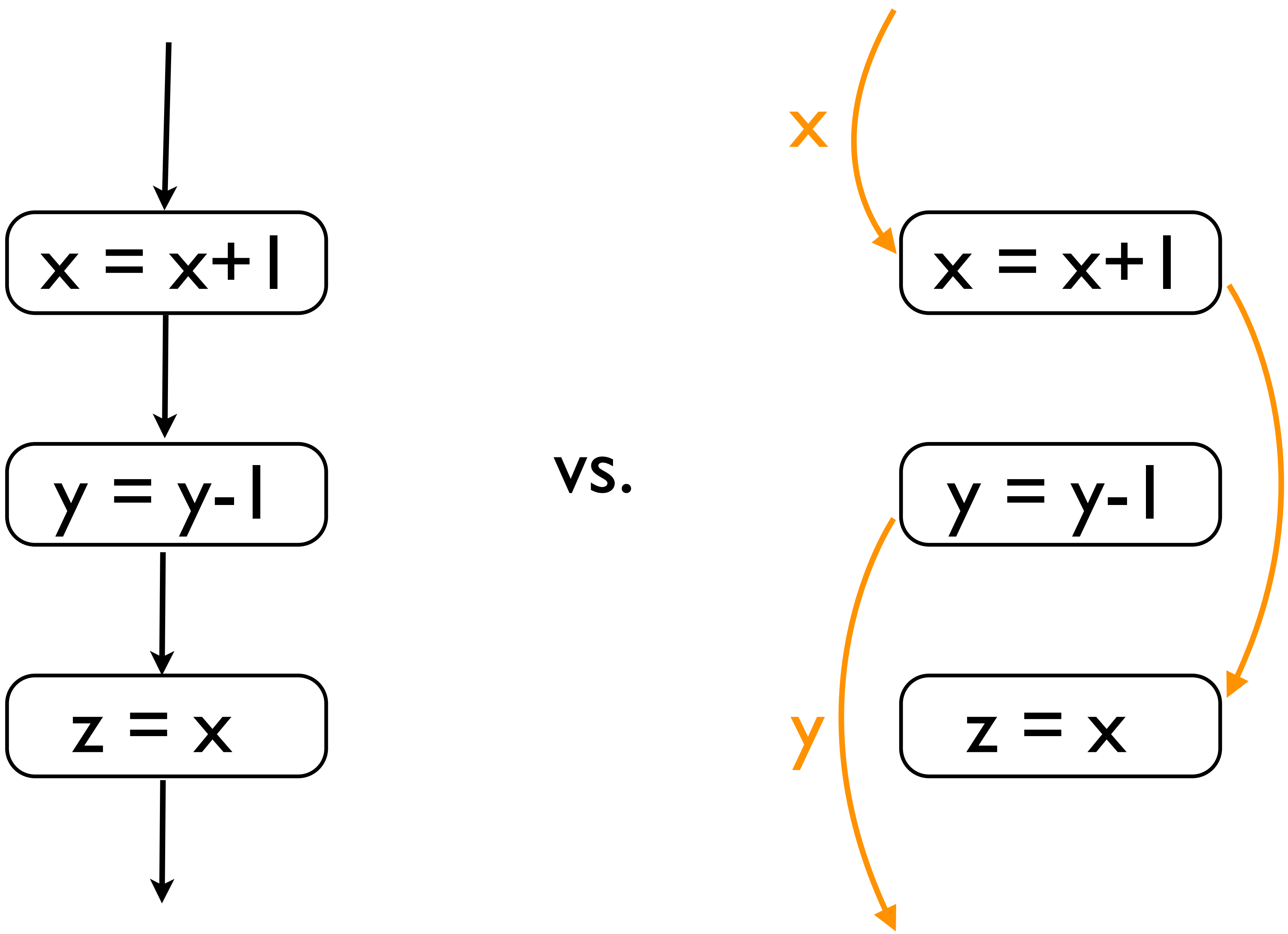
$\quad \hat{X}' := \hat{X}$

$\quad$ **for all** $c \in \mathbb{C}$ **do**

$\quad\quad \hat{X}(c) := \hat{f}_c(\bigsqcup_{\boxed{c' \hookrightarrow c}} \hat{X}(c'))$

until $\hat{X} \sqsubseteq \hat{X}'$

Temporal Localization



Precision Preserving Sparse Analysis Framework

$$\hat{F} : \hat{D} \to \hat{D} \quad \stackrel{\text{sparsify}}{\Longrightarrow} \quad \hat{F}_s : \hat{D} \to \hat{D}$$

$$\mathit{fix}\,\hat{F} \quad \stackrel{\text{still}}{=} \quad \mathit{fix}\,\hat{F}_s$$

Towards Sparse Version

Analyzer computes the fixpoint of $\hat{F}_{-} \in (\mathbb{C} \to \hat{\mathbb{S}}) \to (\mathbb{C} \to \hat{\mathbb{S}})$

- baseline non-sparse one

$$\hat{F}(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \hookrightarrow c}} \hat{X}(c')).$$

- unrealizable sparse version

$$\hat{F}_s(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\leadsto} c}} \hat{X}(c')|_l).$$

- realizable sparse version

$$\hat{F}_a(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\leadsto}_a c}} \hat{X}(c')|_l).$$

Unrealizable Sparse One

$$\hat{F}_s(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\rightsquigarrow} c}} \hat{X}(c')|_l).$$

Data Dependency

$$c_0 \overset{l}{\rightsquigarrow} c_n \quad \triangleq \quad \exists c_0 \ldots c_n \in \textsf{Paths}, l \in \hat{\mathbb{L}}.$$
$$l \in \textsf{D}(c_0) \cap \textsf{U}(c_n) \wedge \forall i \in (0, n).l \notin \textsf{D}(c_i)$$

Unrealizable Sparse One

$$\hat{F}_s(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c\Big(\bigsqcup_{\boxed{c' \overset{l}{\rightsquigarrow} c}} \hat{X}(c')|_l\Big).$$

Data Dependency

$$c_0 \overset{l}{\rightsquigarrow} c_n \quad \triangleq \quad \exists c_0 \ldots c_n \in \mathsf{Paths}, l \in \hat{\mathbb{L}}.$$
$$l \in \mathsf{D}(c_0) \cap \mathsf{U}(c_n) \wedge \forall i \in (0, n). l \notin \mathsf{D}(c_i)$$

Def-Use Sets

$$\mathsf{D}(c) \triangleq \{l \in \hat{\mathbb{L}} \mid \exists \hat{s} \sqsubseteq \bigsqcup_{c' \hookrightarrow c} \mathcal{S}(c'). \hat{f}_c(\hat{s})(l) \neq \hat{s}(l)\}$$

$$\mathsf{U}(c) \triangleq \{l \in \hat{\mathbb{L}} \mid \exists \hat{s} \sqsubseteq \bigsqcup_{c' \hookrightarrow c} \mathcal{S}(c'). \hat{f}_c(\hat{s})|_{\mathsf{D}(c)} \neq \hat{f}_c(\hat{s} \backslash_l)|_{\mathsf{D}(c)}\}$$

Unrealizable Sparse One

$$\hat{F}_s(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\rightsquigarrow} c}} \hat{X}(c')|_l).$$

Data Dependency

$$\begin{aligned} c_0 \overset{l}{\rightsquigarrow} c_n \quad &\triangleq \quad \exists c_0 \ldots c_n \in \mathsf{Paths}, l \in \hat{\mathbb{L}}. \\ &\qquad l \in \mathsf{D}(c_0) \cap \mathsf{U}(c_n) \wedge \forall i \in (0, n).l \notin \mathsf{D}(c_i) \end{aligned}$$

Def-Use Sets

$$\mathsf{D}(c) \triangleq \{l \in \hat{\mathbb{L}} \mid \exists \hat{s} \sqsubseteq \bigsqcup_{c' \hookrightarrow c} \mathcal{S}(c').\hat{f}_c(\hat{s})(l) \neq \hat{s}(l)\}$$

$$\mathsf{U}(c) \triangleq \{l \in \hat{\mathbb{L}} \mid \exists \hat{s} \sqsubseteq \bigsqcup_{c' \hookrightarrow c} \mathcal{S}(c').\hat{f}_c(\hat{s})|_{\mathsf{D}(c)} \neq \hat{f}_c(\hat{s}\backslash_l)|_{\mathsf{D}(c)}\}$$

Precision Preserving

$$\mathit{fix}\hat{F} = \mathit{fix}\hat{F}_s \quad \text{modulo D}$$

Data Dependency Example

$$\begin{tikzcd} x := \&y \arrow[r] \arrow[rr, bend left, "x"] & *p := \&z \arrow[r] & y := x \end{tikzcd}$$

Def	$\{x\}$	$\{v, w\}$	$\{y\}$
Use	ϕ	$\{p, v, w\}$	$\{x\}$

Realizable Sparse One

$$\hat{F}_a(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\rightsquigarrow}_a c}} \hat{X}(c')|_l).$$

Realizable Data Dependency

$$c_0 \overset{l}{\rightsquigarrow}_a c_n \quad \triangleq \quad \exists c_0 \ldots c_n \in \mathsf{Paths}, l \in \hat{\mathbb{L}}.$$
$$l \in \underline{\hat{\mathsf{D}}}(c_0) \cap \underline{\hat{\mathsf{U}}}(c_n) \wedge \forall i \in (0, n).l \notin \hat{\mathsf{D}}(c_i)$$

Realizable Sparse One

$$\hat{F}_a(\hat{X}) = \lambda c \in \mathbb{C}.\hat{f}_c(\bigsqcup_{\boxed{c' \overset{l}{\rightsquigarrow}_a c}} \hat{X}(c')|_l).$$

Realizable Data Dependency

$$c_0 \overset{l}{\rightsquigarrow}_a c_n \quad \triangleq \quad \exists c_0 \ldots c_n \in \mathsf{Paths}, l \in \hat{\mathbb{L}}.$$
$$l \in \underline{\hat{\mathsf{D}}}(c_0) \cap \underline{\hat{\mathsf{U}}}(c_n) \wedge \forall i \in (0, n).l \notin \hat{\mathsf{D}}(c_i)$$

Precision Preserving

$$\mathit{fix}\hat{F} = \mathit{fix}\hat{F}_a \quad \text{modulo } \hat{\mathsf{D}}$$

If the following conditions hold

Conditions on $\hat{\mathsf{D}}$ & $\hat{\mathsf{U}}$

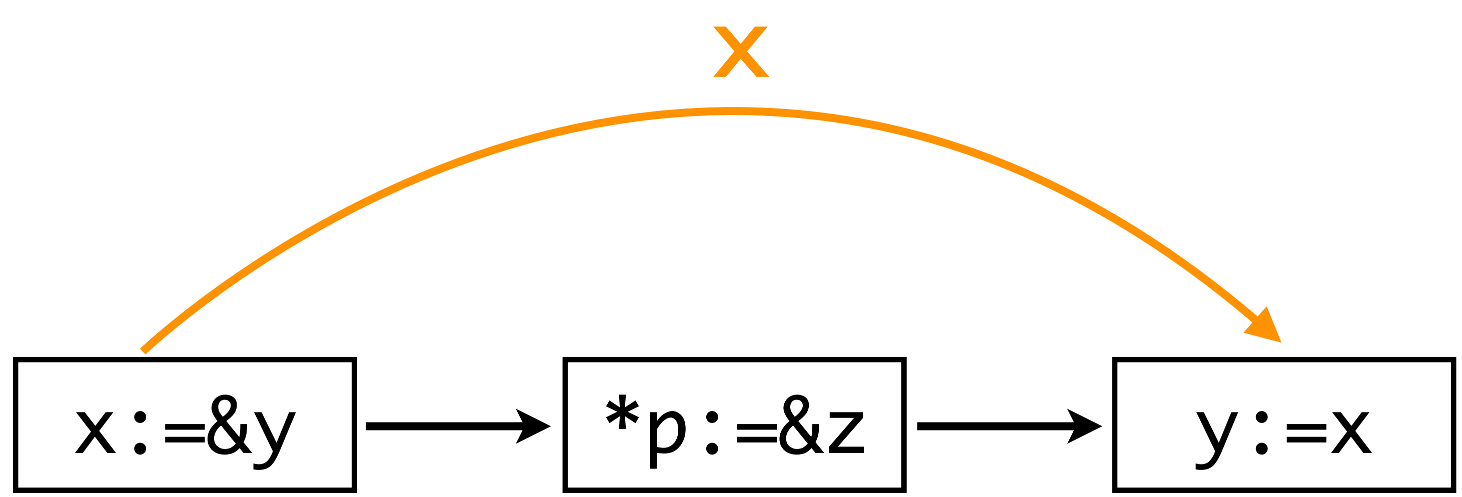
- over-approximation

$$\hat{\mathsf{D}}(c) \supseteq \mathsf{D}(c) \wedge \hat{\mathsf{U}}(c) \supseteq \mathsf{U}(c)$$

- spurious definitions should be also included in uses

$$\hat{\mathsf{D}}(c) - \mathsf{D}(c) \subseteq \hat{\mathsf{U}}(c)$$

Why the Conditions of $\hat{D}$ & $\hat{U}$



Def	{x}	{v, w}	{y}
Use	ϕ	{p, v, w}	{x}

Why the Conditions of $\hat{D}$ & $\hat{U}$

$$
\begin{tikzcd}
\boxed{x:=\&y} \arrow[r] \arrow[rr, bend left, "x"] & \boxed{*p:=\&z} \arrow[r] & \boxed{y:=x}
\end{tikzcd}
$$

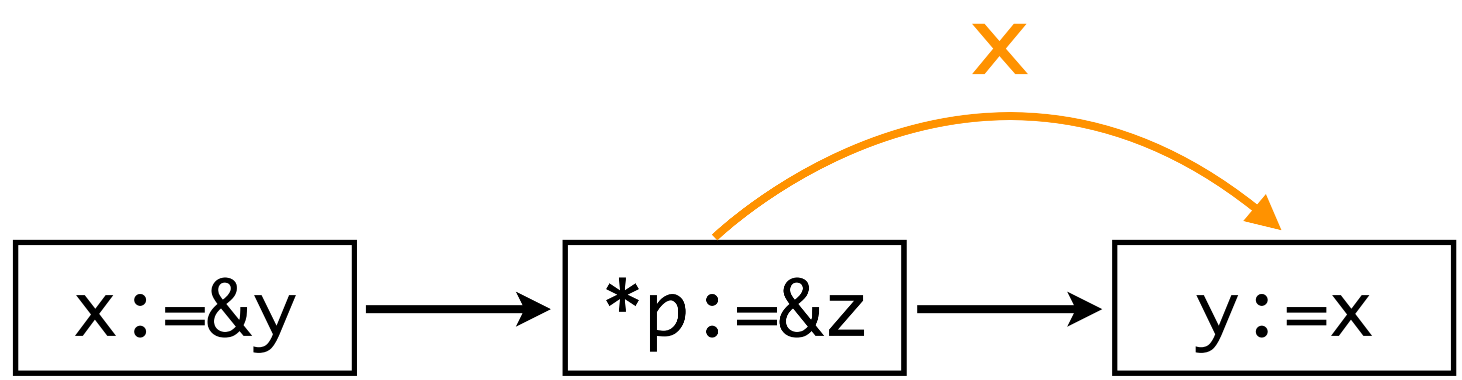
Def	{x}	{v, w, x}	{y}
Use	ϕ	{p, v, w}	{x}

Why the Conditions of $\hat{D}$ & $\hat{U}$

$\boxed{x := \&y} \longrightarrow \boxed{*p := \&z} \longrightarrow \boxed{y := x}$

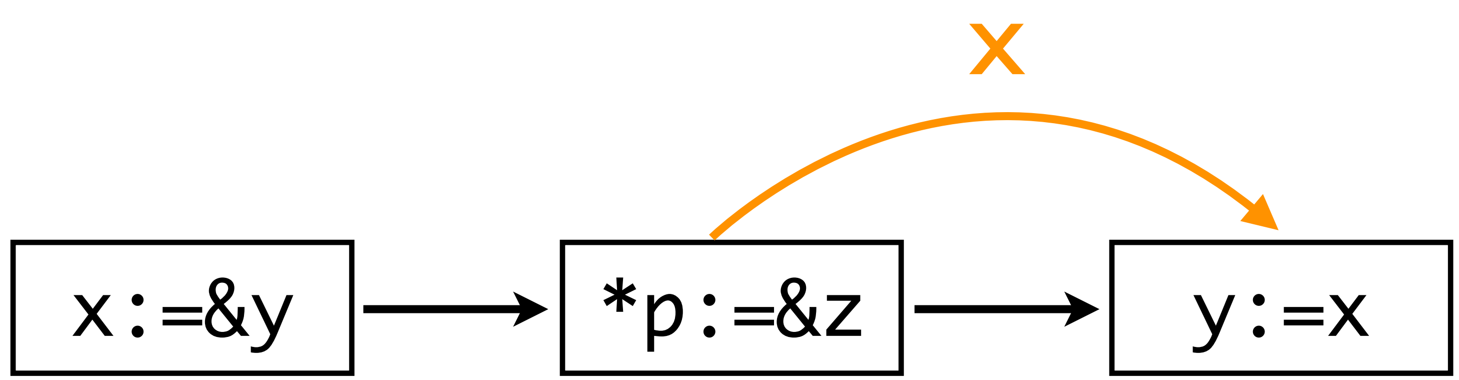
	x:=&y	*p:=&z	y:=x
Def	{x}	{v, w, x}	{y}
Use	ϕ	{p, v, w}	{x}

Why the Conditions of $\hat{D}$ & $\hat{U}$



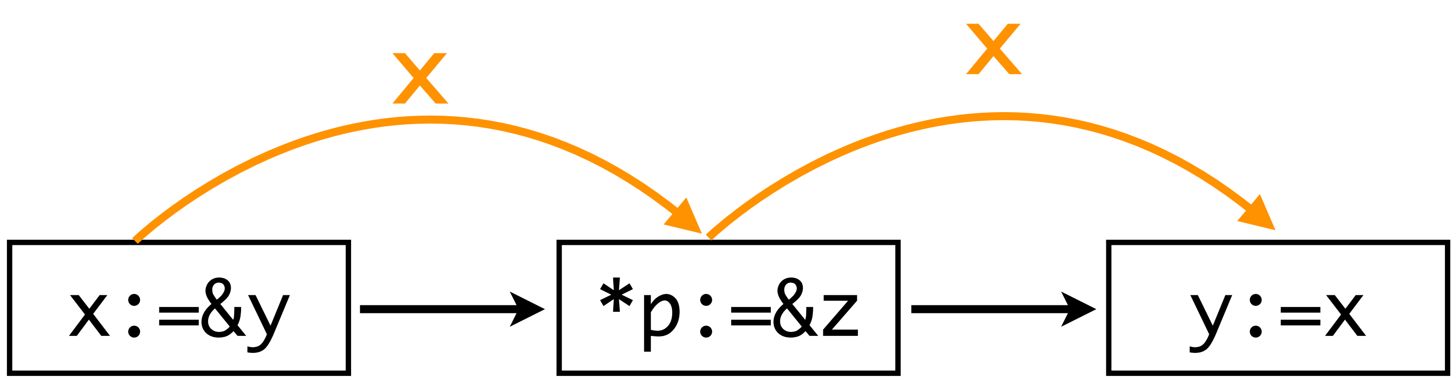
Def	{x}	{v, w, x}	{y}
Use	ϕ	{p, v, w}	{x}

Why the Conditions of $\hat{D}$ & $\hat{U}$



Def	{x}	{v, w, x}	{y}
Use	ϕ	{p, v, w, x}	{x}

Why the Conditions of $\hat{D}$ & $\hat{U}$



Def	{x}	{v, w, x}	{y}
Use	φ	{p, v, w, x}	{x}

Hurdle: $\hat{\mathrm{D}}$ & $\hat{\mathrm{U}}$ Before Analysis?

- Yes, by yet another analysis with further abstraction
- correct design

$$\mathbb{C} \to \hat{\mathbb{S}} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{\mathbb{S}}$$

- abstract semantic function: flow-insensitive

$$\hat{F}_p = \lambda\hat{s}.(\bigsqcup_{c \in \mathbb{C}} \hat{f}_c(\hat{s}))$$

Performance of sound & global Sparrow

The Early Bird

Programs	LOC	Interval$_{vanilla}$		Interval$_{base}$		Spd$\uparrow_1$	Mem$\downarrow_1$	Interval$_{sparse}$						Spd$\uparrow_2$	Mem$\downarrow_2$
		Time	Mem	Time	Mem			Dep	Fix	Total	Mem	$\hat{D}(c)$	$\hat{U}(c)$		
gzip-1.2.4a	7K	772	240	14	65	55 x	73 %	2	1	3	63	2.4	2.5	5 x	3 %
bc-1.06	13K	1,270	276	96	126	13 x	54 %	4	3	7	75	4.6	4.9	14 x	40 %
tar-1.13	20K	12,947	881	338	177	38 x	80 %	6	2	8	93	2.9	2.9	42 x	47 %
less-382	23K	9,561	1,113	1,211	378	8 x	66 %	27	6	33	127	11.9	11.9	37 x	66 %
make-3.76.1	27K	24,240	1,391	1,893	443	13 x	68 %	16	5	21	114	5.8	5.8	90 x	74 %
wget-1.9	35K	44,092	2,546	1,214	378	36 x	85 %	8	3	11	85	2.4	2.4	110 x	78 %
screen-4.0.2	45K	∞	N/A	31,324	3,996	N/A	N/A	724	43	767	303	53.0	54.0	41 x	92 %
a2ps-4.14	64K	∞	N/A	3,200	1,392	N/A	N/A	31	9	40	353	2.6	2.8	80 x	75 %
bash-2.05a	105K	∞	N/A	1,683	1,386	N/A	N/A	45	22	67	220	3.0	3.0	25 x	84 %
lsh-2.0.4	111K	∞	N/A	45,522	5,266	N/A	N/A	391	80	471	577	21.1	21.2	97 x	89 %
sendmail-8.13.6	130K	∞	N/A	∞	N/A	N/A	N/A	517	227	744	678	20.7	20.7	N/A	N/A
nethack-3.3.0	211K	∞	N/A	∞	N/A	N/A	N/A	14,126	2,247	16,373	5,298	72.4	72.4	N/A	N/A
vim60	227K	∞	N/A	∞	N/A	N/A	N/A	17,518	6,280	23,798	5,190	180.2	180.3	N/A	N/A
emacs-22.1	399K	∞	N/A	∞	N/A	N/A	N/A	29,552	8,278	37,830	7,795	285.3	285.5	N/A	N/A
python-2.5.1	435K	∞	N/A	∞	N/A	N/A	N/A	9,677	1,362	11,039	5,535	108.1	108.1	N/A	N/A
linux-3.0	710K	∞	N/A	∞	N/A	N/A	N/A	26,669	6,949	33,618	20,529	76.2	74.8	N/A	N/A
gimp-2.6	959K	∞	N/A	∞	N/A	N/A	N/A	3,751	123	3,874	3,602	4.1	3.9	N/A	N/A
ghostscript-9.00	1,363K	∞	N/A	∞	N/A	N/A	N/A	14,116	698	14,814	6,384	9.7	9.7	N/A	N/A

none

spatial localization

spatial+temporal localization

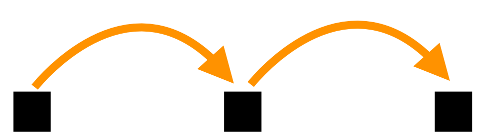
Existing Sparse Techniques

(developed mostly in dfa community)

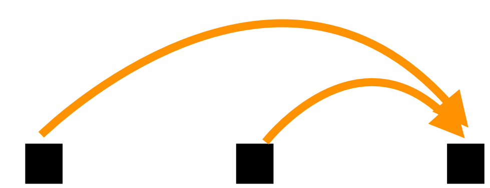
- Different notion of data dependency

$$
\begin{aligned}
c_0 \overset{l}{\rightsquigarrow}_{\mathsf{du}} c_n \;\triangleq\; & \exists c_0 \ldots c_n \in \mathsf{Paths}, l \in \hat{\mathbb{L}}. \\
& l \in \mathsf{D}(c_0) \cap \mathsf{U}(c_n) \wedge \forall i \in (0, n). l \notin \underline{\mathsf{D}_{\mathsf{always}}(c_i)}
\end{aligned}
$$

  - fail to preserve the original accuracy



vs.



- Not general for arbitrary analysis for full C
  - tightly coupled with particular analysis (e.g. pointer analysis for "simple" subsets of C)

Summing Up

Our Sparse Framework

- Define a global safe abstract interpreter

$$\hat{F} \in (\mathbb{C} \to \hat{\mathbb{S}}) \to (\mathbb{C} \to \hat{\mathbb{S}})$$

$$\hat{F}(\hat{X}) = \lambda c \in \mathbb{C}. \hat{f}_c(\bigsqcup_{c' \hookrightarrow c} \hat{X}(c')).$$

- Make it sparse (s&t) with our data dependencies
  - by using a safe pre-analysis for safe def/use sets

$$\hat{F}_a(\hat{X}) = \lambda c \in \mathbb{C}. \hat{f}_c(\bigsqcup_{c' \overset{l}{\rightsquigarrow}_a c} \hat{X}(c')|_l).$$

- Resulting sparse one scales with the same result

Thank you.

for technical details

ropas.snu.ac.kr/~kwang/paper/12-pldi-ohheleleyi.pdf