EXERCISE #11

LATTICES REVIEW

Write your name and answer the following on a piece of paper

- Recall that the set of English words, ranked by substring inclusion, forms a poset but NOT a lattice. Explain why and give an example

Reflexive: every word includes the substring of itself

Anti-symmetric: if two words include substrings of each other, they must be the same word

Transitive: if a is a substring b, and b is a substring of c, then a is a substring of c

Not a lattice: consider the English words a and he. They have no greatest lower bound

ADMINISTRIVIA AND ANNOUNCEMENTS

DATAFLOW FRAMEWORKS

EECS 677: Software Security Evaluation

Drew Davidson

DATAFLOW: THE BIG IDEAS

DATAFLOW FRAMEWORKS

KEY CFG ANNOTATIONS

Mark each block to indicate all possible values coming into that program point

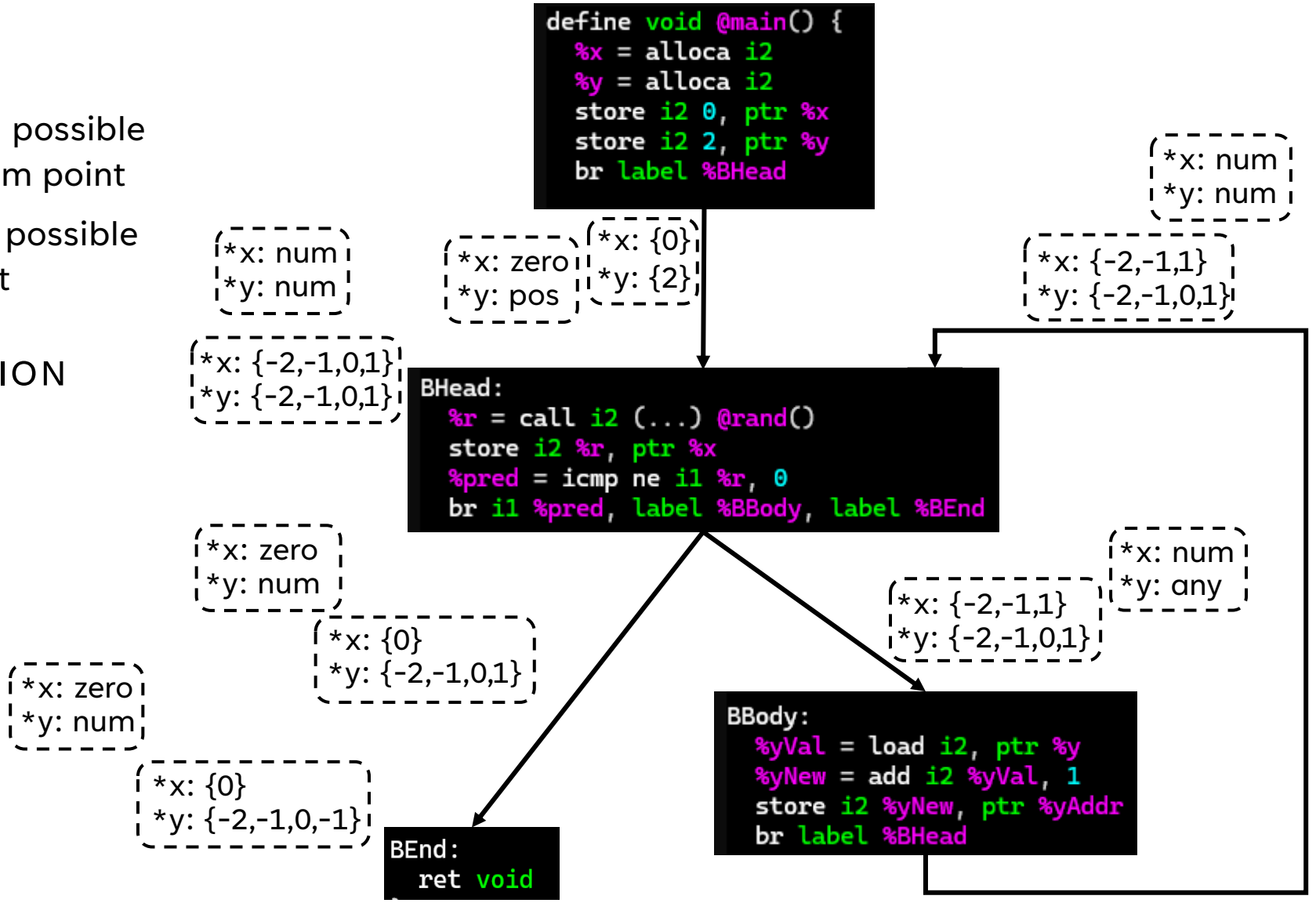
Mark each edge to indicate all possible values coming out of that point

ABSTRACT INTERPRETATION

Abstract domain: abstract summary values

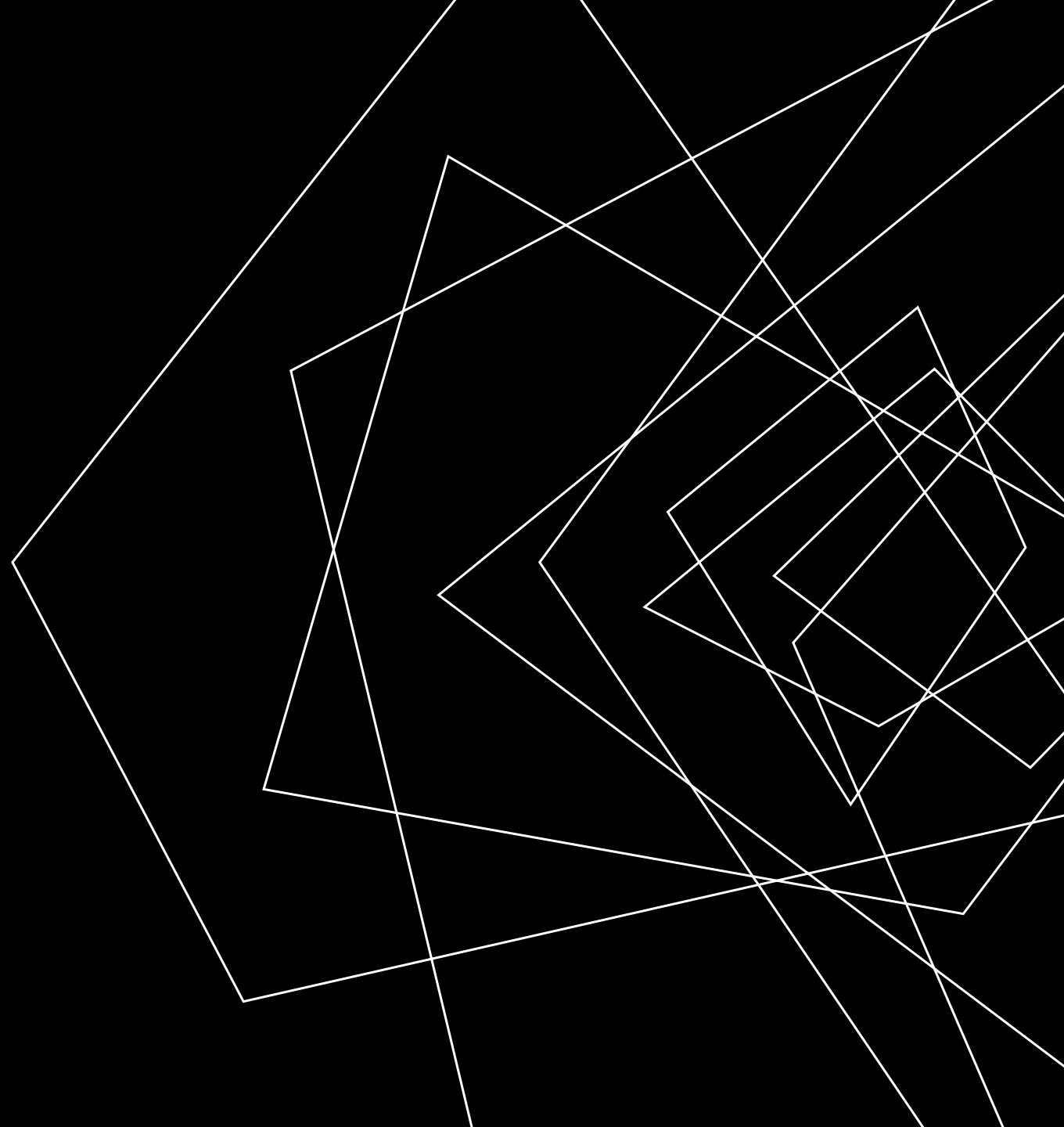
Edge merging: join over lattice elements

Transfer functions deal in abstract domain



LECTURE OUTLINE

- Formalizing dataflow
- Generalizing analysis



THE POTENTIAL OF DATAFLOW

DATAFLOW FRAMEWORKS

DATAFLOW ANALYSIS SHOWED SOME REAL POTENTIAL!

Relatively straightforward computation: update the dataflow fact stored at each basic block boundary until saturation

In our examples, output was always…

- Complete behavioral over-approximation -> no behaviors missed
- Relatively precise -> few “false positive” behaviors
- Saturated -> terminated

We’d like some stronger guarantees

ALGORITHM NEEDS

- Guaranteed termination
- Amenable to chaotic iteration
- Some degree of precision

DATAFLOW FORMALISM JOB SEARCH

DATAFLOW FRAMEWORKS

FORMAL(ISH) REQUIREMENTS

- A fact datatype (ideally of unbounded size)
- An ordering that indicates progress
- Unique solution
- A guarantee of a finite number of steps to hit the maximum value
- An update step that never loses progress

Property of the fact datatype

Property of the update function

FACT DATATYPE NEEDS

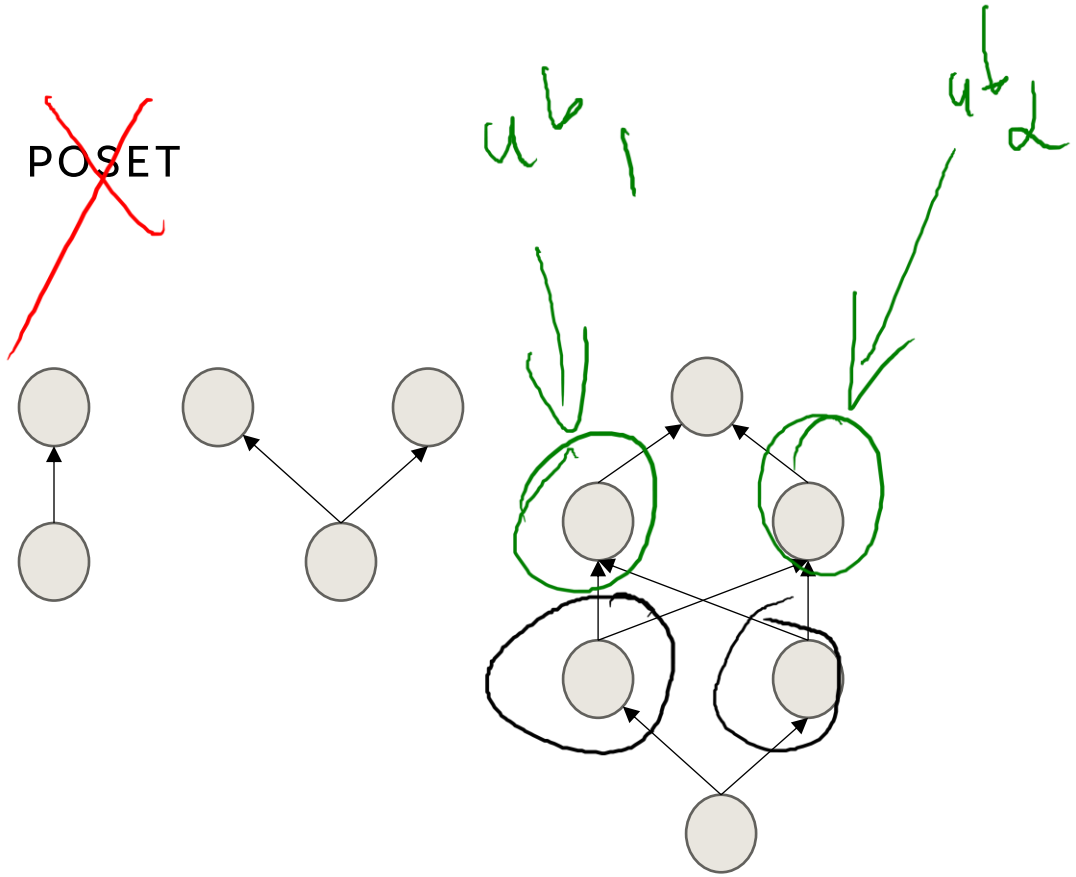
DATAFLOW FRAMEWORKS

FORMAL(ISH) REQUIREMENTS

- A fact datatype (ideally of unbounded size)
- An ordering that indicates progress
- Unique solution
- A guarantee of a finite number of steps to hit the maximum value
- An update step that never loses progress

TRY-OUTS

- poset
- Lattice
- Complete lattice



FACT DATATYPE NEEDS

DATAFLOW FRAMEWORKS

FORMAL(ISH) REQUIREMENTS

- A fact datatype (ideally of unbounded size)
- An ordering that indicates progress
- Unique solution
- A guarantee of a finite number of steps to hit the maximum value
- An update step that never loses progress

TRY-OUTS

- poset
- Lattice
- Complete lattice

FACT DATATYPE NEEDS

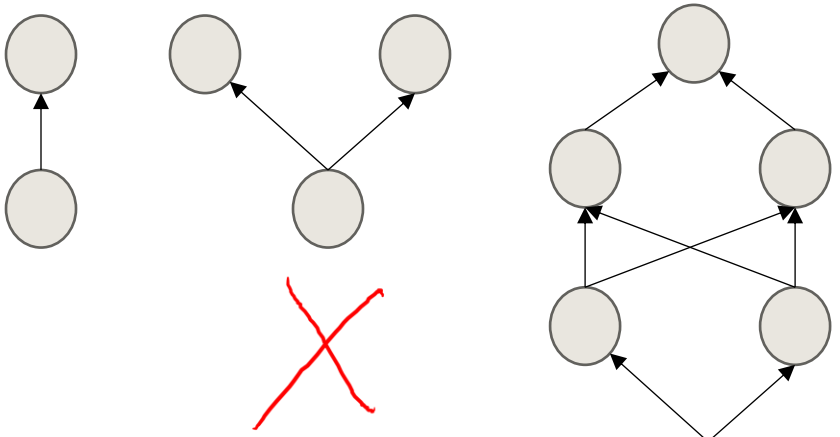
DATAFLOW FRAMEWORKS

FORMAL(ISH) REQUIREMENTS

- A fact datatype (ideally of unbounded size)
- An ordering that indicates progress
- Unique solution
- A guarantee of a finite number of steps to hit the maximum value
- An update step that never loses progress

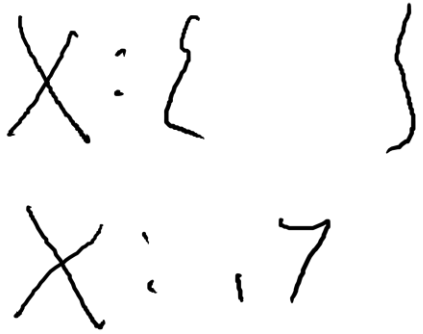
LATTICE

Poset with a least upper bound and a greatest lower bound



TRY-OUTS

- poset
- Lattice
- Complete lattice



FACT DATATYPE NEEDS

DATAFLOW FRAMEWORKS

FORMAL(ISH) REQUIREMENTS

- A fact datatype (ideally of unbounded size)
- An ordering that indicates progress
- Unique solution
- A guarantee of a finite number of steps to hit the maximum value
- An update step that never loses progress

TRY-OUTS

- poset
- Lattice
- Complete lattice

COMPLETE LATTICE

Lattice with a least upper bound and greatest lower bound for all subsets of the set

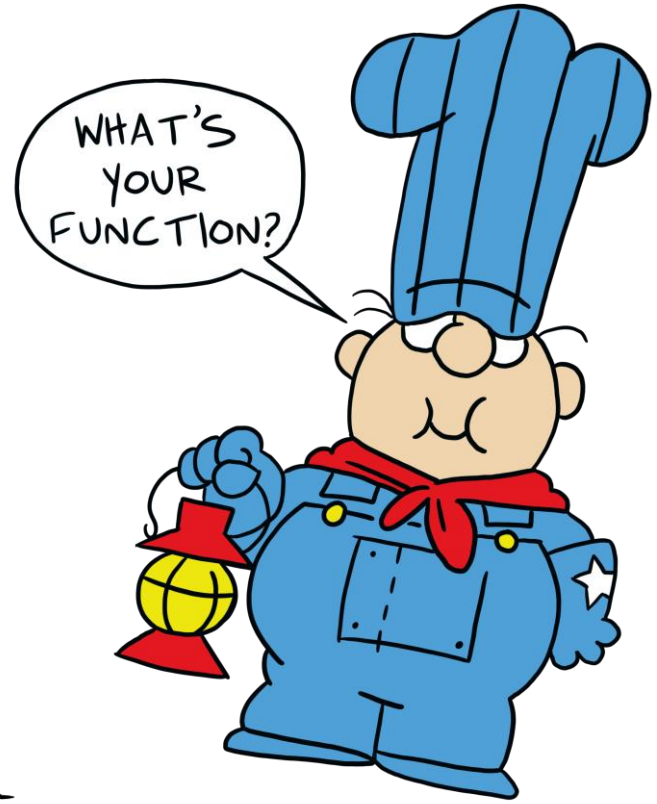
FUNCTION NEEDS

DATAFLOW FRAMEWORKS

SOME BASIC DEFINITIONS

A function f is a **monotonic function** if
$x \subseteq y$ implies $f(x) \subseteq f(y)$

An element z is a **fixpoint** of f iff $z = f(z)$



$1 < 2$

$f(+1)$

$2 < 3$

FUNCTION NEEDS

DATAFLOW FRAMEWORKS

SOME BASIC DEFINITIONS

A function f is a **monotonic function** if
x ⊆ y implies f(x) ⊆ f(y)

An element z is a **fixpoint** of f iff z = f(z)

Example

$f(x) = x \cup \{ b \}$

$f(\{b\}) = \{ b \}$ ← $\{b\}$ is a fixpoint of f

$f(\{a\}) = \{ a, b \}$ ← $\{a\}$ is **not** a fixpoint of f

$f(\{a, b\}) = \{ a, b \}$ ← $\{a, b\}$ is a fixpoint of f

$f(f(\{a\}))$ is a fixpoint of f

$f(f(any\ set))$ is a fixpoint of f

FUNCTION NEEDS

DATAFLOW FRAMEWORKS

SOME BASIC DEFINITIONS

A function f is a **monotonic function** if
x ⊆ y implies f(x) ⊆ f(y)

An element z is a **fixpoint** of f iff z = f(z)

Example

$glob(x) = x \cup \{ b \}$

$glob\ (\{b\}) \longrightarrow \{ b \}$ ← $\{b\}$ is a fixpoint of *glob*

$glob\ (\{a\}) \longrightarrow \{ a, b \}$ ← $\{a\}$ is **not** a fixpoint of *glob*

$glob\ (\{a, b\}) \longrightarrow \{ a, b \}$ ← $\{a, b\}$ is a fixpoint of *glob*

$glob(glob(\{a\}))$ is a fixpoint of *glob*

$glob(glob(any\ set))$ is a fixpoint of *glob*

WHY DOES THIS MATTER?

DATAFLOW FRAMEWORKS

Every finite lattice
is a complete lattice

PRACTICAL UPSHOT

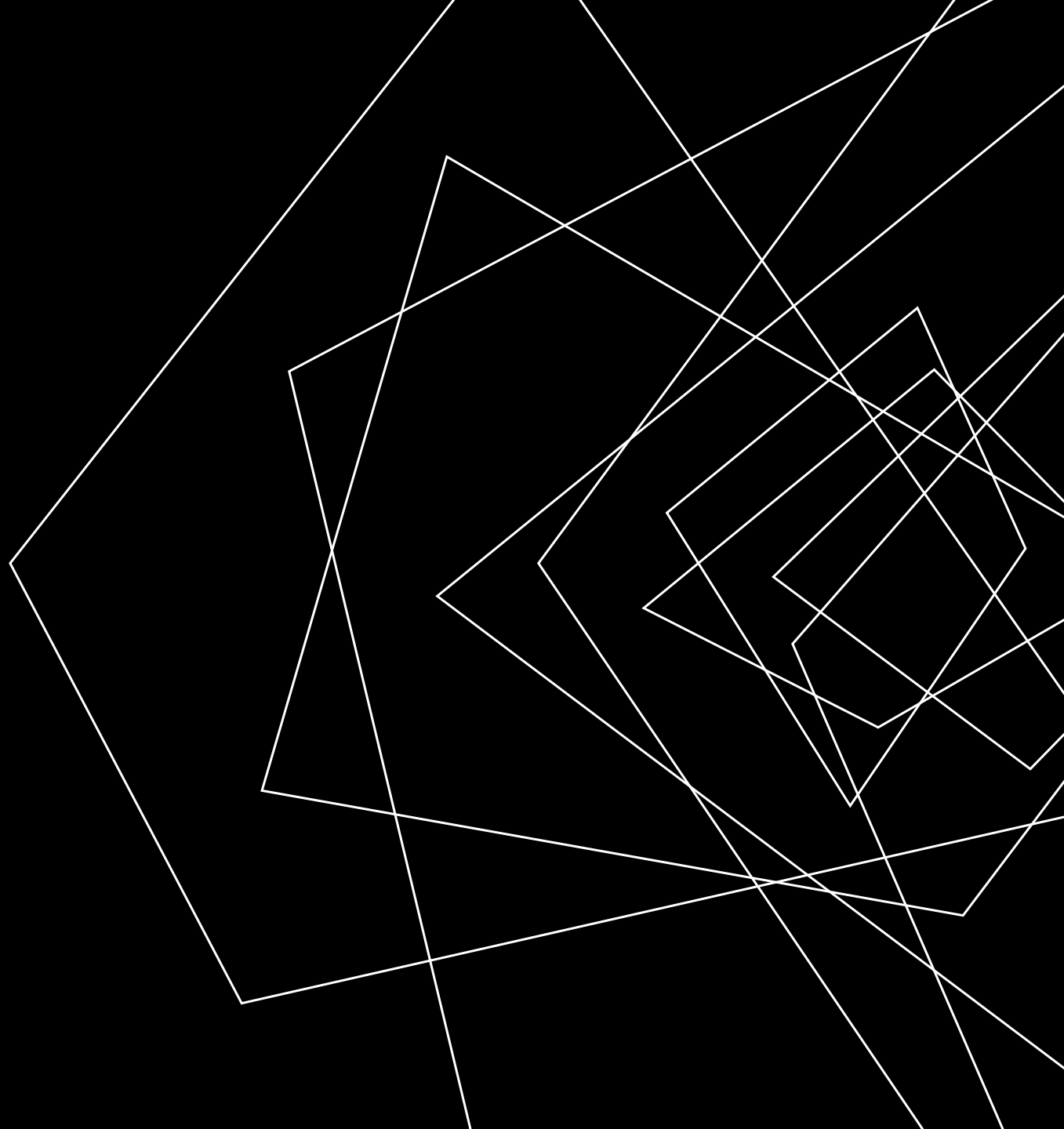
If L is a complete lattice and f is monotonic, then f has a greatest fixpoint and a least fixpoint

If L has no infinite ascending chains, the least fixpoint can be computed by iterative application of f

So the analysis will terminate

LECTURE OUTLINE

- Enhancing Dataflow analysis
- Lattices
- Abstract Interpretation



CHAOTIC ITERATION

STATIC ANALYSIS: CONTROL FLOW GRAPHS

A WORKLIST ALGORITHM

- Select the next worklist item in any order
- Necessarily assumes progress towards some goal

DEALING WITH "UNCOMPUTED" SETS

- Assume a reasonable "initial" value

Surprisingly, not a band with merch at Hot Topic

ANALYSIS PRECISION

ABSTRACT INTERPRETATION

PRECISION / EFFICIENCY TRADEOFF

With a complete lattice we can, in theory, eventually terminate

That's not a very strong guarantee!

The shallower the lattice, the faster the fixpoint

Choose to approximate the lattice

ANALYSIS PRECISION

ABSTRACT INTERPRETATION

PRECISION / EFFICIENCY TRADEOFF

With a complete lattice we can, in theory, eventually terminate

That's not a very strong guarantee!

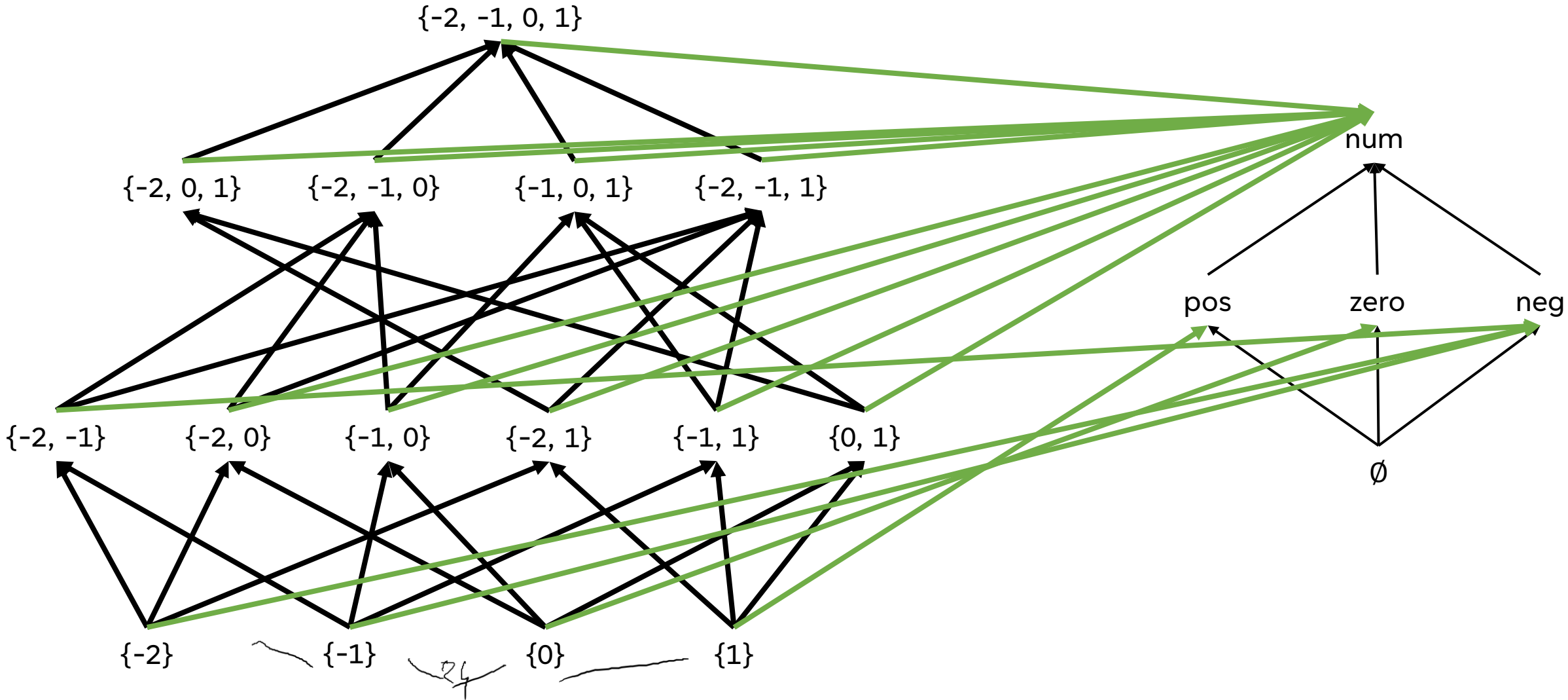
The shallower the lattice, the faster the fixpoint

Choose to approximate the lattice

LET'S CONSIDER A VERY APPROXIMATE LATTICE

ABSTRACT INTERPRETATION

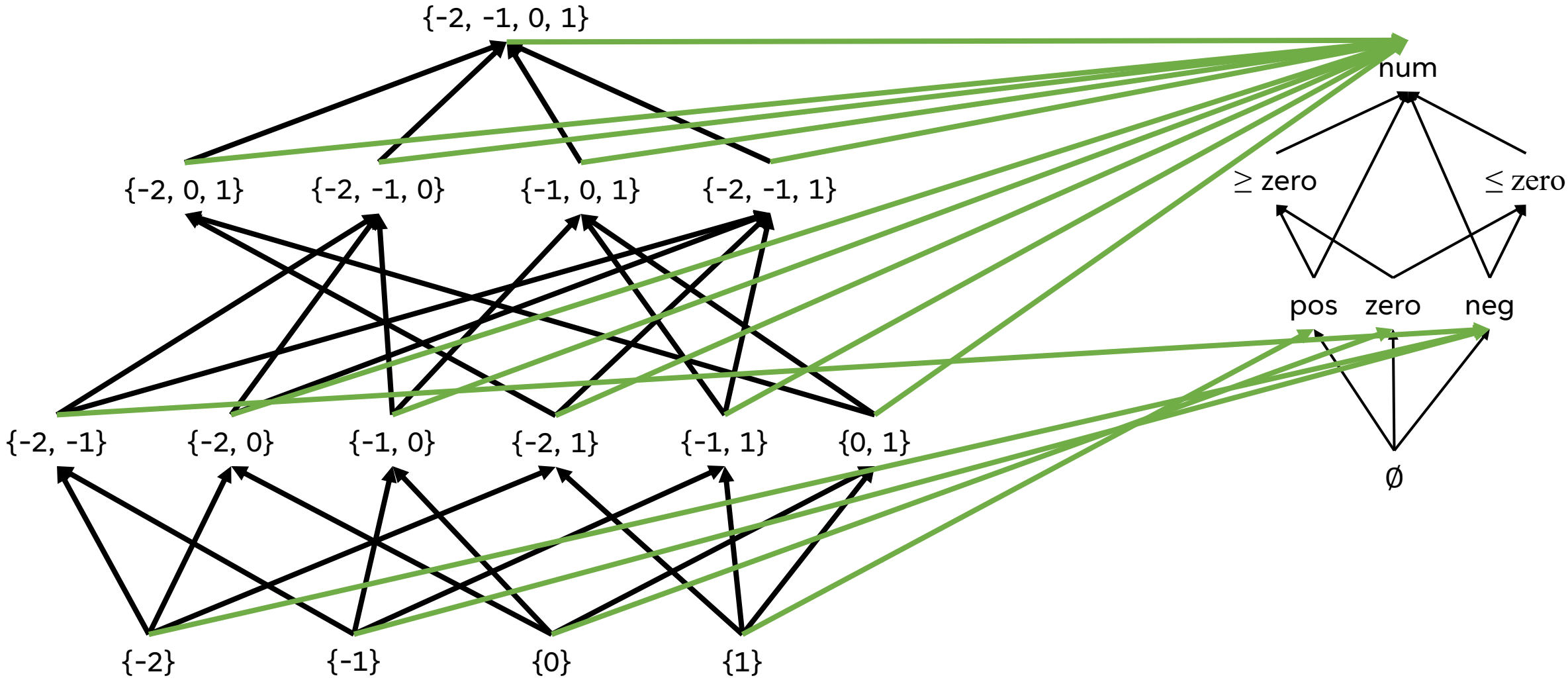
ABSTRACT DOMAIN OF SIGNS



LET'S CONSIDER A VERY APPROXIMATE LATTICE

ABSTRACT INTERPRETATION

ABSTRACT DOMAIN OF SIGNS



ABSTRACT DOMAINS IN PRACTICE

STATIC ANALYSIS

"Singleton Integer Sets"

- You know the number, or you don't

Intervals

- You know a concrete range

Property Existence

- A property does or does not hold

SECTION SUMMARY
STATIC ANALYSIS

STATIC ANALYSIS GIVES US AN IMPORTANT GUARANTEE

- Completeness of bug finding / Soundness of verification
- Thus far we've been using source code

Anything that isn't crystal clear to a static analysis tool probably isn't clear to your fellow programmers, either. The classic hacker disdain for "bondage and discipline languages" is short-sighted – the needs of large, long-lived, multi-programmer projects are just different than the quick work you do for yourself.

- John Carmack's Static Code Analysis post