

## Lattice-Theoretic Framework for Data-Flow Analysis

---

### Last time

- Generalizing data-flow analysis
- Introduced lattices

### Today

- Introduce lattice-theoretic frameworks for data-flow analysis

## Context

---

### Goals

- Provide a single formal model that describes all data-flow analyses
- Formalize the notions of “safe,” “conservative,” and “optimistic”
- Place bounds on time complexity of data-flow analysis
- Correctness proof for IDFA

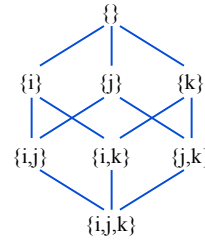
### Approach

- Define **domain** of program properties (flow values) computed by data-flow analysis, and organize the domain of elements as a **lattice**
- Define flow functions and a merge function over this domain using lattice operations
- Exploit lattice theory in achieving goals

## Data-Flow Analysis via Lattices

### Relationship

- Elements of the lattice ( $V$ ) represent flow values (in[] and out[] sets)
  - *e.g.*, Sets of live variables for liveness
- $\top$  represents “best-case” information (initial flow value)
  - *e.g.*, Empty set
- $\perp$  represents “worst-case” information
  - *e.g.*, Universal set
- $\sqcap$  (meet) merges flow values
  - *e.g.*, Set union
- If  $x \sqsubseteq y$ , then  $x$  is a conservative approximation of  $y$ 
  - *e.g.*, Superset



## Data-Flow Analysis Frameworks

### Data-flow analysis framework

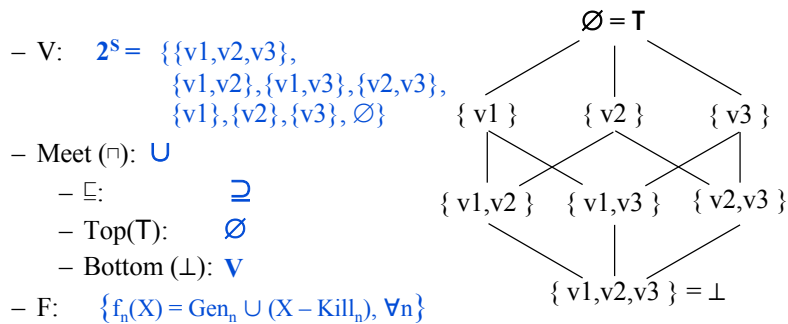
- A set of **flow values** ( $V$ )
- A binary **meet operator** ( $\sqcap$ )
- A set of **flow functions** ( $F$ ) (also known as **transfer functions**)

### Flow Functions

- $F = \{f: V \rightarrow V\}$ 
  - $f$  describes how each node in CFG affects the flow values
- Flow functions map program behavior onto lattices

## Visualizing DFA Frameworks as Lattices

**Example:** Liveness analysis with 3 variables  
 $S = \{v1, v2, v3\}$



Inferior solutions are lower on the lattice  
 More conservative solutions are lower on the lattice

## More Examples

### Reaching definitions

- $V: 2^S$  ( $S = \text{set of all defs}$ )
- $\sqcap: \cup$
- $\sqsubseteq: \supseteq$
- Top( $\top$ ):  $\emptyset$
- Bottom ( $\perp$ ):  $V$
- F: ...

### Reaching Constants

- $V: 2^{v \times c}$ , variables  $v$  and constants  $c$
- $\sqcap: \cap$
- $\sqsubseteq: \subseteq$
- Top( $\top$ ):  $V$
- Bottom ( $\perp$ ):  $\emptyset$
- F: ...

## Tuples of Lattices

---

### Problem

- Simple analyses may require very complex lattices  
(e.g., Reaching constants)

### Solution

- Use a tuple of lattices, one per variable

$$\mathbf{L} = (\mathbf{V}, \sqcap) \equiv (\mathbf{L}_T = (\mathbf{V}_T, \sqcap_T))^N$$

- $\mathbf{V} = (\mathbf{V}_T)^N$
- Meet ( $\sqcap$ ): point-wise application of  $\sqcap_T$
- $(\dots, v_i, \dots) \sqsubseteq (\dots, u_i, \dots) \equiv v_i \sqsubseteq u_i, \forall i$
- Top ( $\top$ ): tuple of tops ( $\top_T$ )
- Bottom ( $\perp$ ): tuple of bottoms ( $\perp_T$ )
- Height ( $L$ ) =  $N * \text{height}(\mathbf{L}_T)$

## Tuples of Lattices Example

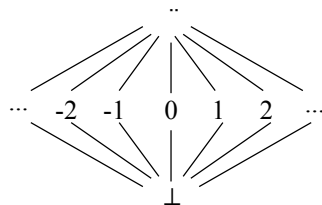
---

### Reaching constants (previously)

- $P = v \times c$ , for variables  $v$  & constants  $c$
- $\mathbf{V}: 2^P$

### Alternatively

- $\mathbf{V} = c \cup \{\top, \perp\}$



**The whole problem is a tuple of lattices, one for each variable**

## Examples of Lattice Domains

### Two-point lattice ( $\top$ and $\perp$ )

- Examples?
- Implementation?

### Set of incomparable values (and $\top$ and $\perp$ )

- Examples?

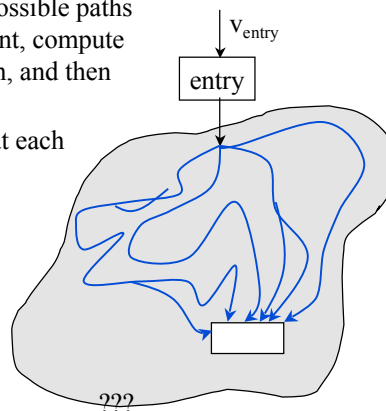
### Powerset lattice ( $2^S$ )

- $\top = \emptyset$  and  $\perp = S$ , or vice versa
- Isomorphic to tuple of two-point lattices

## Solving Data-Flow Analyses

### Goal

- For a forward problem, consider all possible paths from the entry to a given program point, compute the flow values at the end of each path, and then meet these values together
- Meet-over-all-paths (MOP) solution at each program point
- $\sqcap_{\text{all paths } n_1, n_2, \dots, n_i} (f_{n_i}(\dots f_{n_2}(f_{n_1}(v_{\text{entry}}))))$



## Solving Data-Flow Analyses (cont)

### Problems

- Loops result in an infinite number of paths
- Statements following merge must be analyzed for all preceding paths
  - Exponential blow-up

### Solution

- Compute meets early (at merge points) rather than at the end
- Maximum fixed-point (MFP)

### Questions

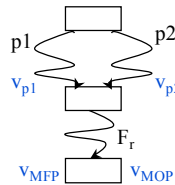
- Is this legal?
- Is this efficient?
- Is this accurate?

## Legality

“Is  $v_{MFP}$  legal?”  $\equiv$  “Is  $v_{MFP} \sqsubseteq v_{MOP}$ ?”

### Look at Merges

$$\begin{aligned}
 v_{MOP} &= F_r(v_{p1}) \sqcap F_r(v_{p2}) \\
 v_{MFP} &= F_r(v_{p1} \sqcap v_{p2}) \\
 v_{MFP} \sqsubseteq v_{MOP} &\equiv F_r(v_{p1} \sqcap v_{p2}) \sqsubseteq F_r(v_{p1}) \sqcap F_r(v_{p2})
 \end{aligned}$$



### Observation

$$\forall x, y \in V \quad f(x \sqcap y) \sqsubseteq f(x) \sqcap f(y) \quad \Leftrightarrow \quad x \sqsubseteq y \Rightarrow f(x) \sqsubseteq f(y)$$

$\therefore v_{MFP}$  legal when  $F_r$  (really, the flow functions) are monotonic

## Monotonicity

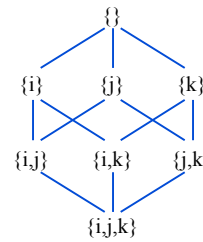
**Monotonicity:**  $(\forall x,y \in V)[x \sqsubseteq y \Rightarrow f(x) \sqsubseteq f(y)]$

- If the flow function  $f$  is applied to two members of  $V$ , the result of applying  $f$  to the “lesser” of the two members will be under the result of applying  $f$  to the “greater” of the two
- Giving a flow function more conservative inputs leads to more conservative outputs (never more optimistic outputs)

**Why else is monotonicity important?**

**For monotonic  $F$  over domain  $V$**

- The maximum number of times  $F$  can be applied to self w/o reaching a fixed point is  $\text{height}(V) - 1$
- IDFA is guaranteed to terminate if the flow functions are monotonic and the lattice has finite height



## Efficiency

**Parameters**

- $n$ : Number of nodes in the CFG
- $k$ : Height of lattice
- $t$ : Time to execute one flow function

**Complexity**

- $O(nkt)$

**Example**

- Reaching definitions?

## Accuracy

### Distributivity

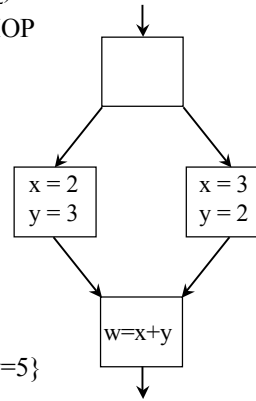
- $f(u \sqcap v) = f(u) \sqcap f(v)$
- $v_{MFP} \sqsubseteq v_{MOP} \equiv F_r(v_{p1} \sqcap v_{p2}) \sqsubseteq F_r(v_{p1}) \sqcap F_r(v_{p2})$
- If the flow functions are distributive, MFP = MOP

### Examples

- Reaching definitions?
- Reaching constants?

$$f(u \sqcap v) = f(\{x=2, y=3\} \sqcap \{x=3, y=2\}) \\ = f(\emptyset) = \emptyset$$

$$f(u) \sqcap f(v) = f(\{x=2, y=3\}) \sqcap f(\{x=3, y=2\}) \\ = [\{x=2, y=3, w=5\} \sqcap \{x=2, y=2, w=5\}] = \{w=5\} \\ \Rightarrow \text{MFP} \neq \text{MOP}$$



## Bitwidth Analysis Paper

**Why did we read this paper?**

**Can all dataflow analyses be defined in terms of Gen and Kill?**

**Do all dataflow analysis problems operate on sets?**

## Concepts

---

### Lattices

- Conservative approximation
- Optimistic (initial guess)
- Data-flow analysis frameworks
- Tuples of lattices

### Data-flow analysis

- Fixed point
- Meet-over-all-paths (MOP)
- Maximum fixed point (MFP)
- Legal/safe (monotonic)
- Efficient
- Accurate (distributive)

## Next Time

---

### Reading

- Ch 8.11 in Muchnick
- all Muchnick readings are for main ideas and examples
- start reading the SSA paper, it is LONG!!

### Lecture

- Program representations (static single assignment)