

# Abstract Interpretation Part II

Mooly Sagiv

Textbook: Chapter 4  
CC79, CC92

## Tentative Schedule

24/5	Operational Semantics
31/5 7/6	Abstract Interpretation
14/6	No class
21/6	Shape Analysis
22/6 9-12 309	
27/6	Predicate Abstraction
3/8 9-12 309	Advanced Topics

Targil 2

Course  
Project

## Outline

- ◆ The Soundness Theorem
- ◆ Intuition about abstract interpretation
- ◆ Methodologies for creating abstractions

---

---

---

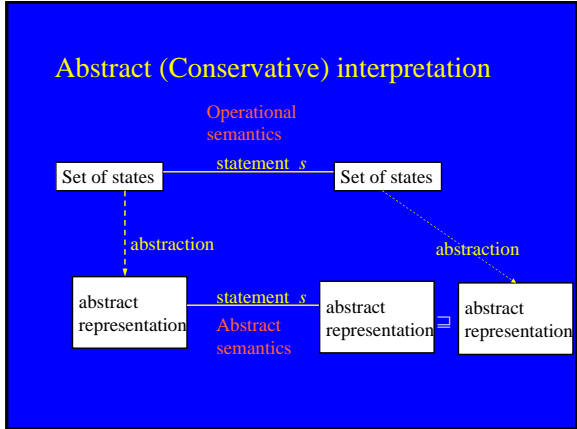
---

---

---

---

---




---

---

---

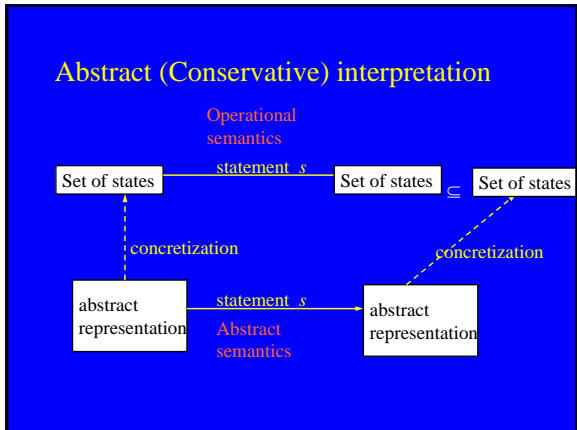
---

---

---

---

---




---

---

---

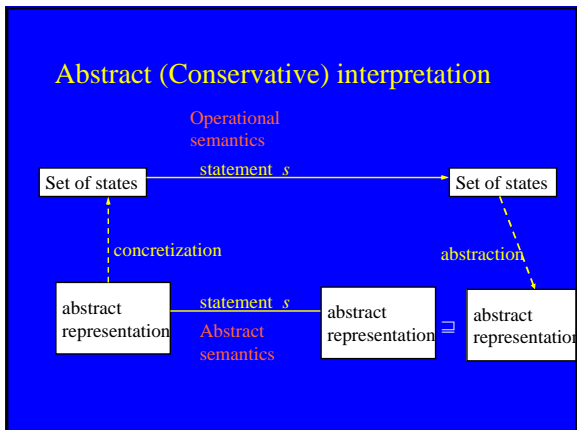
---

---

---

---

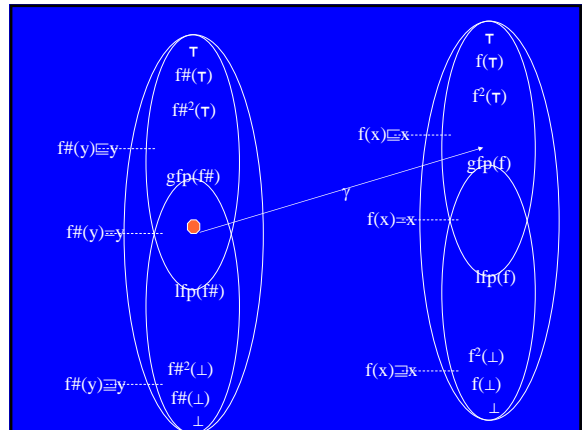
---



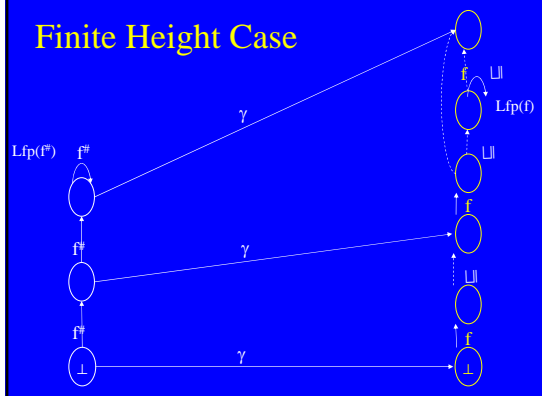


## Soundness Theorem

1. Let  $(\alpha, \gamma)$  form **Galois connection** from  $C$  to  $A$
  2.  $f: C \rightarrow C$  be a monotone function
  3.  $f^\#: A \rightarrow A$  be a monotone function
  4.  $\forall a \in A: f(\gamma(a)) \sqsubseteq \gamma(f^\#(a))$
  5.  $\forall c \in C: \alpha(f(c)) \sqsubseteq f^\#(\alpha(c))$
  6.  $\forall a \in A: \alpha(f(\gamma(a))) \sqsubseteq f^\#(a)$
- }  $\vee$
- $\text{lfp}(f) \sqsubseteq \gamma(\text{lfp}(f^\#))$   
 $\alpha(\text{lfp}(f)) \sqsubseteq \text{lfp}(f^\#)$



## Finite Height Case



---

---

---

---

---

---

---

---

## Local Concrete Semantics

- ◆ For every atomic statement S
  - $\llbracket S \rrbracket : [\text{Var.} \rightarrow Z] \rightarrow [\text{Var.} \rightarrow Z]$
  - $\llbracket x := a \rrbracket s = s[x \mapsto A\llbracket a \rrbracket s]$
  - $\llbracket \text{skip} \rrbracket s = s$
- ◆ For Boolean conditions ...

---

---

---

---

---

---

---

---

## Local Abstract Semantics(CP)

- ◆ For every atomic statement S
  - $\llbracket S \rrbracket^\# : \text{Var.} \rightarrow L \rightarrow \text{Var.} \rightarrow L$
  - $\llbracket x := a \rrbracket^\#(e) = e[x \mapsto \llbracket a \rrbracket^\#(e)]$
  - $\llbracket \text{skip} \rrbracket^\#(e) = e$
- ◆ For Booleans ...

---

---

---

---

---

---

---

---

## Lemma 1

Consider a lattice L.

f: L → L is monotone iff

$$\text{for all } X \subseteq L: \sqcup\{f(z) \mid z \in X\} \sqsubseteq f(\sqcup\{z \mid z \in X\})$$

---

---

---

---

---

---

---

---

---

---

## Assignments in constant propagation

- ◆ Monotone
  - $df_1 \sqsubseteq df_2 \rightarrow \llbracket x := e \rrbracket^\#(df_1) \sqsubseteq \llbracket x := e \rrbracket^\#(df_2)$
- ◆ Local Soundness
  - $\alpha(\{\llbracket x := e \rrbracket \sigma \mid \sigma \in CS\}) \sqsubseteq \llbracket x := e \rrbracket^\#(\alpha(CS))$
- ◆ Best Transformer
- ◆ Homomorphic

---

---

---

---

---

---

---

---

---

---

## Proof of Soundness (Summary)

- ◆ Define an “appropriate” operational semantics
- ◆ Define “collecting” operational semantics
- ◆ Establish a Galois connection between collecting states and abstract states
- ◆ (Local correctness) Show that the abstract interpretation of every atomic statement is **sound** w.r.t. the collecting semantics
- ◆ (Global correctness) Conclude that the result of the iterative analysis is sound w.r.t. the collecting semantics
- ◆ Can be applied between different abstractions

---

---

---

---

---

---

---

---

---

---

## Induced Analysis (Relatively Optimal)

- ◆ It is sometimes possible to show that a given analysis is not only sound but optimal w.r.t. the chosen abstraction
  - but not necessarily optimal!
- ◆ Define  $\llbracket S \rrbracket^\#(df) = \alpha(\{\llbracket S \rrbracket \sigma \mid \sigma \in \gamma(df)\})$
- ◆ But this  $\llbracket S \rrbracket^\#$  may not be computable
- ◆ Derive (at compiler-generation time) an alternative form for  $\llbracket S \rrbracket^\#$
- ◆ A useful measure to decide if the abstraction must lead to overly imprecise results

---

---

---

---

---

---

---

---

## Properties of Abstractions

- ◆ Eagerly forget parts of the state
- ◆ Reduce state space
- ◆ Abstract traces do not necessarily correspond to concrete trace
  - even when best transformer is used
- ◆ Executes the program on traces with “fabricated” states
- ◆ When the abstraction succeeds prove stronger properties

---

---

---

---

---

---

---

---

## Notions of precision

- ◆  $CS = \gamma(df)$
- ◆  $\alpha(CS) = df$
- ◆ Meet(Join) over all paths
- ◆ Using best transformers
- ◆ Good enough

---

---

---

---

---

---

---

---

## Summary

- ◆ Abstract interpretation relates runtime semantics and static information
- ◆ The concrete semantics serves as a tool in designing abstractions
- ◆ Understanding concretization is a must
- ◆ Understand what is preserved/lost

---

---

---

---

---

---

---

---

## Combining Data Flow Analyzes

- ◆ Develop new algorithms from old
- ◆ If I know how to conservatively represent
  - Pointers
  - Integers
- ◆ Do I know how to handle C programs with integers **and** pointers?

---

---

---

---

---

---

---

---

## Combining Data Flow Analyzes

- ◆ Develop new algorithms from old
- ◆ If I know how to conservatively represent
  - Pointers
  - Integers
- ◆ Do I know how to handle C programs with integers **and** pointers?
- ◆ Improve the precision of an analysis
- ◆ Obtain a more efficient analysis

---

---

---

---

---

---

---

---

## Combining Data Flow Analyzers

- ◆ Lattice constructors
  - $L_1 \times L_2$
  - $S \rightarrow L_1$
  - ...
- ◆ Galois connection constructors
- ◆ Constructing the abstract effect of elementary statements
- ◆ Model the “relevant” parts of the program
- ◆ Abstract “irrelevant” parts of the program

## Galois Connections

- ◆ For
  - A complete lattice  
 $(L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \perp_1, \top_1)$
  - A complete lattice  
 $(L_2, \sqsubseteq_2) = (L_2, \sqsubseteq, \sqcup_2, \sqcap_2, \perp_2, \top_2)$
  - $\alpha: L_1 \rightarrow L_2$
  - $\gamma: L_2 \rightarrow L_1$
- ◆ We say that  $(L_1, \alpha, \gamma, L_2)$  is a Galois connection
  - $\alpha$  and  $\gamma$  are monotone
  - For all  $c \in L_1$ :  $\gamma(\alpha(c)) \sqsupseteq c$
  - For all  $a \in L_2$ :  $\alpha(\gamma(a)) \sqsubseteq a$

## Cartesian Products

- ◆ A complete lattice  
 $(L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \perp_1, \top_1)$
- ◆ A complete lattice  
 $(L_2, \sqsubseteq_2) = (L_2, \sqsubseteq, \sqcup_2, \sqcap_2, \perp_2, \top_2)$
- ◆ Define a Poset  $L = (L_1 \times L_2, \sqsubseteq)$  where
  - $(x_1, x_2) \sqsubseteq (y_1, y_2)$  if
    - »  $x_1 \sqsubseteq y_1$  and
    - »  $x_2 \sqsubseteq y_2$
- ◆  $L$  is a complete lattice
- ◆ But what does an element in  $L$  represent?

## Cartesian Products (cont)

- ◆ A complete lattice  
 $(L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \perp_1, \top_1)$
- ◆ A complete lattice  
 $(L_2, \sqsubseteq_2) = (L_2, \sqsubseteq, \sqcup_2, \sqcap_2, \perp_2, \top_2)$
- ◆ Complete lattice  $L = (L_1 \times L_2, \sqsubseteq)$
- ◆ A concrete lattice  $C$  (usually a powerset)
- ◆ A Galois connection  
 $(C, \alpha_1, \gamma_1, L_1)$
- ◆ A Galois connection  
 $(C, \alpha_2, \gamma_2, L_2)$
- ◆ Define  $\alpha: C \rightarrow L_1 \times L_2$  and  $\gamma: L_1 \times L_2 \rightarrow C$ ?
- ◆ Example: Parity  $\times$  Sign

---

---

---

---

---

---

---

---

---

---

## Cartesian Products (cont)

- ◆ A Galois connection  $(C, \alpha_1, \gamma_1, L_1)$
- ◆ A Galois connection  $(C, \alpha_2, \gamma_2, L_2)$
- ◆ A Galois connection  $(C, \alpha, \gamma, L_1 \times L_2)$ 
  - $\alpha(c) = \langle \alpha_1(c), \alpha_2(c) \rangle$
  - $\gamma(\langle a_1, a_2 \rangle) = \gamma_1(a_1) \sqcap \gamma_2(a_2)$
- ◆ Define
  - $L_1 \llbracket \text{st} \rrbracket^\# : L_1 \rightarrow L_1$
  - $L_2 \llbracket \text{st} \rrbracket^\# : L_2 \rightarrow L_2$
- ◆ How to define  $L_1 \times L_2 \llbracket \text{st} \rrbracket^\# : L_1 \times L_2 \rightarrow L_1 \times L_2$ 
  - Preserve soundness
  - Preserve relative optimality (induced)
  - Reasonable
- ◆ Example: Parity  $\times$  Sign

---

---

---

---

---

---

---

---

---

---

## Component-wise combinations

- ◆ Combine several analyses into a single analysis
- ✓ Cartesian products (Direct product)
- ◆ Independent attribute method
- ◆ Relational attribute method
- ◆ Total function space
- ◆ Monotone function space
- ◆ Direct tensor product

---

---

---

---

---

---

---

---

---

---

## Independent Attribute Method

- ◆ A Galois connection  $(C_1, \alpha_1, \gamma_1, L_1)$
- ◆ A Galois connection  $(C_2, \alpha_2, \gamma_2, L_2)$
- ◆ A Galois connection  $(C_1 \times C_2, \alpha, \gamma, L_1 \times L_2)$ 
  - $\alpha(\langle c_1, c_2 \rangle) = \langle \alpha_1(c_1), \alpha_2(c_2) \rangle$
  - $\gamma(\langle a_1, a_2 \rangle) = \langle \gamma_1(a_1), \gamma_2(a_2) \rangle$
- ◆ Define
  - $L_1 \llbracket \text{st} \rrbracket^\# : L_1 \rightarrow L_1$
  - $L_2 \llbracket \text{st} \rrbracket^\# : L_2 \rightarrow L_2$
- ◆ How to define  $L_1 \times L_2 \llbracket \text{st} \rrbracket^\# : L_1 \times L_2 \rightarrow L_1 \times L_2$ 
  - Preserve soundness
  - Preserve relative optimality (induced)

---

---

---

---

---

---

---

---

---

---

## Relational Attribute Method

- ◆ A Galois connection  
( $P(C_1), \alpha_1, \gamma_1, P(L_1)$ ) where  $\beta_1: C_1 \rightarrow L_1$ 
  - $\alpha_1(X) = \cup\{\beta_1(c) \mid c \in X\}$
- ◆ A Galois connection  
( $P(C_2), \alpha_2, \gamma_2, P(L_2)$ ) where  $\beta_2: C_2 \rightarrow L_2$ 
  - $\alpha_2(X) = \cup\{\beta_2(c) \mid c \in X\}$
- ◆ A Galois connection ( $P(C_1 \times C_2), \alpha, \gamma, P(L_1 \times L_2)$ )
  - $\alpha(\langle X_1, X_2 \rangle) = \{ \langle \beta_1(c_1), \beta_2(c_2) \rangle \mid c_1 \in X_1, c_2 \in X_2 \}$
  - $\gamma(\langle Y_1, Y_2 \rangle) = \{ \langle c_1, c_2 \rangle \mid \beta_1(c_1) \in Y_1, \beta_2(c_2) \in Y_2 \}$
- ◆ But how about transformers?

---

---

---

---

---

---

---

---

---

---

## Semantic Reduction

- ◆ Consider a Galois connection  
( $C, \alpha, \gamma, A$ )
- ◆ An operation  $op: A \rightarrow A$  is a **semantic reduction** if
  - For all  $a \in A$ :  $op(a) \sqsubseteq a$  and  $\gamma(op(a)) = \gamma(a)$

---

---

---

---

---

---

---

---

---

---

## Conclusions(1)

- ◆ Good static analysis =
  - Precise enough (for the client)
  - Efficient enough
- ◆ Good static analysis
  - Good domain
    - » Abstract non-important details
    - » Represent relevant concrete information
    - » Precise and efficient abstract meaning of abstract interpreters
    - » Efficient join implementation
    - » Small height or widening

## Conclusions(2)

- ◆ The Theory of Static Analysis is well founded
  - Abstraction
  - Soundness
  - Chaotic iterations
  - Elimination methods
  - Modular methods
- ◆ Weak Parts
  - Transformations
  - Predictable approximations
  - User defined abstractions
  - System