CS 357 D

Lecture 7

Abstract Interpretation
Introduction

http://cs357d.stanford.edu/

April 24, 2007

Abstraction of Physical Systems

Abstraction enables us to do system analysis in one domain and carry over the results into a different domain

Common abstraction:

Physical system

analysis results

modeling

Mathematical model

we analyze a mathematical model of the system and assume that the physical system behaves similarly.

the justification that analysis results can indeed be carried over is necessarily informal, since we cannot establish a formal correspondence between the physical system and the mathematical model; we rely on domain experts and experimentation

Mathematical Abstraction

(concrete)
Mathematical model

analysis results

modeling

(abstract)
Mathematical model

we analyze a, usually simpler, mathematical model of the system and conclude that the more complex model has the same properties.

In this case, property preservation can be formally justified since we can define a formal relationship between the two models.

Here we will be concerned only with this type of abstraction, and in particular with **abstract interpretation**, the theory that relates the semantics of systems in different domains.

Abstract Interpretation (Cousot&Cousot 1977)

The theory of abstract interpretation was introduced by Cousot and Cousot (POPL'77); it has been and still is being used in many different settings, ranging from compiler optimization to language semantics analysis, formal verification, and theorem proving.

From the POPL'77 paper:

"A program denotes computations in some universe of objects. Abstract interpretation of programs consists in using that denotation to describe computations in another universe of abstract objects, so that the results of abstract execution give some information about the actual computations."

Abstract Interpretation -- more quotes

Cousot & Cousot, Journal of Logic and Computation, 1992:

"Abstract interpretation is a method for designing approximate semantics of programs which can be used to gather information about programs in order to provide sound answers to questions about their runtime behaviors. These semantics can then be used to design manual proof methods or to specify automatic program analyses."

Abstract interpretation -- more quotes

Cousot & Cousot, 1992:

"Theoretical point of view: The purpose of abstract interpretation is to design hierarchies of interrelated semantics at various levels of detail."

"Practical point of view: The purpose of abstract interpretation is to design automatic program analysis tools for determining statically dynamic properties of programs."

Abstract interpretation -- basics

Given:

- a concrete system with concrete (standard) semantics
- some notion of the properties we are interested in

We have to choose / construct :

1. Abstract domain
2. Correspondence between abstract and concrete objects
3. Abstract semantics

Abstract interpretation -- basics

Given:

- a concrete system with concrete (standard) semantics
- some notion of the properties we are interested in

Abstract interpretation -- a simple example

Concrete system : multiplication of integers

Question : are the results of these multiplications less than, equal to, or greater than zero?

Concrete domain: sets of integers $\Sigma = 2^Z$

Extend the semantics of multiplication to multiplication of sets:

$$S_1 \times S_2 = \{ n \mid \exists n_1 \in S_1 , n_2 \in S_2 . \; n_1 \times n_2 = n \}$$

Example: $\{ 1 , 2 \} \times \{ 3 , 4 \} = \{ 3 , 4 , 6 , 8 \}$

Abstract interpretation -- basics

We have to choose / construct :

1. Abstract domain
2. Correspondence between abstract and concrete objects
3. Abstract semantics

Abstract interpretation -- a simple example

Question : are the results of these multiplications less than, equal to, or greater than zero?

1. Abstract domain: $\Sigma_A = \{ neg , zero , pos \}$

Abstract interpretation -- a simple example

Question : are the results of these multiplications less than, equal to, or greater than zero?

1. Abstract domain:

$\Sigma_A = \{ neg , zero , pos \}$

$\Sigma_A = \{ -1 , 0 , 1 \}$

Σ_A = { ⠝⠑⠛ , ⠵⠑⠗⠕ , ⠏⠕⠎ }

Σ_A = { ♭ , ♮ , ♯ }

Σ_A = { ☼ , ☁ , ☽ }

Σ_A = { ♔ , ♖ , ♘ }

Abstract interpretation -- a simple example

Question : are the results of these multiplications less than, equal to, or greater than zero?

1. Abstract domain: $\Sigma_A = \{ neg , zero , pos \}$

Choose an abstract domain

Question : are the results of these multiplications less than, equal to, or greater than zero?

1. Abstract domain: $\Sigma_A = \{ neg , zero , pos \}$

2. Correspondence between abstract and concrete objects

   expressed by a concretization function

$$\boxed{\gamma : \Sigma_A \rightarrow \Sigma}$$

Concretization function

{ neg , zero , pos }

$$\boxed{\gamma : \Sigma_A \rightarrow \Sigma}$$

sets of integers

maps **abstract objects** to **concrete objects**

gives meaning to the abstract objects

$\gamma (neg) = \{ n \in Z \mid n < 0 \}$

$\gamma (zero) = \{ 0 \}$

$\gamma (pos) = \{ n \in Z \mid n > 0 \}$

Abstraction function

sets of integers

$$\boxed{\alpha : \Sigma \rightarrow \Sigma_A}$$

{ neg , zero , pos }

maps **concrete objects** into **abstract objects**

$\alpha(\{ 0 \}) = zero$

$\alpha(S) = neg$ if $\forall n \in S . n < 0$

$\alpha(S) = pos$ if $\forall n \in S . n > 0$

$\alpha(S) = ?$ otherwise ??

we need another abstract object to map sets like { 3 , -4 } into

Abstraction function

sets of integers

$\alpha : \Sigma \rightarrow \Sigma_A$

{ neg , zero , pos }

maps **concrete objects** into **abstract objects**

$\alpha(\{ 0 \}) = \text{zero}$

$\alpha(S) = \text{neg}$ if $\forall n \in S \,.\, n < 0$

$\alpha(S) = \text{pos}$ if $\forall n \in S \,.\, n > 0$

$\boxed{\alpha(S) = \top \quad \text{otherwise}}$

introduce new abstract object
$\top$ (top)
with meaning
$\gamma(\top) = \mathbb{Z}$

Abstraction function

sets of integers

$\alpha : \Sigma \rightarrow \Sigma_A$

{ $\bot$, neg , zero , pos, $\top$ }

maps **concrete objects** into **abstract objects**

$\alpha(\{ 0 \}) = \text{zero}$

$\alpha(S) = \text{neg}$ if $\forall n \in S \,.\, n < 0$

$\alpha(S) = \text{pos}$ if $\forall n \in S \,.\, n > 0$

$\alpha(S) = \bot$ if $S = \emptyset$

$\alpha(S) = \top$ otherwise

for symmetry also add new
abstract object $\bot$ (bottom)
with meaning
$\gamma(\bot) = \emptyset$

Abstraction function

sets of integers

$\alpha : \Sigma \rightarrow \Sigma_A$

{ $\bot$, neg , zero , pos, $\top$ }

maps **concrete objects** into **abstract objects**

$\alpha(S) = \bot$ if $S = \emptyset$

$\alpha(\{ 0 \}) = \text{zero}$

$\alpha(S) = \text{neg}$ if $\forall n \in S \,.\, n < 0$

$\alpha(S) = \text{pos}$ if $\forall n \in S \,.\, n > 0$

$\alpha(S) = \top$ otherwise

Concretization function

{ $\bot$, neg , zero , pos, $\top$ }

$\gamma : \Sigma_A \rightarrow \Sigma$

sets of integers

maps **abstract objects** to **concrete objects**

gives meaning to the abstract objects

$\gamma(\bot) = \emptyset$

$\gamma(\text{neg}) = \{ n \in \mathbb{Z} \mid n < 0 \}$

$\gamma(\text{zero}) = \{ 0 \}$

$\gamma(\text{pos}) = \{ n \in \mathbb{Z} \mid n > 0 \}$

$\gamma(\top) = \mathbb{Z}$

Abstraction and Concretization function

$\alpha : \Sigma \to \Sigma_A$

abstraction

$\gamma : \Sigma_A \to \Sigma$

concretization

$\alpha(\ S\ ) = \bot$ if $S = \varnothing$

$\alpha(\ S\ ) = neg$ if $\forall n \in S \,.\, n < 0$

$\alpha(\{ 0 \}) = zero$

$\alpha(\ S\ ) = pos$ if $\forall n \in S \,.\, n > 0$

$\alpha(\ S\ ) = T$ otherwise

$\gamma(\ \bot\ ) = \varnothing$

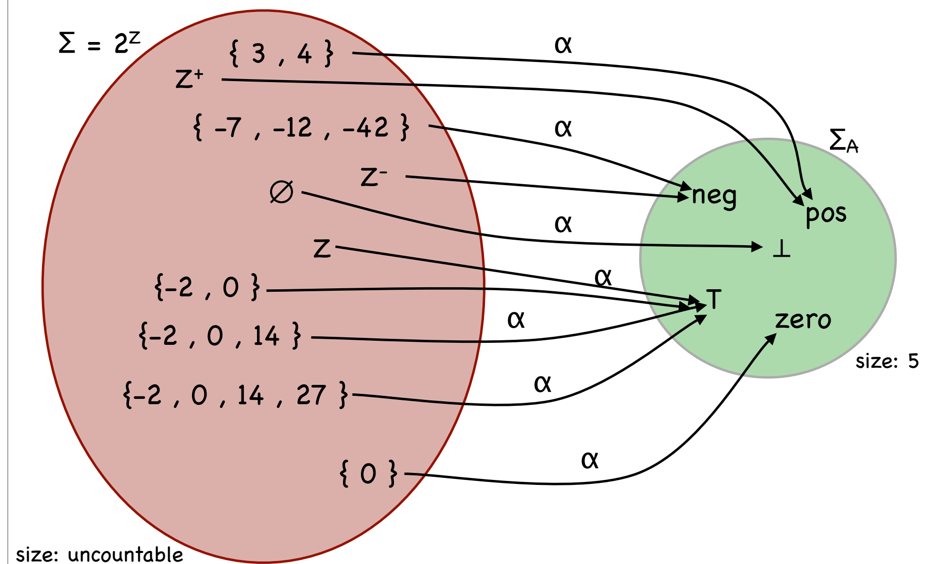
$\gamma(\ neg\ ) = \{ n \in Z \mid n < 0 \} = Z^-$

$\gamma(\ zero\ ) = \{ 0 \}$

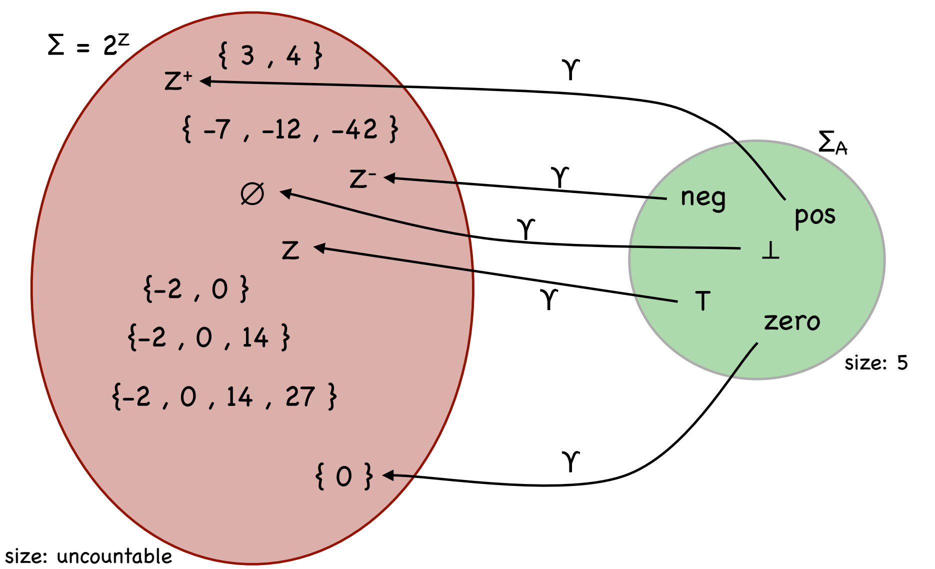
$\gamma(\ pos\ ) = \{ n \in Z \mid n > 0 \} = Z^+$

$\gamma(\ T\ ) = Z$

Abstraction function



Concretization function



Abstract version of multiplication

Concrete multiplication: $x_C : \Sigma \times \Sigma \to \Sigma$

Example: $\{ 1 , 2 \} \, x_C \, \{ 3 , 4 \} = \{ 3 , 4 , 6 , 8 \}$

Abstract multiplication: $x_A : \Sigma_A \times \Sigma_A \to \Sigma_A$

Abstract version of multiplication

Abstract multiplication: $\times_A : \Sigma_A \times \Sigma_A \rightarrow \Sigma_A$

$\times_A$	$\perp$	neg	zero	pos	T
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
neg	$\perp$	pos	zero	neg	T
zero	$\perp$	zero	zero	zero	zero
pos	$\perp$	neg	zero	pos	T
T	$\perp$	T	zero	T	T

Abstract analysis

Concrete question: $n_1 \times n_2 \overset{>}{\underset{<}{=}} ?\ 0$

Procedure:

Abstract n_1 and n_2 : $n_1^A = \alpha(\{n_1\})$ $n_2^A = \alpha(\{n_2\})$

Perform abstract multiplication : $n^A = n_1^A \times^A n_2^A$

Concretize n^A : $S = \Upsilon(n^A)$

if $S = Z^+$ then $n_1 \times n_2 > 0$

if $S = Z^-$ then $n_1 \times n_2 < 0$

if $S = \{0\}$ then $n_1 \times n_2 = 0$

if $S = Z$ then we don't know

Abstract analysis -- Example

$n_1 = 783{,}422$

$n_2 = 409{,}312$

$n_1 \times n_2 \overset{>}{\underset{<}{=}} ?\ 0$

Abstract n_1 and n_2 : $n_1^A = \alpha(\{n_1\}) = \text{pos}$

$n_2^A = \alpha(\{n_2\}) = \text{pos}$

Perform abstract multiplication : $n^A = n_1^A \times^A n_2^A$

$= \text{pos} \times^A \text{pos} = \text{pos}$

Concretize n^A : $S = \Upsilon(n^A) = Z^+$

if $S = Z^+$ then $n_1 \times n_2 > 0$

Conclude: $783{,}422 \times 409{,}312 > 0$

Abstract analysis -- Observations

- The choice of abstract domain was governed by the question. If the question had been to determine whether the result was even or odd, we would have chosen a different abstract domain and abstract semantics.

- The concrete domain is a partially ordered set with the subset relation $\subseteq$ as order.

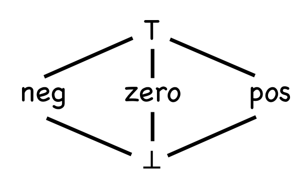
- We can also impose an order $<^A$ on the abstract domain:

$\perp <^A \text{neg}$ $\quad$ $\text{neg} <^A \text{T}$

$\perp <^A \text{zero}$ $\quad$ $\text{zero} <^A \text{T}$

$\perp <^A \text{pos}$ $\quad$ $\text{pos} <^A \text{T}$

$\perp <^A \text{T}$

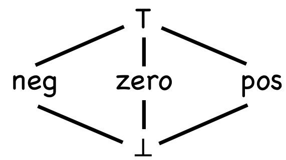


Abstract analysis -- Observations

- α and γ are both monotone:

$$\boxed{S_1 \subseteq S_2 \rightarrow \alpha(S_1) \leq^A \alpha(S_2)}$$

$$a_1 \leq^A a_2 \rightarrow \gamma(a_1) \subseteq \gamma(a_2)$$



Example:

$\{0\} \subseteq \{0, 1, 2\}$

$\alpha(\{0\}) = zero$

$\alpha(\{0, 1, 2\}) = T$

$zero <^A T$

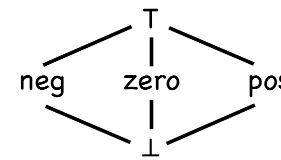
$\alpha(S)$	$= \perp$	if $S = \emptyset$	
$\alpha(\{0\})$	$= zero$		
$\alpha(S)$	$= neg$	if	$\forall n \in S \,.\, n < 0$
$\alpha(S)$	$= pos$	if	$\forall n \in S \,.\, n > 0$
$\alpha(S)$	$= T$	otherwise	

Abstract analysis -- Observations

- α and γ are both monotone:

$$S_1 \subseteq S_2 \rightarrow \alpha(S_1) \leq^A \alpha(S_2)$$

$$\boxed{a_1 \leq^A a_2 \rightarrow \gamma(a_1) \subseteq \gamma(a_2)}$$



Example:

$zero <^A T$

$\gamma(zero) = \{0\}$

$\gamma(T) = Z$

$\{0\} \subseteq Z$

$\gamma(\perp) = \emptyset$

$\gamma(neg) = \{n \in Z \mid n < 0\} = Z^-$

$\gamma(zero) = \{0\}$

$\gamma(pos) = \{n \in Z \mid n > 0\} = Z^+$

$\gamma(T) = Z$

Abstract analysis -- Observations

- The result of abstraction followed by concretization is something larger:

$$S \subseteq \gamma(\alpha(S))$$

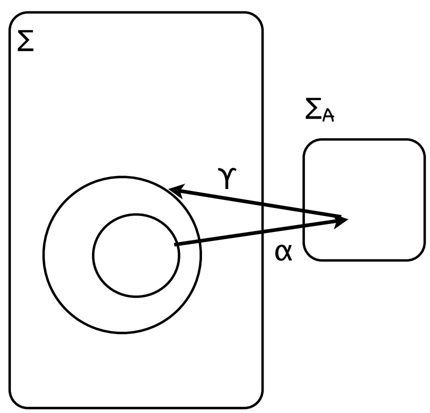
Example:

$S = \{3, 4\}$

$\alpha(S) = pos$

$\gamma(\alpha(S)) = \gamma(pos) = Z^+$

$\{3, 4\} \subseteq Z^+$



Abstract analysis -- Observations

- The result of concretization followed by abstraction is the same object:

$$\alpha(\gamma(a)) = a$$

Example:

$a = pos$

$\gamma(a) = Z^+$

$\alpha(\gamma(a)) = pos$

Abstract analysis -- Observations

- Abstract multiplication over-approximates

$$\gamma(a_1) \times \gamma(a_2) \subseteq \gamma(a_1 \times^A a_2)$$

(in this case it is actually equal)

$\times_A$	$\perp$	neg	zero	pos	T
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
neg	$\perp$	pos	zero	neg	T
zero	$\perp$	zero	zero	zero	zero
pos	$\perp$	neg	zero	pos	T
T	$\perp$	T	zero	T	T

we don't lose anything by doing abstract multiplications

Abstract analysis -- Observations

$$\gamma(a_1) \times \gamma(a_2) = \gamma(a_1 \times^A a_2)$$

Example:

$\gamma(\text{pos}) \times \gamma(\text{pos}) = Z^+ \times Z^+ = Z^+$

pos $\times^A$ pos = pos

$\gamma(\text{pos}) = Z^+$

$\times_A$	$\perp$	neg	zero	pos	T
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
neg	$\perp$	pos	zero	neg	T
zero	$\perp$	zero	zero	zero	zero
pos	$\perp$	neg	zero	pos	T
T	$\perp$	T	zero	T	T

Abstract analysis -- Observations

$$\gamma(a_1) \times \gamma(a_2) = \gamma(a_1 \times^A a_2)$$

Example:

$\gamma(\text{neg}) \times \gamma(\text{zero}) = Z^- \times \{0\} = \{0\}$

neg $\times^A$ zero = zero

$\gamma(\text{zero}) = \{0\}$

$\times_A$	$\perp$	neg	zero	pos	T
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
neg	$\perp$	pos	zero	neg	T
zero	$\perp$	zero	zero	zero	zero
pos	$\perp$	neg	zero	pos	T
T	$\perp$	T	zero	T	T

Galois connection

Let $(\Sigma_A, \leq^A)$ and $(\Sigma, \subseteq)$ be partially ordered sets.

A pair (α, γ) is a **Galois connection** if the following hold:

(1) $\alpha : \Sigma \rightarrow \Sigma_A$ and $\gamma : \Sigma_A \rightarrow \Sigma$

(2) α and γ are monotone

(3) $S \subseteq \gamma(\alpha(S))$ for all $S \in \Sigma$ and

$\boxed{\alpha(\gamma(a)) \leq^A a}$ for all $a \in \Sigma_A$

Note: if $\alpha(\gamma(a)) = a$ then (α, γ) is called a **Galois insertion**

Galois connection

The functions α and Υ determine each other: if one is given, the other follows

Given Υ :

$\alpha(S)$ is the smallest object in Σ_A that represents all of S :

$$\alpha(S) = \inf \{ a \in \Sigma_A \mid S \subseteq \Upsilon(a) \}$$
$$= \cap^A \{ a \in \Sigma_A \mid S \subseteq \Upsilon(a) \} \quad \text{(meet)}$$

Example: $S = \{3, 4\}$

$S \subseteq \Upsilon(T) \qquad S \subseteq \Upsilon(\text{pos})$

$\alpha(\{3, 4\}) = \inf \{\text{pos}, T\} = \text{pos}$

Galois connection

The functions α and Υ determine each other: if one is given, the other follows

Given α:

$\Upsilon(a)$ is the largest object in Σ that is fully described by a :

$$\Upsilon(a) = \sup \{ S \in \Sigma \mid \alpha(S) \leq^A a \}$$
$$= \mathbf{\cup} \{ S \in \Sigma \mid \alpha(S) \leq^A a \}$$

Example: $\alpha(\{3, 4\}) \leq^A \text{pos}$
$\alpha(\{17, 32, 42\}) \leq^A \text{pos}$
....

$\Upsilon(\text{pos}) = \{3, 4\} \cup \{17, 32, 42\} \cup \ldots\ldots = Z^+$

Galois connection

Given Υ :

$\alpha(S)$ is the smallest object in Σ_A that represents all of S :

$$\alpha(S) = \inf \{ a \in \Sigma_A \mid S \subseteq \Upsilon(a) \}$$
$$= \mathbf{\cap}^{\mathbf{A}} \{ a \in \Sigma_A \mid S \subseteq \Upsilon(a) \} \quad \text{(meet)}$$

Given α:

$\Upsilon(a)$ is the largest object in Σ that is fully described by a :

$$\Upsilon(a) = \sup \{ S \in \Sigma \mid \alpha(S) \leq^A a \}$$
$$= \mathbf{\cup} \{ S \in \Sigma \mid \alpha(S) \leq^A a \} \quad \text{(join)}$$