Abstract Interpretation

Prog = (C, I, T, F)

C : infinite sets of prog. states
$I \subseteq C$: initial states
$T \subseteq C \times C$: transition relation
$F \subseteq C$: final states

Abstractions = pairs of functions (α, γ)

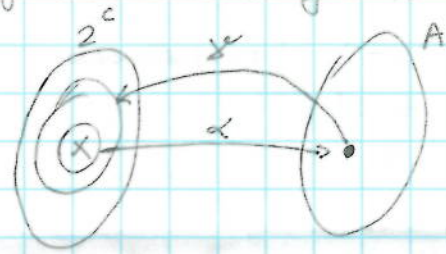
- α maps a set of states to an abstract element in A

$$\alpha : 2^C \to A$$

- γ maps an abstract element to a set of concrete states

$$\gamma : A \to 2^C$$

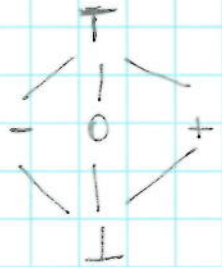
- sets of states ordered by $\subseteq$ (subset)



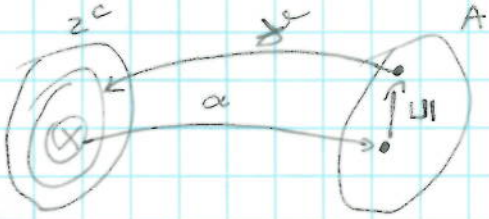
$$\forall x \in C \,.\, x \subseteq \gamma(\alpha(x))$$

- ordering in the abstract domain: A is embedded in a lattice with
  - $\sqsubseteq$ partial order
  - $\sqcup$ join
  - $\sqcap$ meet

Ex: signs abstraction $A = \{\bot, \top, -, 0, +\}$



- Galois connection



Ex: signs abstraction

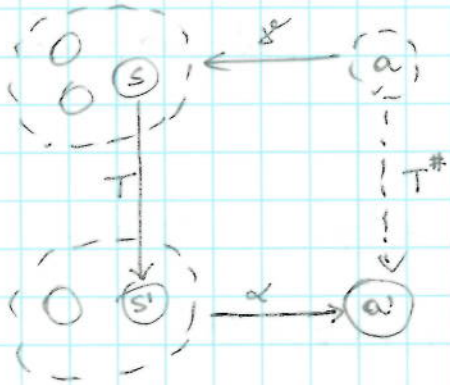
$A = 2^{\mathbb{Z}}$

$$\beta(i) = \begin{cases} 0, & i = 0 \\ +, & i > 0 \\ -, & i < 0 \end{cases}$$

$$\alpha(I) = \bigsqcup \{\beta(i) \mid i \in I\}$$

$\gamma(a) = ??$

Abstract Transitions



$T^{\#} \supseteq \alpha \circ T \circ \gamma$

$T^{\#}(a) \supseteq \alpha(T(\gamma(a)))$

$= (\text{best transformer})$

Ex: express abstraction

	in	out
$x := c$	any	$\alpha(\{c\})$
$x := x+1$	$\{0,+\}$	$+$
	$\{-,\top\}$	$\top$
	$\{\bot\}$	$\bot$
assume $(x<0)$	$\{-,\top\}$	$-$
	$\{0,+,\bot\}$	$\bot$

↑ for any of these, the result is $\bot$

- more arithmetic domains in the abstract interpretation course

Predicate Abstraction

$\alpha : 2^C \to A$ $\quad$ $\gamma : A \to 2^C$

given k predicates, $A = 2^{2^k}$ - sets of boolean valuations
a subset of the set of predicates (those that are true)

$\varphi \in 2^C$ $\quad$ $\varphi^\# \in A = 2^{2^k}$

- assume a set of states is described by a formula
- assume an element of A is a boolean formula over variables $B = \{b_1, \dots, b_k\}$, b_i for predicate P_i

$$\alpha(\varphi) = \exists \overline{x}. \varphi \wedge (\wedge b_i \Leftrightarrow P_i)$$

$\overline{x}$ the variables in φ

$$\gamma(\varphi^\#) = \varphi^\#[b_i \to P_i]$$

Ex: $\varphi = (x = 1 \vee x = 2)$

$P = \{x < 10, x > 1\}$ $\quad$ compute $\varphi^\# = \exists x. [(x=1 \vee x=2) \wedge (b_1 \Leftrightarrow x<10) \wedge (b_2 \Leftrightarrow x>1)]$

$B = \{b_1, b_2\}$

Approach: enumerate all conjuctive clauses (conjunctions of literals) cubes

$x = 1 \vee x = 2$

$x < 10 \wedge x > 1$ (non-empty intersection)	$b_1 \wedge b_2$	YES
$x < 10 \wedge x \leq 1$ (?)	$b_1 \wedge \neg b_2$	YES
$x \geq 10 \wedge x > 1$ (?)	$\neg b_1 \wedge b_2$	NO
~~$x \geq 10 \wedge x \leq 1$~~	~~$\neg b_1 \wedge \neg b_2$~~	unsat

$$\varphi^\# = (b_1 \wedge b_2) \vee (b_1 \wedge \neg b_2)$$

We could have done this from the beginning for each disjunct separately

Abstract weakest preconditions

Usually, $WP(x=e, \varphi) = \varphi[e/x]$

Ex: $WP(x=x+1,\ x<5) = (x+1<5) = (x<4)$

Given a predicate P, compute $WP(x=e, P)$?

Issue: $WP(x=e, P) \notin$ set of predicates considered for abstraction

Ex: a set of predicates $\mathcal{P} = \{x<5, x=2\}$

$WP(x=x+1, x<5) \notin \mathcal{P}$

Solution: compute the largest DNF formula implying WP (strengthening the weakest precondition)

→ disjunction of cubes (formed of a literal for each predicate)

Ex: $x=2 \Rightarrow x<4$

$\underline{WP^{\#}}(x=x+1, x<5) = x=2$

↓

abstracting weakest precondition → now it's just a precondition (not necessarily the strongest)

Abstract Boolean programs

Abstracting assignements

$l: x=e \longrightarrow$

```
if (WP#(x=e, P1)) then b1 = true;
else if (WP#(x=e, ¬P1)) then b1 = false;
else b1 = *;
```

b_i is true after l if
$WP^{\#}(x=e, P_i)$ holds before l

false after l if
$WP^{\#}(x=e, \neg P_i)$ holds before l

*, otherwise

Abstracting conditionals

```
if (φ) then
  ——
else ——.
```

→

```
if (*) {
  assume ¬ W(¬φ)
  ——
} else {
  assume ¬ W(φ)
  ——
}
```

W(Ψ) = the largest DNF formula implying Ψ