TVLA: A system for generating abstract interpreters

Mooly Sagiv

Tel Aviv University

http://www.cs.tau.ac.il/tvla

Tentative Schedule

- Abstract interpretation in the nutshell
- Abstractions of heap allocated storage
- More precise abstract interpretation
- TVLA demo + applications

Abstract Interpretation in the nutshell

Mooly Sagiv

- Principles of Program Analysis
  F. Nielson, H. Nielson, C.L. Hankin (2, 4)
- Patrick Cousot's homepage
- Semantics with Application Nielson & Nielson

Outline

- Goals
- Lattices
- System of equations
- Fixed points
- Chaotic iterations
- Galois Connections
- Soundness

Relation to Program Verification

Abstract Interpretation

- Fully automatic
- But can benefit from specification
- Applicable to a programming language
- Can be very imprecise
- May yield false alarms
- Identify interesting bugs
- Can provide counter examples
- Establish non-trivial properties using effective algorithms

Program Verification

- Requires specification and loop invariants
- Not decidable
- Program specific

- Relative complete
- Always provide counter examples
- Provide useful documentation

Abstract Interpretation Static Analysis

- Automatically identify program properties
  - No user provided loop invariants
- Sound but incomplete methods
  - But can be rather precise
- Non-standard interpretation of the program operational semantics
- Usages
  - Compiler optimization
    - Collect static information for program transformations
  - Code quality tools
    - Identify potential bugs
    - Prove the absence of runtime errors
    - Partial correctness

Example Program

```
    /* x=⊤ */

while  (x !=1)   do {   /* x=⊤ */
          if   (x %2) == 0
    /* x=E */             { x := x / 2; }        /* x=⊤ */
                    else
    /* x=O */            { x := x * 3 + 1;       /* x=E */
                           assert (x %2 ==0); }
}
/* x=O*/
```

Complete Lattices

- A poset is a complete lattice with $\sqsubseteq$
  - Every subset has least and upper bounds

- $(L, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$
  - $\bot = \bigsqcup \emptyset = \bigsqcap L$
  - $\top = \bigsqcup L = \bigsqcap \emptyset$

Integer Powerset Lattice

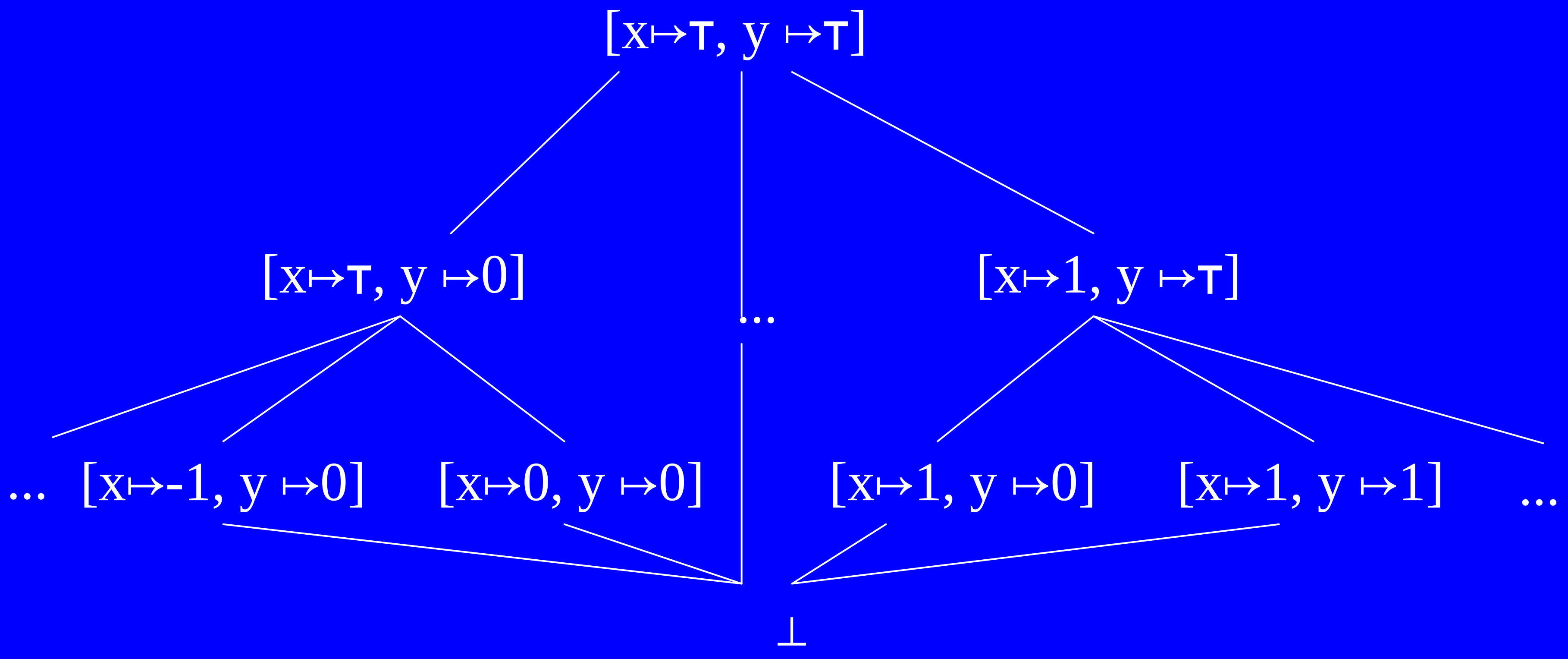
$$\begin{tikzcd}
 & & & & Z & & & \\
 & & & & \vdots & & & \\
 & \{-3, -2\} & \{-3, -1\} & \{-1, 0\} & \{0, 1\} & & \{1, 2\} & \ldots \\
\ldots & \{-3\} \arrow[u] \arrow[ur] & \{-2\} \arrow[ul] & \{-1\} \arrow[ul] \arrow[u] & \{0\} \arrow[ul] \arrow[u] & \{1\} \arrow[ul] \arrow[ur] & \{2\} \arrow[u] & \ldots \\
 & & & & \varnothing \arrow[ullll] \arrow[ulll] \arrow[ull] \arrow[u] \arrow[ur] \arrow[urr] \arrow[urrr] & & &
\end{tikzcd}$$

Odd/Even Lattice

$$\begin{tikzcd}
 & \top \arrow[dl, no head] \arrow[dr, no head] & \\
O \arrow[dr, no head] & & E \arrow[dl, no head] \\
 & \bot &
\end{tikzcd}$$

Constant Propagation Lattice

$[Var \to Z^{\top}]^{\perp}$



Monotone Functions over lattice

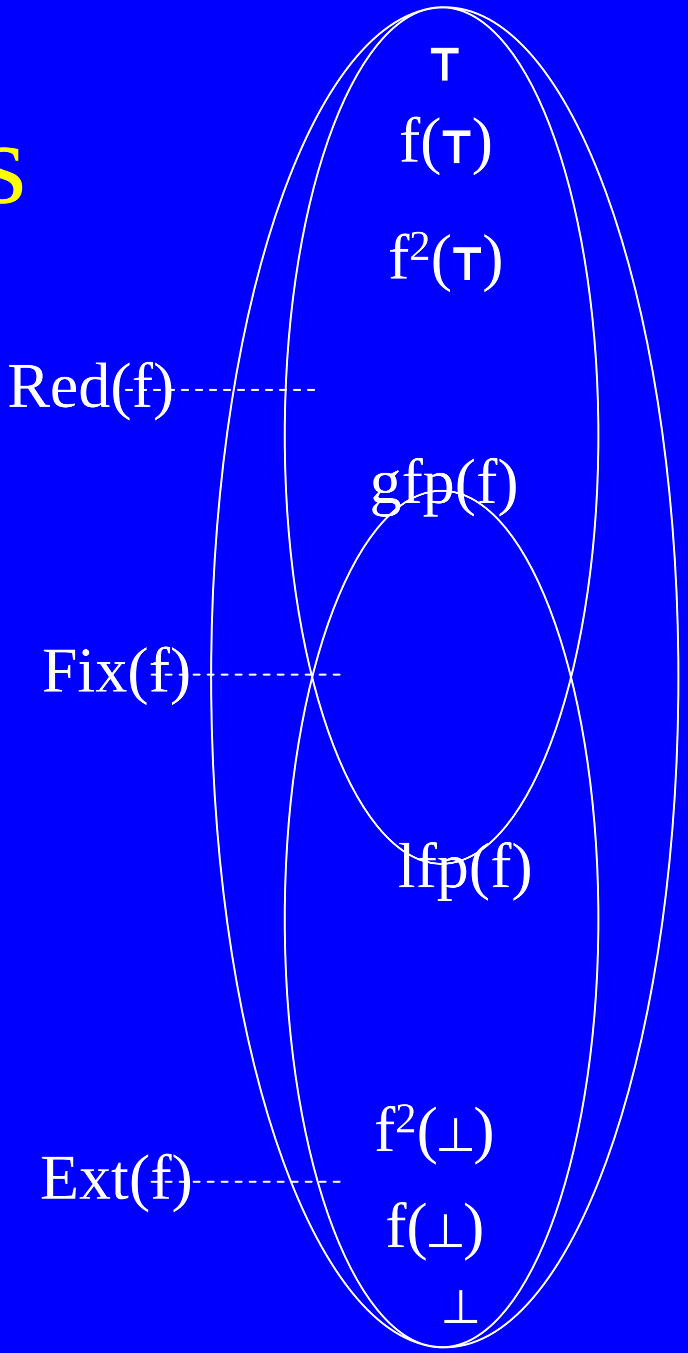
$f: L \to L$ is monotone

$l_1, l_2 \in L$:

$$l_1 \sqsubseteq l_2 \Rightarrow f(l_1) \sqsubseteq f(l_2)$$

Fixed Points

- A monotone function $f: L \rightarrow L$ where $(L, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$ is a complete lattice
- $Fix(f) = \{ l: l \in L, f(l) = l\}$
- $Red(f) = \{l: l \in L, f(l) \sqsubseteq l\}$
- $Ext(f) = \{l: l \in L, l \sqsubseteq f(l)\}$
  - $l_1 \sqsubseteq l_2 \Rightarrow f(l_1) \sqsubseteq f(l_2)$
- Tarski's Theorem 1955: if f is monotone then:
  - $lfp(f) = \sqcap Fix(f) = \sqcap Red(f) \in Fix(f)$
  - $gfp(f) = \sqcup Fix(f) = \sqcup Ext(f) \in Fix(f)$



Concrete and Abstract Interpretation

+	0	1	2	3	…
0	0	1	2	3	…
1	1	2	3	4	…
2	2	3	4	5	…
3	3	4	5	6	…
⋮	⋮	⋮	⋮	⋮	

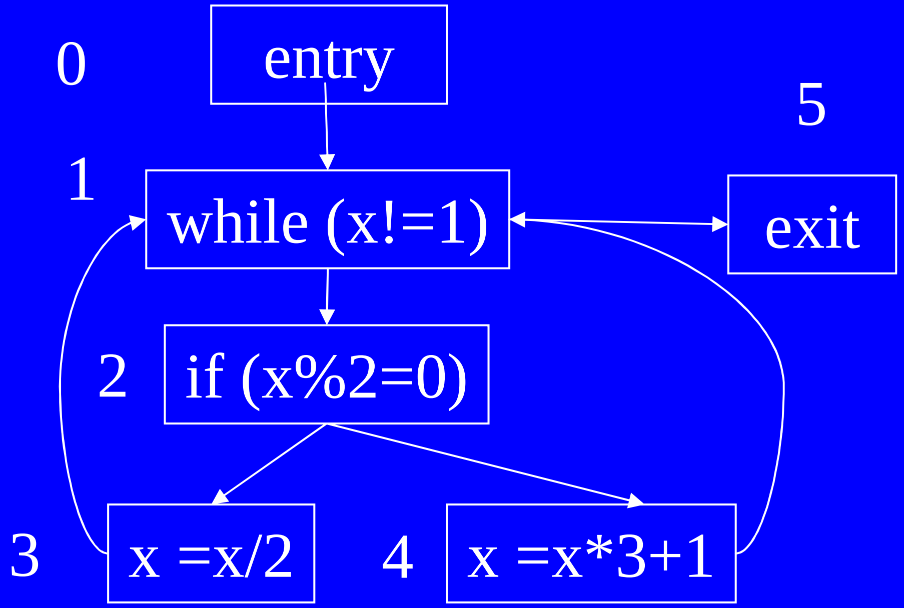
*	0	1	2	3	…
0	0	0	0	0	…
1	0	1	2	3	…
2	0	2	4	6	…
3	0	3	6	9	…
⋮	⋮	⋮	⋮	⋮	

$+^{\#}$	$\perp$	**O**	**E**	$\top$
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
O	$\perp$	E	O	$\top$
E	$\perp$	O	E	$\top$
$\top$	$\perp$	$\top$	$\top$	$\top$

$*^{\#}$	$\perp$	**O**	**E**	$\top$
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
O	$\perp$	O	E	$\top$
E	$\perp$	E	E	E
$\top$	$\perp$	$\top$	E	$\top$

Systems of Equations

- Construct a control flow graph
- A system of equations defined the required solution at every control flow node
- The value of a node depends on its predecessors
- The least fixed point can be computed iteratively



$f_{<0,1>}(X) = X$

$f_{<1,2>}(X) = X$

$f_{<1,5>}(X) = X \sqcap O$

$f_{<2,3>}(X) = X \sqcap E$

$f_{<2,3>}(X) = X \sqcap O$

$f_{<3,1>}(X) = X \;/^{\#}\; E$

$f_{<4,1>}(X) = (X \;*^{\#}\; O) \;+^{\#}\; O$

$X_0 = \top$

$X_1 = f_{<0,1>}(X_0) \sqcup f_{<3,1>}(X_3) \sqcup f_{<4,1>}(X_4)$

$X_2 = f_{<1,2>}(X_1)$

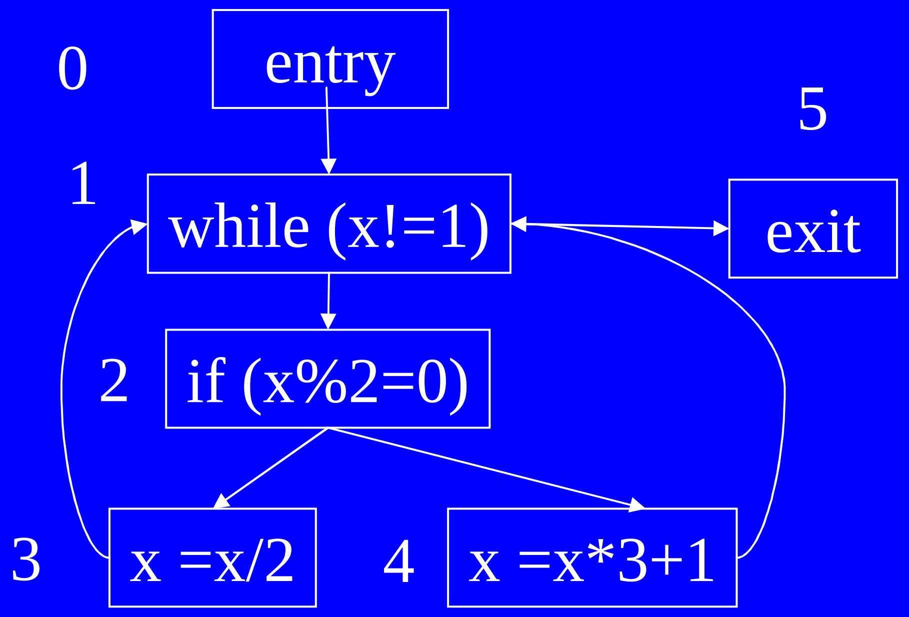
$X_3 = f_{<2,3>}(X_2)$

$X_4 = f_{<2,4>}(X_2)$

$X_5 = f_{<1,5>}(X_1)$

Specialized Chaotic Iterations

```
Chaotic(G(V, E): Graph, entry: Node, L: Lattice, $\iota$: L, f: E $\rightarrow$(L $\rightarrow$L) ){
 for each v in V to n do $X_{in}[v] := \perp$
 X[entry] = $\iota$
 WL = {entry}
 while (WL $\neq$ $\emptyset$ ) do
   select and remove an element u $\in$ WL
   for each v, such that. (u, v) $\in$ E do
          temp = $f_{<e>}$ ($X_{in}[u]$)
          new := $X_{in}(v) \sqcup$ temp
          if (new $\neq X_{in}[v]$) then
                 $X_{in}[v]$ := new;
                 WL := WL $\cup$ {v}
```



$f_{<0,1>}(X) = X$

$f_{<1,2>}(X) = X$

$f_{<1,5>}(X) = X \sqcap O$

$f_{<2,3>}(X) = X \sqcap E$

$f_{<2,3>}(X) = X \sqcap O$

$f_{<3,2>}(X) = X \;/^{\#}\; E$

$f_{<4,2>}(X) = (X \;*^{\#}\; O) \;+^{\#}\; E$

WL	V	$X_{in}[v]$
	in	X[0] :=⊤
{0}	0	X[1] := ⊤
{1}	1	X[2]:=⊤, X[5]:= O
{2,5}	2	X[3]:= E X[4] := O
{3,4,5}	3	
{4, 5}	4	
{5}	5	

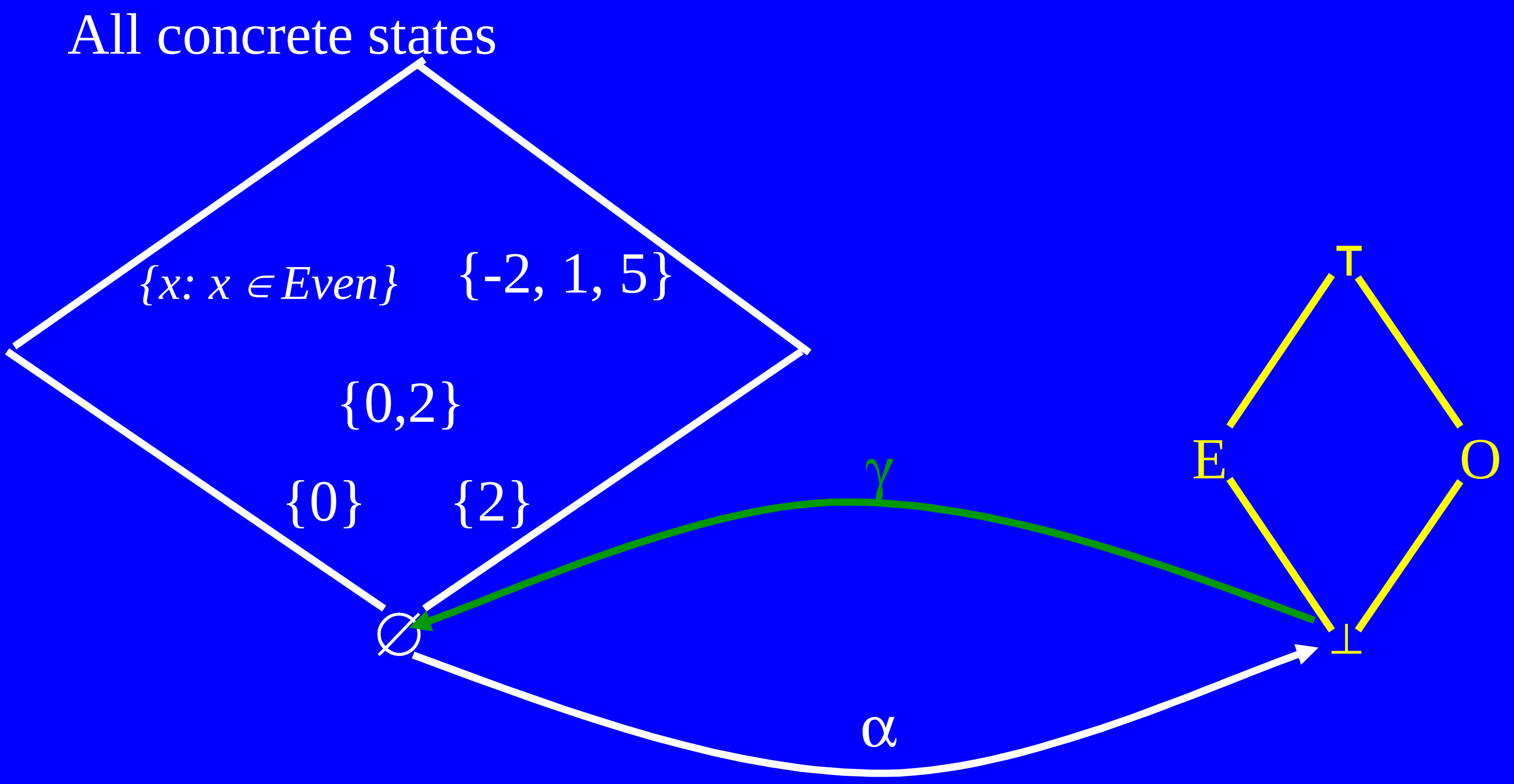
Soundness

- Detected Even (Odd) numbers are indeed such
- Every error will be detected
- The least fixed points represents all occurring runtime states

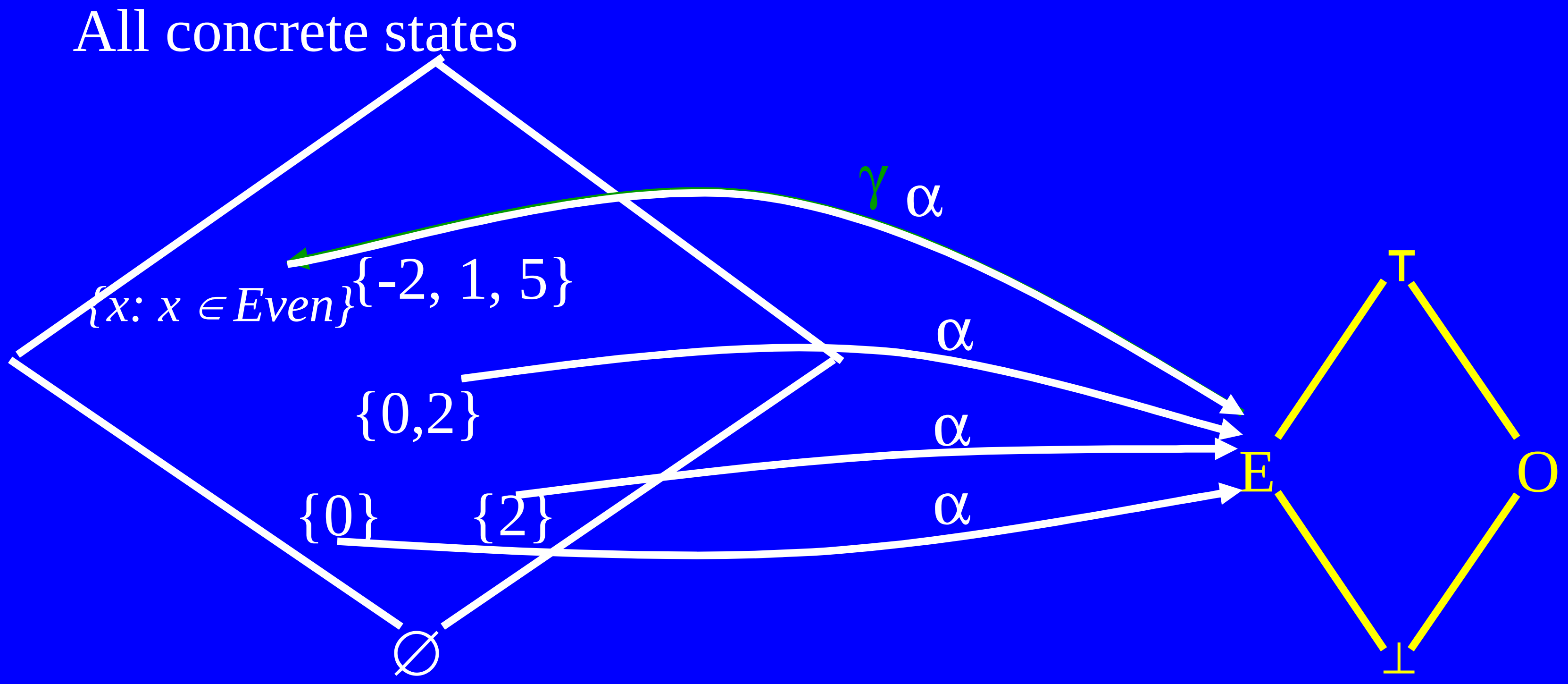
Galois Connections

- α: C $\rightarrow$A and γ: A $\rightarrow$C
- The pair of functions (α, γ) form Galois connection if
  - α and γ are monotone
  - $\forall$ a $\in$ A: $\alpha(\gamma(a)) \sqsubseteq$ a
  - $\forall$ c $\in$ C: c $\sqsubseteq \gamma(\alpha(C))$
- Alternatively if:
  $\forall$ c $\in$ C, $\forall$ a $\in$ A
  $\alpha(c) \sqsubseteq a$ iff $c \sqsubseteq \gamma(a)$
- α and γ uniquely determine each other

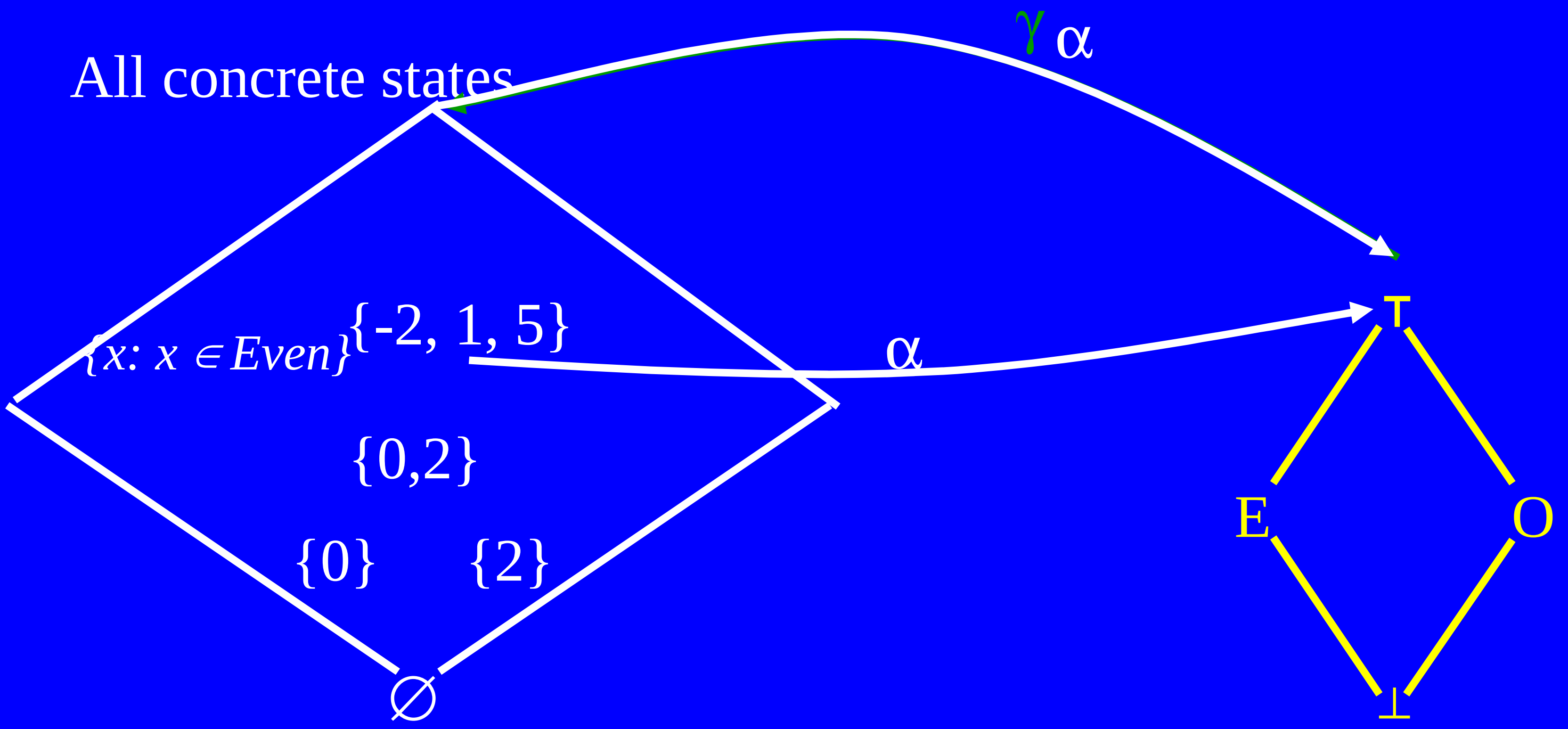
Odd/Even Abstract Interpretation



Odd/Even Abstract Interpretation



Odd/Even Abstract Interpretation



Odd/Even Abstract Interpretation

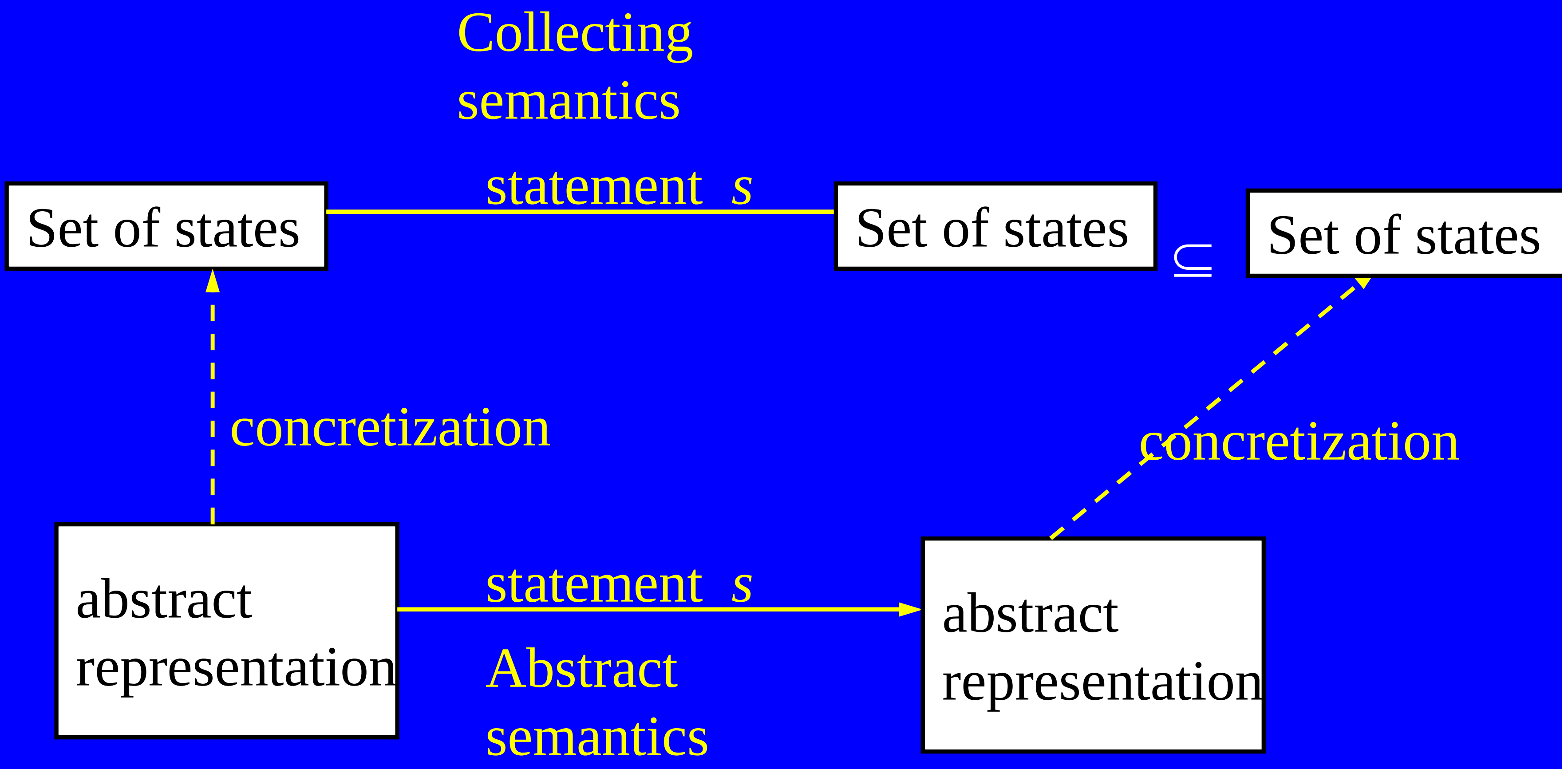
```
α(S) =  if S = ∅ then ⊥
        elseif S ⊆ Even then E
        elseif S ⊆ Odd then O
        else ⊤

γ(X) =  if X = ⊥ then ∅
        elseif X =E then Even
        elseif X=O then Odd
        else  Z
```

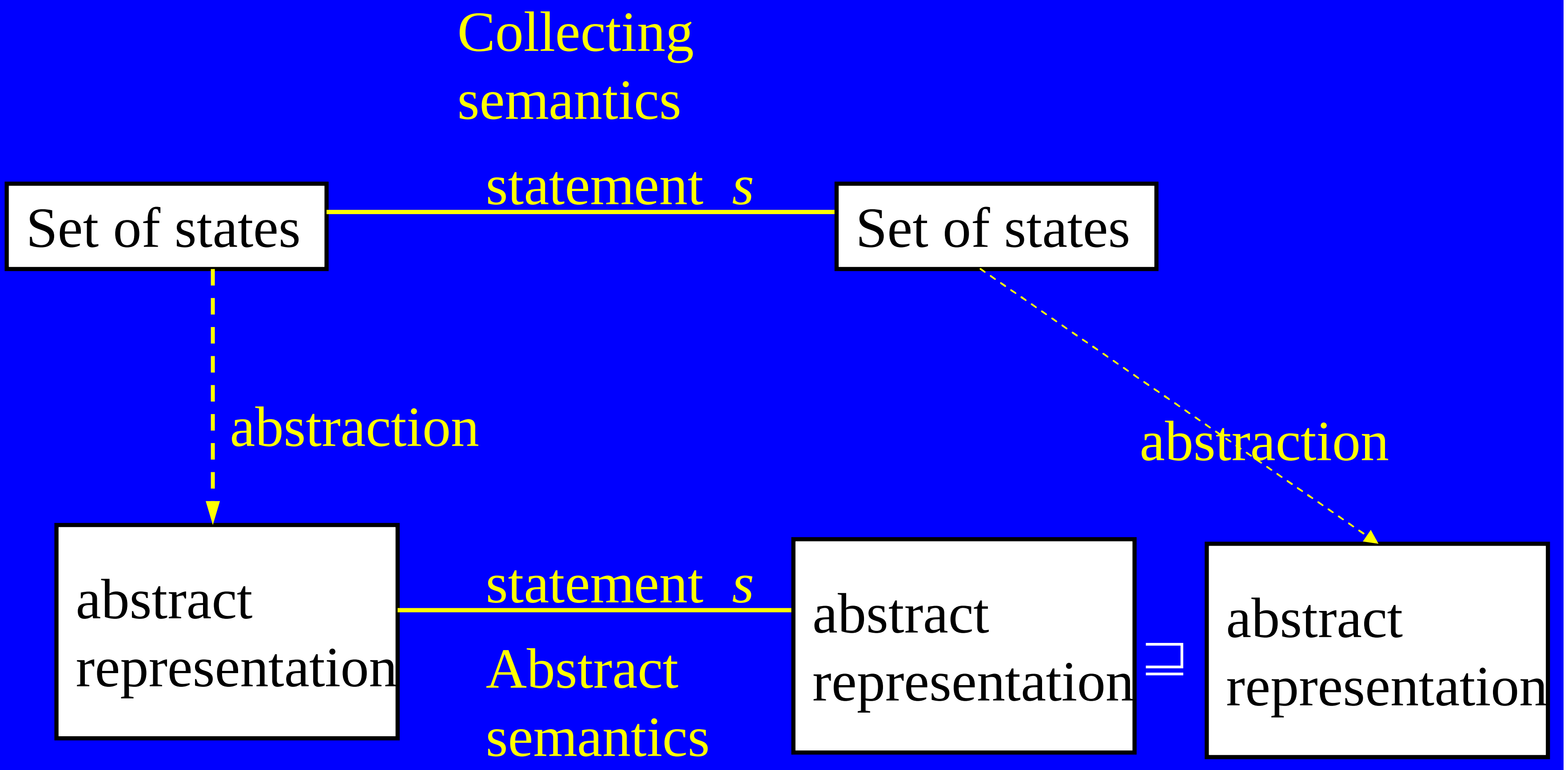
Homework

- Define a Galois connection for constant propagation
- Show that every Galois connection abstraction and concretization determine each other
- Read the orange booklet on TVLA

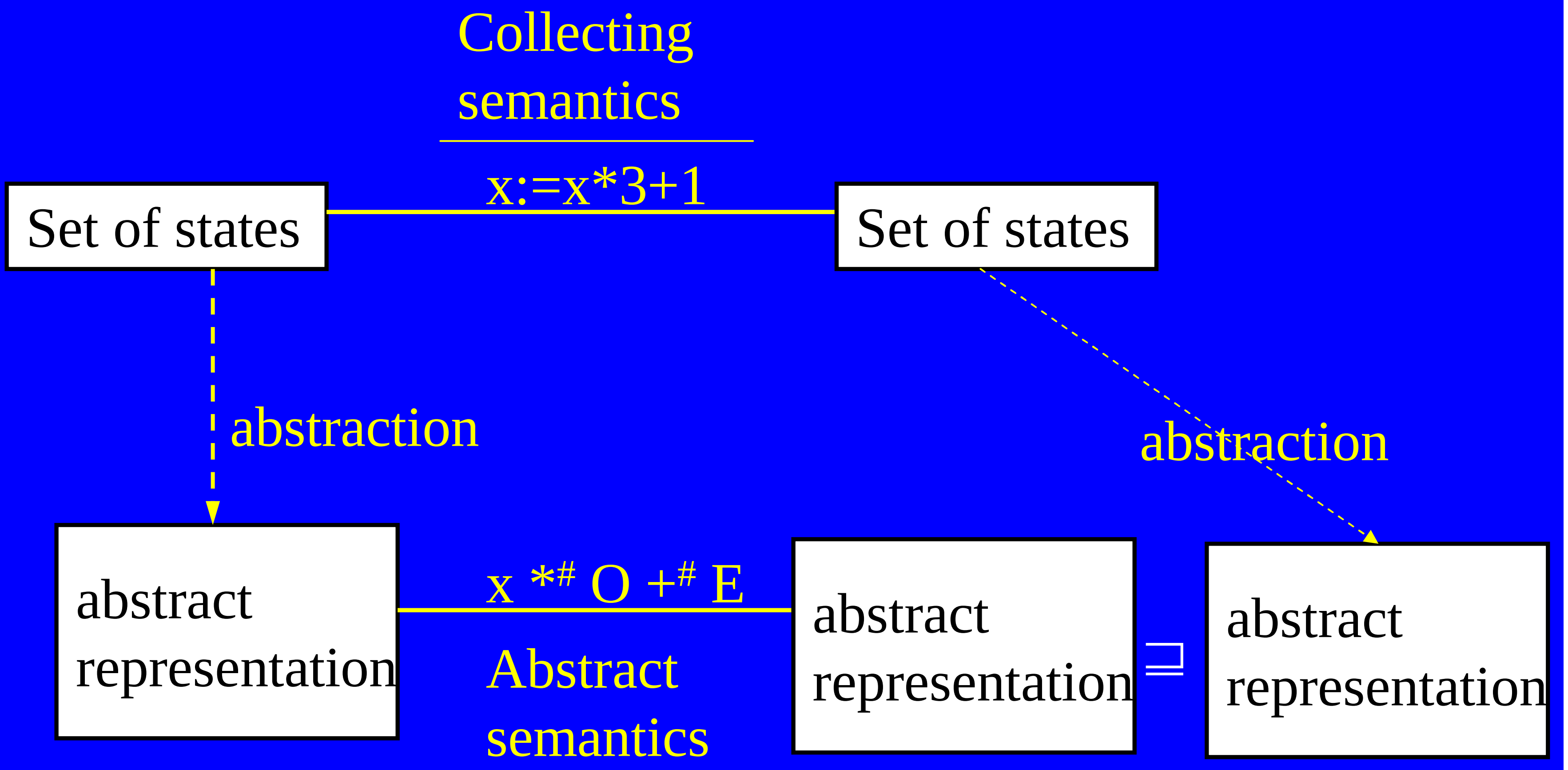
Abstract (Conservative) interpretation



Abstract (Conservative) interpretation



Abstract (Conservative) interpretation



Soundness Theorem(1) [Cousot & Cousot 1979]

1. Let (α, γ) 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. $\forall a \in A: f(\gamma(a)) \sqsubseteq \gamma(f^{\#}(a))$

$$\mathrm{lfp}(f) \sqsubseteq \gamma(\mathrm{lfp}(f^{\#}))$$

$$\alpha(\mathrm{lfp}(f)) \sqsubseteq \mathrm{lfp}(f^{\#})$$

Soundness Theorem(2) [Cousot & Cousot 1979]

1. Let (α, γ) 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. $\forall c \in C: \alpha(f(c)) \sqsubseteq f^{\#}(\alpha(c))$

$\alpha(\mathrm{lfp}(f)) \sqsubseteq \mathrm{lfp}(f^{\#})$

$\mathrm{lfp}(f) \sqsubseteq \gamma(\mathrm{lfp}(f^{\#}))$

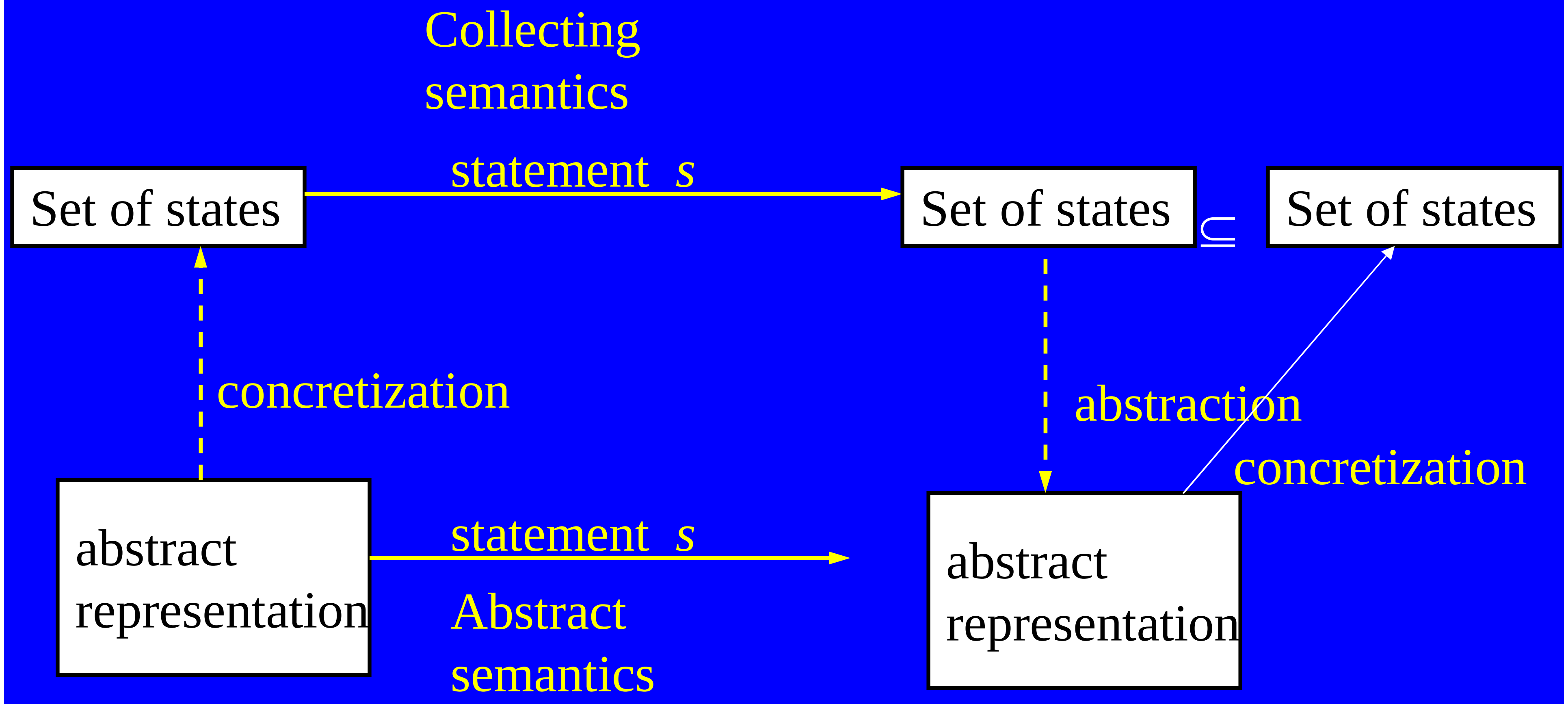
Soundness Theorem(3) [Cousot & Cousot 1979]

1. Let (α, γ) 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. $\forall a \in A: \alpha(f(\gamma(a))) \sqsubseteq f^{\#}(a)$

$\alpha(lfp(f)) \sqsubseteq lfp(f^{\#})$

$lfp(f) \sqsubseteq \gamma(lfp(f^{\#}))$

Best (Conservative) interpretation



C String Static Verifier (Nurit Dor) (CSSV)

- Detects string violations
  - Buffer overflow (update beyond bounds)
  - Unsafe pointer arithmetic
  - References beyond null termination
  - Unsafe library calls
- Assumes procedure contracts
- Handles full C
  - Multi-level pointers, pointer arithmetic, structures, casting, …
- Applied to real programs
  - Public domain software
  - C code from Airbus
  - Very few false alarms

Origins of Abstract Interpretation

- [Naur 1965] The Gier Algol compiler
  "A process which combines the operators and operands of the source text in the manner in which an actual evaluation would have to do it, but which operates on descriptions of the operands, not their value"
- [Reynolds 1969] Interesting analysis which includes infinite domains (context free grammars)
- [Syntzoff 1972] Well foundedness of programs and termination
- [Cousot and Cousot 1976,77,79] The general theory
- [Kamm and Ullman, Kildall 1977] Algorithmic foundations
- [Tarjan 1981] Reductions to semiring problems
- [Sharir and Pnueli 1981] Foundation of the interprocedural case
- [Allen, Kennedy, Cock, Jones, Muchnick and Scwartz]

Conclusions

- Abstract interpretation is a powerful technique
- Scales to different programming styles
- Allows specifications
- Proving near-commutativity becomes hard for complicated language constructs and abstractions
  - Pointers
  - Destructive updates