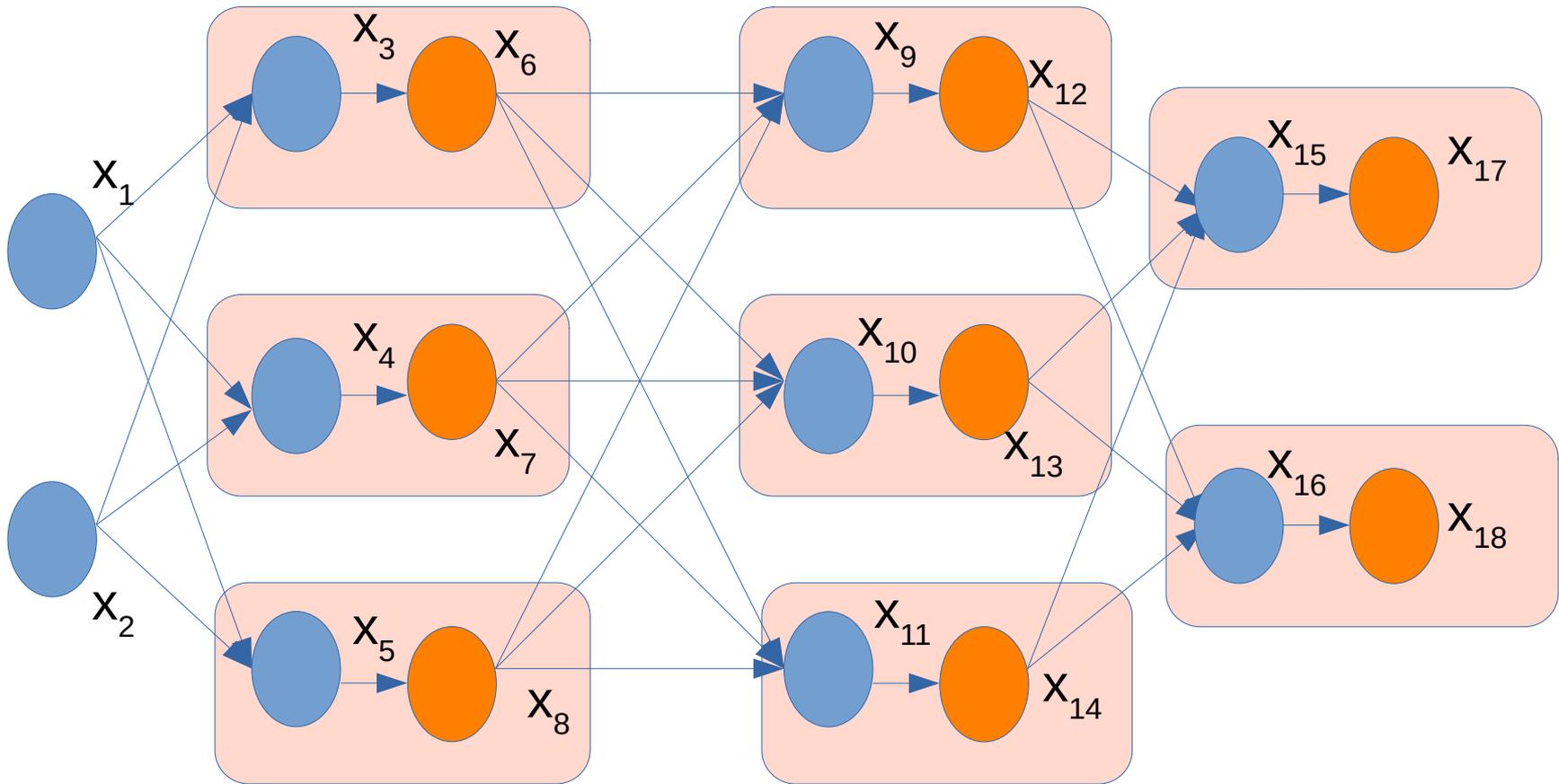


CS781:
A Quick Primer on
Abstract Interpretation for
Neural Networks

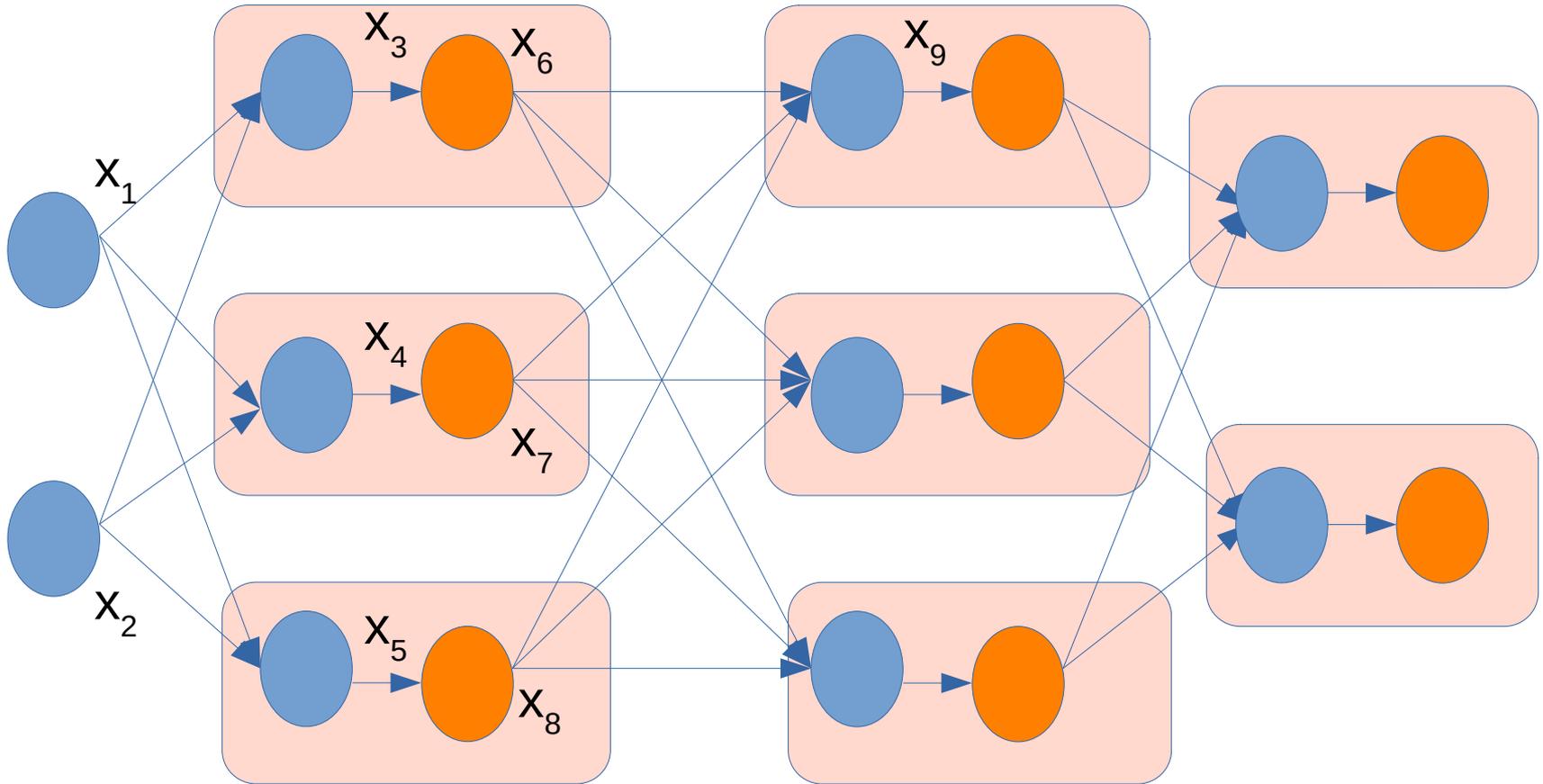
Supratik Chakraborty
IIT Bombay

Notion of State in Neural Network



State: $(x_1, x_2, \dots, x_{18})$ in \mathbb{R}^{18}

State Change in Feed-Forward Neural Network

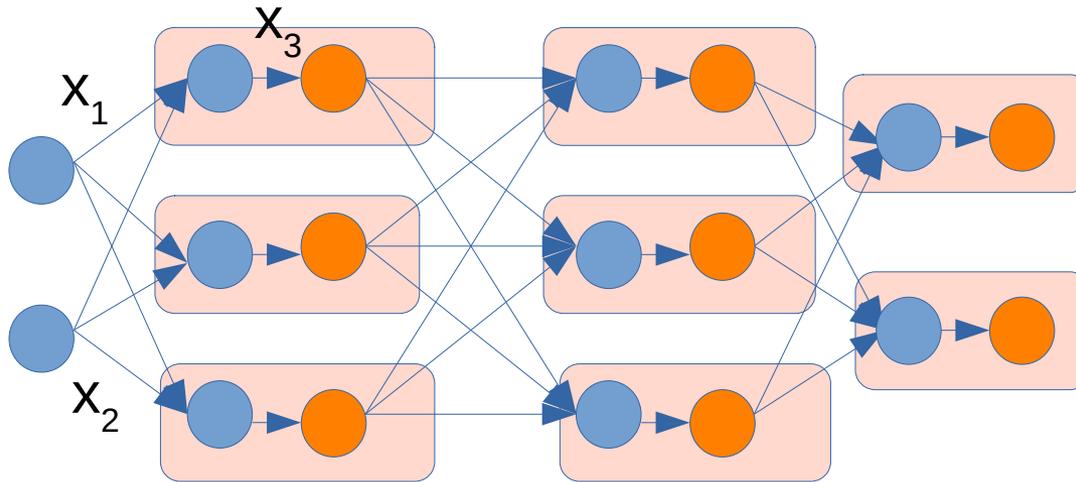


$(x'_1, x'_2, \dots, x'_{17}, x'_{18}) = f(x_1, x_2, \dots, x_{18})$, where

x'_i = new value of x_i obtained as per neuron i 's functionality;

$x'_1 = x_1$, and $x'_2 = x_2$ (inputs assumed to be stable)

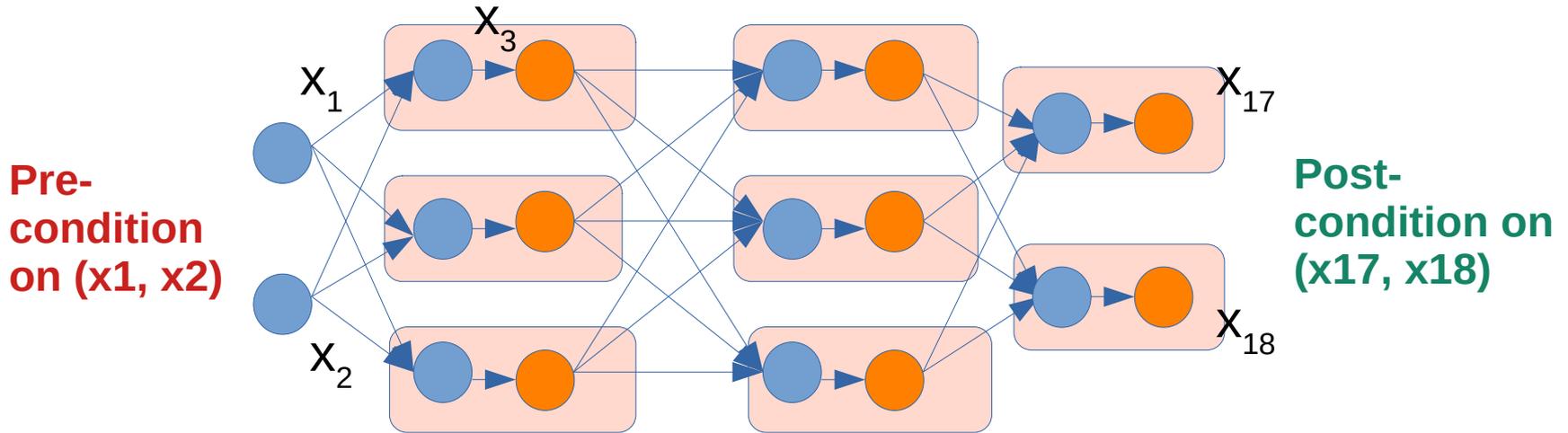
State Change in Feed-Forward NN as a sequence of instrns



$$\begin{aligned} (x'_1, x'_2, x'_3, \dots, x'_{18}) &= f(x_1, x_2, x_3, \dots, x_{18}); \\ (x''_1, x''_2, x''_3, \dots, x''_{18}) &= f(x'_1, x'_2, x'_3, \dots, x'_{18}); \\ &\dots \text{ repeat 18 times (at most)} \end{aligned}$$

NN computation: a sequence of state transitions caused by seq of instructions

Proving Property of a FF NN

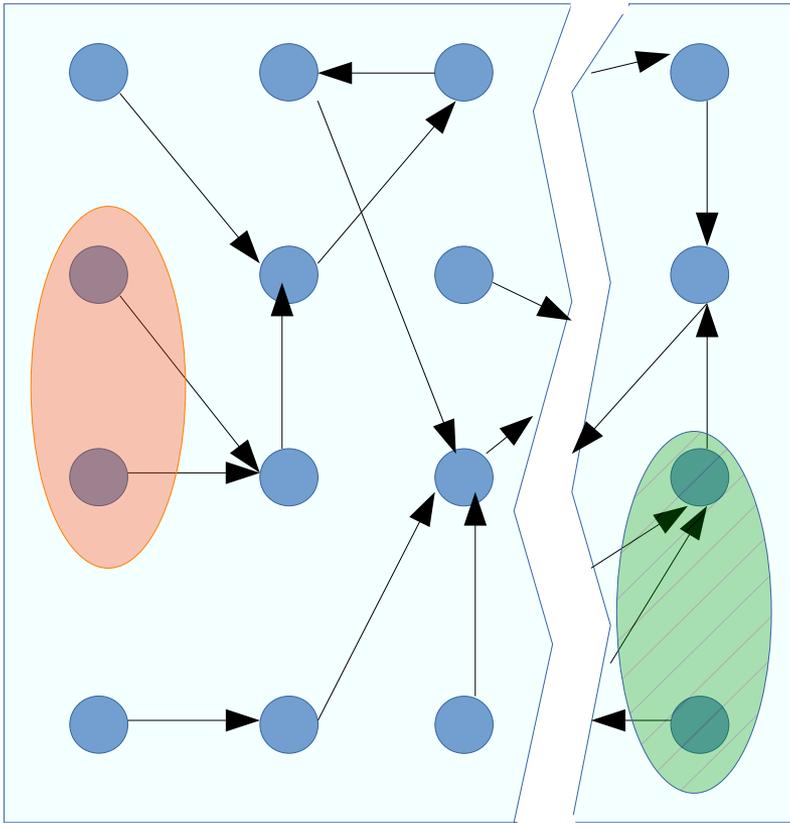


{Pre-condition on (x1, x2)}

$$\begin{aligned}(x'_1, x'_2, \dots, x'_{18}) &= f(x_1, x_2, \dots, x_{18}); \\(x''_1, x''_2, \dots, x''_{18}) &= f(x'_1, x'_2, \dots, x'_{18}); \\&\dots\end{aligned}$$

{Post-condition on (x17, x18)}

NN Computation as a State Transition System



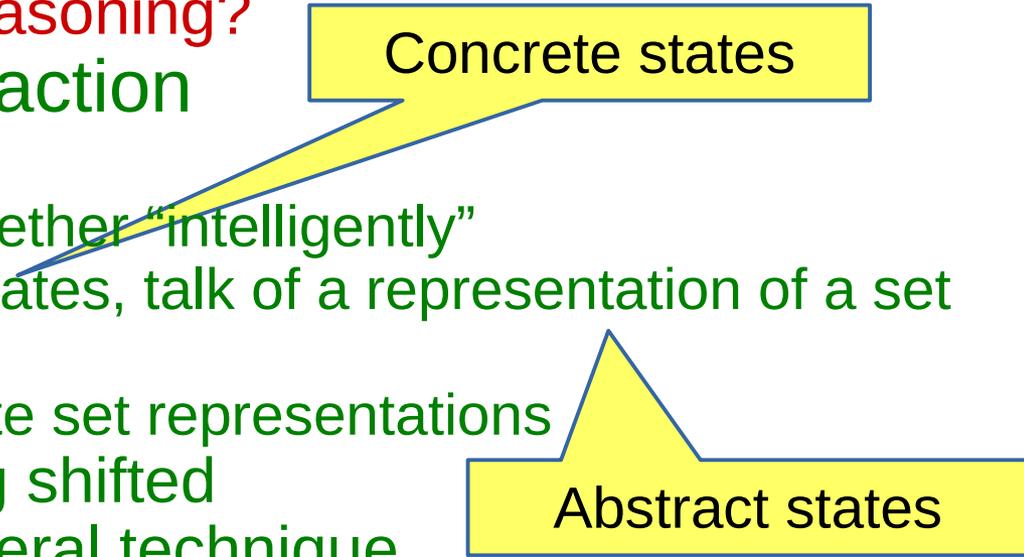
{Pre-condition on (x1, x2)}

$$(x'_1, x'_2, \dots, x'_{18}) = f(x_1, x_2, \dots, x_{18});$$
$$(x''_1, x''_2, \dots, x''_{18}) = f(x'_1, x'_2, \dots, x'_{18});$$

...

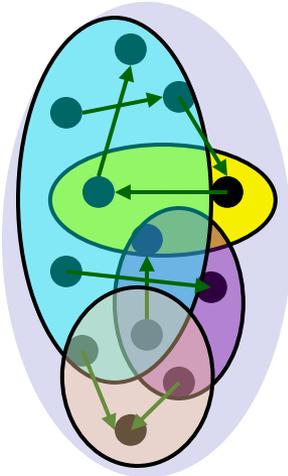
{Post-condition on (x17, x18) }

Dealing with State Space Size

- Infinite state space
 - Difficult to represent using state transition diagram
 - Can we still do some reasoning?
 - Solution: Use of abstraction
 - Naive view
 - Bunch sets of states together “intelligently”
 - Don't talk of individual states, talk of a representation of a set of states
 - Transitions between state set representations
 - Granularity of reasoning shifted
 - Extremely powerful general technique
 - Allows reasoning about large/infinite state spaces
- 
- The diagram consists of two yellow boxes with blue borders. The top box is labeled 'Concrete states' and has a yellow arrow pointing downwards and to the left towards the bottom box. The bottom box is labeled 'Abstract states' and has a yellow arrow pointing upwards and to the right towards the top box. The arrow from the concrete states box points to the text 'Bunch sets of states together “intelligently”' in the list.

A Generic View of Abstraction

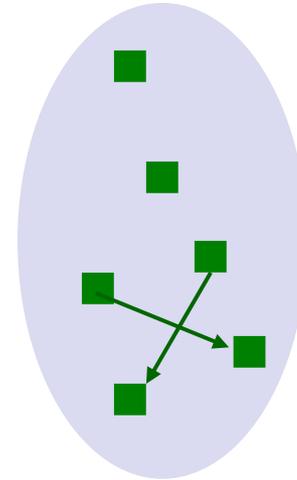
Set of concrete states



Abstraction (α)



Set of abstract states

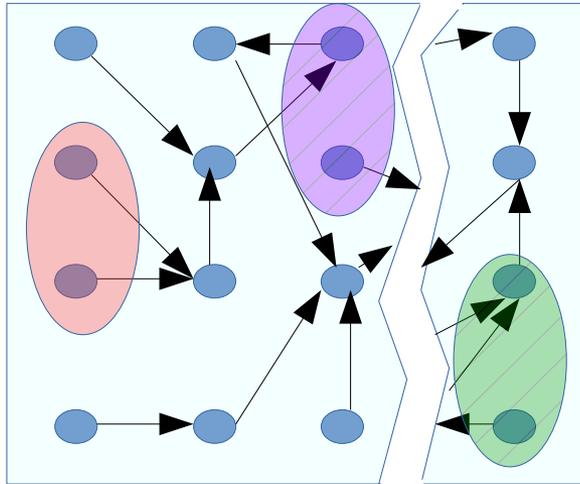


Concretization (γ)



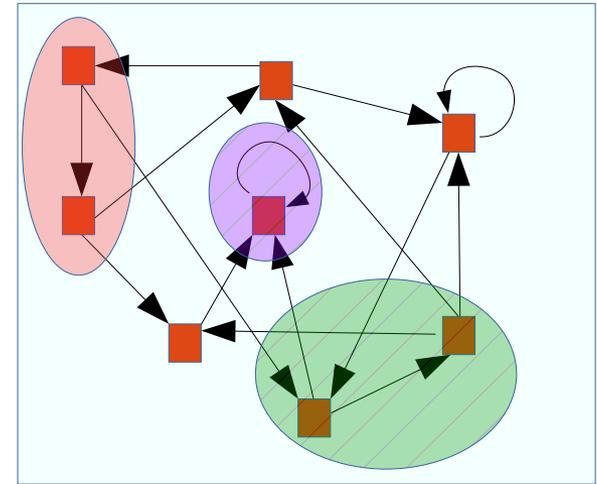
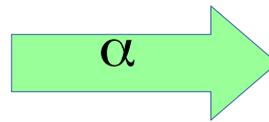
- Every subset of concrete states mapped to unique abstract state
- Desirable to capture containment relations
- Transitions between state sets (abstract states)

The Game Plan



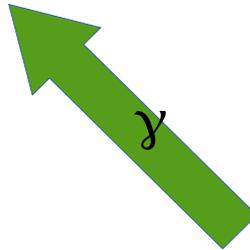
C
O
N
C
R
E
T
E

S
T
A
T
E
S



A
B
S
T
R
A
C
T

S
T
A
T
E
S



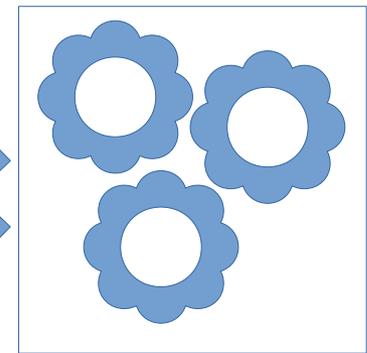
Pre-condition:

NN computation
as a
sequence of
state transitions

Post-condition:

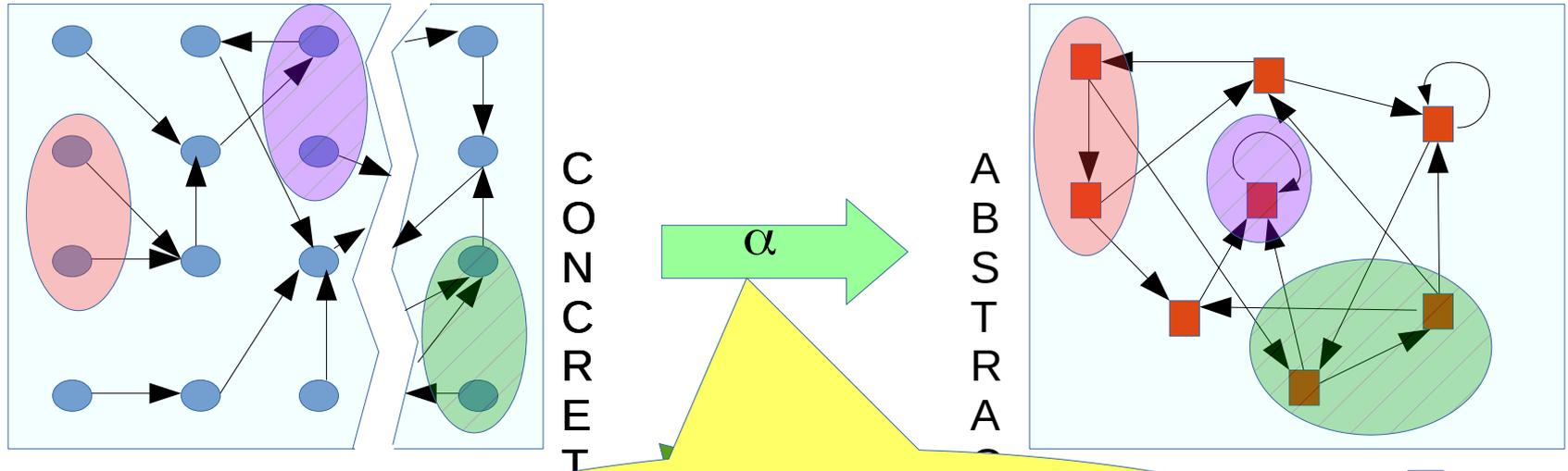
Yes,
Proof

No,
Counterexample



Abstract analysis engine

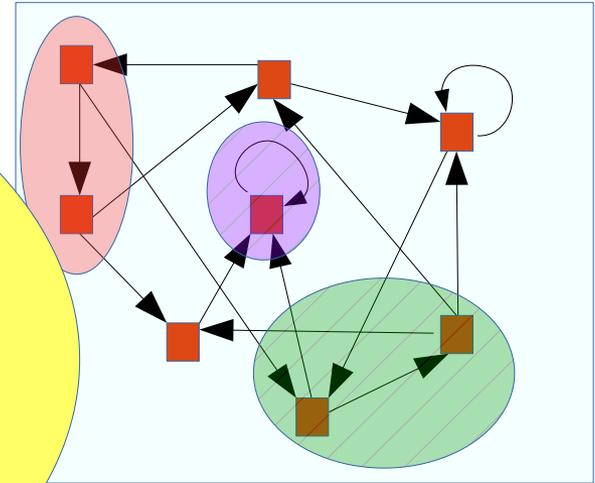
The Game Plan



**How do we choose the right abstraction?
Is there a method beyond domain expertise?
Can we learn from errors in abstraction to build
better (refined) abstractions?
Can refinement be automated?**

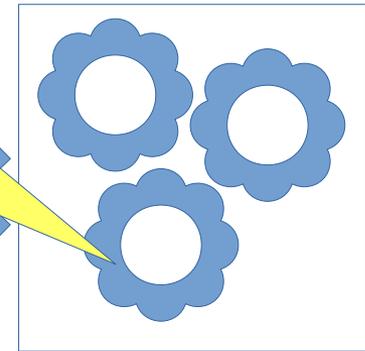
The Game Plan

Abstract state spaces can be infinite.
What can we do to make abstract
analysis practical?
Finite ascending chains
what beyond?



Yes,
Proof

No,
Counterexample



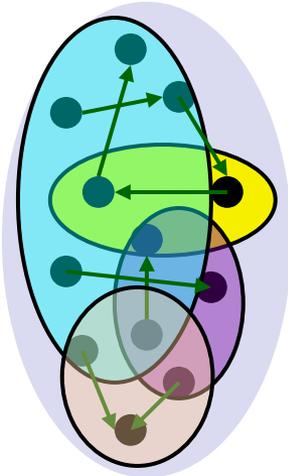
A
T
E
S

A
T
E
S

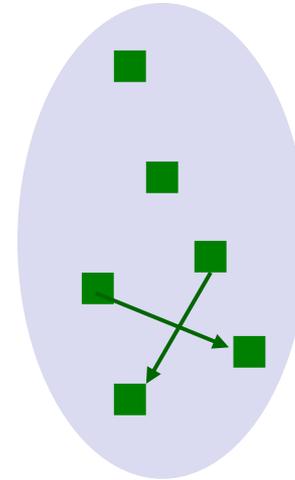
Abstract analysis engine

Desirable Properties of Abstraction

Set of concrete states



Set of abstract states



Abstraction (α)



Concretization (γ)

- Suppose $S_1 \subseteq S_2$: subsets of concrete states
 - Any behaviour starting from a state S_1 can also happen starting from some state in S_2
 - If $S_1 \subseteq S_2$, we want this monotonicity in behaviour in abstract state space too
 - Need ordering of abstract states, similar in spirit to \subseteq

Structure of Concrete State Space

› Set of concrete states: S

• Concrete lattice $\mathbf{C} = (\wp(S), \subseteq, \cup, \cap, S, \emptyset)$

Powerset of S

Partial order

Least upper bound

Greatest lower bound

Bottom element

Top element

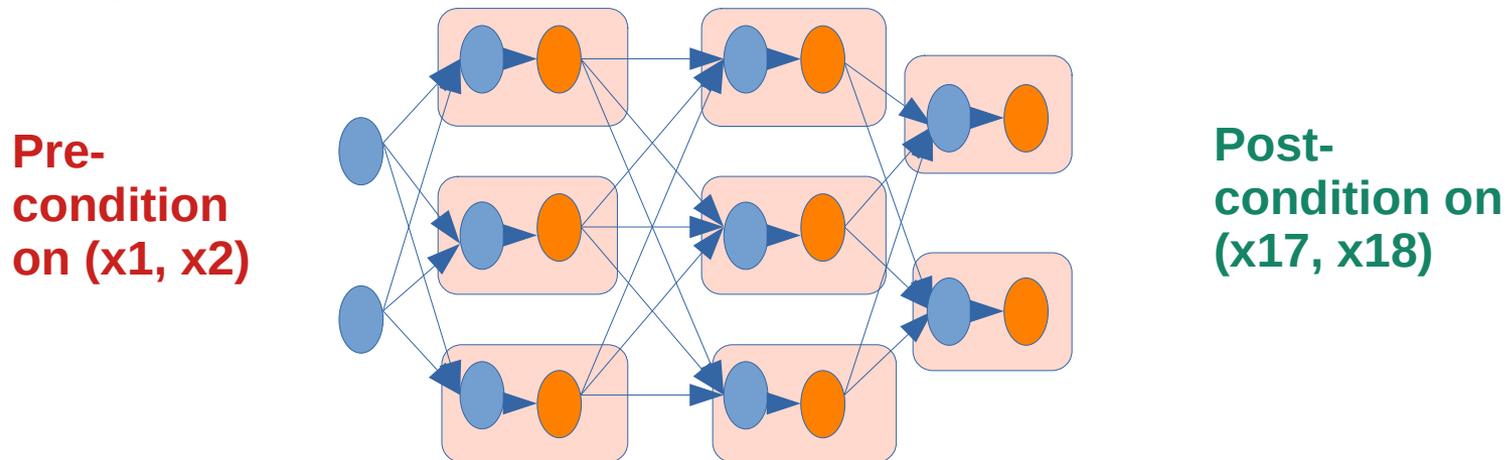
Structure of Abstract State Space

- Abstract lattice $A = (\mathcal{A}, \sqsubseteq, \sqcup, \sqcap, \top, \perp)$
- Abstraction function $\alpha : \wp(S) \rightarrow \mathcal{A}$
 - Monotone: $S_1 \subseteq S_2 \Rightarrow \alpha(S_1) \sqsubseteq \alpha(S_2)$ $S_1, S_2 \subseteq S$
 - $\alpha(S) = \top, \quad \alpha(\emptyset) = \perp$
- Concretization function $\gamma : \mathcal{A} \rightarrow \wp(S)$
 - Monotone: $a_1 \sqsubseteq a_2 \Rightarrow \gamma(a_1) \subseteq \gamma(a_2)$ $a_1, a_2 \in \mathcal{A}$
 - $\gamma(\top) = S, \quad \gamma(\perp) = \emptyset$

A Simple Abstract Domain

Interval Abstract Domain

- Simplest domain for analyzing numerical programs
- Represent values of each variable separately using intervals
- Example:



Represent values of inputs by intervals,
Compute values of hidden layer nodes and outputs as intervals

Interval Abstract Domain

➤ Abstract states: intervals of values of x , (ignore values of y)

$$[-10, 7]: \{ (x, y) \mid -10 \leq x \leq 7 \}$$

- $(-\infty, 20]: \{ (x, y) \mid x \leq 20 \}$

- \sqsubseteq relation: Inclusion of intervals

$$[-10, 7] \sqsubseteq [-20, 9]$$

- \sqcup and \sqcap : union and intersection of intervals

$$[-10, 9] \sqcup [-20, 7] = [-20, 9]$$

$$[-10, 9] \sqcap [-20, 7] = [-10, 7]$$

- \perp is empty interval of x

- \top is $(-\infty, +\infty)$

Interval Abstract Domain

➤ Abstract states: intervals of values of x , (ignore values of y)

$$[-10, 7]: \{ (x, y) \mid -10 \leq x \leq 7 \}$$

- $(-\infty, 20]: \{ (x, y) \mid x \leq 20 \}$

- \sqsubseteq relation: Inclusion of intervals

$$[-10, 7] \sqsubseteq [-20, 9]$$

- \sqcup and \sqcap : union and intersection

$$[-10, 9] \sqcup [-20, 7] = [-20, 9]$$

$$[-10, 9] \sqcap [-20, 7] = [-10, 7]$$

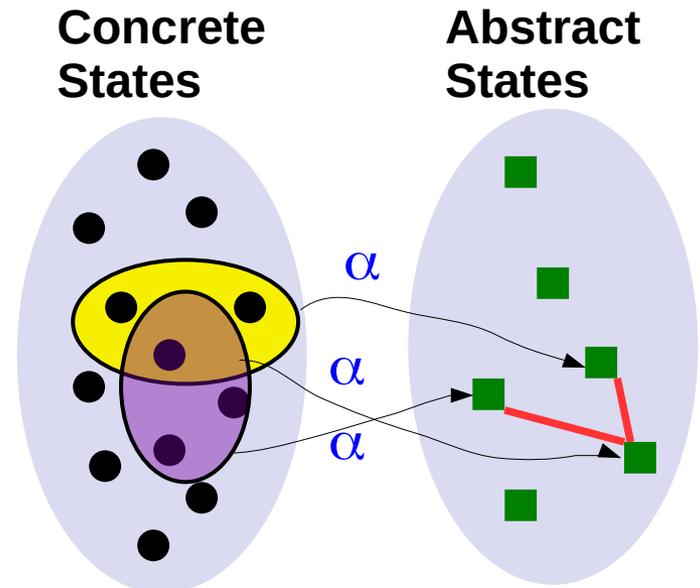
- \perp is empty interval of x

- \top is $(-\infty, +\infty)$

$$\alpha(\{(1, 3), (2, 4), (5, 7)\}) = [1, 5]$$

$$\alpha(\{(5, 7), (7, 6), (9, 10)\}) = [5, 9]$$

$$\alpha(\{(5, 7)\}) = [5, 5]$$



Interval Abstract Domain

- Abstract states: pairs of intervals (one for x , y)
 - $([-10, 7] , (-1, 20])$
 - \sqsubseteq relation: Inclusion of intervals
 - $([-10, 7] , (-1, 20]) \sqsubseteq ([-20, 9] , (-1, +\infty))$
 - \sqcup and \sqcap : union and intersection of intervals
 - $([-10, 9] , (-1, 20]) \sqcap ([-20, 7] , [3, +\infty)) = ([-10, 7] , [3, 20])$
 - $([-10, 9] , (-1, 20]) \sqcup ([-20, 7] , [3, +\infty)) = ([-20, 9] , (-1, +\infty))$
 - \perp is empty interval of x and y
 - \top is $((-\infty, +\infty) , (-\infty, +\infty))$

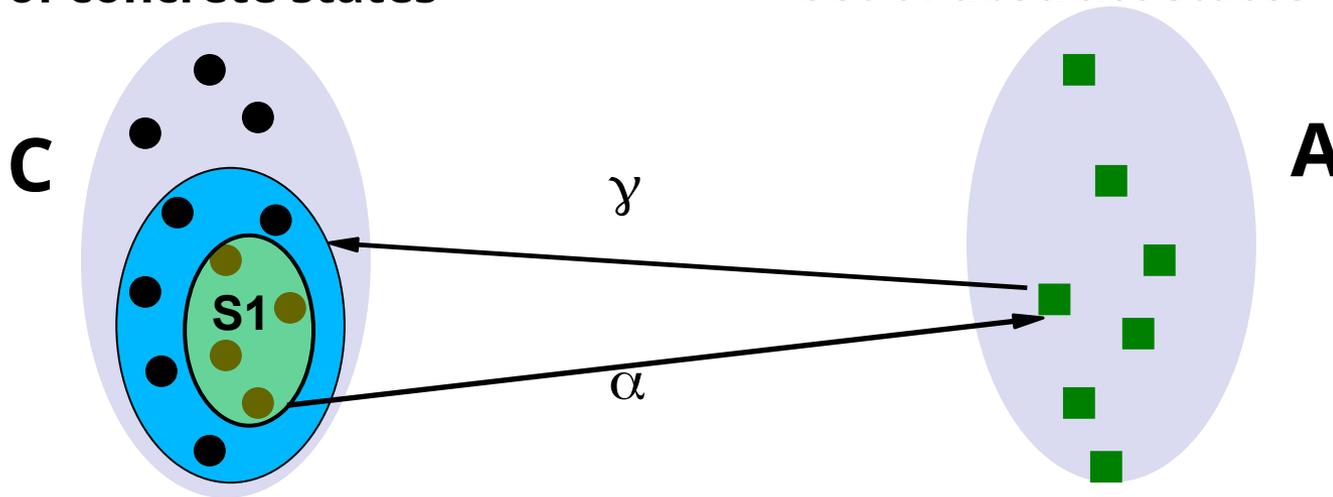
Desirable Properties of α and γ

For all $S_1 \subseteq \mathcal{C}$ $S_1 \subseteq \gamma(\alpha(S_1))$

▪

Set of concrete states

Set of abstract states

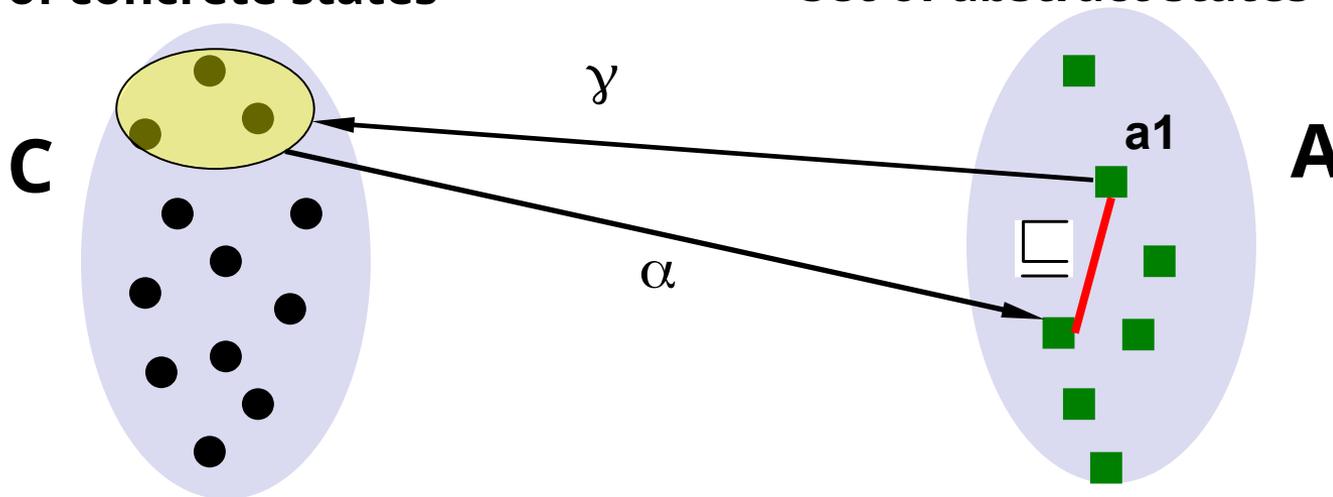


Desirable Properties of α and γ

$$S_1 \subseteq \gamma(\alpha(S_1)) \quad \text{forall } S_1 \subseteq \mathcal{C}$$
$$\alpha(\gamma(a_1)) \sqsubseteq a_1 \quad \text{forall } a_1 \in \mathcal{A}$$

Set of concrete states

Set of abstract states



α and γ form a Galois connection

Desirable Properties of α and γ

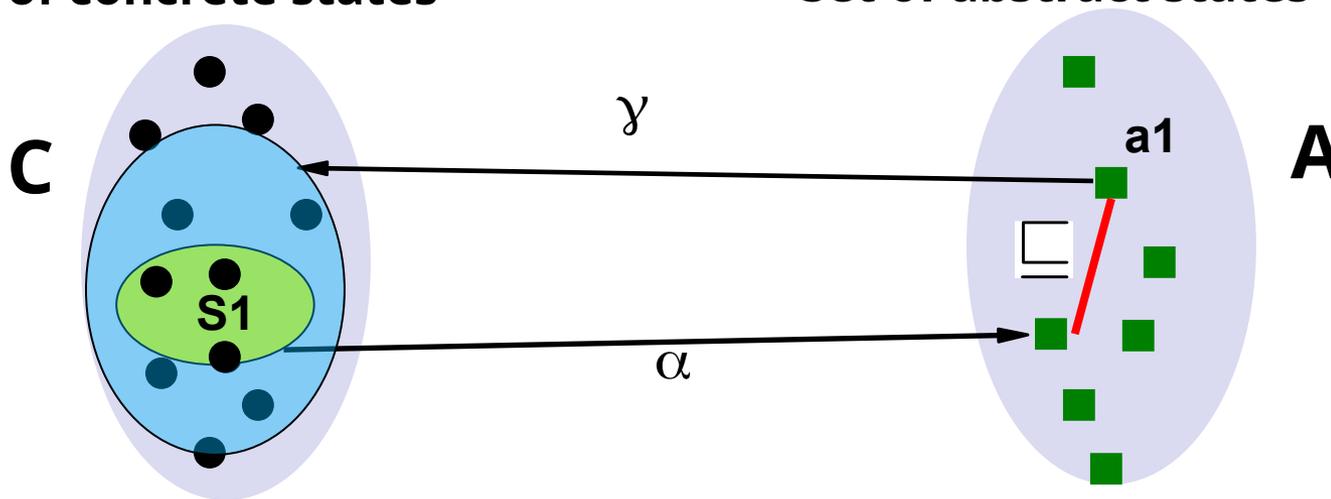
➤ α and γ form a Galois connection

▪ Second (equivalent) view:

$$\alpha(S_1) \sqsubseteq a_1 \Leftrightarrow S_1 \subseteq \gamma(a_1) \text{ for all } S_1 \subseteq S, a_1 \in \mathcal{A}$$

Set of concrete states

Set of abstract states



Homework Problem

Let $\alpha : (S) \rightarrow \mathcal{A}$ and $\gamma : \mathcal{A} \rightarrow (S)$ be monotone maps

Show that

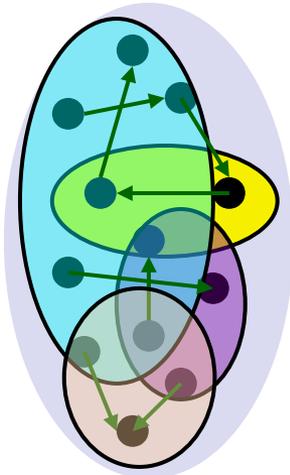
$S_1 \subseteq \gamma(\alpha(S_1))$ for all $S_1 \in (S)$ and $\alpha(\gamma(a_1)) \sqsubseteq a_1$ for all $a_1 \in \mathcal{A}$

holds if and only if

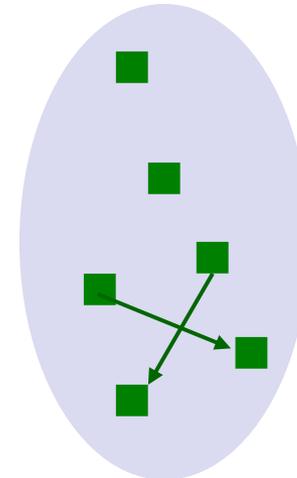
$\alpha(S_1) \sqsubseteq a_1 \Leftrightarrow S_1 \subseteq \gamma(a_1)$ for all $S_1 \in (S)$ and $a_1 \in \mathcal{A}$

Computing Abstract State Transitions

Set of concrete states



Set of abstract states



Abstraction (α)



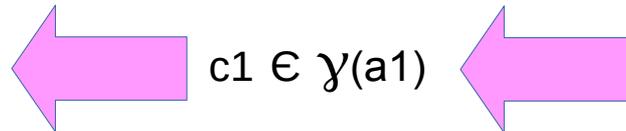
Concretization (γ)

Concrete state c_1



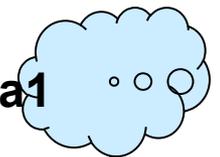
Concrete state c_2

$$(x_1', x_2', x_3') = f(x_1, x_2, x_3)$$

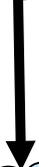


$$c_1 \in \gamma(a_1)$$

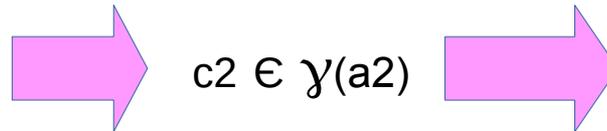
Abstract state a_1



$$(x_1', x_2', x_3') = f(x_1, x_2, x_3)$$



Abstract state a_2



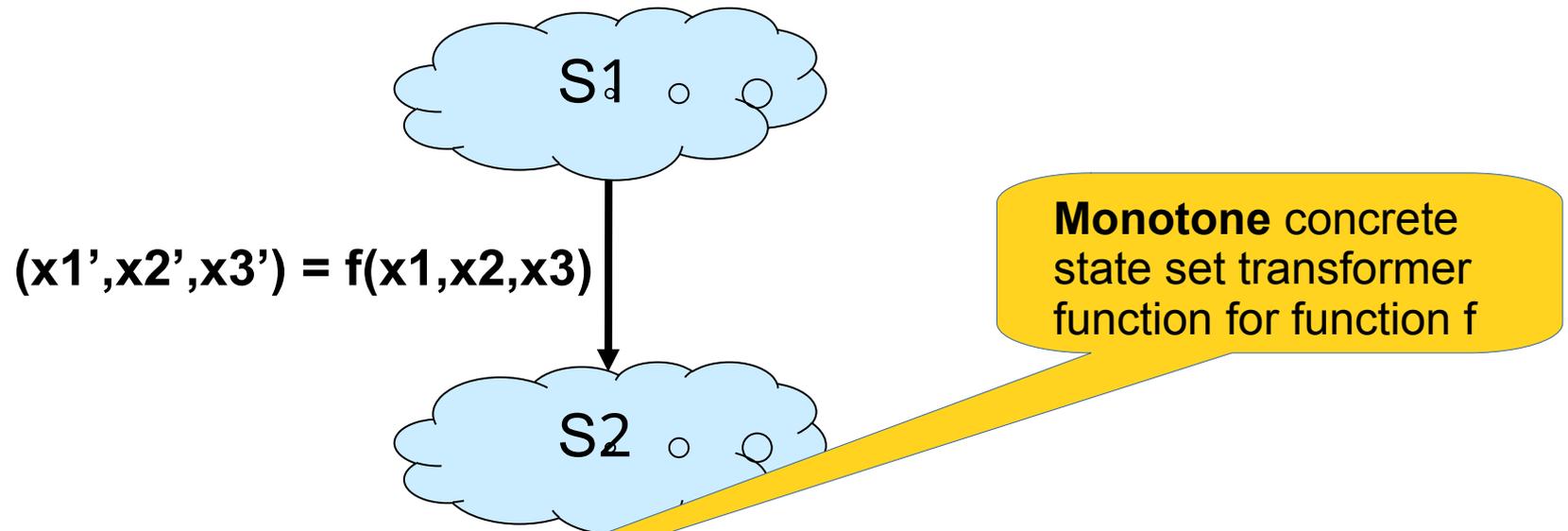
$$c_2 \in \gamma(a_2)$$

Computing Abstract State Transitions

- Concrete state set transformer function

- Example:

$S1 = \{ (x1, x2, x3) \mid \dots \}$: set of concr. states

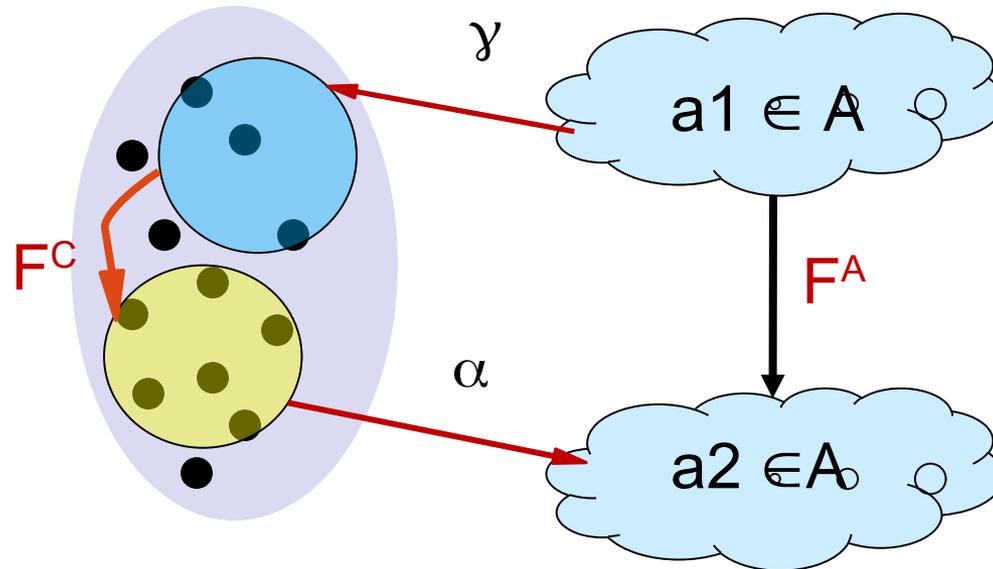


$S2 = \{ (x1', x2', x3') \mid \exists (x1, x2, x3) \in S1, (x1', x2', x3') = f(x1, x2, x3) \}$
 $= F^C(S1)$: set of concrete states

Computing Abstract State Transitions

- Abstract state transformer function
 - Example:

Set of concrete states



$a2 = \alpha(F^C (\gamma (a1)))$ ideally, but $F^A(a1) \sqsupseteq \alpha(F^C (\gamma (a1)))$ often used

Summary

- Abstract interpretation is a general framework for analysis of state transition systems
- Widely used for verification and static analysis of programs
- Recent applications in neural network analysis
- Choice of right abstraction crucial to success
 - Balance between precision and efficiency