Introduction to IC3 & IC3CFA

Tim Lange

Software Modeling and Verification Group, RWTH Aachen

MOVES Seminar, September 23, 2016

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

Outline

Inductive Invariants

Finite State Inductive Strengthening

IC3

IC3 on Control Flow Automata

Inductive Invariants

Inductivity

A property P is inductive if it satisfies initiation and consecution:

$$\text{Initiation: } I \Rightarrow P$$

$$\text{Consecution: } P \wedge T \Rightarrow P'$$

Inductive invariant [MP95]

Given a property P of a system, if P is inductive on S it is an invariant on S.

[MP95] Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995

Inductive Invariants

Inductivity

A property P is inductive if it satisfies initiation and consecution:

$$\text{Initiation: } I \Rightarrow P$$

$$\text{Consecution: } P \wedge T \Rightarrow P'$$

Inductive invariant [MP95]

Given a property P of a system, if P is inductive on S it is an invariant on S.

Not vice versa

Even if P is an invariant on S, it may not be inductive.

[MP95] Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995

Finite State Inductive Strengthening

Proving a property P on a system S

Proving that P holds on S in general is hard. But if P would be inductive, it would be trivial. If we only had a way to make P inductive ...

Inductive Strengthening [BM07]

Try to find a formula F that is an inductive strengthening of P, i.e.

$$\text{Initiation: } I \Rightarrow P \wedge F$$

$$\text{Consecution: } P \wedge F \wedge T \Rightarrow P' \wedge F'$$

[BM07] Aaron R. Bradley and Zohar Manna. "Checking Safety by Inductive Generalization of Counterexamples to Induction". In: *FMCAD*. 2007, pp. 173–180

Finite State Inductive Strengthening

The bad guys

An F-state that has a transition to a $\neg F$-state is called *Counterexample to Induction* (CTI) and is a direct wittness for why F is not inductive.

Finding CTIs

A CTI can be found using a simple satisfiability query

$$sat(F \wedge T \wedge \neg F).$$

If the query is sat, there exists a CTI and we can extract the state from the model of the solver (satisfying variable assignment).

Finite State Inductive Strengthening

Handling CTIs

After finding a CTI s we check whether there exists a *minimal inductive subclause* for $\neg s$. If not, we update P that we have to prove that s is not reachable. Otherwise we can add $\neg s$ to the strengthening F.

Finite State Inductive Strengthening

Handling CTIs

After finding a CTI s we check whether there exists a *minimal inductive subclause* for $\neg s$. If not, we update P that we have to prove that s is not reachable. Otherwise we can add $\neg s$ to the strengthening F.

Happy End?

FSIS terminates if either

- $P \wedge F$ becomes inductive, or
- $I \not\Rightarrow P$ anymore.

IC3

Finding F is hard

In many cases it can be pretty hard to come up with such a strengthening. Especially finding a minimal inductive subclause e that is inductive relative to $P \wedge F$ is hard.

Solution [Bra11]

Instead of computing F directly, compute a sequence $F_0, \dots, F_k$, called frames, to a find an inductive strengthening within these F_i.

[Bra11] Aaron R. Bradley. "SAT-Based Model Checking without Unrolling". In: *VMCAI*. 2011, pp. 70–87

IC3

Frames

For a frame sequence $F_0, \ldots, F_k$ to be an inductive strengthening, the following must hold:

$$I \Rightarrow F_0 \tag{1}$$
$$F_i \Rightarrow F_{i+1} \tag{2}$$
$$F_i \Rightarrow P \tag{3}$$
$$F_i \wedge T \Rightarrow F'_{i+1} \tag{4}$$

Algorithm 1 Outer loop

function bool prove
    Check 0- and 1-step counterexamples
    Initialise frames $F_0 = I, F_1 = P$
    for k = 1 to ... **do**
        Blocking phase
        Propagation phase
        Check termination

IC3

Handling CTIs

Given a CTI c in the last frame F_k, we check whether c is reachable from F_{k-1} in one step.

Thinking inductive:

From an F_{k-1} state that is not c, do we stay in not c after one step? In other words: Is $\neg c$ inductive relative to F_k:

$$F_k \wedge \neg c \wedge T \Rightarrow \neg c$$

How to check validity

Validity of the implication can be solved as $unsat(F_k \wedge \neg c \wedge T \wedge c)$.

IC3

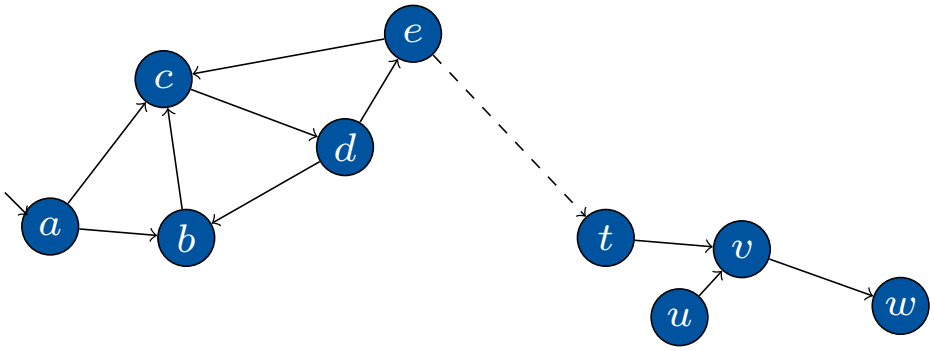
Termination

If for any frame F_i, $0 \leq i < k$ it holds that $F_i = F_{i+1}$ then

$$\begin{aligned} & F_i \wedge T \Rightarrow F_{i+1} \\ \Leftrightarrow\ & F_i \wedge T \Rightarrow F_i \end{aligned}$$

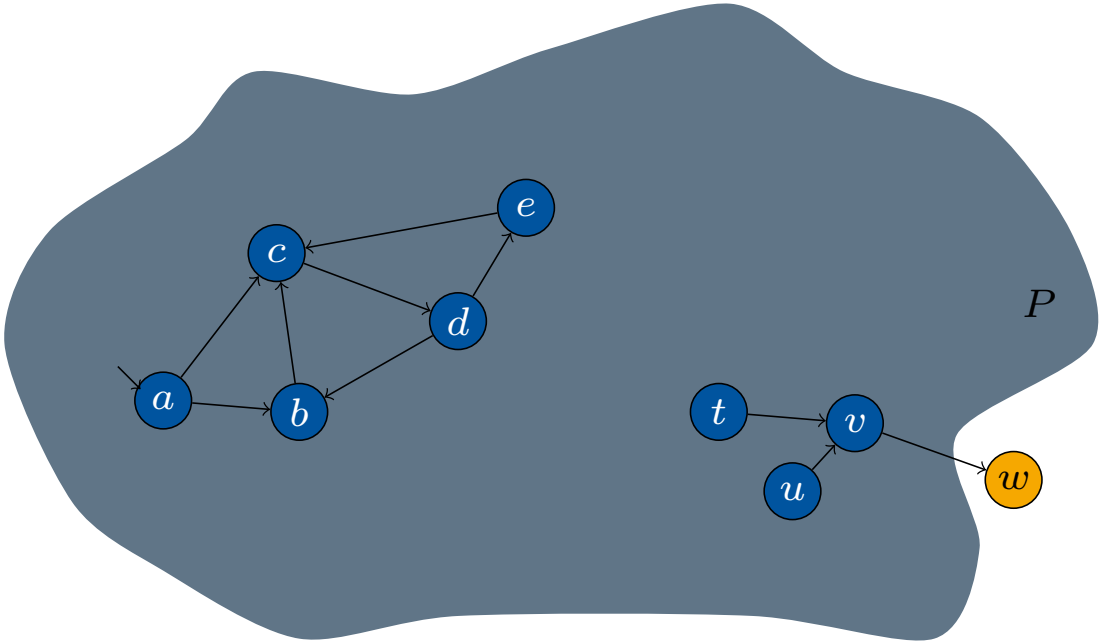
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$



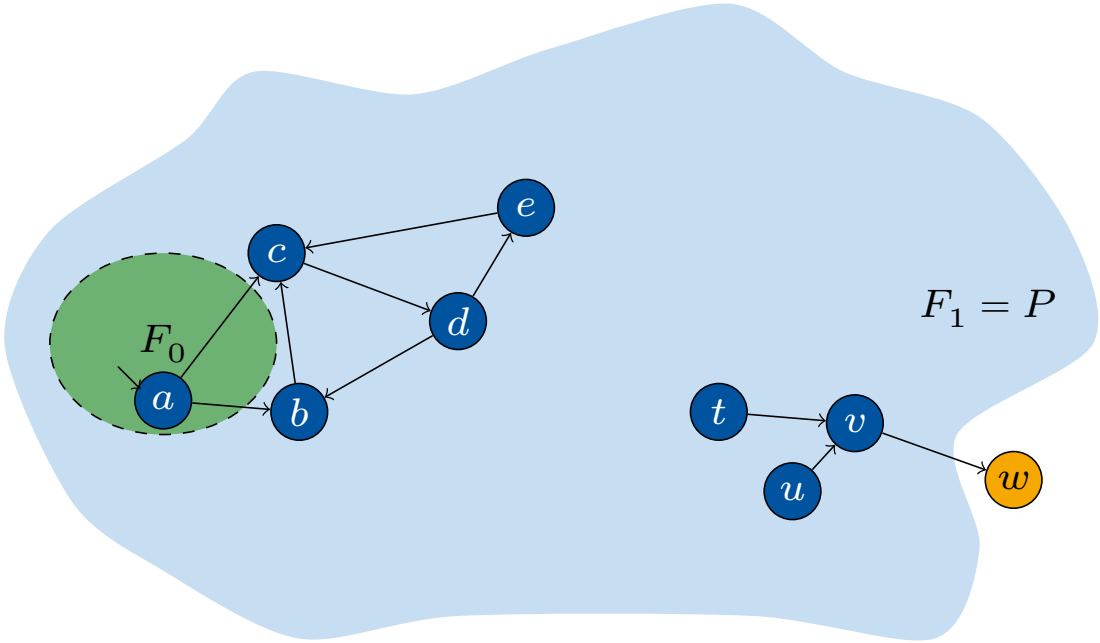
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



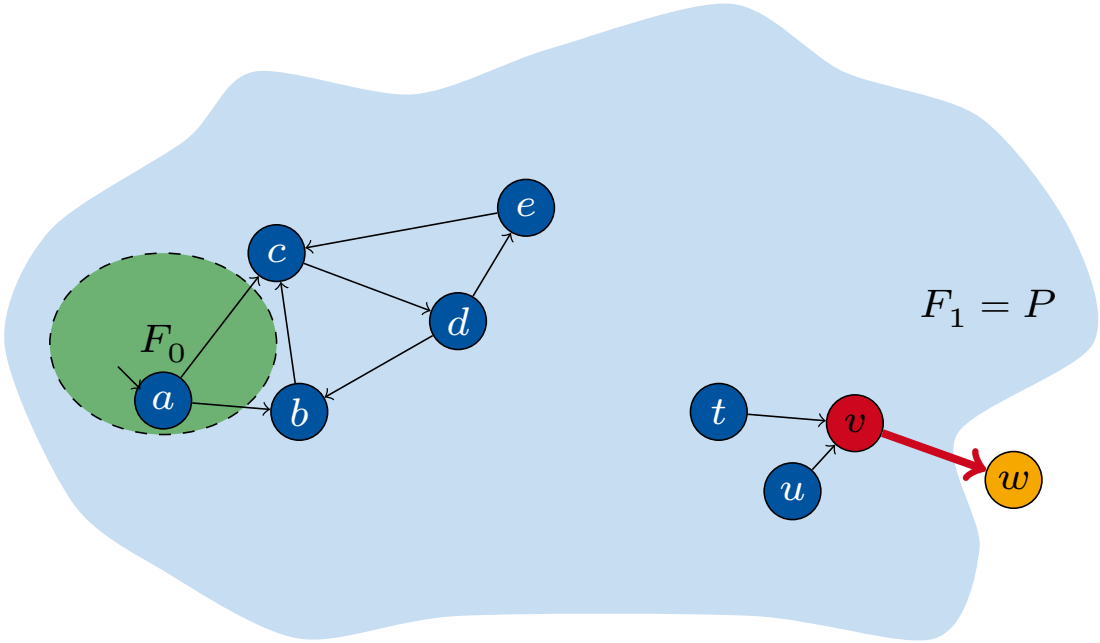
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



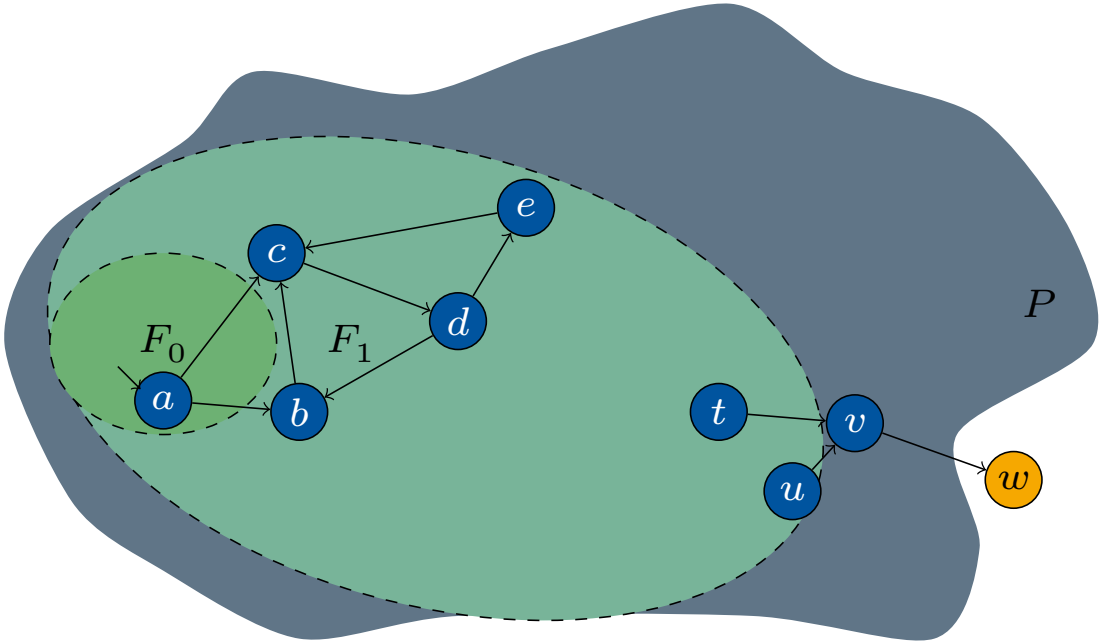
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



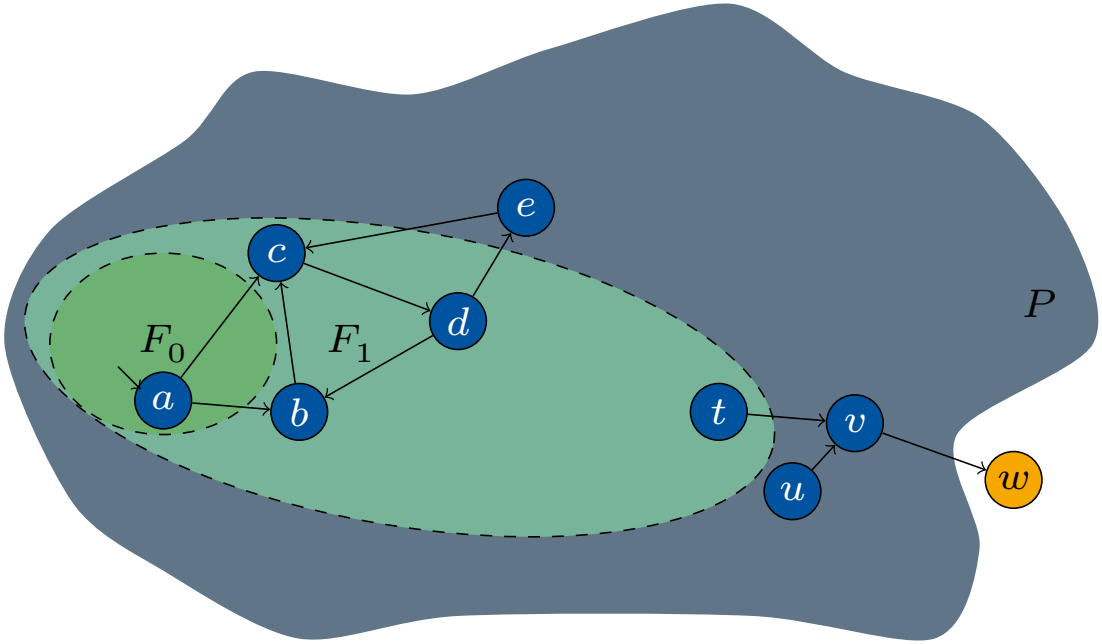
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



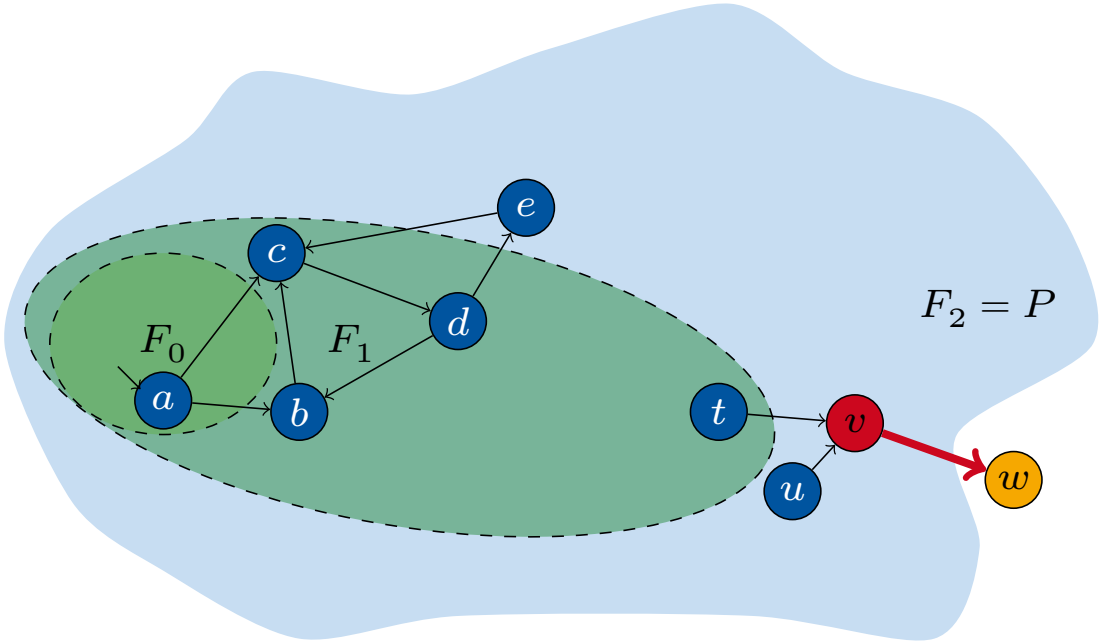
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



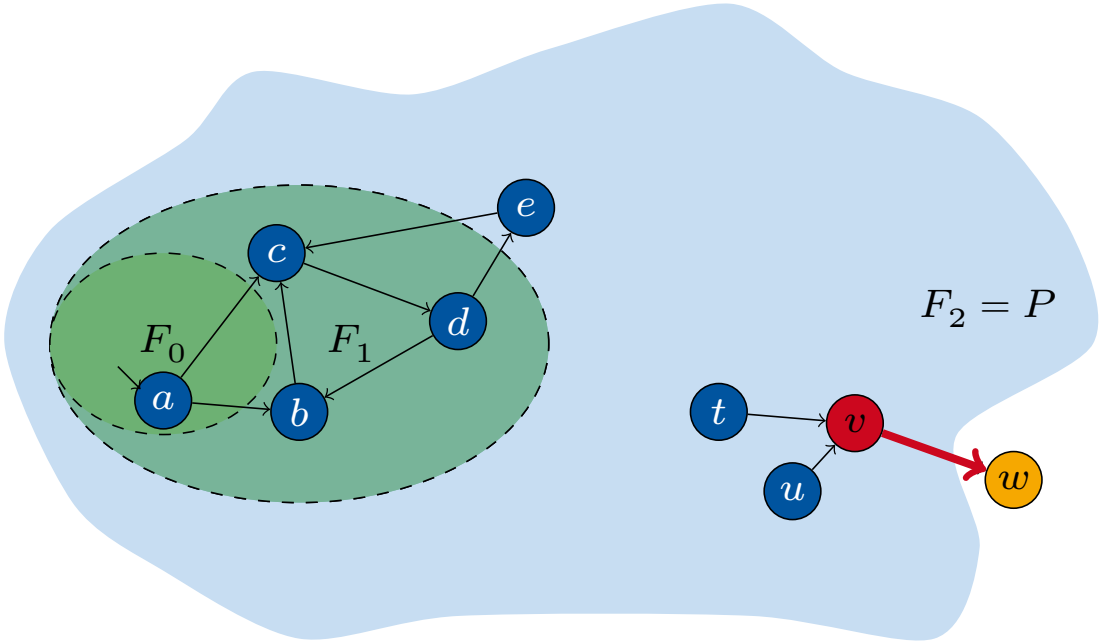
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



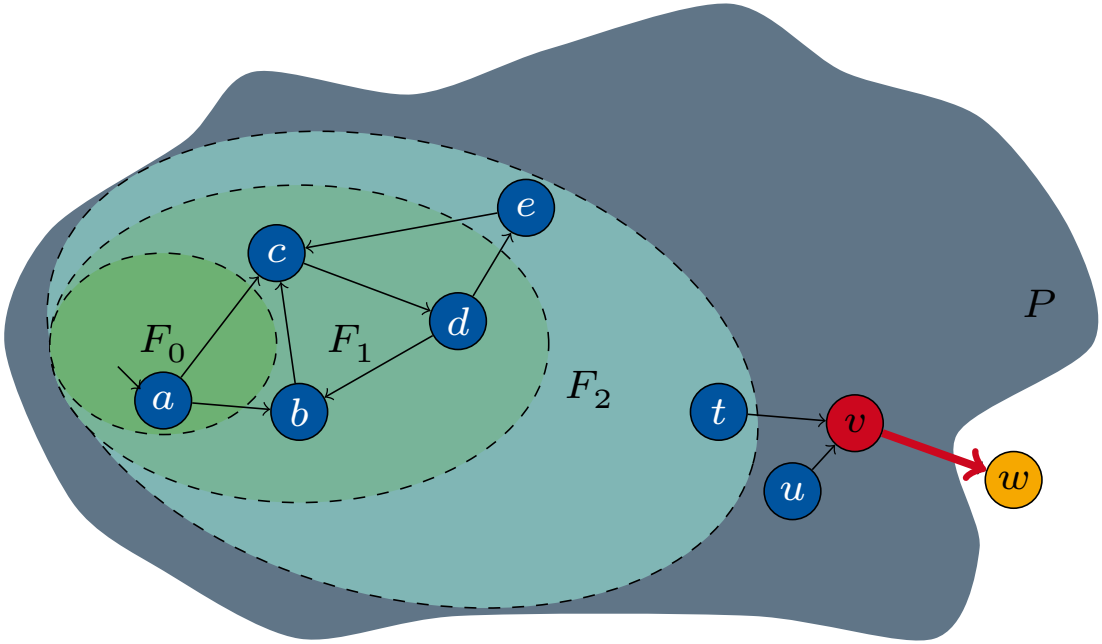
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



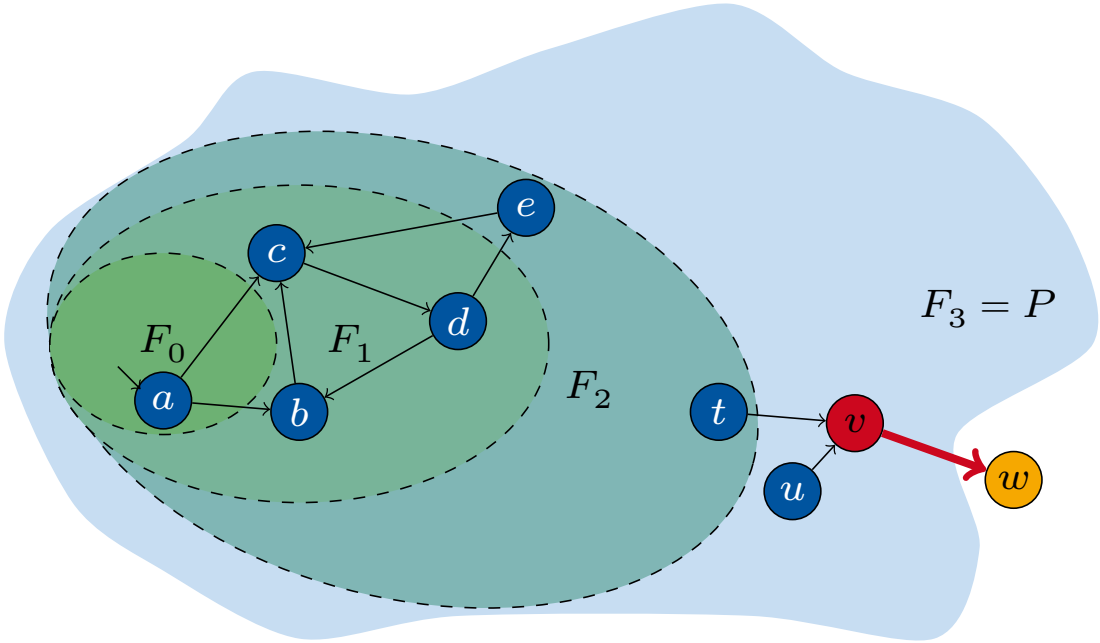
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



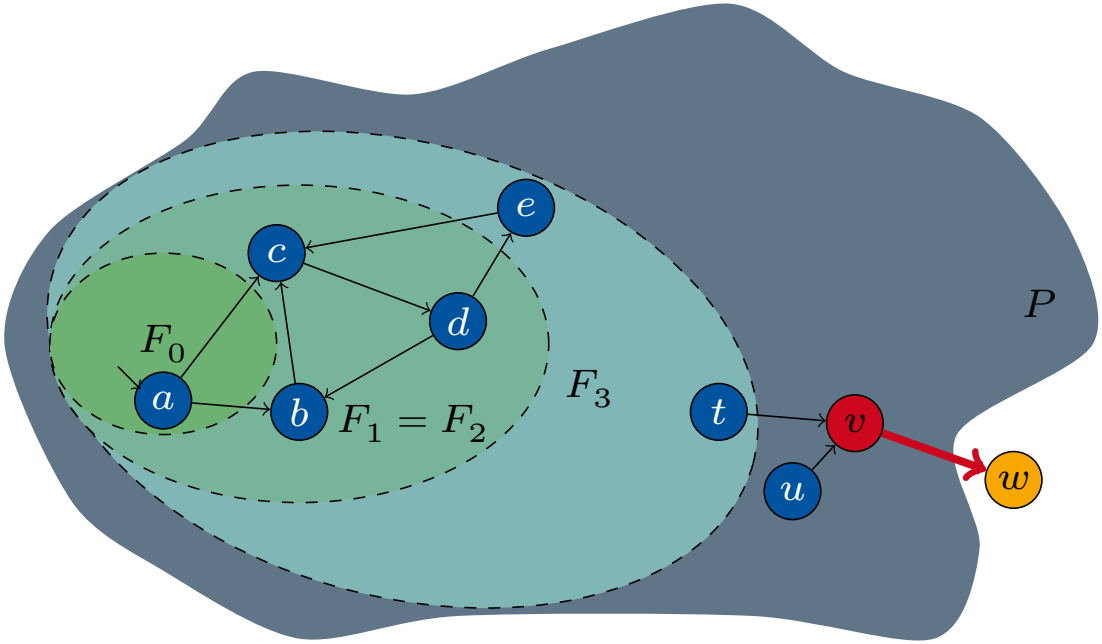
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



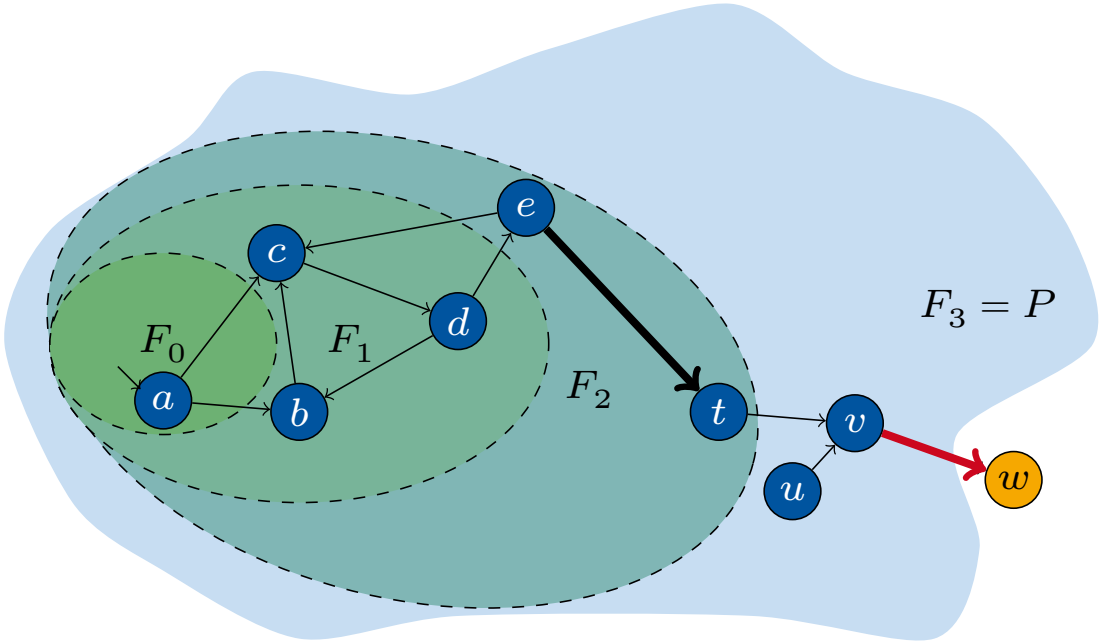
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



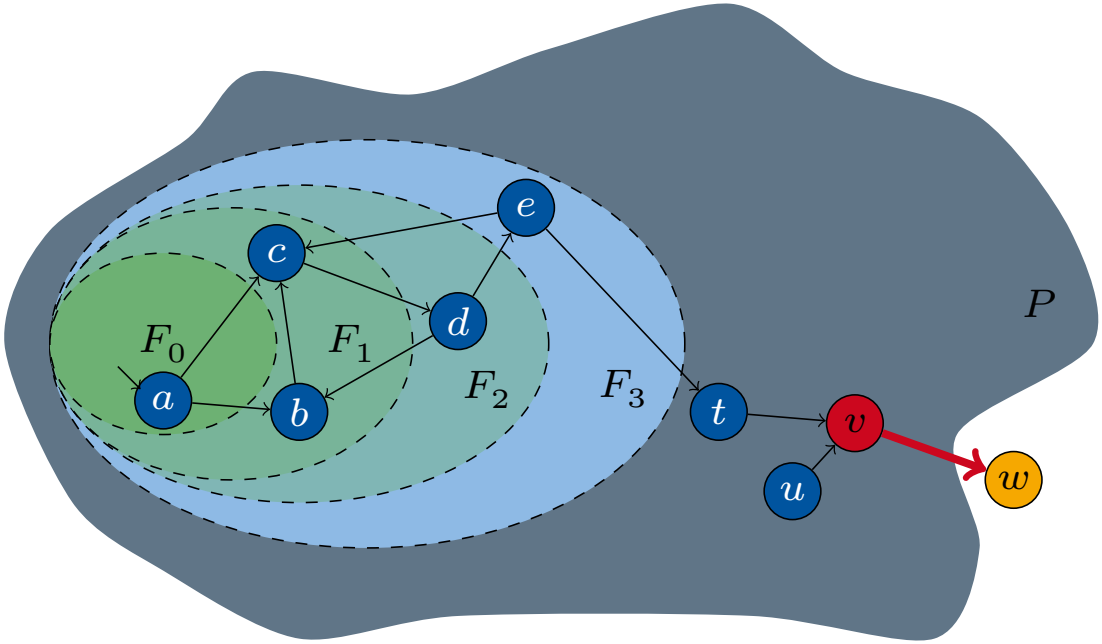
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



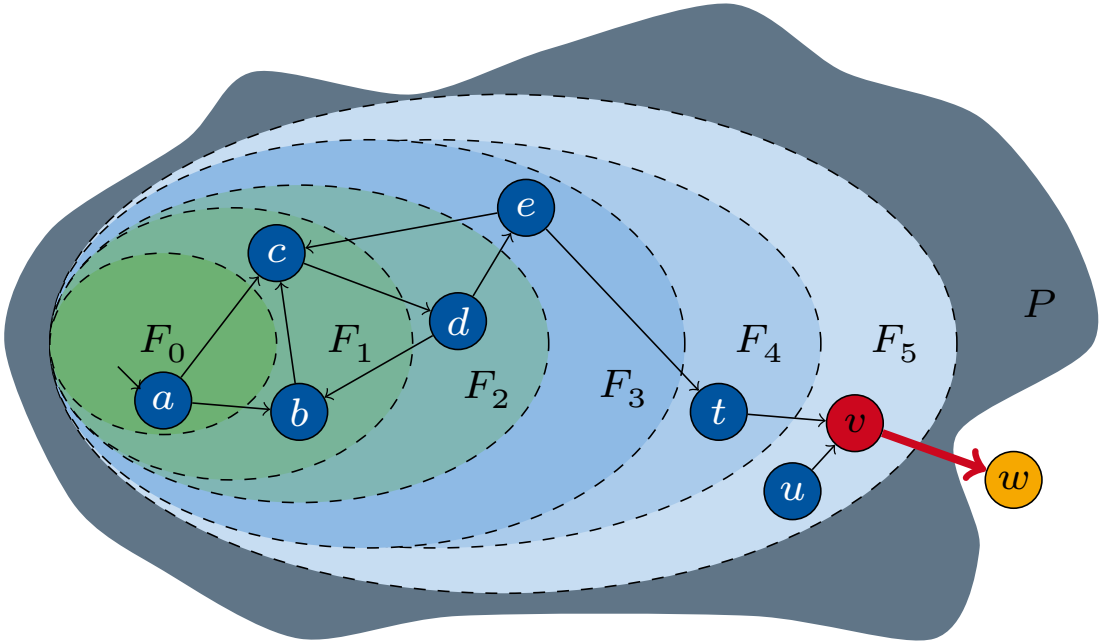
IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



IC3

Consider the transition system $\mathcal{M} = (X, I, T)$ and the property $P(X)$.



IC3 on Control Flow Automata

Motivation

Lifting to software model checking

- IC3 had a deep impact in hardware model checking
- Showed much better performance than known techniques
- Nowadays employed in most major hardware model checking tools

Challenges

- Domain in hardware model checking finite (bit-level)
- How to handle infinite state spaces?
- How to encode finite control flow?

IC3 on Control Flow Automata

Control Flow Automaton (CFA)

A CFA $\mathcal{A} = (L, G, l_0, l_E)$ consists of a set of *locations* $L = \{0, \dots, n\}$ and edges in $G \subseteq L \times QFFO \times L$ labeled with *quantifier-free* first-order formulas, an *initial location* l_0, and an *error location* l_E.

[1] CG12.

IC3 on Control Flow Automata

Control Flow Automaton (CFA)

A CFA $\mathcal{A} = (L, G, l_0, l_E)$ consists of a set of *locations* $L = \{0, \dots, n\}$ and edges in $G \subseteq L \times QFFO \times L$ labeled with *quantifier-free* first-order formulas, an *initial location* l_0, and an *error location* l_E.

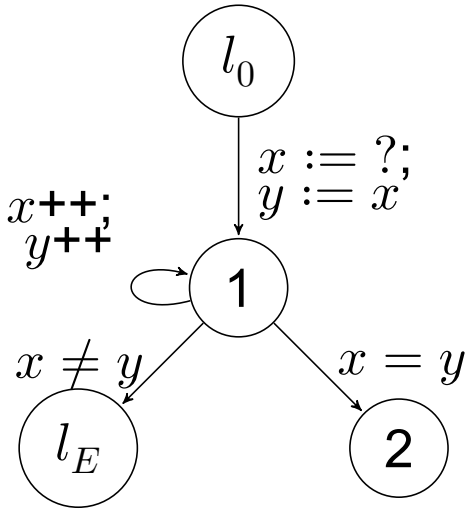
Idea

- Encoding of control flow using special pc variable *not efficient*[1]
- *Extraction* of control flow *advantageous*
- Instead of unrolling into ART [CG12] apply IC3 *directly on CFA*
- For *every location* in the CFA construct frames $F_0, \dots, F_k$
- Frames represent *overapproximations* of i-step reachability *in location*
- Explicit control flow locations allow to take *only single transitions* into account

[1] CG12.

IC3 on Control Flow Automata

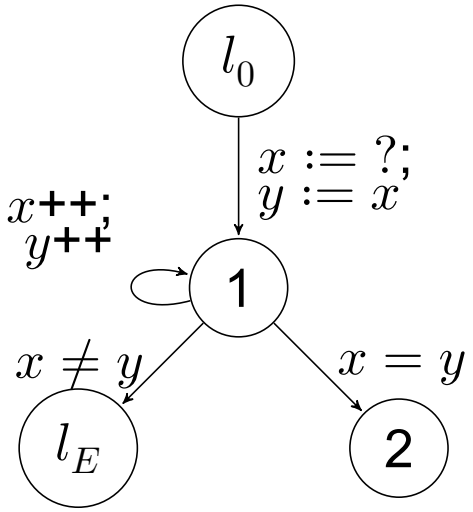
Example



Initial location: l_0
Error location: l_E
Terminating location: 2

IC3 on Control Flow Automata

Example

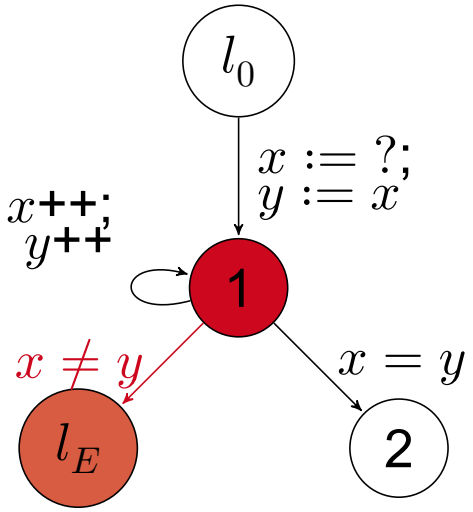


Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	true

IC3 on Control Flow Automata

Example



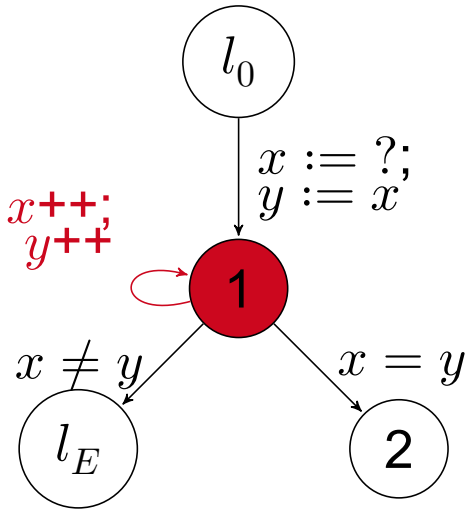
Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	true

CTI (1,$x \neq y$), level 1

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

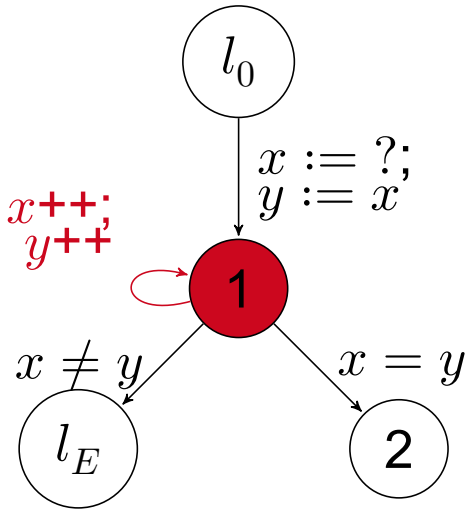
i: \ l:	l_0	1
0	true	false
1	true	true

CTI (1,$x \neq y$), level 1

$$SAT(F_{(0,1)} \wedge \neg(x \neq y) \wedge T_{1\to 1} \wedge x' \neq y')$$

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

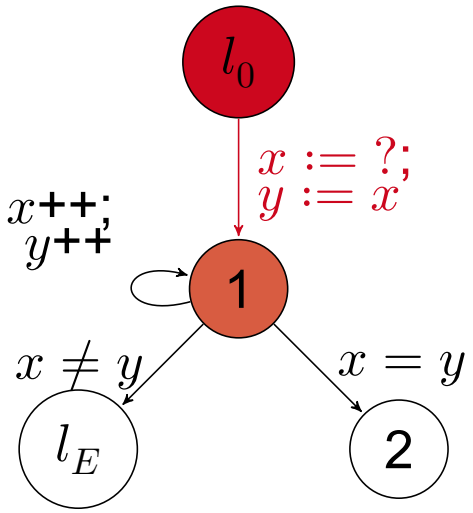
i: \ l:	l_0	1
0	true	false
1	true	true

CTI (1,$x \neq y$), level 1

$SAT(F_{(0,1)} \wedge \neg(x \neq y) \wedge T_{1\to1} \wedge x' \neq y')$ ✗

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	true

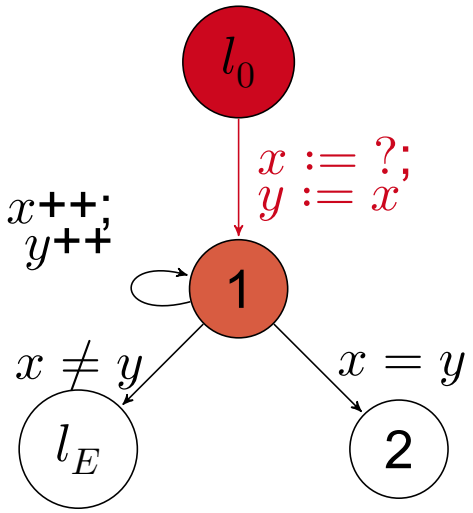
CTI (1,$x \neq y$), level 1

$SAT(F_{(0,1)} \wedge \neg(x \neq y) \wedge T_{1\to 1} \wedge x' \neq y')$ ✗

$SAT(F_{(0,l_0)} \wedge T_{l_0\to 1} \wedge x' \neq y')$

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	true

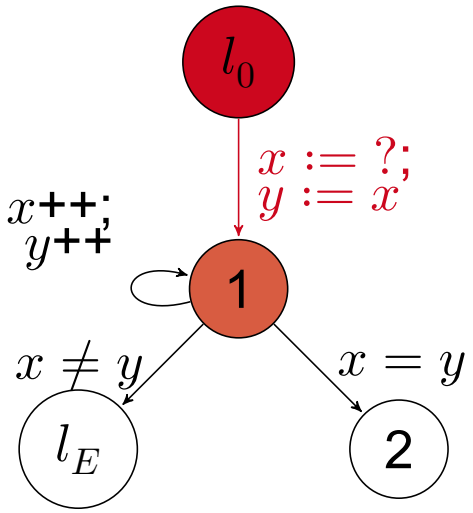
CTI (1,$x \neq y$), level 1

$SAT(F_{(0,1)} \wedge \neg(x \neq y) \wedge T_{1\to 1} \wedge x' \neq y')$ ✗

$SAT(F_{(0,l_0)} \wedge T_{l_0\to 1} \wedge x' \neq y')$ ✗

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	$x = y$

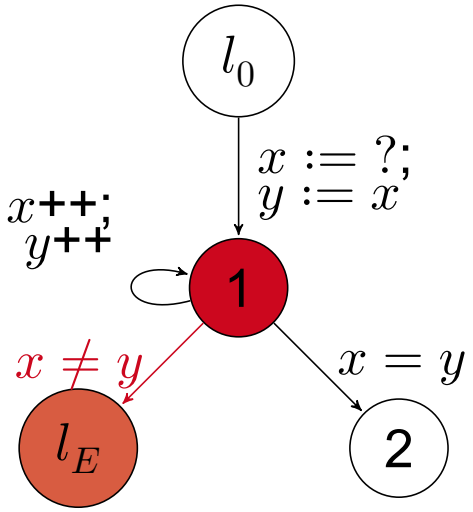
CTI (1,$x \neq y$), level 1

$SAT(F_{(0,1)} \wedge \neg(x \neq y) \wedge T_{1\to 1} \wedge x' \neq y')$ ✗

$SAT(F_{(0,l_0)} \wedge T_{l_0\to 1} \wedge x' \neq y')$ ✗

IC3 on Control Flow Automata

Example

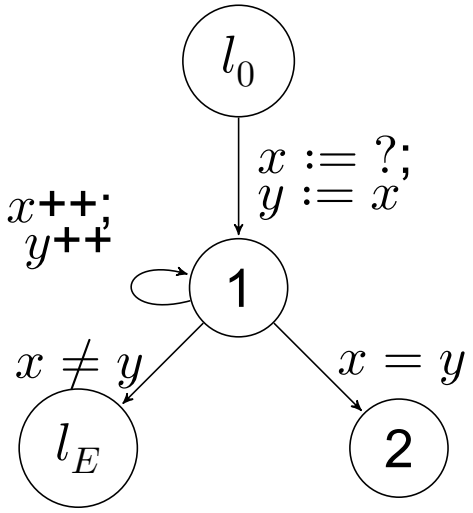


Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	$x = y$

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

i: \ l:	l_0	1
0	true	false
1	true	$x = y$
2	true	$x = y$

References

- Aaron R. Bradley and Zohar Manna. “Checking Safety by Inductive Generalization of Counterexamples to Induction”. In: *FMCAD*. 2007, pp. 173–180.
- Aaron R. Bradley. “SAT-Based Model Checking without Unrolling”. In: *VMCAI*. 2011, pp. 70–87.
- Alessandro Cimatti and Alberto Griggio. “Software Model Checking via IC3”. In: *CAV*. 2012, pp. 277–293.
- Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995.