IC3 Software Model Checking on Control Flow Automata

Tim Lange[1] Martin R. Neuhäußer[2] Thomas Noll[1]

[1] Software Modeling and Verification Group, RWTH Aachen
[2] Siemens AG

FMCAD 2015 at Austin, TX, USA, September 29, 2015

Software Modeling and Verification Chair
RWTH AACHEN UNIVERSITY

Outline

Introduction

Motivation

Lifting to software model checking

- IC3 had a deep impact in hardware model checking
- Showed much better performance than CEGAR and BMC
- 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?

Outline

Introduction

Preliminaries

Original IC3

Related Work

IC3 on Control Flow Automata

Conclusion

Preliminaries

Control Flow Automaton (CFA)

A CFA $\mathcal{A} = (L, G, l_0, l_E)$ consists of a set of locations $L = \{0, \ldots, n\}$ 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.

Transition formula

Given two locations $l_1, l_2 \in L$, we define the transition formula

$$T_{l_1 \to l_2} = \begin{cases} (pc = l_1) \wedge t \wedge (pc' = l_2) & \text{, if } (l_1, t, l_2) \in G \\ false & \text{, otherwise.} \end{cases}$$

Preliminaries

Relative Inductivity [Bra11]

Given a transition formula $T = \bigvee_{(l_1,t,l_2)\in G} T_{l_1 \to l_2}$, a formula φ is inductive relative to another formula ψ if

$$\psi \wedge \varphi \wedge T \Rightarrow \varphi'$$

is valid.

Edge-Relative Inductivity

Given a CFA A and locations $l_1, l_2 \in L$, a formula φ is inductive edge-relative to another formula ψ if

$$\psi \wedge \varphi \wedge T_{l_1 \to l_2} \Rightarrow \varphi'$$

is valid.

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

Preliminaries

Region [Hen+02]

A region $r = (l, s)$ is a pair consisting of location l and formula s. The set of corresponding formulas for r is given as $\{\varphi \mid \varphi \equiv (pc = l \wedge s)\}$. Similarly, for $\neg r$ corresponding formulas are defined as $\{\varphi \mid \varphi \equiv \neg(pc = l \wedge s)\}$.

Edge-Relative Inductive Regions

Assume two regions $r_1 = (l_1, s_1)$, $\neg r_2 = \neg(l_2, s_2)$, we can reduce edge-relative inductivity of $\neg r_2$ to r_1 to

$$s_1 \wedge T_{l_1 \rightarrow l_2} \Rightarrow \neg s_2' \qquad \text{, if } l_1 \neq l_2$$

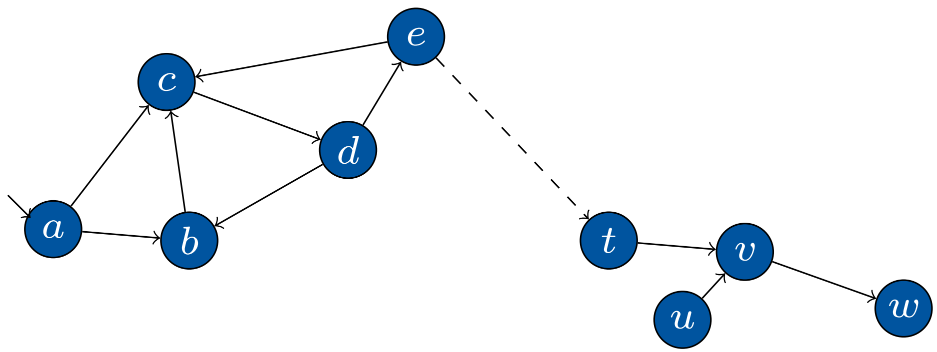
$$s_1 \wedge \neg s_2 \wedge T_{l_1 \rightarrow l_2} \Rightarrow \neg s_2' \qquad \text{, if } l_1 = l_2$$

[Hen+02] Thomas A. Henzinger et al. "Lazy abstraction". In: POPL. 2002, pp. 58–70

Outline

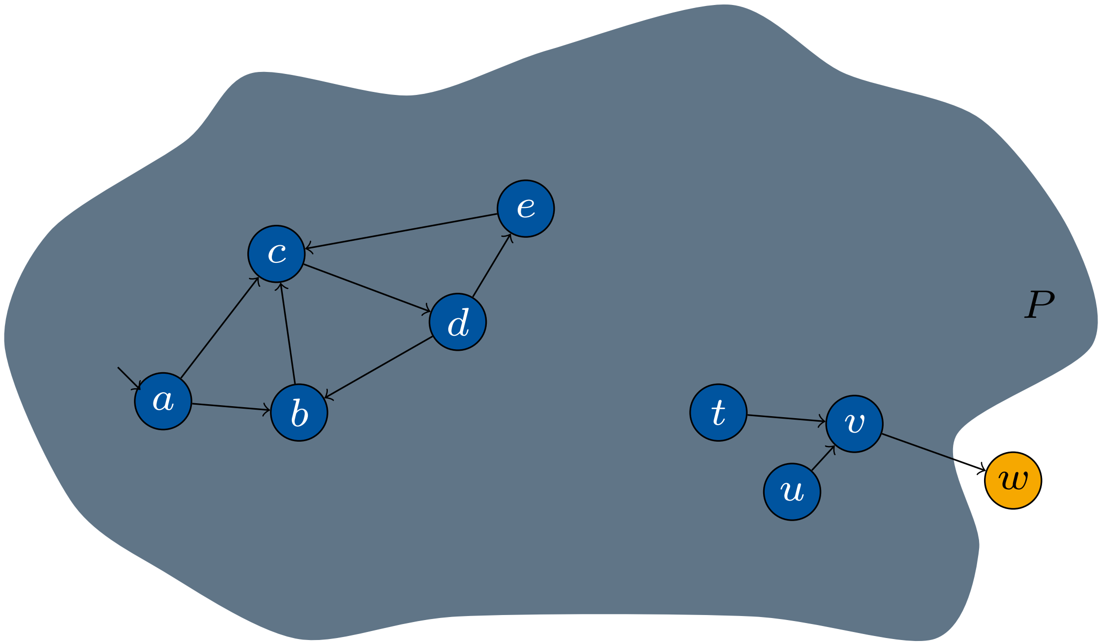
Original IC3

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



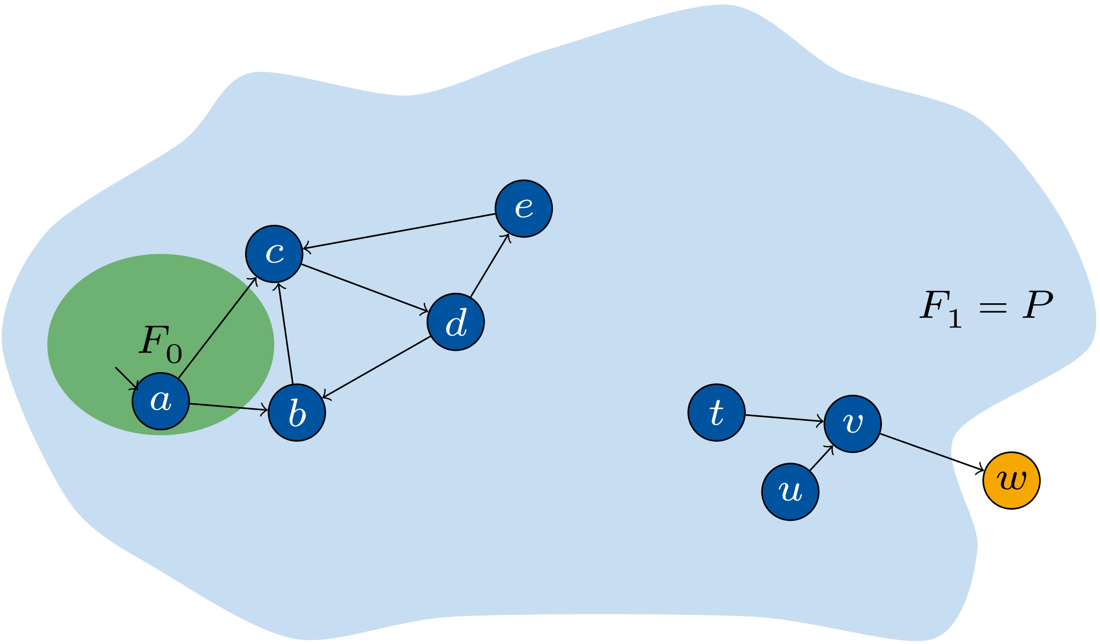
Original IC3

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



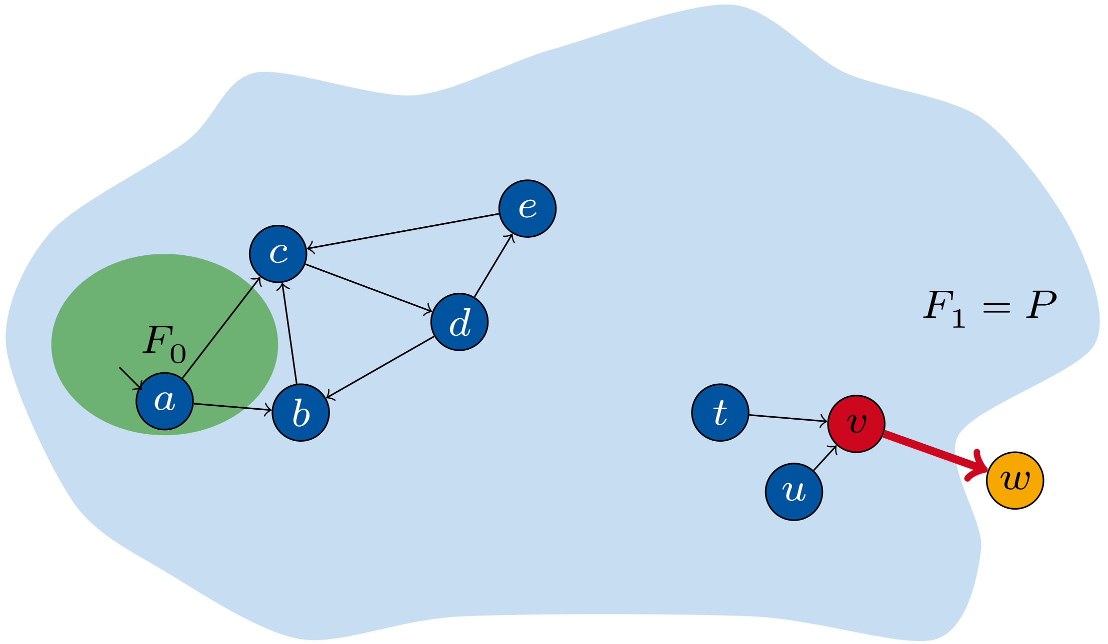
Original IC3

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



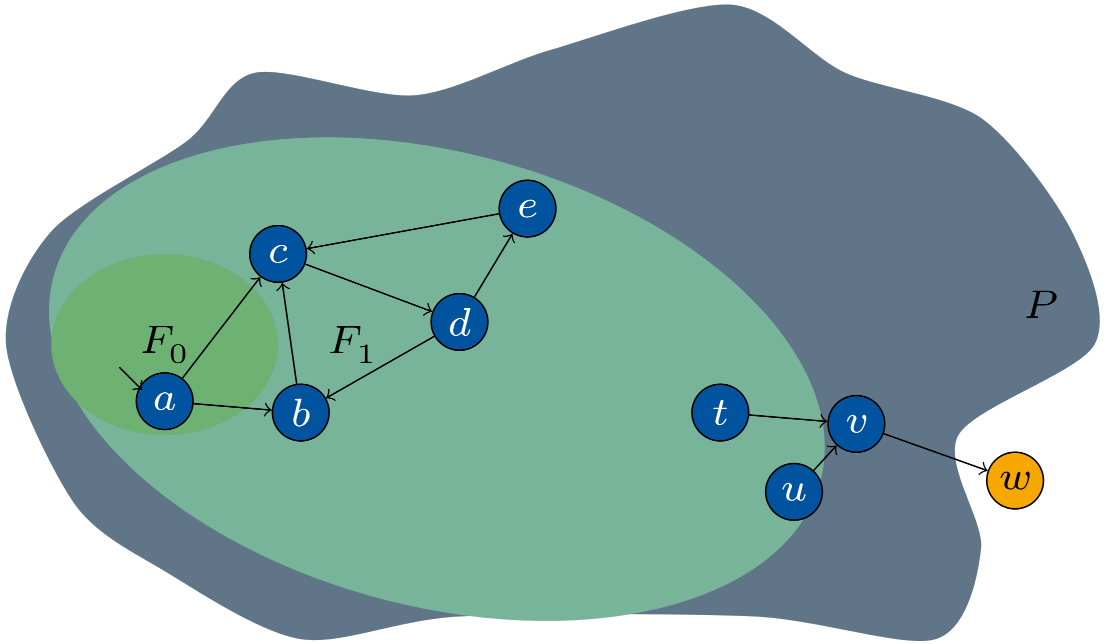
Original IC3

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



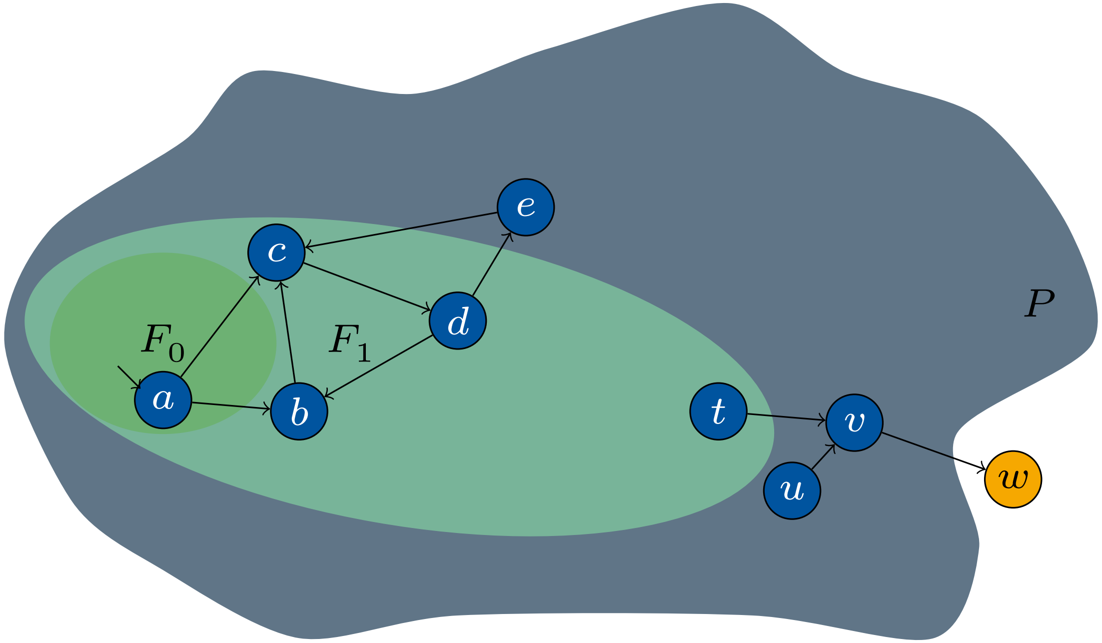
Original IC3

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



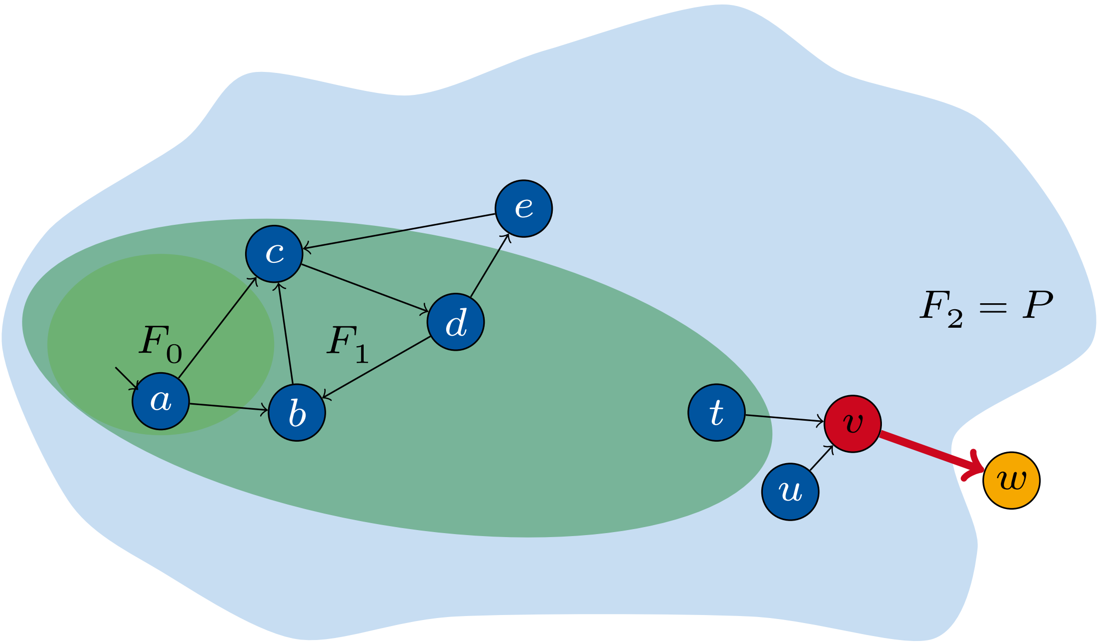
Original IC3

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



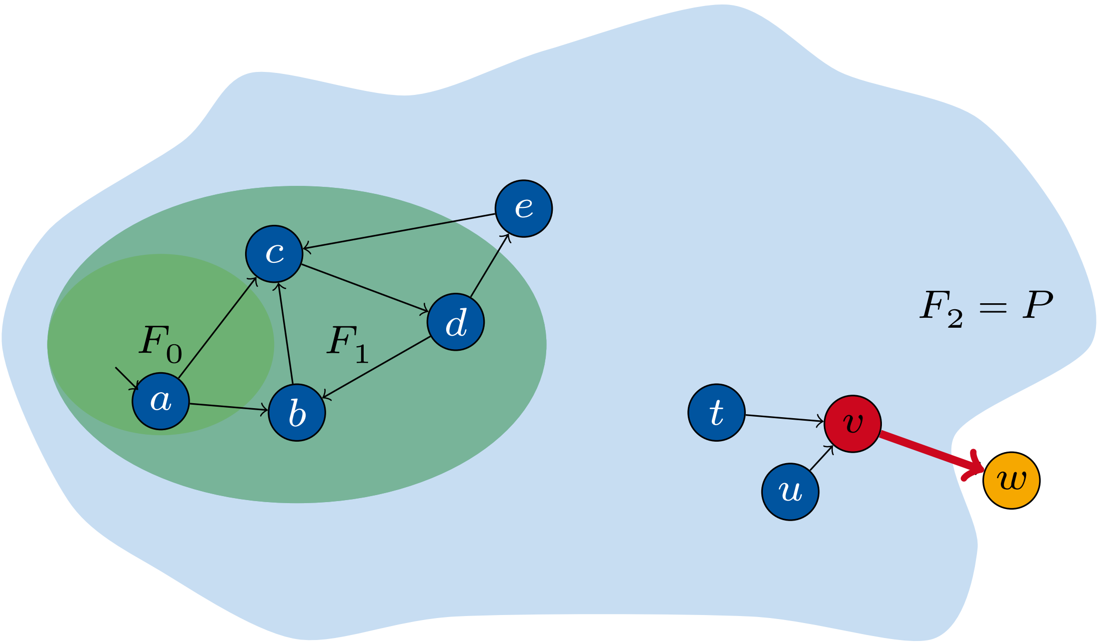
Original IC3

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



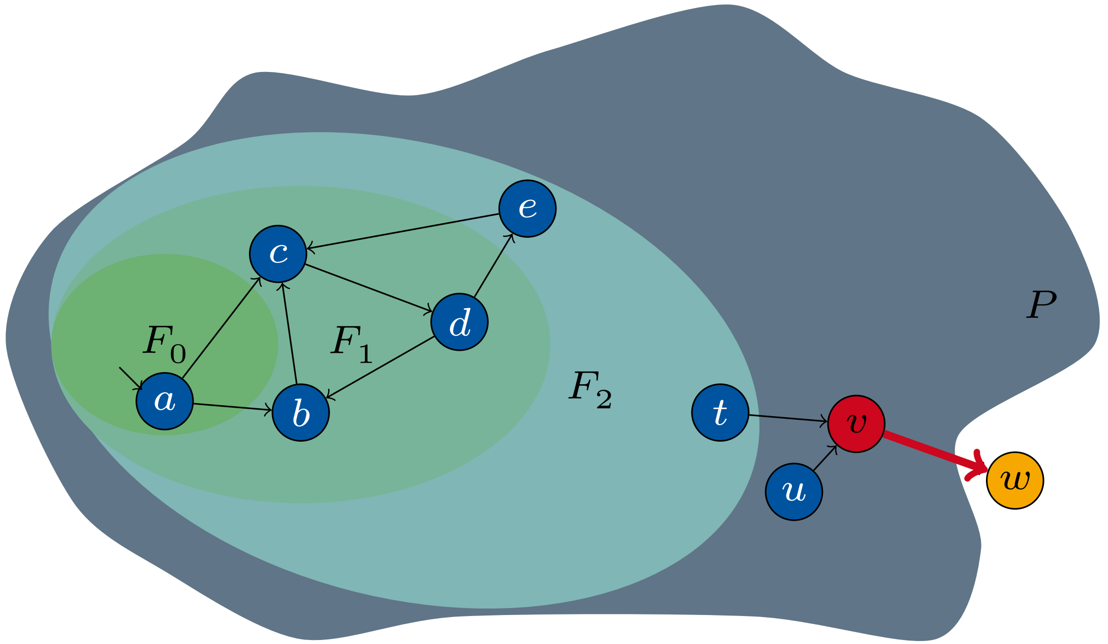
Original IC3

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



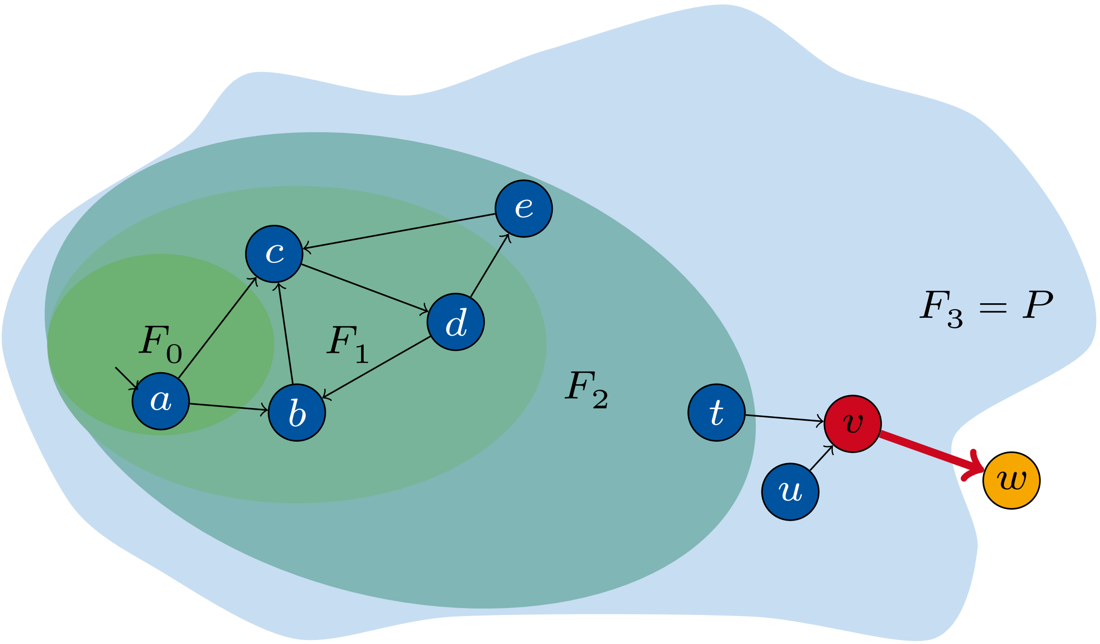
Original IC3

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



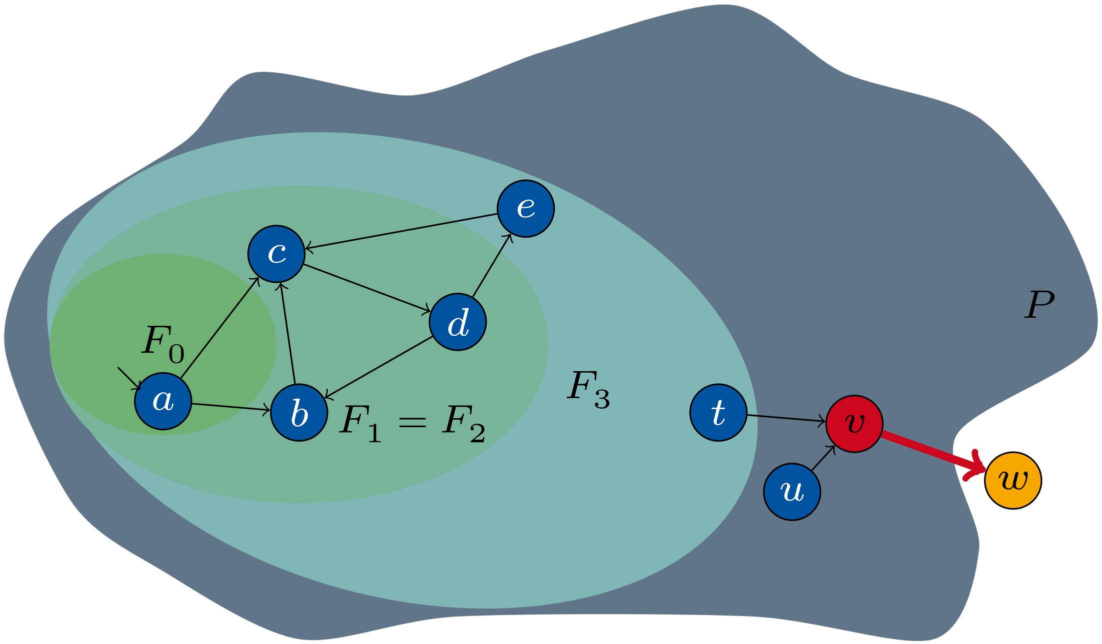
Original IC3

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



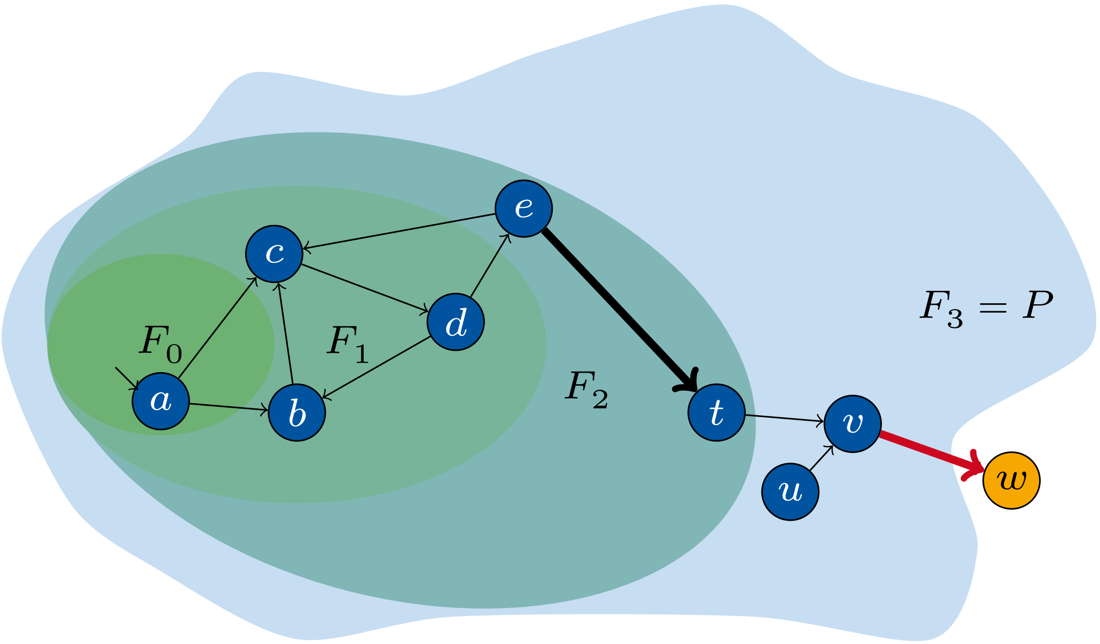
Original IC3

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



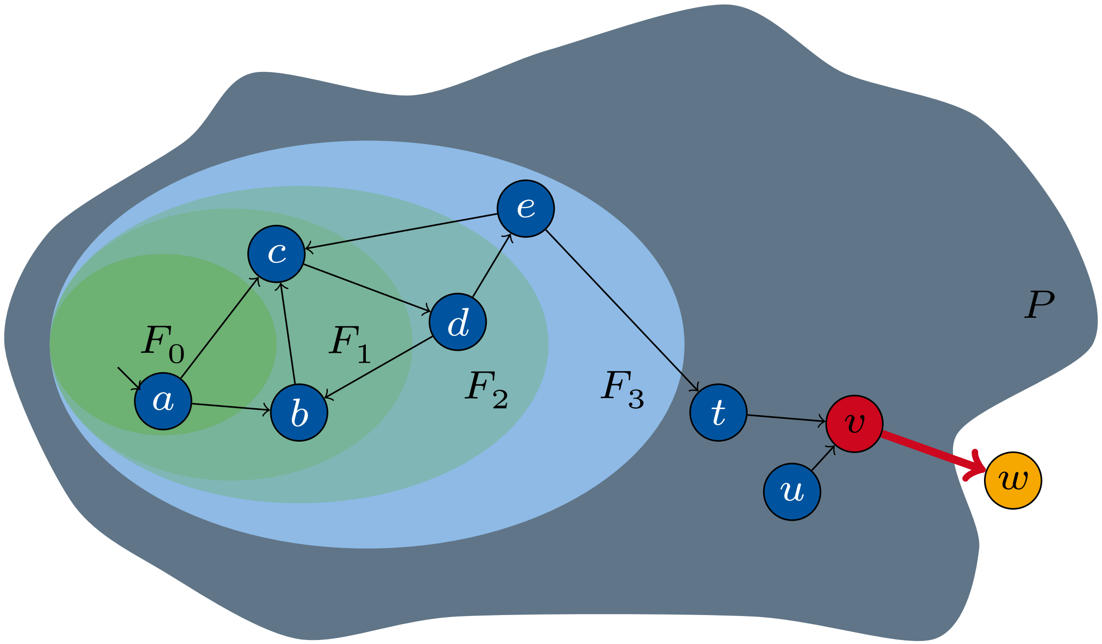
Original IC3

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



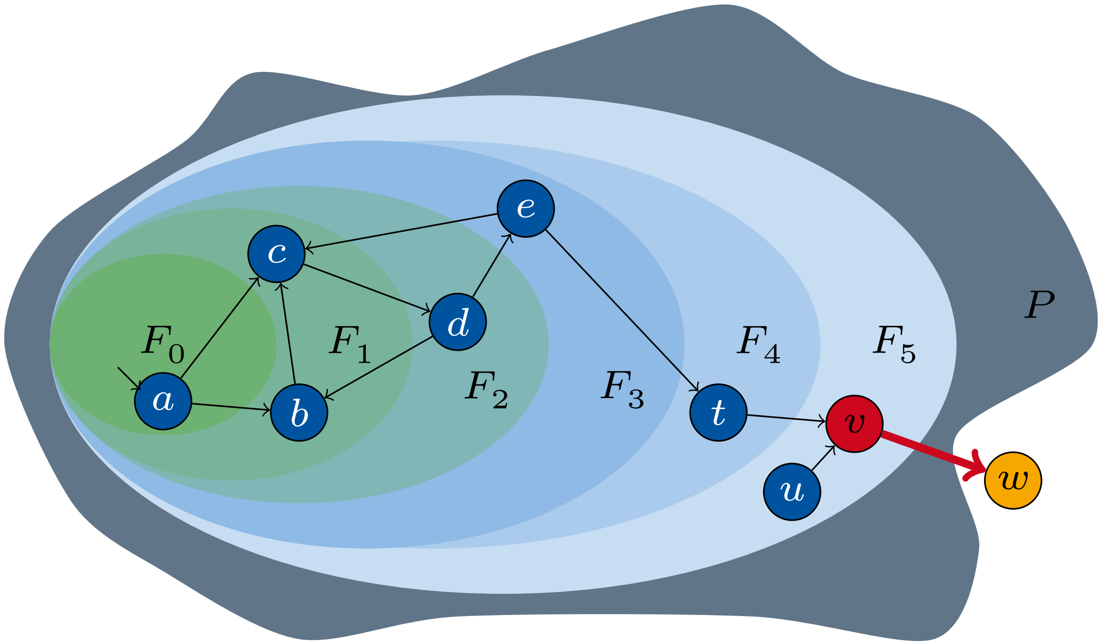
Original IC3

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



Original IC3

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



Outline

Related Work

Abstract reachability tree (ART) unrolling [CG12]

Unroll ART, search error path and refute (similarly to blocking phase of IC3).

Bit-blasting [WK13]

Encode variables as bit-vectors and use bit-blasting with bit-level IC3.

Implicit Abstraction [Cim+14]

Express abstract transitions without explicitly computing the abstract system.

Predicate Abstraction [BBW14]

Use predicate abstraction and refine predicates based on CTIs.

[CG12] Alessandro Cimatti and Alberto Griggio. "Software Model Checking via IC3". In: CAV. 2012, pp. 277–293

[WK13] Tobias Welp and Andreas Kuehlmann. "QF BV model checking with property directed reachability". In: DATE. 2013, pp. 791–796

[Cim+14] Alessandro Cimatti et al. "IC3 Modulo Theories via Implicit Predicate Abstraction". In: TACAS. 2014, pp. 46–61

[BBW14] Johannes Birgmeier, Aaron R. Bradley, and Georg Weissenbacher. "Counterexample to Induction-Guided Abstraction-Refinement (CTIGAR)". . In: CAV. 2014, pp. 831–848

Outline

IC3 on Control Flow Automata

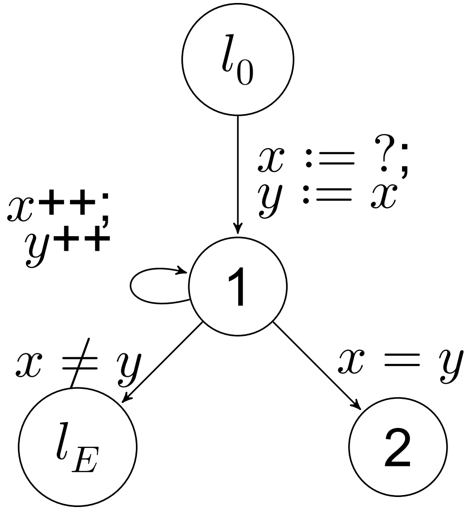
Idea

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

[CG12] Alessandro Cimatti and Alberto Griggio. "Software Model Checking via IC3". In: CAV. 2012, pp. 277–293

IC3 on Control Flow Automata

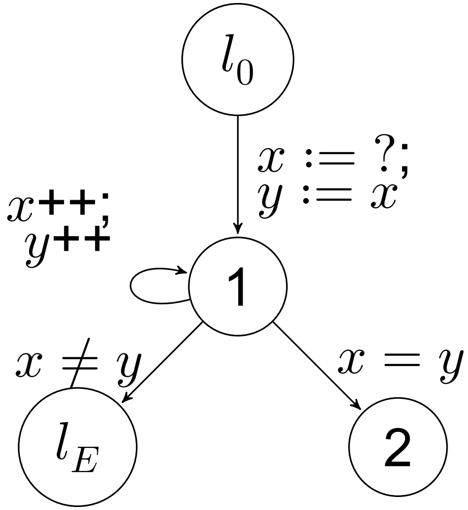
Example



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

IC3 on Control Flow Automata

Example

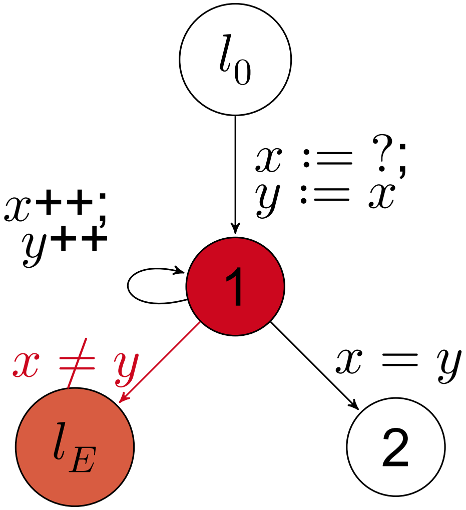


Frames $F_{(i,l)}$

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

IC3 on Control Flow Automata

Example

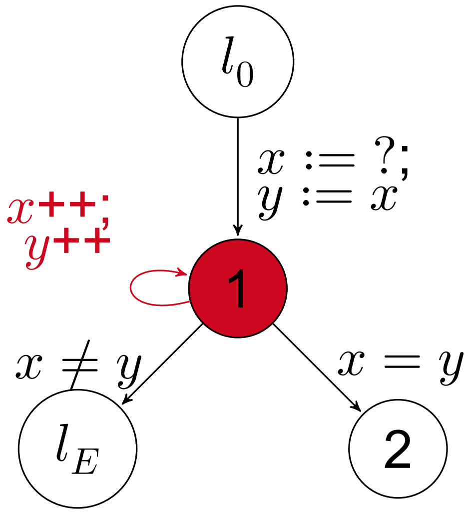


Frames $F_{(i,l)}$

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

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

Example



Frames $F_{(i,l)}$

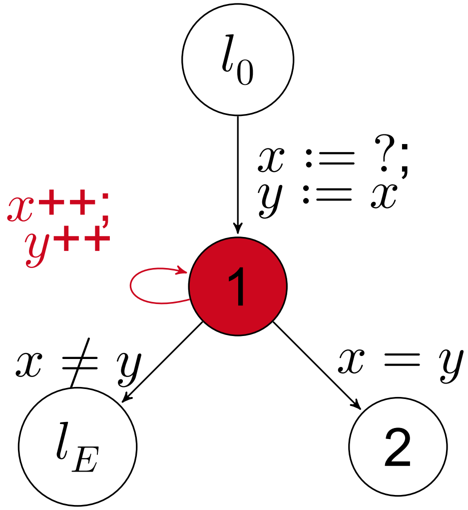
i: \ l:	l_0	1
0	true	false
1	true	true

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



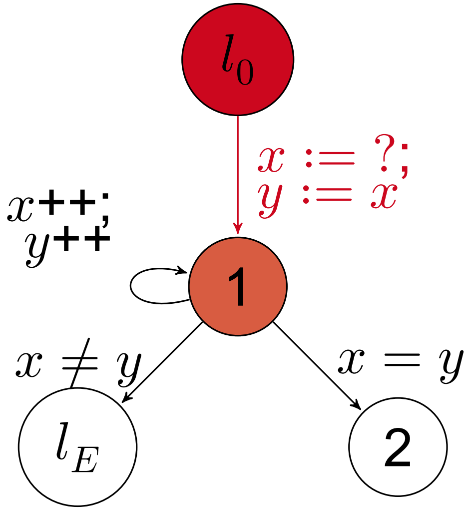
Frames $F_{(i,l)}$

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

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

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

Example



Frames $F_{(i,l)}$

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

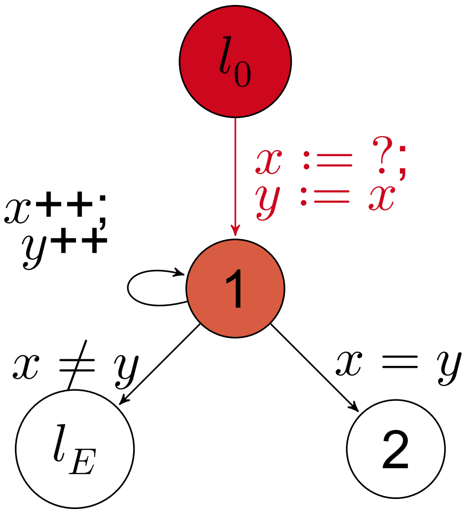
CTI $(1, x \neq y)$, level 1

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

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

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

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

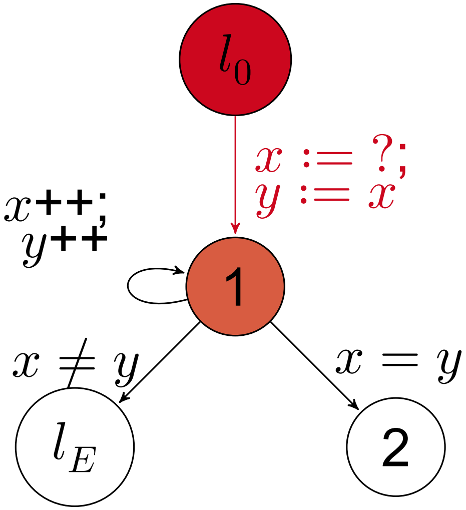
CTI (1,$x \neq y$), level 1

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

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

IC3 on Control Flow Automata

Example



Frames $F_{(i,l)}$

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

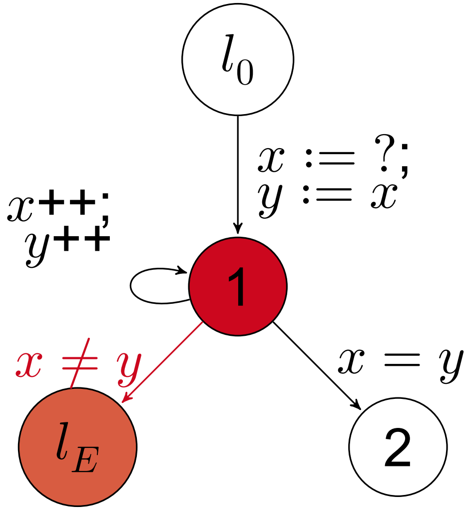
CTI (1,$x \neq y$), level 1

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

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

IC3 on Control Flow Automata

Example

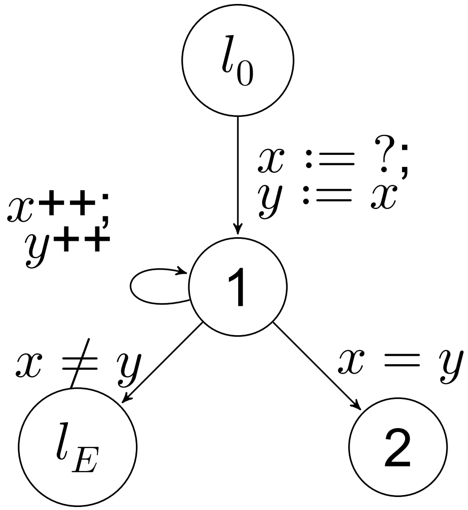


Frames $F_{(i,l)}$

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

IC3 on Control Flow Automata

Example



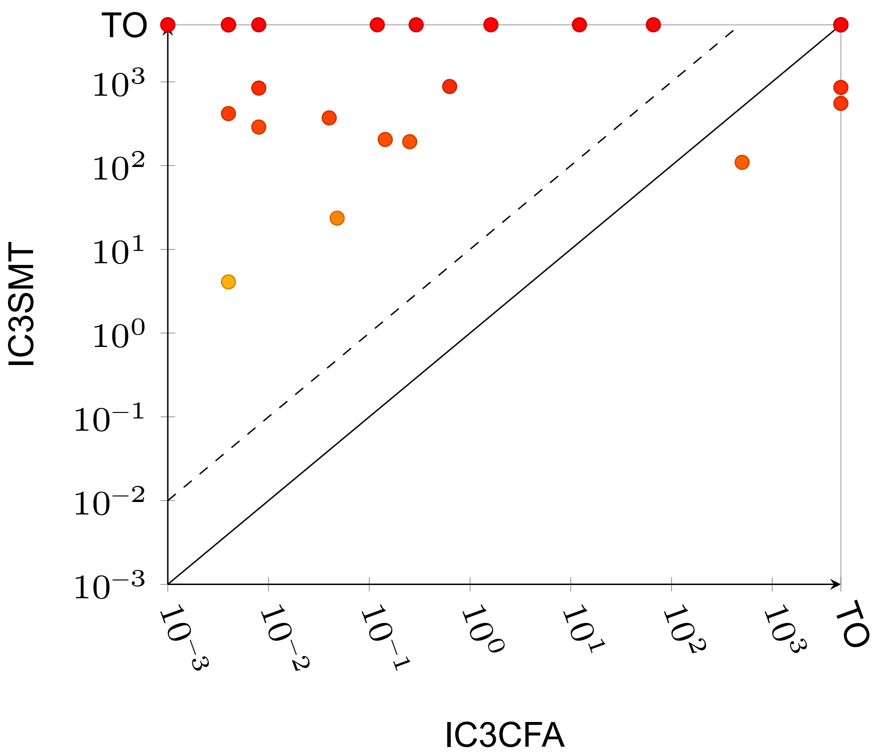
Frames $F_{(i,l)}$

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

IC3 on Control Flow Automata

Evaluation

28 benchmarks from SVCOMP & device drivers, subset of [CG12].

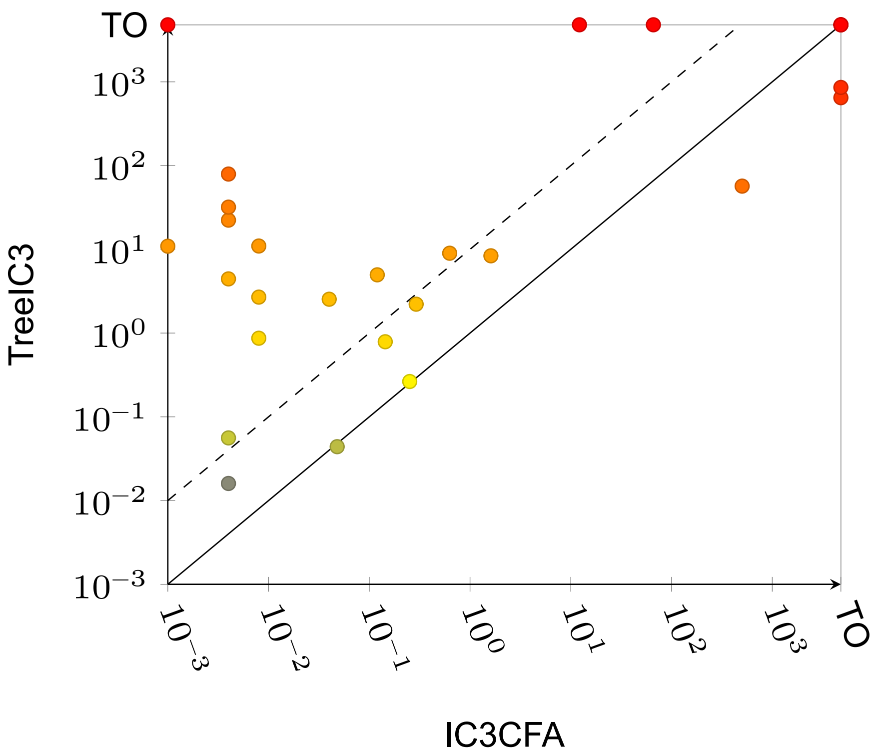


Algorithm	solved	solve time	total time
IC3SMT	13/28	6328s	24328s
IC3CFA	22/28	584s	7784s

[CG12] Alessandro Cimatti and Alberto Griggio. “Software Model Checking via IC3”. In: CAV. 2012, pp. 277–293

Evaluation

28 benchmarks from SVCOMP & device drivers, subset of [CG12].

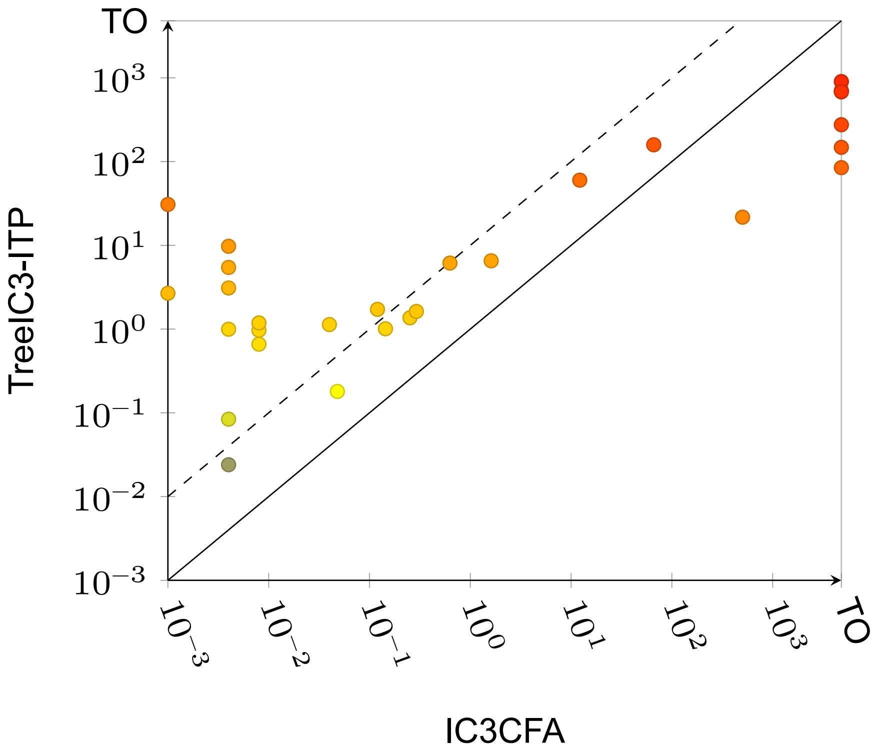


Algorithm	solved	solve time	total time
TreeIC3	21/28	1752s	10152s
IC3CFA	22/28	584s	7784s

[CG12] Alessandro Cimatti and Alberto Griggio. “Software Model Checking via IC3”. In: CAV. 2012, pp. 277–293

Evaluation

28 benchmarks from SVCOMP & device drivers, subset of [CG12].



Algorithm	solved	solve time	total time
TreeIC3-ITP	28/28	3107s	3107s
IC3CFA	22/28	584s	7784s

[CG12] Alessandro Cimatti and Alberto Griggio. "Software Model Checking via IC3". In: CAV. 2012, pp. 277–293

Outline

Conclusion

Contributions

Small SMT queries

Through inspection of only specific transitions, we can use a single edge formula instead of giving the whole transition relation to the solver.

No unrolling

By using F_i frames in every location of the CFA, we can operate on the CFA exclusively. Thus no need for unrolling the CFA.

Stronger relative inductivity

When considering self-loops we can use the stronger relative inductivity that is used in the original IC3.

Conclusion

References

- Johannes Birgmeier, Aaron R. Bradley, and Georg Weissenbacher. “Counterexample to Induction-Guided Abstraction-Refinement (CTIGAR)”. In: CAV. 2014, pp. 831–848.
- 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.
- Alessandro Cimatti et al. “IC3 Modulo Theories via Implicit Predicate Abstraction”. In: TACAS. 2014, pp. 46–61.
- Thomas A. Henzinger et al. “Lazy abstraction”. In: POPL. 2002, pp. 58–70.
- Tobias Welp and Andreas Kuehlmann. “QF BV model checking with property directed reachability”. In: DATE. 2013, pp. 791–796.