What's Happening with IC3?

Aaron Bradley

November 8, 2012

Overview

1. Overview of IC3/PDR
2. What folks are doing with it
3. What we're doing with it (mixed in)

Overview of IC3

IC3: The Big Picture (1)

$$S : (\overline{x},\ \overline{i},\ I(\overline{x}),\ T(\overline{x}, \overline{i}, \overline{x}')) \quad \text{Invariant property} : P$$

i-step over-approximating sets: $F_0,\ F_1,\ \ldots,\ F_k$

Four invariants:

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

Convergence when $\exists i \leq k.\ F_i = F_{i+1}$. Then:

1. $I \Rightarrow F_i$
2. $F_i \wedge T \Rightarrow F'_i$
3. $F_i \Rightarrow P$

$\therefore F_i$ is an inductive strengthening of P.

IC3: The Big Picture (2)

CTI (counterexample to induction): state s that reaches $\neg P$-state

IC3's response:

- Find maximum i (weakest F_i) such that

$$F_i \wedge \neg s \wedge T \Rightarrow \neg s'$$

- Inductively generalize relative to F_i: $c \subseteq \neg s$
- Refine: for $j \leq i + 1$,

$$F_j := F_j \wedge c$$

- If $i < k$, add proof obligation $(s, \; i + 1)$
  ("TO DO: generalize $\neg s$ relative to F_{i+1}")

IC3: The Big Picture (3)

Situation:

$$F_k \wedge T \Rightarrow P'$$

IC3's response:

1. For i from 1 to k, for $c \in F_i$:
   (a) if $F_i \wedge c \wedge T \Rightarrow c'$
   (b) then $F_{i+1} := F_{i+1} \wedge c$
2. If ever $F_i = F_{i+1}$ (syntactic check): converged
3. $k := k + 1$
4. $F_k := F_k \wedge P$

IC3's Themes

- Like BMC, ITP: property directed but aware of initial states
- Induction at two levels:
  - High (typical): convergence, inductive strengthening of P
  - Low: local refinements of F_i's from CTIs
- No unrolling of transition relation
  - Many easy SAT queries
  - Explicit over-approximating sets and relatively inductive clauses
  - Can be used for other purposes [Hassan et al. 12], [Claessen & Sörensson 12], [Vizel et al. 12], [Baumgartner et al. 12]

What folks are doing with it

Ternary Simulation

Failing query with CTI s and primary inputs n:

$$F_i \wedge \neg t \wedge T \not\Rightarrow \neg t'$$

[Bradley 10/11]:

- Apply i-step COI to reduce s to $\bar{s} \subseteq s$
- Find j such that

  $$F_j \wedge \neg \bar{s} \wedge T \Rightarrow \neg \bar{s}'$$

  UNSAT core reveals $c \subseteq \neg \bar{s}$, often with significant reduction

"Efficient Implementation of Property Directed Reachability" [Een et al. 11]

- For each latch ℓ appearing in s:
  1. ternary simulate $s[\ell \mapsto X]$ (with same primary inputs n)
  2. if output is still t (i.e., no Xs), drop ℓ
- Our observation: more aggressive reductions seem counterproductive

Reusing IC3's Artifacts: Introduction

- Proof improvement [Bradley et al. 11]: given strengthening

$$\bigwedge C \text{ of } P$$

  - Stronger: for $c \in C$, apply MIC to find $\bar{c} \subseteq c$ until fixpoint
  - Weaker: find minimal $\overline{C} \subseteq C$ so that $\bigwedge \overline{C}$ is still strengthening
  - Smaller: apply the previous two iteratively
  - Often produces significantly compressed strengthening
  - Useful for FAIR (ω-regular) and IICTL (CTL)
- Extract inductive clauses from SAT run (various groups)
- Reuse i-step clauses or just inductive clauses for different properties (various groups)

Incremental Verification

"Incremental Formal Verification of Hardware" [Chockler et al. 11]

- Contexts: regression verification, coverage computation
- Goal: reuse artifacts from previous analysis on *altered design*
- Invariant finder:
  - From $M_1 \models P$ with strengthening $\bigwedge C$
    - Extract maximal $\overline{C} \subseteq C$ that is inductive for M_2
    - Start IC3 with $\overline{C}$ on M_2
  - From $M_1 \not\models P$ with partial cex $t_0, \ldots, t_n$ and stepwise clauses $C = \bigcup_{i=1}^{k} F_i$
    - Aggressive "shrinking" of assignments to produce $t_0, \ldots, t_n$
    - Search for concretization of $t_0, \ldots, t_n$ on M_2
    - Or extract maximal $\overline{C} \subseteq C$ that is inductive for M_2
    - Start IC3 with $\overline{C}$ on M_2

Lazy Abstraction

"Lazy Abstraction and SAT-Based Reachability in Hardware Model Checking" [Vizel et al. 12]

- Transition relation over-approximating abstraction: $U_0, U_1, \ldots, U_k$
  - subset of state variables: $U_i \subseteq \overline{x}$
  - monotonic: $U_i \subseteq U_{i+1}$
  - U_i induces abstraction

$$\widehat{T}_i(U_i,\ (\overline{x} - U_i) \cup \overline{i},\ U_i') \ \text{of}\ \ T(\overline{x},\ \overline{i},\ \overline{x}')$$

- Abstraction/refinement (overview):
  - Abstraction: run IC3 using $\widehat{T}_i$ for queries relative to F_i
  - If IC3 returns a proof, it's valid for T
  - If IC3 returns an abstract cex, refine:
    - Run IC3's STRENGTHEN using T and current $F_0, \ldots, F_k$
    - If cex: it's concrete
    - If converges: for $j \geq i$, enlarge U_j according to clauses in F_i

Localization

"IC3-Guided Abstraction" [Baumgartner et al. 12]

- Builds on cex- and proof-based abstraction/refinement
- Abstraction: Treat some state variables as primary inputs
- Refinement: Re-introduce eliminated state variables
- Goal: Produce priorities for re-introduction based on an incomplete IC3 run
- **PM1** ("priority method 1"):
  - Initially $p(x) = \infty$ for $x \in \overline{x}$
  - If
    - x appears in clause c added when frontier is k
    - and currently $p(x) = \infty$

    then $p(x) := k$
- **RM1** ("refinement method 1"), in response to spurious cex:
  - Add assigned variables (in spurious cex) with highest priority
- **RM2**:
  - Also start with abstraction using highest-priority variables

Infinite-state Systems: Introduction (1)

Obvious (?) abstraction/refinement algorithm (using SMT):

- Abstraction domain D
- CTI s is an explicit state; keep it that way
- Generalization:
  - $\bar{s}$: strongest (conjunctive) element of D over-approximating s
  - Apply IC3's Generalize to $\bar{s}$ as usual
- Abstraction failure (Type 1):
  - $F_i \wedge \neg\bar{s} \wedge T \not\Rightarrow \neg\bar{s}'$
  - $F_i \wedge \neg s \wedge T \Rightarrow \neg s'$
  - Obtain concrete $\bar{s}$-predecessor t
  - Mark proof obligations involving t or predecessors as **abstract**
  - $t \to s$ is an **abstract-concrete trace boundary**
- Abstraction failure (Type 2)
  - **Abstract** (but still concrete) state u has I-predecessor
  - Revert to an **abstract-concrete trace boundary** $t \to s$
  - Refine: introduce a predicate blocking $\bar{s}$-predecessor t

Infinite-state Systems: Introduction (2)

- SMT queries over concrete transition relation
- Extract and work with concrete states
- Continuum:
  - Aggressive non-refinement: don't refine until a Type 2 failure
  - Aggressive refinement: refine upon Type 1 failure
  - In between: allow some depth of Type 1 failures
  - Balance refinement and IC3 effort

Timed Systems

"SMT-based Induction Methods for Timed Systems" [Kindermann et al. 12]

- Uses standard region abstraction for timed systems
- Basic idea:
  - Predicate abstraction according to region atoms
  - No need for refinement
- Compared IC3 on Booleanized model, Timed-IC3, and Timed-*k*-induction
  - Timed-* significantly superior to IC3 over Booleanized model
  - Timed-IC3 better at proofs
  - Timed-*k*-induction better at cexes (because of BMC)

IC3/Interpolant Hybrid for Software

"Software Model Checking via IC3" [Cimatti et al. 12]

- Lazy abstraction [Henzinger et al. 02], [McMillan 06]
- Unwinding of CFG into tree (à la [McMillan 06])
- No local induction
- Main contribution: hybrid local-global implementation of [McMillan 06], where local aspects are inspired by IC3
- Computes under-approximations of preimages

Constrained Horn Clauses

"Generalized Property Directed Reachability" [Hoder et al. 12]

- "Generalized": IC3/PDR over CHC with recursively-defined predicates [Bjørner et al. 12]
- Extension to linear arithmetic:
  - G-CONFLICT: Computes interpolant from UNSAT query

$$F_i \wedge T \wedge \neg s'$$

    (Compromise: Interpolants are subset of inductive assertions.)
  - G-DECIDE: Weakens single-state CTIs—still under-approximation of preimage
  - Weakening provides more opportunities for interpolants in G-CONFLICT
- Weaknesses/misunderstandings:
  - Stack rather than priority queue for proof obligations (Unnecessary: priority queue for CHC analysis is natural)
  - Interpolant-based generalization rather than full induction-based

Under-approximation of Preimages: Why (Not)?

- Why? (My question...)
  - Preference for enlarged CTIs (e.g., ternary simulation)?
  - No abstraction failure? (But "refining" all the time...)
- Why not?
  - Potentially expensive
  - Lots of refinement-like effort
- My preferences (which could be wrong):
  - Concrete non-enlarged states are OK: just compute best (current) abstractions for generalization attempts
  - Refine the abstraction domain only when necessary:
    - Type 1: $F_i \wedge \neg\bar{s} \wedge T \not\Rightarrow \neg\bar{s}'$ but $F_i \wedge \neg s \wedge T \Rightarrow \neg s'$ (abstract-concrete trace boundary $t \rightarrow s$)
    - Type 2: Spurious cex trace

CHC: Regaining Induction

$$
\begin{array}{ll}
(1) & I(X) \Rightarrow R(X) \\
(2) & R(X) \wedge F_1(X,Y) \Rightarrow U(Y) \\
(3) & U(X) \wedge F_2(X,Y) \Rightarrow R(Y) \\
(4) & R(X) \Rightarrow P(X)
\end{array}
$$

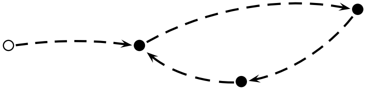
- CTI s_R from (2): provides values for R's parameters
- [Hoder et al. 12]:
  - Strengthen using known information for R and U if possible
  - Otherwise look at predecessor
  - Similar situation for explicit CFG [Cimatti et al. 12]
- Instead: Extend DOWN algorithm, e.g.,
  - From (3), extract s_U to produce expanded CTI $s_R \wedge s_U$
  - Now business as usual until convergence
- Result:
  - Generate up to **one strengthening lemma per predicate**
  - Use **induction to generalize** from CTIs
  - In the spirit of IC3

Temporal Logic: ω-regular Properties (1)

"An Incremental Approach to Model Checking Progress Properties" [Bradley et al. 11]

- Skeleton:

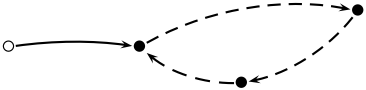
  Together satisfy all fairness constraints.
- Task: Connect states to form lasso.



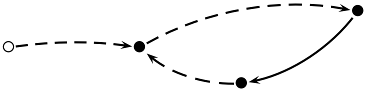
Reach Queries

Each connection task is a reach query.

- *Stem query*: Connect initial condition to a state:



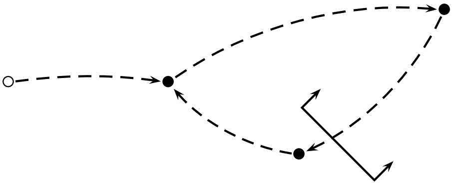
- *Cycle query*: Connect one state to another:



(To itself if skeleton has only one state.)

Discovering SCC-Closed Sets

Negative cycle query $\Rightarrow$ knowledge of SCC structure



- *Inductive* proof: "one-way barrier"
- Each "side" of the proof is SCC-closed
- Subsequent skeletons: all states on one side
- Can also find "skeleton-independent" barriers
  - p such that FGp, where p is extracted from model or property
  - SAT query: $G \wedge p \wedge T \Rightarrow p'$, where G is global non-reachability information

Temporal Logic: ω-regular Properties (2)

"A Liveness Checking Algorithm that Counts"
[Claessen & Sörensson 12]

- Main idea does not derive from IC3: bound number of times signal can be 0
- IC3 is "a very nice fit for the liveness checker" because the clause sets $F_0, \ldots, F_k$ can be saved between safety queries
- Extends "skeleton-independent" barriers [Bradley et al. 11] to statically-derived "stabilizing constraints"

Temporal Logic: CTL

"Incremental, Inductive CTL Model Checking" [Hassan et al. 12]

- With fairness
- Local CTL model checking with two types of generalization:
  - IC3-based: strengthen upper bounds
  - Trace expansion: weaken lower bounds
- Applies solvers according to nodes' semantics:
  - EXp: SAT query
  - EpUq: reachability query (IC3, BMC)
  - EGp: liveness query (FAIR, BMC with LTS)
- Generalizations:
  - Any cex trace is expanded
  - EXp: UNSAT core
  - EpUq: improved strengthening from IC3
  - EGp: global reachability information from FAIR

Conclusion

Lots of advances by a lot of people:

- Improvements (e.g., ternary simulation, other unreported implementation tricks)
- Combining IC3 with other ideas (e.g., lazy abstraction, localization, interpolation, SMT)
- Exploiting aspects of IC3 (e.g., localization, incrementality)

Lots of impressive results in which IC3 plays a humble role:

- HWMCC'12
- Reports from industry

IC3 is just one part of the amazing growth in our field over the past decades, but:

Thanks!