SAT-Based Model Checking:
IC3 and Lazy Abstraction

Verification course,
Lecture 12, June 23, 2015
Incremental Construction of Inductive Clauses for Indubitable Correctness

or simply: IC3
A Simplified Description

“SAT-Based Model Checking without Unrolling”, Aaron Bradley, VMCAI 2011
Notations

• System is modeled as \((V, I, T)\), where:
  - \(V\) is a finite set of variables
  - \(I \subseteq 2^V\) is the set of initial states
  - \(T \subseteq 2^V \times 2^V\) is the set of transitions

• A safety property of the form \(AG P\)
  - \(P\) is a propositional formula over \(V\)
Induction for proving AG P

• The simple case: P is an inductive invariant
  - I ⇒ P
  - P ∧ T ⇒ P′

• Notation: P′ - the value of P in the next state

• I(V) ⇒ P(V)
• P(V) ∧ T(V, V′) ⇒ P(V′)
Induction for proving $AG\; P$

- Usually, $P$ is not an inductive invariant
- BUT - a stronger inductive invariant $R$ may exist (strengthening)
  - $I \Rightarrow R$
  - $R \land T \Rightarrow R'$
  - $R \Rightarrow P$
- $R$ can be computed in various ways (BDDs, k-induction, Interpolation-Sequence,...)
Inductive invariant
The Goal: Find an Inductive Invariant stronger than $P$ by learning relatively inductive facts (incrementally)

- Recall: $F$ is inductive invariant if
  - $I \Rightarrow F$
  - $F \land T \Rightarrow F'$
- If $F$ is stronger than $P$, i.e., $F \Rightarrow P$, then
  - $F \land P \land T \Rightarrow F' \Rightarrow P'$
What Makes IC3 Special?

- **No unrolling** of the transition relation $T$ is required

- All previous approaches require unrolling
  - Searching for an inductive invariant
  - Unrolling = A form of strengthening

- **IC3 strengthens in a different way**
  - Learning relatively inductive facts locally
IC3 Basics

• Iteratively compute Over-Approximated Reachability Sequence (OARS) \(<F_0,F_1,\ldots,F_k>\) s.t.
  
  \[\begin{align*}
    & F_0 = \text{INIT} \\
    & F_i \implies P \quad : \text{P is an invariant up to } k \\
    & F_i \implies F_{i+1} \quad : \text{F}_i \subseteq \text{F}_{i+1} \\
    & F_i \land T \implies F'_{i+1} \quad : \text{Simulates one forward step}
  \end{align*}\]

  \[F_i\] - over-approximates the set of states reachable within \(i\) steps

• If \(F_{i+1} \Rightarrow F_i\) then \text{fixpoint}
IC3 Basics

• P is inductive relative to F if
  - I \implies P
  - F \land P \land T \implies P'

• Notations:
  - Cube s: conjunction of literals
    - \nu_1 \land \nu_2 \land \neg \nu_3 - Represents a state
  - s is a cube \implies \neg s is a clause (DeMorgan)
OARS

\[ R_1 = I \lor \text{Img}(I,T) \]
\[ R_2 = R_1 \lor \text{Img}(R_1,T) \]
A Backward Search

• Search for a predecessor $s$ to some error state: $P \land T \land \neg P'$
  - If none exists, property $P$ holds:
    • $(P \land T \land \neg P') \text{ unsat IFF } (P \land T \Rightarrow P') \text{ valid}$

• Otherwise, try to block $s$
  - $P = P \land \neg s$
  - BUT, first need to show the $s$ is not reachable
IC3 - Initialization

• Check satisfiability of the two formulas:
  - \( I \land \neg P \)
  - \( I \land T \land \neg P' \)

• If both are unsatisfiable then:
  - \( I \Rightarrow P \)
  - \( I \land T \Rightarrow P' \)

• Therefore
  - \( F_0 = I, F_1 = P \)
    • \( \langle F_0, F_1 \rangle \) is OARS
IC3 - Initialization
IC3 - Iteration

- Our OARS contains \( F_0 \) and \( F_1 \)
  - If \( P \) is an inductive invariant - done! 😊
  - Otherwise:
    - \( F_1 \) should be strengthened
IC3 - Iteration

- P is not an inductive invariant
  - $F_1 \land T \land \neg P'$ is satisfiable
  - From the satisfying assignment get the state $s$ that can reach the bad states
IC3 - Iteration

- Is $s$ reachable or not?
  - Hard to know
  - If it is reachable a CEX exists
    • Why?
IC3 - Iteration

- Is s reachable in one transition from the previous set? (Bounded reachability)
  - Check $F_0 \land T \land s'$
  - If satisfiable, s is reachable from $F_0$ (CEX)
  - Otherwise, block it = remove it from $F_1$
    - $F_1 = F_1 \land \neg s$
IC3 - Iteration

• Iterate this process until $F_1 \land T \land \neg P'$ becomes unsatisfiable
  - $F_1 \land T \Rightarrow P'$ holds
  - $F_2$ can be defined to be $P$
    • Any problems/issues with that?
IC3 - Iteration

• New iteration, check $F_2 \land T \land \neg P'$
  - If satisfiable, get $s$ that can reach $\neg P$
  - Now check if $s$ can be reached from $F_1$ by $F_1 \land T \land s'$
  - If it can be reached, get $t$ and try to block it
IC3 - Iteration

- To block $t$, check $F_0 \land T \land t'$
  - If satisfiable, a CEX
  - If not, $t$ is blocked, get a "new" $t$ by $F_1 \land T \land s'$
  - If it can be reached, get $t^*$ and try to block it
  - ......You get the picture 😊
General Iteration
IC3 - Iteration

• Given an OARS \(<F_0,F_1,\ldots,F_k,>\), define \(F_{k+1}=P\)
• Apply a backward search
  - Find predecessor \(s\) in \(F_k\) that can reach a bad state
    • Check \(F_k \land T \land \neg P'\)
  - If none exists \((F_k \land T \Rightarrow P')\), move to next iteration
  - If exists, try to find a predecessor \(t\) to \(s\) in \(F_{k-1}\)
    • \((F_{k-1} \land T \land s')\)
  - If none exists \((F_{k-1} \land T \Rightarrow \neg s')\), \(s\) is removed from \(F_k\)
    • \(F_k = F_k \land \neg s\)
  - Otherwise: Recur on \((t,F_{k-1})\)
    • We call \((t,k-1)\) a proof obligation
• If we can reach I, a CEX exists
That Simple?

- Looks simple
- But this “simple” solution does NOT work
- It amounts to States Enumeration
  - Too many states...
- Does IC3 enumerate states?
  - In general - No.
    - It applies generation for removing more than one state at a time
  - Sometimes, yes (when IC3 does not perform well)
Generalization

Consider the case:

• State $s$ in $F_k$ can reach a bad state in one transition

• $s$ in not reachable (in $k$ transitions):
  - Therefore $F_{k-1} \land T \Rightarrow \neg s'$ holds

• We want to generalize this fact
  - $s$ is a single state
  - Goal: Find a set of states, unreachable in $k$ transitions
Generalization

• We know $F_{k-1} \land T \Rightarrow \neg s'$
• And, $\neg s$ is a clause
• Generalization: Find a sub-clause $c \subseteq \neg s$ s.t.
  $F_{k-1} \land T \Rightarrow c'$
  - Sub clause means less literals
  - Less literals implies less satisfying assignments
    • $(a \lor b \lor c)$ vs. $(a \lor b)$
  - $c \Rightarrow \neg s$  - $c$ is a stronger fact
• $F_k = F_k \land c$
  - More states are removed from $F_k$, making it
    stronger/more precise (closer to $R_k$)
Generalization

• How do we find a sub-clause $c \subseteq \neg s$ s.t. $F_{k-1} \land T \Rightarrow c'$?

• Trial and Error
  - Try to remove literals from $\neg s$ while $F_{k-1} \land T \land \neg c'$ remains unsatisfiable

• Use the UnSAT Core
  - $F_{k-1} \land T \land s'$ is unsatisfiable
Observation 1

• Assume a state $s$ in $F_k$ can reach a bad state in one transition
• Important Fact: $s$ is not in $F_{k-1}$ (!!!)
  - $F_{k-1} \land T \Rightarrow F_k$
  - $F_k \Rightarrow P$
  - If $s$ was in $F_{k-1}$ we would have found it in an earlier iteration
• Therefore: $F_{k-1} \Rightarrow \neg s$
Inductive Generalization

• Assume a state $s$ in $F_k$ can reach a bad state in one transition

• Assume $s$ is not reachable (in $k$ transitions):
  - We get $F_{k-1} \land T \Rightarrow \neg s'$ holds

• BUT, this is equivalent: $F_{k-1} \land \neg s \land T \Rightarrow \neg s'$
  - Since $F_{k-1} \Rightarrow \neg s$

• This looks familiar!
  - $I \Rightarrow \neg s$
    - Otherwise, CEX! ($I \not\Rightarrow \neg s \iff s$ is in $I$)
  - $\neg s$ is inductive relative to $F_{k-1}$
Inductive Generalization

• Find \( c \subseteq s \) s.t.
  \( F_{k-1} \land c \land T \Rightarrow c' \) and \( I \Rightarrow c \) hold

• Define \( F_k^* = F_k \land c \)

• Since \( F_i \Rightarrow F_{i+1} \),
  \( c \) is inductive relative to \( F_{k-1}, F_{k-2}, \ldots, F_0 \)
  - Add \( c \) to all of these sets
  - \( F_i^* = F_i \land c \)

• \( F_i^* \land T \Rightarrow F_{i+1}^* \) hold
Observation 2

• Assume a state $s$ in $F_i$ can reach a bad state in a number of transitions
• $s$ is also in $F_j$ for $j > i$, since $F_i \Rightarrow F_j$
• A longer CEX may exist
  - $s$ may not be reachable in $i$ steps, but it may be reachable in $j$ steps
• If $s$ is blocked in $F_i$, it must be blocked in $F_j$ for $j > i$
  - Otherwise, a CEX exists
Push Forward
Push Forward - summary

• $s$ is removed from $F_i$
  - by conjoining a sub-clause $c$:
    \[ F_i = F_i \land c \]

• $c$ is a clause learnt at level $i$
  Try to push it forward to $j \geq i$
  - If $F_j \land T \implies c'$ holds
    • $c$ is implied by $F_j$ in level $j+1$,
      \[ F_{j+1} = F_{j+1} \land c \]
  - Else: $s$ was not blocked at level $j > i$
    • Add a proof obligation $(s,j)$
    • If $s$ is reachable from $I$, CEX!
IC3 - Key Ingredients

- **Backward Search**
  - Find a state $s$ that can reach a bad state in a number of steps
  - $s$ may not be reachable (over-approximations)

- **Block a State**
  - Do it efficient, block more than $s$
    - Generalization

- **Push Forward**
  - An inductive fact at frame $i$ may also be inductive at higher frames
  - If not, a longer CEX is found
IC3 - High Level Algorithm

If $I \land \neg P$ is SAT return false; // CEX
If $I \land T \land \neg P'$ is SAT return false; // CEX
OARS = $<I,P>$;  // $<F_0,F_1>$
k=1
while (OARS.is_fixpoint() == false) do
  while ($F_k \land T \land \neg P'$ is SAT) do
    s = get_state();
    If (block_state(s, k) == false) return cex; // recursive function
    extend(OARS);
    push_forward();
return valid;
Lazy Abstraction and SAT-Based Reachability (with IC3) in Hardware Model Checking

[Vizel, Grumberg, Shoham 12]
Abstraction

• Fights the state explosion problem
• Removes or simplifies details that are irrelevant

• Abstract model contains less states
• Often - more behaviors
  - Over-approximation
Visible Variables Abstraction
Abstraction-Refinement

• Abstract model may contain spurious behaviors
  – Spurious counterexample may exist

• Refinement is applied to remove the spurious behavior
Lazy Abstraction

• Different abstractions at different steps of verification

• Refinement is applied locally, where needed
Locality in IC3

- **IC3 applies checks of the form**
  - $F_k \land T \land \neg P'$
    - Finds a state in $F_k$ that can reach $\neg P$
  - $F_i \land T \land s'$
    - Finds a predecessor in $F_i$ to the state $s$

- **Using only one T**
  - No unrolling
Our Approach - L-IC3

• Use IC3's local checks for Lazy Abstraction
  - Different abstraction at different time frames
  - Use visible variables abstraction
    • Different variables are visible at different time frames
Concrete Model

INIT → $F_1$ → $F_{k-1}$ → $F_k$
Using Abstraction

INIT $\rightarrow F_1 \rightarrow \cdots \rightarrow F_{k-1} \rightarrow F_k$
Using Lazy Abstraction
Lazy Abstraction + IC3 = L-IC3

- \( \langle F_0, F_1, \ldots, F_k, 1 \rangle \) - Reachable states

- \( \langle U_1, U_2, \ldots, U_{k+1} \rangle \) - Abstractions
  - \( U_i \) - set of visible variables
    - \( U_i \) variables have a next state function
    - The rest, inputs
  - \( U_i \subseteq U_{i+1} \)
    - \( U_{i+1} \) is a refinement of \( U_i \)
L-IC3 Iteration

- Initialize $F_{k+1}$ to $P$
- Initialize $U_{k+1}$ to $U_k$
- Same problem, the sequence may not be an OARS
Abstract Counterexample

\[ F_i \land T_{i+1} \land s' \]

\[ F_k \land T_{k+1} \land \neg P' \]

\[ -P \]
Check Spuriousness

- An abstract CEX of length \( k+1 \) exists
- Use an IC3 iteration with the concrete \( T \)
- If a real CEX exists, it will be found
Check Spuriousness (2)

• If no real CEX exists:
  - Compute a strengthened sequence
    \( \langle F_{r_0}, F_{r_1}, \ldots, F_{r_{k+1}} \rangle \)
    • Strengthening by IC3 algorithm
  - The strengthened sequence is an OARS
  - Strengthening eliminates all (real) CEXs of length \( k+1 \)
Lazy Abstraction Refinement

• If no real CEX is found by (concrete) IC3 even though (abstract) L-IC3 strengthening failed
  - Abstraction is too coarse

• Refine the sequence \(<U_1, U_2, \ldots, U_{k+1}>\) as follows:

• Since \(Fr_i \land T \Rightarrow Fr'_{i+1}\)
  - \(Fr_i \land T \land \neg Fr'_{i+1}\) is unsatisfiable
  - Use the UnSAT Core to add visible variables
    • \(Ur_{i+1} = Ui_{i+1} \cup UCore_i\)
Incrementality

• The concrete IC3 iteration works on the already computed sequence \( \langle F_0, F_1, \ldots, F_{k+1} \rangle \)

• At the end of refinement, L-IC3 continues from iteration \( k+2 \)
## Experiments - Laziness

| Test | #Vars | #TF | #AV | #TF | #AV | #TF | #AV | #TF | #AV | #TF | #AV | #TF | #AV |
|------|-------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Ind 2| 5693  | 7-1 | 31  | 8   | 42  | 9   | 51  | 10-14| 54  |      |     |     |     |
| Ind 3| 11866 | 1   | 323 | 2   | 647 | 3   | 686 | 4   | 699 | 5   | 705 |     |     |     |
|      | 6     | 713 | 7   | 714 | 8   | 728 | 9   | 743 |     |     |     |     |     |     |
| Ind 5| 3854  | 1   | 428 | 2   | 453 | 3   | 495 | 4   | 499 | 5   | 503 |     |     |     |
|      | 6     | 560 | 7   | 574 | 8   | 657 | 9-11| 577 |     |     |     |     |     |     |
Summary

• Lazy abstraction algorithm for hardware model checking
• Abstraction-Refinement is done incrementally
• We compared our method (L-IC3) to Bradley’s method (IC3)
  - Up to two orders of magnitude runtime improvement
Conclusions

• L-IC3 combines two approaches to fight the state-explosion problem
• L-IC3 exposes and exploits the abstraction, implicit in IC3
Intertwined Forward-Backward Reachability Analysis Using Interpolants

[Vizel, Grumberg, Shoham, TACAS 2013]
Interpolants

- Given an inconsistent pair $(A,B)$ of propositional formulas
- There exists a formula $I$ such that:
  - $A \rightarrow I$
  - $I \land B$ is unsatisfiable
  - $I$ is over the common variables of $A$ and $B$
- $I = \text{Itp}(A,B)$
Approximated Forward Reachability

• F(V) - a set of states
• For the unsatisfiable formula F(V) ∧ T(V,V') ∧ ¬P(V'), define:
  \[ A = F(V) \land T(V,V') \]
  \[ B = \lnot P(V') \]
• Approximated forward reachability:
  \[ \text{ApxImg}(F,T) = \text{Itp}(A,B) \]
Backward Reachability Analysis

Does $AGp$ hold?

$B_n = \text{PreImg}(B_{n-1}, T)$

$B_2 = \text{PreImg}(B_1, T)$

$B_1 = \text{PreImg}(\neg P, T)$

Bad $= \neg P$
Duality In a SAT Query

- \( \text{INIT}(V) \land T(V,V') \land \neg P(V') \)
- We tend to read it "Forward"
  - From left to right

Do we reach the bad states?
Duality In a SAT Query

- \textbf{INIT}(V) \land T(V,V') \land \neg P(V')

- We tend to read it "Forward"
  - From left to right

- We can also read it "Backward"
  - From right to left
  - Does the pre-image of the bad states intersect the initial states?
Approximated Backward Reachability

- $B(V)$ - a set of states
- For the unsatisfiable formula $\text{INIT}(V) \land T(V,V') \land B(V')$, define:
  \[ A = T(V,V') \land B(V') \]
  \[ B = \text{INIT}(V) \]

Approximated backward reachability:
$\text{ApxPreImg}(B,T) = \text{Itp}(A,B)$
Dual Approximated Reachability (DAR)

- Compute two sequences of reachable states
  - Forward Sequence: \(<F_0,F_1,\ldots,F_n>\)
  - Backward Sequence: \(<B_0,B_1,\ldots,B_n>\)
- Sequences are over-approximations
  - For the forward sequence:
    - \(F_i(V) \land T(V,V') \rightarrow F_{i+1}(V')\)
    - \(F_i(V) \rightarrow P(V)\)
  - For the backward sequence
    - \(B_{i+1}(V) \leftarrow T(V,V') \land B_i(V')\)
    - \(B_i(V) \rightarrow \neg INIT(V)\)
Dual Approximated Reachability (DAR)

- Two main phases during the computation
  - Local Strengthening
    - No unrolling
  - Global Strengthening
    - Limited unrolling
    - In case the Local Strengthening fails
Dual Approximated Reachability

- Check the formula:
  \[ \text{INIT}(V) \land T(V, V') \land \neg P(V') \]

- If SAT then CEX is found
Dual Approximated Reachability

- UNSAT:

\[ F_0 = \text{INIT} \]

\[ B \]

\[ \text{INIT}(V) \land T(V, V') \land \neg P(V') \]

\[ A \]

\[ B_1 \]

\[ B_0 = \neg P \]
Local Strengthening - Intuition

What if $F_1$ and $B_1$ intersect each other?

There may be a counterexample
Local Strengthening - Intuition

What if $F_1$ and $B_1$ intersect each other?

\[ F_1(V) \land T(V, V') \land B_0(V') \]

\[ F_0(V) \land T(V, V') \land B_1(V') \]
Local Strengthening - Intuition

- Compute forward and backward interpolants
  - $F_2$ is the forward interpolant
  - Backward interpolant strengthens the already existing $B_1$

$$F_1(V) \land T(V, V') \land B_0(V')$$
Local Strengthening - Intuition

- Compute forward and backward interpolants
  - $B_2$ is the backward interpolant
  - $F'_1$ is strengthening the already existing $F_1$

$F_0(V) \land T(V,V') \land B_1(V')$ must be UnSAT
Local Strengthening Fails

$$F_0(V) \land T(V, V') \land B_0(V')$$
Global Strengthening

• Apply unrolling gradually
  – Start from the initial states
  – Try to reach the backward sequence using an increasing number of T’s
Global Strengthening

\[ F_0(V) \land T(V, V') \land T(V', V'') \land T(V'', V'''') \land B_1(V''') \land \neg P \]

\[ F_0(V) \land T(V, V') \land T(V', V'') \land T(V'', V'''') \land T(V''', V''''') \land B_2(V''''') \land \neg P \]

\[ F_0(V) \land T(V, V') \land T(V', V'') \land T(V'', V'''') \land T(V''', V''''') \land B_3(V''''') \land \neg P \]
Interpolation-Sequence

• Given a sequence \(<A_1,\ldots,A_n>\) such that its conjunction is unsatisfiable
• Then, there exists an interpolation sequence \(<I_0,\ldots,I_n>\) such that:
  - \(I_0 = \text{TRUE}, I_n = \text{FALSE}\)
  - \(I_i \land A_{i+1} \implies I_{i+1}\)
  - \(I_i\) is over the common variables of \(A_1,\ldots,A_i\) and \(A_{i+1},\ldots,A_n\)
Global Strengthening

\[ F_0(V) \land T(V, V') \land T(V', V'') \land T(V'', V''') \land B_1(V''') \]
Global Strengthening

• If a CEX exists - Full unrolling
• Otherwise, gradually unroll the model
  – Try to reach the Backward sequence
• When the backward sequence is not reachable
  – Extract interpolation sequence
  – Strengthen forward sequence
  – Reapply Local Strengthening
Summary

• Interpolation-based model checking algorithm
• Uses both Forward and Backward traversals
• Two main phases during the computation
  – Local Strengthening
    • No unrolling
  – Global Strengthening
    • Limited unrolling
    • In case the Local Strengthening fails
• Mostly local – No unrolling
  – When unrolling is used, it is restricted
Summary

We presented several methods for SAT-based (unbounded) model checking

- Over-approximate the (forward) reachability analysis
- Apply different methods for making the over-approximation more precise
Thank You
Model checking:

- E.M. Clarke, A. Emerson, Synthesis of Synchronization Skeletons for Branching Time Temporal Logic, workshop on Logic of programs, 1981


- E.M. Clarke, O. Grumberg, D. Peled, Model Checking, MIT press, 1999
• **BDDs:**

• **BDD-based model checking:**

• **SAT-based Bounded model checking:**
  Symbolic model checking using SAT procedures instead of BDDs,
  A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, Y. Zhu, DAC'99
• **Visible variables abstraction:**

• **Lazy abstraction:**
Interpolation based model checking:

- K. McMillan, Interpolation and SAT-Based Model Checking, CAV'03

- T. Henzinger, R. Jhala, R. Majumdar, K. McMillan, Abstractions from Proofs, POPL'04

- Y. Vizel and O. Grumberg, Interpolation-Sequence Based Model Checking, FMCAD'09

- Y. Vizel, O. Grumberg, S. Shoham, Intertwined Forward-Backward Reachability Analysis Using Interpolants, TACAS'13
Model checking with IC3:

- A. Bradley, SAT-based model checking without unrolling, VMCAI’11

- Y. Vizel, O. Grumberg, S. Shoham, Lazy abstraction and SAT-based reachability in hardware model checking, FMCAD’12