COL750: Foundations of Automatic Verification (Jul-Dec 2024)

(IC3 – SAT-Based Model Checking without Unrolling)

Kumar Madhukar

madhukar@cse.iitd.ac.in

Oct. 30, Nov. 4

Interpolation → Inductive Invariants

initial

saturate

$A \wedge B$
is unsat

BMC
unsat

compute
interpolant

$\overline{a} \vee b \vee c$
$\overline{a} \vee d \vee e$
$(b \vee c \vee d \vee e)$

Interpolating quadruples $(A, B)\ c\ [f]$

W1. $V(c) \subseteq V(A) \cup V(B)$

W2. $V(f) \subseteq V(A) \cap V(B) \cup V(c) \cap V(A)$

$(A, B) \perp [f]$

V1: $A \Rightarrow f$

V2: $B \wedge f \Rightarrow c$

$\boxed{(A, B) \perp [f]}$

R1: $\dfrac{}{(A, B)\ c\ [c]}$ $c \in A$

R2: $\dfrac{}{(A, B)\ c\ [\top]}$ $c \in B$

R3: $\dfrac{(A, B)\ c \vee l\ [f] \quad (A, B)\ d \vee \overline{l}\ [g]}{(A, B)\ c \vee d\ [f \wedge g]}$ $l \notin \epsilon$

R4: $\dfrac{(A, B)\ c \vee l\ [f] \quad (A, B)\ d \vee \overline{l}\ [g]}{(A, B)\ c \vee d\ [f\ l\ \overline{l} \vee g\ l\ \overline{l}]}$ $V(B)$ $l \in V(B)$

$B \wedge f \Rightarrow c \vee e$

$B \wedge f \mid \bar{e} \Rightarrow R_1^c$

$c \vee e$

$B \wedge f \Rightarrow c$ R_2

$B \wedge f \mid \bar{e} \Rightarrow c$ R_3

R_4

w_1	w_2	v_1	v_2
✓	✓	✓	✓
✓	✓	✓	✓
✓	✓	✓	✓
✓	✓	✓	✓

$B \wedge (f \wedge g) \Rightarrow c \vee d$

$B \wedge f \Rightarrow c \vee e$

$B \wedge g \Rightarrow d \vee \bar{e}$

$B \wedge f \wedge g \Rightarrow (c \vee e) \wedge (d \vee \bar{e})$

$\Rightarrow (c \vee d)$

$B \wedge g \Rightarrow d \vee \bar{e}$

$B \wedge g \mid e \Rightarrow d$

Case I: $e \in V(B)$ and $e \in V(A)$

Case I: $e \in V(B)$ and $e \notin V(A)$

$V(f) \subseteq V(A) \cap V(B) \cup V(c \vee e) \cap V(A)$

$V(g) \subseteq V(A) \cap V(B) \cup V(d \vee \bar{e}) \cap V(A)$

$V(f \wedge g) \subseteq V(A) \cap V(B) \cup \boxed{V(c \vee d) \cap V(A)}$

$B \wedge f|_{\bar{e}} \Rightarrow c$

$B \wedge g|_{e} \Rightarrow d$

$$V(f) \subseteq V(A) \cap V(B) \cup V(c \vee e) \cap V(A)$$

$$V(g) \subseteq V(A) \cap V(B) \cup V(d \vee \bar{e}) \cap V(A)$$

$$V(f|_{\bar{e}} \vee g|_{e}) \subseteq V(A) \cap V(B) \cup V(c \vee d) \cap V(A)$$

$(B \wedge f|_{\bar{e}}) \vee (B \wedge g|_{e}) \Rightarrow c \vee d$

$B \wedge [f|_{\bar{e}} \vee g|_{e}] \Rightarrow c \vee d$

Pick a satisfying assignment for A

$A \Rightarrow f$

$\cancel{B \Rightarrow g}$

$A \to g$

$e \to$ true

$\bar{e} \to$ false

$$A = [f|_{\bar{e}} \vee g|_{e}]$$

$\Longrightarrow (A, B) \perp [c \vee d] \leftarrow$ interpolants

Quick remarks about Interpolation

$$\exists a . (a \vee c \vee d) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee \bar{c} \vee \bar{d})$$

$c \oplus d$

- the strongest interpolant of A is obtained from A by existentially quantifying over all local variables in A

$(c \vee d) \wedge (\neg c \vee \neg d)$ $\quad A \Rightarrow I$ $\quad (c \vee d)$

- thus, interpolation can be seen as an over-approximation of quantifier elimination
- in our example, we had obtained the interpolant $c \vee d$, where the strongest interpolation would have been $c \oplus d$

$A = a, c, d$

$B = b, c, d$

Quick remarks about Interpolation

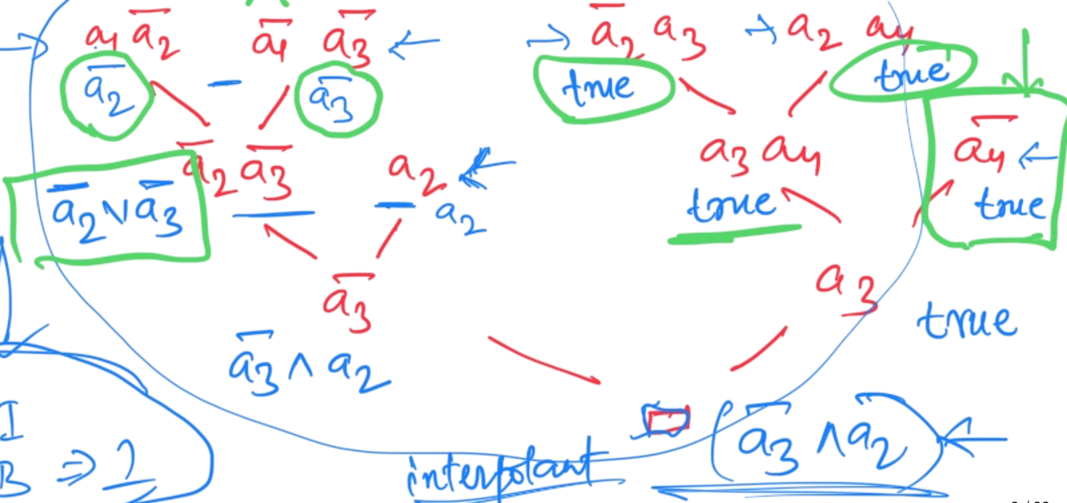
Here is a somewhat easier-to-remember method for annotating the resolution proof to obtain an interpolant:

1. for an initial node corresponding to a clause $c \in A$, annotate with c' where c' is obtained from c by keeping only those literals whose variables occur in B
2. for an initial node corresponding to a clause $c \in B$, annotate with *true*
3. for a derived node with the pivot variable x occurring in B, annotate with the conjunction of its parents' annotations
4. for a derived node with the pivot variable x not occurring in B, annotate with the disjunction of its parents' annotations

$A \Rightarrow I$

$I \wedge B \Rightarrow \bot$

$$\underbrace{(a_1 \vee \overline{a_2}) \wedge (\overline{a_1} \vee \overline{a_3}) \wedge a_2}_{A} \wedge \underbrace{(\overline{a_2} \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a_4}}_{B}$$



$A \Rightarrow I$

$I \wedge B \Rightarrow \bot$

Interpolation and SAT-Based MC

IC3 ←

Next class

- keeps only one candidate invariant (Q)
- when a bad state is reachable from the over-approximation, the over-approximation is not refined
- instead, the over-approximation is discarded completely and the transition system is unrolled further

$(A \wedge B)$ is sat → $Q = S_0$ return violation otherwise $Q = S_0$ increase K

UNSAT interpolant

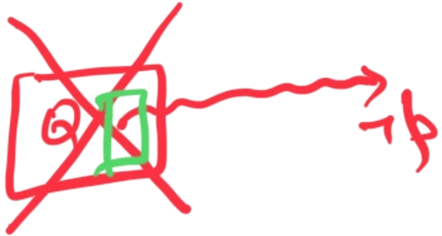
$\boxed{Q} := Q \cup I$

SAT-Based Model Checking without Unrolling

$Q(s_0) \wedge T(s_0, s_1) \wedge T(s_1, s_2) \wedge T(s_2, s_3)$

$\ldots \; T(s_i, s_{i+1})$

- without making copies of the transition relation
- computes over-approximation of the post-image of the set of reachable states
- maintains multiple candidate invariants



Frames and Invariants

- done by maintaining frames – $F_0, F_1, \ldots, F_k$ – which are step-wise assumptions (or over-approximations)

$\boxed{F_0}$ $\boxed{F_1}$ $\boxed{F_2}$ $\boxed{F_3}$ . . . .

- the frames maintain the following invariants

  1. $I_0 \rightarrow F_0$ (F0 contains the initial set of states)
  2. $F_i \rightarrow F_{i+1} \quad (0 \leq i < k)$ (frames are monotonic)
  3. $F_i \rightarrow P \quad (0 \leq i \leq k)$ (none of the frames contain a bad, i.e. $\neg P$, state)
  4. $F_i \wedge T \rightarrow F'_{i+1} \quad (0 \leq i < k)$ (F_i over-approximates i-step reachability)

Inductive Reasoning

to prove that P is an invariant (that every reachable state satisfies P), it suffices to prove that

1. all initial states satisfy P
   $init(x) \rightarrow P(x)$ (*initiation*)
2. a P-state can only be followed by a P-state
   $P(x) \wedge trans(x, x') \rightarrow P(x')$ (*consecution*)

Inductive Reasoning

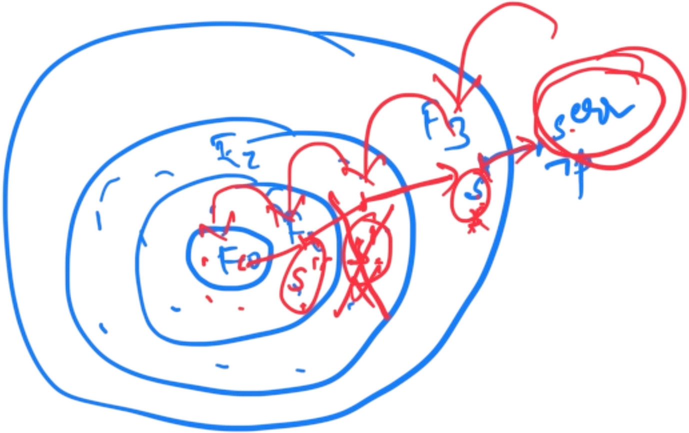
to prove that P is an invariant (that every reachable state satisfies P), it suffices to prove that

1. all initial states satisfy P
   $init(x) \rightarrow P(x)$ (*initiation*)
2. a P-state can only be followed by a P-state
   $P(x) \wedge trans(x, x') \rightarrow P(x')$ (*consecution*)

CTI

however, P itself may not be inductive; it may help to have a stronger assertion in that case

1. $init(x) \rightarrow f(x)$ (*initiation*)
2. $f(x) \wedge \boxed{trans(x, x')} \rightarrow f(x')$ (*consecution*)
3. $f(x) \rightarrow P(x)$ (*safety*)


F3
F2
F0
s
s
sem

Example

$$\begin{cases} x = 1 \Rightarrow x \geq 1 \quad \checkmark \\ x \geq 1 \wedge x' = x + 1 \Rightarrow x' \geq 1 \quad \checkmark \end{cases}$$

```
x = 1;
y = 1;

while(*)
  x, y = x + 1, y + x
```

suppose we want to prove the property, P, that $y \geq 1$ is an invariant

$x=1$
$\wedge y=1$ $\Rightarrow$ $y \geq 1 \wedge x \geq 1$

$$y \geq 1 \wedge x \geq 1 \quad \wedge \quad x' = x+1 \wedge y' = y+x$$
$$\Rightarrow \underline{y' \geq 1 \wedge x' \geq 1}$$

$$y \geq 1 \wedge x \geq 1 \quad \Rightarrow \quad y \geq 1$$

Example

- $(y \geq 1)$ is not an inductive invariant (why? the consecution check fails)
- so, we must look for a strengthening of $(y \geq 1)$
- $(x \geq 0 \wedge y \geq 1)$ is an inductive invariant; but how do we obtain this?
- counterexample to induction (CTI) from the failed consecution check: $[x = -1, y = 1]$
- the strengthening $(x \geq 0)$ must *eliminate* the CTI

$f \Rightarrow P$

$y \geq 1$

- $(x \geq 0)$ is an inductive invariant
- $(y \geq 1)$ is inductive *relative to* $(x \geq 0)$

$$(x \geq 0) \;\wedge\; (y \geq 1) \;\wedge\; y' = y + x \;\wedge\; x' = x + 1 \;\rightarrow\; (y' \geq 1)$$

- thus, an incremental proof is possible

$x \geqslant 1 \wedge y \geqslant 1$

$x \geqslant 1 \wedge y \geqslant 1$

Another example

```
x = 1;
y = 1;

while(*)
  x, y = x + y, y + x
```

$\boxed{x \geq 0}$ is not an inductive invariant

suppose we want to prove the property, P, that $y \geq 1$ is an invariant

Another example

- as in case of previous example, $y \geq 1$ is an invariant but not inductive
- we get a CTI last like time: $[x = -1, y = 1]$
- $(x \geq 0)$ eliminates the CTI but isn't inductive (unlike the last time)
- but it is inductive relative to the property

$$(y \geq 1) \wedge (x \geq 0) \wedge y' = y + x \wedge x' = x + y \rightarrow (x' \geq 0)$$

- seemingly circular reasoning, but not actually so

$$P \wedge \psi \wedge T \rightarrow \psi' \qquad \text{and} \qquad \psi \wedge P \wedge T \rightarrow P'$$

together imply that $\psi \wedge P$ is an inductive invariant

$P \wedge \psi \wedge T \rightarrow P' \wedge \psi'$

- thus, an incremental proof is still possible (though it may not be possible in every case; exercise – construct an example where the entire inductive strengthening must be obtained at once)

$\psi_1, \psi_2, \psi_3, \ldots$

$\psi_1 \wedge \psi_2 \wedge \psi_3$

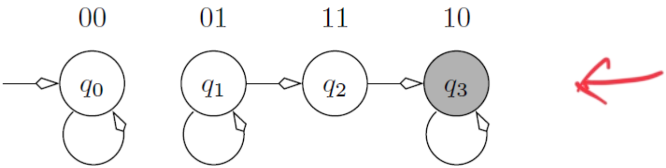
Back to frames and invariants

- check that $I \to P$ (that none of the initial states are bad), and set F_0 to I
- check $(I =) F_0 \wedge T \to P'$ (that bad is not 1-step reachable), and set F_1 to P
- now, we check $F_1 \wedge T \to P'$
- if not, there must be a CTI $s \in F_1$ that can reach $\neg P$ in one step
- but $s \notin F_0$, else it would have been discovered earlier (while checking $F_0 \wedge T \to P'$)
- so, we check if s is reachable from F_0 in one step ($F_0 \wedge \neg s \wedge T \to \neg s'$)
- if yes, then s has a predecessor s_{pre} in F_0 (we need to check if s_{pre} is an initial state, or if it has a predecessor, and so on..)
- if not, then $F_1 := (F_1 \wedge \neg s)$ [it may be better to generalize the CTI instead of just eliminating one state at a time]

$F_1 = F_1 \wedge \neg S$

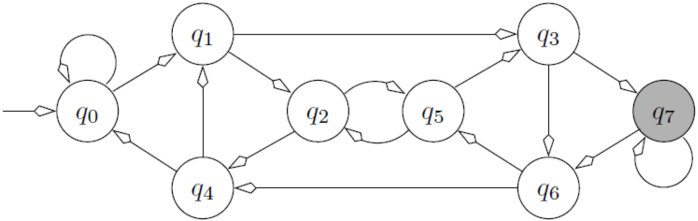
Come here
but it should
not be here.

IC3 on a safe example[1]



[1] Reference: https://theory.stanford.edu/~arbrad/papers/ic3_tut.pdf

IC3 on an unsafe example[2]



[2] Reference: https://theory.stanford.edu/~arbrad/papers/ic3_tut.pdf

Algorithm

procedure PDR (model M, property P)

if $(I_0 \wedge \neg P)$ is SAT, **return** "P does not hold"
$F_0 \leftarrow I_0$; $k \leftarrow 0$;

while true **do**

extendFrontier(M, k)
propagateClauses(M, k)
if $F_i = F_{i+1}$ for some i, **return** "P holds"
$k \leftarrow k + 1$

end while

end procedure

Algorithm

procedure extendFrontier (M, k)

$F_{k+1} \leftarrow P$

while $F_k \wedge T \wedge \neg P'$ is SAT **do**

$s' \leftarrow$ state labelled with $\neg P$ extracted from the satisfying assignment
$s \leftarrow$ predecessor of s' extracted from the satisfying assignment
$removeCTI(M, s, k)$

end while

end procedure

Algorithm

procedure removeCTI (M, s, i)

if $I_0 \wedge s$ is SAT, **return** "P does not hold"

while $F_i \wedge T \wedge \neg s \wedge s'$ is SAT **do**

for $j \in [0, i]$
$F_j \leftarrow F_j \wedge \neg s$
end for

$t \leftarrow$ predecessor of s extracted from the SAT witness
$removeCTI(M, t, i-1)$

end while

end procedure

Algorithm

procedure propagateClauses (M, k)

for $i \in [1, k]$
for every clause $c \in F_i$
if $F_i \wedge T \wedge \neg c'$ is UNSAT
$F_{i+1} \leftarrow F_{i+1} \wedge c$
end if
end for
end for

end procedure

Thank you!