Solving Existentially Quantified Horn Clauses: The Solving Algorithm E-HSF

Tewodros Beyene[1], Corneliu Popeea[1], and Andrey Rybalchenko[1,2]

[1]Technische Universität München

[2]Microsoft Research Cambridge

Rich Model Toolkit COST Action Meeting

Malta, June 17, 2013

E-HSF briefly - Input and Output

- A set of horn clauses as input.
- Some can be existentially quantified;i.e.*with existentially quantified head*.
  - Example: $x = 5 \rightarrow \exists\ y : x + y \geq 10$.
- Extends HSF - algorithm for quantifier free horn clauses.
- As an output, it may
  - return a solution,
  - return a counter example, or
  - simply diverge.

E-HSF briefly - Input and Output

- A set of horn clauses as input.
- Some can be existentially quantified;i.e.*with existentially quantified head*.
  - Example: $x = 5 \rightarrow \exists\, y : x + y \geq 10$.
- Extends HSF - algorithm for quantifier free horn clauses.
- As an output, it may
  - return a solution,
  - return a counter example, or
  - simply diverge.

E-HSF briefly - Input and Output

- A set of horn clauses as input.
- Some can be existentially quantified;i.e.*with existentially quantified head*.
  - Example: $x = 5 \rightarrow \exists\ y : x + y \geq 10$.
- Extends HSF - algorithm for quantifier free horn clauses.
- As an output, it may
  - return a solution,
  - return a counter example, or
  - simply diverge.

An Example

A program with:

- variables $v = (x, y)$,
- initial condition $init(v) = (y \geq 1)$, and
- transition relation $next(v, v') = (x' = x + y)$.

CTL property: $EF\ dst(v)$, where $dst(v) = (x \geq 0)$

Horn clause encoding:

$$\begin{aligned}
&init(v) \rightarrow inv(v),\\
&inv(v) \wedge \neg dst(v) \rightarrow \exists v' : next(v, v') \wedge inv(v') \wedge rank(v, v'),\\
&rank(v, v') \rightarrow ti(v, v'),\\
&ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''),\\
&dwf(ti).
\end{aligned}$$

Unknowns: $inv(v)$, $rank(v, v')$, and $ti(v, v')$.

An Example

A program with:

- variables $v = (x, y)$,
- initial condition $init(v) = (y \geq 1)$, and
- transition relation $next(v, v') = (x' = x + y)$.

CTL property: $EF\ dst(v)$, where $dst(v) = (x \geq 0)$

Horn clause encoding:

$$
\begin{aligned}
& init(v) \rightarrow inv(v), \\
& inv(v) \wedge \neg dst(v) \rightarrow \exists v' : next(v, v') \wedge inv(v') \wedge rank(v, v'), \\
& rank(v, v') \rightarrow ti(v, v'), \\
& ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''), \\
& dwf(ti).
\end{aligned}
$$

Unknowns: $inv(v)$, $rank(v, v')$, and $ti(v, v')$.

An Example

A program with:

- variables $v = (x, y)$,
- initial condition $init(v) = (y \geq 1)$, and
- transition relation $next(v, v') = (x' = x + y)$.

CTL property: $EF\ dst(v)$, where $dst(v) = (x \geq 0)$

Horn clause encoding:

$$init(v) \rightarrow inv(v),$$
$$inv(v) \wedge \neg dst(v) \rightarrow \exists v' : next(v, v') \wedge inv(v') \wedge rank(v, v'),$$
$$rank(v, v') \rightarrow ti(v, v'),$$
$$ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''),$$
$$dwf(ti).$$

Unknowns: $inv(v)$, $rank(v, v')$, and $ti(v, v')$.

An Example - Skolemization

- Application of a skolem relation $rel(v, v')$.
- Lower bound on the guard $grd(v)$ of the skolem relation.

$$
\begin{aligned}
& init(v) \rightarrow inv(v), \\
& inv(v) \land \neg dst(v) \land rel(v, v') \rightarrow next(v, v') \land inv(v') \land rank(v, v'), \\
& inv(v) \land \neg dst(v) \rightarrow grd(v), \\
& rank(v, v') \rightarrow ti(v, v'), \\
& ti(v, v') \land rank(v', v'') \rightarrow ti(v, v''), \\
& dwf(ti).
\end{aligned}
$$

An Example - First E-HSF Iteration I

- Initial candidates for the Skolem relation and its Guard.

$$Defs = \{true \rightarrow rel(v, v'),\ grd(v) \rightarrow true\} \ .$$

- Initialise *Constraint* with the assertion *true*.
- *Clauses* now contains the result of Skolemization and *Defs*.
- Apply the solving algorithm HSF.

An Example - First E-HSF Iteration II

$$
\begin{aligned}
& init(v) \rightarrow inv(v), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow next(v, v'), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow inv(v'), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow rank(v, v'), \\
& inv(v) \wedge \neg dst(v) \rightarrow grd(v), \\
& rank(v, v') \rightarrow ti(v, v'), \\
& ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''), \\
& dwf(ti), \\
& true \rightarrow rel(v, v'), \\
& grd(v) \rightarrow true.
\end{aligned}
$$

An Example - First E-HSF Iteration II

$$
\begin{aligned}
&init(v) \rightarrow inv(v),\\
&inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow next(v, v'),\\
&inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow inv(v'),\\
&inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow rank(v, v'),\\
&inv(v) \wedge \neg dst(v) \rightarrow grd(v),\\
&rank(v, v') \rightarrow ti(v, v'),\\
&ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''),\\
&dwf(ti),\\
&true \rightarrow rel(v, v'),\\
&grd(v) \rightarrow true.
\end{aligned}
$$

An Example - First E-HSF Iteration II

$$
\begin{aligned}
& init(v) \rightarrow inv(v), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow next(v, v'), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow inv(v'), \\
& inv(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow rank(v, v'), \\
& inv(v) \wedge \neg dst(v) \rightarrow grd(v), \\
& rank(v, v') \rightarrow ti(v, v'), \\
& ti(v, v') \wedge rank(v', v'') \rightarrow ti(v, v''), \\
& dwf(ti), \\
& true \rightarrow rel(v, v'), \\
& grd(v) \rightarrow true.
\end{aligned}
$$

An Example - Analysing the First Counter-example I

- Counter example consists of clauses *Cex*

$$\begin{aligned}
&init(v) \rightarrow q_1(v),\\
&q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow next(v, v'),\\
&true \rightarrow q_2(v, v').
\end{aligned}$$

- SSA renaming: $\textsc{Sym}(q_1) = inv$ and $\textsc{Sym}(q_2) = rel$.
- $y \geq 1 \wedge \neg(x \geq 0) \rightarrow x' = x + y$.
- $Cex \setminus Defs$: $y \geq 1 \wedge \neg(x \geq 0) \wedge q_2(v, v') \rightarrow x' = x + y$.
- Use template $v' = Tv + t$ for the skolem relation $rel(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge v' = Tv + t \rightarrow x' = x + y$.

An Example - Analysing the First Counter-example I

- Counter example consists of clauses *Cex*

$$init(v) \rightarrow q_1(v),$$
$$q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow next(v, v'),$$
$$true \rightarrow q_2(v, v').$$

- SSA renaming: $\textsc{Sym}(q_1) = inv$ and $\textsc{Sym}(q_2) = rel$.
- $y \geq 1 \wedge \neg(x \geq 0) \rightarrow x' = x + y$.
- $Cex \setminus Defs$: $y \geq 1 \wedge \neg(x \geq 0) \wedge q_2(v, v') \rightarrow x' = x + y$.
- Use template $v' = Tv + t$ for the skolem relation $rel(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge v' = Tv + t \rightarrow x' = x + y$.

An Example - Analysing the First Counter-example I

- Counter example consists of clauses *Cex*

$$\begin{aligned} &init(v) \rightarrow q_1(v), \\ &q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow next(v, v'), \\ &true \rightarrow q_2(v, v'). \end{aligned}$$

- SSA renaming: $\textsc{Sym}(q_1) = inv$ and $\textsc{Sym}(q_2) = rel$.
- $y \geq 1 \wedge \neg(x \geq 0) \rightarrow x' = x + y$.
- $Cex \setminus Defs$: $y \geq 1 \wedge \neg(x \geq 0) \wedge q_2(v, v') \rightarrow x' = x + y$.
- Use template $v' = Tv + t$ for the skolem relation $rel(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge v' = Tv + t \rightarrow x' = x + y$.

An Example - Analysing the First Counter-example I

- Counter example consists of clauses *Cex*

$$
\begin{aligned}
& init(v) \rightarrow q_1(v), \\
& q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow next(v, v'), \\
& true \rightarrow q_2(v, v').
\end{aligned}
$$

- SSA renaming: $\textsc{Sym}(q_1) = inv$ and $\textsc{Sym}(q_2) = rel$.
- $y \geq 1 \wedge \neg(x \geq 0) \rightarrow x' = x + y$.
- $Cex \setminus Defs$: $y \geq 1 \wedge \neg(x \geq 0) \wedge q_2(v, v') \rightarrow x' = x + y$.
- Use template $v' = Tv + t$ for the skolem relation $rel(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge v' = Tv + t \rightarrow x' = x + y$.

An Example - Analysing the First Counter-example II

- For our example,
  - T is a matrix of unknown coefficients $\left(\begin{smallmatrix} t_{xx} & t_{xy} \\ t_{yx} & t_{yy} \end{smallmatrix}\right)$,
  - t is a vector of unknown free coefficient (t_x, t_y),
  - $x' = t_{xx}x + t_{xy}y + t_x$ and $y' = t_{yx}x + t_{yy}y + t_y$
- $y \geq 1 \wedge \neg(x \geq 0) \wedge$ $x' = t_{xx}x + t_{xy}y + t_x \wedge$ $y' = t_{yx}x + t_{yy}y + t_y$ $\rightarrow x' = x + y$.
- Conjoin with *Constraint* (true at the start), and solve.
- SMT provides $x' = x + y \wedge y' = 10$ as solution.
- Update skolem relation definitions:

$$Defs = \{x' = x + y \wedge y' = 10 \rightarrow rel(v, v'),\ grd(v) \rightarrow true\}$$

An Example - Analysing the First Counter-example II

- For our example,
  - T is a matrix of unknown coefficients $\begin{pmatrix} t_{xx} & t_{xy} \\ t_{yx} & t_{yy} \end{pmatrix}$,
  - t is a vector of unknown free coefficient (t_x, t_y),
  - $x' = t_{xx}x + t_{xy}y + t_x$ and $y' = t_{yx}x + t_{yy}y + t_y$
- $y \geq 1 \wedge \neg(x \geq 0) \wedge$ $x' = t_{xx}x + t_{xy}y + t_x \wedge$ $y' = t_{yx}x + t_{yy}y + t_y$ $\rightarrow x' = x + y$.
- Conjoin with *Constraint* (true at the start), and solve.
- SMT provides $x' = x + y \wedge y' = 10$ as solution.
- Update skolem relation definitions:

$$Defs = \{x' = x + y \wedge y' = 10 \rightarrow rel(v, v'),\ grd(v) \rightarrow true\}$$

An Example - Analysing the First Counter-example II

- For our example,
  - T is a matrix of unknown coefficients $\begin{pmatrix} t_{xx} & t_{xy} \\ t_{yx} & t_{yy} \end{pmatrix}$,
  - t is a vector of unknown free coefficient (t_x, t_y),
  - $x' = t_{xx}x + t_{xy}y + t_x$ and $y' = t_{yx}x + t_{yy}y + t_y$
- $y \geq 1 \wedge \neg(x \geq 0) \wedge$ $x' = t_{xx}x + t_{xy}y + t_x \wedge$ $y' = t_{yx}x + t_{yy}y + t_y$ $\rightarrow x' = x + y$.
- Conjoin with *Constraint* (true at the start), and solve.
- SMT provides $x' = x + y \wedge y' = 10$ as solution.
- Update skolem relation definitions:

$$Defs = \{x' = x + y \wedge y' = 10 \rightarrow rel(v, v'),\ grd(v) \rightarrow true\}$$

An Example - Analysing the First Counter-example II

- For our example,
  - T is a matrix of unknown coefficients $\left(\begin{smallmatrix} t_{xx} & t_{xy} \\ t_{yx} & t_{yy} \end{smallmatrix}\right)$,
  - t is a vector of unknown free coefficient (t_x, t_y),
  - $x' = t_{xx}x + t_{xy}y + t_x$ and $y' = t_{yx}x + t_{yy}y + t_y$
- $y \geq 1 \wedge \neg(x \geq 0) \wedge$ $x' = t_{xx}x + t_{xy}y + t_x \wedge$ $y' = t_{yx}x + t_{yy}y + t_y$ $\rightarrow x' = x + y$.
- Conjoin with *Constraint* (true at the start), and solve.
- SMT provides $x' = x + y \wedge y' = 10$ as solution.
- Update skolem relation definitions:

$$\mathit{Defs} = \{x' = x + y \wedge y' = 10 \rightarrow \mathit{rel}(v, v'),\ \mathit{grd}(v) \rightarrow \mathit{true}\}$$

An Example - Second E-HSF Iteration

- Counter example is obtained with *Cex*.

$$
\begin{aligned}
& init(v) \rightarrow q_1(v), \\
& q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow q_3(v, v'), \\
& x' = x + y \wedge y' = 10 \rightarrow q_2(v, v'), \\
& q_3(v, v') \rightarrow q_4(v, v'),
\end{aligned}
$$

- SSA renaming: $\textsc{Sym}(q_1) = inv, \textsc{Sym}(q_2) = rel, \textsc{Sym}(q_3) = rank,$ $\textsc{Sym}(q_4) = ti$.
- $Cex \setminus Defs$: $init(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow q_4(v, v')$.

An Example - Second E-HSF Iteration

- Counter example is obtained with *Cex*.

$$
\begin{aligned}
&init(v) \rightarrow q_1(v),\\
&q_1(v) \wedge \neg dst(v) \wedge q_2(v, v') \rightarrow q_3(v, v'),\\
&x' = x + y \wedge y' = 10 \rightarrow q_2(v, v'),\\
&q_3(v, v') \rightarrow q_4(v, v'),
\end{aligned}
$$

- SSA renaming: $\textsc{Sym}(q_1) = inv, \textsc{Sym}(q_2) = rel, \textsc{Sym}(q_3) = rank$, $\textsc{Sym}(q_4) = ti$.
- *Cex* $\setminus$ *Defs*: $init(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow q_4(v, v')$.

An Example - Analysing the Second Counter-example I

- $\textsc{Sym}(q_4) = ti$ and $dwf(ti) \in Skolemized$ implies violation of disjunctive well-foundedness.
- Construct templates $bound(v)$ and $decrease(v, v')$.
  - $bound(v) = (r_x x + r_y y \geq r_0)$.
  - $decrease(v, v') = (r_x x' + r_y y' \leq r_x x + r_y y - 1)$.
- $init(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow q_4(v, v')$.
- $init(v) \wedge \neg dst(v) \wedge v' = Tv + t \rightarrow bound(v) \wedge decrease(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge x' = t_{xx}x + t_{xy}y + t_x \wedge y' = t_{yx}x + t_{yy}y + t_y \rightarrow r_x x + r_y y \geq r_0 \wedge r_x x' + r_y y' \leq r_x x + r_y y - 1$.

An Example - Analysing the Second Counter-example I

- $\textsc{Sym}(q_4) = ti$ and $dwf(ti) \in Skolemized$ implies violation of disjunctive well-foundedness.
- Construct templates $bound(v)$ and $decrease(v, v')$.
  - $bound(v) = (r_x x + r_y y \geq r_0)$.
  - $decrease(v, v') = (r_x x' + r_y y' \leq r_x x + r_y y - 1)$.
- $init(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow q_4(v, v')$.
- $init(v) \wedge \neg dst(v) \wedge\ v' = Tv + t \rightarrow bound(v) \wedge decrease(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge\ x' = t_{xx}x + t_{xy}y + t_x \wedge y' = t_{yx}x + t_{yy}y + t_y \rightarrow r_x x + r_y y \geq r_0 \wedge r_x x' + r_y y' \leq r_x x + r_y y - 1$.

An Example - Analysing the Second Counter-example I

- $\mathrm{SYM}(q_4) = ti$ and $dwf(ti) \in Skolemized$ implies violation of disjunctive well-foundedness.
- Construct templates $bound(v)$ and $decrease(v, v')$.
  - $bound(v) = (r_x x + r_y y \geq r_0)$.
  - $decrease(v, v') = (r_x x' + r_y y' \leq r_x x + r_y y - 1)$.
- $init(v) \wedge \neg dst(v) \wedge rel(v, v') \rightarrow q_4(v, v')$.
- $init(v) \wedge \neg dst(v) \wedge v' = Tv + t \rightarrow bound(v) \wedge decrease(v, v')$.
- $y \geq 1 \wedge \neg(x \geq 0) \wedge x' = t_{xx} x + t_{xy} y + t_x \wedge y' = t_{yx} x + t_{yy} y + t_y \rightarrow r_x x + r_y y \geq r_0 \wedge r_x x' + r_y y' \leq r_x x + r_y y - 1$.

An Example - Analysing the Second Counter-example II

- Add this constraint to *Constraint*, and apply SMT solver:
  - $x \leq -1$ for bound,
  - $x' \geq x + 1$ for decrease, and
  - $x' = x + 1 \land y' = 1$ for the template $v' = Tv + t$.
- But, solution for $rel(v, v')$ is not compatible with the one obtained at the first iteration...$x' = x + y \land y' = 10$.
- Hence, modify *Defs*:

$$Defs = \{x' = x + 1 \land y' = 1 \rightarrow rel(v, v'),\ grd \rightarrow true\}$$

An Example - Analysing the Second Counter-example II

- Add this constraint to *Constraint*, and apply SMT solver:
  - $x \leq -1$ for bound,
  - $x' \geq x + 1$ for decrease, and
  - $x' = x + 1 \wedge y' = 1$ for the template $v' = Tv + t$.
- But, solution for $rel(v, v')$ is not compatible with the one obtained at the first iteration...$x' = x + y \wedge y' = 10$.
- Hence, modify *Defs*:

$$Defs = \{x' = x + 1 \wedge y' = 1 \rightarrow rel(v, v'),\ grd \rightarrow true\}$$

An Example - Analysing the Second Counter-example II

- Add this constraint to *Constraint*, and apply SMT solver:
  - $x \leq -1$ for bound,
  - $x' \geq x + 1$ for decrease, and
  - $x' = x + 1 \land y' = 1$ for the template $v' = Tv + t$.
- But, solution for $rel(v, v')$ is not compatible with the one obtained at the first iteration... $x' = x + y \land y' = 10$.
- Hence, modify *Defs*:

$$Defs = \{x' = x + 1 \land y' = 1 \rightarrow rel(v, v'),\ grd \rightarrow true\}$$

An Example - Third E-HSF Iteration

Application of *HSF* returns a solution such that

$$
\begin{aligned}
inv(v) &= y \geq 1 \ , \\
rel(v) &= (x' = x + 1 \wedge y' = 1) \ , \\
rank(v, v') &= (x \leq -1 \wedge x' \geq x + 1) \ , \\
ti(v, v') &= (x \leq -1 \wedge x' \geq x + 1) \ .
\end{aligned}
$$

E-HSF finishes here!

A little detail: Skolemization

- Reformulates the problem as a problem of finding witnesses for the existentially quantified variables.
- For the clause $body(v) \rightarrow \exists w : head(v, w)$, the skolem relation $rel(v, w)$ determines which value w satisfies $head(v, w)$ for a given v.
- Each v such that $body(v)$ holds is required to be in the domain of the skolem relation.
- Domain of skolem relation $rel(v, w)$ represented as the guard $grd(v)$.

A little detail: Skolemization

- Reformulates the problem as a problem of finding witnesses for the existentially quantified variables.
- For the clause $body(v) \rightarrow \exists w : head(v, w)$, the skolem relation $rel(v, w)$ determines which value w satisfies $head(v, w)$ for a given v.
- Each v such that $body(v)$ holds is required to be in the domain of the skolem relation.
- Domain of skolem relation $rel(v, w)$ represented as the guard $grd(v)$.

A little detail: Skolemization

- Reformulates the problem as a problem of finding witnesses for the existentially quantified variables.
- For the clause $body(v) \rightarrow \exists w : head(v, w)$, the skolem relation $rel(v, w)$ determines which value w satisfies $head(v, w)$ for a given v.
- Each v such that $body(v)$ holds is required to be in the domain of the skolem relation.
- Domain of skolem relation $rel(v, w)$ represented as the guard $grd(v)$.

A little detail: Skolemization

- Reformulates the problem as a problem of finding witnesses for the existentially quantified variables.
- For the clause $body(v) \rightarrow \exists w : head(v, w)$, the skolem relation $rel(v, w)$ determines which value w satisfies $head(v, w)$ for a given v.
- Each v such that $body(v)$ holds is required to be in the domain of the skolem relation.
- Domain of skolem relation $rel(v, w)$ represented as the guard $grd(v)$.

A little detail: Skolem Template

- Templates determine the search space for:
  - skolem relations,
  - their guards, and
  - termination arguments for well-foundedness.
- Template functions GRDT and RELT should satisfy the following condition: for each (grd, rel) that results from skolemization of a given existential clause, the implication

$$\mathrm{GRDT}(grd)(v) \rightarrow \exists w : \mathrm{RELT}(rel)(v, w) \tag{1}$$

  is valid.
- Established by choosing templates accordingly!

A little detail: Skolem Template

- Templates determine the search space for:
  - skolem relations,
  - their guards, and
  - termination arguments for well-foundedness.
- Template functions GrdT and RelT should satisfy the following condition: for each (grd, rel) that results from skolemization of a given existential clause, the implication

$$\text{GrdT}(grd)(v) \to \exists w : \text{RelT}(rel)(v, w) \tag{1}$$

  is valid.
- Established by choosing templates accordingly!

A little detail: Skolem Template

- Templates determine the search space for:
  - skolem relations,
  - their guards, and
  - termination arguments for well-foundedness.
- Template functions $\textsc{GrdT}$ and $\textsc{RelT}$ should satisfy the following condition: for each (grd, rel) that results from skolemization of a given existential clause, the implication

$$\textsc{GrdT}(grd)(v) \rightarrow \exists w : \textsc{RelT}(rel)(v, w) \quad (1)$$

  is valid.
- Established by choosing templates accordingly!

E-HSF briefly - Algorithm

```
 1: skolemize each existential clause by creating a skolem relation.
 2: for current set of candidate skolem solutions do
 3:     if all clauses are satisfied then
 4:         terminate declaring SAT
 5:     else
 6:         analyse the counter example path,
 7:         if a skolem relation is not involved then
 8:             terminate declaring UNSAT
 9:         else
10:             encode clauses without the skolem solutions as a constraint,
                store the constraint into induced constraint,
11:             if induced constraint is valid then
12:                 update candidate skolem solutions, and go to 2
13:             else
14:                 terminate declaring UNSAT
15:             end if
16:         end if
17:     end if
18: end for
```

Correctness

The algorithm E-HSF relies on the following propositions.

Lemma (Skolemization preserves satisfiability)

The set of clauses *Clauses* is equi-satisfiable with the set of clauses computed by SKOLEMIZE when domains of Skolem relations contain corresponding guards. Formally, *Clauses* is equi-satisfiable with the set

$$\{grd(v) \rightarrow \exists w : rel(v, w) \mid grd \in Grds \wedge rel \in Rels \wedge \\ Parent(grd) = Parent(rel)\} \cup Skolemized \ .$$

Soundness and Progress

Theorem (Soundness)

If HSF is sound, i.e., it returns solutions for given sets of clauses, and if $\textsc{GrdT}(grd)(v) \rightarrow \exists w : \textsc{RelT}(rel)(v, w)$ holds for each $grd \in Grds$ and $rel \in Rels$ such that $Parent(grd) = Parent(rel)$, then, upon termination, E-HSF returns a solution for $Clauses$.

Theorem (Progress of refinement)

E-HSF does not consider any error derivation(counter-example) more than once.

Soundness and Progress

Theorem (Soundness)

If HSF is sound, i.e., it returns solutions for given sets of clauses, and if $\textsc{GrdT}(grd)(v) \rightarrow \exists w : \textsc{RelT}(rel)(v, w)$ holds for each $grd \in Grds$ and $rel \in Rels$ such that $Parent(grd) = Parent(rel)$, then, upon termination, E-HSF returns a solution for $Clauses$.

Theorem (Progress of refinement)

E-HSF does not consider any error derivation(counter-example) more than once.

Implementation and Application

- Implementation based on HSF and the Z3 solver.
- Applied to verification of CTL properties.
- Input transition system described using Prolog facts:
  - $init(v)$, and
  - $next(v, v')$.
- CTL propery to be proved or disproved as forall-exists Horn clauses.
- ... like the example.

Implementation and Application

- Implementation based on HSF and the Z3 solver.
- Applied to verification of CTL properties.
- Input transition system described using Prolog facts:
  - $init(v)$, and
  - $next(v, v')$.
- CTL propery to be proved or disproved as forall-exists Horn clauses.
- ... like the example.

Implementation and Application

- Implementation based on HSF and the Z3 solver.
- Applied to verification of CTL properties.
- Input transition system described using Prolog facts:
  - $init(v)$, and
  - $next(v, v')$.
- CTL propery to be proved or disproved as forall-exists Horn clauses.
- ... like the example.

Implementation and Application

- Implementation based on HSF and the Z3 solver.
- Applied to verification of CTL properties.
- Input transition system described using Prolog facts:
  - $init(v)$, and
  - $next(v, v')$.
- CTL propery to be proved or disproved as forall-exists Horn clauses.
- ... like the example.

Experiment

- On industrial examples reported in [1].
- For a program and CTL property ϕ, two verification tasks:
  - prove ϕ, and
  - prove $\neg\phi$.
- For our examples, linear templates are sufficiently expressive.
  - $\textsc{RelT}(next)(v,v')$=$(next(v,v') \wedge w' = Tv + t \wedge Gv \leq g)$, and
  - $\textsc{GrdT}(next)(v,v')$=$(Gv \leq g \wedge \exists v' : next(v,v'))$, where w is a subset of v that is left unconstrained by $next(v,v')$.
- Linear ranking functions for dealing with well-foundedness:
  - $\textsc{DecreaseT}(v)$=$Rv \geq r$, and
  - $\textsc{BoundT}(v,v')$=$Rv' \leq Rv - 1$.

[1] Reasoning about Nondeterminism in Programs, Byron Cook, Eric Koskinen

Experiment

- On industrial examples reported in [1].
- For a program and CTL property ϕ, two verification tasks:
  - prove ϕ, and
  - prove $\neg\phi$.
- For our examples, linear templates are sufficiently expressive.
  - $\textsc{RelT}(\mathit{next})(v, v')$=$(\mathit{next}(v, v') \wedge w' = Tv + t \wedge Gv \leq g)$, and
  - $\textsc{GrdT}(\mathit{next})(v, v')$=$(Gv \leq g \wedge \exists v' : \mathit{next}(v, v'))$, where w is a subset of v that is left unconstrained by $\mathit{next}(v, v')$.
- Linear ranking functions for dealing with well-foundedness:
  - $\textsc{DecreaseT}(v)$=$Rv \geq r$, and
  - $\textsc{BoundT}(v, v')$=$Rv' \leq Rv - 1$.

[1] Reasoning about Nondeterminism in Programs, Byron Cook, Eric Koskinen

Experiment

- On industrial examples reported in [1].
- For a program and CTL property ϕ, two verification tasks:
  - prove ϕ, and
  - prove $\neg\phi$.
- For our examples, linear templates are sufficiently expressive.
  - $\textsc{RelT}(\mathit{next})(v, v')$=$(\mathit{next}(v, v') \wedge w' = Tv + t \wedge Gv \leq g)$, and
  - $\textsc{GrdT}(\mathit{next})(v, v')$=$(Gv \leq g \wedge \exists v' : \mathit{next}(v, v'))$, where w is a subset of v that is left unconstrained by $\mathit{next}(v, v')$.
- Linear ranking functions for dealing with well-foundedness:
  - $\textsc{DecreaseT}(v)$=$Rv \geq r$, and
  - $\textsc{BoundT}(v, v')$=$Rv' \leq Rv - 1$.

[1] Reasoning about Nondeterminism in Programs, Byron Cook, Eric Koskinen

Experiment

- On industrial examples reported in [1].
- For a program and CTL property ϕ, two verification tasks:
  - prove ϕ, and
  - prove $\neg\phi$.
- For our examples, linear templates are sufficiently expressive.
  - $\textsc{RelT}(\mathit{next})(v, v')$=$(\mathit{next}(v, v') \land w' = Tv + t \land Gv \leq g)$, and
  - $\textsc{GrdT}(\mathit{next})(v, v')$=$(Gv \leq g \land \exists v' : \mathit{next}(v, v'))$, where w is a subset of v that is left unconstrained by $\mathit{next}(v, v')$.
- Linear ranking functions for dealing with well-foundedness:
  - $\textsc{DecreaseT}(v)$=$Rv \geq r$, and
  - $\textsc{BoundT}(v, v')$=$Rv' \leq Rv - 1$.

[1] Reasoning about Nondeterminism in Programs, Byron Cook, Eric Koskinen

Results

Program		Property ϕ	$\models_{CTL} \phi$			$\models_{CTL} \neg\phi$		
			Result	Time	Name	Result	Time	Name
OS.frag 1	P1	$AG(a = 1 \rightarrow AF(r = 1))$	✓	1.2s	1	×	2.7s	29
	P2	$EF(a = 1 \wedge EG(r \neq 5))$	✓	0.6s	30	×	5.2s	2
	P3	$AG(a = 1 \rightarrow EF(r = 1))$	✓	4.8s	3	×	0.1s	31
	P4	$EF(a = 1 \wedge AG(r \neq 1))$	✓	0.6s	32	×	0.4s	4
OS.frag 2	P5	$AG(s = 1 \rightarrow AF(u = 1))$	✓	6.1s	5	×	0.2s	33
	P6	$EF(s = 1 \wedge EG(u \neq 1))$	✓	1.4s	34	×	3.6s	6
	P7	$AG(s = 1 \rightarrow EF(u = 1))$	✓	12.9s	7	×	0.2s	35
	P8	$EF(s = 1 \wedge AG(u \neq 1))$	✓	44.7s	36	×	3.8s	8
OS.frag 3	P9	$AG(a = 1 \rightarrow AF(r = 1))$	✓	51.3s	9	×	120.0s	37
	P10	$EF(a = 1 \wedge EG(r \neq 1))$	✓	132.0s	38	×	45.9s	10
	P11	$AG(a = 1 \rightarrow EF(r = 1))$	✓	67.6s	11	×	3.9s	39
	P12	$EF(a = 1 \wedge AG(r \neq 1))$	✓	67.9s	12	×	3.8s	40
OS.frag 4	P13	$AF(io = 1) \vee AF(ret = 1)$	✓	37m54s	13	T/O	-	41
	P14	$EG(io \neq 1) \wedge EG(ret \neq 1)$	T/O	-	42	×	136.6s	14
	P15	$EF(io = 1) \wedge EF(ret = 1)$	T/O	-	15	×	1.4s	43
	P16	$AG(io \neq 1) \vee AG(ret \neq 1)$	✓	0.1s	44	×	874.5s	16
OS.frag 5	P17	$AG(AF(w \geq 1))$	✓	3.0s	17	×	0.1s	45
	P18	$EF(EG(w < 1)$	✓	0.5s	46	×	3.5s	18
	P19	$AG(EF(w \geq 1))$	✓	3.3s	19	×	0.1s	47
	P20	$EF(AG(w < 1)$	✓	0.7s	48	×	0.1s	20
PGrSQL	P21	$AG(AF(w = 1)$	✓	2.8s	21	×	0.1s	49
	P22	$EF(EG(w \neq 1)$	✓	2.2s	50	×	5.0s	22
	P23	$AG(EF(w = 1)$	✓	4.5s	23	×	0.1s	51
	P24	$EF(AG(w \neq 1)$	✓	3.4s	52	×	0.7s	24
SW Upd	P25	$c > 5 \rightarrow AF(r > 5)$	✓	3.2s	25	×	0.1s	53
	P26	$c > 5 \wedge EG(r \leq 5)$	×	0.1s	54	×	1.3s	26
	P27	$c > 5 \rightarrow EF(r > 5)$	×	0.2s	27	×	0.1s	55
	P28	$c > 5 \wedge AG(r \leq 5)$	×	0.1s	56	×	0.3s	28

Summary

- Algorithm to solve existentially quantified horn clauses.
- Application to verification of CTL properties.
- Problems that require witness computation:
  - template-based program synthesis/repair.
  - deductive game solving.

Summary

- Algorithm to solve existentially quantified horn clauses.
- Application to verification of CTL properties.
- Problems that require witness computation:
  - template-based program synthesis/repair.
  - deductive game solving.

Summary

- Algorithm to solve existentially quantified horn clauses.
- Application to verification of CTL properties.
- Problems that require witness computation:
  - template-based program synthesis/repair.
  - deductive game solving.

Summary

- Algorithm to solve existentially quantified horn clauses.
- Application to verification of CTL properties.
- Problems that require witness computation:
  - template-based program synthesis/repair.
  - deductive game solving.

Summary

- Algorithm to solve existentially quantified horn clauses.
- Application to verification of CTL properties.
- Problems that require witness computation:
    - template-based program synthesis/repair.
    - deductive game solving.