Skolem Functions for Factored Formulas

A. K. John[1], S. Shah[1], S. Chakraborty[1], **A. Trivedi**[1,2], S. Akshay[1]

IIT Bombay[1] and CU Boulder[2]

September 29, 2015

Collaborators

Ajith K. John

Shetal Shah

Supratik Chakraborty

Akshay S.

Skolem functions and their applications

CEGAR for Skolem functions

Experimental Results

Skolem functions and their applications

CEGAR for Skolem functions

Experimental Results

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x \; F \equiv F[x \mapsto \psi].$$

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x\ F \equiv F[x \mapsto \psi].$$

Example 1. Find a Skolem function for x in formula $F(x, y_1, y_2) = x \wedge y_1 \wedge y_2$.

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x\ F \equiv F[x \mapsto \psi].$$

Example 1. Find a Skolem function for x in formula $F(x, y_1, y_2) = x \wedge y_1 \wedge y_2$.

1. Note that

$$\exists x.(x \wedge y_1 \wedge y_2) \equiv (1 \wedge y_1 \wedge y_2) \vee (0 \wedge y_1 \wedge y_2)$$

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x \; F \equiv F[x \mapsto \psi].$$

Example 1. Find a Skolem function for x in formula $F(x, y_1, y_2) = x \wedge y_1 \wedge y_2$.

1. Note that

$$\exists x.(x \wedge y_1 \wedge y_2) \equiv (1 \wedge y_1 \wedge y_2) \vee (0 \wedge y_1 \wedge y_2) \equiv y_1 \wedge y_2$$

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x \; F \equiv F[x \mapsto \psi].$$

Example 1. Find a Skolem function for x in formula $F(x, y_1, y_2) = x \wedge y_1 \wedge y_2$.

1. Note that

$$\exists x.(x \wedge y_1 \wedge y_2) \equiv (1 \wedge y_1 \wedge y_2) \vee (0 \wedge y_1 \wedge y_2) \equiv y_1 \wedge y_2$$

2. Hence a Skolem function for x if $\psi(y_1, y_2) = 1$.

Skolem Functions

Definition (Skolem functions)

Given a propositional function $F(x, Y)$, a Skolem function for $x \in X$ in $F(x, Y)$ is a function $\psi(Y)$ such that

$$\exists x\ F \equiv F[x \mapsto \psi].$$

Example 1. Find a Skolem function for x in formula $F(x, y_1, y_2) = x \wedge y_1 \wedge y_2$.

1. Note that

$$\exists x.(x \wedge y_1 \wedge y_2) \equiv (1 \wedge y_1 \wedge y_2) \vee (0 \wedge y_1 \wedge y_2) \equiv y_1 \wedge y_2$$

2. Hence a Skolem function for x if $\psi(y_1, y_2) = 1$.
3. Are Skolem functions unique?

Skolem Functions

Definition (Skolem function vector)

Given a propositional function $F(X, Y)$, a Skolem function vector for $X = (x_1, \ldots x_n)$ in F is a vector of functions $\mathbf{\Psi} = (\psi_1, \ldots, \psi_n)$ such that

$$\exists x_1 \ldots x_n \; F \equiv (\cdots (F[x_1 \mapsto \psi_1]) \cdots [x_n \mapsto \psi_n]).$$

Skolem Functions

Definition (Skolem function vector)

Given a propositional function $F(X, Y)$, a Skolem function vector for $X = (x_1, \ldots x_n)$ in F is a vector of functions $\mathbf{\Psi} = (\psi_1, \ldots, \psi_n)$ such that

$$\exists x_1 \ldots x_n \; F \equiv (\cdots(F[x_1 \mapsto \psi_1]) \cdots [x_n \mapsto \psi_n]).$$

Algorithm 1. $\textsc{SkolemGeneration}(F(x_1, \ldots, x_n, Y))$.

1. Input: Propositional formula $F(x_1, x_2, \ldots, x_n, Y)$
2. Output: Skolem function set $\mathbf{\Psi} = \{\psi_1, \ldots, \psi_n\}$
3. For $i = 1$ to n
   3.1 $\psi_i = \textsc{SkolemFun}(F, x_i)$
   3.2 $F = \exists x_i F = F[x_i \mapsto \psi_i]$
4. Return $\{\psi_1, \psi_2, \ldots, \psi_n\}$.

Skolem Functions

Definition (Skolem function vector)

Given a propositional function $F(X, Y)$, a Skolem function vector for $X = (x_1, \ldots x_n)$ in F is a vector of functions $\mathbf{\Psi} = (\psi_1, \ldots, \psi_n)$ such that

$$\exists x_1 \ldots x_n \; F \equiv (\cdots (F[x_1 \mapsto \psi_1]) \cdots [x_n \mapsto \psi_n]).$$

Algorithm 1. $\textsc{SkolemGeneration}(F(x_1, \ldots, x_n, Y))$.

1. Input: Propositional formula $F(x_1, x_2, \ldots, x_n, Y)$
2. Output: Skolem function set $\mathbf{\Psi} = \{\psi_1, \ldots, \psi_n\}$
3. For $i = 1$ to n
   - 3.1 $\psi_i = \textsc{SkolemFun}(F, x_i)$
   - 3.2 $F = \exists x_i F = F[x_i \mapsto \psi_i]$
4. Return $\{\psi_1, \psi_2, \ldots, \psi_n\}$.

Example 2. Find a Skolem function vector (ψ_1, ψ_2) for (x_1, x_2) in formula $F(x_1, x_2, y_1, y_2) = x_1 \wedge \overline{x_2} \wedge y_1 \wedge y_2$.

Our focus and applications

Skolem Generation for Factored Formulas

1. Given a propositional function $F(X, Y)$ as conjunction of factors,

$$f^1(X_1, Y_1) \wedge f^2(X_2, Y_2) \cdots \wedge f^r(X_r, Y_r)$$

find Skolem functions for variables in X.

Our focus and applications

Skolem Generation for Factored Formulas

1. Given a propositional function $F(X, Y)$ as conjunction of factors,

$$f^1(X_1, Y_1) \wedge f^2(X_2, Y_2) \cdots \wedge f^r(X_r, Y_r)$$

   find Skolem functions for variables in X.
2. We make no assumption about validity of $\exists X F(X, Y)$.

Our focus and applications

Skolem Generation for Factored Formulas

1. Given a propositional function $F(X, Y)$ as conjunction of factors,

$$f^1(X_1, Y_1) \wedge f^2(X_2, Y_2) \cdots \wedge f^r(X_r, Y_r)$$

   find Skolem functions for variables in X.
2. We make no assumption about validity of $\exists X F(X, Y)$.

Skolem function generation is a key verification/synthesis problem due to its applications in:

1. Quantifier elimination, of course
2. Generating certificates in Quantified Boolean Formula (QBF) solving
3. Program synthesis
   - Combinatorial Sketching for Finite Programs [SLTB$^+$06]
   - Complete Functional Synthesis [KMPS10]
4. Finding winning/optimal strategies in certain two-player games [AMN05]

Our focus and applications

Skolem Generation for Factored Formulas

1. Given a propositional function $F(X,Y)$ as conjunction of factors,

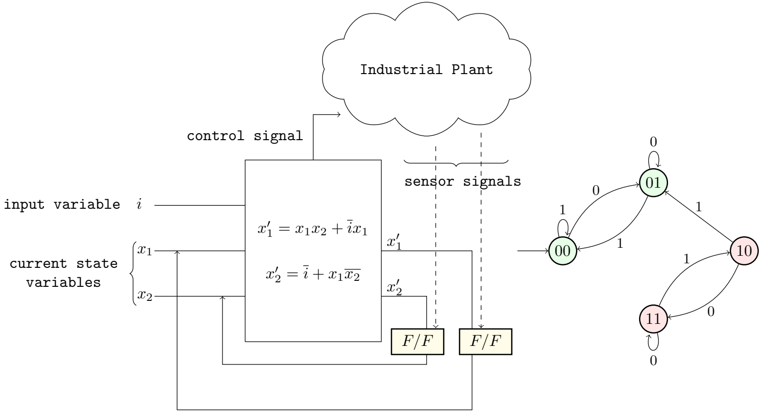
$$f^1(X_1,Y_1) \wedge f^2(X_2,Y_2) \cdots \wedge f^r(X_r,Y_r)$$

   find Skolem functions for variables in X.
2. We make no assumption about validity of $\exists X F(X,Y)$.

Skolem function generation is a key verification/synthesis problem due to its applications in:

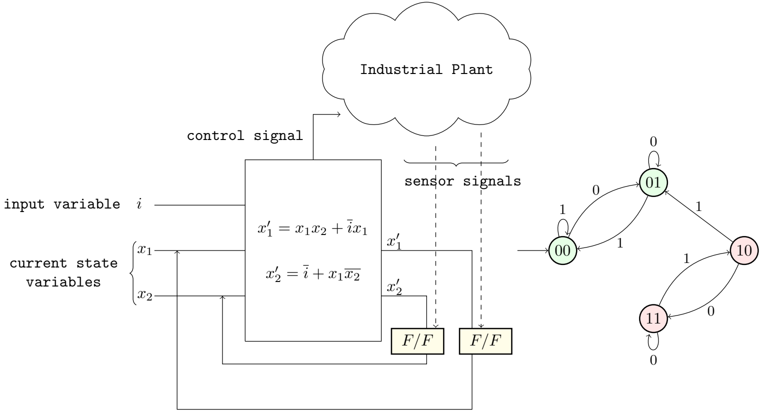
1. Quantifier elimination, of course
2. Generating certificates in Quantified Boolean Formula (QBF) solving
3. Program synthesis
   - Combinatorial Sketching for Finite Programs [SLTB+06]
   - Complete Functional Synthesis [KMPS10]
4. Finding winning/optimal strategies in certain two-player games [AMN05]

Plant Control Problem



1. How should the primary input of the controller be driven so that the controller transitions to a desirable state in one step, whenever possible?

Plant Control Problem

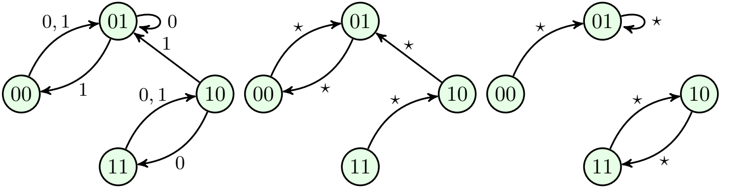


1. How should the primary input of the controller be driven so that the controller transitions to a desirable state in one step, whenever possible?
2. Solution: Find Skolem function for the primary input variable:

$$\exists x'_1 \exists x'_2 \left((x'_1 = x_1x_2 + \bar{i}x_1) \wedge (x'_2 = \bar{i} + x_1\overline{x_2}) \wedge \mathsf{Good}(x'_1, x'_2)\right)$$

Graph Decomposition Problem

$\textsc{NotCovered} = 1$ $\quad$ $\textsc{NotCovered} = \begin{array}{l}(\neg y_1 \wedge y_2 \wedge \neg x)\\ \vee (y_1 \wedge \neg y_2 \wedge \neg x)\end{array}$



$y_1' = (\neg x \wedge y_1) \vee (y_1 \wedge y_2)$
$y_2' = \neg y_2 \vee (\neg x \wedge \neg y_1)$

❶

$y_1' = y_1 \wedge y_2$
$y_2' = \neg y_2$

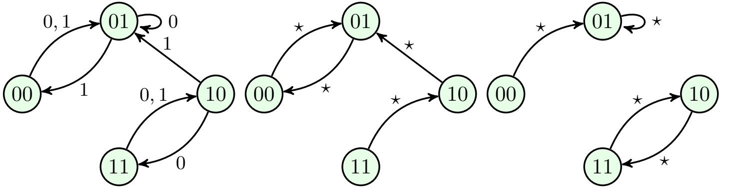
❷

$y_1' = y_1$
$y_2' = \neg y_2 \vee \neg y_1$

1. Compute a disjunctive decomposition of implicitly specified state transition graphs of sequential circuits [Tri03, TCP08].

Graph Decomposition Problem

$\textsc{NotCovered} = 1$ $\quad \textsc{NotCovered} = \begin{array}{l}(\neg y_1 \wedge y_2 \wedge \neg x) \\ \vee (y_1 \wedge \neg y_2 \wedge \neg x)\end{array}$



$y_1' = (\neg x \wedge y_1) \vee (y_1 \wedge y_2)$
$y_2' = \neg y_2 \vee (\neg x \wedge \neg y_1)$

❶ $y_1' = y_1 \wedge y_2$
$y_2' = \neg y_2$

❷ $y_1' = y_1$
$y_2' = \neg y_2 \vee \neg y_1$

1. Compute a disjunctive decomposition of implicitly specified state transition graphs of sequential circuits [Tri03, TCP08].
2. Solution: Find Skolem function for the input variables X in:

$$\wedge_i \textsc{NotCovered}_i(X, Y).$$

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   - succinct Skolem functions if there exists a short proof of validity.

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   – succinct Skolem functions if there exists a short proof of validity.
   – are not applicable when $\exists X F(X, Y)$ is not valid.

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X,Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X,Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS+07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X, Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS$^+$07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]
     1.5 Bloqqer tool [HSB14]

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X, Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS$^+$07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]
     1.5 Bloqqer tool [HSB14]
2. Generate Skolem functions matching a given template.
   - Template-based program verification and program synthesis by Srivastava, Gulwani, and Foster [SGF13]

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X, Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS$^+$07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]
     1.5 Bloqqer tool [HSB14]
2. Generate Skolem functions matching a given template.
   - Template-based program verification and program synthesis by Srivastava, Gulwani, and Foster [SGF13]
   - effective when the set of candidate Skolem functions is known and small

Existing Approaches

1. Extract Skolem function from the proof of validity of $\exists X F(X,Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X,Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS+07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]
     1.5 Bloqqer tool [HSB14]
2. Generate Skolem functions matching a given template.
   - Template-based program verification and program synthesis by Srivastava, Gulwani, and Foster [SGF13]
   - effective when the set of candidate Skolem functions is known and small
   - it is not always reasonable assumption

Existing Approaches

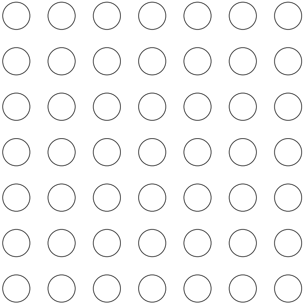
1. Extract Skolem function from the proof of validity of $\exists X F(X, Y)$
   - succinct Skolem functions if there exists a short proof of validity.
   - are not applicable when $\exists X F(X, Y)$ is not valid.
   - References:
     1.1 *sKizzo: a suite to evaluate and certify QBFs* by Benedetti [Ben05]
     1.2 *Resolution Proofs and Skolem functions* by Jiang and Balabanov [JB11]
     1.3 *A first step towards a Unified Proof Checkers for QBF* by Jussila et al. [JBS$^+$07]
     1.4 *Efficient extraction of Skolem functions from QRAT proofs* by Heule, Seidl, and Biere [HSB14]
     1.5 Bloqqer tool [HSB14]
2. Generate Skolem functions matching a given template.
   - Template-based program verification and program synthesis by Srivastava, Gulwani, and Foster [SGF13]
   - effective when the set of candidate Skolem functions is known and small
   - it is not always reasonable assumption
3. Composition based approaches
   - Quantifier elimination via functional composition by Jiang [Jia09]
   - Techniques in Symbolic model checking by Trivedi [Tri03]
   - Work well for small-sized formulas
   - Compositions cause formula blow up and memory out

Skolem functions and their applications

CEGAR for Skolem functions

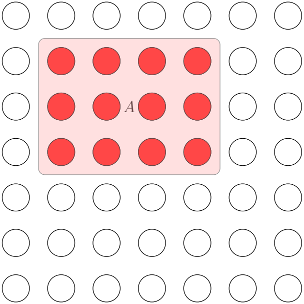
Experimental Results

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.



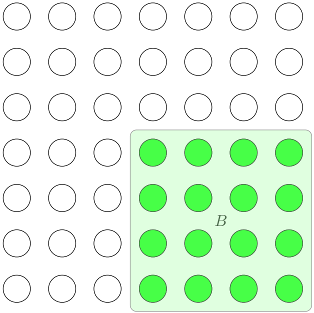
— Set of All valuations to Y.

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.



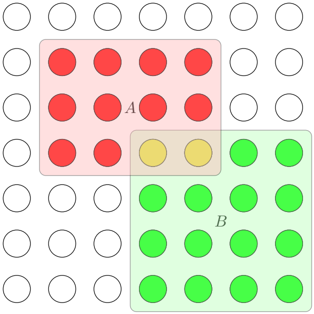
— $A(Y) =$ Can't set x to 1 to satisfy $F = \neg F(x, Y)[x \mapsto 1]$

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.



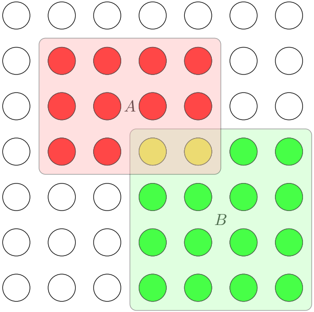
— $B(Y) =$ Can't set x to 0 to satisfy $F = \neg F(x, Y)[x \mapsto 0]$

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.



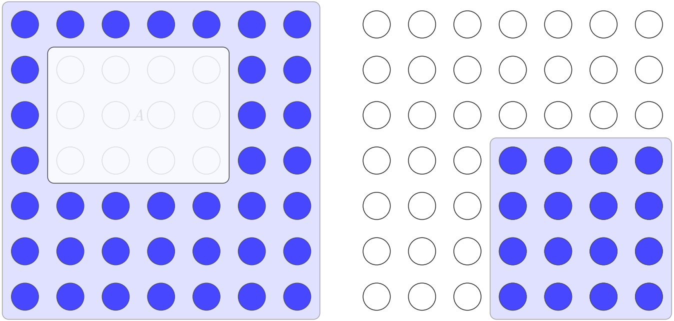
— $A(Y)$ = Can't set x to 1 to satisfy $F = \neg F(x, Y)[x \mapsto 1]$
— $B(Y)$ = Can't set x to 0 to satisfy $F = \neg F(x, Y)[x \mapsto 0]$

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.



— $A(Y) =$ Can't set x to 1 to satisfy $F = \neg F(x, Y)[x \mapsto 1]$
— $B(Y) =$ Can't set x to 0 to satisfy $F = \neg F(x, Y)[x \mapsto 0]$
— A Skolem function for x in F is any Interpolant of $(B \setminus A)$ and $(A \setminus B)$

Find $\psi(Y)$ such that $\exists x F(x, Y) \equiv F(\psi(Y), Y)$.

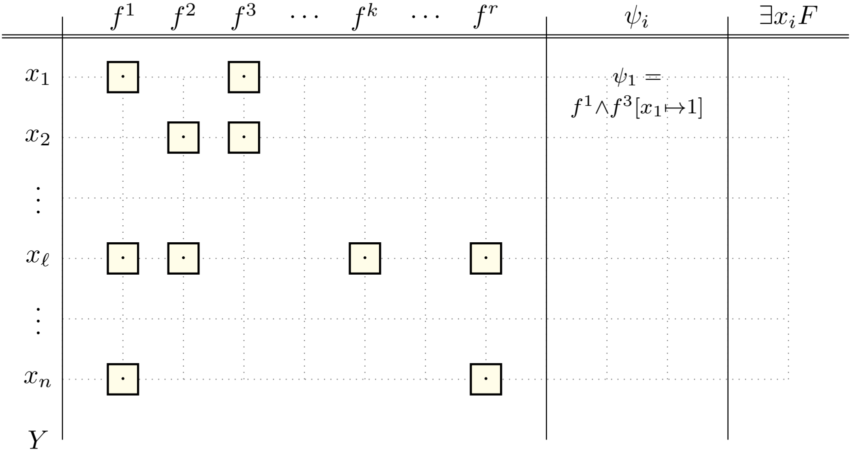


- $A(Y) =$ Can't set x to 1 to satisfy $F = \neg F(x, Y)[x \mapsto 1]$
- $B(Y) =$ Can't set x to 0 to satisfy $F = \neg F(x, Y)[x \mapsto 0]$
- A Skolem function for x in F is any Interpolant of $(B \setminus A)$ and $(A \setminus B)$
- E.g. $\neg A = F(x, Y)[x \mapsto 1] = F(1, Y)$
- and $B = \neg F(x, Y)[x \mapsto 0] = \neg F(0, Y)$.

Monolithic Skolem Generation [Jia09, Tri03]

	f^1	f^2	f^3	$\cdots$	f^k	$\cdots$	f^r	ψ_i	$\exists x_i F$
x_1	$\boxed{\cdot}$		$\boxed{\cdot}$						
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$						
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

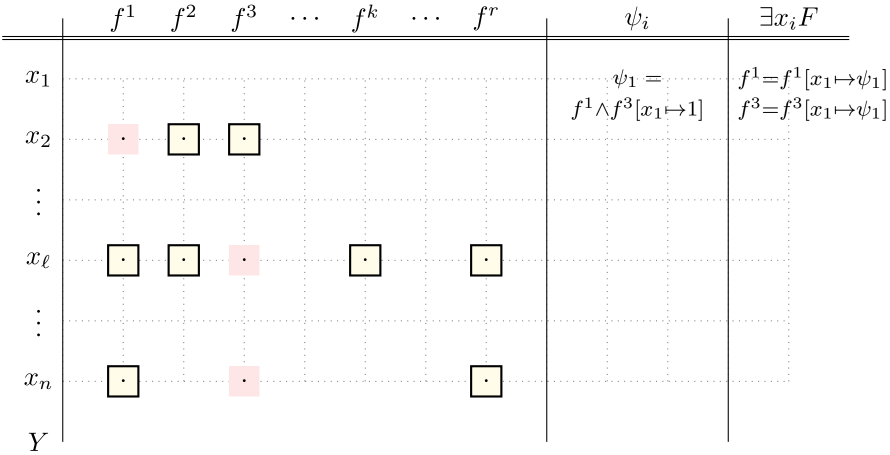
Monolithic Skolem Generation [Jia09, Tri03]



Monolithic Skolem Generation [Jia09, Tri03]

	f^1	f^2	f^3	$\dots$	f^k	$\dots$	f^r	ψ_i	$\exists x_i F$
x_1	$\boxed{\cdot}$		$\boxed{\cdot}$					$\psi_1 =$ $f^1 \wedge f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1]$ $f^3 = f^3[x_1 \mapsto \psi_1]$
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$						
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

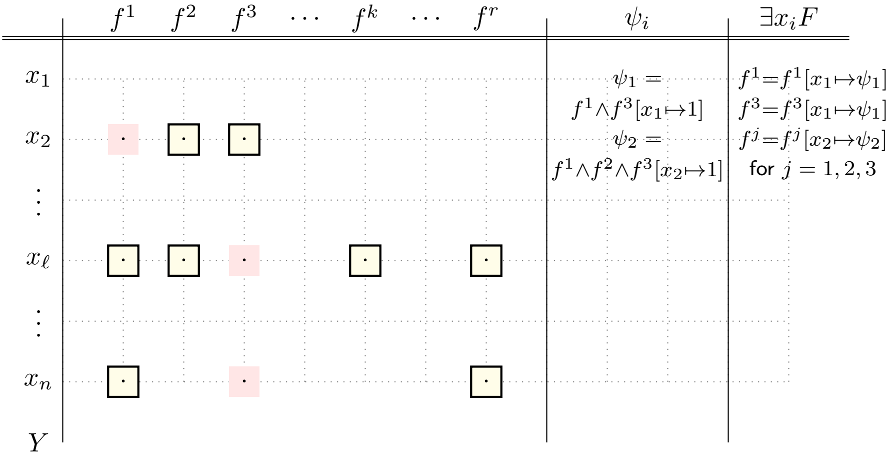
Monolithic Skolem Generation [Jia09, Tri03]



Monolithic Skolem Generation [Jia09, Tri03]

	f^1	f^2	f^3	$\dots$	f^k	$\dots$	f^r	ψ_i	$\exists x_i F$
x_1								$\psi_1 = f^1 \wedge f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1]$ $f^3 = f^3[x_1 \mapsto \psi_1]$
x_2	$\cdot$	$\boxed{\cdot}$	$\boxed{\cdot}$					$\psi_2 = f^1 \wedge f^2 \wedge f^3[x_2 \mapsto 1]$	
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$	$\cdot$		$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$		$\cdot$				$\boxed{\cdot}$		
Y									

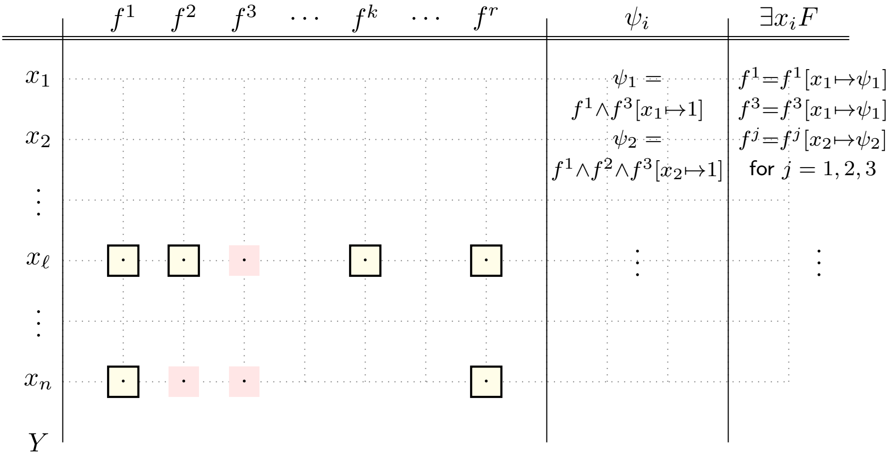
Monolithic Skolem Generation [Jia09, Tri03]



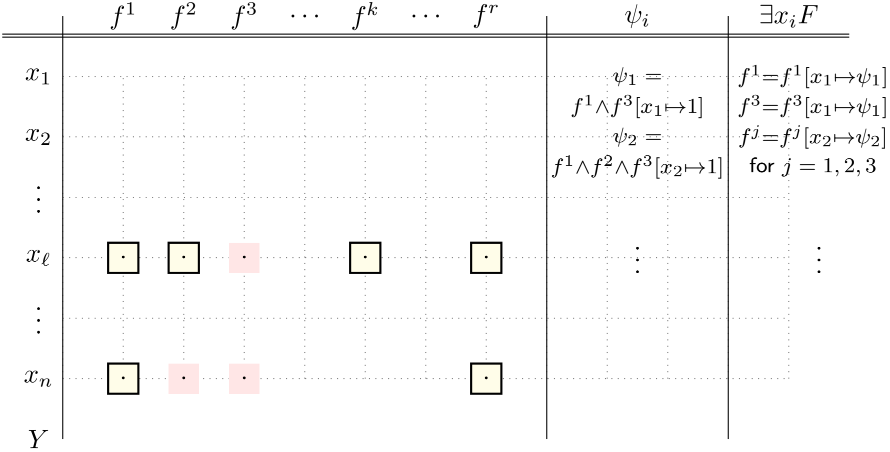
Monolithic Skolem Generation [Jia09, Tri03]

	f^1	f^2	f^3	$\dots$	f^k	$\dots$	f^r	ψ_i	$\exists x_i F$
x_1								$\psi_1 = f^1 \wedge f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1]$ $f^3 = f^3[x_1 \mapsto \psi_1]$
x_2								$\psi_2 = f^1 \wedge f^2 \wedge f^3[x_2 \mapsto 1]$	$f^j = f^j[x_2 \mapsto \psi_2]$ for $j = 1, 2, 3$
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$	$\cdot$		$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$	$\cdot$	$\cdot$				$\boxed{\cdot}$		
Y									

Monolithic Skolem Generation [Jia09, Tri03]

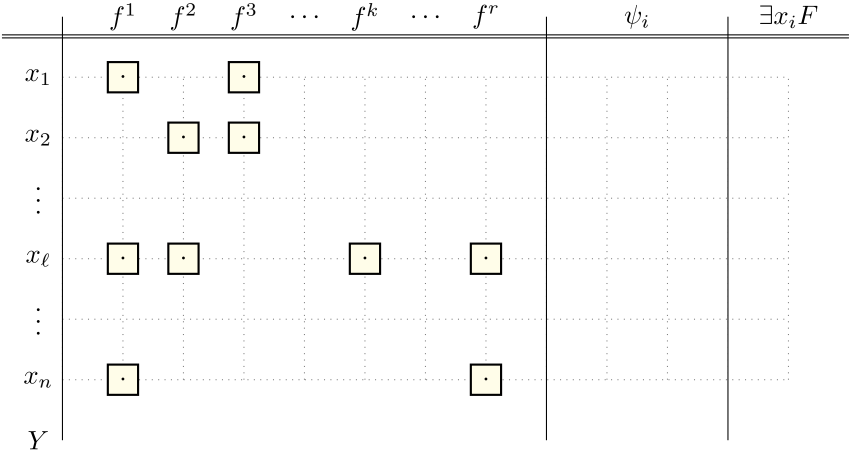


Monolithic Skolem Generation [Jia09, Tri03]

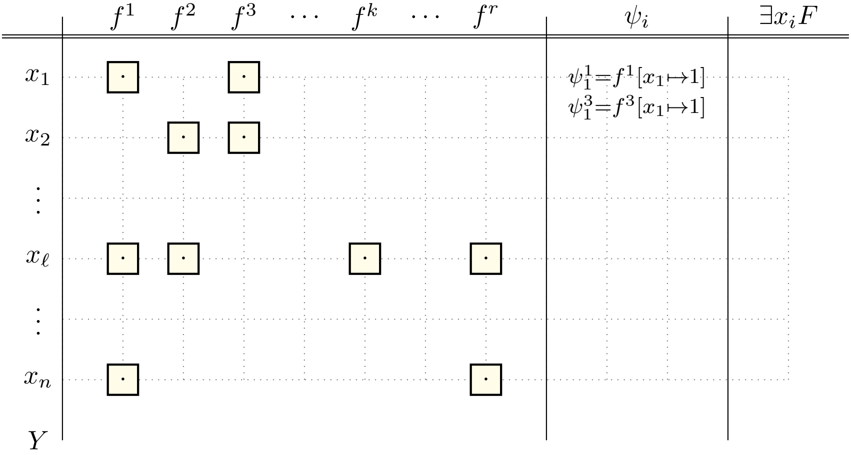


Blowup in sizes of factors after each quantifier elimination

How to avoid such blowup



How to avoid such blowup



How to avoid such blowup

	f^1	f^2	f^3	$\dots$	f^k	$\dots$	f^r	ψ_i	$\exists x_i F$
x_1	$\boxed{\cdot}$		$\boxed{\cdot}$					$\psi_1^1 = f^1[x_1 \mapsto 1]$ $\psi_1^3 = f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1^1]$ $f^3 = f^3[x_1 \mapsto \psi_1^3]$
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$						
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

How to avoid such blowup

	f^1	f^2	f^3	$\dots$	f^k	$\dots$	f^r	ψ_i	$\exists x_i F$
x_1								$\psi_1^1 = f^1[x_1 \mapsto 1]$ $\psi_1^3 = f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1^1]$ $f^3 = f^3[x_1 \mapsto \psi_1^3]$
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$						
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

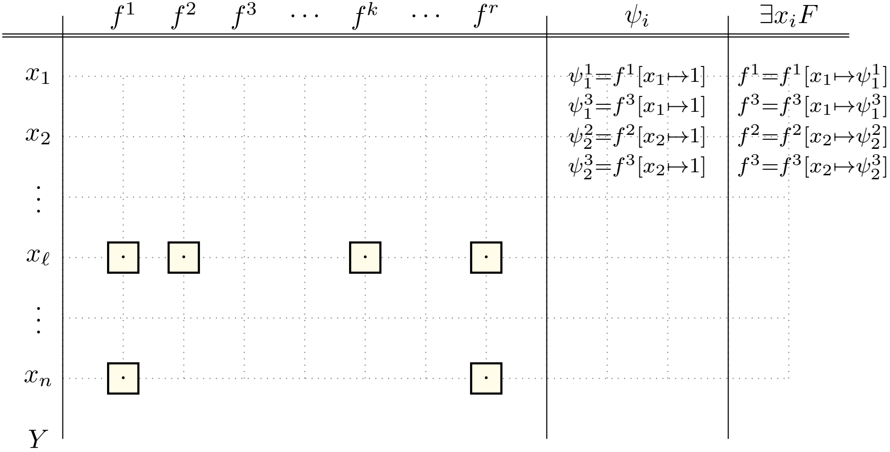
How to avoid such blowup

	f^1	f^2	f^3	$\cdots$	f^k	$\cdots$	f^r	ψ_i	$\exists x_i F$
x_1								$\psi_1^1 = f^1[x_1 \mapsto 1]$ $\psi_1^3 = f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1^1]$ $f^3 = f^3[x_1 \mapsto \psi_1^3]$
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$					$\psi_2^2 = f^2[x_2 \mapsto 1]$ $\psi_2^3 = f^3[x_2 \mapsto 1]$	
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

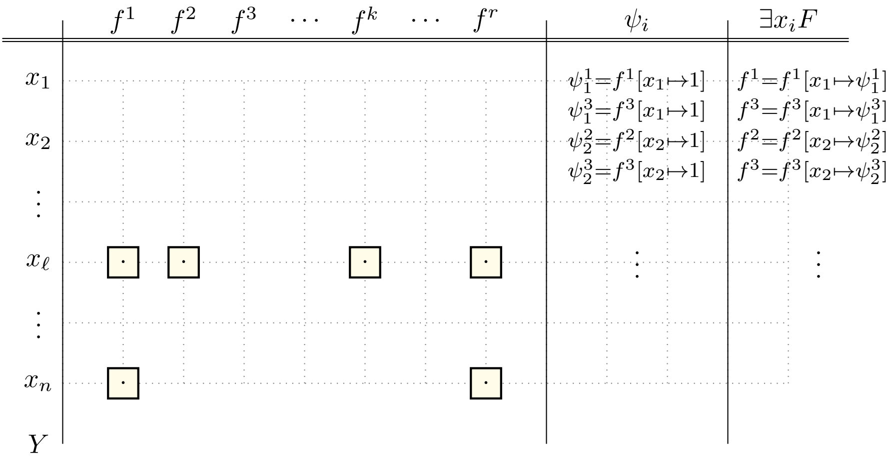
How to avoid such blowup

	f^1	f^2	f^3	$\cdots$	f^k	$\cdots$	f^r	ψ_i	$\exists x_i F$
x_1								$\psi_1^1 = f^1[x_1 \mapsto 1]$ $\psi_1^3 = f^3[x_1 \mapsto 1]$	$f^1 = f^1[x_1 \mapsto \psi_1^1]$ $f^3 = f^3[x_1 \mapsto \psi_1^3]$
x_2		$\boxed{\cdot}$	$\boxed{\cdot}$					$\psi_2^2 = f^2[x_2 \mapsto 1]$ $\psi_2^3 = f^3[x_2 \mapsto 1]$	$f^2 = f^2[x_2 \mapsto \psi_2^2]$ $f^3 = f^3[x_2 \mapsto \psi_2^3]$
$\vdots$									
x_ℓ	$\boxed{\cdot}$	$\boxed{\cdot}$			$\boxed{\cdot}$		$\boxed{\cdot}$		
$\vdots$									
x_n	$\boxed{\cdot}$						$\boxed{\cdot}$		
Y									

How to avoid such blowup

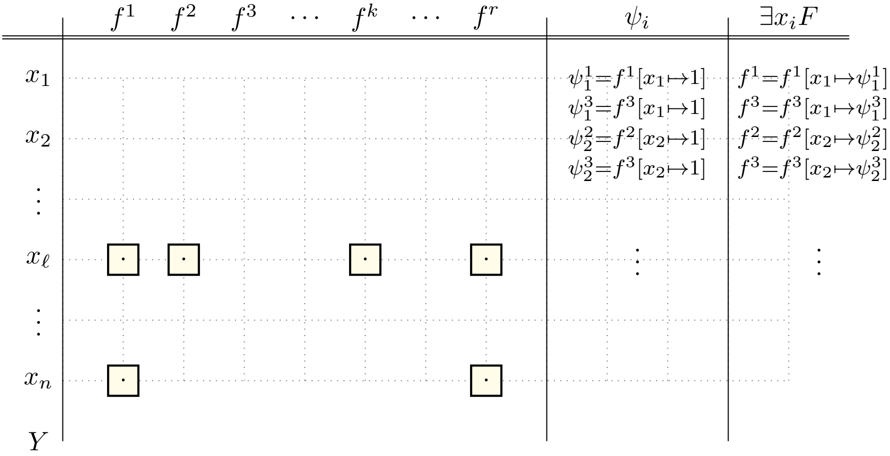


How to avoid such blowup



— Skolem functions are in factored form: $\psi_i = \wedge \psi_i^k$

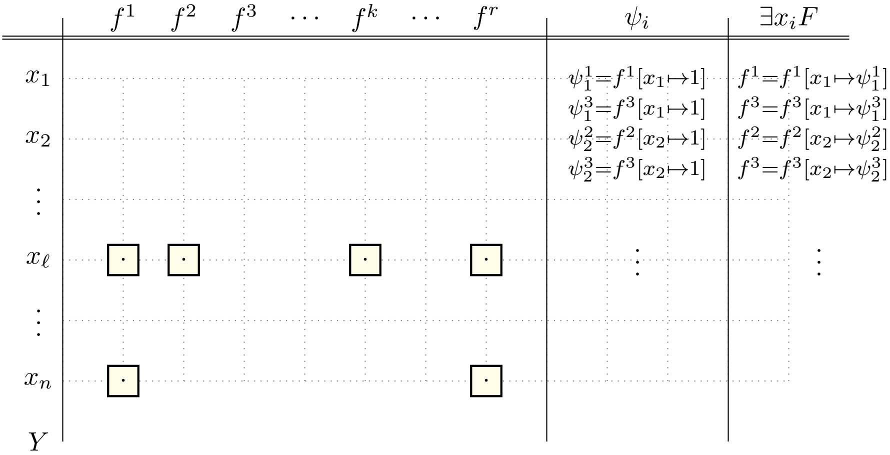
How to avoid such blowup



— Skolem functions are in factored form: $\psi_i = \wedge \psi_i^k$

— Problem: $\exists x(f^1 \wedge f^2) \neq (\exists x f^1) \wedge (\exists x f^2)$

How to avoid such blowup



— Skolem functions are in factored form: $\psi_i = \wedge \psi_i^k$

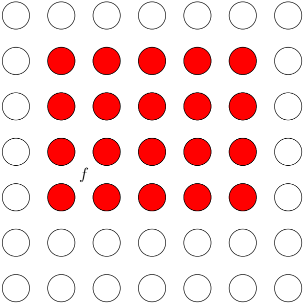
— Problem: $\exists x (f^1 \wedge f^2) \neq (\exists x f^1) \wedge (\exists x f^2)$

— Abstraction of $\exists x_i F$ and of ψ_i

Counterexample-Guided Abstraction Refinement

– Given propositional functions $f(X)$ and $g(X)$, we say that f is an abstraction of g if

$$g \implies f$$

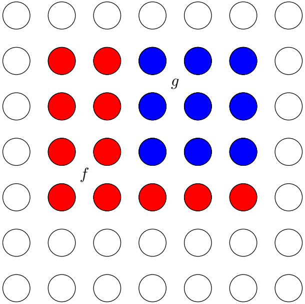


– We will also say that g is a refinement of g.

Counterexample-Guided Abstraction Refinement

- Given propositional functions $f(X)$ and $g(X)$, we say that f is an abstraction of g if

$$g \implies f$$



- We will also say that g is a refinement of g.

CEGAR: Contd

- An abstract Skolem function is a function that is an abstraction of a proper Skolem function.
- An abstraction Skolem function may not be a proper Skolem function.
- Given a formula $F(x_1, \ldots, x_n, Y)$ and functions $\Psi = \{\psi_1, \psi_2, \ldots, \psi_n\}$ how do we check if Ψ is a proper Skolem vector?

CEGAR: Contd

- An abstract Skolem function is a function that is an abstraction of a proper Skolem function.
- An abstraction Skolem function may not be a proper Skolem function.
- Given a formula $F(x_1, \ldots, x_n, Y)$ and functions $\Psi = \{\psi_1, \psi_2, \ldots, \psi_n\}$ how do we check if Ψ is a proper Skolem vector?
- Simply check if the following formula $\textsc{IsSkolem}(F, \Psi)$ is satisfiable:

$$F(X', Y) \wedge_{i=1}^{n} (x_i \iff \psi_i) \wedge \neg F(X, Y)$$

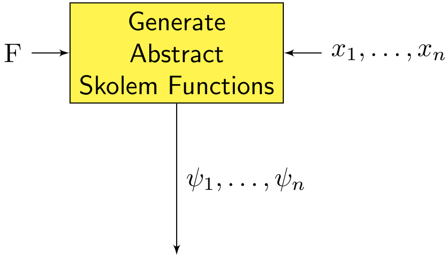
CEGAR: Contd

- An abstract Skolem function is a function that is an abstraction of a proper Skolem function.
- An abstraction Skolem function may not be a proper Skolem function.
- Given a formula $F(x_1, \ldots, x_n, Y)$ and functions $\Psi = \{\psi_1, \psi_2, \ldots, \psi_n\}$ how do we check if Ψ is a proper Skolem vector?
- Simply check if the following formula $\textsc{IsSkolem}(F, \Psi)$ is satisfiable:

$$F(X', Y) \wedge_{i=1}^{n} (x_i \iff \psi_i) \wedge \neg F(X, Y)$$

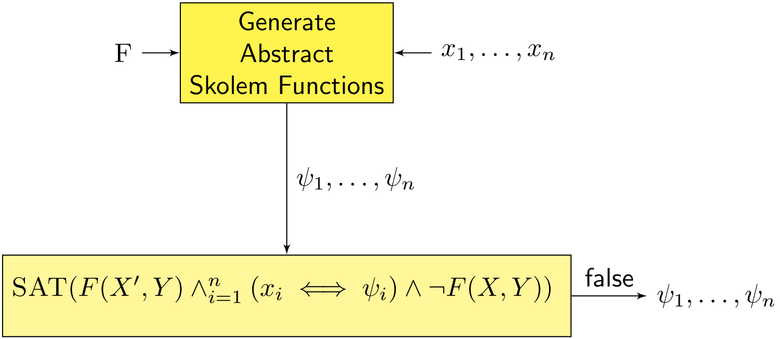
- If this formula is unsatisfiable, then $\psi_1, \ldots, \psi_n$ are proper Skolem functions for $x_1, \ldots, x_n$
- Otherwise, satisfying assignment helps us to refine Skolem function.

CEGAR: Contd.

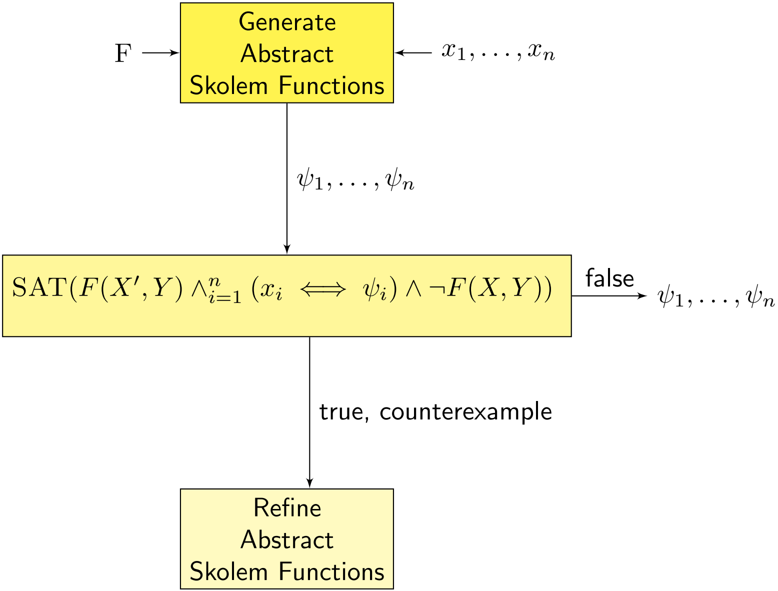


F
Generate Abstract Skolem Functions
$x_1, \ldots, x_n$
$\psi_1, \ldots, \psi_n$

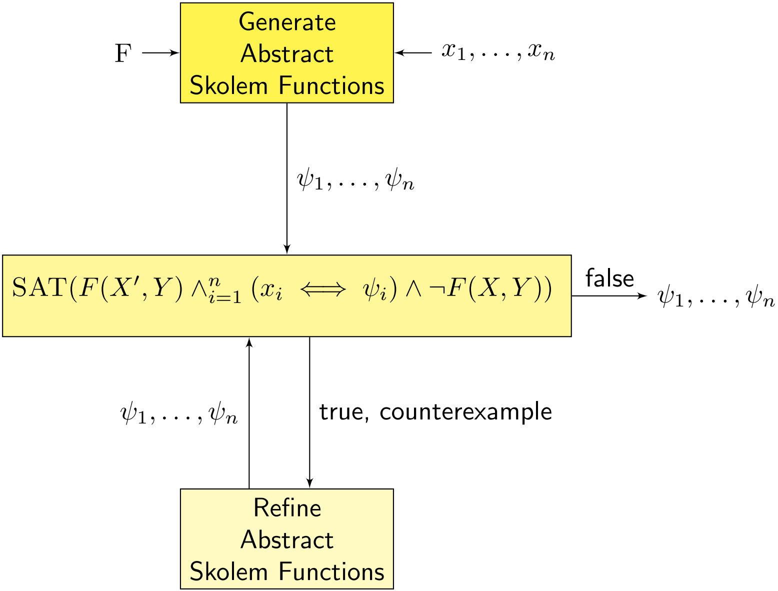
CEGAR: Contd.


F
Generate
Abstract
Skolem Functions
$x_1, \dots, x_n$
$\psi_1, \dots, \psi_n$
$\mathrm{SAT}(F(X', Y) \wedge_{i=1}^{n} (x_i \iff \psi_i) \wedge \neg F(X, Y))$
false
$\psi_1, \dots, \psi_n$

CEGAR: Contd.

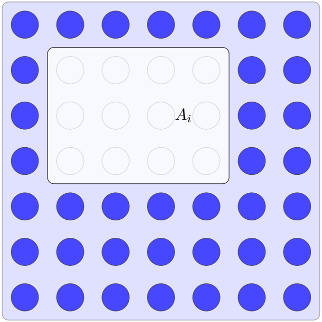

Generate Abstract Skolem Functions
F
$x_1, \ldots, x_n$
$\psi_1, \ldots, \psi_n$
$\mathrm{SAT}(F(X', Y) \wedge_{i=1}^{n} (x_i \iff \psi_i) \wedge \neg F(X, Y))$
false
$\psi_1, \ldots, \psi_n$
true, counterexample
Refine Abstract Skolem Functions

CEGAR: Contd.



F
Generate Abstract Skolem Functions
$x_1, \ldots, x_n$
$\psi_1, \ldots, \psi_n$
$\mathrm{SAT}(F(X', Y) \wedge_{i=1}^{n} (x_i \iff \psi_i) \wedge \neg F(X, Y))$
false
$\psi_1, \ldots, \psi_n$
$\psi_1, \ldots, \psi_n$
true, counterexample
Refine Abstract Skolem Functions

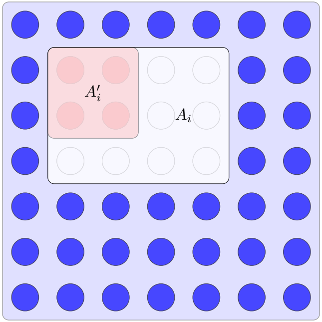
Abstraction and Refinement



$A_i = \neg\exists x_1 \ldots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

1. Ideally when we need to compute Skolem function for x_i we need to have access to $F_i = \exists x_1, \ldots x_{i-1} F$.
2. Then, to compute Skolem function we can compute the set $A_i = \neg F_i[x \mapsto 1]$ and a proper Skolem function would be $\neg A$.

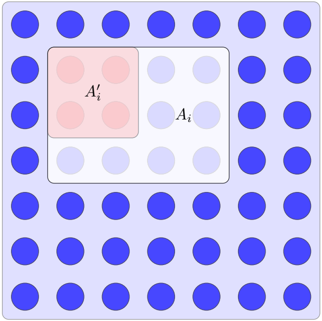
Abstraction and Refinement



$A_i = \neg \exists x_1 \ldots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

1. Ideally when we need to compute Skolem function for x_i we need to have access to $F_i = \exists x_1, \ldots x_{i-1} F$.
2. Then, to compute Skolem function we can compute the set $A_i = \neg F_i[x \mapsto 1]$ and a proper Skolem function would be $\neg A$.
3. However, due to factorwise quantification, we only know an abstraction F_i' of F_i.
4. Hence, the set A_i' computed using F_i' would be a refinement of the proper A_i.

Abstraction and Refinement

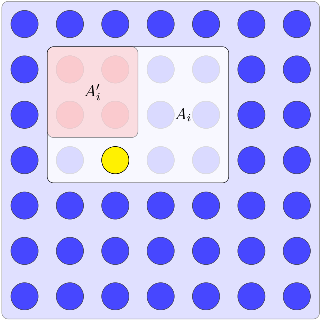


$A_i = \neg\exists x_1 \dots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

$A_i' = \neg F_i'[x_i \mapsto 1]$ and $\psi_i' = \neg A_i'$

1. Ideally when we need to compute Skolem function for x_i we need to have access to $F_i = \exists x_1, \dots x_{i-1} F$.
2. Then, to compute Skolem function we can compute the set $A_i = \neg F_i[x \mapsto 1]$ and a proper Skolem function would be $\neg A$.
3. However, due to factorwise quantification, we only know an abstraction F_i' of F_i.
4. Hence, the set A_i' computed using F_i' would be a refinement of the proper A_i.
5. This implies that the Skolem function computed as $\neg A_i'$ will be an abstract Skolem function.

Abstraction and Refinement

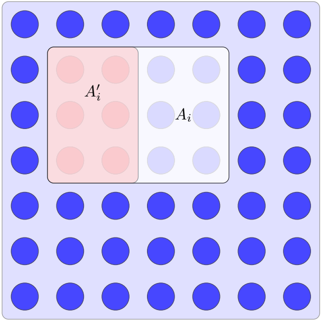


$A_i = \neg\exists x_1 \dots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

$A_i' = \neg F_i'[x_i \mapsto 1]$ and $\psi_i' = \neg A_i'$

1. When we check if $\psi_1, \dots, \psi_n$ are proper Skolem functions, and we get a counterexample, it pinpoints a valuation for which abstract Skolem function returns 1 when it should not.

Abstraction and Refinement

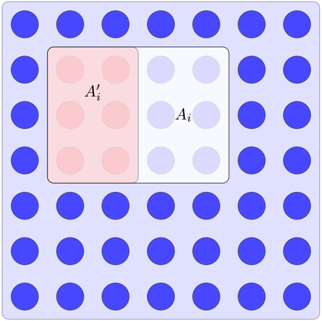


$A_i = \neg\exists x_1 \ldots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

$A'_i = \neg F'_i[x_i \mapsto 1]$ and $\psi'_i = \neg A'_i$

1. When we check if $\psi_1, \ldots, \psi_n$ are proper Skolem functions, and we get a counterexample, it pinpoints a valuation for which abstract Skolem function returns 1 when it should not.
2. We refine Skolem function candidates for $\psi_{i+1} \ldots \psi_n$ such so as to remove this incorrect valuation (and potentially several others).

Abstraction and Refinement



$A_i = \neg\exists x_1 \ldots x_{i-1} F[x_i \mapsto 1]$ and $\psi_i = \neg A_i$

$A'_i = \neg F'_i[x_i \mapsto 1]$ and $\psi'_i = \neg A'_i$

1. When we check if $\psi_1, \ldots, \psi_n$ are proper Skolem functions, and we get a counterexample, it pinpoints a valuation for which abstract Skolem function returns 1 when it should not.
2. We refine Skolem function candidates for $\psi_{i+1} \ldots \psi_n$ such so as to remove this incorrect valuation (and potentially several others).
3. CEGAR loop continues in this way until we find proper Skolem functions.

Skolem functions and their applications

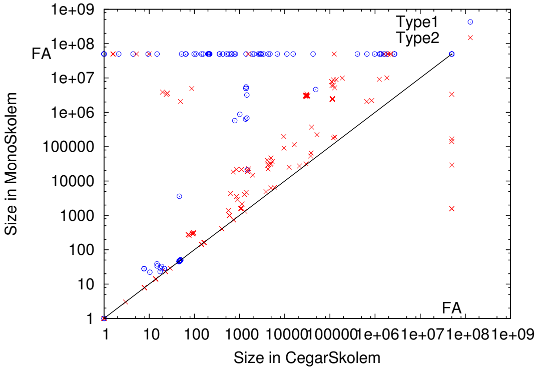
CEGAR for Skolem functions

Experimental Results

Benchmarks

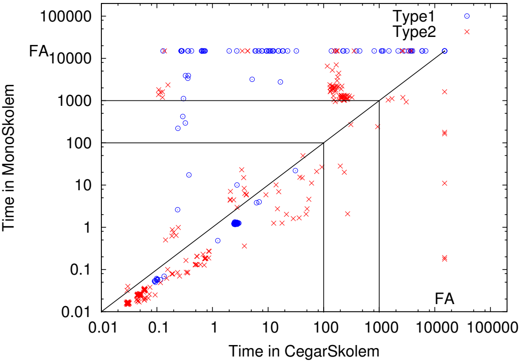
1. We compared the performance of the CEGAR based algorithm with
   1.1 an implementation of the monolithic algorithm
   1.2 The tool Bloqqer (a QRAT based Skolem function generation tool).
2. Our benchmarks were obtained by considering the disjunctive decomposition problem for sequential circuits from HWMCC10 benchmark suite
3. We divided our benchmarks into TYPE-1 formula where $\exists X F(X, Y)$ is valid (160 benchmarks) and TYPE-2 formulas where $\exists X F(X, Y)$ is not valid (264 benchmarks).
4. We used ABC library to represent and manipulate functions as AIGs and used default SAT solver provided by ABC (a variant of miniSAT).
5. We compared these algorithms with respect to Skolem function size and total time taken to generate Skolem functions
6. The maximum time and memory usage was restricted to 2 hours and 32GB.

Monolithic Vs CEGAR: Size



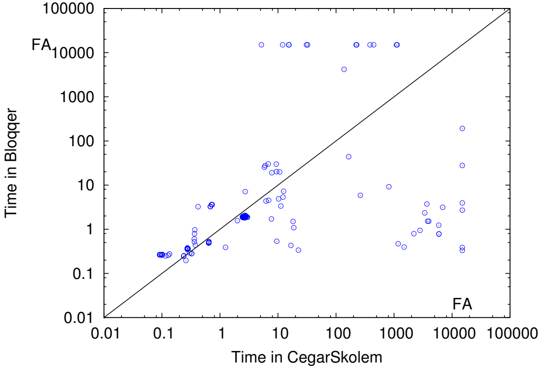
1. There is no instance on which CEGAR generates Skolem functions that are larger on average than Monolithic.

Monolithic Vs CEGAR: Time



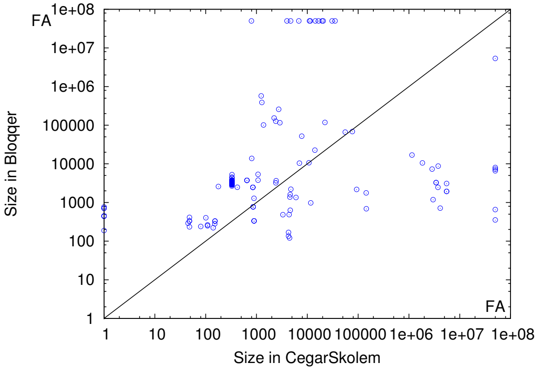
1. Due to repeated calls to SAT solver, CEGAR took more time than Monolithic, but for those examples total time in < 100 seconds.
2. For timed between 100 and 300, Monolithic performed much worse taking more than 1000 seconds (due to large sizes of Skolem functions)
3. Monolithic timed out for 83 benchmarks, while CEGAR for 10

Bloqqer Vs CEGAR: Time



1. Out of 160 TYPE-1 benchmarks Bloqqer generated Skolem functions for 148 benchmarks and gave NOT_VERIFIED message for the remaining.
2. CEGAR was successful for 154 benchmarks.
3. For the benchmarks where Bloqqer gave NOT VERIFIED message, 8 of these 12 were large benchmarks with $1000+$ factors and variables

Bloqqer Vs CEGAR: Size



1. For the 142 common benchmarks, in majority of the cases $(108/142)$ CEGAR generated smaller Skolem functions.

Conclusion

1. Presented a Counterexample guided abstraction refinement based algorithm to generate Skolem functions for factored propositional formulas
2. Experiments show that for complex functions, our algorithm significantly outperformed two state-of-the-art algorithms
3. As a future work, we plan to explore integration with more efficient SAT-solvers, and refinement using multiple counter-examples in parallel.

Conclusion

1. Presented a Counterexample guided abstraction refinement based algorithm to generate Skolem functions for factored propositional formulas
2. Experiments show that for complex functions, our algorithm significantly outperformed two state-of-the-art algorithms
3. As a future work, we plan to explore integration with more efficient SAT-solvers, and refinement using multiple counter-examples in parallel.

Thank you

Rajeev Alur, P. Madhusudan, and Wonhong Nam.
Symbolic computational techniques for solving games.
International Journal on Software Tools for Technology Transfer, 7(2):118–128, 2005.

Marco Benedetti.
sKizzo: A Suite to Evaluate and Certify QBFs.
In *Proc. of CADE*, pages 369–376. Springer-Verlag, 2005.

Marijn Heule, Martina Seidl, and Armin Biere.
Efficient Extraction of Skolem Functions from QRAT Proofs.
In *Proc. of FMCAD*, 2014.

J.-H. R. Jiang and V Balabanov.
Resolution proofs and Skolem functions in QBF evaluation and applications.
In *Proc. of CAV*, pages 149–164. Springer, 2011.

T. Jussila, A. Biere, C. Sinz, D. KrÃ¶ning, and C. Wintersteiger.
A First Step Towards a Unified Proof Checker for QBF.
In *Proc. of SAT*, volume 4501 of *LNCS*, pages 201–214. Springer, 2007.

J.-H. R. Jiang.
Quantifier elimination via functional composition.

In *Proc. of CAV*, pages 383–397. Springer, 2009.

Viktor Kuncak, Mikaël Mayer, Ruzica Piskac, and Philippe Suter.
Complete functional synthesis.
SIGPLAN Not., 45(6):316–329, June 2010.

S. Srivastava, S. Gulwani, and J. S. Foster.
Template-based program verification and program synthesis.
STTT, 15(5-6):497–518, 2013.

Armando Solar-Lezama, Liviu Tancau, Rastislav Bodik, Vijay Saraswat, and Sanjit Seshia.
Combinatorial sketching for finite programs.
In *IN 12TH INTERNATIONAL CONFERENCE ON ARCHITECTURAL SUPPORT FOR PROGRAMMING LANGUAGES AND OPERATING SYSTEMS (ASPLOS 2006*, pages 404–415. ACM Press, 2006.

D. Thomas, S. Chakraborty, and P.K. Pandya.
Efficient guided symbolic reachability using reachability expressions.
STTT, 10(2):113–129, 2008.

Ashutosh Trivedi.
Techniques in symbolic model checking, 2003.