Algorithms for Boolean Functional Synthesis

Supratik Chakraborty

Indian Institute of Technology Bombay

Joint work with S. Akshay, Jatin Arora, Ajith John, S. Krishna, Divya Raghunathan, Shetal Shah

Boolean Relations and Functions

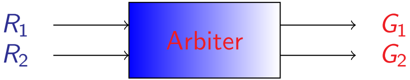
- Boolean functions: fundamental building blocks in computing.

Boolean Relations and Functions

- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs

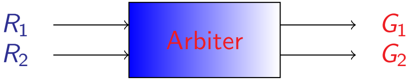
Boolean Relations and Functions

- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



Boolean Relations and Functions

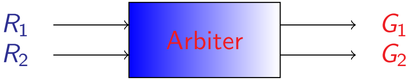
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



  - $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2))$

Boolean Relations and Functions

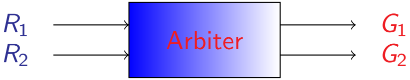
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



    - $((R_1 \lor R_2) \to (G_1 \lor G_2)) \land \neg(G_1 \land G_2)$

Boolean Relations and Functions

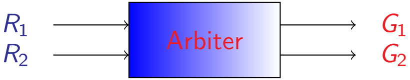
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



  - $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \rightarrow R_1)$

Boolean Relations and Functions

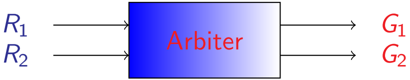
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
    - Relation between inputs and outputs
    - E.g. (Simplified) Arbiter



- $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \rightarrow R_1)$
  $\wedge (G_2 \rightarrow R_2)$

Boolean Relations and Functions

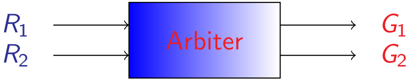
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



    - $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \rightarrow R_1) \wedge (G_2 \rightarrow R_2)$
  - Doesn't specify how to obtain G_1, G_2 as functions of R_1, R_2.

Boolean Relations and Functions

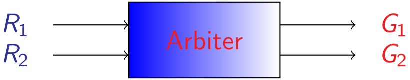
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



    - $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \rightarrow R_1) \wedge (G_2 \rightarrow R_2)$
  - Doesn't specify how to obtain G_1, G_2 as functions of R_1, R_2.
- But we need them in **functional form**
  - Outputs as functions of inputs

Boolean Relations and Functions

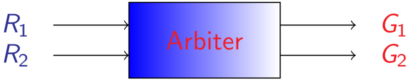
- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



    - $((R_1 \vee R_2) \to (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \to R_1) \wedge (G_2 \to R_2)$
  - Doesn't specify how to obtain G_1, G_2 as functions of R_1, R_2.
- But we need them in **functional form**
  - Outputs as functions of inputs
  - Multiple solutions:
    - $G_1 = (R_1 \wedge \neg R_2)$, $G_2 = R_2$
    - $G_1 = R_1$, $G_2 = (\neg R_1 \wedge R_2)$

Boolean Relations and Functions

- Boolean functions: fundamental building blocks in computing.
- Often easy to specify **relationally**;
  - Relation between inputs and outputs
  - E.g. (Simplified) Arbiter



  - $((R_1 \vee R_2) \rightarrow (G_1 \vee G_2)) \wedge \neg(G_1 \wedge G_2) \wedge (G_1 \rightarrow R_1) \wedge (G_2 \rightarrow R_2)$
  - Doesn't specify how to obtain G_1, G_2 as functions of R_1, R_2.
- But we need them in **functional form**
  - Outputs as functions of inputs
  - Multiple solutions:
    - $G_1 = (R_1 \wedge \neg R_2)$, $G_2 = R_2$
    - $G_1 = R_1$, $G_2 = (\neg R_1 \wedge R_2)$

Boolean Functional Synthesis

Synthesizing Boolean functions from a relational specification.

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(\mathrm{X})$ for each y_j s.t.

$$\forall \mathrm{X}(\ \exists y_1 \ldots y_m\ \varphi(\mathrm{X}, y_1 \ldots y_m) \ \Leftrightarrow\ \varphi(\mathrm{X}, F_1(\mathrm{X}), \ldots F_m(\mathrm{X}))\ )$$

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(X)$ for each y_j s.t.

$$\forall X(\ \exists y_1 \ldots y_m\ \varphi(X, y_1 \ldots y_m)\ \Leftrightarrow\ \varphi(X, F_1(X), \ldots F_m(X))\ )$$

$F_j(X)$ is also called a *Skolem function* for y_j in φ.

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(X)$ for each y_j s.t.

$$\forall X(\ \exists y_1 \dots y_m\ \varphi(X, y_1 \dots y_m)\ \Leftrightarrow\ \varphi(X, F_1(X), \dots F_m(X))\ )$$

$F_j(X)$ is also called a *Skolem function* for y_j in φ.

- Uninteresting if $|X|$ is “small” (say, constant)
  - Tabulate with $2^{|X|}$ calls to $\mathrm{SAT}(\varphi(X, Y))$

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(X)$ for each y_j s.t.

$$\forall X(\ \exists y_1 \dots y_m\ \varphi(X, y_1 \dots y_m)\ \Leftrightarrow\ \varphi(X, F_1(X), \dots F_m(X))\ )$$

$F_j(X)$ is also called a *Skolem function* for y_j in φ.

- Uninteresting if $|X|$ is "small" (say, constant)
  - Tabulate with $2^{|X|}$ calls to $\mathrm{SAT}(\varphi(X, Y))$
- What if $\forall X \exists Y\ \varphi(X, Y) = 0$ ("unrealizable" specification) ?

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(X)$ for each y_j s.t.

$$\forall X(\ \exists y_1 \dots y_m\ \varphi(X, y_1 \dots y_m)\ \Leftrightarrow\ \varphi(X, F_1(X), \dots F_m(X))\ )$$

$F_j(X)$ is also called a *Skolem function* for y_j in φ.

- Uninteresting if $|X|$ is "small" (say, constant)
  - Tabulate with $2^{|X|}$ calls to $\mathrm{SAT}(\varphi(X, Y))$
- What if $\forall X \exists Y\ \varphi(X, Y) = 0$ ("unrealizable" specification) ?
  - Interesting as long as $\exists X \exists Y\ \varphi(X, Y) = 1$

Boolean Functional Synthesis (BFnS)

Formal definition

Given Boolean relation $\varphi(x_1, .., x_n, y_1, .., y_m)$

- x_i *input* variables (vector X)
- y_j *output* variables (vector Y)

Synthesize Boolean functions $F_j(X)$ for each y_j s.t.

$$\forall X(\ \exists y_1 \dots y_m \ \varphi(X, y_1 \dots y_m) \ \Leftrightarrow \ \varphi(X, F_1(X), \dots F_m(X)) \ )$$

$F_j(X)$ is also called a *Skolem function* for y_j in φ.

- Uninteresting if $|X|$ is "small" (say, constant)
  - Tabulate with $2^{|X|}$ calls to $\mathrm{SAT}(\varphi(X, Y))$
- What if $\forall X \exists Y \ \varphi(X, Y) = 0$ ("unrealizable" specification) ?
  - Interesting as long as $\exists X \exists Y \ \varphi(X, Y) = 1$
  - $F(X)$ must give right value of Y for all X s.t. $\exists Y \varphi(X, Y) = 1$
    - $F(X)$ inconsequential for other X

A challenging example: Bounded Integer Factorization

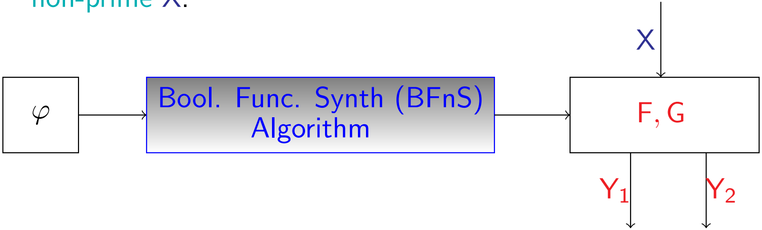
- n-bit integers Y_1, Y_2; $2n$ bit integer X

A challenging example: Bounded Integer Factorization

- n-bit integers Y_1, Y_2; $2n$ bit integer X
- Relational specification $\varphi(X, Y_1, Y_2)$
  - $(X = Y_1 \times_{[n]} Y_2) \wedge (Y_1 \neq 1_{[n]}) \wedge (Y_2 \neq 1_{[n]})$

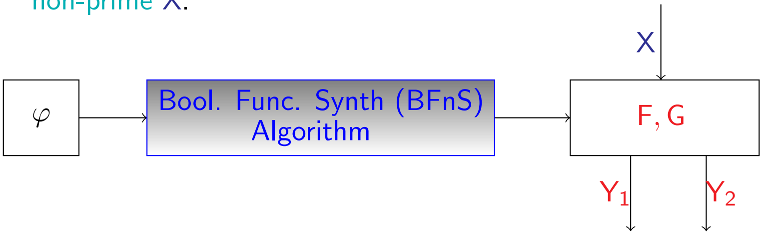
A challenging example: Bounded Integer Factorization

- n-bit integers Y_1, Y_2; $2n$ bit integer X
- Relational specification $\varphi(X, Y_1, Y_2)$
  - $(X = Y_1 \times_{[n]} Y_2) \wedge (Y_1 \neq 1_{[n]}) \wedge (Y_2 \neq 1_{[n]})$
- Synthesize $F(X), G(X)$ s.t. $\varphi(X, F(X), G(X)) = 1$ for all non-prime X.



A challenging example: Bounded Integer Factorization

- n-bit integers Y_1, Y_2; $2n$ bit integer X
- Relational specification $\varphi(X, Y_1, Y_2)$
  - $(X = Y_1 \times_{[n]} Y_2) \wedge (Y_1 \neq 1_{[n]}) \wedge (Y_2 \neq 1_{[n]})$
- Synthesize $F(X), G(X)$ s.t. $\varphi(X, F(X), G(X)) = 1$ for all non-prime X.



- For every non-prime X, finds non-trivial factors
- From prime X, values of $F(X)$ and $G(X)$ inconsequential.
  - $\exists Y_1, Y_2\ \varphi(X, Y_1, Y_2) = 0$ for such X.

Applications of Boolean Functional Synthesis

1. Cryptanalysis: Interesting but hard for synthesis!
2. Disjunctive decomposition of symbolic transition relations [Trivedi et al'02]
3. Quantifier elimination, of course!
   - $\exists Y\ \varphi(X, Y) \equiv \varphi(X, F(X))$
4. Certifying QBF-SAT solvers
   - Nice survey of applications by Shukla et al'19
5. Reactive controller synthesis
   - Synthesizing moves to stay within winning region
6. Program synthesis
   - Combinatorial sketching [Solar-Lezama et al'06, Srivastava et al'13]
   - Complete functional synthesis [Kuncak et al'10]
7. Repair/partial synthesis of circuits [Fujita et al'13]

Existing Approaches

1. Closely related to most general Boolean unifiers
    - Boole'1847, Lowenheim'1908, Macii'98
2. Extract Sk. functions from proof of validity of $\forall X \exists Y \varphi(X, Y)$
    - Bendetti'05, Jussilla et al'07, Balabanov et al'12, Heule et al'14
3. Using templates: Solar-Lezama et al'06, Srivastava et al'13
4. Self-substitution + function composition: Jiang'09, Trivedi'03
5. Synthesis from special normal form representation of specification
    - From ROBDDs: Kukula et al'00, Kuncak et al'10, Fried et al'16, Tabajara et al'17
    - From SynNNF: Akshay et al'09
6. Incremental determinization: Rabe et al'17,'18
7. Quantifier instantiation techniques in SMT solvers
    - Barrett et al'15, Bierre et al'17
8. Input/output component separation: C. et al'18
9. Guess/learn Skolem function candidate + check + repair
    - John et al'15, Akshay et al'17,'18,'20, Golia et al'20

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

- BFnS is *NP*-hard
  - Unlikely, we will get a poly-time algorithm

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

- BFnS is *NP*-hard
  - Unlikely, we will get a poly-time algorithm

- What about size of Skolem functions?
  - Does there always exist compact Skolem functions, although synthesizing may take exponential time?

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

- BFnS is *NP*-hard
  - Unlikely, we will get a poly-time algorithm

- What about size of Skolem functions?
  - Does there always exist compact Skolem functions, although synthesizing may take exponential time?
- Lower bound results in circuit-size refer to monotone circuits [Razbarov 1985; Alon and Boppana 1987]

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

- BFnS is *NP*-hard
  - Unlikely, we will get a poly-time algorithm

- What about size of Skolem functions?
  - Does there always exist compact Skolem functions, although synthesizing may take exponential time?
- Lower bound results in circuit-size refer to monotone circuits [Razbarov 1985; Alon and Boppana 1987]
  - Monotone circuit
    - Output can't change $1 \rightarrow 0$ due to an input changing $0 \rightarrow 1$.

How Hard (or Easy) Is BFnS?

- Boolean circuit: DAG with AND-, OR-, NOT-labeled nodes
- Input: $\varphi(X, Y)$ as $(|X| + |Y|)$-input, 1-output circuit
- Output: Sk. func. vector $F(X)$: $|X|$-input, $|Y|$-output circuit

- BFnS is *NP*-hard
  - Unlikely, we will get a poly-time algorithm

- What about size of Skolem functions?
  - Does there always exist compact Skolem functions, although synthesizing may take exponential time?
- Lower bound results in circuit-size refer to monotone circuits [Razbarov 1985; Alon and Boppana 1987]
  - Monotone circuit
    - Output can't change $1 \to 0$ due to an input changing $0 \to 1$.
  - Skolem functions need not be monotone
  - Different argument for lower bounds on Skolem circuits

Some Good and Bad News

Bad news: [CAV2018]

- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.

Some Good and Bad News

Bad news: [CAV2018]

- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.
- Unless non-uniform exponential-time hypothesis fails, there exist relational specs φ for which Skolem function sizes must be exponential in $|F|$.

Some Good and Bad News

Bad news: [CAV2018]

- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.
- Unless non-uniform exponential-time hypothesis fails, there exist relational specs φ for which Skolem function sizes must be exponential in $|F|$.

Efficient algorithms for Boolean functional synthesis unlikely

Some Good and Bad News

Bad news: [CAV2018]

- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.
- Unless non-uniform exponential-time hypothesis fails, there exist relational specs φ for which Skolem function sizes must be exponential in $|F|$.

Efficient algorithms for Boolean functional synthesis unlikely

Good news: [CAV2018,FMCAD2019]

- If φ is represented in special normal form, synthesis solvable in polynomial (in $|\varphi|$) time and space.

Some Good and Bad News

Bad news: [CAV2018]

- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.
- Unless non-uniform exponential-time hypothesis fails, there exist relational specs φ for which Skolem function sizes must be exponential in $|F|$.

Efficient algorithms for Boolean functional synthesis unlikely

Good news: [CAV2018,FMCAD2019]

- If φ is represented in special normal form, synthesis solvable in polynomial (in $|\varphi|$) time and space.
  - Synthesis Negation Normal Form (SynNNF)
    - Talk in "Beyond Satisfiability" workshop on Mar 23
  - Reasonably common in practice

Experiments: Guess-check-repair algorithms work well in practice

Overview of Guess-Check-Repair Paradigm

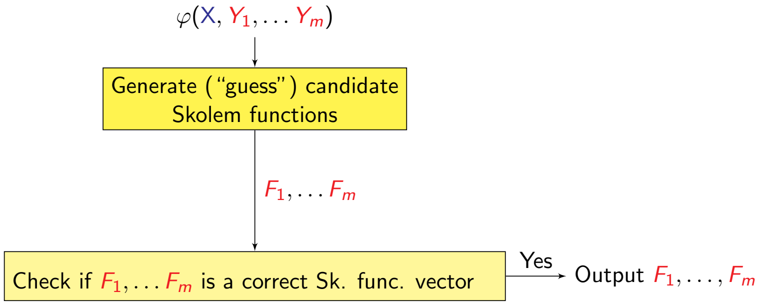
$$\varphi(X, Y_1, \ldots Y_m)$$

$\downarrow$

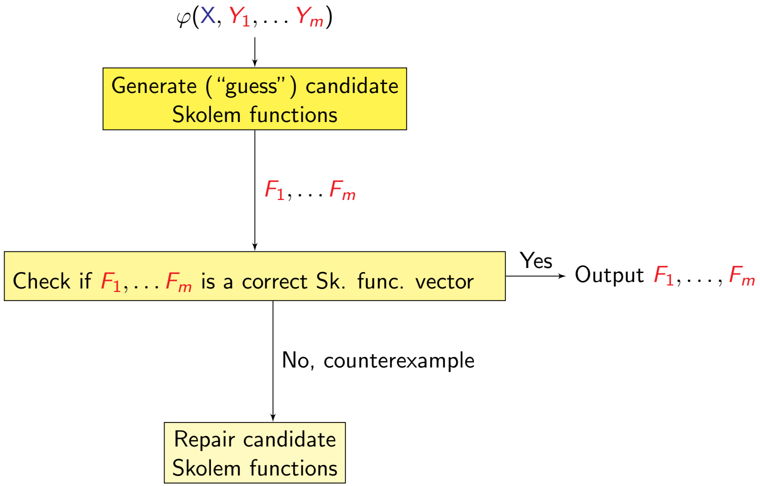
Generate ("guess") candidate
Skolem functions

$\downarrow$ $F_1, \ldots F_m$

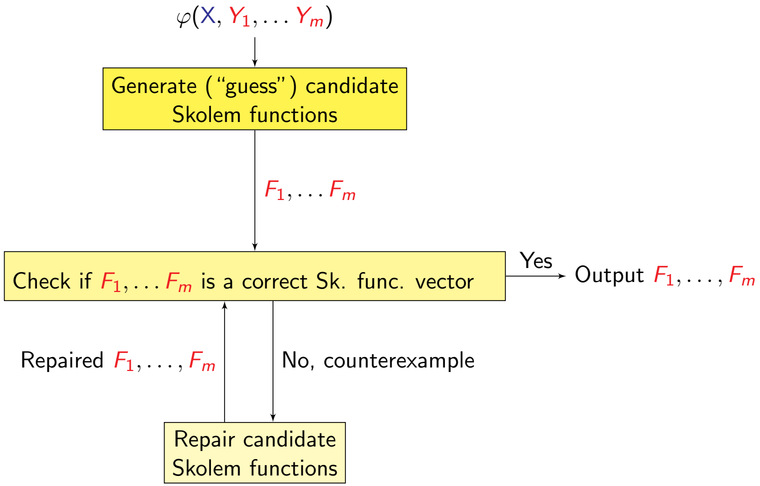
Overview of Guess-Check-Repair Paradigm



Overview of Guess-Check-Repair Paradigm



Overview of Guess-Check-Repair Paradigm

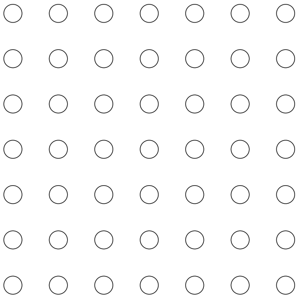


"Guess"-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

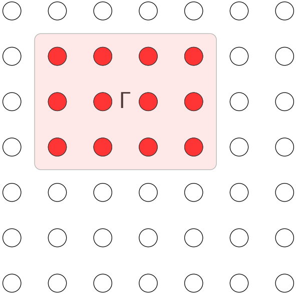
Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



— Set of all valuations of X.

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

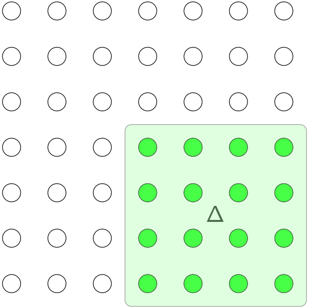


— Can't set y to 1 to satisfy φ: $\Gamma(X) \ \triangleq\ \neg\varphi(X, y)[y \mapsto 1]$

E.g. If $\varphi \equiv (x_1 \vee y) \wedge (x_1 \vee x_2 \vee \neg y)$, then
$\Gamma(X) \ = \ \neg((x_1 \vee 1) \wedge (x_1 \vee x_2 \vee 0)) \ = \ \neg(x_1 \vee x_2) \ = \ \neg x_1 \wedge \neg x_2$

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

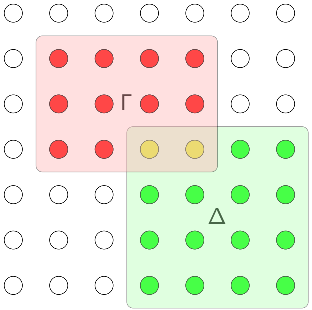


— Can't set y to 0 to satisfy φ: $\Delta(X) \triangleq \neg\varphi(X, y)[y \mapsto 0]$

E.g. If $\varphi \equiv (x_1 \vee y) \wedge (x_1 \vee x_2 \vee \neg y)$, then
$\Delta(X) = \neg((x_1 \vee 0) \wedge (x_1 \vee x_2 \vee 1)) = \neg x_1$

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

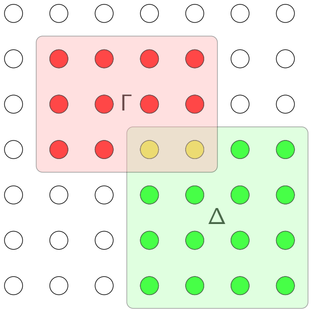
Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



— Can't set y to 1 to satisfy φ: $\Gamma(X) \triangleq \neg\varphi(X, y)[y \mapsto 1]$
— Can't set y to 0 to satisfy φ: $\Delta(X) \triangleq \neg\varphi(X, y)[y \mapsto 0]$

“Guess”-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



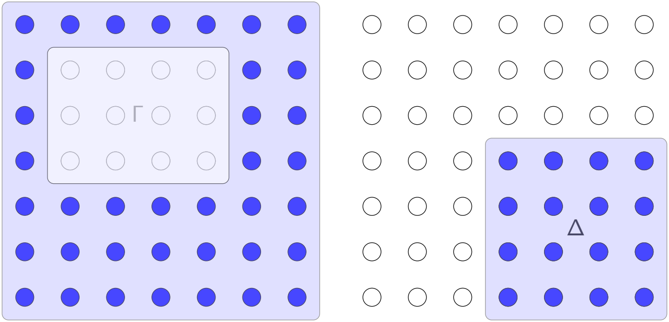
Lemma [Trivedi’03, Jiang’09,Fried et al’16]

Every Skolem function for y in φ must

- Evaluate to 1 in $(\Delta \setminus \Gamma)$ and to 0 in $(\Gamma \setminus \Delta)$
- Be an **interpolant** of $(\Delta \setminus \Gamma)$ and $(\Gamma \setminus \Delta)$

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

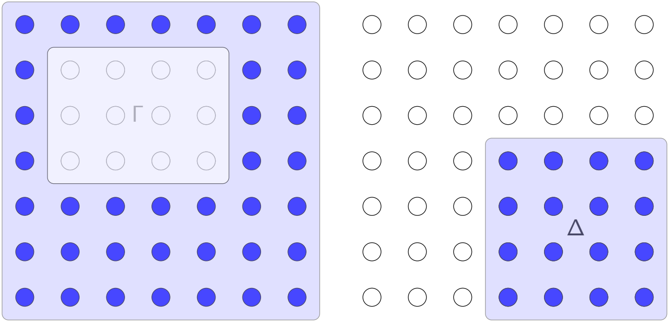


— Specific interpolants of $(\Delta \setminus \Gamma)$ & $(\Gamma \setminus \Delta)$

- $\neg\Gamma \triangleq \varphi(X, y)[y \mapsto 1] \equiv \varphi(X, 1)$
- $\Delta \triangleq \neg\varphi(X, y)[y \mapsto 0] \equiv \neg\varphi(X, 0)$.

"Guess"-ing candidate Skolem functions ($|Y| = 1$)

Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



— Specific interpolants of $(\Delta \setminus \Gamma)$ & $(\Gamma \setminus \Delta)$

- $\neg\Gamma \triangleq \varphi(X, y)[y \mapsto 1] \equiv \varphi(X, 1)$: Easy solution for 1 output var
- $\Delta \triangleq \neg\varphi(X, y)[y \mapsto 0] \equiv \neg\varphi(X, 0)$.

“Guess”-ing Game: $(|\mathsf{Y}| = 2)$

Suppose relational spec is $\varphi(\mathsf{X}, y_1, y_2)$

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general

“Guess”-ing Game: ($|Y| = 2$)

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$

"Guess"-ing Game: ($|Y| = 2$)

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
  - y_2 **must be** $\neg y_1$

"Guess"-ing Game: ($|Y| = 2$)

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
    - y_2 **must be** $\neg y_1$
- For what values of X can we not set y_1 to 1 (or 0)?

"Guess"-ing Game: ($|Y| = 2$)

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
  - y_2 **must be** $\neg y_1$
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(X) = \neg\exists y_2\ \varphi(X, 1, y_2) = 0$

"Guess"-ing Game: $(|Y| = 2)$

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
    - y_2 **must be** $\neg y_1$
- For what values of X can we not set y_1 to 1 (or 0)?
    - $\Gamma^{y_1}(X) = \neg\exists y_2\ \varphi(X, 1, y_2) = 0$
    - $\Delta^{y_1}(X) = \neg\exists y_2\ \varphi(X, 0, y_2) = 0$

"Guess"-ing Game: ($|Y| = 2$)

Suppose relational spec is $\varphi(X, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(X, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
  - y_2 **must be** $\neg y_1$
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(X) = \neg \exists y_2\ \varphi(X, 1, y_2) = 0$
  - $\Delta^{y_1}(X) = \neg \exists y_2\ \varphi(X, 0, y_2) = 0$
- From $\Gamma^{y_1}(X)$ and $\Delta^{y_1}(X)$, find Skolem function $F_1(X)$ for y_1
  - E.g. $F_1(X) = \neg \Gamma^{y_1}(X) = 1$

"Guess"-ing Game: $(|\mathrm{Y}| = 2)$

Suppose relational spec is $\varphi(\mathrm{X}, y_1, y_2)$

- Skolem function for y_2 depends on that for y_1 in general
- E.g. $\varphi(\mathrm{X}, y_1, y_2) \equiv (x_1 \vee x_2 \vee y_1 \vee y_2) \wedge (y_1 \oplus y_2)$
  - y_2 **must be** $\neg y_1$
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(\mathrm{X}) = \neg\exists y_2\ \varphi(\mathrm{X}, 1, y_2) = 0$
  - $\Delta^{y_1}(\mathrm{X}) = \neg\exists y_2\ \varphi(\mathrm{X}, 0, y_2) = 0$
- From $\Gamma^{y_1}(\mathrm{X})$ and $\Delta^{y_1}(\mathrm{X})$, find Skolem function $F_1(\mathrm{X})$ for y_1
  - E.g. $F_1(\mathrm{X}) = \neg\Gamma^{y_1}(\mathrm{X}) = 1$
- To find Skolem function for y_2, consider y_2 as sole output in $\varphi(\mathrm{X}, F_1(\mathrm{X}), y_2)$
  - E.g. $\varphi(\mathrm{X}, 1, y_2) = \neg y_2$
  - $\Gamma^{y_2}(\mathrm{X}) = \neg\varphi(\mathrm{X}, 1, 1) = 1$; $\Delta^{y_2}(\mathrm{X}) = \neg\varphi(\mathrm{X}, 1, 0) = 0$
  - $F_2(\mathrm{X}) = \neg\Gamma^{y_2}(\mathrm{X}) = 0$

“Guess”-ing Game: ($|Y| > 2$)

Suppose relational spec is $\varphi(X, y_1, \boxed{Y_{2..m}})$

- Skolem function for $\boxed{Y_{2..m}}$ depends on that for y_1 in general
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(X) = \neg\exists\, Y_{2..m}\ \varphi(X, 1, \boxed{Y_{2..m}})$
  - $\Delta^{y_1}(X) = \neg\exists\, Y_{2..m}\ \varphi(X, 0, \boxed{Y_{2..m}})$
- From $\Gamma^{y_1}(X)$ and $\Delta^{y_1}(X)$, find Skolem function $F_1(X)$ for y_1
- To find Skolem function for y_2, consider y_2 as sole output in $\varphi(X, F_1(X), y_2, \boxed{Y_{3..m}})$

“Guess”-ing Game: ($|Y| > 2$)

Suppose relational spec is $\varphi(X, y_1, \boxed{Y_{2..m}})$

- Skolem function for $\boxed{Y_{2..m}}$ depends on that for y_1 in general
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(X) = \neg\exists\, Y_{2..m}\ \varphi(X, 1, \boxed{Y_{2..m}})$
  - $\Delta^{y_1}(X) = \neg\exists\, Y_{2..m}\ \varphi(X, 0, \boxed{Y_{2..m}})$
- From $\Gamma^{y_1}(X)$ and $\Delta^{y_1}(X)$, find Skolem function $F_1(X)$ for y_1
- To find Skolem function for y_2, consider y_2 as sole output in $\varphi(X, F_1(X), y_2, \boxed{Y_{3..m}})$

Drawbacks of approach:

- Existential quant elimination over long sequences of outputs expensive
- Nested compositions lead to blowup of representation

"Guess"-ing Game: ($|Y| > 2$)

Suppose relational spec is $\varphi(X, y_1, \boxed{Y_{2..m}})$

- Skolem function for $\boxed{Y_{2..m}}$ depends on that for y_1 in general
- For what values of X can we not set y_1 to 1 (or 0)?
  - $\Gamma^{y_1}(X) = \neg\exists Y_{2..m}\ \varphi(X, 1, \boxed{Y_{2..m}})$
  - $\Delta^{y_1}(X) = \neg\exists Y_{2..m}\ \varphi(X, 0, \boxed{Y_{2..m}})$
- From $\Gamma^{y_1}(X)$ and $\Delta^{y_1}(X)$, find Skolem function $F_1(X)$ for y_1
- To find Skolem function for y_2, consider y_2 as sole output in $\varphi(X, F_1(X), y_2, \boxed{Y_{3..m}})$

Drawbacks of approach:

- Existential quant elimination over long sequences of outputs expensive
- Nested compositions lead to blowup of representation

Can we work around these drawbacks?

A Useful Observation

Fix a linear ordering of outputs: $y_1 \prec y_2 \prec \cdots \prec y_m$

A Useful Observation

Fix a linear ordering of outputs: $y_1 \prec y_2 \prec \cdots \prec y_m$

Express

- y_m as $G_m(\mathsf{X}, y_1, \ldots y_{m-1})$

A Useful Observation

Fix a linear ordering of outputs: $y_1 \prec y_2 \prec \cdots \prec y_m$

Express

- y_m as $G_m(X, y_1, \ldots y_{m-1})$
- y_{m-1} as $G_{m-1}(X, y_1, \ldots y_{m-2})$
- $\vdots$
- y_1 as $G_1(X)$

A Useful Observation

Fix a linear ordering of outputs: $y_1 \prec y_2 \prec \cdots \prec y_m$

Express

- y_m as $G_m(X, y_1, \ldots y_{m-1})$
- y_{m-1} as $G_{m-1}(X, y_1, \ldots y_{m-2})$
- $\vdots$
- y_1 as $G_1(X)$

A $|X|$-input, $|Y|$-output circuit computing the desired Skolem function vector $(F_1, \ldots F_m)$ can be constructed with

- $\#\text{gates} \leq \sum_{i=1}^{m} \#\text{gates}(G_i) + 2m$
- $\#\text{wires} \leq \sum_{i=1}^{m} \#\text{wires}(G_i) + \frac{m(m-1)}{2}$

A Useful Observation

Fix a linear ordering of outputs: $y_1 \prec y_2 \prec \cdots \prec y_m$

Express

- y_m as $G_m(X, y_1, \ldots y_{m-1})$
- y_{m-1} as $G_{m-1}(X, y_1, \ldots y_{m-2})$
- $\vdots$
- y_1 as $G_1(X)$

A $|X|$-input, $|Y|$-output circuit computing the desired Skolem function vector $(F_1, \ldots F_m)$ can be constructed with

- $\#\text{gates} \leq \sum_{i=1}^{m} \#\text{gates}(G_i) + 2m$
- $\#\text{wires} \leq \sum_{i=1}^{m} \#\text{wires}(G_i) + \frac{m(m-1)}{2}$

Sufficient to compute the G_i functions

“Guess”-ing Compositionally: A High-level View

Suppose $\varphi(\mathsf{X}, Y) \equiv \varphi_1(\mathsf{X}, Y) \;\wedge\; \varphi_2(\mathsf{X}, Y)$, where $Y = y_1, \ldots y_m$

“Guess”-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \;\wedge\; \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_{1}^{y_1} \;\triangleq\; \neg\exists y_2 \ldots y_m \; \varphi_1(X, 1, y_2 \ldots y_m)$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m) \quad \Delta_1^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$

"Guess"-ing Compositionally: A High-level View

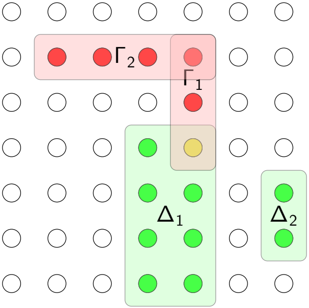
Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$$\begin{array}{lcllcl}
\Gamma_1^{y_1} & \triangleq & \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m) & \Delta_1^{y_1} & \triangleq & \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots) \\
\Gamma_2^{y_1} & \triangleq & \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots) & \Delta_2^{y_1} & \triangleq & \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)
\end{array}$$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

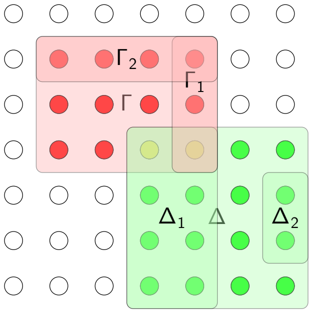
$$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m) \qquad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$$
$$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots) \qquad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$$



"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m) \qquad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$$
$$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots) \qquad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$$



Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \wedge \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \vee \Gamma_2^{y_1} \ \Rightarrow \Gamma^{y_1}$
If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \wedge \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \vee \Delta_2^{y_1} \ \Rightarrow \Delta^{y_1}$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(\mathsf{X}, Y) \equiv \varphi_1(\mathsf{X}, Y) \;\vee\; \varphi_2(\mathsf{X}, Y)$, where $Y = y_1, \ldots y_m$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \;\vee\; \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_2(X, 1, \ldots)$ $\quad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \;\Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \;\Leftrightarrow \Delta^{y_1}$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \;\vee\; \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_2(X, 1, \ldots)$ $\quad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \;\Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \;\Leftrightarrow \Delta^{y_1}$

What if calculating $\Gamma_1^{y_i}$ or $\Delta_1^{y_i}$ hard?

“Guess”-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \vee \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots)$ $\quad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \ \Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \ \Leftrightarrow \Delta^{y_1}$

What if calculating $\Gamma_1^{y_i}$ or $\Delta_1^{y_i}$ hard?

- Long sequences of quantification are of concern!

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \vee \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots)$ $\quad \Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \ \Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \ \Leftrightarrow \Delta^{y_1}$

What if calculating $\Gamma_1^{y_i}$ or $\Delta_1^{y_i}$ hard?

- Long sequences of quantification are of concern!
- Using under-approximations of $\Gamma_1^{y_i}$ and $\Delta_1^{y_i}$ yields under-approximations of Γ^{y_i} and Δ^{y_i}

“Guess”-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \;\vee\; \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m\; \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad$ $\Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m\; \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m\; \varphi_2(X, 1, \ldots)$ $\quad$ $\Delta_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m\; (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \;\Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m\; (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \;\Leftrightarrow \Delta^{y_1}$

What if calculating $\Gamma_1^{y_i}$ or $\Delta_1^{y_i}$ hard?

- Long sequences of quantification are of concern!
- Using under-approximations of $\Gamma_1^{y_i}$ and $\Delta_1^{y_i}$ yields under-approximations of Γ^{y_i} and Δ^{y_i}
  - Not so for over-approximations!
    - ${\Gamma_1}^{y_i} \vee (\wedge)\; {\Gamma_2}^{y_i} \Rightarrow (\Leftrightarrow)\Gamma^{y_i}$
    - ${\Delta_1}^{y_i} \vee (\wedge)\; {\Delta_2}^{y_i} \Rightarrow (\Leftrightarrow)\Delta^{y_i}$

"Guess"-ing Compositionally: A High-level View

Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \vee \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

$\Gamma_1^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \ \varphi_1(X, 1, y_2 \ldots y_m)$ $\quad \Delta_1^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \ \varphi_1(X, 0, \ldots)$

$\Gamma_2^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \ \varphi_2(X, 1, \ldots)$ $\quad \Delta_2^{y_1} \ \triangleq \ \neg\exists y_2 \ldots y_m \varphi_2(X, 0, \ldots)$

Lemma

If $\Gamma^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 1, \ldots)$, then $\Gamma_1^{y_1} \wedge \Gamma_2^{y_1} \ \Leftrightarrow \Gamma^{y_1}$

If $\Delta^{y_1} \triangleq \neg\exists y_2 \ldots y_m \ (\varphi_1 \vee \varphi_2)(X, 0, \ldots)$, then $\Delta_1^{y_1} \wedge \Delta_2^{y_1} \ \Leftrightarrow \Delta^{y_1}$

What if calculating $\Gamma_1^{y_i}$ or $\Delta_1^{y_i}$ hard?

- Long sequences of quantification are of concern!
- Using under-approximations of $\Gamma_1^{y_i}$ and $\Delta_1^{y_i}$ yields under-approximations of Γ^{y_i} and Δ^{y_i}
  - Not so for over-approximations!
    - ${\Gamma_1}^{y_i} \vee (\wedge) \ {\Gamma_2}^{y_i} \Rightarrow (\Leftrightarrow)\Gamma^{y_i}$
    - ${\Delta_1}^{y_i} \vee (\wedge) \ {\Delta_2}^{y_i} \Rightarrow (\Leftrightarrow)\Delta^{y_i}$
- Fortunately, non-trivial under-approx of Γ^{y_i} and Δ^{y_i} not hard to obtain

"Guess"-ing with under-approximations of Γ, Δ

- Suppose $\gamma_{1}^{y_i} \Rightarrow \Gamma_{1}^{y_i}$; $\delta_{1}^{y_i} \Rightarrow \Delta_{1}^{y_i}$

"Guess"-ing with under-approximations of Γ, Δ

- Suppose $\gamma_1^{y_i} \Rightarrow \Gamma_1^{y_i}$; $\delta_1^{y_i} \Rightarrow \Delta_1^{y_i}$
- $\varphi \equiv \varphi_1 \wedge \varphi_2$
  - $\gamma_1^{y_i} \vee \gamma_1^{y_i} \quad \Rightarrow \quad \Gamma_1^{y_i} \vee \Gamma_1^{y_i} \quad \Rightarrow \quad \Gamma^{y_i}$

"Guess"-ing with under-approximations of Γ, Δ

- Suppose $\gamma_1^{y_i} \Rightarrow \Gamma_1^{y_i}$; $\delta_1^{y_i} \Rightarrow \Delta_1^{y_i}$
- $\varphi \equiv \varphi_1 \wedge \varphi_2$
  - $\gamma_1^{y_i} \vee \gamma_1^{y_i} \quad \Rightarrow \quad \Gamma_1^{y_i} \vee \Gamma_1^{y_i} \quad \Rightarrow \quad \Gamma^{y_i}$
- $\varphi \equiv \varphi_1 \vee \varphi_2$
  - $\gamma_1^{y_i} \wedge \gamma_1^{y_i} \quad \Rightarrow \quad \Gamma_1^{y_i} \wedge \Gamma_1^{y_i} \quad \Leftrightarrow \quad \Gamma^{y_i}$
- Similarly for Δ^{y_i}

"Check"-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall \mathsf{X}(\ \exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y}) \ \Leftrightarrow \ \varphi(\mathsf{X}, \mathsf{F}(\mathsf{X})\ )\ ?$$

"Check"-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall \mathsf{X} \left(\; \exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y}) \;\Leftrightarrow\; \varphi(\mathsf{X}, \mathsf{F}(\mathsf{X})) \; \right) \text{ ?}$$

Can we avoid using a QBF solver?

“Check”-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall \mathsf{X}(\ \exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y}) \ \Leftrightarrow \ \varphi(\mathsf{X}, \mathsf{F}(\mathsf{X}))\ ) \ ?$$

Can we avoid using a QBF solver?

Yes, we can! [FMCAD15]

- Propositional error formula $\varepsilon(\mathsf{X}, \mathsf{Y}, \mathsf{Y}')$:

$$\left(\varphi(\mathsf{X}, \mathsf{Y}') \wedge \bigwedge_{j=1}^{m} (Y_j \Leftrightarrow F_j) \wedge \neg\varphi(\mathsf{X}, \mathsf{Y})\right)$$

"Check"-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall \mathsf{X}\big(\ \exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y})\ \Leftrightarrow\ \varphi(\mathsf{X}, \mathsf{F}(\mathsf{X}))\ \big)\ ?$$

Can we avoid using a QBF solver?

Yes, we can! [FMCAD15]

- Propositional error formula $\varepsilon(\mathsf{X}, \mathsf{Y}, \mathsf{Y}')$:

$$\Big(\varphi(\mathsf{X}, \mathsf{Y}') \wedge \bigwedge_{j=1}^{m} (Y_j \Leftrightarrow F_j) \wedge \neg\varphi(\mathsf{X}, \mathsf{Y})\Big)$$

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector

"Check"-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall \mathsf{X}(\ \exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y}) \ \Leftrightarrow \ \varphi(\mathsf{X}, \mathsf{F}(\mathsf{X}))\ ) \text{ ?}$$

Can we avoid using a QBF solver?

Yes, we can! [FMCAD15]

- Propositional error formula $\varepsilon(\mathsf{X}, \mathsf{Y}, \mathsf{Y}')$:

$$\left(\varphi(\mathsf{X}, \mathsf{Y}') \wedge \bigwedge_{j=1}^{m} (Y_j \Leftrightarrow F_j) \wedge \neg\varphi(\mathsf{X}, \mathsf{Y})\right)$$

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector
  - Say, σ = satisfying assignment of ε

“Check”-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall X\big(\ \exists Y \varphi(X, Y) \ \Leftrightarrow\ \varphi(X, F(X))\ \big)\ ?$$

Can we avoid using a QBF solver?

Yes, we can! [FMCAD15]

- Propositional error formula $\varepsilon(X, Y, Y')$:

$$\Big(\varphi(X, Y') \wedge \bigwedge_{j=1}^{m}(Y_j \Leftrightarrow F_j) \wedge \neg\varphi(X, Y)\Big)$$

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector
  - Say, σ = satisfying assignment of ε
  - On input $\sigma(X)$, F evaluates to $\sigma(Y)$, where
    - $\varphi(\sigma(X), \sigma(Y)) = 0$
    - $\varphi(\sigma(X), \sigma(Y')) = 1$

"Check"-ing correctness of candidate Skolem func. vector

Given candidate Skolem functions $F_1, \ldots F_m$,

$$\text{Is } \forall X(\ \exists Y \varphi(X, Y) \ \Leftrightarrow \ \varphi(X, F(X))\ ) \ ?$$

Can we avoid using a QBF solver?

Yes, we can! [FMCAD15]

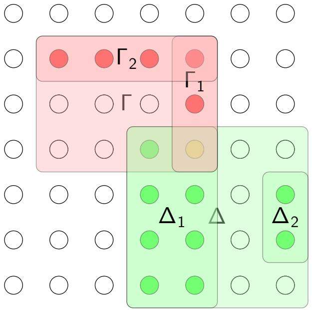
- Propositional error formula $\varepsilon(X, Y, Y')$:

$$\left(\varphi(X, Y') \wedge \bigwedge_{j=1}^{m} (Y_j \Leftrightarrow F_j) \wedge \neg\varphi(X, Y)\right)$$

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector
  - Say, σ = satisfying assignment of ε
  - On input $\sigma(X)$, F evaluates to $\sigma(Y)$, where
    - $\varphi(\sigma(X), \sigma(Y)) = 0$
    - $\varphi(\sigma(X), \sigma(Y')) = 1$
  - σ is counterexample to the claim that $F_1, \ldots F_m$ is a correct Skolem function vector

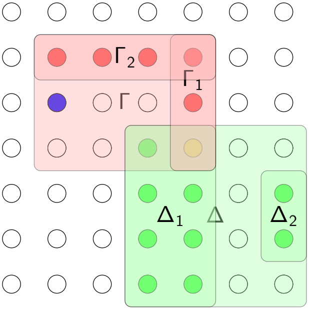
Repairing candidate Skolem functions: A High-level View

$$\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$$



$$\varphi(X, Y) \equiv \varphi_1(X, Y) \;\wedge\; \varphi_2(X, Y)$$

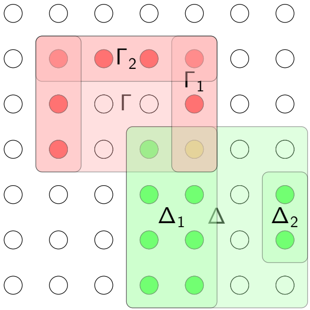
Counterexample



Repairing candidate Skolem functions: A High-level View

$\varphi(X, Y) \equiv \varphi_1(X, Y) \;\wedge\; \varphi_2(X, Y)$

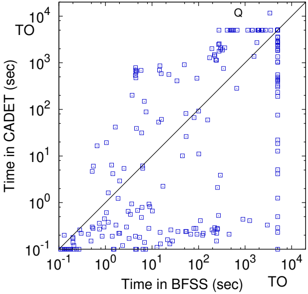
Expansion around CEX

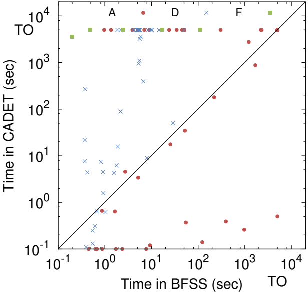


- Always work with under-approximations of Γ and Δ
- Since "proposed" Skolem function is $\neg\Gamma$, intermediate approximations of Skolem functions are over-approximations (abstractions)

BFSS vis-a-vis CADET [Rabe & Seshia'16]

[Comparisons with other tools in paper]

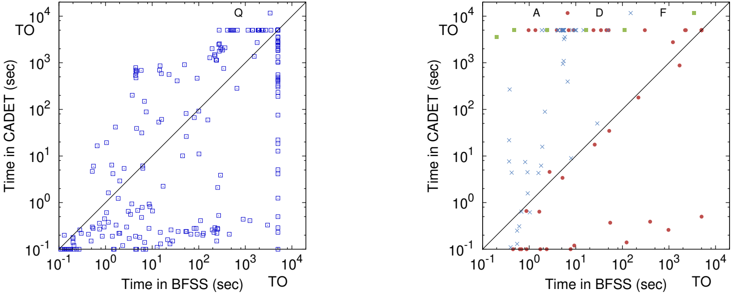




Q: QBFEval, A: Arithmetic, F: Factorization, D: Disjunctive Decomposition. TO: Timeout (3600 sec)

Comparison with other tools

BFSS vis-a-vis CADET [Rabe & Seshia'16]
[Comparisons with other tools in paper]



Q: QBFEval, A: Arithmetic, F: Factorization, D: Disjunctive Decomposition. TO: Timeout (3600 sec)

- Mixed results: tools have orthogonal strengths
- Using CADET and BFSS as a portfolio solver sounds promising