Counterexample-Guided Repair in Boolean Functional Synthesis

Supratik Chakraborty

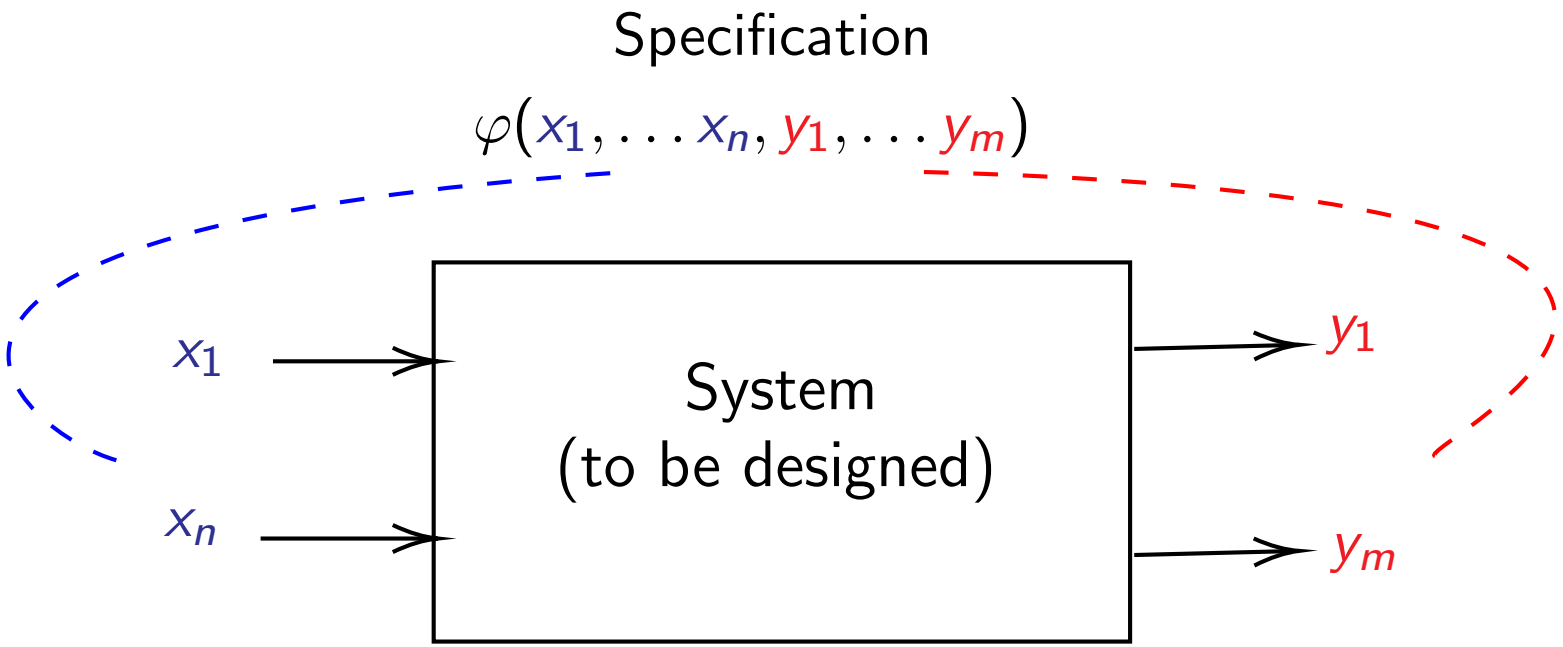
Indian Institute of Technology Bombay

Joint work with

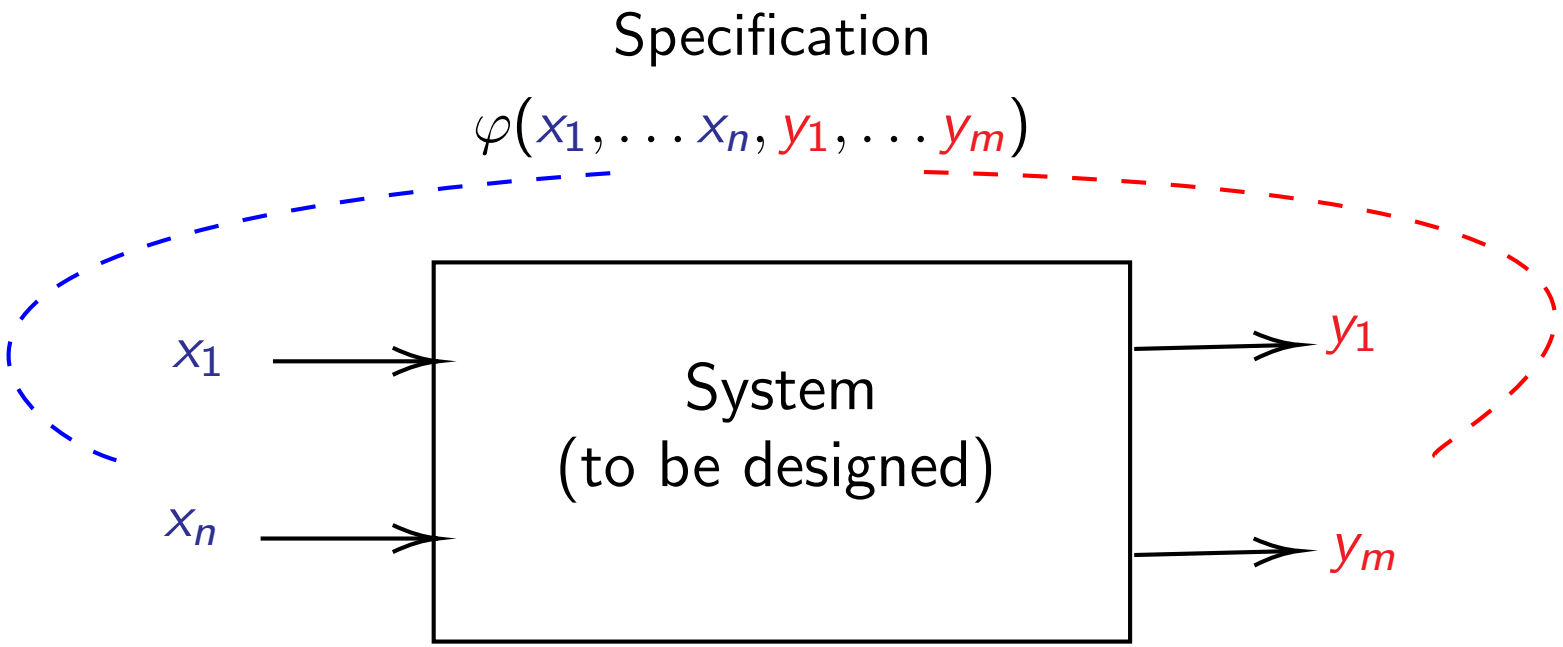
S. Akshay, Shubham Goel, Ajith John, Sumit Kulal, Shetal Shah

Formal Methods Update Meeting, 2021

Synthesis: A Generic View

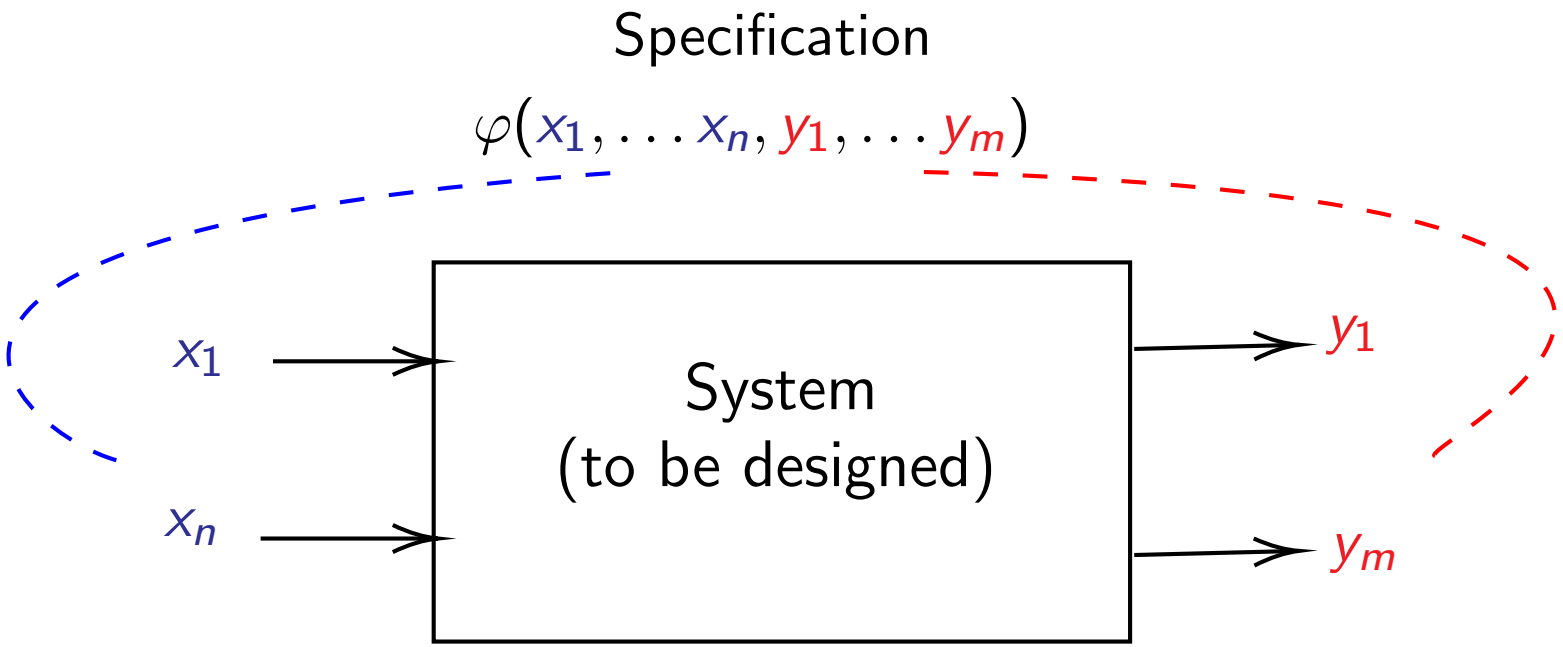


Synthesis: A Generic View



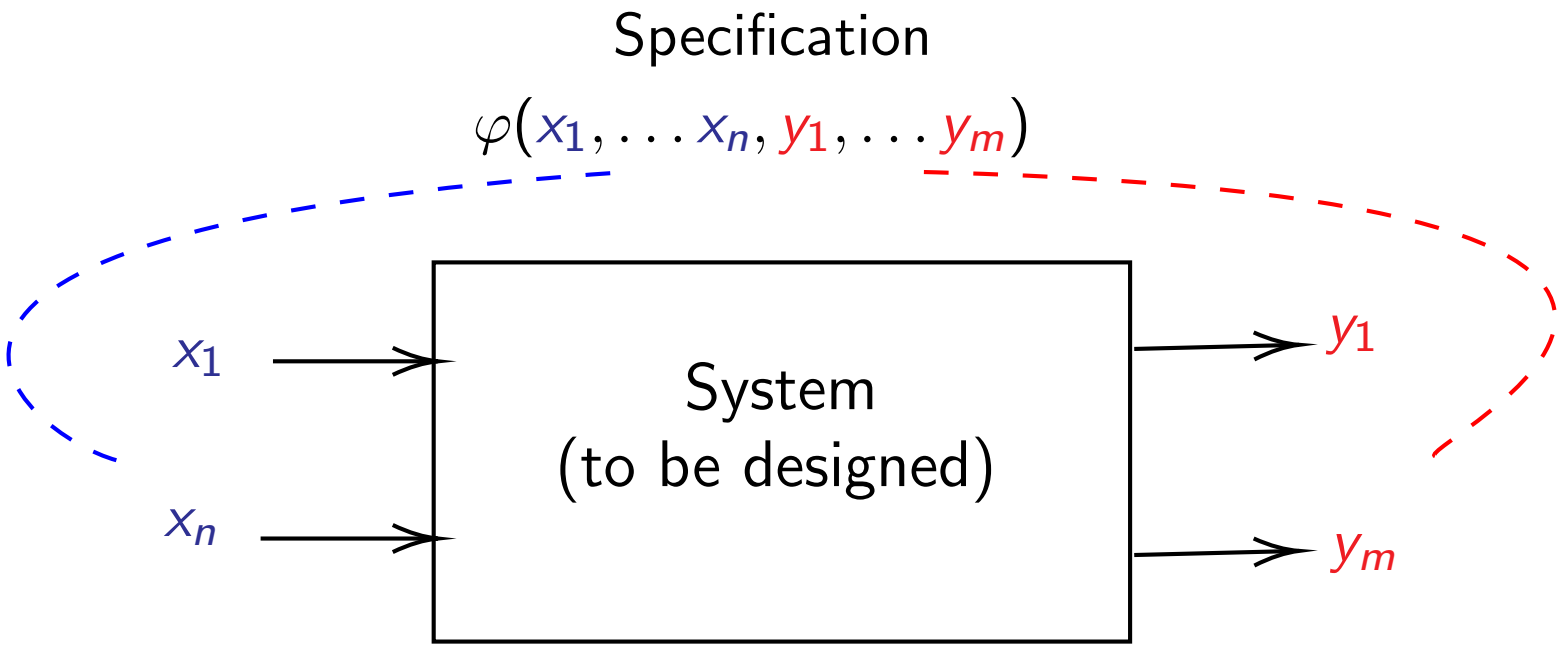
- Goal: Automatically synthesize system s.t. it satisfies $\varphi(x_1, .., x_n, y_1, .., y_m)$
  - x_i *input* variables (vector X)
  - y_j *output* variables (vector Y)

Synthesis: A Generic View



- Goal: Automatically synthesize system s.t. it satisfies $\varphi(x_1, .., x_n, y_1, .., y_m)$ **whenever possible**.
  - x_i *input* variables (vector X)
  - y_j *output* variables (vector Y)

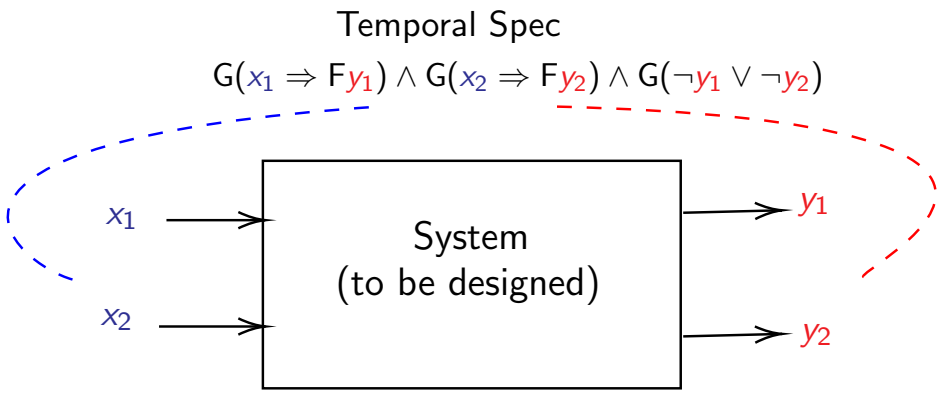
Synthesis: A Generic View



- Goal: Automatically synthesize system s.t. it satisfies $\varphi(x_1, .., x_n, y_1, .., y_m)$ **whenever possible**.
  - x_i *input* variables (vector X)
  - y_j *output* variables (vector Y)
- Need Y as functions F of
  - "History" of X and Y, "State" of system, ...

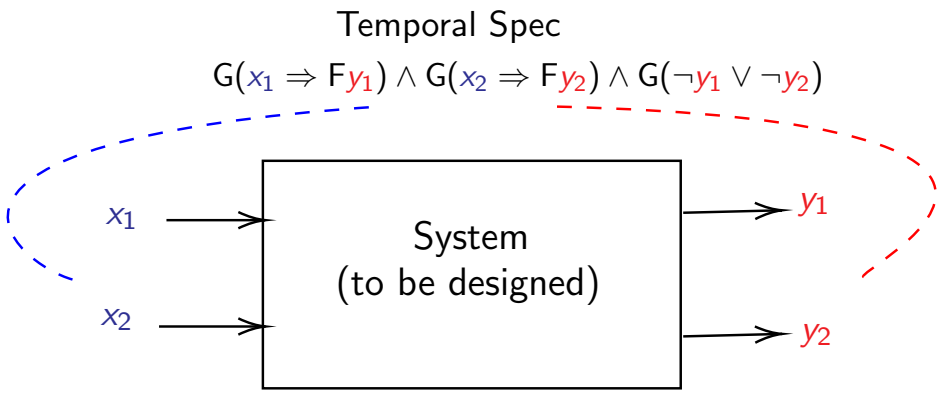
  such that $\varphi(X, F)$ is satisfied.

Synthesis: Some Examples

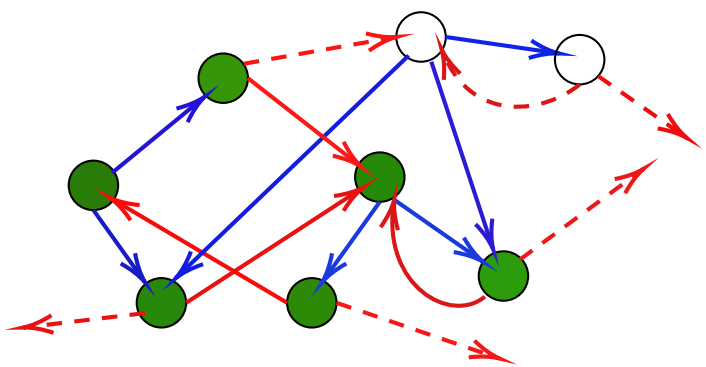


- Synthesize Y as function of
  - State (summarizing "history" of X and Y)

Synthesis: Some Examples

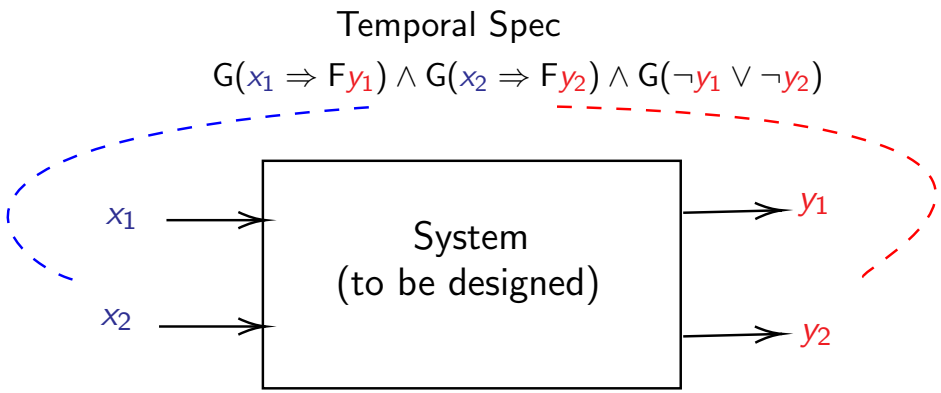


- Synthesize Y as function of
  - State (summarizing "history" of X and Y)

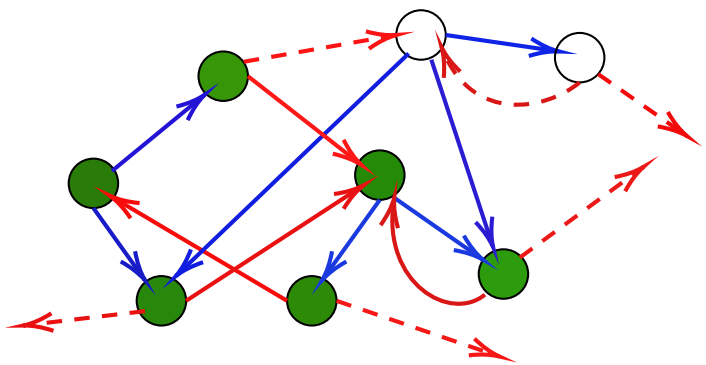


Winning Region

Synthesis: Some Examples



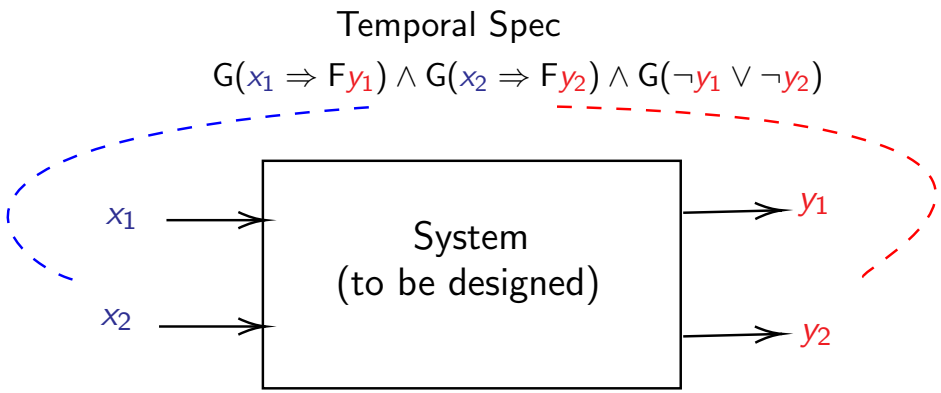
- Synthesize Y as function of
  - State (summarizing "history" of X and Y)



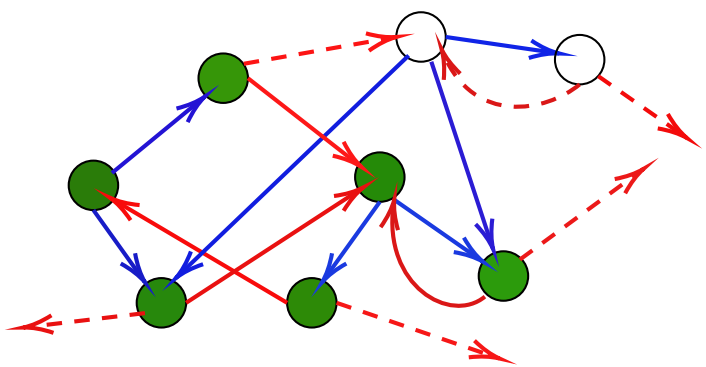
Winning Region

- Synthesize *winning strategy* to stay within *winning region*

Synthesis: Some Examples



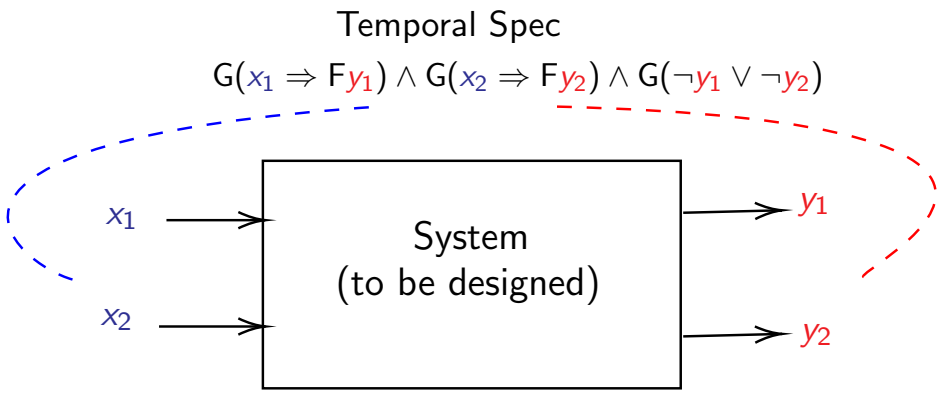
- Synthesize Y as function of
  - State (summarizing "history" of X and Y)



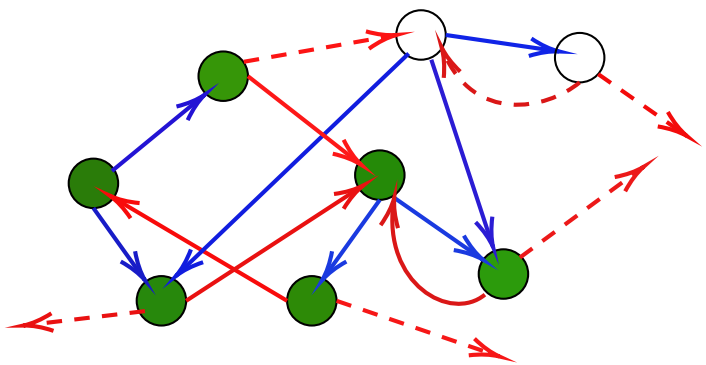
Winning Region

- Synthesize *winning strategy* to stay within *winning region*
  - WinRgn(NxtSt(state, Y)) = 1

Synthesis: Some Examples



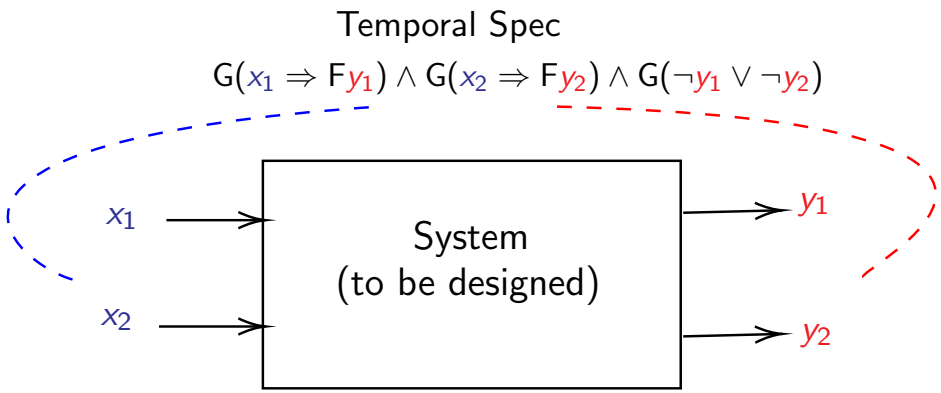
- Synthesize Y as function of
  - State (summarizing "history" of X and Y)



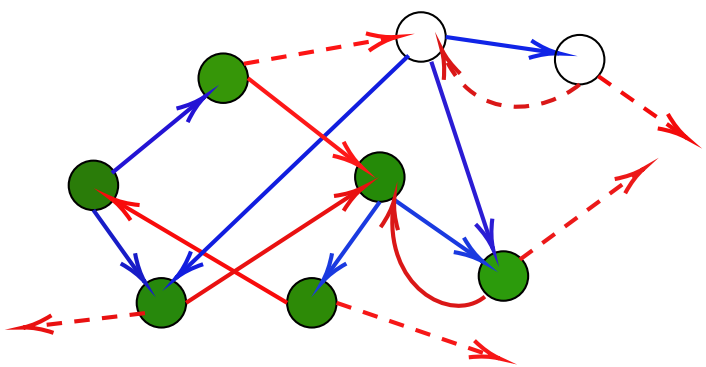
Winning Region

- Synthesize *winning strategy* to stay within *winning region*
  - WinRgn(NxtSt(state, Y)) = 1
    - No temporal operators

Synthesis: Some Examples



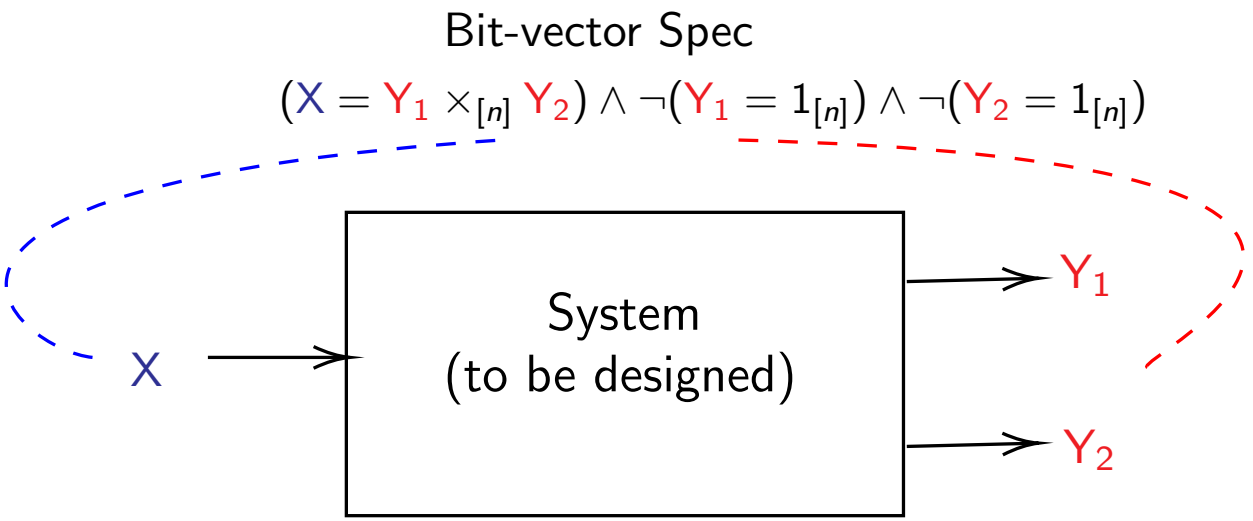
- Synthesize Y as function of
  - State (summarizing "history" of X and Y)



Winning Region

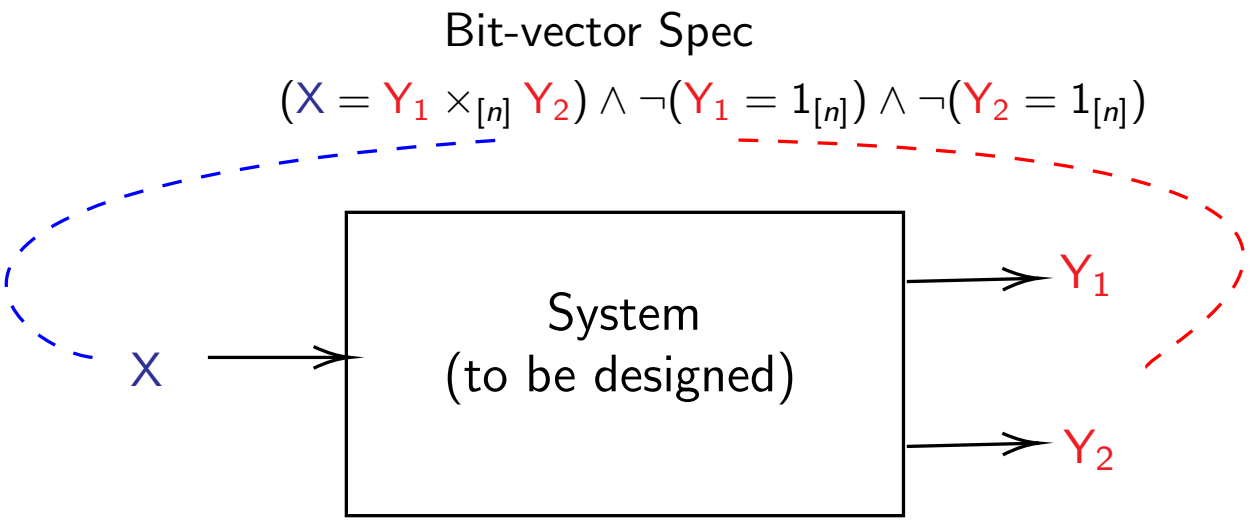
- Synthesize *winning strategy* to stay within *winning region*
  - WinRgn(NxtSt(state, Y)) = 1
    - No temporal operators
  - Not always satisfiable

Synthesis: Some Examples



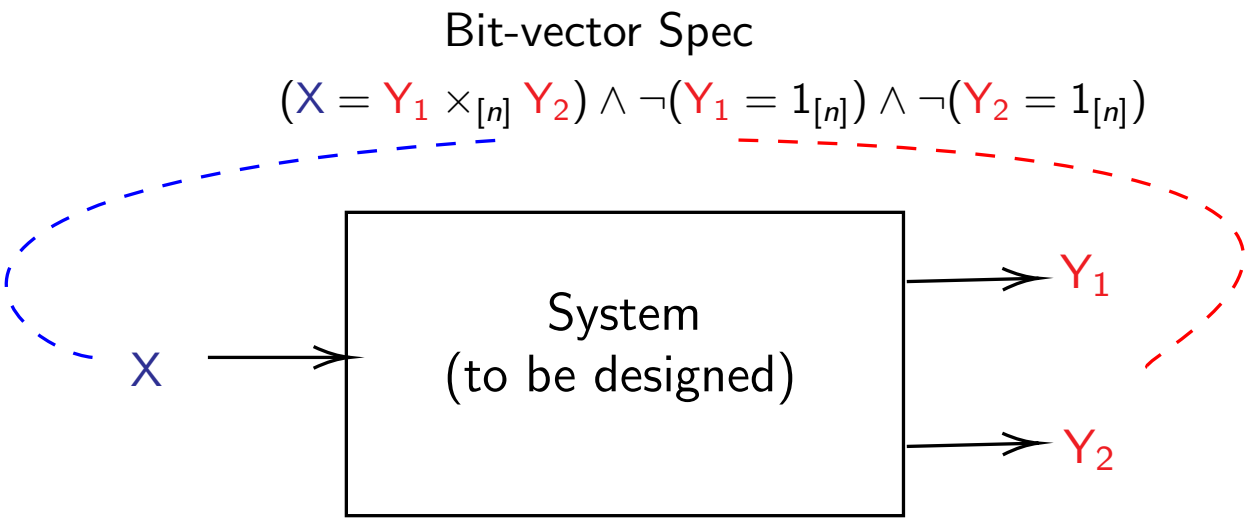
- Synthesize Y_1, Y_2 as functions of X

Synthesis: Some Examples



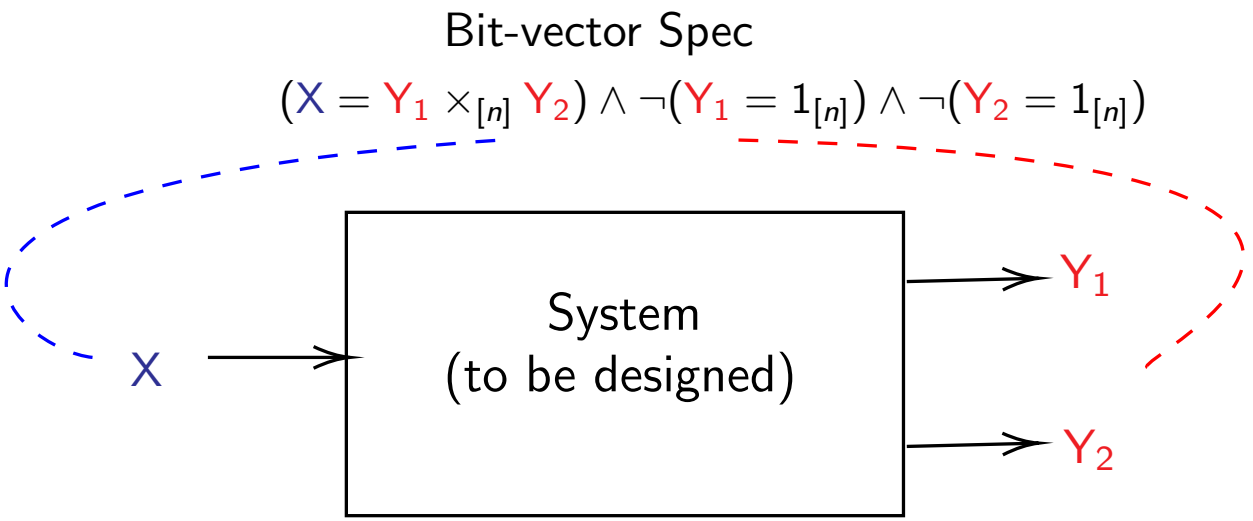
- Synthesize Y_1, Y_2 as functions of X
  - Spec has no temporal operators

Synthesis: Some Examples



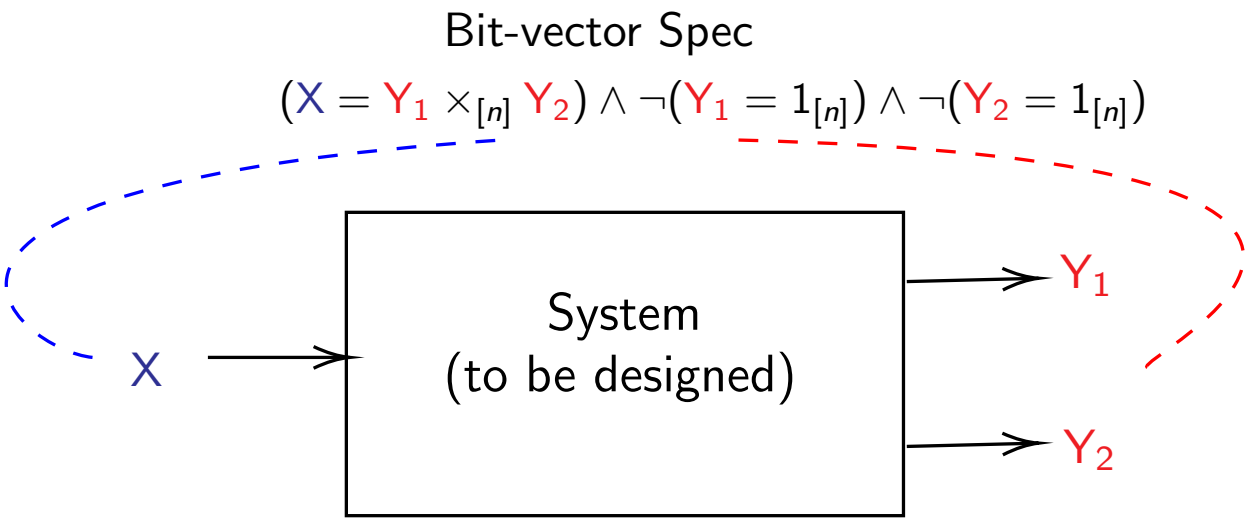
- Synthesize Y_1, Y_2 as functions of X
  - Spec has no temporal operators
  - Y_1, Y_2 must be non-trivial factors of X

Synthesis: Some Examples



- Synthesize Y_1, Y_2 as functions of X
  - Spec has no temporal operators
  - Y_1, Y_2 must be non-trivial factors of X
  - Not always satisfiable (if X is prime)

Synthesis: Some Examples



- Synthesize Y_1, Y_2 as functions of X
  - Spec has no temporal operators
  - Y_1, Y_2 must be non-trivial factors of X
  - Not always satisfiable (if X is prime)
  - Efficient solution would break crypto systems

Boolean Functional Synthesis

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

Formal definition

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

- x_i *input* variables (vector $\mathbf{X}$)
- y_j *output* variables (vector $\mathbf{Y}$)

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

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

Boolean Functional Synthesis

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 $F_j(\mathsf{X})$ for each y_j s.t.

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

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

Boolean Functional Synthesis

Formal definition

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

- x_i *input* variables (vector $\mathbf{X}$)
- y_j *output* variables (vector $\mathbf{Y}$)

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

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

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

- What if $\forall \mathbf{X} \exists \mathbf{Y}\ \varphi(\mathbf{X}, \mathbf{Y}) = 0$?

Boolean Functional Synthesis

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 $F_j(\mathsf{X})$ for each y_j s.t.

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

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

- What if $\forall \mathsf{X} \exists \mathsf{Y}\ \varphi(\mathsf{X}, \mathsf{Y}) = 0$?
  - Interesting as long as $\exists \mathsf{X} \exists \mathsf{Y}\ \varphi(\mathsf{X}, \mathsf{Y}) = 1$

Boolean Functional Synthesis

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 $F_j(\mathsf{X})$ for each y_j s.t.

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

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

- What if $\forall \mathsf{X} \exists \mathsf{Y}\ \varphi(\mathsf{X}, \mathsf{Y}) = 0$?
  - Interesting as long as $\exists \mathsf{X} \exists \mathsf{Y}\ \varphi(\mathsf{X}, \mathsf{Y}) = 1$
  - $\mathsf{F}(\mathsf{X})$ must give right value of Y for all X s.t. $\exists \mathsf{Y} \varphi(\mathsf{X}, \mathsf{Y}) = 1$
    - $\mathsf{F}(\mathsf{X})$ inconsequential for other X

Boolean Functional Synthesis

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.

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

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

- What if $\forall \mathrm{X} \exists \mathrm{Y}\ \varphi(\mathrm{X}, \mathrm{Y}) = 0$?
  - Interesting as long as $\exists \mathrm{X} \exists \mathrm{Y}\ \varphi(\mathrm{X}, \mathrm{Y}) = 1$
  - $\mathrm{F}(\mathrm{X})$ must give right value of Y for all X s.t. $\exists \mathrm{Y} \varphi(\mathrm{X}, \mathrm{Y}) = 1$
    - $\mathrm{F}(\mathrm{X})$ inconsequential for other X
  - Given X, $\mathrm{F}(\mathrm{X})$, easy to check if $\exists \mathrm{Y}\ \varphi(\mathrm{X}, \mathrm{Y}) = \varphi(\mathrm{X}, \mathrm{F}(\mathrm{X})) = 0$

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 $1 \to 0$ due to an input changing $0 \to 1$.

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,LICS2021]

- If φ represented in Synthesis Negation Normal Form (SynNNF), synthesis in polynomial (in $|\varphi|$) time and space.

Some Good and Bad News

Bad news: [CAV2018]

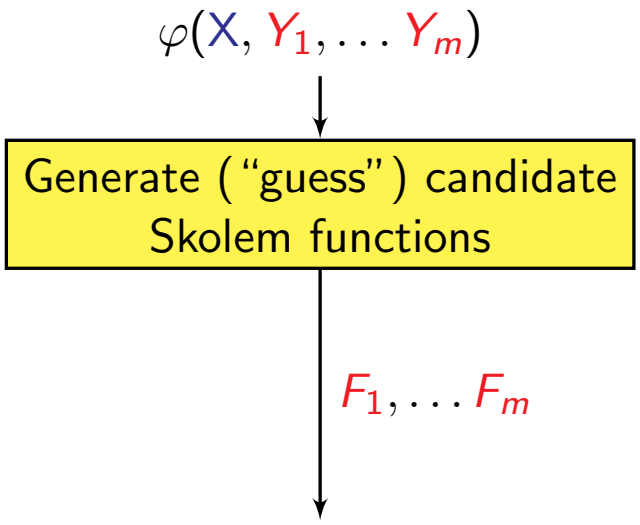
- Unless $\Pi_2^P = \Sigma_2^P$, there exist relational specs φ for which Skolem function sizes must be super-polynomial in $|\varphi|$.
- Unless non-uniform exponential-time hypothesis fails, there exist relational specs φ for which Skolem function sizes must be exponential in $|F|$.

Efficient algorithms for Boolean functional synthesis unlikely

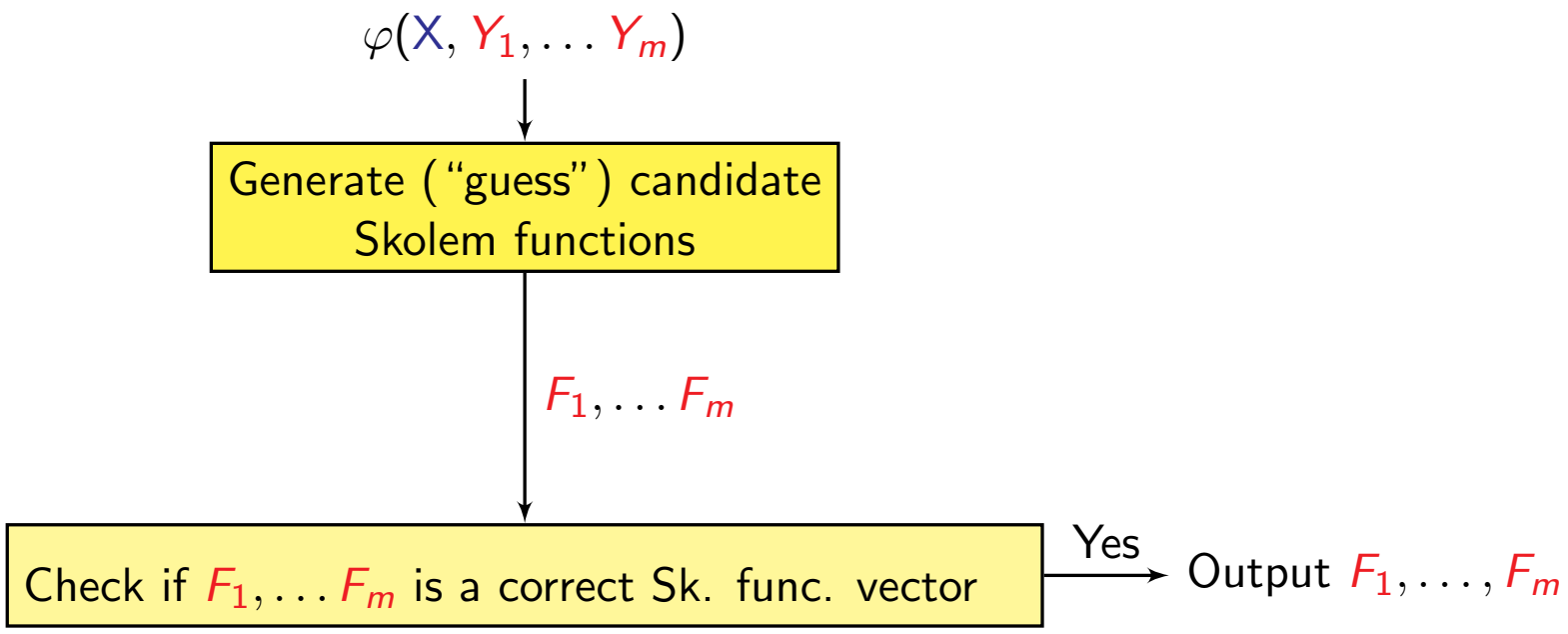
Good news: [CAV2018,FMCAD2019,LICS2021]

- If φ represented in Synthesis Negation Normal Form (SynNNF), synthesis in polynomial (in $|\varphi|$) time and space.
- φ poly-time (size) synthesizable **iff** it is poly-time (size) compilable to Subset And Unrealizable Negation Normal Form (SAUNF)

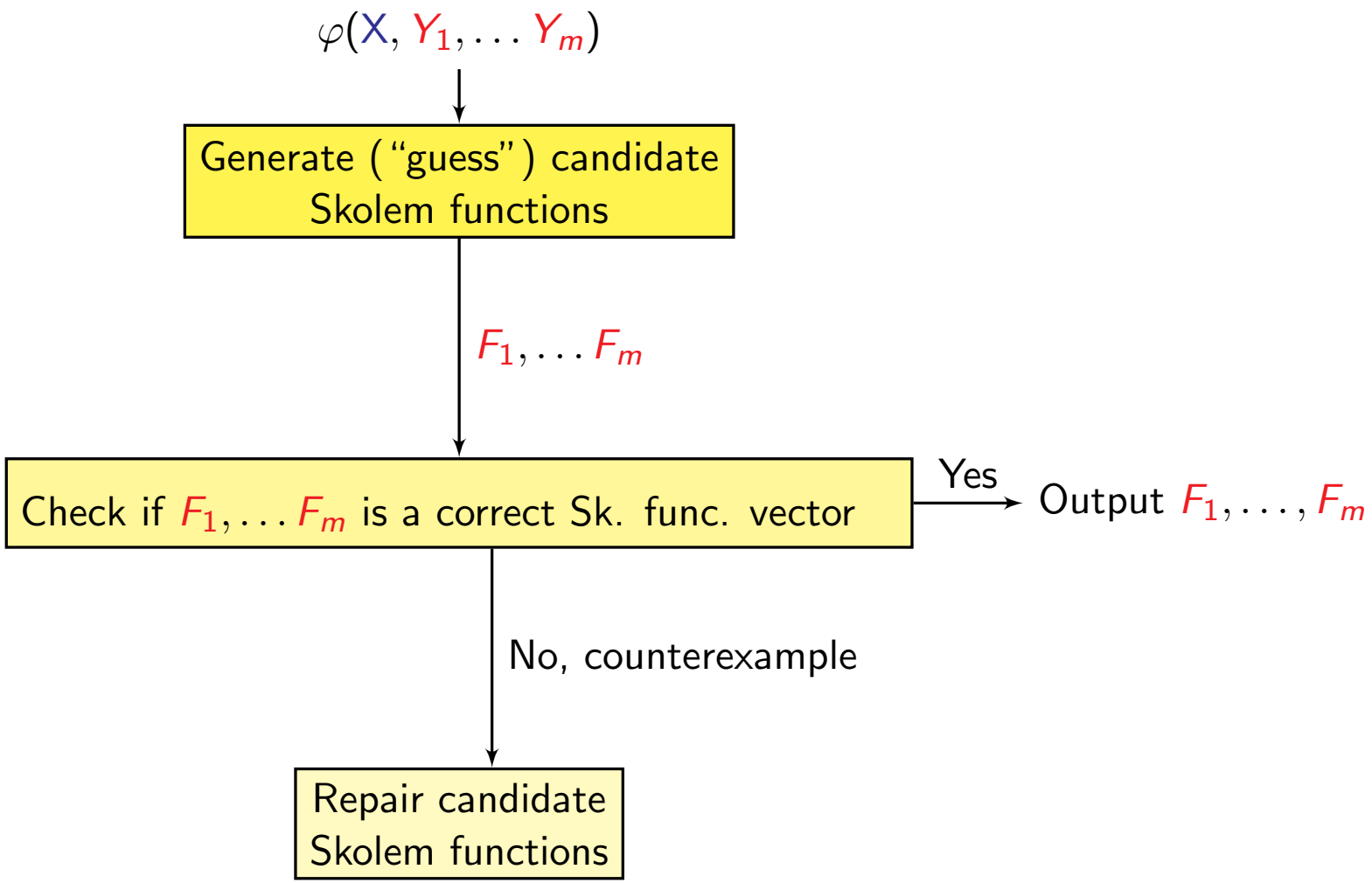
Guess-Check-Repair: A Useful Paradigm



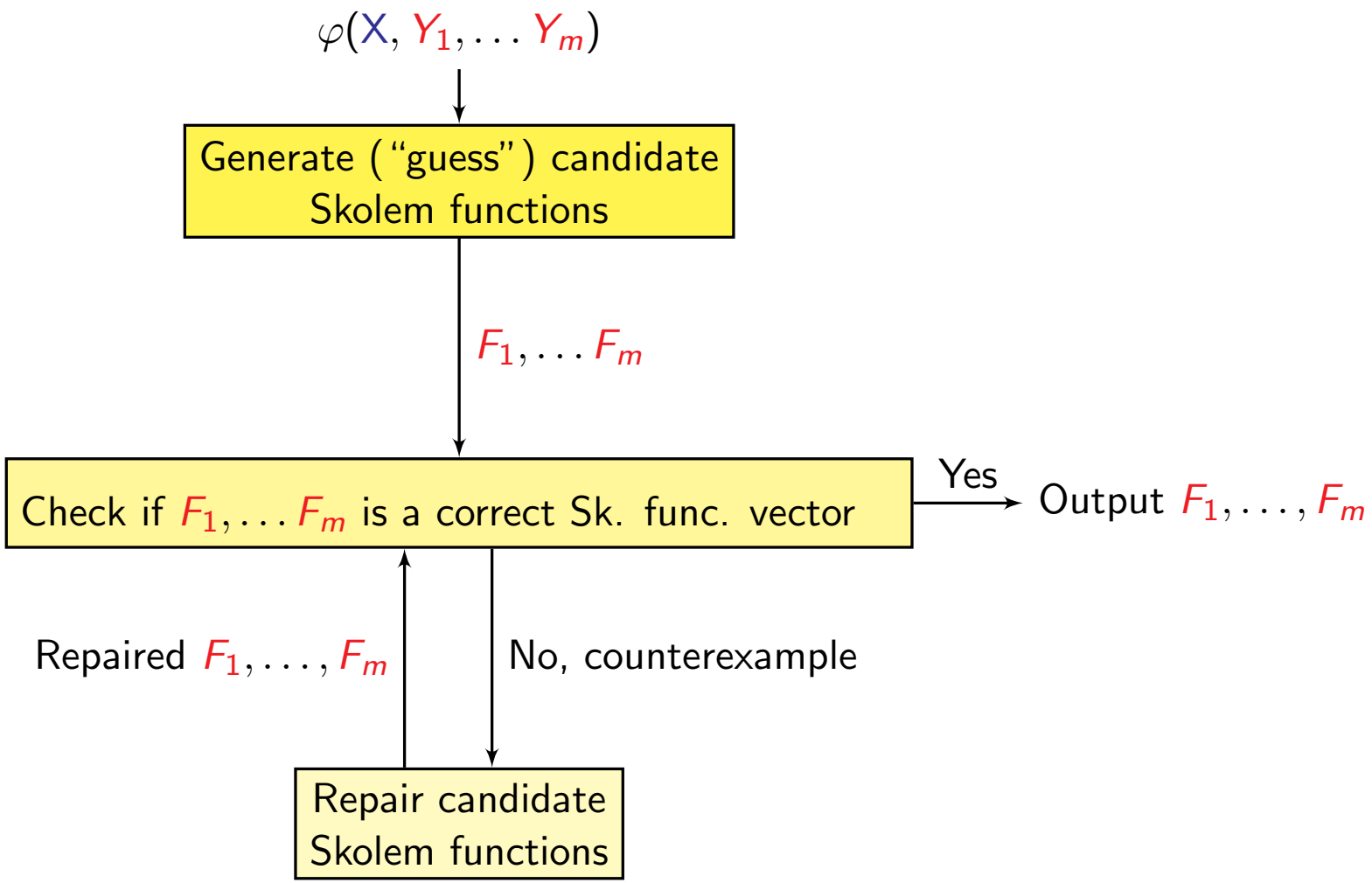
Guess-Check-Repair: A Useful Paradigm



Guess-Check-Repair: A Useful Paradigm



Guess-Check-Repair: A Useful Paradigm

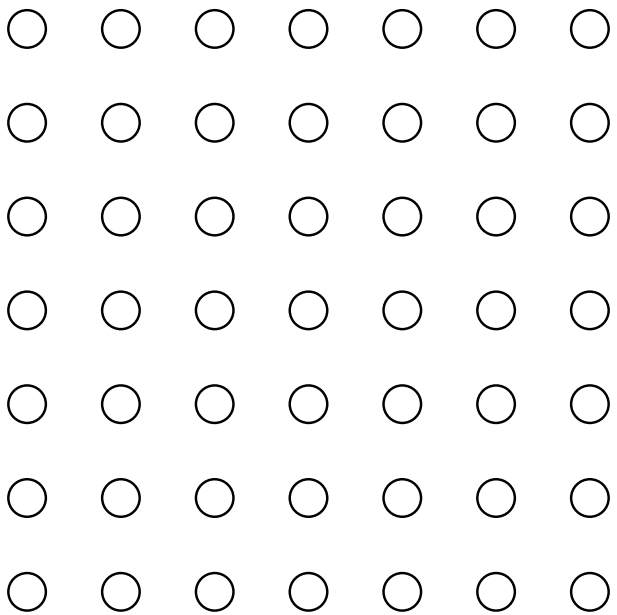


"Guess"-ing candidate Skolem functions ($|Y| = 1$)

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

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

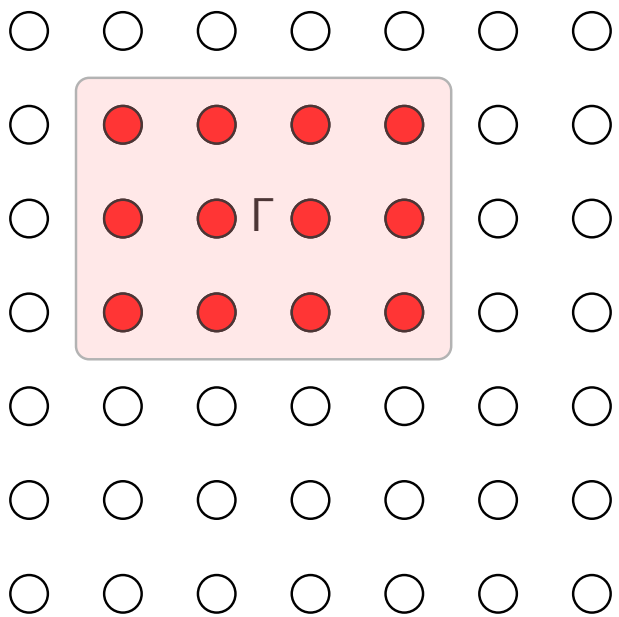
Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



— Set of all valuations of X.

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

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

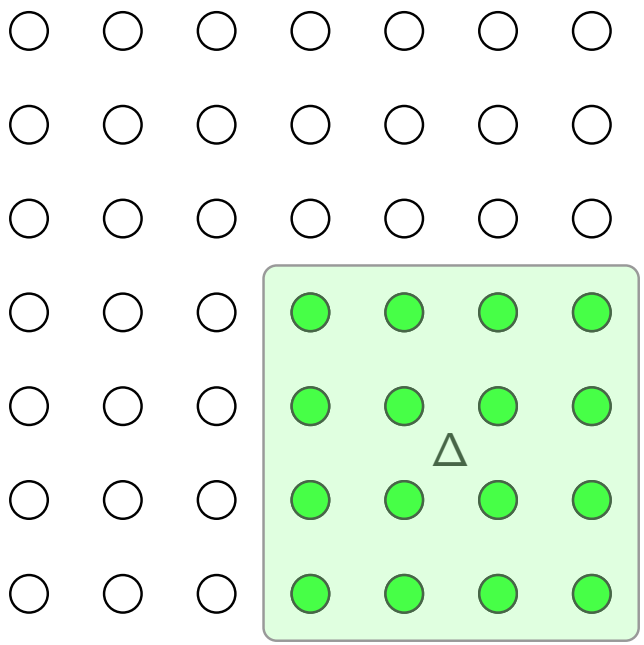


— Can't set y to 1 to satisfy φ: $\Gamma(X) \triangleq \neg\varphi(X, y)[y \mapsto 1]$

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 $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

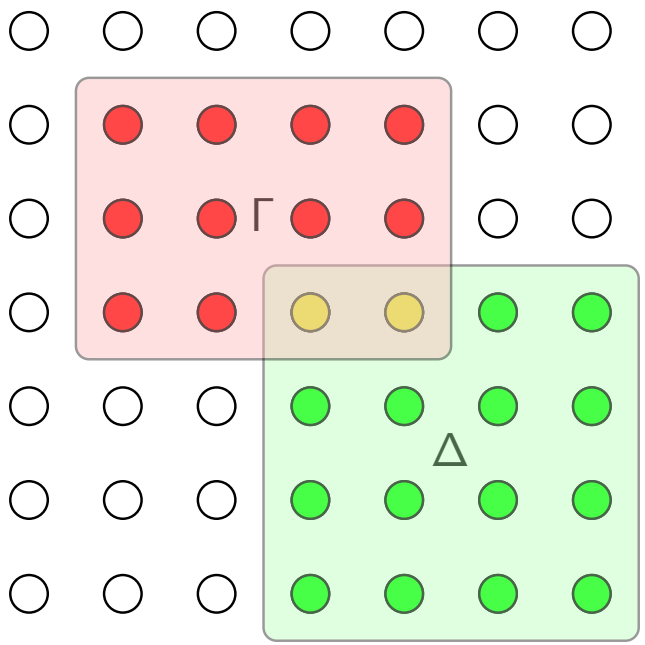


— Can't set y to 0 to satisfy φ: $\Delta(X) \triangleq \neg\varphi(X, y)[y \mapsto 0]$

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 ($|Y| = 1$)

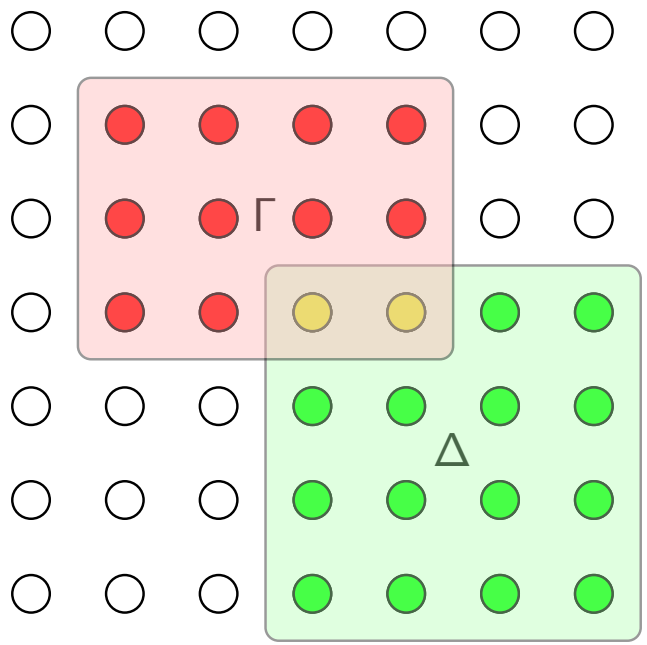
Find $F(X)$ such that $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



— 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 $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$



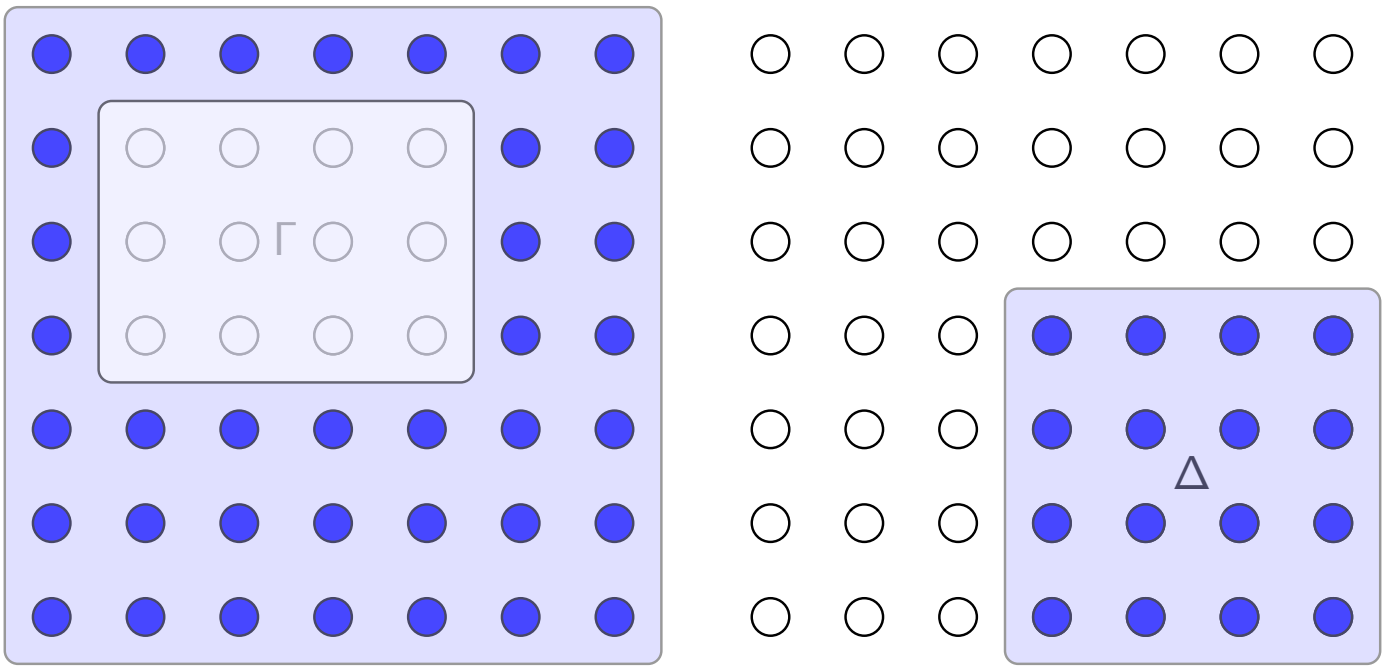
Lemma [Trivedi'03, Jiang'09,Fried et al'16]

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 $\exists y\ \varphi(X, y) \equiv \varphi(X, F(X))$

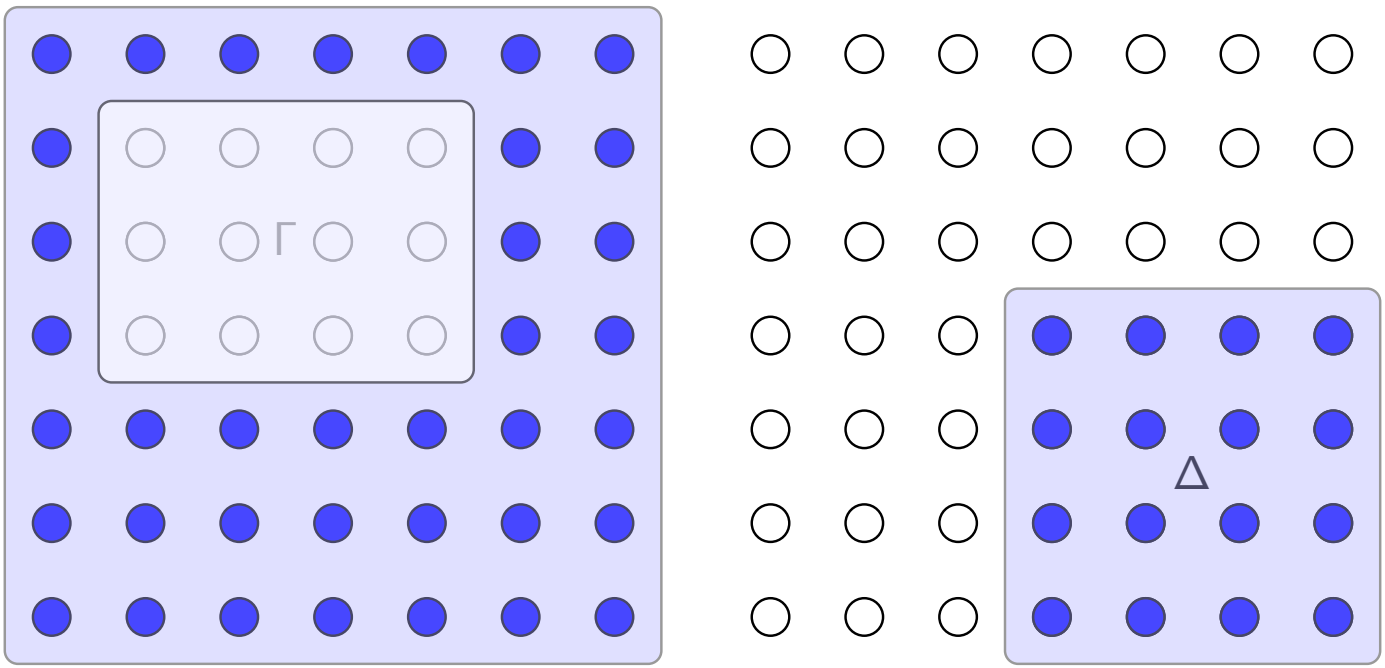


— Specific interpolants of $(\Delta \setminus \Gamma)$ & $(\Gamma \setminus \Delta)$

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

"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: $(|Y| = 2)$

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

"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

"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: ($|Y| = 2$)

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. $F_1(X) = \neg\Gamma^{y_1}(X) = 1$
- To find Skolem function for y_2, consider y_2 as sole output in $\varphi(X, F_1(X), y_2)$
  - E.g. $\varphi(X, 1, y_2) = \neg y_2$
  - $\Gamma^{y_2}(X) = \neg\varphi(X, 1, 1) = 1$; $\Delta^{y_2}(X) = \neg\varphi(X, 1, 0) = 0$
  - $F_2(X) = \neg\Gamma^{y_2}(X) = 0$

"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 $G_m(X, y_1, \ldots y_{m-1})$

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

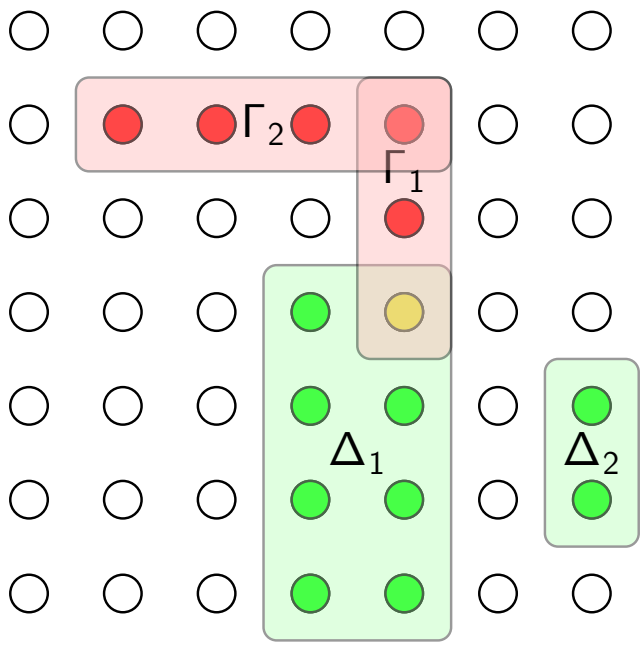
Suppose $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \wedge \ \varphi_2(X, Y)$, where $Y = y_1, \ldots y_m$

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

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

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

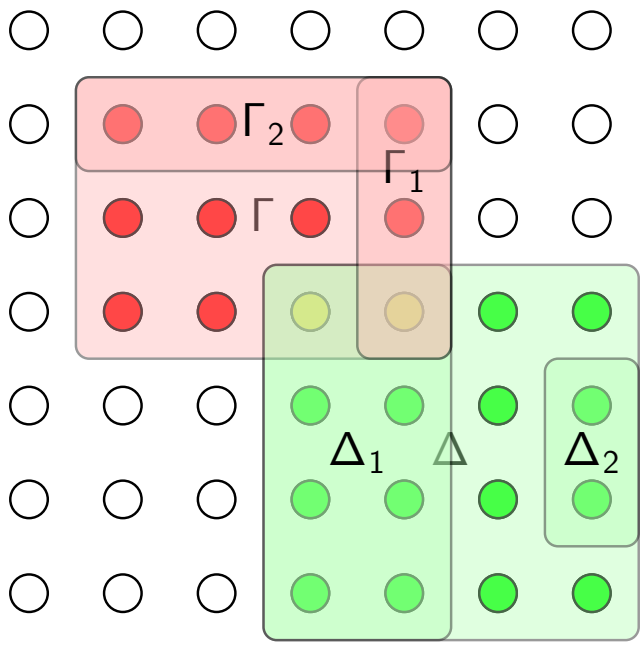
$$\Gamma_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 1, y_2 \ldots y_m) \qquad \Delta_1^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_1(X, 0, \ldots)$$
$$\Gamma_2^{y_1} \triangleq \neg\exists y_2 \ldots y_m \; \varphi_2(X, 1, \ldots) \qquad \Delta_2^{y_1} \triangleq \neg\exists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



"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 $\varphi(X, Y) \equiv \varphi_1(X, Y) \ \vee \ \varphi_2(X, Y)$, 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 $\gamma_1^{y_i} \Rightarrow \Gamma_1^{y_i}$; $\delta_1^{y_i} \Rightarrow \Delta_1^{y_i}$

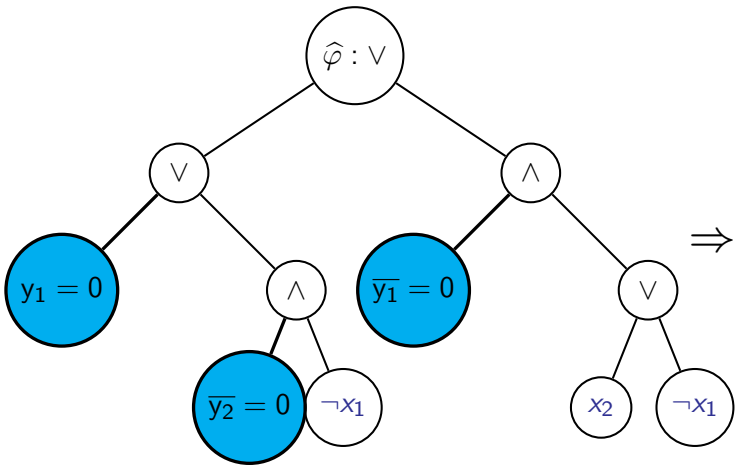
"Guess"-ing with under-approximations of Γ, Δ

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

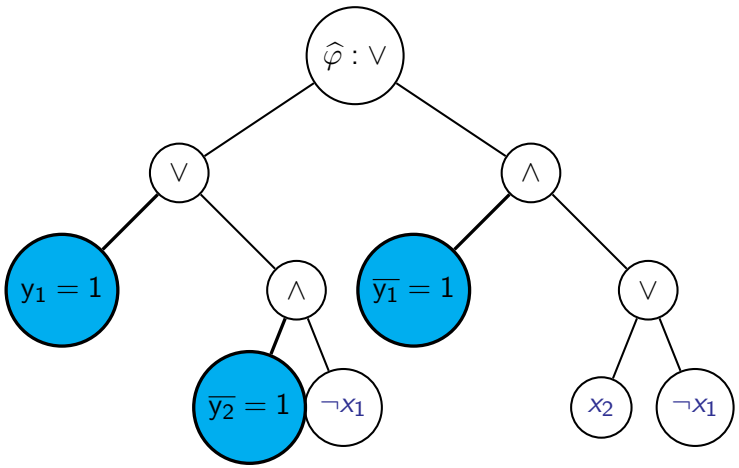
"Guess"-ing with under-approximations of Γ, Δ

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

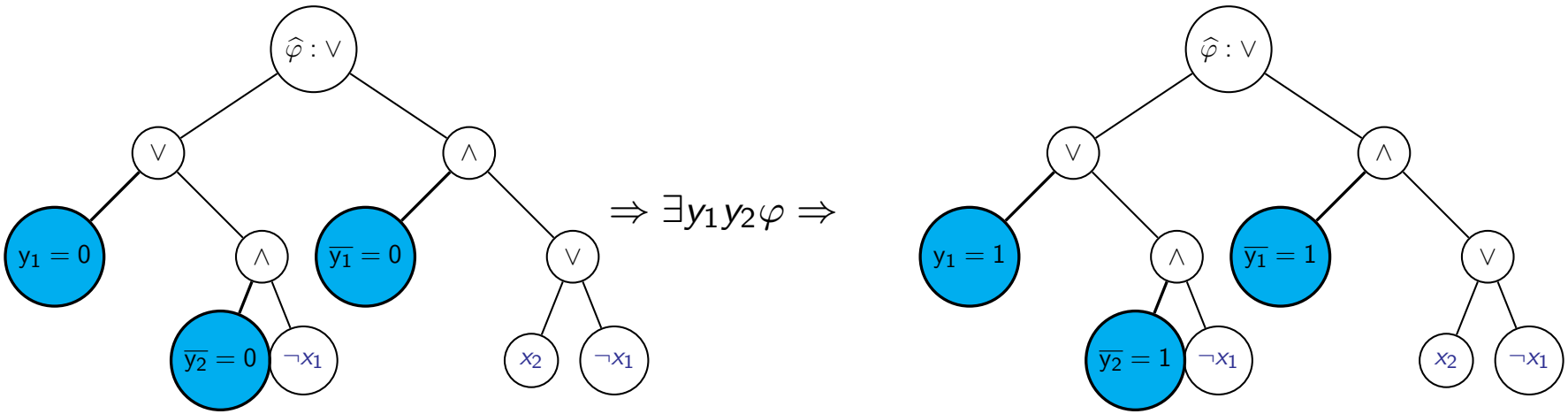
Illustrating Approximations



$\Rightarrow \exists y_1 y_2 \varphi \Rightarrow$



Illustrating Approximations



- $\widehat{\varphi}(x_1 \ldots x_n, \overbrace{0..0}^{i}, y_{i+1} \ldots y_m, \overbrace{0..0}^{i}, \neg y_{i+1} \ldots \neg y_m) \Rightarrow \exists y_1 \ldots y_i \ \varphi(\ldots)$
- $\widehat{\varphi}(x_1 \ldots x_n, \overbrace{1..1}^{i}, y_{i+1} \ldots y_m, \overbrace{1..1}^{i}, \neg y_{i+1} \ldots \neg y_m) \Leftarrow \exists y_1 \ldots y_i \ \varphi(\ldots)$

Checking correctness of "guess"-ed Skolem functions

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

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

Checking correctness of "guess"-ed Skolem functions

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?

Checking correctness of "guess"-ed Skolem functions

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! [ACGKS'15]

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

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

Checking correctness of "guess"-ed Skolem functions

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! [ACGKS'15]

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

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

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

Checking correctness of "guess"-ed Skolem functions

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! [ACGKS'15]

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

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

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

Checking correctness of "guess"-ed Skolem functions

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! [ACGKS'15]

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

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

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector
- Suppose σ: satisfying assignment of ε
  - $\varphi(\sigma[X], \sigma[Y'])] = 1, \quad \sigma[Y] = F(\sigma[X]), \quad \varphi(\sigma[X], \sigma[Y]) = 0$

Checking correctness of "guess"-ed Skolem functions

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! [ACGKS'15]

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

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

- ε unsatisfiable iff $F_1, \ldots F_m$ is correct Skolem function vector
- Suppose σ: satisfying assignment of ε
  - $\varphi(\sigma[X],\sigma[Y'])=1, \quad \sigma[Y]=F(\sigma[X]), \quad \varphi(\sigma[X],\sigma[Y])=0$
  - σ is **counterexample** to the claim that $F_1, \ldots F_m$ is a correct Skolem function vector

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$ Over-approx

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$

Under-approximations

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\quad \gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

- $\sigma \models \mu$ **... μ generalizes σ**

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

- $\sigma \models \mu$ **... μ generalizes σ**
- $\mu \Rightarrow \gamma_j \wedge \delta_j$

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

- $\sigma \models \mu$ **... μ generalizes σ**
- $\mu \Rightarrow \gamma_j \wedge \delta_j$
  - $\Rightarrow \forall y_j \ldots \forall y_m \, \neg\varphi(X, y_1, \ldots y_{j-1}, y_j, y_{j+1}, \ldots y_m)$

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

- $\sigma \models \mu$ **... μ generalizes σ**
- $\mu \Rightarrow \gamma_j \wedge \delta_j$
  - $\Rightarrow \forall y_j \ldots \forall y_m \, \neg\varphi(X, y_1, \ldots y_{j-1}, y_j, y_{j+1}, \ldots y_m)$
  - If $\pi \models \mu$, no extension of π satisfies φ **... counterexample**

Counterexample Generalization

Recall: Skolem functions guessed from approximations of $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{i-1}, y_i, y_{i+1}, \ldots y_m)$

- Let $\exists y_{i+1} \ldots \exists y_m \, \varphi(X, Y) \Rightarrow \Theta_i(X, y_1, \ldots y_{i-1}, y_i)$
- Let $\delta_i = \neg\Theta_i|_{y_i=0}$; $\gamma_i = \neg\Theta_i|_{y_i=1}$
- Initial guess $G_i(X, y_1, \ldots y_{i-1}) \in \{\delta_i, \neg\gamma_i\}$ **... 1-sided error**
  - $G_i = \delta_i$ **cannot err** if it evaluates to 1
  - $G_i = \neg\gamma_i$ **cannot err** if it evaluates to 0

Generalized counterexample

Given $\sigma \models \varepsilon(X, Y, Y')$ and δ_i, γ_i for $1 \leq i \leq m$
Find function $\mu(X, y_1, \ldots y_{j-1})$ for some $j \in \{1, \ldots m\}$ s.t.

- $\sigma \models \mu$ **... μ generalizes σ**
- $\mu \Rightarrow \gamma_j \wedge \delta_j$
  - $\Rightarrow \forall y_j \ldots \forall y_m \, \neg\varphi(X, y_1, \ldots y_{j-1}, y_j, y_{j+1}, \ldots y_m)$
  - If $\pi \models \mu$, no extension of π satisfies φ **... counterexample**

Must ensure that $(X, G_1, \ldots G_{j-1})$ never evaluates to π

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$
  - **Only** source of error: under-approximation of $\neg\exists y_j, \ldots \exists y_m \, \varphi(X, y_1, \ldots y_{j-2}, y_{j-1}, y_j, \ldots y_m)$

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$
  - **Only** source of error: under-approximation of $\neg\exists y_j, \ldots \exists y_m\, \varphi(X, y_1, \ldots y_{j-2}, y_{j-1}, y_j, \ldots y_m)$
  - Repair: **Expand under-approximation**
    - If G_{j-1} is $\neg\gamma_{j-1}$, $\quad \gamma_{j-1} \leftarrow \gamma_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$
    - If G_{j-1} is δ_{j-1}, $\quad \delta_{j-1} \leftarrow \delta_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$
  - **Only** source of error: under-approximation of $\neg\exists y_j, \ldots \exists y_m\, \varphi(X, y_1, \ldots y_{j-2}, y_{j-1}, y_j, \ldots y_m)$
  - Repair: **Expand under-approximation**
    - If G_{j-1} is $\neg\gamma_{j-1}$, $\quad \gamma_{j-1} \leftarrow \gamma_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$
    - If G_{j-1} is δ_{j-1}, $\quad \delta_{j-1} \leftarrow \delta_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$

Counter-example guided repair by expanding δ_i's and γ_i's.

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$
  - **Only** source of error: under-approximation of $\neg\exists y_j, \ldots \exists y_m\, \varphi(X, y_1, \ldots y_{j-2}, y_{j-1}, y_j, \ldots y_m)$
  - Repair: **Expand under-approximation**
    - If G_{j-1} is $\neg\gamma_{j-1}$, $\quad \gamma_{j-1} \leftarrow \gamma_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$
    - If G_{j-1} is δ_{j-1}, $\quad \delta_{j-1} \leftarrow \delta_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$

Counter-example guided repair by expanding δ_i's and γ_i's.

Expansion-based repair

Repairing "guess"-ed candidate Skolem functions

- Every model of $\mu(X, y_1, \ldots y_{j-1})$ gives a problematic combination of $G_1, \ldots G_{j-1}$ values
- Flip G_{j-1} whenever μ holds
  - Recall $G_{j-1} \in \{\neg\gamma_{j-1}, \delta_{j-1}\}$
  - **Only** source of error: under-approximation of $\neg\exists y_j, \ldots \exists y_m\, \varphi(X, y_1, \ldots y_{j-2}, y_{j-1}, y_j, \ldots y_m)$
  - Repair: **Expand under-approximation**
    - If G_{j-1} is $\neg\gamma_{j-1}$, $\quad \gamma_{j-1} \leftarrow \gamma_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$
    - If G_{j-1} is δ_{j-1}, $\quad \delta_{j-1} \leftarrow \delta_{j-1} \vee \mu|_{\sigma[y_{j-1}]}$

Counter-example guided repair by expanding δ_i's and γ_i's.

Expansion-based repair

Simple argument for termination – expansions can't go on forever

Counterexample-guided repair

Can we always find μ?

Counterexample-guided repair

Can we always find μ?

Yes!!!

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$
  - $\mu_{\mathrm{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$
  - $\mu_{\mathrm{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$

... Why?

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$
  - $\mu_{\mathrm{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$
  - $\mu_{\mathrm{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\mathrm{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\mathrm{worst-case}}$
  - $\mu_{\mathrm{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$
- Choose smallest j s.t. $\sigma \models \delta_j \wedge \gamma_j$

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\text{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\text{worst-case}}$
  - $\mu_{\text{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$
- Choose smallest j s.t. $\sigma \models \delta_j \wedge \gamma_j$

Repairing multiple Skolem functions

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\text{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\text{worst-case}}$
  - $\mu_{\text{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$
- Choose smallest j s.t. $\sigma \models \delta_j \wedge \gamma_j$

Repairing multiple Skolem functions

Observation: $(\mu|_{y_{j-1}=0} \ \wedge \ \mu|_{y_{j-1}=1}) \ \Rightarrow \ \ \forall y_{j-1} \ldots \forall y_m \neg\varphi$

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\text{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\text{worst-case}}$
  - $\mu_{\text{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$
- Choose smallest j s.t. $\sigma \models \delta_j \wedge \gamma_j$

Repairing multiple Skolem functions

Observation: $(\mu|_{y_{j-1}=0} \ \wedge \ \mu|_{y_{j-1}=1}) \ \Rightarrow \ \forall y_{j-1} \ldots \forall y_m \neg\varphi$

- Update μ to $\mu|_{y_{j-1}=0} \wedge \mu|_{y_{j-1}=1}$ and repeat with j decremented

Counterexample-guided repair

Can we always find μ?

Yes!!!

- $\mu_{\text{worst-case}}$: ClausalForm($\sigma[X, y_1, \ldots y_{m-1}]$)
  - $\sigma \models \mu_{\text{worst-case}}$
  - $\mu_{\text{worst-case}} \Rightarrow \gamma_m \wedge \delta_m$ **... Why?**
    - $G_m \in \{\varphi|_{y_m=1},\ \neg\varphi|_{y_m=0}\}$ **always correct** Skolem function
    - And yet, $\varphi(\sigma[X, y_1, \ldots y_{m-1}], G_m(\sigma[X, y_1, \ldots y_{m-1}])) = 0$
- Choose smallest j s.t. $\sigma \models \delta_j \wedge \gamma_j$

Repairing multiple Skolem functions

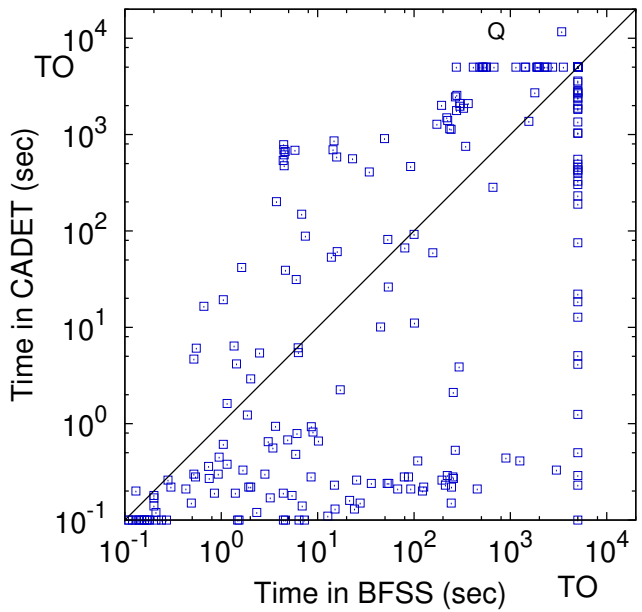
Observation: $(\mu|_{y_{j-1}=0} \ \wedge \ \mu|_{y_{j-1}=1}) \ \Rightarrow \ \forall y_{j-1} \ldots \forall y_m \neg\varphi$

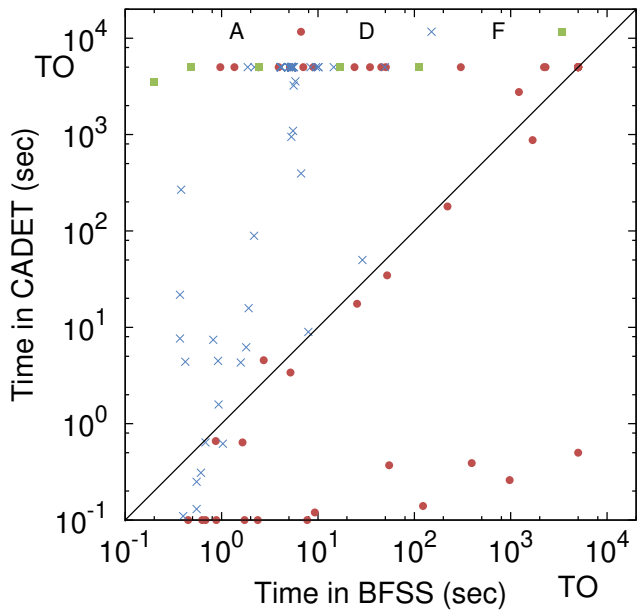
- Update μ to $\mu|_{y_{j-1}=0} \wedge \mu|_{y_{j-1}=1}$ and repeat with j decremented
- Can repair multiple Skolem functions starting from one counterexample

Experimental Comparison

BFSS (Cex-guided-repair) vis-a-vis CADET (incremental determinization) [Rabe & Seshia'16]

[Comparisons with other tools in FMSD 2020 paper]



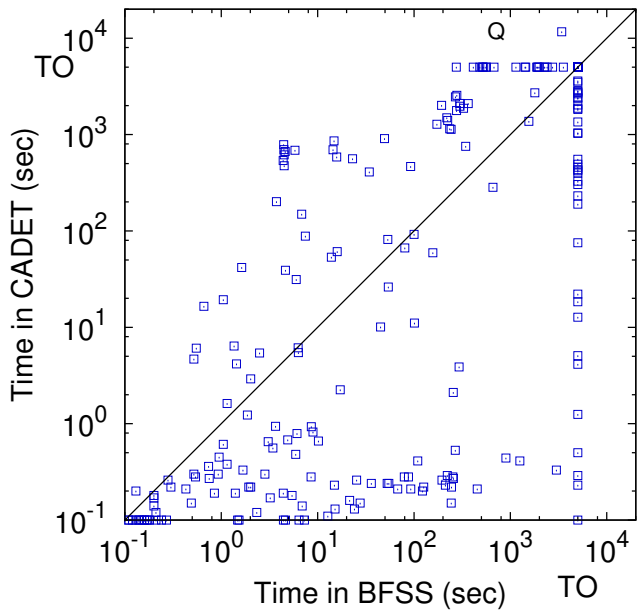


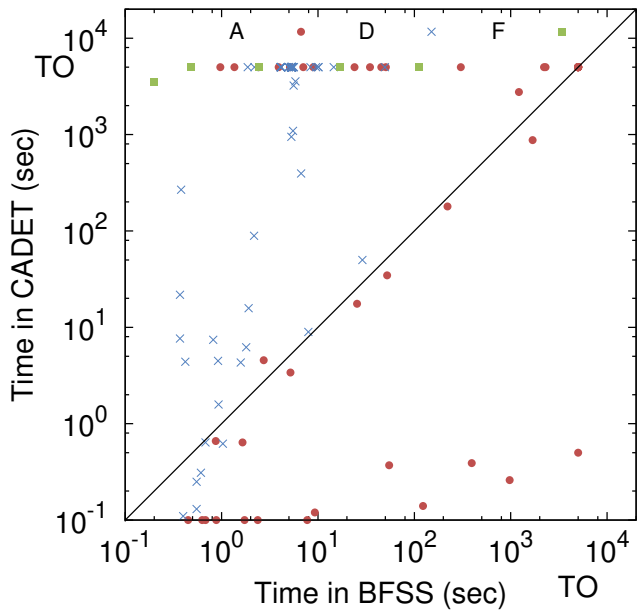
Q: QBFEval, A: Arithmetic, F: Factorization, D: Disjunctive Decomposition. TO: Timeout (3600 sec)

Experimental Comparison

BFSS (Cex-guided-repair) vis-a-vis CADET (incremental determinization) [Rabe & Seshia'16]

[Comparisons with other tools in FMSD 2020 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

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before
- "Repair" using unsatisfiable cores

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before
- "Repair" using unsatisfiable cores
  - $\sigma \models \varepsilon(X, Y, Y')$
  - Candidates to repair: F_i s.t. $y_i \neq Y'_i$

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before
- "Repair" using unsatisfiable cores
  - $\sigma \models \varepsilon(X, Y, Y')$
  - Candidates to repair: F_i s.t. $y_i \neq Y'_i$
  - $R_i(X, Y) = \varphi(X, Y) \wedge \bigwedge_{i=1}^{n}(x_i \Leftrightarrow \sigma[x_i]) \wedge (y_i \Leftrightarrow \sigma[Y'_i])$

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

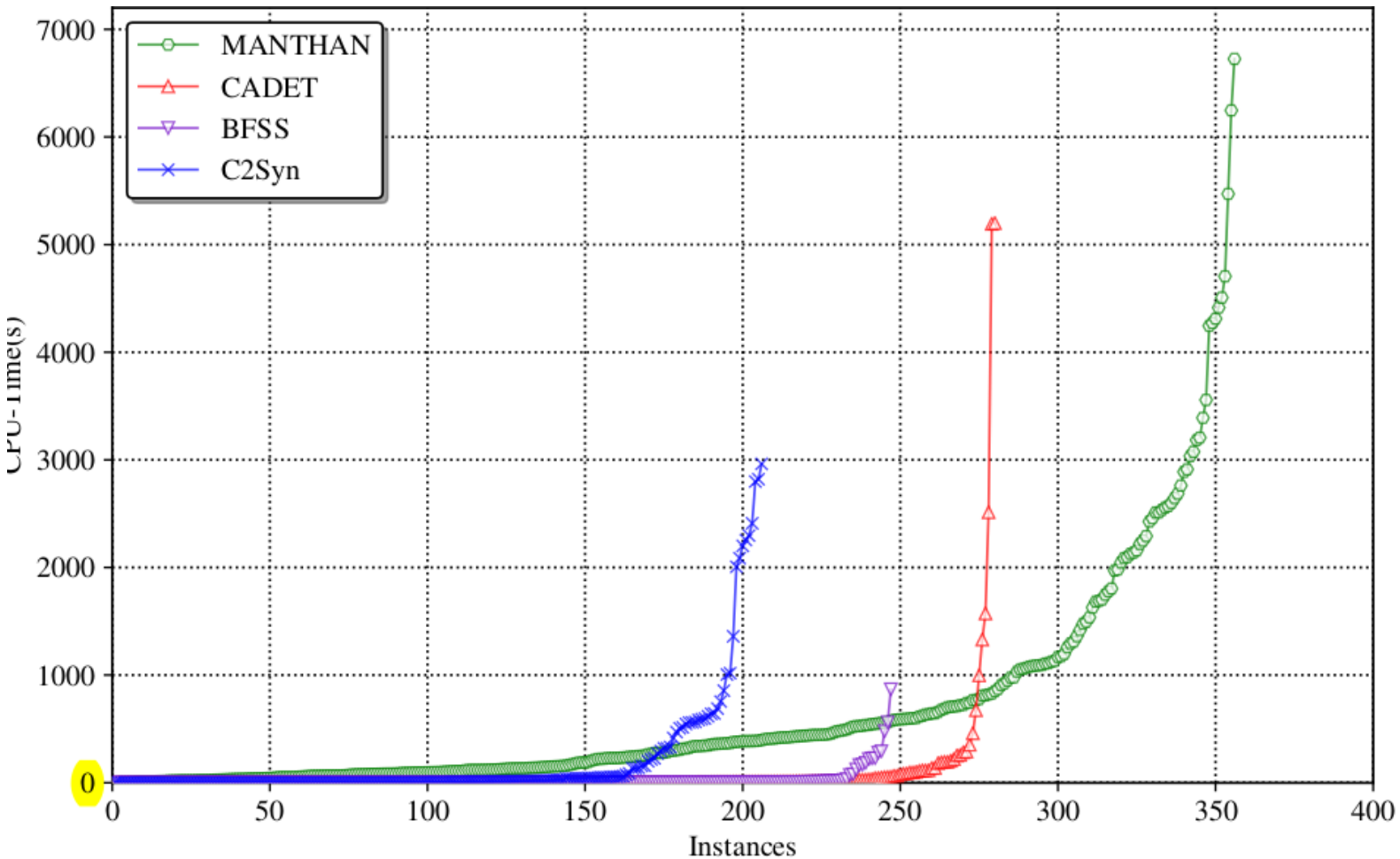
- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before
- "Repair" using unsatisfiable cores
  - $\sigma \models \varepsilon(X, Y, Y')$
  - Candidates to repair: F_i s.t. $y_i \neq Y'_i$
  - $R_i(X, Y) = \varphi(X, Y) \wedge \bigwedge_{i=1}^{n}(x_i \Leftrightarrow \sigma[x_i]) \wedge (y_i \Leftrightarrow \sigma[Y'_i])$
  - Find repair formula β from Unsat Core of $R_i(X, Y)$

Another Flavour of Guess-Check-Repair

Manthan: Golia et al [GMR'20]

- "Guess" using machine learning techniques
  - Sample solutions of $\varphi(X, Y)$: Data-driven!!!
  - Learn "Decision Tree" for y_i in terms of $X, y_1, \ldots y_{i-1}$
    - Ok to not match data exactly
    - Can have 2-sided errors!
    - Repair learnt decision trees later!
    - Often results in small decision trees
- "Check" using error formula $\varepsilon(X, Y, Y')$ as before
- "Repair" using unsatisfiable cores
  - $\sigma \models \varepsilon(X, Y, Y')$
  - Candidates to repair: F_i s.t. $y_i \neq Y'_i$
  - $R_i(X, Y) = \varphi(X, Y) \wedge \bigwedge_{i=1}^{n}(x_i \Leftrightarrow \sigma[x_i]) \wedge (y_i \Leftrightarrow \sigma[Y'_i])$
  - Find repair formula β from Unsat Core of $R_i(X, Y)$
  - Update
    - F_i to $F_i \vee \beta$ if $\sigma[y_i] = 0$
    - F_i to $F_i \wedge \neg\beta$ if $\sigma[y_i] = 1$

Some more experimental comparisons [GMR20]



[Courtesy "Manthan: A Data Driven Approach to Boolean Functional Synthesis", Golia et al, CAV 2020]

Conclusion

- Boolean functional synthesis has diverse applications, including in temporal synthesis

Conclusion

- Boolean functional synthesis has diverse applications, including in temporal synthesis
- Guess-check-repair: a powerful paradigm

Conclusion

- Boolean functional synthesis has diverse applications, including in temporal synthesis
- Guess-check-repair: a powerful paradigm
- Different approaches to guessing possible
  - Based on approximations of quantifier elimination
  - Based on machine learning

Conclusion

- Boolean functional synthesis has diverse applications, including in temporal synthesis
- Guess-check-repair: a powerful paradigm
- Different approaches to guessing possible
  - Based on approximations of quantifier elimination
  - Based on machine learning
- Different approaches to counterexample-guided repair possible
  - Expansion of under-approximations
  - Unsatisfiable core based repair

Conclusion

- Boolean functional synthesis has diverse applications, including in temporal synthesis
- Guess-check-repair: a powerful paradigm
- Different approaches to guessing possible
  - Based on approximations of quantifier elimination
  - Based on machine learning
- Different approaches to counterexample-guided repair possible
  - Expansion of under-approximations
  - Unsatisfiable core based repair
- Recent results (BFSS, Manthan) extremely promising!!!