Solving Higher-Order Quantified Boolean Satisfiability via Higher-Order Model Checking

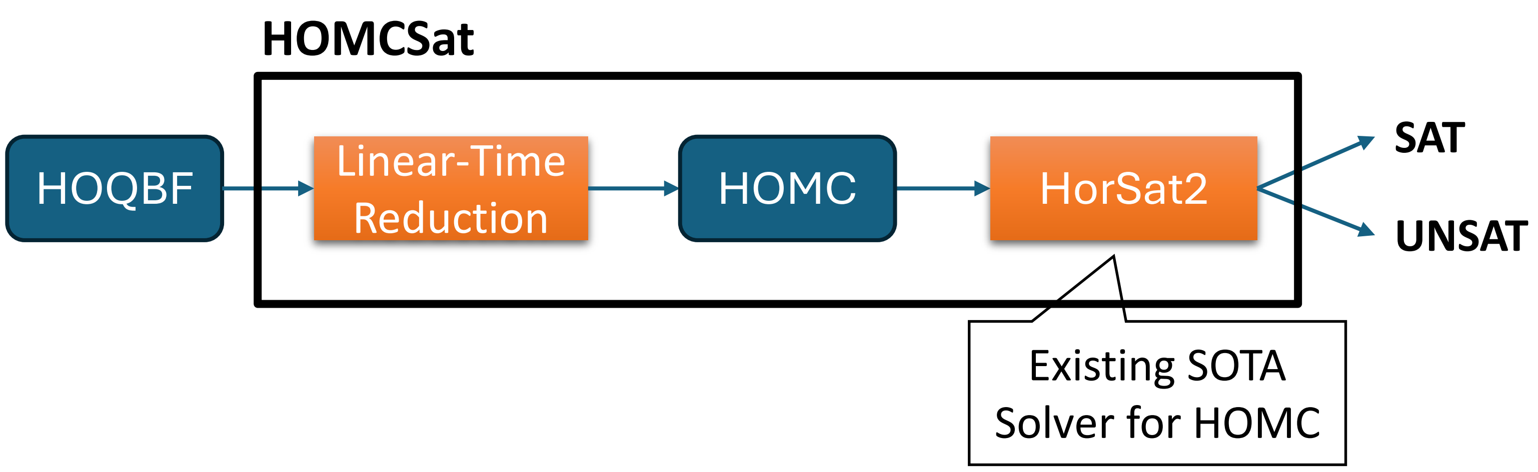
<u>Hiroshi Unno</u> (Tohoku University)

Takeshi Tsukada (Chiba University)

Jie-Hong Roland Jiang (National Taiwan University)

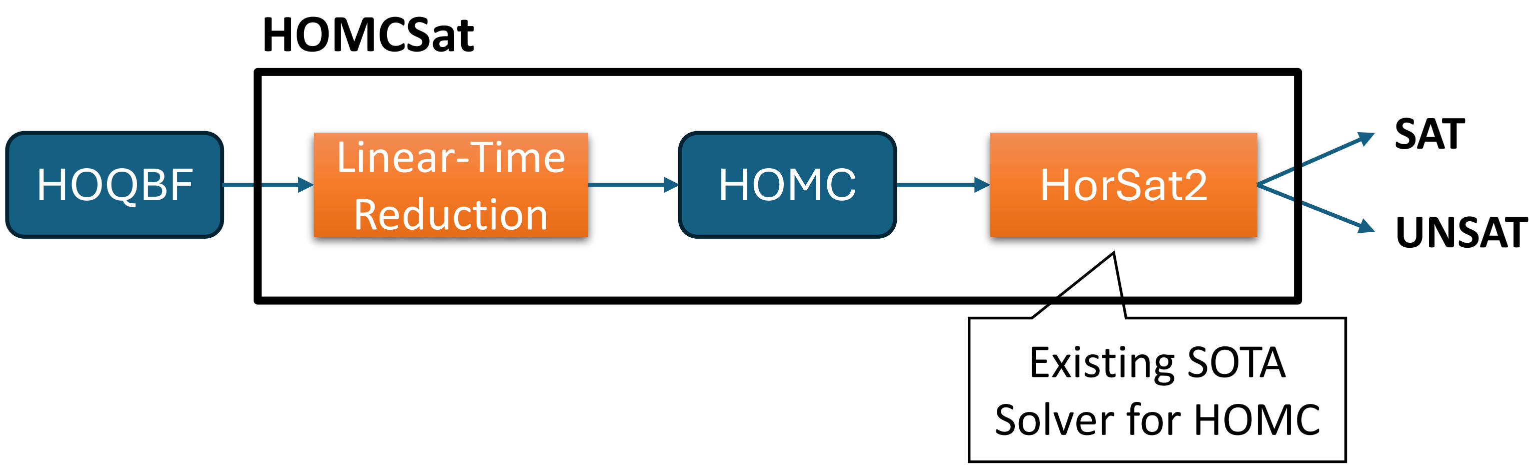
Main Contribution

- We present the first solver **HOMCSat** for **Higher-Order Quantified Boolean Formulas (HOQBF)** based on a new **linear-time reduction** to **Higher-Order Model Checking (HOMC)**



Main Contribution

- We present the first solver **HOMCSat** for **Higher-Order Quantified Boolean Formulas (HOQBF)** based on a new **linear-time reduction** to **Higher-Order Model Checking (HOMC)**



Higher-Order QBF (HOQBF)

- A generalization of **Quantified Boolean Formulas (QBF)** with **higher-order quantifiers**

$$\varphi ::= \mathrm{true} \mid \mathrm{false} \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \leftrightarrow \varphi_2$$
$$\mid \forall x^A.\varphi \mid \exists x^A.\varphi \mid f(x_1, \ldots, x_n)$$

$$A ::= \mathrm{bool} \mid A_1 \to A_2$$

function type

Apply $f^{A_1 \to \ldots \to A_n \to \mathrm{bool}}$ to arguments $x_1^{A_1}, \ldots, x_n^{A_n}$

Def.1 HOQBF Satisfiability (HOSAT)

Ask if a closed formula φ is equivalent to true.

Expressiveness of HOQBF

$2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}}\}k$

- Capable of succinct encoding of k-EXPTIME problems [Chistikov+'22]
  - Potential applications: memory consistency verification, planning for multi-agent systems, secure system synthesis, and solving quantified bit-vector arithmetic problems
- **QBF** is a fragment where type A is fixed to bool

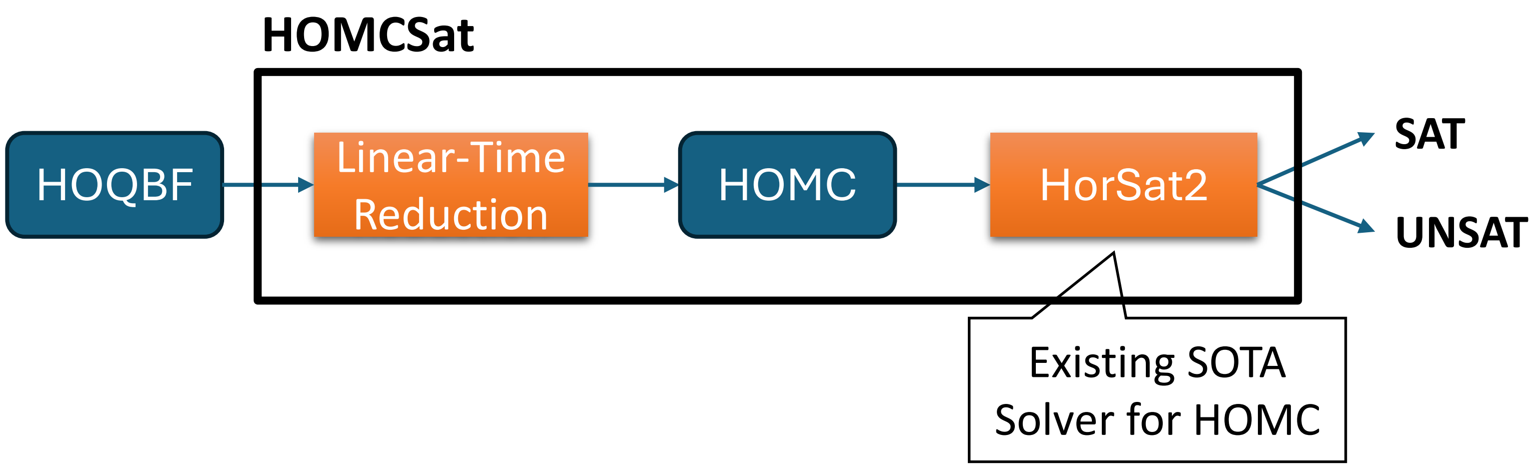
Ex.1 $\forall x^{\mathrm{bool}}.\exists y^{\mathrm{bool}}.\forall z^{\mathrm{bool}}.(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$

- **Dependency QBF (DQBF)** and **Second-Order QBF (SOQBF)** [Jiang'23] are strict fragments

Ex.2 $\forall f^{\mathrm{bool}\to\mathrm{bool}}.\exists g^{\mathrm{bool}\to\mathrm{bool}}.\exists z^{\mathrm{bool}}.\left(f\left(g(z)\right)\right) \leftrightarrow z$

Main Contribution

- We present the first solver **HOMCSat** for **Higher-Order Quantified Boolean Formulas (HOQBF)** based on a new **linear-time reduction** to **Higher-Order Model Checking (HOMC)**



Higher-Order Model Checking (HOMC)

- A generalization from model checking of **while-programs** to that of **higher-order functional programs**

$$t ::= \mathrm{true} \mid \mathrm{false} \mid x^A \mid \mathrm{if}\ t_1\ \mathrm{then}\ t_2\ \mathrm{else}\ t_3 \mid t_1\ t_2$$

$$\mid \lambda x^A.t \mid \mathrm{let\ rec}\ f^{A_1 \to A_2} = t_1\ \mathrm{in}\ t_2$$

apply t_1 to t_2

function that takes an argument x and returns t

let-binding (f can occur recursively in t_1)

Ex.3 Functional program that returns the truth value of Ex.1

$\text{let rec not} = \lambda x^{\text{bool}}.\,\text{if } x \text{ then false else true in}$
$\text{let rec and} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then } y \text{ else false in}$
$\text{let rec or} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then true else } y \text{ in}$

Ex.1 $\forall x^{\text{bool}}.\,\exists y^{\text{bool}}.\,\forall z^{\text{bool}}.\,(x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)$

Ex.3 Functional program that returns the truth value of Ex.1

$$\text{let rec not} = \lambda x^{\text{bool}}.\,\text{if } x \text{ then false else true in}$$
$$\text{let rec and} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then } y \text{ else false in}$$
$$\text{let rec or} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then true else } y \text{ in}$$
$$\text{let rec } f = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\lambda z^{\text{bool}}.$$
$$\quad \text{and (or (or } x\ y)\ z)\ \big(\text{or}\ \big(\text{or (not } x)\ (\text{not } y)\big)\ (\text{not } z)\big) \text{ in}$$

Ex.1 $\forall x^{\text{bool}}.\,\exists y^{\text{bool}}.\,\forall z^{\text{bool}}.\,\overbrace{(x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)}^{f}$

Ex.3 Functional program that returns the truth value of Ex.1

$\text{let rec not} = \lambda x^{\text{bool}}.\,\text{if}\ x\ \text{then false else true in}$

$\text{let rec and} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if}\ x\ \text{then}\ y\ \text{else false in}$

$\text{let rec or} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if}\ x\ \text{then true else}\ y\ \text{in}$

$\text{let rec}\ f = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\lambda z^{\text{bool}}.$

$\quad \text{and}\ (\text{or}\ (\text{or}\ x\ y)\ z)\ \big(\text{or}\ (\text{or}\ (\text{not}\ x)\ (\text{not}\ y))\ (\text{not}\ z)\big)\ \text{in}$

$\text{let rec}\ f_z = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{and}\ (f\ x\ y\ \text{true})\ (f\ x\ y\ \text{false})\ \text{in}$

$$\textbf{Ex.1}\quad \forall x^{\text{bool}}.\,\exists y^{\text{bool}}.\,\overbrace{\forall z^{\text{bool}}.\,\overbrace{(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)}^{f}}^{f_z}$$

Ex.3 Functional program that returns the truth value of Ex.1

$$\text{let rec not} = \lambda x^{\text{bool}}.\,\text{if } x \text{ then false else true in}$$
$$\text{let rec and} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then } y \text{ else false in}$$
$$\text{let rec or} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then true else } y \text{ in}$$
$$\text{let rec } f = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\lambda z^{\text{bool}}.$$
$$\quad \text{and (or (or } x\ y)\ z)\ \big(\text{or}\ (\text{or (not } x)\ (\text{not } y))\ (\text{not } z)\big) \text{ in}$$
$$\text{let rec } f_z = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{and } (f\ x\ y\ \text{true})\ (f\ x\ y\ \text{false}) \text{ in}$$
$$\text{let rec } f_{yz} = \lambda x^{\text{bool}}.\,\text{or } (f_z\ \text{true})\ (f_z\ \text{false}) \text{ in}$$

Ex.1 $\forall x^{\text{bool}}.\,\overbrace{\exists y^{\text{bool}}.\,\overbrace{\forall z^{\text{bool}}.\,\overbrace{(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)}^{f}}^{f_z}}^{f_{yz}}$

Ex.3 Functional program that returns the truth value of Ex.1

$\text{let rec not} = \lambda x^{\text{bool}}.\,\text{if } x \text{ then false else true in}$

$\text{let rec and} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then } y \text{ else false in}$

$\text{let rec or} = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{if } x \text{ then true else } y \text{ in}$

$\text{let rec } f = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\lambda z^{\text{bool}}.$

$\quad \text{and (or (or } x\ y)\ z) \left(\text{or}\left(\text{or (not } x) \text{ (not } y)\right) \text{(not } z)\right) \text{ in}$

$\text{let rec } f_z = \lambda x^{\text{bool}}.\,\lambda y^{\text{bool}}.\,\text{and } (f\ x\ y \text{ true})\ (f\ x\ y \text{ false}) \text{ in}$

$\text{let rec } f_{yz} = \lambda x^{\text{bool}}.\,\text{or } (f_z \text{ true})\ (f_z \text{ false}) \text{ in}$

$\text{and } \left(f_{yz} \text{ true}\right) \left(f_{yz} \text{ false}\right)$

Ex.1 $\forall x^{\text{bool}}.\,\exists y^{\text{bool}}.\,\forall z^{\text{bool}}.\,(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$

Higher-Order Model Checking

Def.2 **Higher-Order Model Checking (HOMC)**

Ask if a well-typed closed term t of type bool is not evaluated to false (i.e. the evaluation diverges or results in true).

Thm.1 **Decidability [Ong LICS'06]**

The higher-order model checking of order-n term is $(n-1)$-EXPTIME complete.

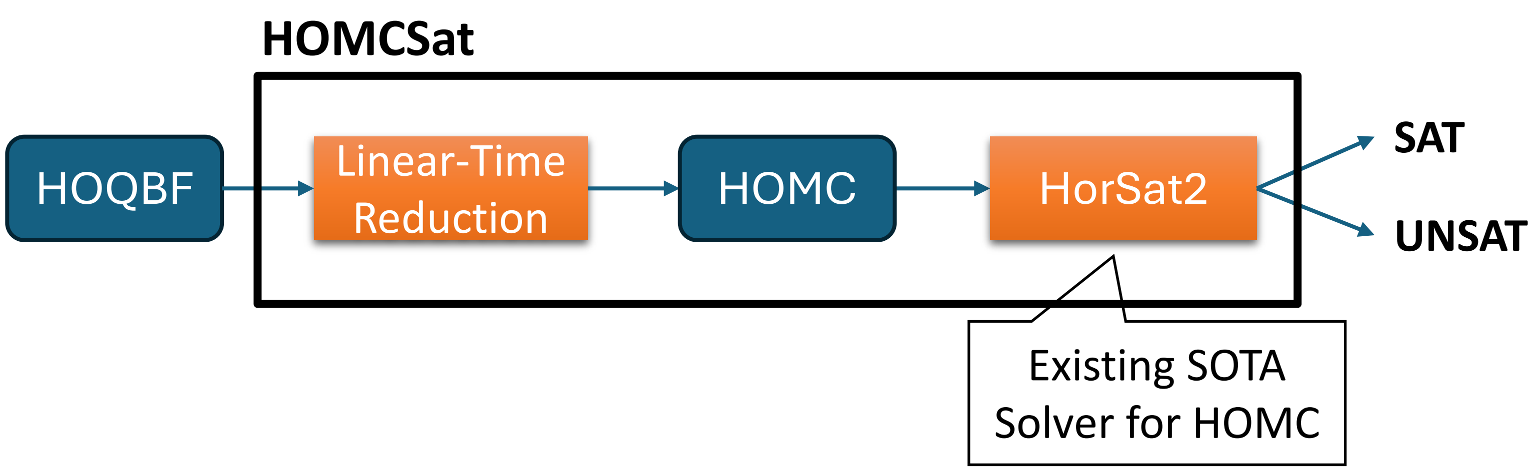
$$\mathrm{order}(\mathrm{bool}) = 1$$

$$\mathrm{order}(A_1 \to A_2) = \max\big(\mathrm{order}(A_1) + 1, \mathrm{order}(A_2)\big)$$

practical model checkers (e.g., **HorSat2**) have been developed and applied to formal verification, despite the high computational complexity

Main Contribution

- We present the first solver **HOMCSat** for **Higher-Order Quantified Boolean Formulas (HOQBF)** based on a new **linear-time reduction** to **Higher-Order Model Checking (HOMC)**



Reduction of HOSAT to HOMC

- **Thm.1** tells us that there exists a polynomial-time many-one reduction from order-n **HOSAT** to order-$(n+1)$ **HOMC**
- We present a more direct reduction that uses **HOMC** for reasoning about **higher-order quantifiers**

Thm.2

There exists a linear-time **HOSAT** to **HOMC** reduction $\varphi \mapsto P_\varphi$ where an order-n **HOQBF** formula is reduced to order $(n+1)$ **HOMC**.

$$P_{\text{true}} \triangleq \text{true} \quad P_{\text{false}} \triangleq \text{false} \quad P_{f(x_1,\ldots,x_n)} \triangleq f\ x_1 \ldots x_n$$

$$P_{\neg\varphi} \triangleq \text{not}\ P_{\varphi} \quad P_{\varphi_1 \wedge \varphi_2} \triangleq \text{and}\ P_{\varphi_1}\ P_{\varphi_2} \quad P_{\varphi_1 \vee \varphi_2} \triangleq \text{or}\ P_{\varphi_1}\ P_{\varphi_2}$$

$P_{\mathrm{true}} \triangleq \mathrm{true} \quad P_{\mathrm{false}} \triangleq \mathrm{false} \quad P_{f(x_1,\ldots,x_n)} \triangleq f\ x_1 \ldots x_n$

$P_{\neg\varphi} \triangleq \mathrm{not}\ P_{\varphi} \quad P_{\varphi_1 \wedge \varphi_2} \triangleq \mathrm{and}\ P_{\varphi_1}\ P_{\varphi_2} \quad P_{\varphi_1 \vee \varphi_2} \triangleq \mathrm{or}\ P_{\varphi_1}\ P_{\varphi_2}$

$$P_{\forall x^A.\varphi} \triangleq \mathrm{let\ rec}\ f = \lambda x^A. \\ \quad \mathrm{and}\ P_{\varphi}\left(\mathrm{if}\ \mathrm{ismax}_A\ x\ \mathrm{then\ true\ else}\ f(\mathrm{succ}_A\ x)\right) \\ \mathrm{in}\ f\ \mathrm{zero}_A$$

where ($\mathrm{zero}_A, \mathrm{succ}_A, \mathrm{ismax}_A$) is an **enumeration structure** for type A that satisfies: for some k,

- $\mathrm{ismax}_A\left(\mathrm{succ}_A^k\ \mathrm{zero}_A\right)$
- $A \cong \left\{\mathrm{zero}_A, \mathrm{succ}_A\ \mathrm{zero}_A, \ldots, \mathrm{succ}_A^k\ \mathrm{zero}_A\right\}$

$P_{\text{true}} \triangleq \text{true} \quad P_{\text{false}} \triangleq \text{false} \quad P_{f(x_1,\ldots,x_n)} \triangleq f\ x_1 \ldots x_n$

$P_{\neg\varphi} \triangleq \text{not}\ P_\varphi \quad P_{\varphi_1 \wedge \varphi_2} \triangleq \text{and}\ P_{\varphi_1}\ P_{\varphi_2} \quad P_{\varphi_1 \vee \varphi_2} \triangleq \text{or}\ P_{\varphi_1}\ P_{\varphi_2}$

$$P_{\forall x^A.\varphi} \triangleq \text{let rec}\ f = \lambda x^A.\ \text{and}\ P_\varphi\ (\text{if ismax}_A\ x\ \text{then true else}\ f(\text{succ}_A\ x))\ \text{in}\ f\ \text{zero}_A$$

$$P_{\exists x^A.\varphi} \triangleq \text{let rec}\ f = \lambda x^A.\ \text{or}\ P_\varphi\ (\text{if ismax}_A\ x\ \text{then false else}\ f(\text{succ}_A\ x))\ \text{in}\ f\ \text{zero}_A$$

where ($\text{zero}_A, \text{succ}_A, \text{ismax}_A$) is an **enumeration structure** for type A that satisfies: for some k,

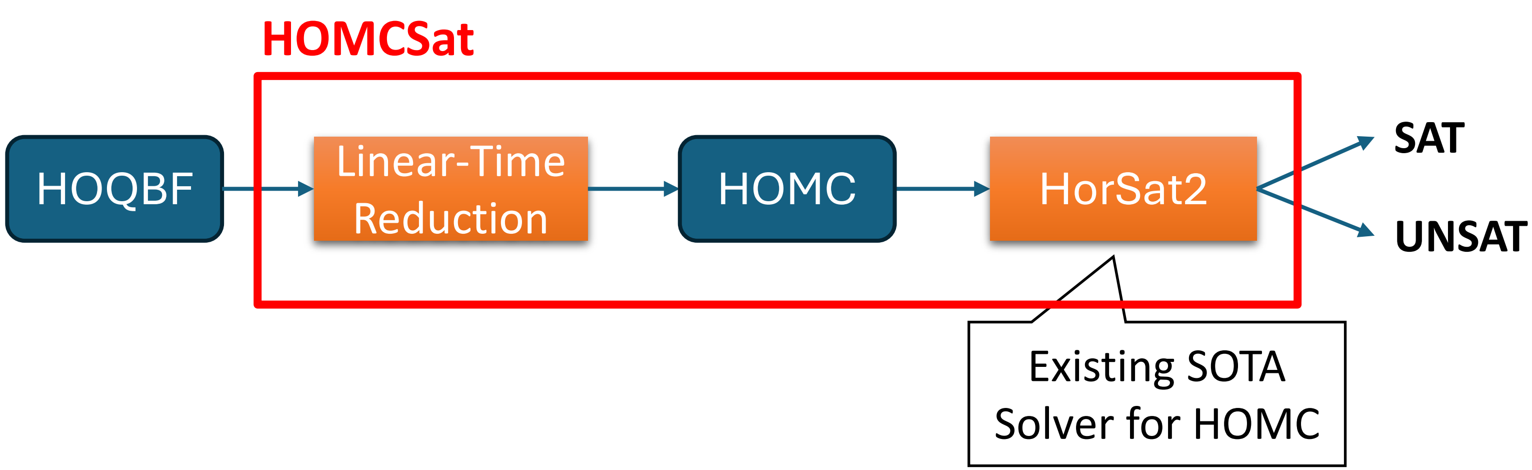
- $\text{ismax}_A(\text{succ}_A^k\ \text{zero}_A)$
- $A \cong \{\text{zero}_A, \text{succ}_A\ \text{zero}_A, \ldots, \text{succ}_A^k\ \text{zero}_A\}$

Inductive Definition of Enumeration Structures

- Case $\mathrm{bool} = \{\mathrm{false}, \mathrm{true}\}$
  - $\mathrm{zero}_{\mathrm{bool}} \triangleq \mathrm{false}$
  - $\mathrm{succ}_{\mathrm{bool}} \triangleq \lambda x^{\mathrm{bool}}.\, \mathrm{true}$
  - $\mathrm{ismax}_{\mathrm{bool}} \triangleq \lambda x^{\mathrm{bool}}.\, x$
- Case $(\mathrm{bool} \to A) \cong A \times A$ — 2-digit numbers from A
  - $\mathrm{zero}_{\mathrm{bool}\to A} \triangleq \lambda x^{\mathrm{bool}}.\, \mathrm{zero}_A$
  - $\mathrm{succ}_{\mathrm{bool}\to A} \triangleq \lambda f^{\mathrm{bool}\to A}.$
    $\text{if } \mathrm{ismax}_A\ (f\ \mathrm{false})$ — Is the least significant digit the maximum?
    $\text{then } \lambda x^{\mathrm{bool}}.\, \text{if } x \text{ then } \mathrm{succ}_A\ (f\ \mathrm{true}) \text{ else } \mathrm{zero}_A$ — Carry-over calculation
    $\text{else } \lambda x^{\mathrm{bool}}.\, \text{if } x \text{ then } f\ \mathrm{true} \text{ else } \mathrm{succ}_A\ (f\ \mathrm{false})$ — Increment the least significant digit
  - $\mathrm{ismax}_{\mathrm{bool}\to A} \triangleq \lambda f^{\mathrm{bool}\to A}.\, \mathrm{and}\ \big(\mathrm{ismax}_A\ (f\ \mathrm{true})\big)\ \big(\mathrm{ismax}_A\ (f\ \mathrm{false})\big)$
- See the paper for a discussion of the other case

Main Contribution

- We present the first solver **HOMCSat** for **Higher-Order Quantified Boolean Formulas (HOQBF)** based on a new **linear-time reduction** to **Higher-Order Model Checking (HOMC)**



Implementation and Evaluation

- **HOMCSat** implemented using **HorSat2** as **HOMC**

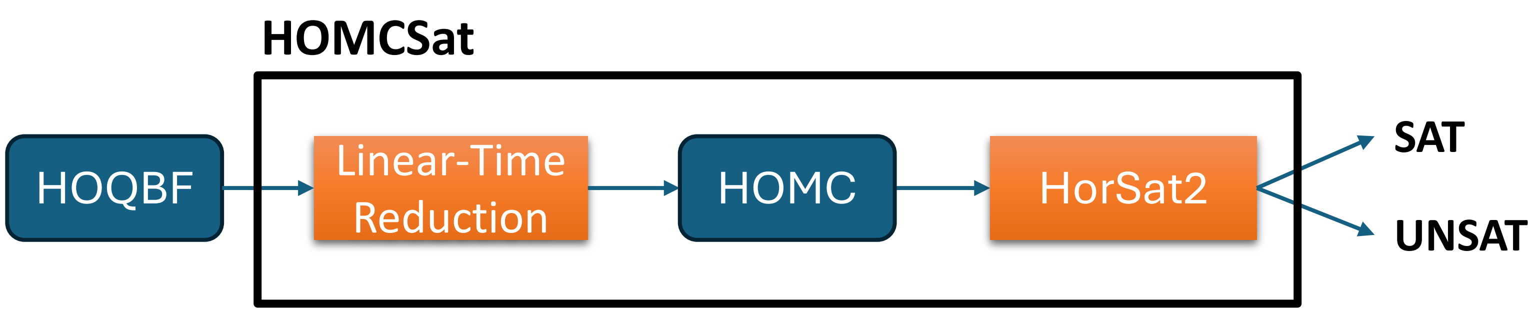
problem	O	V	A	result	time (s)
example1	1	3	2	SAT	0.032
example2	2	3	1	SAT	0.024
aaai23_ex1	2	5	4	SAT	0.437
aaai23_ex2	2	7	4	UNSAT	12.274
sym-asym-b	2	5	0	UNSAT	0.190
sym-asym-bb	2	9	0	T/O	N/A
sym-asym-id	2	7	1	SAT	0.070
SB-theorem	2	16	3	SAT	0.152
left-total	2	3	1	SAT	0.035
right-total	2	3	1	UNSAT	0.035
left-uniq	2	4	0	UNSAT	0.035
right-uniq	2	4	0	SAT	0.032
refl-cl-exist	2	11	2	SAT	3.415
refl-cl-uniq	2	23	2	SAT	124.717
sym-cl-exist	2	13	2	SAT	7.717
sym-cl-uniq	2	27	2	M/O	N/A
tran-cl-exist	2	15	2	SAT	30.286
tran-cl-uniq	2	31	2	M/O	N/A
f_h-neq-g_h	3	3	2	SAT	0.111
cps-arity1	3	5	4	SAT	11.785
cps-arity2	4	6	4	M/O	N/A

O: order
V: number of quantified variables
A: number of quantifier alternations

- Successfully solved **HOQBFs** that express typical properties of Boolean functions and binary relations
- Revealed limitations and possible future direction of **HOMC**
  - **HorSat2** struggled to scale when handling cases with **extensive branching** due to numerous quantifiers, a situation uncommon in human-written programs where **HorSat2** has been applied so far

Conclusion and Future Work

- Bridged two distinct fields: **HOMC** and **HOSAT**
- Opened the door to mutually exchanging techniques to advance both fields
  - Improving **HorSat2** by compactly representing reachable states using BDDs or ZDDs in symbolic model checkers
  - Incorporating abstraction, pruning, and propagation techniques used in **SAT**, **QBF**, and **DQBF** solvers
  - Implementing and applying Skolem function synthesis



Synthesizing Skolem Functions

- Some applications require **Skolem functions**, i.e., satisfying assignments to existential quantifiers
- The paper discusses extraction from a witness (namely, a typing derivation) returned by HOMC

Ex.1 $\forall x^{\mathbf{bool}}. \exists y^{\mathbf{bool}}. \forall z^{\mathbf{bool}}. (x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)$

A Skolem function for **Ex.1**: $Y(x) = \neg x$

Ex.2 $\forall f^{\mathbf{bool} \to \mathbf{bool}}. \exists g^{\mathbf{bool} \to \mathbf{bool}}. \exists z^{\mathbf{bool}}. \Big(f\big(g(z)\big)\Big) \leftrightarrow z$

Skolem functions for **Ex.2**: $G(f, x) = \mathbf{true},\ Z(f) = f(\mathbf{true})$