Moving Definition Variables in Quantified Boolean Formulas

Joseph E. Reeves, Marijn J. H. Heule, and Randal E. Bryant

Carnegie Mellon University

Appeared in TACAS'22

Overview

- Definition variables allow compact representation of Quantified Boolean Formulas (QBF)

Overview

- Definition variables allow compact representation of Quantified Boolean Formulas (QBF)
- Definition variables often placed in the innermost quantifier level, causing a loss of structure

Overview

- Definition variables allow compact representation of Quantified Boolean Formulas (QBF)
- Definition variables often placed in the innermost quantifier level, causing a loss of structure
- We propose moving definition variables closer to their respective defining variables

Overview

- Definition variables allow compact representation of Quantified Boolean Formulas (QBF)
- Definition variables often placed in the innermost quantifier level, causing a loss of structure
- We propose moving definition variables closer to their respective defining variables
- We show movement improves QBF solver performance, and is verifiable in the QRAT proof system

What are QBF?

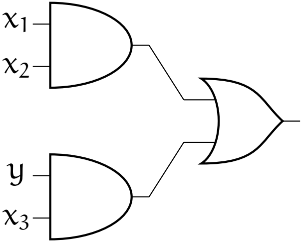
- **Quantified Boolean formulas (QBF)** are

  formulas of propositional logic + quantifiers

- *Examples*:
  - $(x \vee \overline{y}) \wedge (\overline{x} \vee y)$ (propositional logic $x \leftrightarrow y$)
  - $\exists x \forall y (x \vee \overline{y}) \wedge (\overline{x} \vee y)$
    Is there a value for x such that for all values of y the formula is true?
  - $\forall y \exists x (x \vee \overline{y}) \wedge (\overline{x} \vee y)$
    For all values of y, is there a value for x such that the formula is true?

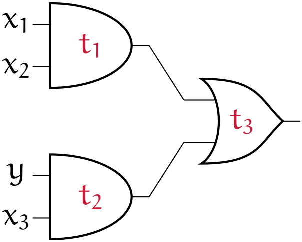
Circuit Problem

$\exists x_1 x_2 \forall y \exists x_3.$(Circuit output is 1)



Circuit Problem

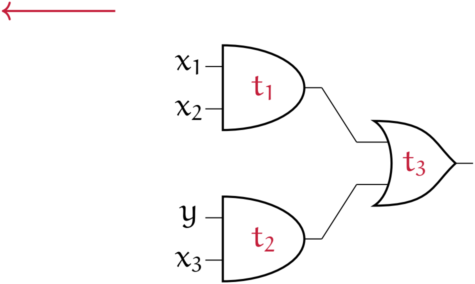
$\exists x_1 x_2 \forall y \exists x_3 \exists t_1 t_2 t_3$.(Circuit output is 1)



Add definition variables to capture output of each gate

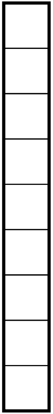
Circuit Problem

$\exists x_1 x_2 t_1 \forall y \exists x_3 \exists t_2 t_3.$(Circuit output is 1)



Example: For $t_1 \leftrightarrow x_1 \wedge x_2$, t_1 is defined by x_1, x_2 with clauses, $(t_1 \vee \overline{x}_1 \vee \overline{x}_2) \wedge (\overline{t}_1 \vee x_1) \wedge (\overline{t}_1 \vee x_2)$

Linear Domino Game



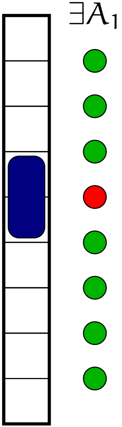
Board with $1 \times N$ squares

- Players alternate placing dominos
- First player who can't place domino loses

QBF encoding of two player games

- Existential player (A) and Universal player (B)
- If the formula is true, A has some winning move sequence for all B moves

Linear Domino Game



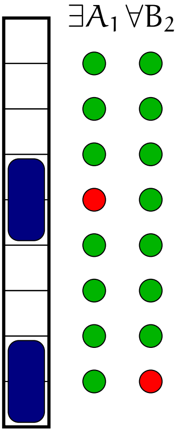
Board with $1 \times \mathrm{N}$ squares

- Players alternate placing dominos
- First player who can't place domino loses

QBF encoding of two player games

- Existential player (A) and Universal player (B)
- If the formula is true, A has some winning move sequence for all B moves

Linear Domino Game



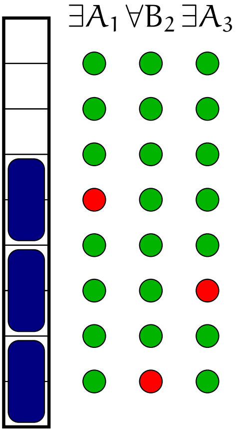
Board with $1 \times N$ squares

- Players alternate placing dominos
- First player who can't place domino loses

QBF encoding of two player games

- Existential player (A) and Universal player (B)
- If the formula is true, A has some winning move sequence for all B moves

Linear Domino Game



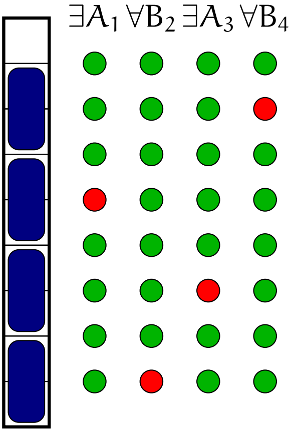
Board with $1 \times N$ squares

- Players alternate placing dominos
- First player who can't place domino loses

QBF encoding of two player games

- Existential player (A) and Universal player (B)
- If the formula is true, A has some winning move sequence for all B moves

Linear Domino Game



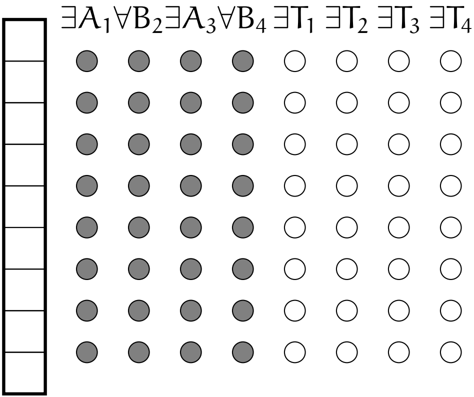
Board with $1 \times N$ squares

- Players alternate placing dominos
- First player who can't place domino loses

QBF encoding of two player games

- Existential player (A) and Universal player (B)
- If the formula is true, A has some winning move sequence for all B moves

Linear Domino Game Encoding



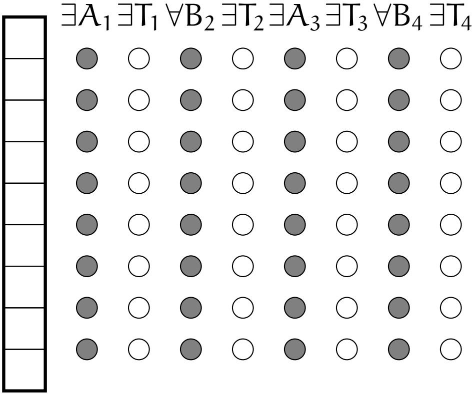
Problem Variables

- Attempt to place domino i on step t

Definition Variables

- Track state of board after each step
- Conventionally at innermost quantification level

Linear Domino Game Encoding



Problem Variables

- Attempt to place domino i on step t

Definition Variables

- Track state of board after each step
- Conventionally at innermost quantification level

Moving Definition Variables

- Right after their defining input variables
- Board state (T_i) updated after each move (A_i/B_i)

Definitions

- The Tseiten transformation adds many definitions
- These definitions are necessary for a compact representation in Prenex Conjunctive Normal Form
- Automated transformations often place definition variables in the innermost quantifier level

Motivating Example

- Consider the formula $\exists x_1 x_2 \forall y \exists t_1.(t_1 \leftrightarrow x_1 \wedge x_2) \wedge \ldots$

Motivating Example

- Consider the formula $\exists x_1 x_2 \forall y \exists t_1.(t_1 \leftrightarrow x_1 \land x_2) \land \ldots$
- Moving t_1 closer to its defining variables yields: $\exists x_1 x_2 t_1 \forall y.(t_1 \leftrightarrow x_1 \land x_2) \land \ldots$

Universal Reduction

- Universal Reduction removes a literal $l \in C$ if l is universally quantified and inner to all existential variables in C

Universal Reduction

- Universal Reduction removes a literal $l \in C$ if l is universally quantified and inner to all existential variables in C

- Given $\exists x_1 x_2 t_1 \forall y.(t_1 \leftrightarrow x_1 \land x_2) \land \ldots (y \lor t_1)$

Universal Reduction

- Universal Reduction removes a literal $l \in C$ if l is universally quantified and inner to all existential variables in C
- Given $\exists x_1 x_2 t_1 \forall y.(t_1 \leftrightarrow x_1 \wedge x_2) \wedge \ldots (y \vee t_1)$
- After moving t_1, y can be removed from the clause $(y \vee t_1)$ by universal reduction

Proofs in QBF

- QRAT is a clausal proof system, i.e., clauses with the QRAT property can be added or deleted from the formula

Proofs in QBF

- QRAT is a clausal proof system, i.e., clauses with the QRAT property can be added or deleted from the formula
- QRAT steps are equivalence preserving

Proofs in QBF

- QRAT is a clausal proof system, i.e., clauses with the QRAT property can be added or deleted from the formula
- QRAT steps are equivalence preserving
- A sequence of steps deriving the empty clause is a refutation proof if each clause addition is QRAT

Proofs in QBF

- QRAT is a clausal proof system, i.e., clauses with the QRAT property can be added or deleted from the formula
- QRAT steps are equivalence preserving
- A sequence of steps deriving the empty clause is a refutation proof if each clause addition is QRAT
- A sequence of steps deriving the empty formula is a satisfaction proof if each clause deletion is QRAT

Proofs in QBF

- QRAT is a clausal proof system, i.e., clauses with the QRAT property can be added or deleted from the formula
- QRAT steps are equivalence preserving
- A sequence of steps deriving the empty clause is a refutation proof if each clause addition is QRAT
- A sequence of steps deriving the empty formula is a satisfaction proof if each clause deletion is QRAT
- QRAT steps can be checked efficiently with the proof checker QRAT-TRIM

Performing Movement

Problem: Cannot change Q-Level of a variable in QRAT

Performing Movement

Problem: Cannot change Q-Level of a variable in QRAT

Solution: Introduce *fresh* variable x' at desired Q-Level

x' replaces x in formula through clause additions/deletions:

Performing Movement

Problem: Cannot change Q-Level of a variable in QRAT

Solution: Introduce *fresh* variable x' at desired Q-Level

x' replaces x in formula through clause additions/deletions:

1. *Add the defining clauses* $\delta(x')$ *and* $\delta(\overline{x}')$.
2. *Add the equivalence clauses* $x \leftrightarrow x'$.
3. *Add and remove the remaining clauses* $\rho(x)$ *and* $\rho(\overline{x})$.
4. *Remove the equivalence clauses* $x \leftrightarrow x'$.
5. *Remove the defining clauses* $\delta(x)$ *and* $\delta(\bar{x})$.

Moving Variables

- Given a set of definitions (definition type, definition variable, and defining variables)

Moving Variables

- Given a set of definitions (definition type, definition variable, and defining variables)
- A definition variable can be moved to its innermost defining variable

Moving Variables

- Given a set of definitions (definition type, definition variable, and defining variables)
- A definition variable can be moved to its innermost defining variable
- Iterate from outer Q-level inwards, moving all possible definition variables

CNF-based Definition Detection Tools

KISSAT

- Detects definitions independently
- Syntactic patterns: AND/OR, XOR, ITE, BiEQ
- Semantic using internal solver KITTEN

CNF-based Definition Detection Tools

Kissat

- Detects definitions independently
- Syntactic patterns: AND/OR, XOR, ITE, BiEQ
- Semantic using internal solver Kitten

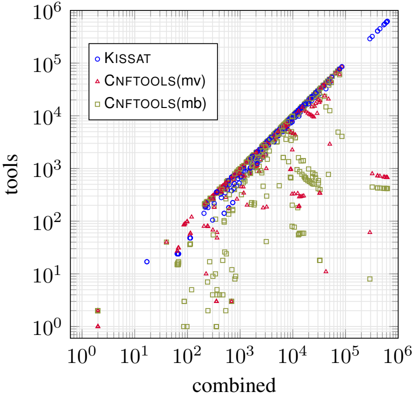
Cnftools

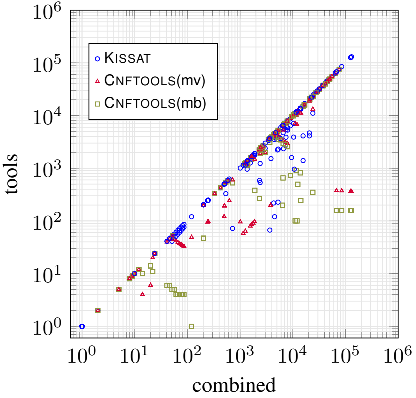
- Hierarchical detection, starting with root clauses (max-variable or min-blocked heuristic)
- Syntactic patterns: AND/OR, BiEQ, FULL
- Weaker semantic checking
- Monotonic checking

Definition Detection Tools Evaluation Setup

- 494 formulas from QBFEVAL'20
- 10 second timeout for each detection tool
- KISSAT iterates over all variables once or until timeout
- CNFTOOLS iterates over root variables until timeout

Found and Moved Definitions





Moved Definitions by Type

Detection Tool	Found	Moved	BiEQ	AND/OR	One-Sided	XOR
CNFTOOLS(mv)	3,525,559	1,032,807	21,198	969,630	37,642	0
CNFTOOLS(mb)	2,856,306	935,336	4,619	891,027	39,863	0
KISSAT	9,243,158	1,567,746	308,987	1,215,036	—	42,364
combined	9,624,654	1,664,655	309,793	1,273,381	37,646	42,476

- Movement in 157 formulas (found definitions in all 494)
- No semantic definitions moved by KISSAT
- CNFTOOLS slow, e.g., finds small portion of XORs, but finds monotonic definitions
- Combination important to maximize movement

Definition Movement Evaluation Setup

- 157 formulas with variable movement
- 5000 second timeout (with movement + Bloqqer time)
- Bloqqer run for 100 seconds
- Solvers Caqe and RareQS work well with preprocessors
- Solvers DepQBF and GhostQ discourage preprocessing

Original Vs. Movement

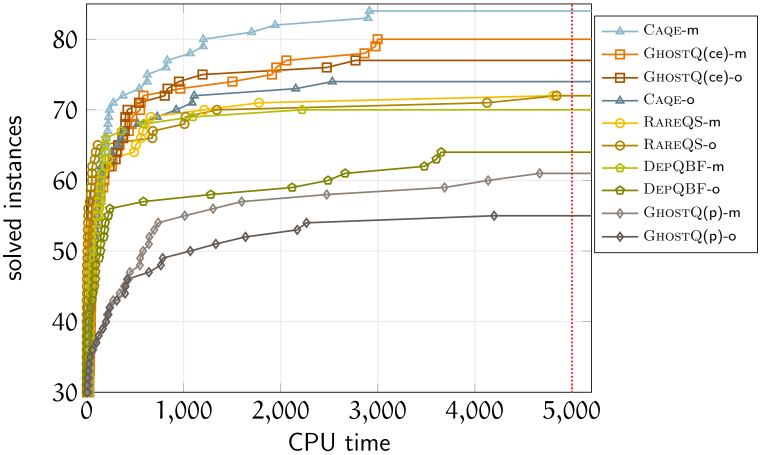


Figure: Cumulative number of solved instances

Bloqqer Vs. Movement then Bloqqer

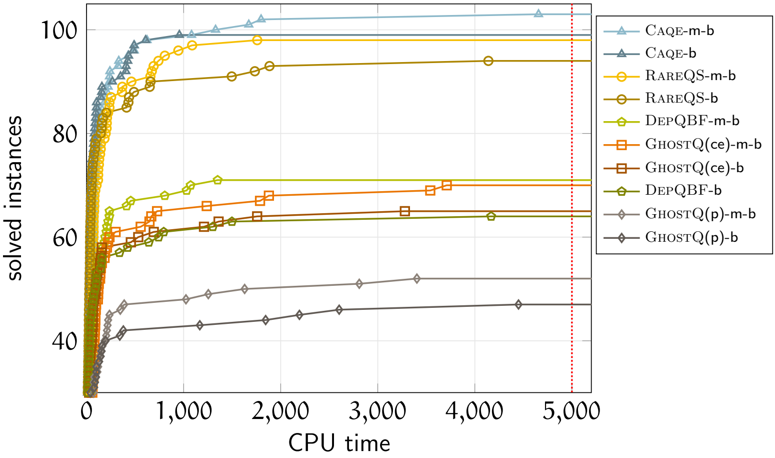


Figure: Cumulative number of solved instances after applying Bloqqer (-b) or movement then Bloqqer (m-b).

- Bloqqer solves an additional 3 formulas after movement

Instances Solved

Solver	Original	Moved	BLOQQER	Moved-BLOQQER
CAQE	74	84	99	**103**
GHOSTQ(p)	55	**61**	47	52
GHOSTQ(ce)	77	**80**	65	70
RAREQS	72	72	94	**98**
DEPQBF	64	70	64	**71**

- BLOQQER bad for for GHOSTQ and DEPQBF
- BLOQQER better than movement alone for others
- Options with movement give best results

PGBDDQ and Ldomino Benchmark

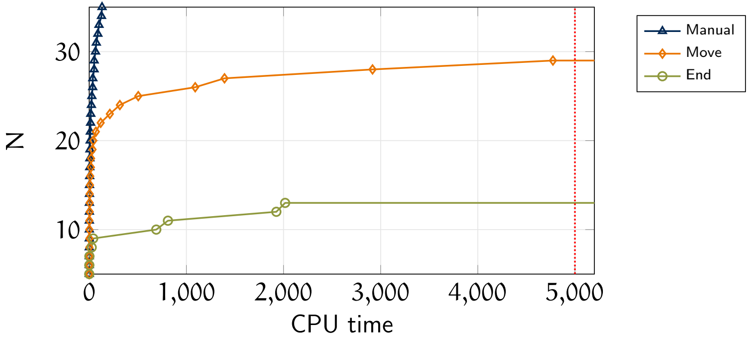


Figure: Boards of size N with definition variable placement: End - innermost quantifier level; Moved - variables moved; Manual

- End placement leads to memory outs
- Movement leads to time outs

Future Work

- More possible movement (semantic for KISSAT and monotonic for CNFTOOLS)
- Might not move all the way to defining variables
- Definition information may be useful to solvers or preprocessors directly

The authors are supported by the NSF under grant CCF-2108521.

A special thanks to the Star Exec community.

Questions?