QBFFAM: A TOOL FOR GENERATING QBF FAMILIES FROM PROOF COMPLEXITY

Olaf Beyersdorff, Luca Pulina, Martina Seidl, Ankit Shula

JOHANNES KEPLER
UNIVERSITY LINZ

Preliminaries

We consider QBFs $\Pi.\phi$ that are

Preliminaries

We consider QBFs $\Pi.\phi$ that are

- in prenex conjunctive normal form (PCNF)
- closed

Example

$\exists x \forall a \exists yz.\phi$ with $\phi = ((x \vee a \vee y) \wedge (\neg x \vee a \vee \neg y) \wedge (\neg x \vee \neg a \vee y \vee z) \wedge (\neg x \vee a))$

Preliminaries

We consider QBFs $\Pi.\phi$ that are

- in prenex conjunctive normal form (PCNF)
- closed

Example

$\exists x \forall a \exists y z.\phi$ with $\phi = ((x \vee a \vee y) \wedge (\neg x \vee a \vee \neg y) \wedge (\neg x \vee \neg a \vee y \vee z) \wedge (\neg x \vee a))$

Every QBF can be translated to an equivalent formula in PCNF.

QBF Semantics

- $\forall x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **or** $\Pi.\psi[\neg x]$ false

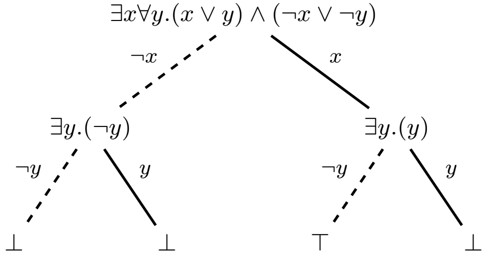
JKU

QBF Semantics

- $\forall x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **or** $\Pi.\psi[\neg x]$ false
- $\exists x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **and** $\Pi.\psi[\neg x]$ false

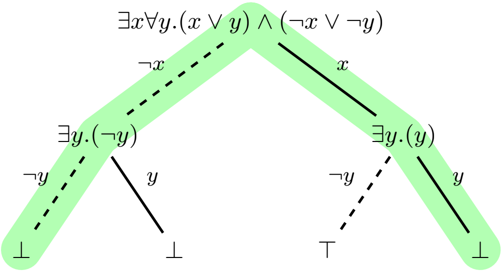
QBF Semantics

- $\forall x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **or** $\Pi.\psi[\neg x]$ false
- $\exists x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **and** $\Pi.\psi[\neg x]$ false
- Example:



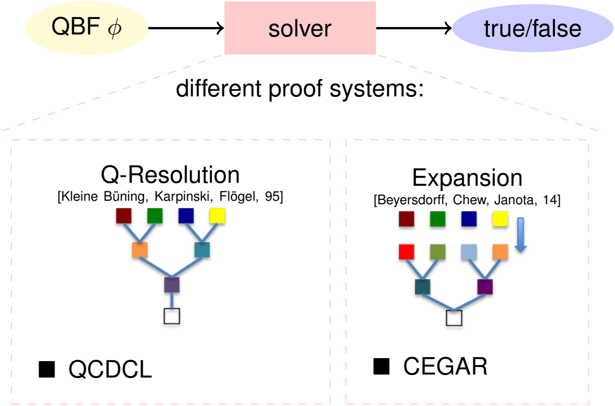
QBF Semantics

- $\forall x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **or** $\Pi.\psi[\neg x]$ false
- $\exists x \Pi.\psi$ false $\Leftrightarrow \Pi.\psi[x]$ **and** $\Pi.\psi[\neg x]$ false
- Example:



Summary: Solving Approaches for QBF

Summary: Solving Approaches for QBF



Resolution for QBF

Let $\Pi.\psi$ be a QBF. The Q-Res calculus consists of the following rules:

Resolution Rule (R):

$C_1 \vee x, \;\; C_2 \vee \bar{x} \;\; \xrightarrow{\text{R}} \;\; C_1 \vee C_2$

$C_1 \vee x, C_2 \vee \bar{x}$ already derived
$C_1 \vee C_2$ is no tautology
x is existential

Universal Reduction (U):

$D \vee l \;\; \xrightarrow{\text{U}} \;\; D$

$D \vee l$ already derived
l is universal
for all existential literals $k \in D\colon\ k < l$

JKU

Resolution for QBF

Let $\Pi.\psi$ be a QBF. The Q-Res calculus consists of the following rules:

Resolution Rule (R):

$C_1 \vee x, \; C_2 \vee \bar{x} \;\xrightarrow{\text{R}}\; C_1 \vee C_2$

$C_1 \vee x, C_2 \vee \bar{x}$ already derived
$C_1 \vee C_2$ is no tautology
x is existential

Universal Reduction (U):

$D \vee l \;\xrightarrow{\text{U}}\; D$

$D \vee l$ already derived
l is universal
for all existential literals $k \in D\colon k < l$

Example: $\exists x \forall y.((x \vee y) \wedge (\bar{x} \vee \bar{y}))$

$$x \vee y \;\xrightarrow{\text{U}}\; x$$
$$\bar{x} \vee \bar{y} \;\xrightarrow{\text{U}}\; \bar{x}$$
$$x, \; \bar{x} \;\xrightarrow{\text{R}}\; \bot$$

JKU

Resolution for QBF

Let $\Pi.\psi$ be a QBF. The Q-Res calculus consists of the following rules:

Resolution Rule (R):

$$C_1 \vee x,\ \ C_2 \vee \bar{x} \ \xrightarrow{\text{R}}\ C_1 \vee C_2$$

$C_1 \vee x, C_2 \vee \bar{x}$ already derived
$C_1 \vee C_2$ is no tautology
x is existential

Universal Reduction (U):

$$D \vee l \ \xrightarrow{\text{U}}\ D$$

$D \vee l$ already derived
l is universal
for all existential literals $k \in D\colon k < l$

Example: $\exists x \forall y.((x \vee y) \wedge (\bar{x} \vee \bar{y}))$

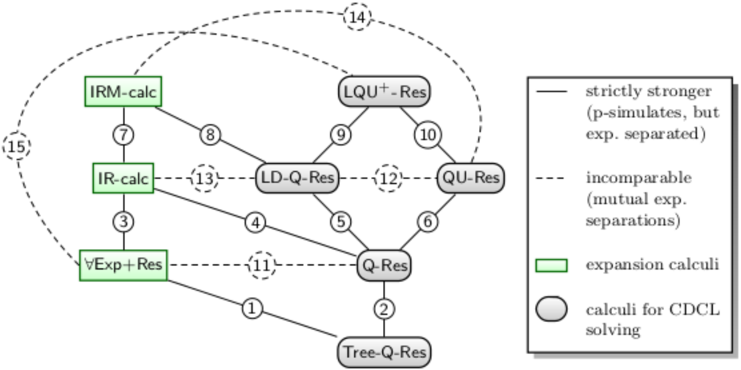
$$x \vee y \ \xrightarrow{\text{U}}\ x$$
$$\bar{x} \vee \bar{y} \ \xrightarrow{\text{U}}\ \bar{x}$$
$$x,\ \ \bar{x} \ \xrightarrow{\text{R}}\ \bot$$

Extensions:

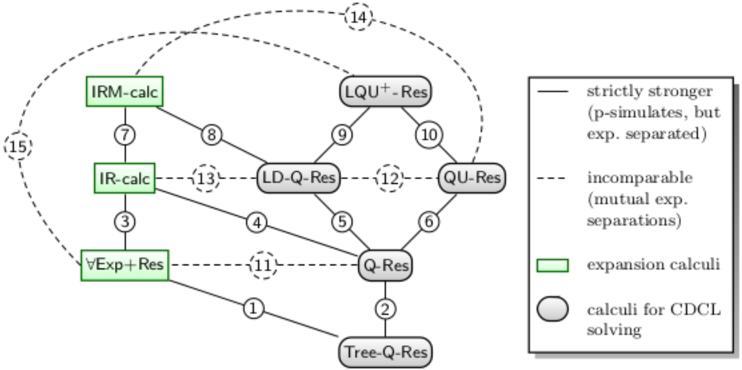
- universal resolution
- long-distance resolution
- symmetries

JKU

Proof Complexity Landscape



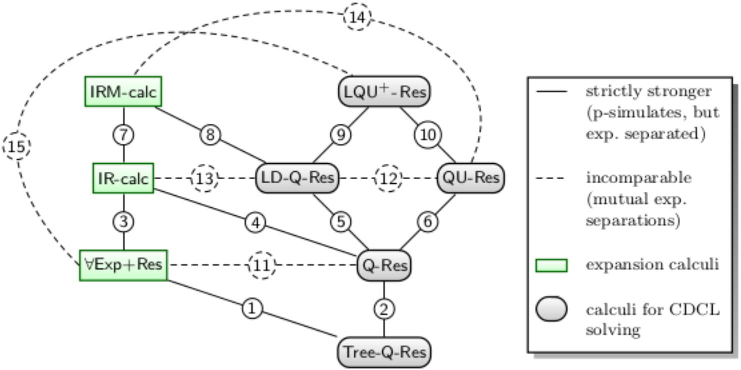
Proof Complexity Landscape



conclusion: different proof systems have different strength

JKU

Proof Complexity Landscape



conclusion: different proof systems have different strength

⇒ showing separations relies on certain formula families

Example: KBFK-Formulas

For $n \in \mathbb{N}$, the formula $\texttt{KBKF}_n$ is defined by the prefix

$$\exists x_1 y_1 \forall a_1 \exists x_2 y_2 \forall a_2 \ldots \exists x_n y_n \forall a_n \exists z_1 \ldots z_n$$

and the following clauses:

- $C_1 = (\bar{x}_1 \vee \bar{y}_1)$
- for $j = 1, \ldots, n-1$:
  $C_{2j} = (x_j \vee \bar{a}_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$
  $C_{2j+1} = (y_j \vee a_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$.
- $C_{2n} = (x_n \vee \bar{a}_n \vee \bar{z}_1 \vee \ldots \vee \bar{z}_n)$,
  $C_{2n+1} = (y_n \vee a_n \vee \bar{z}_1 \vee \ldots \vee \bar{z}_n)$
- for $j = 1, \ldots, n$:
  $B_{2j-1} = (a_j \vee z_j)$ and $B_{2j} = (\bar{a}_j \vee z_j)$.

Example: KBFK-Formulas

For $n \in \mathbb{N}$, the formula $\texttt{KBKF}_n$ is defined by the prefix

$$\exists x_1 y_1 \forall a_1 \exists x_2 y_2 \forall a_2 \dots \exists x_n y_n \forall a_n \exists z_1 \dots z_n$$

and the following clauses:

- $C_1 = (\bar{x}_1 \vee \bar{y}_1)$
- for $j = 1, \dots, n-1$:
  $C_{2j} = (x_j \vee \bar{a}_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$
  $C_{2j+1} = (y_j \vee a_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$.
- $C_{2n} = (x_n \vee \bar{a}_n \vee \bar{z}_1 \vee \dots \vee \bar{z}_n)$,
  $C_{2n+1} = (y_n \vee a_n \vee \bar{z}_1 \vee \dots \vee \bar{z}_n)$
- for $j = 1, \dots, n$:
  $B_{2j-1} = (a_j \vee z_j)$ and $B_{2j} = (\bar{a}_j \vee z_j)$.

no short Q-resolution proofs

Example: KBFK-Formulas

For $n \in \mathbb{N}$, the formula $\texttt{KBKF}_n$ is defined by the prefix

$$\exists x_1 y_1 \forall a_1 \exists x_2 y_2 \forall a_2 \dots \exists x_n y_n \forall a_n \exists z_1 \dots z_n$$

and the following clauses:

- $C_1 = (\bar{x}_1 \lor \bar{y}_1)$
- for $j = 1, \dots, n-1$:
  $C_{2j} = (x_j \lor \bar{a}_j \lor \bar{x}_{j+1} \lor \bar{y}_{j+1})$
  $C_{2j+1} = (y_j \lor a_j \lor \bar{x}_{j+1} \lor \bar{y}_{j+1})$.
- $C_{2n} = (x_n \lor \bar{a}_n \lor \bar{z}_1 \lor \dots \lor \bar{z}_n)$,
  $C_{2n+1} = (y_n \lor a_n \lor \bar{z}_1 \lor \dots \lor \bar{z}_n)$
- for $j = 1, \dots, n$:
  $B_{2j-1} = (a_j \lor z_j)$ and $B_{2j} = (\bar{a}_j \lor z_j)$.

Symmetry:
$\sigma_i = (x_i \; y_i) \; (\bar{x}_i \; \bar{y}_i) \; (a_i \; \bar{a}_i)$

JKU

Example: KBFK-Formulas

For $n \in \mathbb{N}$, the formula $\texttt{KBKF}_n$ is defined by the prefix

$$\exists x_1 y_1 \forall a_1 \exists x_2 y_2 \forall a_2 \ldots \exists x_n y_n \forall a_n \exists z_1 \ldots z_n$$

and the following clauses:

- $C_1 = (\bar{x}_1 \vee \bar{y}_1)$
- for $j = 1, \ldots, n-1$:
  $C_{2j} = (x_j \vee \bar{a}_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$
  $C_{2j+1} = (y_j \vee a_j \vee \bar{x}_{j+1} \vee \bar{y}_{j+1})$.
- $C_{2n} = (x_n \vee \bar{a}_n \vee \bar{z}_1 \vee \ldots \vee \bar{z}_n)$,
  $C_{2n+1} = (y_n \vee a_n \vee \bar{z}_1 \vee \ldots \vee \bar{z}_n)$
- for $j = 1, \ldots, n$:
  $B_{2j-1} = (a_j \vee z_j)$ and $B_{2j} = (\bar{a}_j \vee z_j)$.

Symmetry:
$\sigma_i = (x_i\ y_i)\ (\bar{x}_i\ \bar{y}_i)\ (a_i\ \bar{a}_i)$

short Q-resolution proofs with symmetries

JKU

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

- support of 12 formula families:

KBKF	KBKF_LD	KBKF_QU
Parity	LQParity	QUParity
EQ	EQ-Sq	BEQ
LONSING	TRAPDOOR	CR

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

- support of 12 formula families:

`KBKF`	`KBKF_LD`	`KBKF_QU`
`Parity`	`LQParity`	`QUParity`
`EQ`	`EQ-Sq`	`BEQ`
`LONSING`	`TRAPDOOR`	`CR`

- all generated formulas are false

JKU

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

- support of 12 formula families:

`KBKF`	`KBKF_LD`	`KBKF_QU`
`Parity`	`LQParity`	`QUParity`
`EQ`	`EQ-Sq`	`BEQ`
`LONSING`	`TRAPDOOR`	`CR`

- all generated formulas are false
- QDIMACS format

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

- support of 12 formula families:

`KBKF`	`KBKF_LD`	`KBKF_QU`
`Parity`	`LQParity`	`QUParity`
`EQ`	`EQ-Sq`	`BEQ`
`LONSING`	`TRAPDOOR`	`CR`

- all generated formulas are false
- QDIMACS format
- available at

  `https://github.com/marseidl/qbffam.git`

QBFFam: A Formula Generator

tool for generating formulas used in proof complexity

- support of 12 formula families:

`KBKF`	`KBKF_LD`	`KBKF_QU`
`Parity`	`LQParity`	`QUParity`
`EQ`	`EQ-Sq`	`BEQ`
`LONSING`	`TRAPDOOR`	`CR`

- all generated formulas are false
- QDIMACS format
- available at

  `https://github.com/marseidl/qbffam.git`

- application: testing and comparing solvers

Formula Families Supported By QBFFam

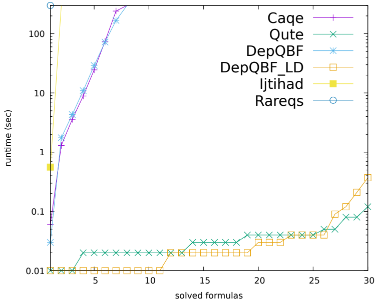
formula family	#alt	#vars	#cl	QRes	QRes-LD	QRes-QU	QRes-LQU$^+$	∀Exp-Res	IR-calc	IRM-calc	QRes-SYM
KBKF	$n+1$	$4n$	$4n+1$	✗	✓	✓	✓	✗	✗	✓	✓
KBKF_LD	$n+1$	$4n$	$4n+1$	✗	✗	✓	✓	✗	✗	✗	✓
KBKF_QU	$n+1$	$5n$	$4n+1$	✗	✓	✗	✓	✗	✗	✓	✓
Parity	2	2n	$4n-2$	✗	✓	✗	✓	✓	✓	✓	✓
LQParity	2	2n	$8n-6$	✗	✗	✗	✓	✓	✓	✓	✓
QUParity	2	2n+1	$8n-6$	✗	✗	✗	✗	✓	✓	✓	✓
EQ	3	$3n$	$2n+1$	✗	✓	✗	✓	✗	✗	✓	✓
EQ-Sq	3	n^2+4n	$5n^2$	✗	✓	✗	✓	✗	✗	✓	✓
BEQ	4	$6n+2$	$5n+2$	✗	✓	✗	✓	✗	✗	✓	✗
CP	2	n^2	$2n$	✓	✓	✓	✓	✓	✓	✓	✓
TRAPDOOR	3	$O(n^2)$	$O(n^2)$	✓	✓	✓	✓	✓	✓	✓	✓
LONSING	2	$O(n^2)$	$O(n^2)$	✓	✓	✓	✓	✓	✓	✓	✓

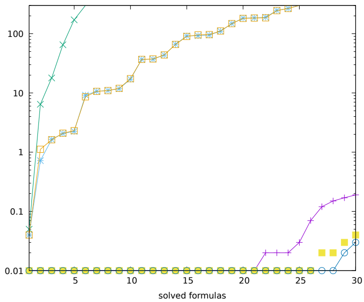
✓ ... short proofs (poly size) ✗ ... no short proofs (exponential lower bounds)

#alt ... number of quantifier alternations

#vars ... number of variables #cl ... number of clauses

Some Experiments





Summary

- QBFFam: generator for prominent formula families from proof complexity
- scalable test cases with known result
- characterization of solving behavior
- `https://github.com/marseidl/qbffam.git`

Future Work

- more evaluations
- true formulas
- formulas based on graphs

Thank you very much!