Quantified Boolean Formulas: (Solving and) Proof Complexity

Meena Mahajan

The Institute of Mathematical Sciences (HBNI), Chennai, India.

25 March 2023

Propositional Satisfiability

- SAT: Satisfiability.
  eg. Is there an assignment to x, y, z satisfying all the clauses $(x \vee y \vee z), (x \vee \neg\, y \vee \neg\, z), (\neg\, x \vee y \vee \neg\, z), (\neg\, x \vee \neg\, y \vee z)$?
- Quintessential NP-complete problem.
- Very hard – in theory.

  In practice – a solved problem! Many good SAT solvers around.

Propositional Satisfiability

- SAT: Satisfiability.
  eg. Is there an assignment to x, y, z satisfying all the clauses $(x \vee y \vee z), (x \vee \neg\, y \vee \neg\, z), (\neg\, x \vee y \vee \neg\, z), (\neg\, x \vee \neg\, y \vee z)$?
- Quintessential NP-complete problem.
- Very hard – in theory.

  In practice – a solved problem! Many good SAT solvers around.
- Ambitious ongoing programs to design good solvers for problems harder than SAT.
- Focus of this talk: QBF.

QBF: Quantified Boolean Formulas

- We consider QBFs that are
  - totally quantified (no unbound variables),
  - in prenex form,
  - with inner propositional formula in CNF.
- e.g. Is this formula true?

$$\underbrace{\exists e\ \forall u\ \exists c\ \exists d}_{\text{quantifier prefix}}\quad \underbrace{(\neg e \vee c)(e \vee d)(\neg u \vee c)(u \vee d)(\neg c \vee \neg d)}_{\text{matrix}}$$

QBF: Quantified Boolean Formulas

- We consider QBFs that are
  - totally quantified (no unbound variables),
  - in prenex form,
  - with inner propositional formula in CNF.
- e.g. Is this formula true?

$$\underbrace{\exists e\ \forall u\ \exists c\ \exists d}_{\text{quantifier prefix}}\quad \underbrace{(\neg e \vee c)(e \vee d)(\neg u \vee c)(u \vee d)(\neg c \vee \neg d)}_{\text{matrix}}$$

- QBF subsumes SAT. eg. Is this QBF true?

$$\exists x \exists y \exists z (x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z)$$

QBF: Quantified Boolean Formulas

- We consider QBFs that are
  - totally quantified (no unbound variables),
  - in prenex form,
  - with inner propositional formula in CNF.
- e.g. Is this formula true?

$$\underbrace{\exists e\ \forall u\ \exists c\ \exists d}_{\text{quantifier prefix}}\quad \underbrace{(\neg e \vee c)(e \vee d)(\neg u \vee c)(u \vee d)(\neg c \vee \neg d)}_{\text{matrix}}$$

- QBF subsumes SAT. eg. Is this QBF true?

$$\exists x \exists y \exists z (x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z)$$

- QBF more succinctly expressive than SAT; PSPACE-complete.

QBF Proof Complexity

- Quite a few QBF solvers developed in the last couple of decades.
- Underlying solver heuristics are formal proof systems:
  Runs of QBF solver on false QBFs provide proofs of falsity.

QBF Proof Complexity

- Quite a few QBF solvers developed in the last couple of decades.
- Underlying solver heuristics are formal proof systems:
  Runs of QBF solver on false QBFs provide proofs of falsity.
- Formalising the right proof system to capture solver reasoning — highly non-trivial.
- Lower bounds in formal proof system
  (no short proof of falsity)
  $\Downarrow$
  no short runs of QBF solver.

QBF Proof Complexity

- Quite a few QBF solvers developed in the last couple of decades.
- Underlying solver heuristics are formal proof systems:
  Runs of QBF solver on false QBFs provide proofs of falsity.
- Formalising the right proof system to capture solver reasoning — highly non-trivial.
- Lower bounds in formal proof system
  (no short proof of falsity)
  $\Downarrow$
  no short runs of QBF solver.
- Proof Complexity –
  Formalising sound-and-complete proof systems underlying solvers;
  Demonstrating limitations by proving lower bounds.

QBF semantics: The two-player evaluation game

- QBF $Q\vec{x} \cdot F(x)$
- Two players, $P_\exists$ and $P_\forall$, step through quantifier prefix left-to-right. $P_\exists$ picks values for $\exists$ variables, $P_\forall$ for $\forall$ variables.

  Assignment constructed on a run: $\tilde{a}$.

  $P_\exists$ wins a run of the game if $F(\tilde{a})$ true. Otherwise $P_\forall$ wins.
- $Q\vec{x} \cdot F(x)$ true if and only if $P_\exists$ has a winning strategy. (model, Skolem function, ...)
- $Q\vec{x} \cdot F(x)$ false if and only if $P_\forall$ has a winning strategy. (countermodel, Herbrand function, ...)

QBF semantics: The two-player evaluation game

- QBF $Q\vec{x} \cdot F(x)$
- Two players, $P_\exists$ and $P_\forall$, step through quantifier prefix left-to-right. $P_\exists$ picks values for $\exists$ variables, $P_\forall$ for $\forall$ variables.

  Assignment constructed on a run: $\tilde{a}$.

  $P_\exists$ wins a run of the game if $F(\tilde{a})$ true. Otherwise $P_\forall$ wins.
- $Q\vec{x} \cdot F(x)$ true if and only if $P_\exists$ has a winning strategy. (model, Skolem function, ...)
- $Q\vec{x} \cdot F(x)$ false if and only if $P_\forall$ has a winning strategy. (countermodel, Herbrand function, ...)

Focus in this talk on refutations: proving false QBF s false.

How to prove that a false QBF is false

- Start with initial set of clauses.
- Derive and add clauses to set until falseness is obvious.

How to prove that a false QBF is false

- Start with initial set of clauses.
- Derive and add clauses to set until falseness is obvious.

- To achieve soundness:
  - Addition rules must preserve (some/every) $P_\exists$ winning strategy.
  - Finally derive empty clause $\square$.
    (This defeats every potential $P_\exists$ strategy.)
- To achieve completeness:
  - From a $P_\forall$ winning strategy, use rules to derive $\square$.

Early SAT solvers

- Early SAT solvers: classical backtracking.

  The DPLL algorithm (Davis-Putnam, Davis-Logemann-Loveland).
  - Decide / guess the value of some variables.
  - Propagate values in unit clauses.
  - Backtrack on falsified clauses.

Early SAT solvers

- Early SAT solvers: classical backtracking.

  The DPLL algorithm (Davis-Putnam, Davis-Logemann-Loveland).

  - Decide / guess the value of some variables.
  - Propagate values in unit clauses.
  - Backtrack on falsified clauses.

- Correctness:
  DPLL run reporting UNSAT yields refutation in Resolution.

$$\frac{x \vee A \qquad \overline{x} \vee B}{A \vee B}$$

Lifting a SAT solver to QBF

Early thinking: DPLL looks for satisfying assignments.
What would QDPLL do? Look for a model, a $P_\exists$ winning strategy.

Lifting a SAT solver to QBF

Early thinking: DPLL looks for satisfying assignments.
What would QDPLL do? Look for a model, a $P_\exists$ winning strategy.

- Do not propagate universal variables – these values are not controlled by $P_\exists$ and so cannot be forced.

Lifting a SAT solver to QBF

Early thinking: DPLL looks for satisfying assignments.
What would QDPLL do? Look for a model, a $P_\exists$ winning strategy.

- Do not propagate universal variables – these values are not controlled by $P_\exists$ and so cannot be forced.
- Make decisions in quantifier-block order (level-order).

Lifting a SAT solver to QBF

Early thinking: DPLL looks for satisfying assignments.
What would QDPLL do? Look for a model, a $P_\exists$ winning strategy.

- Do not propagate universal variables – these values are not controlled by $P_\exists$ and so cannot be forced.
- Make decisions in quantifier-block order (level-order).
- While learning, perform reductions wherever possible – remove universal literals from clauses if removal sound.

Treatment of Universal Variables

- Must be treated differently; $\exists x \forall y [x = y]$ false but $\exists x \exists y [x = y]$ true.
- Allow adversarial instantiation. When is this sound?
- Allow usage as resolution pivot – is this sound?

The Reduction rule

- When adversarial instantiation is sound, and how to implement it:
  - In the two-player game, if a partial run results in a purely universal clause, $P_\forall$ can/will falsify it.
  - If a derived clause has a "trailing" (rightmost) universal literal, $P_\exists$ must be prepared for $P_\forall$ to adversarially falsify it.

The Reduction rule

- When adversarial instantiation is sound, and how to implement it:
  - In the two-player game, if a partial run results in a purely universal clause, $P_\forall$ can/will falsify it.
  - If a derived clause has a "trailing" (rightmost) universal literal, $P_\exists$ must be prepared for $P_\forall$ to adversarially falsify it.
- Implementing adversarial instantiation super-soundly:
  Universal reduction.

  For clauses:
  $\frac{A \vee u}{A}$ ($\mathrm{var}(u)$ is universal, and right of all variables in A)

The Reduction rule

- When adversarial instantiation is sound, and how to implement it:
  - In the two-player game, if a partial run results in a purely universal clause, $P_\forall$ can/will falsify it.
  - If a derived clause has a "trailing" (rightmost) universal literal, $P_\exists$ must be prepared for $P_\forall$ to adversarially falsify it.
- Implementing adversarial instantiation super-soundly:
  Universal reduction.

  For clauses:
  $\frac{A \vee u}{A}$ ($\mathrm{var}(u)$ is universal, and right of all variables in A)

  For other types of lines:
  $\frac{\ell(x_1, x_2, \ldots x_m, u)}{\ell(x_1, x_2, \ldots x_m, b)}$ ($\mathrm{var}(u)$ is universal, right of all x_i; $b \in \{\top, \bot\}$)

Reduction-based refutational proof systems

- A sound and complete proof system: Q-Res.
  [KleineBüning,Karpinski,Flögel 1995] Two rules:
  * Resolution: (super-sound) $\dfrac{x \vee A \qquad \overline{x} \vee B}{A \vee B}$ ($\mathrm{var}(x)$ is existential)
  * Universal reduction: (super-sound)
    $\dfrac{A \vee u}{A}$ ($\mathrm{var}(u)$ is universal, and right of all variables in A)

Reduction-based refutational proof systems

- A sound and complete proof system: Q-Res.
  [KleineBüning,Karpinski,Flögel 1995] Two rules:
  * Resolution: (super-sound) $\dfrac{x \vee A \qquad \overline{x} \vee B}{A \vee B}$ ($\mathrm{var}(x)$ is existential)
  * Universal reduction: (super-sound)
    $\dfrac{A \vee u}{A}$ ($\mathrm{var}(u)$ is universal, and right of all variables in A)
- The QURes proof system (a.k.a. Res+$\forall$Red):
  Resolution + Universal Reduction. [vanGelder 2012]

Reduction-based refutational proof systems

- A sound and complete proof system: Q-Res.
  [KleineBüning,Karpinski,Flögel 1995] Two rules:
  * Resolution: (super-sound) $\dfrac{x \vee A \qquad \overline{x} \vee B}{A \vee B}$ ($\mathrm{var}(x)$ is existential)
  * Universal reduction: (super-sound)
    $\dfrac{A \vee u}{A}$ ($\mathrm{var}(u)$ is universal, and right of all variables in A)
- The QURes proof system (a.k.a. Res+$\forall$Red):
  Resolution + Universal Reduction. [vanGelder 2012]
- For (essentially) any line-based propositional proof system P,
  QBF proof system $P + \forall$Red: [Beyersdorff,Bonacina,Chew,Pich 2016,20]
  - CP+$\forall$Red [Beyersdorff,Chew,M,Shukla 2016]
  - $\mathcal{C}$-Frege + $\forall$Red
  - EFrege + $\forall$Red
  - OBDD($\wedge$, weaken) + $\forall$Red [Mengel,Slivovsky 2021]
  - $\vdots$

A resolution+reduction (QRes) refutation

$\exists e_1 \; \forall u_1 \; \exists c_1 \; \exists d_1 \quad \exists e_2 \; \forall u_2 \; \exists c_2 \; \exists d_2$

$(\bar{e}_1 \; c_1)$ $(e_1 \; d_1)$ $(\bar{e}_2 \; c_2)$ $(e_2 \; d_2)$ $(\bar{c}_1 \; \bar{d}_1 \; \bar{c}_2 \; \bar{d}_2)$

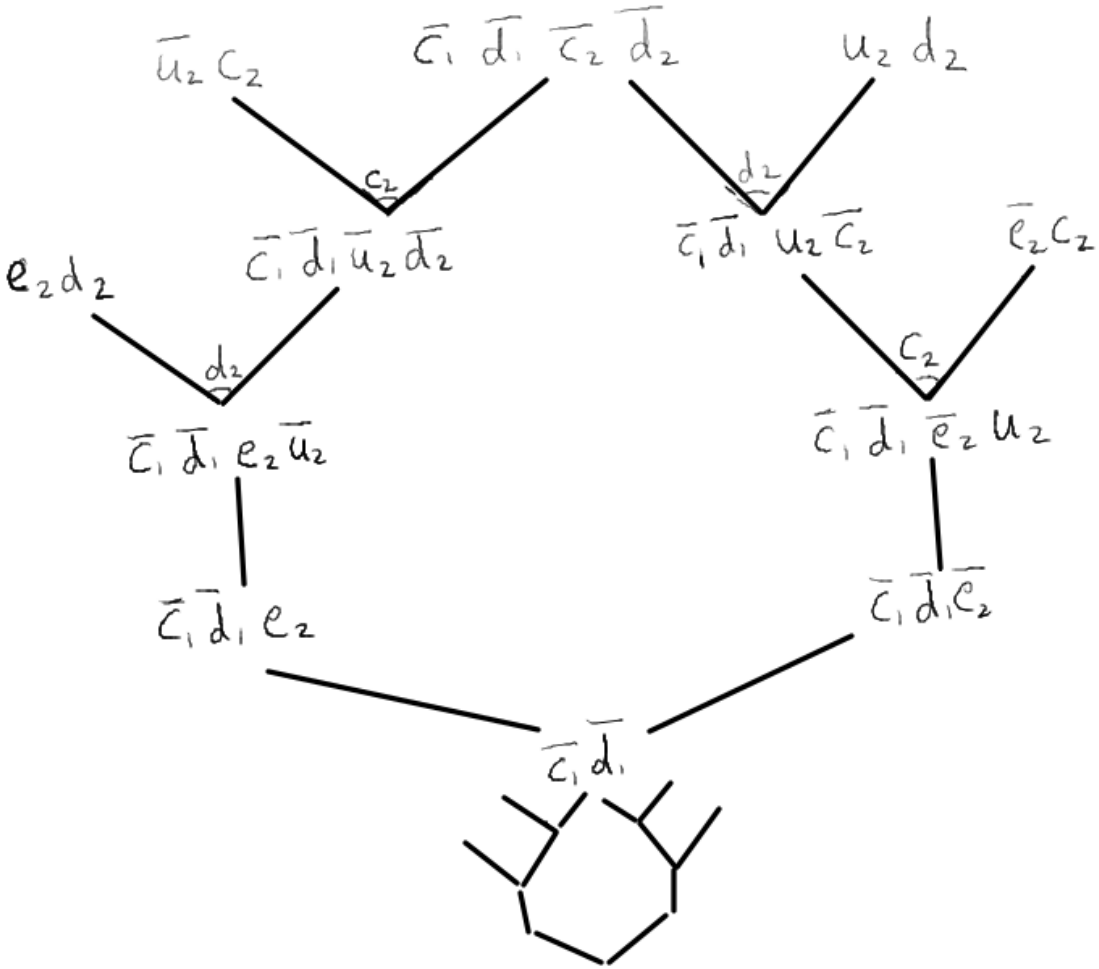
$(\bar{u}_1 \; c_1)$ $(u_1 \; d_1)$ $(\bar{u}_2 \; c_2)$ $(u_2 \; d_2)$

A resolution+reduction (QRes) refutation

$\exists e_1 \forall u_1 \exists c_1 \exists d_1 \quad \exists e_2 \forall u_2 \exists c_2 \exists d_2$

$(\bar{e}_1\ c_1)$ $(e_1\ d_1)$ $(\bar{e}_2\ c_2)$ $(e_2\ d_2)$ $(\bar{c}_1\ \bar{d}_1\ \bar{c}_2\ \bar{d}_2)$

$(\bar{u}_1\ c_1)$ $(u_1\ d_1)$ $(\bar{u}_2\ c_2)$ $(u_2\ d_2)$



A resolution+reduction (QURes) refutation

$\exists d_1 \exists e_1 \forall x_1$

$\exists d_2 \exists e_2 \forall x_2$

$\exists f_1 \exists f_2$

$(\bar{d}_1 \bar{e}_1)$

$(d_1 x_1 \bar{d}_2 \bar{e}_2)\ (e_1 \bar{x}_1 \bar{d}_2 \bar{e}_2)$

$(d_2 x_2 \bar{f}_1 \bar{f}_2)(e_2 \bar{x}_2 \bar{f}_1 \bar{f}_2)$

$(x_1 f_1)\ \ (\bar{x}_1 f_1)$

$(x_2 f_2)\ \ (\bar{x}_2 f_2)$

A resolution+reduction (QURes) refutation

$\exists d_1 \exists e_1 \forall x_1$

$\exists d_2 \exists e_2 \forall x_2$

$\exists f_1 \exists f_2$

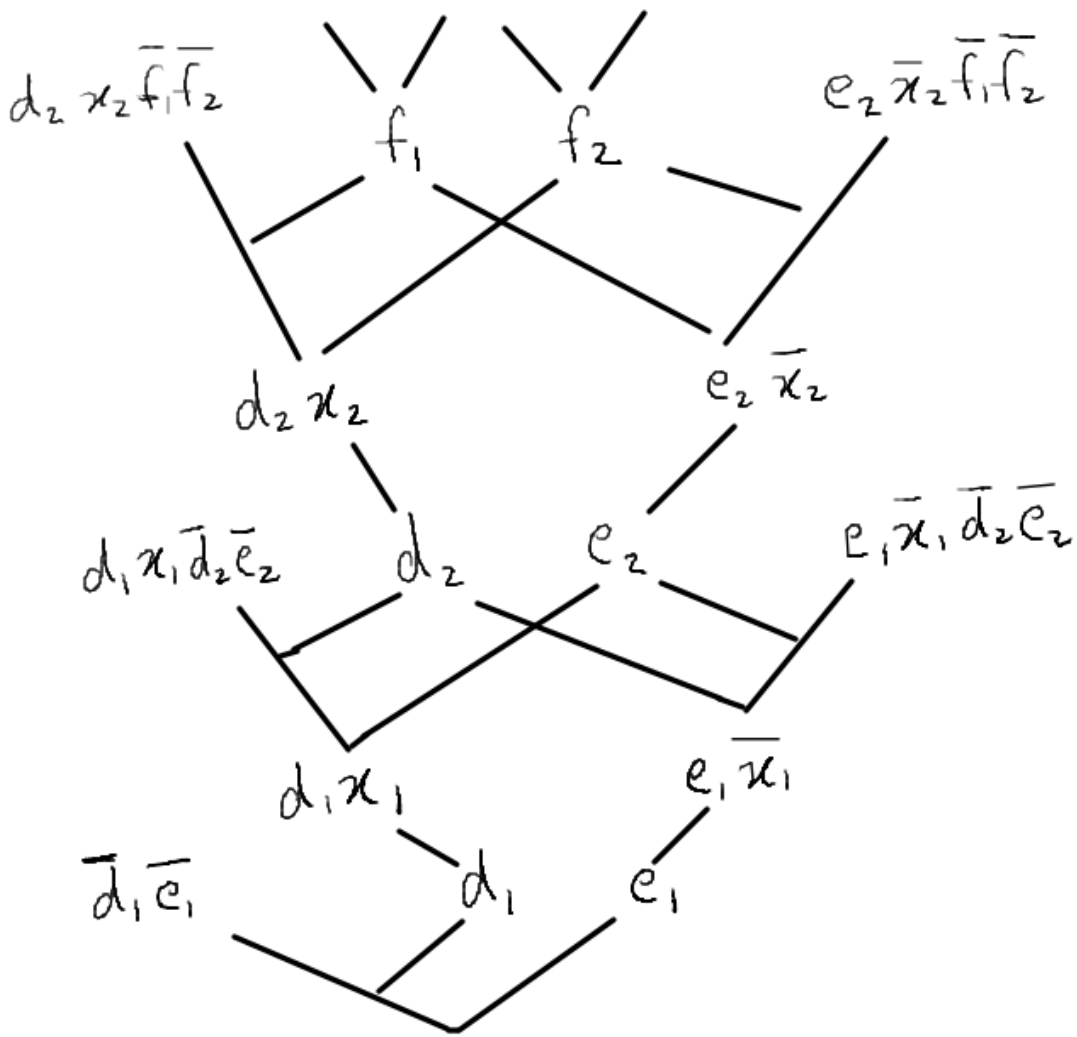
$(\bar{d}_1 \bar{e}_1)$

$(d_1 x_1 \bar{d}_2 \bar{e}_2)\ (e_1 \bar{x}_1 \bar{d}_2 \bar{e}_2)$

$(d_2 x_2 \bar{f}_1 \bar{f}_2)(e_2 \bar{x}_2 \bar{f}_1 \bar{f}_2)$

$(x_1 f_1)\ (\bar{x}_1 f_1)$

$(x_2 f_2)\ (\bar{x}_2 f_2)$



The Expansion framework

Unfolding the universal quantifier:

$$\forall u Q\vec{x} \cdot F(u, \vec{x}) \text{ is true}$$

$$\Updownarrow$$

$$[Q\vec{x} \cdot F(0, \vec{x})] \wedge [Q\vec{x} \cdot F(1, \vec{x})] \text{ is true}$$

$$\Updownarrow$$

$$Q\vec{x}^{u/0} Q\vec{x}^{u/1} \cdot \left[F(0, \vec{x}^{u/0}) \wedge F(1, \vec{x}^{u/1})\right] \text{ is true}$$

The Expansion framework

Unfolding the universal quantifier:

$$\forall u Q\vec{x} \cdot F(u, \vec{x}) \text{ is true}$$

$$\Updownarrow$$

$$[Q\vec{x} \cdot F(0, \vec{x})] \wedge [Q\vec{x} \cdot F(1, \vec{x})] \text{ is true}$$

$$\Updownarrow$$

$$Q\vec{x}^{u/0} Q\vec{x}^{u/1} \cdot \left[F(0, \vec{x}^{u/0}) \wedge F(1, \vec{x}^{u/1})\right] \text{ is true}$$

- Expand the initial formula judiciously, on the fly.

  Then use standard resolution.
- Formula size blow-up? Not always too bad.

Limited Expansion

$$\exists (X_{i,j})_{n\times n} \forall z \exists a_1 \dots a_n b_1 \dots b_n \begin{array}{ll} (X_{ij}\ z\ a_i) & i,j \in [n] \\ (\overline{X}_{ij}\ \overline{z}\ b_j) & i,j \in [n] \\ (\overline{a}_1 \dots \overline{a}_n) & \\ (\overline{b}_1 \dots \overline{b}_n) & \end{array}$$

Initially $2n^2 + 2$ clauses.

Limited Expansion

$$\exists (X_{i,j})_{n\times n} \forall z \exists a_1 \dots a_n b_1 \dots b_n \begin{array}{ll} (X_{ij}\ z\ a_i) & i,j \in [n] \\ (\overline{X}_{ij}\ \overline{z}\ b_j) & i,j \in [n] \\ (\overline{a}_1 \dots \overline{a}_n) & \\ (\overline{b}_1 \dots \overline{b}_n) & \end{array}$$

Initially $2n^2 + 2$ clauses. After expansion, $2n^2 + 4$ clauses.
Only $2n^2 + 2$ expanded clauses really needed.

$$\begin{array}{ll} (X_{ij}\ a_i^{z/0}) & i,j \in [n] \\ (\overline{X}_{ij}\ b_j^{z/1}) & i,j \in [n] \\ (\overline{a}_1^{z/0} \dots \overline{a}_n^{z/0}) & \\ (\overline{b}_1^{z/1} \dots \overline{b}_n^{z/1}) & \\ (\overline{a}_1^{z/1} \dots \overline{a}_n^{z/1}) & \text{useless} \\ (\overline{b}_1^{z/0} \dots \overline{b}_n^{z/0}) & \text{useless} \end{array}$$

Expansion-based systems

- $\forall$Exp+Res [Janota,Marques-Silva 2015]
  - For each existential variable, specify the copy – values of all universals to the left.

Expansion-based systems

- $\forall$Exp+Res [Janota,Marques-Silva 2015]
  - For each existential variable, specify the copy – values of all universals to the left.
- IR (Instantiation and Resolution) [Beyersdorff,Chew,Janota 2014]
  - When using an axiom, for each existential variable, specify a set of copies – values of all universals to the left, in this axiom.
  - At any stage, further restrict the set of copies by extending values to more universals

Reusing Expansion

$\exists e_1 \forall u_1 \exists c_1 \exists d_1 \quad \exists e_2 \forall u_2 \exists c_2 \exists d_2$

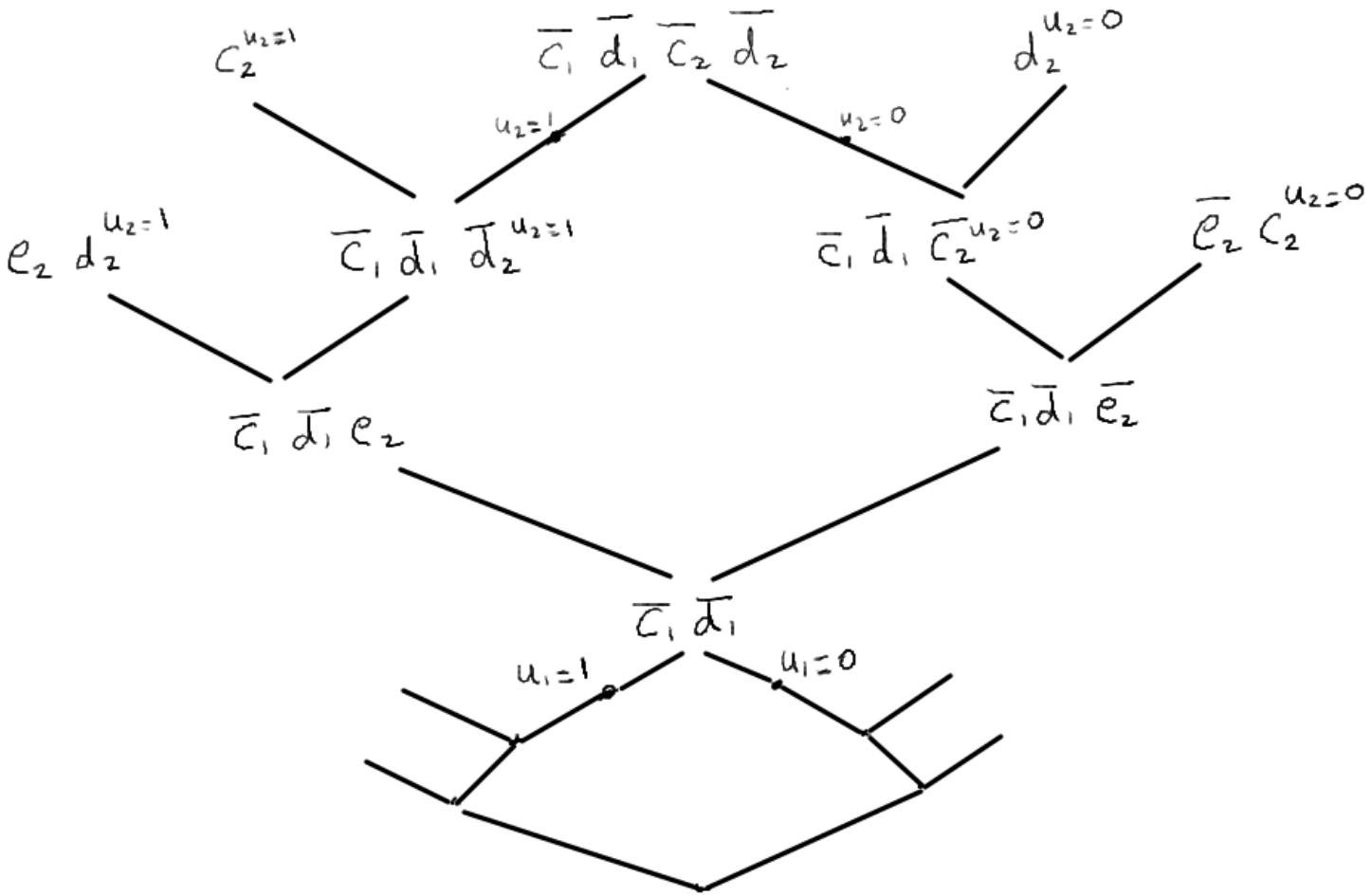
$(\bar{e}_1\ c_1)$ $(e_1\ d_1)$ $(\bar{e}_2\ c_2)$ $(e_2\ d_2)$ $(\bar{c}_1\ d_1\ \bar{c}_2\ d_2)$

$(\bar{u}_1\ c_1)$ $(u_1\ d_1)$ $(\bar{u}_2\ c_2)$ $(u_2\ d_2)$

Reusing Expansion

$\exists e_1 \forall u_1 \exists c_1 \exists d_1 \quad \exists e_2 \forall u_2 \exists c_2 \exists d_2$

$(\bar{e}_1\ c_1)\ (e_1\ d_1)\ (\bar{e}_2\ c_2)\ (e_2\ d_2)\ (\bar{c}_1\ \bar{d}_1\ \bar{c}_2\ \bar{d}_2)$
$(\bar{u}_1\ c_1)\ (u_1\ d_1)\ (\bar{u}_2\ c_2)\ (u_2\ d_2)$



The CDCL paradigm in SAT solvers

- SAT solving: revolutionized by **CDCL** Conflict-driven Clause Learning
  - Start as in DPLL.
  - On reaching a falsified assignment, don't just backtrack.
  - Learn a new clause in the process.
- Speeds up search:
  - Allows backing further up.
  - Enables more subsequent propagations.

The CDCL paradigm in SAT solvers

- SAT solving: revolutionized by **CDCL** Conflict-driven Clause Learning
  - Start as in DPLL.
  - On reaching a falsified assignment, don't just backtrack.
  - Learn a new clause in the process.
- Speeds up search:
  - Allows backing further up.
  - Enables more subsequent propagations.
- QCDCL: adaptation to QBF.

The Merging paradigm

- QBF proof systems based on universal reduction, expansion:
  None of these systems properly explains QCDCL runs.

  The learning process in QCDCL seems to remember when $P_\forall$ has a non-trivial srategy for some variable.

The Merging paradigm

- QBF proof systems based on universal reduction, expansion:
  None of these systems properly explains QCDCL runs.

  The learning process in QCDCL seems to remember when $P_\forall$ has a non-trivial srategy for some variable.
- Long-distance resolution – permit merging of complementary universal literals; $u \vee \neg u = u^*$.

The Merging paradigm

- QBF proof systems based on universal reduction, expansion:
  None of these systems properly explains QCDCL runs.

  The learning process in QCDCL seems to remember when $P_\forall$ has a non-trivial srategy for some variable.
- Long-distance resolution – permit merging of complementary universal literals; $u \vee \neg u = u^*$.
- Creating seeming tautologies can be meaningful and sound.

  $\exists x \forall u (x \vee u)(\neg x \vee \neg u)$

$$\dfrac{\dfrac{x \vee u \qquad \neg x \vee \neg u}{u^*}}{\square}$$

The Merging paradigm

- QBF proof systems based on universal reduction, expansion:
  None of these systems properly explains QCDCL runs.

  The learning process in QCDCL seems to remember when $P_\forall$ has a non-trivial srategy for some variable.
- Long-distance resolution – permit merging of complementary universal literals; $u \vee \neg u = u^*$.
- Creating seeming tautologies can be meaningful and sound.

  $\exists x \forall u (x \vee u)(\neg x \vee \neg u)$

  $$\dfrac{\dfrac{x \vee u \qquad \neg x \vee \neg u}{u^*}}{\square}$$
- Long-Distance QResolution LD-Q-Res.
  - Allow u and $\neg u$ to be combined into u^*, provided u right of pivot.
  - Disallow resolution with pivot x if $u < x$ and antecedents contain "conflicting" $u, \neg u, u^*$.

An LDQRes refutation

$\exists d_1 \exists e_1 \forall x_1$

$\exists d_2 \exists e_2 \forall x_2$

$\exists f_1 \exists f_2$

$(\bar{d}_1 \bar{e}_1)$

$(d_1 x_1 \bar{d}_2 \bar{e}_2)\ (e_1 \bar{x}_1 \bar{d}_2 \bar{e}_2)$

$(d_2 x_2 \bar{f}_1 \bar{f}_2)(e_2 \bar{x}_2 \bar{f}_1 \bar{f}_2)$

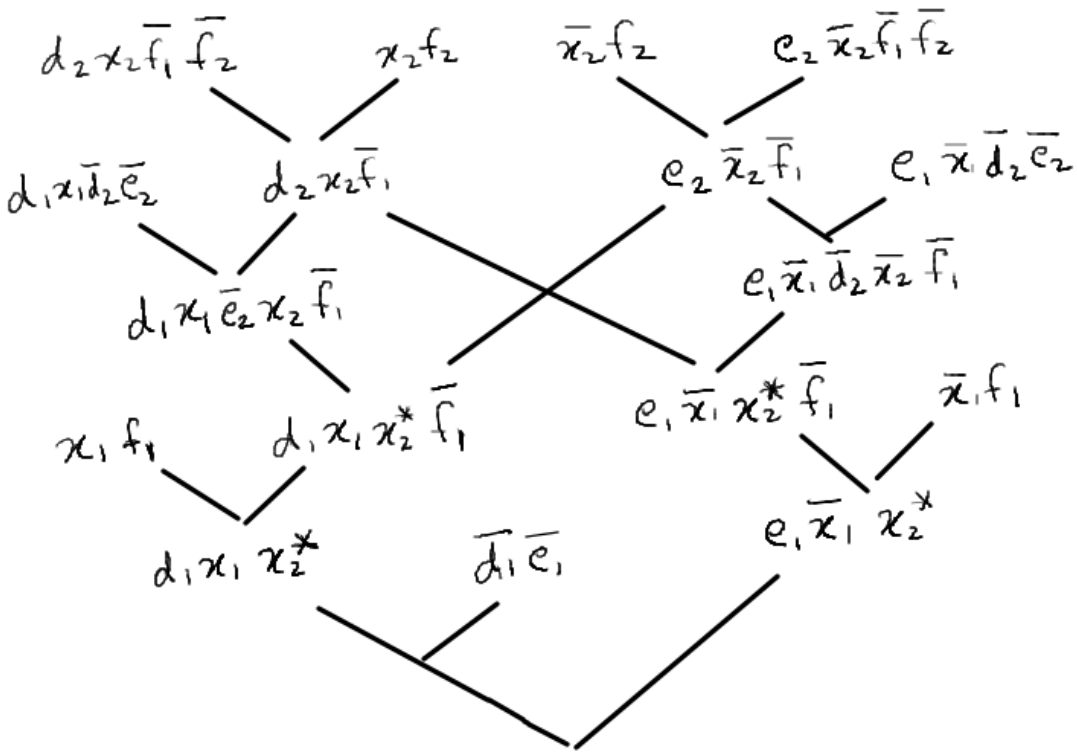
$(x_1 f_1)\ \ (\bar{x}_1 f_1)$

$(x_2 f_2)\ \ (\bar{x}_2 f_2)$

An LDQRes refutation

$\exists d_1 \exists e_1 \forall x_1$
$\exists d_2 \exists e_2 \forall x_2$
$\exists f_1 \exists f_2$

$(\bar{d}_1\ \bar{e}_1)$
$(d_1\ x_1\ \bar{d}_2\ \bar{e}_2)\ (e_1\ \bar{x}_1\ \bar{d}_2\ \bar{e}_2)$
$(d_2\ x_2\ \bar{f}_1\ \bar{f}_2)\ (e_2\ \bar{x}_2\ \bar{f}_1\ \bar{f}_2)$
$(x_1\ f_1)\ \ (\bar{x}_1\ f_1)$
$(x_2\ f_2)\ \ (\bar{x}_2\ f_2)$



Wider applicability of Merging

Proof-theoretically, merging of literals is sound in other settings too.

- Q-Res with merging $\rightarrow$ LD-Q-Res.
  [Zhang,Malik 2002],[Balabanov,Jiang 2012],[Egly,Lonsing,Widl 2013]
  (underlies, but is more powerful than, QCDCL.)
- QU-Res with merging on existential pivots $\rightarrow$LQU-Res.
- QU-Res with merging on all pivots $\rightarrow$ LQU$^+$-Res.
  [Balabanov,Jiang,Widl 2014]
- merging in annotated expansions in IR $\rightarrow$ IRM.
  [Beyersdorff,Chew,Janota 2014,15,19]

Wider applicability of Merging

Proof-theoretically, merging of literals is sound in other settings too.

- Q-Res with merging $\rightarrow$ LD-Q-Res.
  [Zhang,Malik 2002],[Balabanov,Jiang 2012],[Egly,Lonsing,Widl 2013]
  (underlies, but is more powerful than, QCDCL.)
- QU-Res with merging on existential pivots $\rightarrow$LQU-Res.
- QU-Res with merging on all pivots $\rightarrow$ LQU$^+$-Res.
  [Balabanov,Jiang,Widl 2014]
- merging in annotated expansions in IR $\rightarrow$ IRM.
  [Beyersdorff,Chew,Janota 2014,15,19]

A somewhat different use of merging: MergeRes

Separate out partial strategies of $P_\forall$ player; perform resolution on existential sub-clauses while building up the strategies.
[Beyersdorff,Blinkhorn,M JAR20]

Discarding spurious dependencies

- In the 2-player game, while assigning a value to u, $P_\forall$ knows the values of all existential x preceding u. But does (s)he really need to know them all?

Discarding spurious dependencies

- In the 2-player game, while assigning a value to u, $P_\forall$ knows the values of all existential x preceding u. But does (s)he really need to know them all?
- $\exists x \exists y \forall u \exists z \quad [x = z] \wedge [y = u]$ is false. But
  - The only winning strategy for $P_\forall$ is $u = \neg y$.
  - $P_\forall$ has no use for info about x.

Solvers and proof systems with dependency schemes

- Can the *structure of the clauses*
  – (clause-variable incidence, clause-pivot-clause connectivity, ...) –
  help identify such useless "dependencies"?

  Yes; some can be identified.
  (Proving soundness becomes even more complicated.)

Solvers and proof systems with dependency schemes

- Can the *structure of the clauses*
  – (clause-variable incidence, clause-pivot-clause connectivity, ...) –
  help identify such useless "dependencies"?

  Yes; some can be identified.
  (Proving soundness becomes even more complicated.)
- Does identifying them help with shorter solver runs/proofs?

  Not always, but often enough.

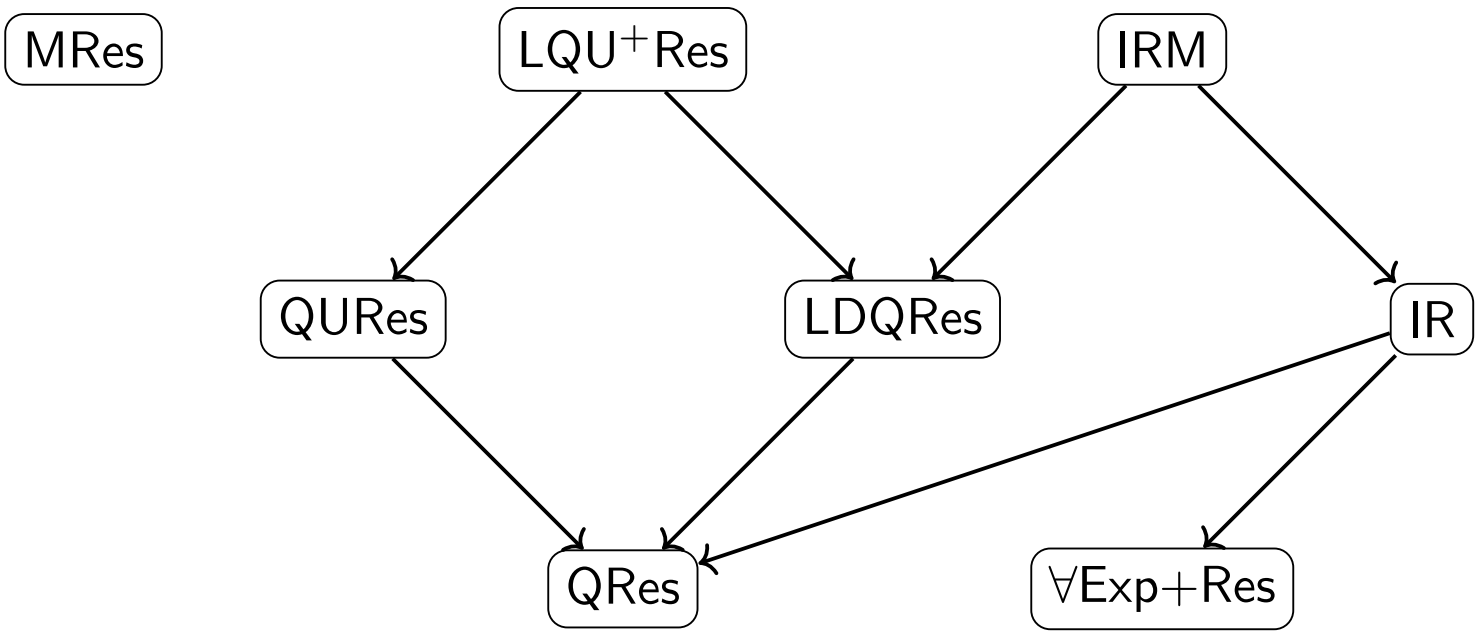
Solvers and proof systems with dependency schemes

- Can the *structure of the clauses*
  – (clause-variable incidence, clause-pivot-clause connectivity, ...) –
  help identify such useless "dependencies"?

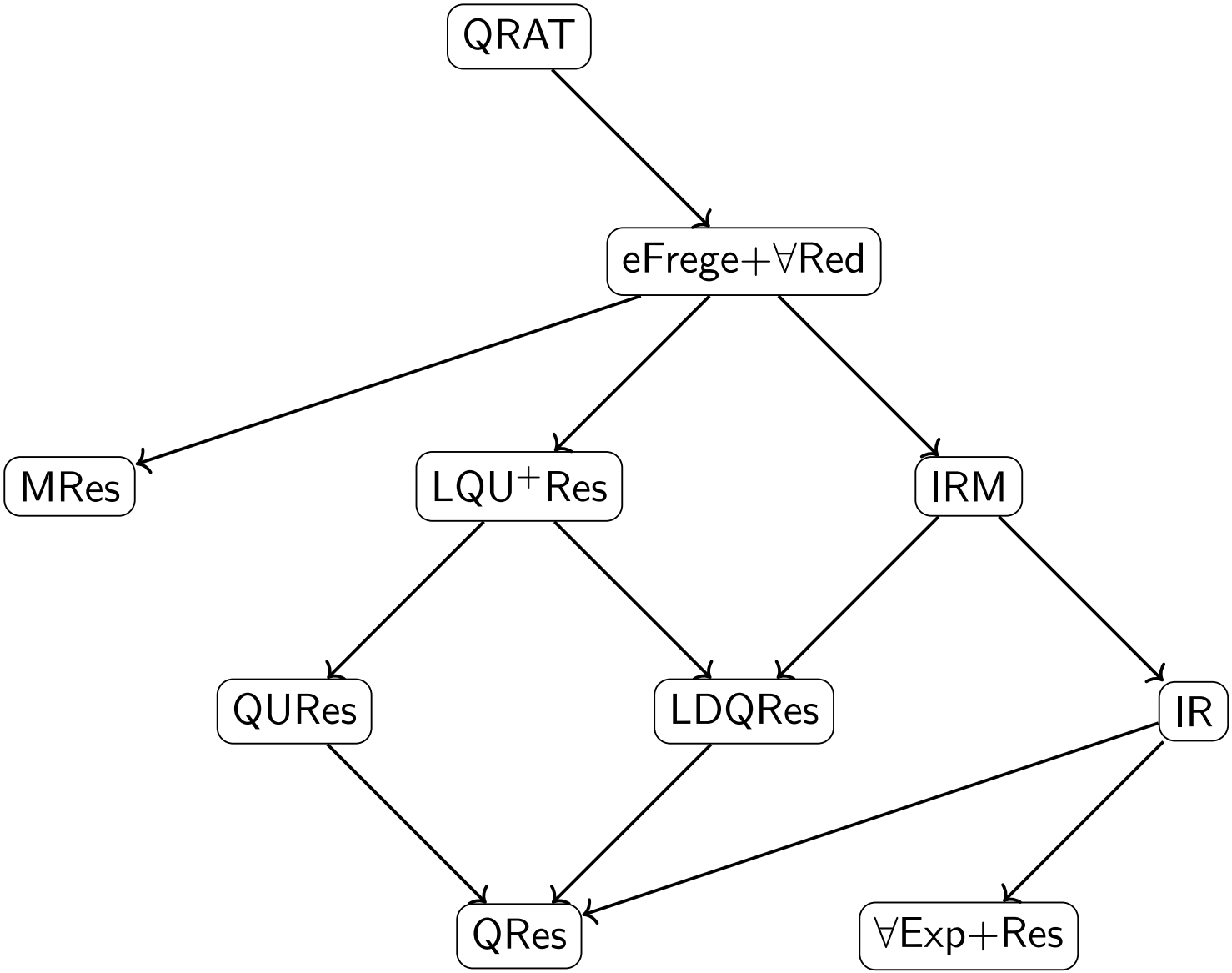
  Yes; some can be identified.
  (Proving soundness becomes even more complicated.)
- Does identifying them help with shorter solver runs/proofs?

  Not always, but often enough.
- Dependency Schemes: Trivial, Standard, Reflexive resolution paths, Tautology-free resolution paths, Implication-free resolution paths.
  [Biere,Lonsing 2010] [Samer,Szeider 2009] [Slivovsky,Szeider 2012,16] [Beyersdorff,Blinkhorn,Peitl 2020,21] ...

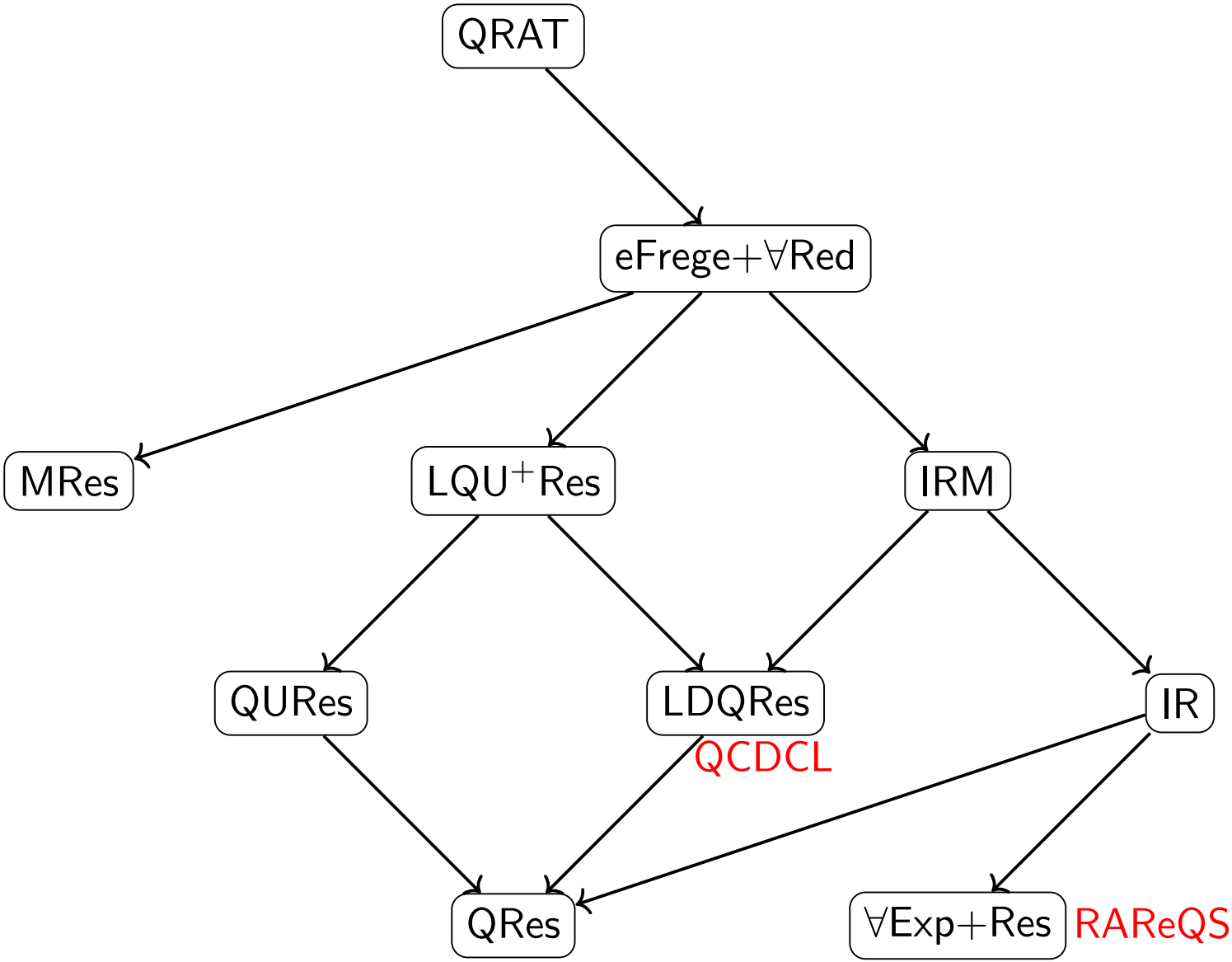
The simulation order



The simulation order



The simulation order



Proving lower bounds for QBF proof systems

- Each QBF proof system P, restricted to QBFs with no universal variables, is a propositional proof system P'.
- Hardness in P' implies hardness in P.
- A QBF hard for P may contain subformulas hard for P', in a somewhat obfuscated way. Is its hardness in P due to P' hardness, or due to the obfuscation, or both?

Discounting propositional hardness

- To truly understand the strengths/weaknesses of QBF proof systems, discard purely propositional sources of hardness.

Discounting propositional hardness

- To truly understand the strengths/weaknesses of QBF proof systems, discard purely propositional sources of hardness.

  How to formalise this?

- Propositional proofs decide (un)satisfiability; NP-complete.
  Let the QBF proof system use an NP-oracle.

  Let a QBF solver use a SAT solver as a subroutine.

Discounting propositional hardness

- To truly understand the strengths/weaknesses of QBF proof systems, discard purely propositional sources of hardness.

  How to formalise this?

- Propositional proofs decide (un)satisfiability; NP-complete.
  Let the QBF proof system use an NP-oracle.

  Let a QBF solver use a SAT solver as a subroutine.

- eg in $P + \forall$Red systems, allow entailment for free;
  count only reduction steps.
  Lower bounds on this count are "genuine" QBF lower bounds.
  [Chen 2016] [Beyersdorff,Hinde,Pich 2017]

Lifting propositional lower bound techniques

- Short proofs can be made narrow.
  A modified relation, factoring in alternation depth, works for Q-Res and QU-Res. [Beyersdorff,Blinkhorn,Peitl,M 2022]

Lifting propositional lower bound techniques

- Short proofs can be made narrow.
  A modified relation, factoring in alternation depth, works for Q-Res and QU-Res. [Beyersdorff,Blinkhorn,Peitl,M 2022]
- Proofs yield interpolant circuits.
  Works for most resolution-based QBF systems.
  [Beyersdorff,Chew,M,Shukla 2017]

Lifting propositional lower bound techniques

- Short proofs can be made narrow.
  A modified relation, factoring in alternation depth, works for Q-Res and QU-Res. [Beyersdorff,Blinkhorn,Peitl,M 2022]
- Proofs yield interpolant circuits.
  Works for most resolution-based QBF systems.
  [Beyersdorff,Chew,M,Shukla 2017]
- Random formulas are hard.
  Difficult to formalise: what is "random enough"?
  (A fully random k-CNF matrix will, w.h.p., have a purely universal clause, so trivial to refute in any reduction-based system!)
  Works for some structured randomness, but uses new idea.

Transferring computational (circuit) lower bounds

Strategy Extraction.

- Key idea: Proofs contains information about countermodels.

Transferring computational (circuit) lower bounds

Strategy Extraction.

- Key idea: Proofs contains information about countermodels.
- Examine a proof.
- Construct a circuit of a special type for computing the winning strategy of the $P_\forall$ player.
- Circuit type: depends on the proof system
  Circuit size: depends on the proof size

Transferring computational (circuit) lower bounds

Strategy Extraction.

- Key idea: Proofs contains information about countermodels.
- Examine a proof.
- Construct a circuit of a special type for computing the winning strategy of the $P_\forall$ player.
- Circuit type: depends on the proof system
  Circuit size: depends on the proof size
- If the winning strategy is hard to compute in the relevant circuit model, then all proofs in the proof system must be large.

Used for many lower bounds.

Semantic measures for QBF hardness

- Cost of a formula: [Beyersdorff,Blinkhorn,Hinde 2018,19].
  Yields lower bounds for reduction-based systems,
  even for random formulas.
  Limited by universal block lengths.

Semantic measures for QBF hardness

- Cost of a formula: [Beyersdorff,Blinkhorn,Hinde 2018,19].
  Yields lower bounds for reduction-based systems,
  even for random formulas.
  Limited by universal block lengths.
- Weight of a formula: [Beyersdorff,Blinkhorn 2018,20]
  Not limited by block length.
  Yields lower bounds for expansion-based systems.

Semantic measures for QBF hardness

- Cost of a formula: [Beyersdorff,Blinkhorn,Hinde 2018,19].
  Yields lower bounds for reduction-based systems,
  even for random formulas.
  Limited by universal block lengths.
- Weight of a formula: [Beyersdorff,Blinkhorn 2018,20]
  Not limited by block length.
  Yields lower bounds for expansion-based systems.
- Gauge of a formula: [Beyersdorff,Böhm 2021]
  Yields lower bounds for a fragment of Q-Res and for QCDCL,
  for Σ^3 formulas of a specific type.

Semantic measures for QBF hardness

- Cost of a formula: [Beyersdorff,Blinkhorn,Hinde 2018,19].
  Yields lower bounds for reduction-based systems,
  even for random formulas.
  Limited by universal block lengths.
- Weight of a formula: [Beyersdorff,Blinkhorn 2018,20]
  Not limited by block length.
  Yields lower bounds for expansion-based systems.
- Gauge of a formula: [Beyersdorff,Böhm 2021]
  Yields lower bounds for a fragment of Q-Res and for QCDCL,
  for Σ^3 formulas of a specific type.

All these measures can be large even if winning strategy trivial to compute.

Summary

- Correspondence between QBF solvers and QBF proof systems not yet clean-and-clear. In fact, far from clear.
- Proof-theoretic studies establish provable limitations of some solvers.
- Proof-theoretic studies can suggest new heuristics.

Summary

- Correspondence between QBF solvers and QBF proof systems not yet clean-and-clear. In fact, far from clear.
- Proof-theoretic studies establish provable limitations of some solvers.
- Proof-theoretic studies can suggest new heuristics.

Thank You!