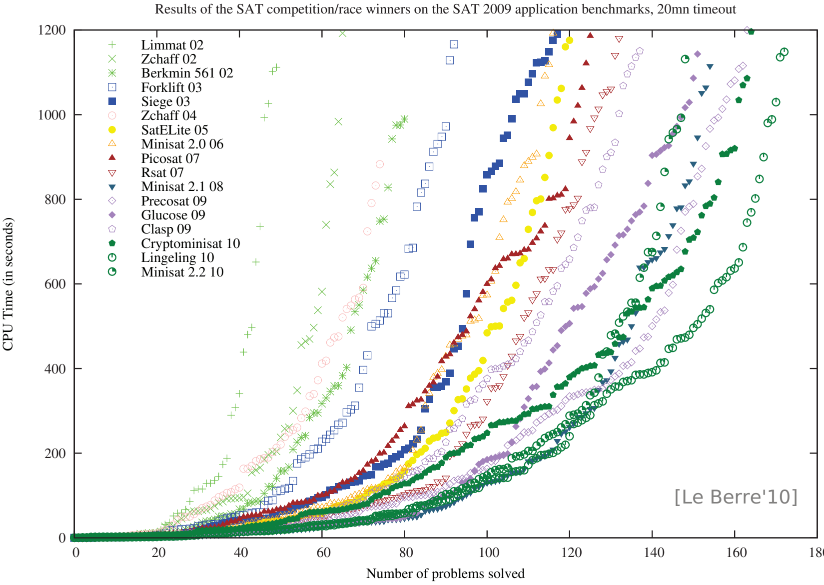
Quantified Boolean Formulas
Part 1

Uwe Egly

Knowledge-Based Systems Group
Institute of Information Systems
Vienna University of Technology

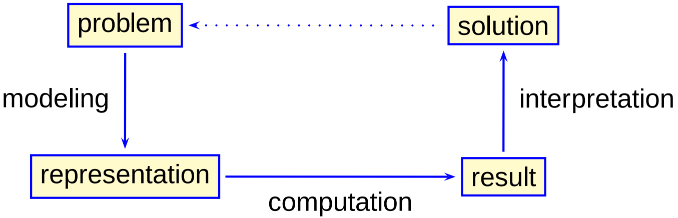
Results of the SAT 2009 application benchmarks

for leading solvers from 2002 to 2010



Success story of SAT: Why is it important?

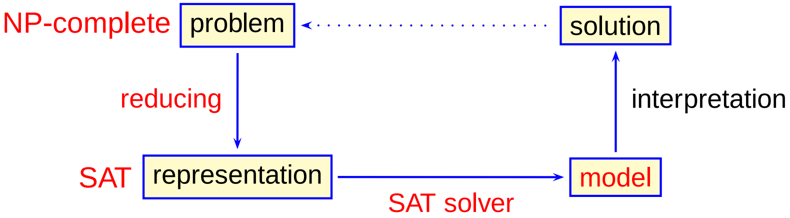
Allows us to implement problem solving programs rapidly



We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.

Success story of SAT: Why is it important?

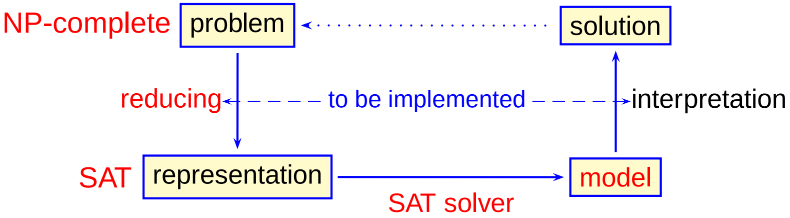
Allows us to implement problem solving programs rapidly



We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.

Success story of SAT: Why is it important?

Allows us to implement problem solving programs rapidly



We want to model a problem by compiling it into a suitable representation s.t. the result of the compiled problem can be interpreted as a solution to the original problem.

What if my problem is more difficult than SAT?

- We know how to implement solvers for NP-complete problems, e.g., planning, SAT for some equational logics, ...
- Prototypical implementation: reduce problem to a SAT problem and solve it with a “good” SAT solver
- Problem: What happens if the problem is to hard to be efficiently (polynomially) reduced to SAT?
- Solution: Use a more “expressive SAT problem” based on Quantified Boolean Formulas (QBFs)
- QBFs admit Boolean quantifiers in formulas and enable succinct problem representations for problems “harder than NP”

Outline

The syntax of quantified Boolean formulas (QBFs)

Let $\mathcal{P}$ be a set of propositional (Boolean) variables

Inductive definition of the set of QBFs (wrt $\mathcal{P}$)

B1: For every propositional variable $p \in \mathcal{P}$, p is a QBF

B2: For every truth constant $t \in \{\bot, \top\}$, t is a QBF

S1: If Φ is a QBF, then $\neg\Phi$ is a QBFs

S2: If Φ_1, Φ_2 are QBFs, then $\Phi_1 \circ \Phi_2$ ($\circ \in \{\wedge, \vee, \rightarrow\}$) are QBFs

S3: If Φ is a QBF, then $\mathrm{Q}p\,\Phi$ ($\mathrm{Q} \in \{\forall, \exists\}$, $p \in \mathcal{P}$) is a QBF

Further connectives like $\leftrightarrow$ or $\oplus$ can be defined as usual

Observations and examples

QBFs are allowed to be in non-prenex form, i.e., quantifiers are not only allowed in an initial prefix, but also deeply inside QBFs.

Example: $\forall p \left((\exists q\,(p \land q)) \rightarrow \exists r\,(r \lor p)\right)$

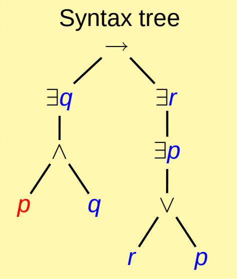
Free variables are allowed, i.e., there may be occurrences of propositional variables which have no quantification.

Example: $(\exists q\,(p \land q)) \rightarrow \exists r \exists p\,(r \lor p)$

Observations and examples

QBFs are allowed to be in non-prene … iers are not only allowed in an initial prefix, bu … e QBFs.

Example: $\forall p \; ((\exists q \; (p \wedge q)) -$



Free variables are allowed, i.e., there may be occurrences of propositional variables which have no quantification.

Example: $(\exists q \; (p \wedge q)) \rightarrow \exists r \exists p \; (r \vee p)$

Normal forms

Prenex normal form (PNF), prefix, matrix, PCNF, closed

- Let $\mathsf{Q}_i \in \{\forall, \exists\}$ and $p_i \in \mathcal{P}$. A QBF

$$\Phi = \mathsf{Q}_1 p_1 \ldots \mathsf{Q}_n p_n \, \psi$$

  is in prenex (normal) form (PNF) if ψ is purely propositional
- $\mathsf{Q}_1 p_1 \mathsf{Q}_2 p_2 \cdots \mathsf{Q}_n p_n$ is the prefix of Φ; ψ is the matrix of Φ.
- Φ is in PCNF if ψ is in CNF
- Φ is closed if the variables in ψ are in $\{p_1, \ldots, p_n\}$

Convention: Each quantifier binds another variable and bound variables do not occur free.

Examples for normal forms

closed, non-prenex	$(\forall x \forall y\ (x \rightarrow y)) \wedge (\exists u \exists v\ (u \wedge v))$
open, non-prenex	$(\forall x \forall y\ (x \rightarrow y)) \wedge (\exists u\ (u \wedge v))$
closed, PCNF	$\forall x \forall y \exists z\ ((z \vee x \vee y) \wedge (\neg z \vee x \vee y))$
alternative notation 1	$\forall x\, y\, z\, ((z \vee x \vee y) \wedge (\neg z \vee x \vee y))$
alternative notation 2	$\forall P\, ((z \vee x \vee y) \wedge (\neg z \vee x \vee y))$ if $P = \{x, y, z\}$

Generating a prenex form (cf predicate logic)

Apply the following rules until a PNF is obtained

R_1	$\mathrm{Q}x\, \Phi \circ \mathrm{Q}y\, \Psi$	$\Rightarrow$	$\mathrm{Q}x\,\mathrm{Q}y\, (\Phi \circ \Psi)$	x not free in Ψ, y not free in Φ
R_2	$(\mathrm{Q}x\, \Phi) \rightarrow \Psi$	$\Rightarrow$	$\mathrm{Q}^{-}x\, (\Phi \rightarrow \Psi)$	x not free in Ψ
R_3	$\Phi \rightarrow (\mathrm{Q}y\, \Psi)$	$\Rightarrow$	$\mathrm{Q}y\, (\Phi \rightarrow \Psi)$	y not free in Φ
R_4	$\forall x\, \Phi \wedge \forall y\, \Psi$	$\Rightarrow$	$\forall x\, (\Phi \wedge \Psi[y/x])$	
R_5	$\exists x\, \Phi \vee \exists y\, \Psi$	$\Rightarrow$	$\exists x\, (\Phi \vee \Psi[y/x])$	

Remarks

- $\mathrm{Q} \in \{\forall, \exists\}$, $(\mathrm{Q}, \mathrm{Q}^{-})$ is $(\forall, \exists)$ or $(\exists, \forall)$ and $\circ \in \{\wedge, \vee\}$
- In general, the PNF of Φ is not unique
  (depends, e.g., on rule choice: R_1 vs R_4 if both are applicable)
- Φ and all of its prenex forms are logically equivalent (why?)

Outline

The semantics of QBFs

- Based on interpretations I represented as sets of atoms
- An atom p is true under I iff $p \in I$

Inductive definition of the truth value, $\nu_I(\Phi)$, of a QBF Φ under an interpretation I:

1. if $\Phi = \top$, then $\nu_I(\Phi) = 1$;
2. if $\Phi = p \in \mathcal{P}$, then $\nu_I(\Phi) = 1$ if $p \in I$, and $\nu_I(\Phi) = 0$ otherwise;
3. if $\Phi = \neg\Psi$, then $\nu_I(\Phi) = 1 - \nu_I(\Psi)$;
4. if $\Phi = (\Phi_1 \wedge \Phi_2)$, then $\nu_I(\Phi) = min(\{\nu_I(\Phi_1), \nu_I(\Phi_2)\})$;
5. if $\Phi = \forall p\, \Psi$, then $\nu_I(\Phi) = \nu_I(\Psi[p/\top] \;\wedge\; \Psi[p/\bot])$;
6. if $\Phi = \exists p\, \Psi$, then $\nu_I(\Phi) = \nu_I(\Psi[p/\top] \;\vee\; \Psi[p/\bot])$.

Truth conditions for $\bot$, $\vee$, $\rightarrow$, $\leftrightarrow$ follow from the above "as usual"

The semantics of QBFs (cont'd)

Notations

- Φ is true under I iff $\nu_I(\Phi) = 1$, otherwise Φ is false under I
- If $\nu_I(\Phi) = 1$, then I is a model of Φ (and Φ is satisfiable)
- If Φ is true under any interpretation, then Φ is valid
- Two sets of QBFs (or ordinary Boolean formulas) are logically equivalent iff they possess the same models

Observations

- A closed QBF is either valid or unsatisfiable, because it is either true under each interpretation I or false under each I.
- Hence, for closed QBFs, there is no need to refer to particular interpretations.

Evaluation of a QBF with a free variable

Let Φ be $\exists x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ and $I = \{y\}$

$$\begin{aligned}
\nu_I(\Phi) &= \nu_I((\neg\top \vee y) \wedge (\top \vee \neg y) \vee (\neg\bot \vee y) \wedge (\bot \vee \neg y)) \\
&= \max\{\min\{\nu_I(\neg\top \vee y), \underbrace{\nu_I(\top \vee \neg y)}_{=1}\} \min\{\underbrace{\nu_I(\neg\bot \vee y)}_{=1}, \nu_I(\bot \vee \neg y)\}\} \\
&= \max\{\nu_I(\neg\top \vee y), \nu_I(\bot \vee \neg y)\} \\
&= \max\{\nu_I(y), \nu_I(\neg y)\} \quad = \quad 1
\end{aligned}$$

- I contains (some) free variables of Φ
- Evaluation result here is independent from I
- A similar evaluation of $\forall x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ results 0

Evaluation of a QBF with a free variable

Let Φ be $\exists x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ and $I = \{y\}$

$$
\begin{aligned}
\nu_I(\Phi) &= \nu_I((\neg\top \vee y) \wedge (\top \vee \neg y) \vee (\neg\bot \vee y) \wedge (\bot \vee \neg y)) \\
&= \max\{\min\{\nu_I(\neg\top \vee y), \underbrace{\nu_I(\top \vee \neg y)}_{=1}\}\ \min\{\underbrace{\nu_I(\neg\bot \vee y)}_{=1}, \nu_I(\bot \vee \neg y)\}\} \\
&= \max\{\nu_I(\neg\top \vee y), \nu_I(\bot \vee \neg y)\} \\
&= \max\{\nu_I(y), \nu_I(\neg y)\} \ = \ 1
\end{aligned}
$$

- I contains (some) free variables of Φ
- Evaluation result here is independent from I
- A similar evaluation of $\forall x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ results 0

Evaluation of a QBF with a free variable

Let Φ be $\exists x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ and $I = \{y\}$

$$
\begin{aligned}
\nu_I(\Phi) &= \nu_I((\neg\top \vee y) \wedge (\top \vee \neg y) \vee (\neg\bot \vee y) \wedge (\bot \vee \neg y)) \\
&= \max\{\min\{\nu_I(\neg\top \vee y), \underbrace{\nu_I(\top \vee \neg y)}_{=1}\} \min\{\underbrace{\nu_I(\neg\bot \vee y)}_{=1}, \nu_I(\bot \vee \neg y)\}\} \\
&= \max\{\nu_I(\neg\top \vee y), \nu_I(\bot \vee \neg y)\} \\
&= \max\{\nu_I(y), \nu_I(\neg y)\} \quad = \quad 1
\end{aligned}
$$

- I contains (some) free variables of Φ
- Evaluation result here is independent from I
- A similar evaluation of $\forall x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ results 0

Evaluation of a QBF with a free variable

Let Φ be $\exists x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ and $I = \{y\}$

$$
\begin{aligned}
\nu_I(\Phi) &= \nu_I((\neg\top \vee y) \wedge (\top \vee \neg y) \vee (\neg\bot \vee y) \wedge (\bot \vee \neg y)) \\
&= \max\{\min\{\nu_I(\neg\top \vee y), \underbrace{\nu_I(\top \vee \neg y)}_{=1}\} \min\{\underbrace{\nu_I(\neg\bot \vee y)}_{=1}, \nu_I(\bot \vee \neg y)\}\} \\
&= \max\{\nu_I(\neg\top \vee y), \nu_I(\bot \vee \neg y)\} \\
&= \max\{\nu_I(y), \nu_I(\neg y)\} \ = \ 1
\end{aligned}
$$

- I contains (some) free variables of Φ
- Evaluation result here is independent from I
- A similar evaluation of $\forall x\ ((\neg x \vee y) \wedge (x \vee \neg y))$ results 0

More examples of QBF evaluations

Let φ be $(p \rightarrow q) \wedge (q \rightarrow p)$

- $\exists p \, \exists q \; \varphi$ is true (since φ is sat and all its variables are bound)
- $\forall p \, \forall q \; \varphi$ is false (since φ is not valid and all its vars are bound)
- $\exists q \, \forall p \; \varphi$ is false
- $\forall p \, \exists q \; \varphi$ is true ➥ quantifier ordering matters!

Satisfiability and validity can be expressed in QBFs:

- $\exists V \, \psi(V)$ is true iff ψ is satisfiable
- $\forall V \, \psi(V)$ is true iff ψ is valid

Certificates for QBFs

A certificate provides evidence of satisfiability of a QBF

- One possibility to certify the truth of a closed QBF: Witness functions/formulas (WFs) for existential quantifiers which depend on (some) dominating universal quantifiers

  Example: $\forall x_1 \forall x_2 \exists y\ (x_1 \vee x_2 \vee \neg y) \wedge (\neg x_1 \vee y)$

☞ Take $y = x_1$: $\forall x_1 \forall x_2\ (x_1 \vee x_2 \vee \neg x_1) \wedge (\neg x_1 \vee x_1)$ becomes true

☞ This can be checked with a validity checker for propositional logic

- WFs are sometimes the constructed solution to a problem

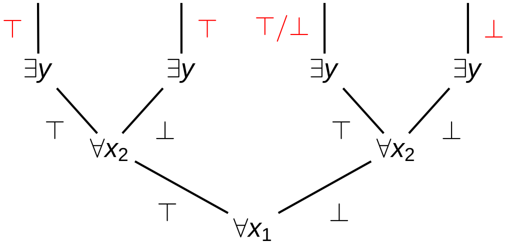
For a broader discussion, see V. Balabanov, J.-H. R. Jiang: Resolution proofs and Skolem functions in QBF evaluation and applications CAV 2011. [link]

Certificates for QBFs (cont'd)

A certificate provides evidence of satisfiability of a QBF

- Others are e.g. tree-like strategies for the choice of truth values of $\exists$ quantifiers depending on dominating $\forall$ ones

Example: $\forall x_1\, \forall x_2\, \exists y\; (x_1 \vee x_2 \vee \neg y) \wedge (\neg x_1 \vee y)$



Outline

For which classes of problems do we need QBFs?

- NP-complete problems can be efficiently reduced to SAT
- Q: Why is another SAT formalism based on QBFs needed?
- A: There are even “harder” problems than SAT
  A Garey-Johnson like compendium of such problem can be found here [link]
- The SAT problem for QBFs provides a target formalism to which such computationally hard problems can be reduced

Informal definition of important complexity classes

class	model of computation	expense wrt resource
P	deterministic	polynomial time
NP	non-deterministic	polynomial time
PSPACE	deterministic	polynomial space
NPSPACE	non-deterministic	polynomial space
EXPTIME	deterministic	exponential time
NEXPTIME	non-deterministic	exponential time

Relations between some complexity classes

- $\mathrm{P} \subseteq_{=?} \mathrm{NP} \subseteq_{=?} \mathrm{PSPACE}$
- $\mathrm{PSPACE} = \mathrm{NPSPACE}$
- $\mathrm{PSPACE} \subseteq_{=?} \mathrm{EXPTIME}$
- $\mathrm{P} \subset \mathrm{EXPTIME}$
- $\mathrm{NP} \subset \mathrm{NEXPTIME}$

The polynomial hierarchy (PH)

The PH consists of classes Σ_k^P, Π_k^P, and Δ_k^P, where

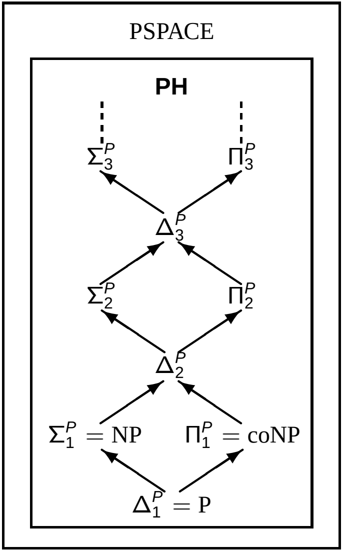
$$\Sigma_0^P = \Pi_0^P = \Delta_0^P = \mathrm{P};$$

and for $k \geq 1$:

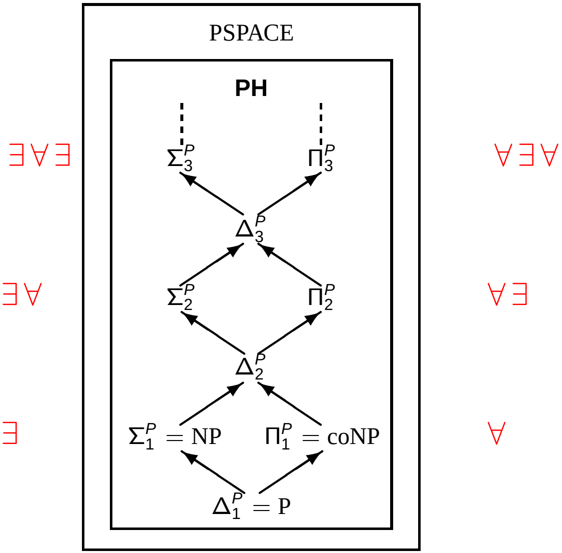
$$\begin{aligned} \Delta_{k+1}^P &= \mathrm{P}^{\Sigma_k^P}; \\ \Sigma_{k+1}^P &= \mathrm{NP}^{\Sigma_k^P}; \\ \Pi_{k+1}^P &= \mathrm{co}-\Sigma_{k+1}^P. \end{aligned}$$

A^B: set of decision problems solvable by a Turing machine in class A augmented by an oracle for some complete problem in class B

The polynomial hierarchy (PH) (cont'd)



The polynomial hierarchy (PH) (cont'd)



Prenex QBFs and complexity classes (Wrathall 1976)

Eval. problems for prenex QBFs and their complexities

Given a propositional formula φ with its atoms partitioned into $i \geq 1$ pairwise distinct sets $V_1, \ldots, V_i$, deciding whether $\exists V_1 \forall V_2 \ldots \mathsf{Q}_i V_i\, \varphi$ is true is Σ_i^P-complete, where $\mathsf{Q}_i = \exists$ if i is odd and $\mathsf{Q}_i = \forall$ if i is even, Dually, deciding whether $\forall V_1 \exists V_2 \ldots \mathsf{Q}_i' V_i\, \varphi$ is true is Π_i^P-complete, where $\mathsf{Q}_i' = \forall$ if i is odd and $\mathsf{Q}_i = \exists$ if i is even.

Examples of evaluation problems (EPs)

- The EP of $\exists V_1\, \varphi(V_1)$ is Σ_1^P-complete (= NPC)
- The EP of $\forall V_1\, \varphi(V_1)$ is Π_1^P-complete (= co-NPC)
- The EP of $\forall V_1 \exists V_2 \forall V_3\, \varphi(V_1, V_2, V_3)$ is Π_3^P-complete

➥ Important for reductions: If we know the complexity of our problem, we can choose the appropriate quantifier prefix for the target QBF

How to handle non-prenex QBFs?

Extend the complexity landscape to arbitrary closed QBFs

- Take the maximal number of quantifier alternations along a path in the syntax tree of a QBF into account
- Almost all QBFs can be translated into equivalent QBFs in PNF without increasing the number of quantifier alternations (Which are the problematic QBFs?)
- Translation procedure is fast but non-deterministic
- Can heavily influence the performance of QBF solvers
- Details in E. et al. Comparing Different Prenexing Strategies for Quantified Boolean Formulas. Proc. SAT 2003, pp. 214-228.

Outline

Generating PCNFs

A QBF in prenex conjunctive normal form (PCNF)

- starts with a quantifier prefix and
- consists of a conjuction of clauses (=disjunction of literals) (often represented as a set of clauses).
- Clauses sometimes represented as sets of literals (can cause problems)

Why are formulas in PCNF necessary?

- Most QBF solvers require the input being in PCNF
- ☞ Translation procedure required
- This procedure can be based on distributivity or Tseitin

Generating PCNFs (cont'd)

The Tseitin-based algorithm works in three steps

1. Generate a prenex form $\Psi_p : Q_i X_i \cdots Q_k X_k \, \psi$ of the input QBF Ψ. Then the matrix ψ is purely propositional.
2. Use Tseitin's translation to transform ψ into CNF.
3. Place the $\exists$ quantifiers for the newly introduced variables $\ell_1, \cdots \ell_m$ abbreviating $\varphi_1, \ldots, \varphi_m$ "correctly", e.g.,
   - place all the new $\exists$ at the end of the quantifier prefix, or
   - place $\exists \ell_i$ ($1 \leq i \leq m$) after all quantifiers of those variables which occur in φ_i.

Outline

Tricky use of Boolean quantification

Trick 1: Introduce abbreviations for subformulas

- Given propositional formula φ:

$$(A \lor \neg B \lor C \lor D) \land (A \lor \neg B \lor C \lor \neg E) \land (A \lor \neg B \lor C \lor F)$$

- Idea: Introduce a “definition” to abbreviate $A \lor \neg B \lor C$
- Obtain a QBF Φ:

$$\exists y\,(y \leftrightarrow A \lor \neg B \lor C) \land (y \lor D) \land (y \lor \neg E) \land (y \lor F)$$

- $A \lor \neg B \lor C$ occurs only once!
- Φ is logically equivalent to φ (mainly because of $\exists y$)

Most examples from U. Bubeck, H. Kleine Büning: Encoding Nested Boolean Functions as Quantified Boolean Formulas. JSAT 8:101-116 (2012). [link]

Tricky use of Boolean quantification (cont'd)

Trick 2: "Unify" conjunctively connected instances

- Given propositional formula φ:

$$\varphi_1(A_1, B_1) \wedge \varphi_1(A_2, B_2) \wedge \varphi_1(A_3, B_3)$$

- We have three different instances of $\varphi_1(A, B)$
- Obtain a QBF Φ:

$$\forall u \forall v \ \Big(\bigvee_{i=1}^{3} ((u \leftrightarrow A_i) \wedge (v \leftrightarrow B_i)) \Big) \rightarrow \varphi_1(u, v)$$

- φ_1 occurs only once!
- Φ is logically equivalent to φ

Tricky use of Boolean quantification (cont'd)

Trick 3: Non-copying iterative squaring

- Given formula $\Psi(x_0, x_n)$ with $n = 2^i$:

$$\exists x_1 \cdots \exists x_{n-1} \left(\varphi(x_0, x_2) \wedge \varphi(x_2, x_3) \wedge \cdots \wedge \varphi(x_{n-1}, x_n)\right)$$

- Idea: Take y in the middle and split the formula:

$$\Psi_{2^i}(x_0, x_n) \colon \exists y \, (\Psi_{2^{i-1}}(x_0, y) \wedge \Psi_{2^{i-1}}(y, x_n))$$

- Use Trick 2 and get $\Psi_{2^i}(x_0, x_n)$:

$$\exists y \forall u \forall v \left[\left(((u, v) \leftrightarrow (x_0, y)) \vee ((u, v) \leftrightarrow (y, x_n))\right) \rightarrow \Psi_{2^{i-1}}(u, v)\right]$$

- We come back to this trick in the example