# On the complexity of $k$-DQBF

Long-Hin Fung, Tony Tan
National Taiwan University

The 26th International Conference on Theory and Applications of Satisfiability Testing
04 - 08 July 2023
Alghero, Italy

# DQBF: Dependency Quantified Boolean Formula

QBF: $\forall x_1 \forall x_2 \forall x_3 \; \exists y_1 \; \forall x_4 \forall x_5 \; \exists y_2 \quad \varphi$

The value of $y_1$ is a function on $x_1, x_2, x_3$, i.e., depends on all $x_1, x_2, x_3$.

The value of $y_2$ is a function on $x_1, \ldots, x_5$.

DQBF: $\forall x_1 \forall x_2 \forall x_3 \; \exists y_1(x_1, x_3) \; \forall x_4 \forall x_5 \; \exists y_2(x_2, x_3, x_5) \quad \varphi$

The value of $y_1$ is a function on $x_1, x_3$.

The value of $y_2$ is a function on $x_2, x_3, x_5$.

**Theorem (Patterson and Reif, 1979)**
*Checking whether a DQBF formula is true or not is NEXP-complete.*

A lot of research on DQBF since 2012 and there is DQBF track in SAT competition.

# The theme of this talk — The intuitive version

- DQBF = SAT (in succinct form).

  This has been observed by many researchers.
  (Bubeck 2010, Frohlich, et. al. 2014, Balabanov, Jiang 2015, et. al.)

- Many results on SAT also hold for DQBF.

# Preliminaries

**(Def.)** DQBF: $\forall x_1 \cdots \forall x_n \; \exists y_1(\bar{z}_1) \cdots \exists y_k(\bar{z}_k) \quad \varphi$

where each $\bar{z}_i \subseteq \{x_1, \ldots, x_n\}$.

We call it $k$-DQBF.

**(Def.)** It is *satisfiable* if there is $(f_1, \ldots, f_k)$ where each $f_i$ is a boolean function $f_i : \{0,1\}^{n_i} \to \{0,1\}$ such that $\varphi$ is a tautology when $y_i$ is replaced with $f_i(\bar{z}_i)$, and $n_i$ is the length of $\bar{z}_i$.

**(Def.)** The number of solutions := The number of different $(f_1, \ldots, f_k)$.

**(Def.)** sat(DQBF): On input DQBF, decide if it is satisfiable.

**(Def.)** sat($k$-DQBF): Restricted to $k$-DQBF.

# Observation: $k$-DQBF = $k$-CNF (in succinct form)

**(Expansion)** $\forall x_1 \cdots \forall x_n \; \exists y_1(\bar{z}_1) \cdots \exists y_k(\bar{z}_k) \; \varphi$ is equivalent to:

$$\bigwedge_{a_1 \cdots a_n \in \{0,1\}^n} \varphi \; [x_1/a_1, \ldots, x_n/a_n, \; y_1/f_1(\bar{c}_1), \ldots, y_k/f_k(\bar{c}_k)]$$

where each $\bar{c}_i$ is $a_1 \cdots a_n|_{\bar{z}_i}$

Each $f_i(\bar{c}_i)$ is treated as a boolean variable.

For each $a_1 \cdots a_n$, the formula:

$$\varphi \; [x_1/a_1, \ldots, x_n/a_n, y_1/f_1(\bar{c}_1), \ldots, y_k/f_k(\bar{c}_k)]$$

is a formula with $k$ variables.

It can be rewritten as $k$-CNF formula, e.g., by building the truth table.

A clause for each row (in the truth table) with 0 value.

# Observation: $k$-DQBF = $k$-CNF (in succinct form) – Cont'd

A DQBF:

$$\forall x_1 \cdots \forall x_n \ \exists y_1(\bar{z}_1) \cdots \exists y_k(\bar{z}_k) \quad \varphi$$

represents a $k$-CNF formula:

$$\bigwedge_{a_1 \cdots a_n b_1 \cdots b_k \in \{0,1\}^{n+k} \text{ s.t. } \varphi(a_1 \cdots a_n b_1 \cdots b_k)=0} C_{\bar{a},\bar{b}}$$

where $C_{\bar{a},\bar{b}}$ is a clause with variables $f_1(\bar{c}_1), \ldots, f_k(\bar{c}_k)$:

- If $b_i = 0$, then $f_i(\bar{c}_i)$ is in $C_{\bar{a},\bar{b}}$.
- If $b_i = 1$, then $\neg f_i(\bar{c}_i)$ is in $C_{\bar{a},\bar{b}}$.

**(Question)** Are there more resemblances between sat($k$-DQBF) and $k$-SAT?

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# sat(1-DQBF) is coNP-complete

**(Fact)** 1-CNF formula $\ell_1 \wedge \ell_2 \wedge \cdots \wedge \ell_m$ is not satisfiable iff $\ell_i = \neg\ell_j$, for some $i, j$.

A 1-DQBF: $\forall x_1 \cdots \forall x_n \exists y(\bar{z})\ \varphi$ represents a 1-CNF formula:

$$\bigwedge_{a_1\cdots a_n b \in \{0,1\}^{n+1} \text{ s.t. } \varphi(a_1\cdots a_n b)=0} C_{a_1\cdots a_n b}$$

**(coNP-membership)** The NP algorithm for non-satisfiability:

On input $\forall x_1 \cdots \forall x_n \exists y(\bar{z})\ \varphi$:

- Guess two assignments $(a_1, \ldots, a_n, 0)$ and $(a'_1, \cdots, a'_n, 1)$ such that:

$$a_1 \cdots a_n|_{\bar{z}} \ = \ a'_1 \cdots a'_n|_{\bar{z}}$$

- Verify that they are both *non-satisfying* assignments of $\varphi$.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# sat(2-DQBF) is PSPACE-complete

**(Membership)** Use the same idea that 2-SAT is NL-complete.

2-CNF formula with variables $u_1, \ldots, u_n$:

$$\bigwedge_{1 \leqslant i \leqslant m} (\ell_{i,1} \to \ell_{i,2}) \qquad \text{where each } \ell_{i,1}, \ell_{i,2} \text{ are literals}$$

It is a graph with nodes $u_1, \ldots, u_n, \neg u_1, \ldots, \neg u_n$.

It is not satisfiable iff there is a path from a variable $u$ to $\neg u$ and vice versa.

**(Algo for sat(2-DQBF))** On 2-DQBF $\forall x_1 \cdots \forall x_n \exists y_1(\bar{z}_1) \exists y_2(\bar{z}_2)\ \varphi$:

- Guess a variable $f_i(\bar{c})$.
- Guess a path from $f_i(\bar{c})$ to $\neg f_i(\bar{c})$ and vice versa.
  The edges correspond to the clauses $C_{a_1 \cdots a_n b_1 b_2}$ with $\varphi(a_1 \cdots a_n b_1 b_2) = 0$.

# sat(2-DQBF) is PSPACE-complete — Cont'd

**(Hardness)** Reduction from 2-colorability in succinct representation.

**(Def.)** A (boolean) circuit $C(\bar{x}_1, \bar{x}_2)$ represents a graph $G(C)$, where $\bar{x}_1, \bar{x}_2$ are vectors of $n$ boolean variables:

- The set of vertices is $\{0,1\}^n$.
- $(u, v)$ is an edge iff $C(u, v) = 1$.

**Succinct 2-colorability:** On input circuit $C$, decide if $G(C)$ is 2-colorable.

**Theorem (Papadimitriou and Yannakakis, 1986)**
*Succinct 2-colorability is PSPACE-complete.*

**(Main idea)** When the set of vertices is $\{0,1\}^n$, view a coloring as a function $f\colon \{0,1\}^n \to \{0,1\}$.

**(The reduction)** On input circuit $C(\bar{x}_1, \bar{x}_2)$, output the following DQBF:

$$\forall \bar{x}_1 \forall \bar{x}_2 \; \exists y_1(\bar{x}_1) \exists y_2(\bar{x}_2) \quad \bar{x}_1 = \bar{x}_2 \;\to\; y_1 = y_2$$
$$\wedge \quad C(\bar{x}_1, \bar{x}_2) \to y_1 \neq y_2$$

**Theorem**

*sat(2-DQBF) is PSPACE-complete.*

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF)

$$\forall x_1 \cdots \forall x_n \ \exists y_1(\bar{z}_1) \cdots \exists y_k(\bar{z}_k) \quad \varphi$$

Combine $f_1, \cdots, f_k$ into a function $f : \{0,1\}^{n+k} \to \{0,1\}$ such that

$$f(\bar{a}, \bar{b}) = 1 \text{ if and only if } b_i = f_i(\bar{a}|_{\bar{z}_i})$$

However, we can't express for any $\bar{a}$, there is exactly one $\bar{b}$ such that $f(\bar{a}, \bar{b}) = 1$ with DQBF

Construct the monotonic encoding $g : \{0,1\}^{n+k} \to \{0,1\}$ of $f_1, \cdots, f_k$ such that

- For every $\bar{a} \in \{0,1\}^n$, $g(\bar{a}, \cdot)$ is monotonic
- If $f(\bar{a}, \bar{b}) = 1$, then $g(\bar{a}, \bar{c}) = \begin{cases} 0 & \text{if } \bar{c} <_{lex} \bar{b} \\ 1 & \text{if } \bar{b} \leqslant_{lex} \bar{c} \end{cases}$

| $x_1$ | $\cdots$ | $x_n$ | $v_1$ | $\cdots$ | $v_k$ | f | g |
|---|---|---|---|---|---|---|---|
| 0 | $\cdots$ | 0 | 0 | $\cdots$ | 0 | 0 | 0 |
| ⋮ | | ⋮ | ⋮ | | ⋮ | ⋮ | ⋮ |
| ⋮ | | ⋮ | | | | 0 | 0 |
| 0 | $\cdots$ | 0 | $u_1$ | $\cdots$ | $u_k$ | 1 | 1 |
| | | | | | | 0 | 1 |
| ⋮ | | ⋮ | ⋮ | | ⋮ | ⋮ | ⋮ |
| 0 | $\cdots$ | 0 | 1 | $\cdots$ | 1 | 0 | 1 |

# Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF)

$$\forall x_1 \cdots \forall x_n \ \exists y_1(\bar{z}_1) \cdots \exists y_k(\bar{z}_k) \quad \varphi$$

Combine $f_1, \cdots, f_k$ into a function $f : \{0,1\}^{n+k} \to \{0,1\}$ such that

$$f(\bar{a}, \bar{b}) = 1 \text{ if and only if } b_i = f_i(\bar{a}|_{\bar{z}_i})$$

However, we can't express for any $\bar{a}$, there is exactly one $\bar{b}$ such that $f(\bar{a}, \bar{b}) = 1$ with DQBF

Construct the monotonic encoding $g : \{0,1\}^{n+k} \to \{0,1\}$ of $f_1, \cdots, f_k$ such that

- For every $\bar{a} \in \{0,1\}^n$, $g(\bar{a}, \cdot)$ is monotonic
- If $f(\bar{a}, \bar{b}) = 1$, then $g(\bar{a}, \bar{c}) = \begin{cases} 0 & \text{if } \bar{c} <_{lex} \bar{b} \\ 1 & \text{if } \bar{b} \leqslant_{lex} \bar{c} \end{cases}$

**(The reduction)** Output the following DQBF:

$$\forall x_1 \cdots \forall x_n \forall v_1 \cdots \forall v_k \quad \forall x_1' \cdots \forall x_n' \forall v_1' \cdots \forall v_k' \quad \forall x_1'' \cdots \forall x_n'' \forall v_1'' \cdots \forall v_k''$$

$$\exists y_1(x_i \text{ and } v_i) \quad \exists y_2(x_i' \text{ and } v_i') \quad \exists y_3(x_i'' \text{ and } v_i'')$$

$$y_1 = y_2 = g \quad \wedge \quad y_3 = f$$

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The notion of projections

**Theorem (Papadimitriou, Yannakakis 1986)**
*If there is a projection from SAT to a graph problem $\mathcal{P}$, then the succinct version of $\mathcal{P}$ is NEXP-hard.*

**(Recall)** A (boolean) circuit $C(\bar{x}_1, \bar{x}_2)$ represents a graph $G(C)$, where $\bar{x}_1, \bar{x}_2$ are vectors of $n$ boolean variables:

- The set of vertices is $\{0,1\}^n$.
- $(u, v)$ is an edge iff $C(u, v) = 1$.

**(Def.)** The succinct version of a graph problem $\mathcal{P}$: The input is a circuit representing a graph.

# The notion of projections – continued

**(Recall)** A polynomial time (Karp) reduction is a function $F : \{0,1\}^* \to \{0,1\}^*$ computable in polynomial time such that for every $w \in \{0,1\}^*$, $F(w)$ is of length $p(|w|)$ for some polynomial $p$.

**(Def.)** $F$ is a *projection*, if there is a polynomial time algorithm $\mathcal{A}$:

**Input:** $1^n$ and an index $j$ where $1 \leqslant j \leqslant p(n)$.

**Output:** $0$, $1$, $X_i$, or $1 - X_i$, where $1 \leqslant i \leqslant n$ such that:

If $z_j = \mathcal{A}(1^n, j)$ for each $1 \leqslant j \leqslant p(n)$, then for every $w_1 \cdots w_n \in \{0,1\}^n$:

$$F(w_1 \cdots w_n) \;=\; z_1 \cdots z_{p(n)}\Big|_{X_1/w_1,\ldots,X_n/w_n}$$

# Lifting projections in NP to reductions in NEXP

**Theorem (Papadimitriou, Yannakakis 1986)**
*If there is a projection from SAT to a graph problem $\mathcal{P}$, then the succinct version of $\mathcal{P}$ is NEXP-hard.*

The projection can be turned into a polynomial time reduction from an NEXP-complete problem to succinct $\mathcal{P}$.

We observe that the projection can be turned into a reduction from sat(DQBF) to succinct $\mathcal{P}$.

**Corollary**
*If there is a projection from SAT to a graph problem $\mathcal{P}$, then there is a polynomial time reduction from sat(DQBF) to succinct $\mathcal{P}$.*

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# The main results

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

# On the class NTIME$[t(n)]$

**Theorem (Chen, et. al., 2022)**

*For every $L \in NTIME[t(n)]$, there is a reduction from $L$ to sat($DQBF$) using $O(\log t(n))$ universal variables and $O(1)$ existential variables.*
*The runtime of the reduction is $O(\max\{n, poly(\log t(n))\})$.*

The constant $O(1)$ depends on $L$.

Using our previous reduction, we obtain the following corollary

**Corollary**

*For every $L \in NTIME[t(n)]$, there is a reduction from $L$ to sat(3-$DQBF$) using $O(\log t(n))$ universal variables.*
*The runtime of the reduction is $O(\max\{n, poly(\log t(n))\})$.*

# On the class NSPACE$[s(n)]$

**Theorem**

*For every $L \in NSPACE[s(n)]$, there is a deterministic algorithm $\mathcal{A}$ with run time $O(s(n)^2)$ such that:*
*On input $w$, it outputs a 2-DQBF $\Psi$ with $O(s(|w|))$ universal variables such that $w \in L$ if and only if $\Psi$ is not satisfiable.*

**(Intuitive proof)** Suppose $\mathcal{M}$ decides $L$ in space $s(n)$. Reduce it to 2-CNF formula (of size exponential in $s(n)$).

On input word $w$, construct the formula $F_w$ that states the following:

- The variables are $X_C$, where the index $C$ ranges over all the configurations of $\mathcal{M}$ on $w$.
- For every two configurations $C_1$ and $C_2$ where $C_2$ is the next configuration of $C_1$, we have an implication $X_{C_1} \to X_{C_2}$.
- For the initial configuration $C_0$, we have the implication $\neg X_{C_0} \to X_{C_0}$.
- For the initial configuration $C_0$ and the accepting configuration $C_{acc}$, we have the implication $X_{C_{acc}} \to \neg X_{C_0}$.

$\mathcal{M}$ accepts $w$ if and only if $F_w$ is not satisfiable.

# On the class NSPACE$[s(n)]$ – cont'd

**(Modifying it to 2-DQBF)** Encode each configuration $C$ as 0-1 strings of length $O(s(n))$.

Represent each variable $X_C$ as variable $f(C)$.

The desired 2-DQBF is:

$$\forall \bar{x}_1 \forall \bar{x}_2 \; \exists y_1(\bar{x}_1) \exists y_2(\bar{x}_2) \quad (\bar{x}_1 = \bar{x}_2 \rightarrow y_1 = y_2) \; \wedge \; \varphi$$

where $\varphi$ states the following:

(b) If $\bar{x}_2$ is the next configuration of $\bar{x}_1$, then $y_1 \rightarrow y_2$.

(c) If $\bar{x}_1$ and $\bar{x}_2$ encode the initial configuration, then $\neg y_1 \rightarrow y_2$.

(d) If $\bar{x}_1$ encodes the initial configuration and $\bar{x}_2$ encodes the accepting configuration, then $y_2 \rightarrow \neg y_1$.

# Recap

| $k$ | sat($k$-DQBF) | $k$-SAT |
|---|---|---|
| 1 | coNP-complete | trivial |
| 2 | PSPACE-complete | NL-complete |
| 3 | NEXP-complete | NP-complete |

- Parsimonious polynomial time reduction from sat(DQBF) to sat(3-DQBF).
- Lifting polynomial time (Karp) reductions from SAT to languages in NP to reductions from DQBF to languages in NEXP.
- Reductions from languages in NTIME[$t(n)$] to 3-DQBF as well as from languages in NSPACE[$s(n)$] to 2-DQBF.

## Concluding remarks

Our hope:

- Ideas used to develop SAT solvers can be used for DQBF and vice versa.
- Richer benchmarks and applications of DQBF solvers.
- DQBF can be *the* problem in NEXP, just like SAT in NP and QBF in PSPACE.

Thank you very much!