Read-once branching programs as proof lines

Dmitry Itsykson

Ben Gurion University of the Negev

Workshop on Proof Complexity and Meta-Mathematics
March 23, 2023

Semantic proof systems

- Let $\varphi = \bigwedge_{i \in I} C_i$ be an unsatisfiable CNF formula.
- Proof lines: Boolean predicates represented somehow:
  - **Resolution**: clauses $(x \vee y \vee \neg z)$
  - **Cutting planes**: linear inequalities with integer coefficients $x - 2y + z \geq 2$
  - **Th(k)**: degree k inequalities with integer coefficients $2xy - yzt + x \geq 3$
  - **Res($\oplus$)**: disjunctions of linear equalities over $\mathbb{F}_2$ $(x + y = 1) \vee (x + z + t = 0) \vee (z = 1)$
- Semantic rule: $\frac{D_1, D_2}{D_3}$ if $D_1, D_2 \models D_3$.
- Semantic refutation of φ: $D_1, D_2, \ldots, D_s$ such that
  - $D_s \equiv 0$
  - D_i either represents a clause of φ or $\frac{D_j, D_k}{D_i}$, where $j, k \leq i$.
- Length: s. Size: $\sum_{i=1}^{s} |D_i|$.

Semantic proof systems

- Let $\varphi = \bigwedge_{i \in I} C_i$ be an unsatisfiable CNF formula.
- Proof lines: Boolean predicates represented somehow:
  - **Resolution**: clauses $(x \vee y \vee \neg z)$
  - **Cutting planes**: linear inequalities with integer coefficients $x - 2y + z \geq 2$
  - **Th(k)**: degree k inequalities with integer coefficients $2xy - yzt + x \geq 3$
  - **Res(⊕)**: disjunctions of linear equalities over $\mathbb{F}_2$
    $(x + y = 1) \vee (x + z + t = 0) \vee (z = 1)$
- Semantic rule: $\frac{D_1, D_2}{D_3}$ if $D_1, D_2 \models D_3$.
- Semantic refutation of φ: $D_1, D_2, \ldots, D_s$ such that
  - $D_s \equiv 0$
  - D_i either represents a clause of φ or $\frac{D_j, D_k}{D_i}$, where $j, k \leq i$.
- Length: s. Size: $\sum_{i=1}^{s} |D_i|$.

Semantic proof systems

- Let $\varphi = \bigwedge_{i \in I} C_i$ be an unsatisfiable CNF formula.
- Proof lines: Boolean predicates represented somehow:
  - **Resolution**: clauses $(x \vee y \vee \neg z)$
  - **Cutting planes**: linear inequalities with integer coefficients $x - 2y + z \geq 2$
  - **Th(k)**: degree k inequalities with integer coefficients $2xy - yzt + x \geq 3$
  - **Res(⊕)**: disjunctions of linear equalities over $\mathbb{F}_2$ $(x + y = 1) \vee (x + z + t = 0) \vee (z = 1)$
- Semantic rule: $\frac{D_1, D_2}{D_3}$ if $D_1, D_2 \models D_3$.
- Semantic refutation of φ: $D_1, D_2, \ldots, D_s$ such that
  - $D_s \equiv 0$
  - D_i either represents a clause of φ or $\frac{D_j, D_k}{D_i}$, where $j, k \leq i$.
- Length: s. Size: $\sum_{i=1}^{s} |D_i|$.

On lower bounds for semantic proof systems

- If proof lines are too strong, there are upper bounds for all formulas:
  - **CNF formulas**: every UNSAT CNF has a short refutation.
  - **Semantic PC over reals**: every UNSAT 3CNF has a short refutation.
    - $(x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee t) \wedge \dots$
    - $xy(1-z) + x(1-y)t + \dots = 0$, $(x^2-x)^2 + (y^2-y)^2 + \dots = 0$
- [Krajícek, 1995] If proof lines have small deterministic communication complexity, then CliqueColoring is hard.
  - Resolution, CP*
- [Beame, Pitassi, Segerlind, 2007] If proof lines have small randomized communication complexity, then lifted Tseitin formulas are hard for tree-like refutations.
  - Tree-like Th(k), tree-like Res($\oplus$).

On lower bounds for semantic proof systems

- If proof lines are too strong, there are upper bounds for all formulas:
  - **CNF formulas**: every UNSAT CNF has a short refutation.
  - **Semantic PC over reals**: every UNSAT 3CNF has a short refutation.
    - $(x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee t) \wedge \dots$
    - $xy(1-z) + x(1-y)t + \dots = 0$, $(x^2 - x)^2 + (y^2 - y)^2 + \dots = 0$
- [Krajícek, 1995] If proof lines have small deterministic communication complexity, then CliqueColoring is hard.
  - Resolution, CP^*
- [Beame, Pitassi, Segerlind, 2007] If proof lines have small randomized communication complexity, then lifted Tseitin formulas are hard for tree-like refutations.
  - Tree-like Th(k), tree-like Res($\oplus$).

On lower bounds for semantic proof systems

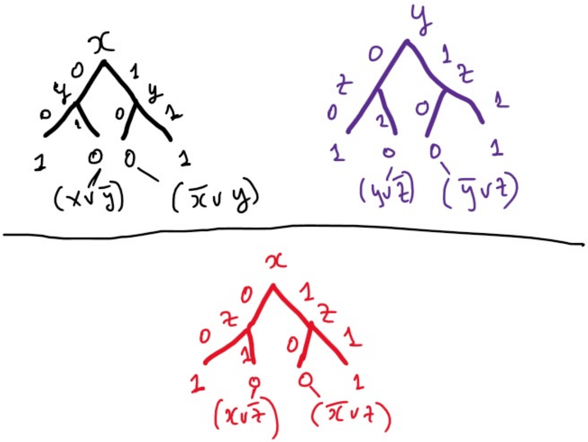
- If proof lines are too strong, there are upper bounds for all formulas:
  - **CNF formulas**: every UNSAT CNF has a short refutation.
  - **Semantic PC over reals**: every UNSAT 3CNF has a short refutation.
    - $(x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee t) \wedge \dots$
    - $xy(1-z) + x(1-y)t + \cdots = 0$, $(x^2-x)^2 + (y^2-y)^2 + \cdots = 0$
- [Krajícek, 1995] If proof lines have small deterministic communication complexity, then CliqueColoring is hard.
  - Resolution, CP^*
- [Beame, Pitassi, Segerlind, 2007] If proof lines have small randomized communication complexity, then lifted Tseitin formulas are hard for tree-like refutations.
  - Tree-like Th(k), tree-like Res($\oplus$).

On lower bounds for semantic proof systems

- If proof lines are too strong, there are upper bounds for all formulas:
  - **CNF formulas**: every UNSAT CNF has a short refutation.
  - **Semantic PC over reals**: every UNSAT 3CNF has a short refutation.
    - $(x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee t) \wedge \dots$
    - $xy(1-z) + x(1-y)t + \dots = 0$, $(x^2 - x)^2 + (y^2 - y)^2 + \dots = 0$
- [Krajícek, 1995] If proof lines have small deterministic communication complexity, then CliqueColoring is hard.
  - Resolution, CP^*
- [Beame, Pitassi, Segerlind, 2007] If proof lines have small randomized communication complexity, then lifted Tseitin formulas are hard for tree-like refutations.
  - Tree-like Th(k), tree-like Res($\oplus$).

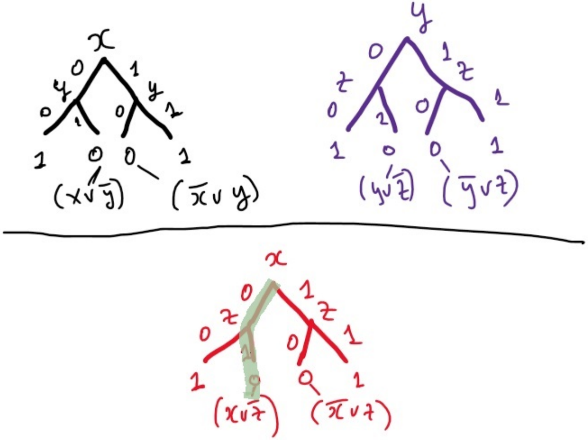
Reasoning by decision trees

Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.



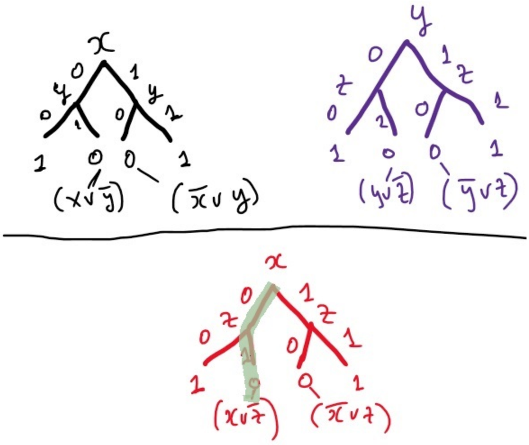
Reasoning by decision trees

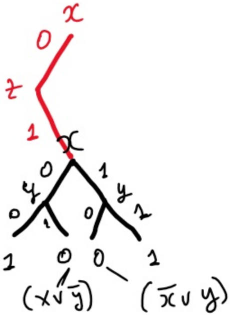
Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.



Reasoning by decision trees

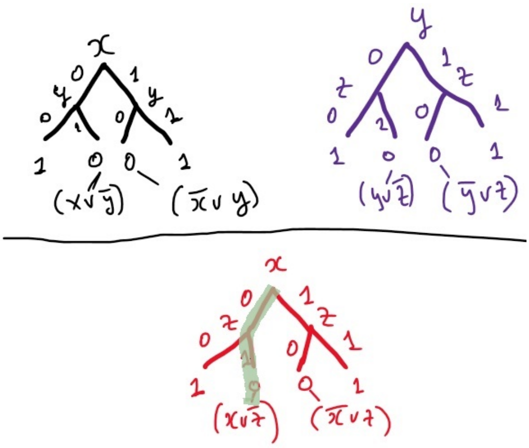
Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.

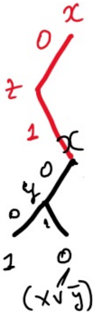




Reasoning by decision trees

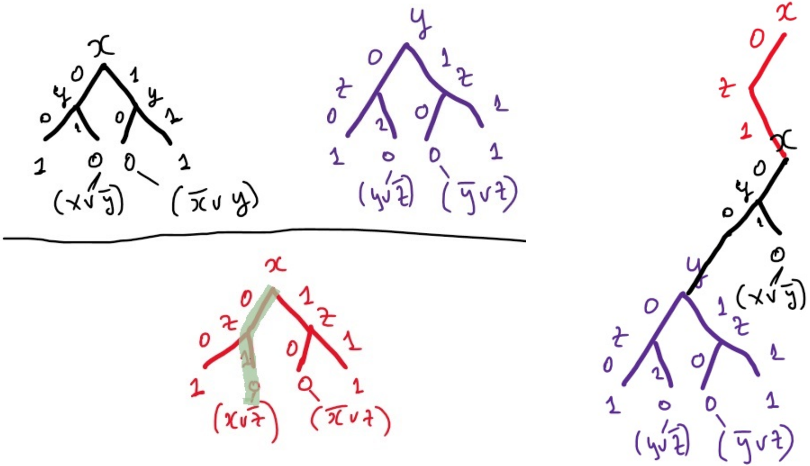
Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.





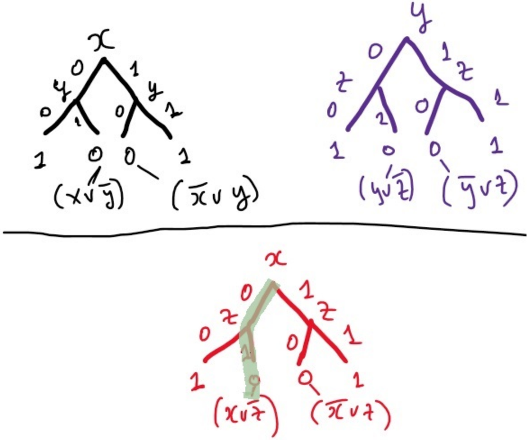
Reasoning by decision trees

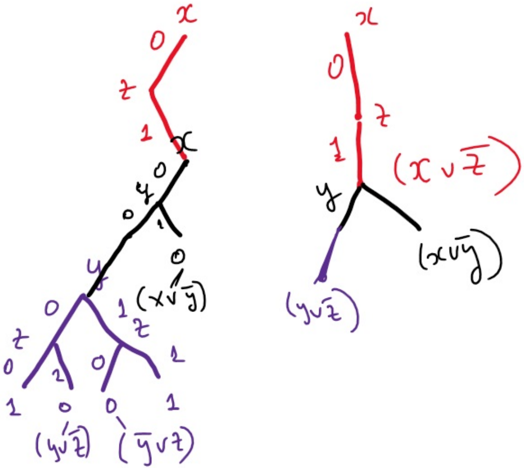
Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.



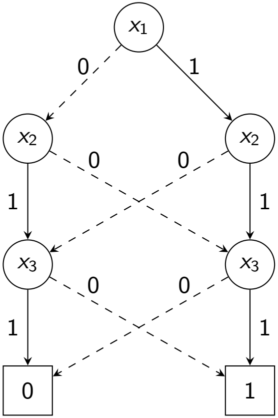
Reasoning by decision trees

Prop. Semantic calculus of decision trees is polynomially equivalent to Resolution.



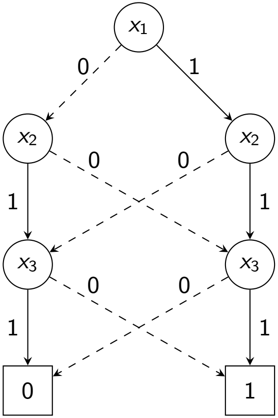


Branching programs



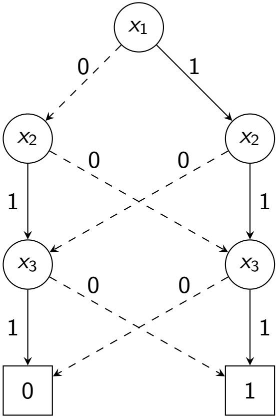
- 1-BP: every path contains different variables.
- OBDD: in all paths variables appear in the same order
- There are small OBDD-representations of clauses, parities and linear inequalities with small coefficients.
- Binary operations for OBDDs in the same order can be computed in polynomial time.
- If partition agrees with the order, then communication complexity of an OBDD of size S is at most $\lceil \log S \rceil + 1$.

Branching programs



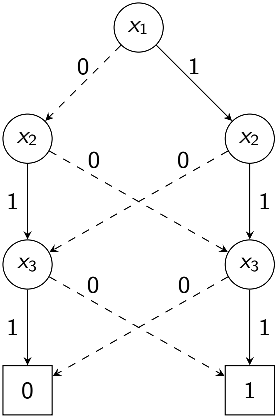
- 1-BP: every path contains different variables.
- OBDD: in all paths variables appear in the same order
- There are small OBDD-representations of clauses, parities and linear inequalities with small coefficients.
- Binary operations for OBDDs in the same order can be computed in polynomial time.
- If partition agrees with the order, then communication complexity of an OBDD of size S is at most $\lceil \log S \rceil + 1$.

Branching programs



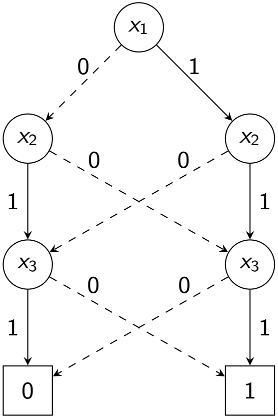
- 1-BP: every path contains different variables.
- OBDD: in all paths variables appear in the same order
- There are small OBDD-representations of clauses, parities and linear inequalities with small coefficients.
- Binary operations for OBDDs in the same order can be computed in polynomial time.
- If partition agrees with the order, then communication complexity of an OBDD of size S is at most $\lceil \log S \rceil + 1$.

Branching programs



- 1-BP: every path contains different variables.
- OBDD: in all paths variables appear in the same order
- There are small OBDD-representations of clauses, parities and linear inequalities with small coefficients.
- Binary operations for OBDDs in the same order can be computed in polynomial time.
- If partition agrees with the order, then communication complexity of an OBDD of size S is at most $\lceil \log S \rceil + 1$.

Branching programs



- 1-BP: every path contains different variables.
- OBDD: in all paths variables appear in the same order
- There are small OBDD-representations of clauses, parities and linear inequalities with small coefficients.
- Binary operations for OBDDs in the same order can be computed in polynomial time.
- If partition agrees with the order, then communication complexity of an OBDD of size S is at most $\lceil \log S \rceil + 1$.

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule ($\wedge$): $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule ($\exists$): $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi_1}}{D_2^{\pi_2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule ($\wedge$): $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule ($\exists$): $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi_1}}{D_2^{\pi_2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule ($\wedge$): $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule ($\exists$): $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi_1}}{D_2^{\pi_2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: OBDD($\wedge$), OBDD($\wedge$, w),...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule $(\wedge)$: $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule $(\exists)$: $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi_1}}{D_2^{\pi_2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule $(\wedge)$: $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule $(\exists)$: $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi_1}}{D_2^{\pi_2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule $(\wedge)$: $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule $(\exists)$: $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi 1}}{D_2^{\pi 2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD-proofs

- [Atserias, Kolaitis, Vardi, 2004] OBDD-based proof systems.
- $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_t$ is unsatisfiable CNF.
- Choose order π; every C_i is represented as π-ordered OBDD.
- Rules:
  - Conjunction rule ($\wedge$): $\frac{D_1^\pi, D_2^\pi}{(D_1 \wedge D_2)^\pi}$
  - Weakening rule (w): $\frac{D^\pi}{D_1^\pi}$ if $D \models D_1$.
  - Projection rule ($\exists$): $\frac{D^\pi}{\exists x D^\pi}$
    - Partial case of weakening rule
  - Reordering rule (r): $\frac{D_1^{\pi 1}}{D_2^{\pi 2}}$ if $D_1^{\pi_1} \equiv D_2^{\pi_2}$
- Goal: to derive a constant false OBDD.
- Particular system has its set of rules: $\mathrm{OBDD}(\wedge)$, $\mathrm{OBDD}(\wedge, \mathrm{w})$,...

OBDD($\wedge, \exists$)-proofs

- OBDD($\wedge, \exists$)-proofs
- [Atserias, Kolaitis, Vardi, 2004]
  - Short proofs of unsatisfiable linear systems over $\mathbb{F}_2$:
    $$\exists x \begin{cases} x + y + z = 1 \\ x + t + f = 0 \end{cases} \iff y + z + t + f = 1.$$
  - OBDD($\wedge, \exists$) simulates and strictly stronger than resolution:
    $$\exists x \begin{cases} x \vee C \\ \neg x \vee D \end{cases} \iff C \vee D.$$
  - [Chen, Zhang 2009] Short proof of the pigeonhole principle
- **Open question**: whether $\mathrm{OBDD}(\wedge, \exists)$ simulates CP^*?

OBDD($\wedge, \exists$)-proofs

- OBDD($\wedge, \exists$)-proofs
- [Atserias, Kolaitis, Vardi, 2004]
  - Short proofs of unsatisfiable linear systems over $\mathbb{F}_2$:
    $$\exists x \begin{cases} x + y + z = 1 \\ x + t + f = 0 \end{cases} \iff y + z + t + f = 1.$$
  - OBDD($\wedge, \exists$) simulates and strictly stronger than resolution:
    $$\exists x \begin{cases} x \vee C \\ \neg x \vee D \end{cases} \iff C \vee D.$$
  - [Chen, Zhang 2009] Short proof of the pigeonhole principle
- **Open question**: whether OBDD($\wedge, \exists$) simulates CP*?

OBDD($\wedge, \exists$)-proofs

- OBDD($\wedge, \exists$)-proofs
- [Atserias, Kolaitis, Vardi, 2004]
  - Short proofs of unsatisfiable linear systems over $\mathbb{F}_2$:
    $$\exists x \begin{cases} x + y + z = 1 \\ x + t + f = 0 \end{cases} \iff y + z + t + f = 1.$$
  - OBDD($\wedge, \exists$) simulates and strictly stronger than resolution:
    $$\exists x \begin{cases} x \vee C \\ \neg x \vee D \end{cases} \iff C \vee D.$$
  - [Chen, Zhang 2009] Short proof of the pigeonhole principle
- **Open question**: whether $\mathrm{OBDD}(\wedge, \exists)$ simulates CP^*?

OBDD($\wedge$, weakening)-proofs

- [Atserias, Kolaitis, Vardi, 2004] $\mathrm{OBDD}(\wedge, \mathrm{w})$ simulates CP*
- [Buss, I., Knop, Sokolov, 2018] $\mathrm{OBDD}(\wedge, \mathrm{w})$ has short proofs of Clique-Coloring principle.
- [Atserias, Kolaitis, Vardi, 2004] There is an order π s.t. all $\pi - \mathrm{OBDD}(\wedge, \mathrm{w})$ proofs of Clique-Coloring are of exp. size.
- [Krajicek, 2008] $2^{n^{\Omega(1)}}$-lower bound for dag-like $\mathrm{OBDD}(\wedge, \mathrm{w})$-proofs:
  - $\varphi(x)$ is a formula hard for one order π;
  - $\mathcal{K}(\varphi) = (\sigma \text{ encodes a permutation}) \wedge \varphi(\sigma(x))$;
- [Segerlind, 2008]
  - Orification: $\varphi(x_1, \ldots, x_n) \mapsto \varphi^{\vee_m} = \varphi(\bigvee_{i=1}^m y_{1,i}, \ldots, \bigvee_{i=1}^m y_{n,i})$.
  - $\mathcal{S}(\varphi) = \bigwedge_{\sigma \in \Pi} ((z \text{ encodes } \sigma) \to \varphi^{\vee_m}(\sigma(y)))$, where Π is a small family of 2-independent permutations.
  - $\mathrm{OBDD}(\wedge, \mathrm{w})$ does not simulate $\mathrm{Res}(O(\log n))$.
- [Buss, I., Knop, Sokolov, 2018] Reordering rule makes proof systems stronger.
  - $\mathcal{S}$(Clique-Coloring) separates $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$.

OBDD($\wedge$, weakening)-proofs

- [Atserias, Kolaitis, Vardi, 2004] $\mathrm{OBDD}(\wedge, \mathrm{w})$ simulates CP*
- [Buss, I., Knop, Sokolov, 2018] $\mathrm{OBDD}(\wedge, \mathrm{w})$ has short proofs of Clique-Coloring principle.
- [Atserias, Kolaitis, Vardi, 2004] There is an order π s.t. all $\pi - \mathrm{OBDD}(\wedge, \mathrm{w})$ proofs of Clique-Coloring are of exp. size.
- [Krajicek, 2008] $2^{n^{\Omega(1)}}$-lower bound for dag-like $\mathrm{OBDD}(\wedge, \mathrm{w})$-proofs:
  - $\varphi(x)$ is a formula hard for one order π;
  - $\mathcal{K}(\varphi) = (\sigma$ encodes a permutation$) \wedge \varphi(\sigma(x))$;
- [Segerlind, 2008]
  - Orification: $\varphi(x_1, \dots, x_n) \mapsto \varphi^{\vee_m} = \varphi(\bigvee_{i=1}^m y_{1,i}, \dots, \bigvee_{i=1}^m y_{n,i})$.
  - $\mathcal{S}(\varphi) = \bigwedge_{\sigma \in \Pi} ((z \text{ encodes } \sigma) \to \varphi^{\vee_m}(\sigma(y)))$, where Π is a small family of 2-independent permutations.
  - $\mathrm{OBDD}(\wedge, \mathrm{w})$ does not simulate $\mathrm{Res}(O(\log n))$.
- [Buss, I., Knop, Sokolov, 2018] Reordering rule makes proof systems stronger.
  - $\mathcal{S}$(Clique-Coloring) separates $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$.

OBDD($\wedge$, weakening)-proofs

- [Atserias, Kolaitis, Vardi, 2004] $\mathrm{OBDD}(\wedge, \mathrm{w})$ simulates CP*
- [Buss, I., Knop, Sokolov, 2018] $\mathrm{OBDD}(\wedge, \mathrm{w})$ has short proofs of Clique-Coloring principle.
- [Atserias, Kolaitis, Vardi, 2004] There is an order π s.t. all $\pi - \mathrm{OBDD}(\wedge, \mathrm{w})$ proofs of Clique-Coloring are of exp. size.
- [Krajicek, 2008] $2^{n^{\Omega(1)}}$-lower bound for dag-like $\mathrm{OBDD}(\wedge, \mathrm{w})$-proofs:
  - $\varphi(x)$ is a formula hard for one order π;
  - $\mathcal{K}(\varphi) = (\sigma \text{ encodes a permutation}) \wedge \varphi(\sigma(x))$;
- [Segerlind, 2008]
  - Orification: $\varphi(x_1, \ldots, x_n) \mapsto \varphi^{\vee_m} = \varphi(\bigvee_{i=1}^m y_{1,i}, \ldots, \bigvee_{i=1}^m y_{n,i})$.
  - $\mathcal{S}(\varphi) = \bigwedge_{\sigma \in \Pi} ((z \text{ encodes } \sigma) \to \varphi^{\vee_m}(\sigma(y)))$, where Π is a small family of 2-independent permutations.
  - $\mathrm{OBDD}(\wedge, \mathrm{w})$ does not simulate $\mathrm{Res}(O(\log n))$.
- [Buss, I., Knop, Sokolov, 2018] Reordering rule makes proof systems stronger.
  - $\mathcal{S}$(Clique-Coloring) separates $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$.

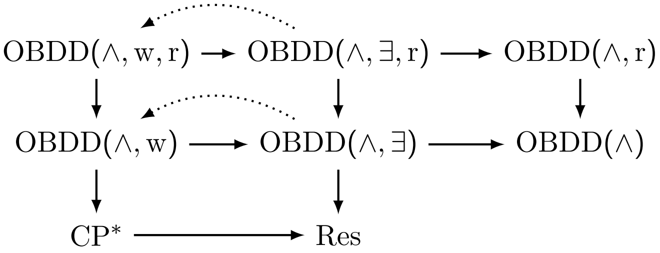
OBDD($\wedge$, weakening)-proofs

- [Atserias, Kolaitis, Vardi, 2004] $\mathrm{OBDD}(\wedge, \mathrm{w})$ simulates CP*
- [Buss, I., Knop, Sokolov, 2018] $\mathrm{OBDD}(\wedge, \mathrm{w})$ has short proofs of Clique-Coloring principle.
- [Atserias, Kolaitis, Vardi, 2004] There is an order π s.t. all $\pi - \mathrm{OBDD}(\wedge, \mathrm{w})$ proofs of Clique-Coloring are of exp. size.
- [Krajicek, 2008] $2^{n^{\Omega(1)}}$-lower bound for dag-like $\mathrm{OBDD}(\wedge, \mathrm{w})$-proofs:
  - $\varphi(x)$ is a formula hard for one order π;
  - $\mathcal{K}(\varphi) = (\sigma \text{ encodes a permutation}) \wedge \varphi(\sigma(x))$;
- [Segerlind, 2008]
  - Orification: $\varphi(x_1, \ldots, x_n) \mapsto \varphi^{\vee_m} = \varphi(\bigvee_{i=1}^m y_{1,i}, \ldots, \bigvee_{i=1}^m y_{n,i})$.
  - $\mathcal{S}(\varphi) = \bigwedge_{\sigma \in \Pi}((z \text{ encodes } \sigma) \to \varphi^{\vee_m}(\sigma(y)))$, where Π is a small family of 2-independent permutations.
  - $\mathrm{OBDD}(\wedge, \mathrm{w})$ does not simulate $\mathrm{Res}(O(\log n))$.
- [Buss, I., Knop, Sokolov, 2018] Reordering rule makes proof systems stronger.
  - $\mathcal{S}$(Clique-Coloring) separates $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$.

OBDD($\wedge$, weakening)-proofs

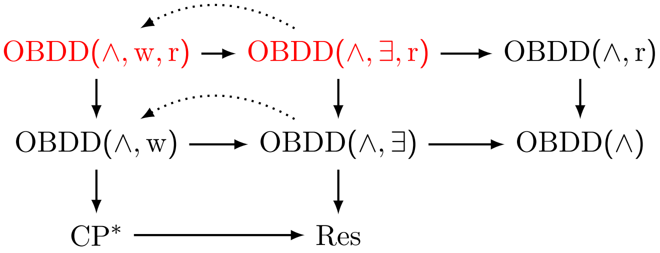
- [Atserias, Kolaitis, Vardi, 2004] $\mathrm{OBDD}(\wedge, \mathrm{w})$ simulates CP*
- [Buss, I., Knop, Sokolov, 2018] $\mathrm{OBDD}(\wedge, \mathrm{w})$ has short proofs of Clique-Coloring principle.
- [Atserias, Kolaitis, Vardi, 2004] There is an order π s.t. all $\pi - \mathrm{OBDD}(\wedge, \mathrm{w})$ proofs of Clique-Coloring are of exp. size.
- [Krajicek, 2008] $2^{n^{\Omega(1)}}$-lower bound for dag-like $\mathrm{OBDD}(\wedge, \mathrm{w})$-proofs:
  - $\varphi(x)$ is a formula hard for one order π;
  - $\mathcal{K}(\varphi) = (\sigma$ encodes a permutation$) \wedge \varphi(\sigma(x))$;
- [Segerlind, 2008]
  - Orification: $\varphi(x_1, \dots, x_n) \mapsto \varphi^{\vee_m} = \varphi(\bigvee_{i=1}^{m} y_{1,i}, \dots, \bigvee_{i=1}^{m} y_{n,i})$.
  - $\mathcal{S}(\varphi) = \bigwedge_{\sigma \in \Pi} ((z \text{ encodes } \sigma) \to \varphi^{\vee_m}(\sigma(y)))$, where Π is a small family of 2-independent permutations.
  - $\mathrm{OBDD}(\wedge, \mathrm{w})$ does not simulate $\mathrm{Res}(O(\log n))$.
- [Buss, I., Knop, Sokolov, 2018] Reordering rule makes proof systems stronger.
  - $\mathcal{S}$(Clique-Coloring) separates $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$.

OBDD picture



- If there is a path consisting of solid (straight) edges from Π_1 and Π_2, then Π_1 simulates Π_2.
- If there is a path from Π_1 to Π_2, but every such path contains a dotted (arched) edge, then it is **open**, whether Π_1 simulates Π_2.
- If there are no paths from Π_1 to Π_2 at all, then Π_1 does not simulate Π_2.

OBDD picture



- If there is a path consisting of solid (straight) edges from Π_1 and Π_2, then Π_1 simulates Π_2.
- If there is a path from Π_1 to Π_2, but every such path contains a dotted (arched) edge, then it is **open**, whether Π_1 simulates Π_2.
- If there are no paths from Π_1 to Π_2 at all, then Π_1 does not simulate Π_2.

Hardness of automation

Theorem. [I., Riazanov, 2022] There exists a polytime function $\mathcal{R}$ mapping CNF formulas to CNF formulas: for any 3-CNF ϕ with n variables

- if $\phi \in \mathrm{SAT}$, then $\mathcal{R}(\phi)$ has a resolution refutation of size at most n^{α};
- if $\phi \in \mathrm{UNSAT}$, then any $\mathrm{OBDD}(\wedge, \mathrm{w})$ refutation of $\mathcal{R}(\phi)$ has size $2^{\Omega(n)}$.

Corollary. It is NP-hard to automate $\mathrm{OBDD}(\wedge, \mathrm{w})$ and $\mathrm{OBDD}(\wedge, \exists)$.

Proof strategy:

1. Prove for one particular variable order.
   - Lifting from resolution blockwidth (Atserias, Muller 2019) to dag-like communication protocols with $o(n)$ participants in the number-in-the-hand model. Similar theorem for non-automatability of Cutting Planes and $n+1$ participants was proved by [Göös, Koroth, Mertz, Pitassi, 2020].
2. Apply Segerlind's transformation.

Hardness of automation

Theorem. [I., Riazanov, 2022] There exists a polytime function $\mathcal{R}$ mapping CNF formulas to CNF formulas: for any 3-CNF ϕ with n variables

- if $\phi \in \mathrm{SAT}$, then $\mathcal{R}(\phi)$ has a resolution refutation of size at most n^α;
- if $\phi \in \mathrm{UNSAT}$, then any $\mathrm{OBDD}(\wedge, \mathrm{w})$ refutation of $\mathcal{R}(\phi)$ has size $2^{\Omega(n)}$.

Corollary. It is NP-hard to automate $\mathrm{OBDD}(\wedge, \mathrm{w})$ and $\mathrm{OBDD}(\wedge, \exists)$.

Proof strategy:

1. Prove for one particular variable order.
    - Lifting from resolution blockwidth (Atserias, Muller 2019) to dag-like communication protocols with $o(n)$ participants in the number-in-the-hand model. Similar theorem for non-automatability of Cutting Planes and $n+1$ participants was proved by [Göös, Koroth, Mertz, Pitassi, 2020].
2. Apply Segerlind's transformation.

Hardness of automation

Theorem. [I., Riazanov, 2022] There exists a polytime function $\mathcal{R}$ mapping CNF formulas to CNF formulas: for any 3-CNF ϕ with n variables

- if $\phi \in \mathrm{SAT}$, then $\mathcal{R}(\phi)$ has a resolution refutation of size at most n^α;
- if $\phi \in \mathrm{UNSAT}$, then any $\mathrm{OBDD}(\wedge, \mathrm{w})$ refutation of $\mathcal{R}(\phi)$ has size $2^{\Omega(n)}$.

Corollary. It is NP-hard to automate $\mathrm{OBDD}(\wedge, \mathrm{w})$ and $\mathrm{OBDD}(\wedge, \exists)$.

Proof strategy:

1. Prove for one particular variable order.
   - Lifting from resolution blockwidth (Atserias, Muller 2019) to dag-like communication protocols with $o(n)$ participants in the number-in-the-hand model. Similar theorem for non-automatability of Cutting Planes and $n+1$ participants was proved by [Göös, Koroth, Mertz, Pitassi, 2020].
2. Apply Segerlind's transformation.

1-BP($\wedge$)

- 1-BP($\wedge$) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP($\wedge$):
  - $\mathrm{PM}(G)$: graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then $\mathrm{PM}(G)$ and Tseitin(G) require 1-BP($\wedge$) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP($\wedge$).
- Extension rule can not decrease the size of 1-NBP($\wedge$) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP($\wedge$) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).

1-BP($\wedge$)

- 1-BP($\wedge$) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP($\wedge$):
  - PM(G): graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then PM(G) and Tseitin(G) require 1-BP($\wedge$) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP($\wedge$).
- Extension rule can not decrease the size of 1-NBP($\wedge$) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP($\wedge$) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).

1-BP(∧)

- 1-BP(∧) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP(∧):
  - PM(G): graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then PM(G) and Tseitin(G) require 1-BP(∧) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP(∧).
- Extension rule can not decrease the size of 1-NBP(∧) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP(∧) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).

1-BP($\wedge$)

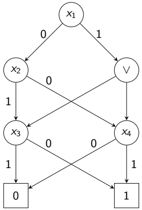
- 1-BP($\wedge$) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP($\wedge$):
  - PM(G): graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then PM(G) and Tseitin(G) require 1-BP($\wedge$) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP($\wedge$).
- Extension rule can not decrease the size of 1-NBP($\wedge$) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP($\wedge$) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).

1-BP($\wedge$)

- 1-BP($\wedge$) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP($\wedge$):
  - PM(G): graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then PM(G) and Tseitin(G) require 1-BP($\wedge$) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP($\wedge$).
- Extension rule can not decrease the size of 1-NBP($\wedge$) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP($\wedge$) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).

1-BP($\wedge$)

- 1-BP($\wedge$) has short refutations for formulas based on bipartite graphs: PHP, Tseitin formulas on bipartite graphs, etc.
- [Buss, I., Knop, Riazanov, Sokolov, 2021] Lower bound for 1-BP($\wedge$):
  - PM(G): graph $G(V, E)$ has a perfect matching:
    - Every $v \in V$ is covered: $\bigvee_{v \in e} x_e$
    - v is not covered twice.
  - **Theorem.** If G is good enough expander, then PM(G) and Tseitin(G) require 1-BP($\wedge$) of size $2^{\Omega(n)}$.
  - **Proof idea:** Consider a moment, when 1-BP contains $\theta(|V|)$ clauses of the first type, then prove that the size of 1-BP representation is exponential.
- Lower bound also holds for 1-NBP($\wedge$).
- Extension rule can not decrease the size of 1-NBP($\wedge$) proof.
- $\varphi \wedge$ (extension axioms) is easy for tree-like Resolution;
- 1-NBP($\wedge$) does not simulate tree-like Resolution.
- Exponential lower bound for 1-NBP($\wedge, \exists_{cn}$).



OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S :=$ {clauses of ϕ}.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
    - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
    - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
    - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
    - Easy formulas: Tseitin formulas, pigeonhole principle.
    - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

OBDD-based SAT algorithms

Input: CNF formula ϕ

1. Choose order π, D^π. Initially $D \equiv 1$.
2. $S := \{\text{clauses of } \phi\}$.
3. While $S \neq \emptyset$ apply the following operations:
   - Conjunction ($\wedge$): Choose $C \in S$; $S := S - C$; $D^\pi := D^\pi \wedge C$
   - Projection ($\exists$): If x does not appear in S, then $D^\pi := (\exists x D)^\pi$
   - Reordering (r): Choose π' and $F^{\pi'}$ such that $F \equiv D$; $\pi := \pi'$ and $D := F$.
4. If $S = \emptyset$ then report whether D is satisfiable or not.

Running time is polynomially connected with the size of the largest D.

- (Aguirre, Vardi 2001), (Pan, Vardi 2005). SAT-solving by symbolic quantifier elimination: $\mathrm{OBDD}(\wedge, \exists)$ algorithms.
  - Easy formulas: Tseitin formulas, pigeonhole principle.
  - Hard formulas: formulas that are hard for $\mathrm{OBDD}(\wedge, \mathrm{w})$

Hard formulas for 1-$\mathrm{NBP}(\wedge, \exists)$ SAT algorithms

- [Itsykson et al, 2017] Hard satisfiable formulas:
  - $C \subseteq \{0,1\}^n$ is a linear code with a large distance and its parity check matrix has $O(1)$ ones in every row and some expansion property.
  - Formula encodes that $x \in C$.
- [I., 2021] Hard unsatisfiable formulas:
  - Weak point: to apply projection on x we have to download all clauses that contain x. Adding extra clauses can make a formula harder.
  - Hard formulas based on tradeoff: either we do not use projection rule and have to solve hard for 1-$\mathrm{NBP}(\wedge)$ formula or we have to download too many clauses and simulate work of 1-$\mathrm{NBP}(\wedge, \exists)$-algorithm on hard satisfiable formulas.
  - 1-$\mathrm{NBP}(\wedge, \exists)$-algorithms do not simulate tree-like Resolution.
- [Ovcharov, 2022] $\mathrm{BPHP}_{2^\ell}^{2^\ell+1}$ are hard for $\mathrm{OBDD}(\wedge, \exists, \mathrm{r})$ algorithms.

Hard formulas for 1-$\mathrm{NBP}(\wedge, \exists)$ SAT algorithms

- [Itsykson et al, 2017] Hard satisfiable formulas:
  - $C \subseteq \{0,1\}^n$ is a linear code with a large distance and its parity check matrix has $O(1)$ ones in every row and some expansion property.
  - Formula encodes that $x \in C$.
- [I., 2021] Hard unsatisfiable formulas:
  - Weak point: to apply projection on x we have to download all clauses that contain x. Adding extra clauses can make a formula harder.
  - Hard formulas based on tradeoff: either we do not use projection rule and have to solve hard for 1-$\mathrm{NBP}(\wedge)$ formula or we have to download too many clauses and simulate work of 1-$\mathrm{NBP}(\wedge, \exists)$-algorithm on hard satisfiable formulas.
  - 1-$\mathrm{NBP}(\wedge, \exists)$-algorithms do not simulate tree-like Resolution.
- [Ovcharov, 2022] $\mathrm{BPHP}^{2^\ell+1}_{2^\ell}$ are hard for $\mathrm{OBDD}(\wedge, \exists, \mathrm{r})$ algorithms.

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?

Open questions

1. Prove natural lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w})$. Hard candidate: binary pigeonhole principle.
2. Separate $\mathrm{OBDD}(\wedge, \exists)$ and $\mathrm{OBDD}(\wedge, \mathrm{w})$. Separation candidate: Clique Coloring principle.
3. Prove lower bound for $\mathrm{OBDD}(\wedge, \mathrm{w}, \mathrm{r})$.
4. Does AC_0-Frege simulate $\mathrm{OBDD}(\wedge)$? Does resolution quasi-polynomially simulate $\mathrm{OBDD}(\wedge)$?
5. Separate dag-like and tree-like $\mathrm{OBDD}(\wedge)$.
6. Prove that random 3CNFs are hard for $\mathrm{OBDD}(\wedge)$.
7. Prove superpolynomial lower bound for 2-$\mathrm{BP}(\wedge)$
8. Is $\mathrm{OBDD}(\wedge)$ automatable?