Equivalences between TFNP^{dt} and Low-Level Proof Systems with an example in $\mathsf{PPA}^{dt} \cong \mathbb{F}_2$-Nullstellensatz

Presented by Mark Chen, Hao Cui and Jiaqian Li

Spring, 2025

Search Problems (Review)

Unrelativized total search problems in **NP**, TFNP

Let $\mathcal{O}$ be encoded by $[m(n)]$, where m is a polynomial.

Definition (Search problem)

A search problem is $S \subseteq \{0,1\}^n \times \mathcal{O}$, such that provided $x \in \{0,1\}^n$, we need to find one of the elements in

$$S(x) := \{o \in \mathcal{O} : (x, o) \in S\}.$$

Search Problems (Review)

Unrelativized total search problems in **NP**, TFNP

Let $\mathcal{O}$ be encoded by $[m(n)]$, where m is a polynomial.

Definition (Search problem)

A search problem is $S \subseteq \{0,1\}^n \times \mathcal{O}$, such that provided $x \in \{0,1\}^n$, we need to find one of the elements in

$$S(x) := \{o \in \mathcal{O} : (x, o) \in S\}.$$

Definition (Total search problem)

A search problem $S \in \{0,1\}^n \times \mathcal{O}$ is *total* if $\forall x \in \{0,1\}^n$, $|S(x)| \geq 1$.

Definition (TFNP)

A total search problem $S \in \mathsf{TFNP}$ that is in **NP**, i.e. $\exists$a poly-time Turing machine $\mathcal{M}$ such that $\forall x \in \{0,1\}^n, \forall y \in S(x)$, $\mathcal{M}(x, y) = 1 \iff (x, y) \in S$.

Bridging $\mathsf{TFNP}^{\mathsf{dt}}$ with Low-level Proof Systems

Query total search problems in query **NP**, $\mathsf{TFNP}^{\mathsf{dt}}$

Definition ($\mathsf{TFNP}^{\mathsf{dt}}$)

A total search problem $S \subseteq \{0,1\}^N \times \mathcal{O}$ is in $\mathsf{TFNP}^{\mathsf{dt}}$ if $\forall y \in \mathcal{O}$, there exists a polylog$(N)$-depth decision tree T_y such that $\forall x \in \{0,1\}^N$,

$$T_y(x) = 1 \iff (x, y) \in S.$$

Definition ($\mathsf{TFNP}^{\mathsf{dt}}$)

A total search problem $S \subseteq \{0,1\}^N \times \mathcal{O}$ is in $\mathsf{TFNP}^{\mathsf{dt}}$ if $\forall y \in \mathcal{O}$, there exists a polylog$(N)$-depth decision tree T_y such that $\forall x \in \{0,1\}^N$,

$$T_y(x) = 1 \iff (x, y) \in S.$$

Remark: One can think of x instance given as a truth table, that typically has $N \approx 2^n$, so polylog(N) = quasipoly(n)-depth trees are meant to capture the query analogue of being “poly-time computable.”

Next, we give a concrete sub-class as example for both TFNP and $\mathsf{TFNP}^{\mathsf{dt}}$. We first define the class PPA in TFNP.

Next, we give a concrete sub-class as example for both TFNP and $\mathsf{TFNP}^{\mathsf{dt}}$. We first define the class PPA in TFNP.

Definition ($\mathsf{PPA} \subseteq \mathsf{TFNP}$)

A search problem $S \in \mathsf{PPA}$ iff it is a TFNP problem whose totality is guaranteed by "hand-shaking lemma."

Lemma (Hand-shaking)

The sum of the degrees of all vertices in a graph is equal to twice the number of edges (in particular, the sum must be even).

We take for granted that all total TFNP problems by "hand-shaking" is poly-time reducible to End-of-undirected-Line:

Definition (End-of-undirected-Line, Informal)

Given the succinct description (as a poly-sized circuit) of a very large degree-2 undirected acyclic graph (with exponentially many nodes). We are also given v_0 which has exactly one neighbor.

Find $v' \neq v_0$ which also has only one neighbor.

We take for granted that all total TFNP problems by "hand-shaking" is poly-time reducible to End-of-undirected-Line:

Definition (End-of-undirected-Line, Informal)

Given the succinct description (as a poly-sized circuit) of a very large degree-2 undirected acyclic graph (with exponentially many nodes). We are also given v_0 which has exactly one neighbor.

Find $v' \neq v_0$ which also has only one neighbor.

Lemma

End-of-undirected-Line *is PPA-complete (by poly-time many-one reduction).*

The notion that $\mathsf{PPA} \subseteq \mathsf{TFNP}$ is the set of all TFNP problems that are poly-time many-one reducible to End-of-undirected-Line is helpful.

Unrelativized to query model, roughly speaking, replace the "efficiently reducible" with "reducible with a family of low-height trees."

The notion that $\mathsf{PPA} \subseteq \mathsf{TFNP}$ is the set of all TFNP problems that are poly-time many-one reducible to End-of-undirected-Line is helpful.

Unrelativized to query model, roughly speaking, replace the "efficiently reducible" with "reducible with a family of low-height trees."

Next up, we will (1) define a query analogue of a $\mathsf{PPA}^{\mathsf{dt}}$-complete problem and (2) specify how "low-depth" decision trees reductions, $\{\mathcal{T}_v\}_{v\in V}$, define problems contained in $\mathsf{PPA}^{\mathsf{dt}}$.

Definition ($\mathsf{PPA}^{\mathsf{dt}}$-OddDeg)

Fix an input $x \in \{0,1\}^N$. An *instance* is a tuple

$$(V, v^* \in V, \{T_v\}_{v \in V}), \qquad |V| \le \mathrm{poly}(N),$$

where

- each T_v is a polylog(N)-height *deterministic decision tree* that, on x, outputs a list $T_v(x) \subseteq V$ with $|T_v(x)| \le 2$;
- the (implicit) graph is:

$$G_x = (V, E_x), \quad \text{where } (u,v) \in E_x \iff u \in T_v(x) \wedge v \in T_u(x).$$

The **search task**: (1) output v^* if $\deg_{G_x}(v^*) \neq 1$, or (2) output a vertex $w \in V$ such that $w \neq v^*$ and $\deg_{G_x}(w)$ is odd.

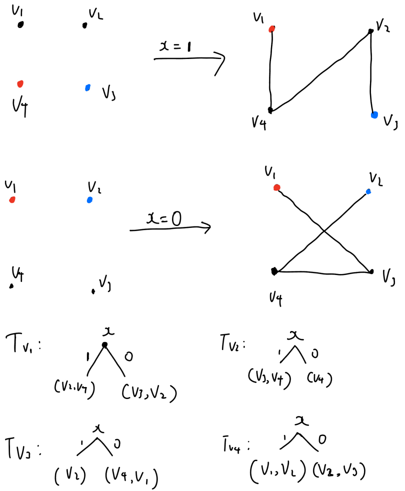
Why is $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg (defn. 9) a query analogue of END-OF-UNDIRECTED-LINE?

Why is $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg (defn. 9) a query analogue of END-OF-UNDIRECTED-LINE?

- On tt x, an END-OF-UNDIRECTED-LINE graph, G_x, is defined.
- This G_x is promise-true instance, solution to which is either (1) refuting its promise, or (2) finding an PPA-style solution.

Bridging TFNP^{dt} with Low-level Proof Systems

Query Complexity: Query PPA class, PPA^{dt} — one-bit, four-vertex example



Bridging $\mathsf{TFNP}^{\mathsf{dt}}$ with Low-level Proof Systems

Query Complexity: Query PPA class, PPA^{dt} — low-depth DT reduction

Class definition: A total search problem belongs to $\mathsf{PPA}^{\mathsf{dt}}$ iff it can be reduced to $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg by the following reduction:

Definition (Low-depth decision tree reduction)

Let S be a $\mathsf{TFNP}^{\mathsf{dt}}$ problem with inputs $x \in \{0,1\}^N$. A *decision-tree reduction* from S to $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg consists of:

1. **Graph constructor.** A family $\{T_v\}_{v \in V}$ of polylog(N)-depth decision trees (recall general $\mathsf{TFNP}^{\mathsf{dt}}$ definition) whose leaves describe the PPA graph G_x on poly(N) vertices.
2. **Solution translator.** A polylog(N)-height decision tree R that, on input (x, w) where w is a valid odd-degree vertex of G_x, outputs a witness $y \in S(x)$.

The reduction is *efficient* if every tree above has depth $O(\text{polylog}(N))$.

Bridging $\mathsf{TFNP}^{\mathsf{dt}}$ with Low-level Proof Systems

Query Complexity: Query PPA class, PPA^{dt} — low-depth DT reduction

Class definition: A total search problem belongs to $\mathsf{PPA}^{\mathsf{dt}}$ iff it can be reduced to $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg by the following reduction:

Definition (Low-depth decision tree reduction)

Let S be a $\mathsf{TFNP}^{\mathsf{dt}}$ problem with inputs $x \in \{0,1\}^N$. A *decision-tree reduction* from S to $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg consists of:

1. **Graph constructor.** A family $\{T_v\}_{v \in V}$ of polylog(N)-depth decision trees (recall general $\mathsf{TFNP}^{\mathsf{dt}}$ definition) whose leaves describe the PPA graph G_x on poly(N) vertices.
2. **Solution translator.** A polylog(N)-height decision tree R that, on input (x, w) where w is a valid odd-degree vertex of G_x, outputs a witness $y \in S(x)$.

The reduction is *efficient* if every tree above has depth $O(\text{polylog}(N))$.

Soundness. Low-depth DT that solves $\mathsf{PPA}^{\mathsf{dt}}$-OddDeg + reduction $\Longrightarrow$ low-depth decision tree that solves $S \in \mathsf{PPA}^{\mathsf{dt}}$.

Definition (Cost of $\mathcal{T} \in \mathsf{PPA}^{dt}$)

Take any PPA^{dt} instance S, it can be WLOG described by the complete problem $\mathcal{T} = (V, v^* \in V, \{T_v\}_{v \in V})$ on input $x \in \{0,1\}^N$, its cost is said to be

$$\max_{v \in V, x \in \{0,1\}^n} \{\text{height of } \mathcal{T}_v(x)\}.$$

Definition (Cost of $\mathcal{T} \in \mathsf{PPA}^{dt}$)

Take any PPA^{dt} instance S, it can be WLOG described by the complete problem $\mathcal{T} = (V, v^* \in V, \{T_v\}_{v \in V})$ on input $x \in \{0,1\}^N$, its cost is said to be

$$\max_{v \in V, x \in \{0,1\}^n} \{\text{height of } \mathcal{T}_v(x)\}.$$

Definition ($\mathsf{PPA}^{dt}(S)$)

$$\mathsf{PPA}^{dt}(S) = \min_{\mathcal{T} \text{ that solves } S} \{\text{cost of } \mathcal{T}\}.$$

Definition ($\mathbb{F}$-Nullstellensatz)

Let $\mathbb{F}$ be a field. An $\mathbb{F}$-Nullstellensatz instance is defined over the polynomial ring of the field $\mathbb{F}[z_1, z_2, \ldots, z_n]$, as such:

- For $i \in [m]$, $p_i \in \mathbb{F}[z_1, z_2, \ldots, z_n]$.
- $P := \{p_i = 0 : i \in [m]\}$ is an unsatisfiable system of polynomial equations.

Goal is to find a $\mathbb{F}$-Nullstellensatz refutation of P, which is a sequence of polynomials in the same polynomial ring:
$q_1, q_2, \ldots, q_m \in \mathbb{F}[z_1, z_2 \ldots, z_n]$, such that $\sum_{i\in[m]} q_i p_i = 1$, syntactically.

Definition ($\mathbb{F}$-Nullstellensatz)

Let $\mathbb{F}$ be a field. An $\mathbb{F}$-Nullstellensatz instance is defined over the polynomial ring of the field $\mathbb{F}[z_1, z_2, \ldots, z_n]$, as such:

- For $i \in [m]$, $p_i \in \mathbb{F}[z_1, z_2, \ldots, z_n]$.
- $P := \{p_i = 0 : i \in [m]\}$ is an unsatisfiable system of polynomial equations.

Goal is to find a $\mathbb{F}$-Nullstellensatz refutation of P, which is a sequence of polynomials in the same polynomial ring:
$q_1, q_2, \ldots, q_m \in \mathbb{F}[z_1, z_2 \ldots, z_n]$, such that $\sum_{i \in [m]} q_i p_i = 1$, syntactically.

Definition ($\mathbb{F}$-Nullstellensatz degree, $\mathsf{NS}_{\mathbb{F}}(P)$)

$$\mathsf{NS}_{\mathbb{F}}(P) = \min_{(q_1, \ldots, q_m)} \max_{q_i} \{\deg(q_i)\}.$$

Definition ($\mathsf{PPA}^{dt}(S)$)

$$\mathsf{PPA}^{dt}(S) = \min_{\mathcal{T} \text{ that solves } S} \max_{v \in V, x \in \{0,1\}^n} \{\text{height of } \mathcal{T}_v(x)\}.$$

Definition ($\mathbb{F}$-Nullstellensatz degree, $\mathsf{NS}_{\mathbb{F}}(P)$)

$$\mathsf{NS}_{\mathbb{F}}(P) = \min_{(q_1, \ldots, q_m)} \max_{q_i} \{\deg(q_i)\}.$$

Bridging $\mathsf{TFNP}^{\mathsf{dt}}$ with Low-level Proof Systems

Landscape

Many equivalences have been set up between query complexity and low-level proof system complexity (summarized in [BFI23]):

- $\mathsf{FP}^{dt} \cong \mathsf{TreeRes}$ [LNNW95].
- $\mathsf{PLS}^{dt} \cong \mathsf{Res}$ [BKT14].
- $\mathsf{PPA}^{dt} \cong \mathbb{F}_2$-NS (given in this presentation as an example) [GKRS19].
- $\mathsf{PPA}_q^{dt} \cong \mathbb{F}_q$-NS, where q is a prime [Kam19].
- $\mathsf{PPADS}^{dt} \cong$ unary-NS [GHJ+24].
- $\mathsf{PPAD}^{dt} \cong$ unary-SA [GHJ+24].
- $\mathsf{SOPL}^{dt} \cong \mathsf{RevRes}$ [GHJ+24].
- $\mathsf{EOPL}^{dt} \cong \mathsf{RevResT}$ [GHJ+24].

Theorem (Main)

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) = \Theta(\mathit{PPA}^{dt}(S(F)))$.

$\mathsf{PPA}^{\mathsf{dt}} \cong \mathbb{F}_2$-NS

Theorem stated and agenda for proofs

Theorem (Main)

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) = \Theta(\mathsf{PPA}^{dt}(S(F)))$.

We prove the main theorem by showing the following directions:

Lemma (Stage ①; [BCE+95])

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) \leq O(\mathsf{PPA}^{dt}(S(F)))$.

Lemma (Stage ②; (Theorem 4) of [GKRS19])

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) \geq \Omega(\mathsf{PPA}^{dt}(S(F)))$.

A common framework for proving equivalences between query problems and low-level proof systems is using the depth of trees that solve FCSP as the translation layer. We will illustrate this framework in the two directions we set out to prove.

$\mathsf{PPA}^{\mathsf{dt}} \cong \mathbb{F}_2$-NS

A common framework

A common framework for proving equivalences between query problems and low-level proof systems is using the depth of trees that solve FCSP as the translation layer. We will illustrate this framework in the two directions we set out to prove.

Definition (False-Clause Search Problem)

Given an unsatisfiable k-CNF, $F = \bigwedge_{i \in [m]} C_i$, and also given an α as an assignment to F, find i such that $C_i(x) = 0$.

Direction 1: NS $\leq$ PPA^{dt}

Overview

Lemma

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) \leq O(\mathsf{PPA}^{dt}(S(F)))$.

On a high level, NS can simulate PPA^{dt} effectively, in the sense that there is a low-degree family of polynomials that simulate a low-height tree that solves a PPA$^{\mathsf{dt}}$ problem.

Proof.

[BCE+95]. In the extended presentation in the uploaded recording. ∎

Lemma

Let F be an unsatisfiable k-CNF. Then, $\mathsf{NS}_{\mathbb{F}_2}(F) \geq \Omega(\mathsf{PPA}^{dt}(S(F)))$. Equivalently, $\mathsf{PPA}^{dt}(S(F)) \leq O(\mathsf{NS}_{\mathbb{F}_2}(F))$.

- Similarly, fix $F = \bigwedge_{i\in[m]} C_i$, and let p_i be the natural polynomial encoding of C_i.
- Fix a degree-d $\mathbb{F}_2$-Nullstellensatz refutation of F: $\sum_{i\in[m]} p_i q_i = 1$.
- It suffices to construct a cost-d PPA-decision tree $\mathcal{T} := (V, v^*, \{o_v\}_{v\in V}, \{\mathcal{T}_v\}_{v\in V})$ solving $S(F)$ (from the given refutation $\sum_{i\in[m]} p_i q_i = 1$).

- Expanding $\sum_{i\in[m]} p_i q_i$ into the sum of monomials.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (\underline{V, v^*, \{o_v\}}, \{\mathcal{T}_v\})$

- Expanding $\sum_{i \in [m]} p_i q_i$ into the sum of monomials.
- Each monomial corresponds to a unique vertex in V
- The 1-term on RHS of $\sum_{i \in [m]} p_i q_i = 1$ corresponds to $v^* \in V$.

- Expanding $\sum_{i\in[m]} p_i q_i$ into the sum of monomials.
- Each monomial corresponds to a unique vertex in V
- The 1-term on RHS of $\sum_{i\in[m]} p_i q_i = 1$ corresponds to $v^* \in V$.
- We can further divide $V = V_1 \cup \cdots \cup V_m \cup V^*$, where V_i contains vertices corresponding to monomials from the expansion of $p_i q_i$, and $V^* = \{v^*\}$.

- Expanding $\sum_{i \in [m]} p_i q_i$ into the sum of monomials.
- Each monomial corresponds to a unique vertex in V
- The 1-term on RHS of $\sum_{i \in [m]} p_i q_i = 1$ corresponds to $v^* \in V$.
- We can further divide $V = V_1 \cup \cdots \cup V_m \cup V^*$, where V_i contains vertices corresponding to monomials from the expansion of $p_i q_i$, and $V^* = \{v^*\}$.
- $\forall i$, for every $v \in V_i$, let $o_v = C_i$ be its associated solution.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \{\mathcal{T}_v\})$

Example

Let $F = (x_1 \vee x_2) \wedge (\neg x_1) \wedge (\neg x_2)$.

- The polynomial equations corresponding to F are $p_1 = (1 + x_1)(1 + x_2), p_2 = x_1, p_3 = x_2$.
- One $\mathbb{F}_2$-Nullstellensatz refutation is $q_1 = 1, q_2 = 1 + x_2, q_3 = 1$.
- $\sum_{i=1}^{3} p_i q_i = (1 + x_1 + x_2 + x_1 x_2) + (x_1 + x_1 x_2) + (x_2) = 1$.

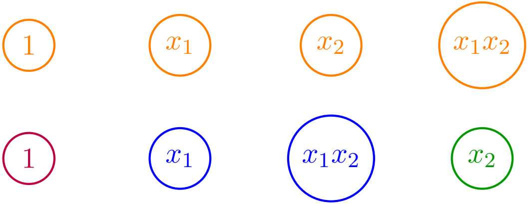


Figure: Vertices of $\mathcal{T}$

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \{\mathcal{T}_v\})$

- Since the polynomials $q_i p_i$ come from a valid $\mathbb{F}_2$-Nullstellensatz refutation, each monomial occurs an even number of times globally in the construction of V.
- Fix a global perfect matching M.
- Note that every edge in M are from two different subsets V_i and V_j ($i \neq j$, or V^*).

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

- Since the polynomials $q_i p_i$ come from a valid $\mathbb{F}_2$-Nullstellensatz refutation, each monomial occurs an even number of times globally in the construction of V.
- Fix a global perfect matching M.
- Note that every edge in M are from two different subsets V_i and V_j ($i \neq j$, or V^*).

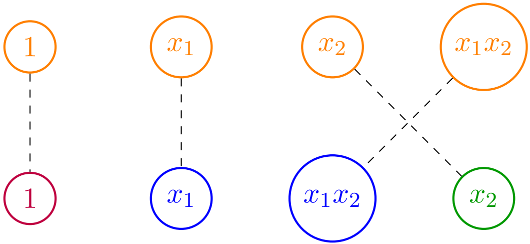


Figure: A global matching of V

On input x, constructing the edges of G_x (the vertex set is V):

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

On input x, constructing the edges of G_x (the vertex set is V):

- (Out-group edges) For each $e \in M$ corresponding to a monomial $m = (v_i, v_j)$, add e_m to G_x if $m(x) = 1$.
- By adding e_m to G_x we mean including v_j in $\mathcal{T}_{v_i}(x)$ and v_i in $\mathcal{T}_{v_j}(x)$.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

On input x, constructing the edges of G_x (the vertex set is V):

- (Out-group edges) For each $e \in M$ corresponding to a monomial $m = (v_i, v_j)$, add e_m to G_x if $m(x) = 1$.
- By adding e_m to G_x we mean including v_j in $\mathcal{T}_{v_i}(x)$ and v_i in $\mathcal{T}_{v_j}(x)$.

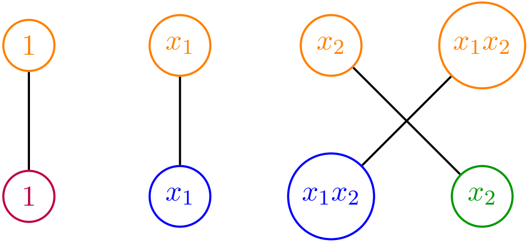


Figure: Out-group edges when $(x_1, x_2) = (1, 1)$

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \{\mathcal{T}_v\})$

On input x, constructing the edges of G_x (the vertex set is V):

- (Out-group edges) For each $e \in M$ corresponding to a monomial $m = (v_i, v_j)$, add e_m to G_x if $m(x) = 1$.
- By adding e_m to G_x we mean including v_j in $\mathcal{T}_{v_i}(x)$ and v_i in $\mathcal{T}_{v_j}(x)$.

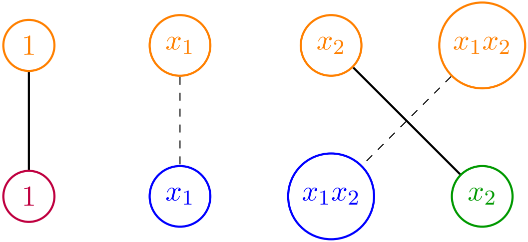


Figure: Out-group edges when $(x_1, x_2) = (0, 1)$

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

On input x, constructing the edges of G_x (the vertex set is V):

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

On input x, constructing the edges of G_x (the vertex set is V):

- (<u>In-group edges</u>) Consider group V_i.
- If $C_i(x) = 0$, i.e., C_i is a valid solution, don't add any edges in V_i.
- Otherwise, $C_i(x) = 1$ and $p_i(x) = 0$.

On input x, constructing the edges of G_x (the vertex set is V):

- (In-group edges) Consider group V_i.
- If $C_i(x) = 0$, i.e., C_i is a valid solution, don't add any edges in V_i.
- Otherwise, $C_i(x) = 1$ and $p_i(x) = 0$.
- Let $\rho := x \restriction \mathrm{vars}(p_i)$, and T_ρ be the multiset of *non-zero* monomials by applying ρ to each monomial in V_i (i.e., from $p_i \cdot q_i$).
- $p_i = 0 \implies p_i q_i = 0$. Therefore, each monomial $m' = m(\rho)$ in T_ρ occurs an even number of times.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

On input x, constructing the edges of G_x (the vertex set is V):

- (In-group edges) Consider group V_i.
- If $C_i(x) = 0$, i.e., C_i is a valid solution, don't add any edges in V_i.
- Otherwise, $C_i(x) = 1$ and $p_i(x) = 0$.
- Let $\rho := x \restriction \mathrm{vars}(p_i)$, and T_ρ be the multiset of *non-zero* monomials by applying ρ to each monomial in V_i (i.e., from $p_i \cdot q_i$).
- $p_i = 0 \implies p_i q_i = 0$. Therefore, each monomial $m' = m(\rho)$ in T_ρ occurs an even number of times.
- Fix another perfect matching M_ρ.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \{\mathcal{T}_v\})$

On input x, constructing the edges of G_x (the vertex set is V):

- (In-group edges) Consider group V_i.
- If $C_i(x) = 0$, i.e., C_i is a valid solution, don't add any edges in V_i.
- Otherwise, $C_i(x) = 1$ and $p_i(x) = 0$.
- Let $\rho := x \restriction \text{vars}(p_i)$, and T_ρ be the multiset of *non-zero* monomials by applying ρ to each monomial in V_i (i.e., from $p_i \cdot q_i$).
- $p_i = 0 \implies p_i q_i = 0$. Therefore, each monomial $m' = m(\rho)$ in T_ρ occurs an even number of times.
- Fix another perfect matching M_ρ.
- For each edge $e \in M_\rho$ corresponding to a monomial $m' := m(\rho)$, add e to G_x if $m(x) = 1 (= m'(x))$.

On input x, constructing the edges of G_x (the vertex set is V):

- (In-group edges) Consider group V_i.

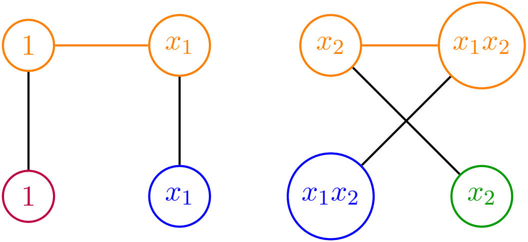


Figure: In-group edges when $(x_1, x_2) = (1, 1)$

- Complexity. For a vertex $v \in V_i$, its incident edge (of both types) can be decided by *the variables in* p_i plus *the variables in its corresponding monomial.* Therefore, height of $\mathcal{T}_v \leq d + k = O(d)$.

Direction 2: $\mathsf{PPA}^{dt} \leq \mathsf{NS}$

Constructing $\mathcal{T} = (V, v^*, \{o_v\}, \underline{\{\mathcal{T}_v\}})$

- Complexity. For a vertex $v \in V_i$, its incident edge (of both types) can be decided by *the variables in p_i* plus *the variables in its corresponding monomial*. Therefore, height of $\mathcal{T}_v \leq d + k = O(d)$.
- Correctness. Fix x. $\deg(v^*) = 1$ always holds.
  For a monomial m in a true clause, let v be its corresponding vertex.
  - If $m(x) = 0$, $\deg(v) = 0$.
  - Otherwise, $m(x) \neq 0 \implies m(\rho) \neq 0$. The vertex corresponding to m will have both an out- and in-group neighbor. $\deg(v) = 2$.

Conclusion

- TFNP^{dt}, PPA^{dt} and the decision tree reduction;
- $\mathsf{NS}_{\mathbb{F}_2}(F) = \Theta(\mathsf{PPA}^{dt}(S(F)))$ for any unsatisfiable k-CNF F.
- A step of the lifting paradigm:

$$\text{Proof-complexity LB} \Longrightarrow \text{Query-complexity LB}$$

$$\overset{\text{lifting}}{\Longrightarrow} \text{Comm. LB} \overset{\text{KW}}{\Longrightarrow} \text{Model of Computation LB},$$

which further implies the lower bound of Monotone Span Programs / Linear SSS.

[BCE+95] Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi.
The relative complexity of np search problems.
In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 303–314, 1995.

[BFI23] Sam Buss, Noah Fleming, and Russell Impagliazzo.
Tfnp characterizations of proof systems and monotone circuits.
In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, pages 30–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023.

[BKT14] Samuel R Buss, Leszek Aleksander Kołodziejczyk, and Neil Thapen.
Fragments of approximate counting.
The Journal of Symbolic Logic, 79(2):496–525, 2014.

[GHJ+24] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao.
Separations in proof complexity and tfnp.
Journal of the ACM, 71(4):1–45, 2024.

[GKRS19] Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov.
Adventures in monotone complexity and tfnp.
In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, pages 38–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019.

[Kam19] Pritish Kamath.
Some hardness escalation results in computational complexity theory.
PhD thesis, Massachusetts Institute of Technology, 2019.

[LNNW95] László Lovász, Moni Naor, Ilan Newman, and Avi Wigderson.

Search problems in the decision tree model.
SIAM Journal on Discrete Mathematics, 8(1):119–132, 1995.