1 Definitions

1.1 Tseitin formulas

Let $G(V,E)$ be a graph. Let $c : V \to \{0,1\}$ be a **charge function**. A Tseitin formula $\mathrm{T}(G,c)$ depends on the propositional variables x_e for $e \in E$. For each vertex $v \in V$ we define the parity condition of v as

$$P_v := \left(\sum_{e \text{ is incident to } v} x_e \equiv c(v) \bmod 2 \right).$$

The **Tseitin formula** $\mathrm{T}(G,c)$ is the conjunction of parity conditions of all the vertices: $\bigwedge_{v \in V} P_v$. Tseitin formulas is represented in CNF as follows: we represent P_v in CNF in the canonical way for all $v \in V$.

Lemma 1. *A Tseitin formula* $\mathrm{T}(G,c)$ *is satisfiable if and only if for every connected component* $C(U, E_U)$ *of the graph* G*, the condition* $\sum_{u \in U} c(u) \equiv 0 \bmod 2$ *holds.*

1.2 NBP and 1-NBP

A **nondeterministic branching program** (NBP) represents a Boolean function $f : \{0,1\}^n \to \{0,1\}$. A nondeterministic branching program for a function $f(x_1, x_2, \dots, x_n)$ is a directed acyclic graph. It has three types of nodes: guessing nodes (OR nodes), nodes labeled with a variable (we call them just labeled nodes) and two sinks; the unique source is either a guessing node or a labeled node.

- Sinks are labeled with 0 and 1.
- Each of labeled nodes labeled with a variable from $\{x_1, x_2, \dots, x_n\}$ and has exactly two outgoing edges: the first edge is labeled with 0, the second is labeled with 1.
- Each of guessing nodes has two outgoing unlabeled edges.

The value of every node is defined recursively. Each node v of a branching program computes a function $f_v : \{0,1\}^n \to \{0,1\}$.

- For a $k \in \{0,1\}$, the sink s labeled with k, computes the function $f_s \equiv k$.
- Assume that a node v is labeled with x_i, the outgoing edge from v labeled with 0 ends in a node v_0 and the outgoing edge labeled with 1 ends in a node v_1. Then $f_v(x_1, \dots, x_n)$ equals $f_{v_1}(x_1, \dots, x_n)$ if $x_i = 1$ and equals $f_{v_0}(x_1, \dots, x_n)$ if $x_i = 0$.
- Assume that a node v is a guessing node and v_0 and v_1 are two direct successors of v. Then $f_v(x_1, \dots, x_n)$ equals $f_{v_1}(x_1, \dots, x_n) \vee f_{v_0}(x_1, \dots, x_n)$.

We say that the branching program computes the function computed in its source.

The size of a nondeterministic branching program is the number of nodes in it.

A nondeterministic branching program is **read-once** (1-NBP) if every path in it contains at most one occurrence of each variable.

1.3 BP and 1-BP

A **branching program** (BP) is a NBP such that there are no guessing nodes.

A **read-once** branching program (1-BP) is a 1-NBP such that there no guessing nodes.

1.4 OBDD

Let π be a permutation of the set $\{1, \ldots, n\}$ (an order).

A π**-ordered nondeterministic binary decision diagram** ($\pi-$NOBDD) is a 1-NBP such that on every path from the source to a sink variable $x_{\pi(i)}$ can not appear before $x_{\pi(j)}$ if $i > j$. A **nondeterministic ordered binary decision diagram** (NOBDD) is a π-ordered nondeterministic binary decision diagram for some π.

A π**-ordered binary decision diagram** ($\pi-$OBDD) is a $\pi-$NOBDD such that there are no guessing nodes. A **ordered binary decision diagram** (OBDD) is a π-ordered binary decision diagram for some π.

2 Minimal read-once branching program for satisfiable Tseitin formulas is OBDD

Our goal is to prove that if one wants to compute the value of satisfiable Tseitin formula, then NBP is not stronger (in terms of size) than OBDD.

We use the notation $\#G$ for the number of connected components of G.

Lemma 2. *If a Tseitin formula* $\mathrm{T}(G, c)$ *is satisfiable, then it has* $2^{|E|-|V|+\#G}$ *satisfiable assignments.*

Lemma 3. *For every* $f \in \mathcal{F}_J(G, c)$ *the size of the set* $\{\alpha \in A_{G,c} \mid f = \mathrm{T}(G, c)|_{\alpha_J}\}$ *equals* $2^{|E|-2|V|+\#G_{E\setminus J}+\#G_J}$.

For a graph $G(V, E)$ and a set of edges $J \subseteq E$ we introduce the notation $\mathrm{comp}_J(G) = |V| - \#G_{E\setminus J} - \#G_J + \#G$.

Lemma 4. *The size of the set* $\mathcal{F}_J(G, c)$ *is equal to* $2^{\mathrm{comp}_J(G)}$.

Proof. $2^{\mathrm{comp}_J(G)} = 2^{|E|-|V|+\#G}/2^{|E|-2|V|+\#G_{E\setminus J}+\#G_J}$. □

Lemma 5. *Let* D *be a* 1-NBP *computing a satisfiable* $\mathrm{T}(G, c)$*. Let* s *be a node of* D *such that there is an accepting path passing through* s*. Then the following holds:*

1) every two paths from the source to s *assign values to the same set of variables* $\{x_e \mid e \in J\}$*, where* $J \subseteq E$*;*

2) the maximal number of accepting paths passing through s *corresponding to different satisfying assignments of* $\mathrm{T}(G, c)$ *is at most* $2^{|E|-2|V|+\#G_{E\setminus J}+\#G_J}$.

Let D be a nondeterministic OBDD using the order of variables $x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$. For every $i \in [n]$, the i-th level of D is the set of nodes labeled with variable $x_{\pi(i)}$.

Lemma 6. *Let* $\mathrm{T}(G, c)$ *be a satisfiable Tseitin formula based on a graph* $G(V, E)$*. Let* π *be a permutation of* $[|E|]$*. For every* i *from* 0 *to* $|E|$ *we denote by* J_i *the set* $\{e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(i)}\}$ *of the first* i *edges according to permutation* π*.*

1) Let D *be a nondeterministic* π*-ordered* OBDD *computing* $\mathrm{T}(G, c)$*. Then for every* i *from* $[|E|]$*, the* i*-th level of* D *contains at least* $2^{\mathrm{comp}_{J_{i-1}}(G)}$ *nodes.*

2) If D *is a minimal* π-OBDD *computing* $\mathrm{T}(G, c)$*, then the* i*-th level of* D *has exactly* $2^{\mathrm{comp}_{J_i}(G)}$ *nodes and for every node* s *from the* i*-th level of* D *there are exactly* $2^{|E|-2|V|+\#G_{E\setminus J_{i-1}}+\#G_{J_{i-1}}}$ *accepting paths going through* s*.*

Lemma 7. *The size of any* 1-NBP *computing a satisfiable* $\mathrm{T}(G, c)$ *is at least the minimal size of* OBDD *computing* $\mathrm{T}(G, c)$*.*

Sketch. Consider a minimal read-once nondeterministic branching program D computing $\mathrm{T}(G, c)$. Consider a set of accepting paths P.

For every node v of D we denote by $q(v)$ the number of paths from P passing through v. For every $p \in P$, we consider the value $\gamma(p) = \sum_{v\in p} \frac{1}{q(v)}$, where summation goes over all vertices of p.

Notice that $\sum_{p\in P} \gamma(p) = |D| - 1$, since every node v except the 0-sink was calculated $q(v)$ times with weight $\frac{1}{q(v)}$. Hence, $|D| - 1 \geqslant |P| \min\limits_{p\in P} \gamma(p)$. Let $\min\limits_{p\in P} \gamma(p)$ be achieved on a path $p^* \in P$. Let π be a permutation corresponding to the order of the edges in p^* in the direction from the source to the 1-sink.

Let D' be a minimal $\pi-$OBDD computing $T(G,c)$, let P' be set of accepting paths of D', and let p' be the path in D' corresponding to the path p^* in D. For any node v of D' we denote by $q'(v)$ the number of accepting paths passing through v. For any path p in D' we define $\gamma'(p) = \sum_{v \in p} \frac{1}{q'(v)}$.

By previous lemmas, $\gamma'(p)$ does not depend on p, and $q(v) \geqslant q'(v')$ for each pair of corresponding vertices v and v' on the paths p' and p^*, so $\gamma(p') \leqslant \gamma(p^*)$ and

$$|D'| - 1 = \sum_{p \in P'} \gamma'(p) = |P'|\gamma(p') \leqslant |P|\gamma(p^*) \leqslant \sum_{p \in P} \gamma(p) = |D| - 1,$$

and, thus, $|D| \geqslant |D'|$. □