Relationship between circuit complexity and symmetry

Satoshi Tazawa *

ABSTRACT

It is already shown that a Boolean function for a NP-complete problem can be computed by a polynomial-sized circuit if its variables have enough number of automorphisms. Looking at this previous study from the different perspective gives us the idea that the small number of automorphisms might be a barrier for a polynomial time solution for NP-complete problems.

Here I show that by interpreting a Boolean circuit as a graph, the small number of graph automorphisms and the large number of subgraph automorphisms in the circuit establishes the exponential circuit lower bound for NP-complete problems. As this strategy violates the largeness condition in Natural proof, this result shows that \( P \neq NP \) without any contradictions to the existence of pseudorandom functions.

Categories and Subject Descriptors

F.2 [Theory of Computation]: Analysis of algorithms and problem complexity

General Terms

Theory

Keywords

Circuit complexity, graph automorphism, \( P \neq NP \)

1. PRELIMINARY AND OUTLINE OF THIS PAPER

In this paper, unbounded depth Boolean circuits with standard gates AND, OR and NOT are discussed. AND gates and OR gates have 2 fan-in and unbounded fan-out. In section 3, a NP-complete Boolean function \( f_k \) for the \( k \)-clique problem with \( n \) vertices is discussed. \( f_k \) can be written as two different ways,

\[
f_k(e_{12}, e_{13}, \ldots, e_{(n-1)n}) = f_k(x_1, x_2, x_3, \ldots, x_{(n)^2})
\]

(The former emphasizes the variables in \( f_k \) as edges, the latter emphasizes the number of variables in \( f_k \), and both expressions are used.)

Section 3.1 shows that the proof strategy in Section3 is non-Naturalizable.

In section 4, the relationship to other open problems in computational complexity theory is discussed.

2. INTRODUCTION

Many approaches have been proposed to solve the famous \( P \neq NP \) problem. Among them, circuit complexity has been studied in order to separate the complexity classes. As the exponential circuit lower bound for some NP-complete problem means \( P \neq NP \), much effort is devoted to show such lower bound. Although no exponential circuit lower bound for NP-complete problems is known for general circuit, the exponential circuit lower bound for problems in NP is obtained with restrictions on its depth, kinds of gates available, and so on. However such attempts cannot be extended to general circuits, and the reason why these attempts fail in general circuits is discussed in Natural proof. Natural proof showed that proof strategies which are natural or naturalizable can not succeed in establishing the exponential lower bound for NP-complete problems under the assumption that there exist the pseudorandom functions. As it is widely believed that pseudorandom functions exist, a promising approach needs to be non-naturalizable (In other words, it needs to violate one of the conditions; constructivity, largeness, usefulness) in Natural proof.

Apart from this, it is already shown that a Boolean function for a NP-complete problem can be computed by a polynomial-sized circuit if its variables have enough number of automorphisms and many difficult SAT instances do not have symmetries. Looking at this previous study from the different perspective gives us the idea that the small number of automorphisms might be a barrier for a polynomial time solution for NP-complete problems.

Here I show that by interpreting a Boolean circuit as a graph, the small number of graph automorphisms (global symmetry) and the large number of subgraph automorphisms (local symmetry) in the circuit establishes the exponential circuit lower bound for NP-complete problems. As this strategy violates the largeness condition in Natural proof,
this result shows that $P \neq NP$ without any contradictions to the existence of pseudorandom functions.

3. PROOF

Before going into the discussion of the Boolean circuit of a NP-complete problem, it is necessary to explain the detailed outline of the proof. In order to show the exponential circuit lower bound, it is necessary to derive an idea from the following well-known fact:

\[ A \quad \text{C}^\infty \text{ function } f(x) \text{ can be written as an infinite series} \]

\[ f(x) = \sum_{n=0}^{\infty} a_n x^n \quad (2) \]

If constraints on $f(x)$ are given, for example $f^{(n)}(0) = 1$, we can specify the form of $f^{(n)}$ as $f^{(n)}(0) = n!a_n = 1 \iff a_n = \frac{1}{n!}$. As a result, (2) can be written as

\[ f(x) = \sum_{n=0}^{\infty} \frac{1}{n!} x^n \quad (3) \]

This function $f(x)$ has a simpler form $f(x) = e^x$. Similarly, any Boolean function $f(x_1, x_2, \ldots, x_n)$ can be written in the disjunctive normal form.

\[ f(x_1, x_2, \ldots, x_n) = \bigvee_{j=1}^{m} C_j \quad (4) \]

If constraints on $f(x_1, x_2, \ldots, x_n)$ are given, we can specify the form of $f(x_1, x_2, \ldots, x_n)$, $f(x_1, x_2, \ldots, x_n)$ may have a simpler form than expressed in the disjunctive normal form. In order to establish the exponential circuit lower bound for a Boolean function of a NP-complete problem, it is reasonable to specify the form of $f(x_1, x_2, \ldots, x_n)$ before its size is measured. Of course, the size of the Boolean circuit should not be measured in the disjunctive normal form as conversion into the disjunctive normal form sometimes results in an exponential explosion in the formula. So it is necessary to determine the lower bound of the size of the circuits which are logically equivalent to $f(x_1, x_2, \ldots, x_n)$ in the specified form (4).

In order to separate the $P/poly$ and $NP$, a Boolean function $f_k$ for the $k$-clique problem with $n$ vertices (NP-complete problem) is discussed. A graph with $n$ vertices can be encoded in binary using $\binom{n}{2}$ bits (Each bit represents one of possible edges). In order to specify the form of $f_k$, the symmetry of variables in the Boolean circuit needs to be encoded in binary using $\binom{n}{2}$ bits. For a permutation $\sigma \in S_{\binom{n}{2}}$, $\sigma \in \text{Aut}(f_k)$

\[ \text{if } f_k(x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(\binom{n}{2})}) = f_k(x_1, x_2, \ldots, x_{\binom{n}{2}}) \quad (5) \]

A Boolean function $f_k$ has no automorphism except trivial automorphisms caused by permutations of labels on vertices. That is, for any permutation $\sigma \in S_n$, it follows that

\[ f_k(\epsilon_{12}, \epsilon_{13}, \ldots, \epsilon_{(n-1)n}) = f_k(\epsilon_{\sigma(1)}\epsilon(2), \epsilon_{\sigma(1)}\epsilon(3), \ldots, \epsilon_{\sigma(n-1)}\epsilon(n)) \quad (6) \]

$f_k$ is relatively asymmetrical based on the fact

\[ \frac{\text{The number of automorphisms of } f_k}{\text{The number of all possible permutations}} = \frac{n!}{\binom{n}{2}!} \quad (7) \]

However this information is not enough to specify the form of $f_k$ as there exist many kinds of asymmetrical circuits. So it is necessary to examine not only the global symmetry but also the local symmetry of the circuit as a graph. Unlike ordinary graphs, exchangeability of gates should be taken into considerations. Gates AND($\land$) and OR($\lor$) should be regarded as exchangeable gates when used in the forms

\[ a \land b (\Leftrightarrow a) \land b, a \lor b (\Leftrightarrow a \lor b), \neg a \land \neg b (\Leftrightarrow \neg a \land \neg b), \neg a \lor \neg b (\Leftrightarrow \neg a \lor \neg b). \quad (8) \]

A NOT gate($\neg$) works as an inexchangable gate when used in the forms

\[ \neg a \land \neg b \land a, \neg a \lor \neg b \lor a \quad (9) \]

In this paper, in order to measure the local symmetry of variables $X$ in a Boolean circuit $f$, we express the Boolean function $f$ in the disjunctive normal form and examine the automorphisms of $X$ after applying false values 0 to the remaining variables.

For example, let $f$ denote a Boolean function

\[ f = (x_1 \land x_2 \land x_3) \lor (x_2 \land x_4 \land \neg x_5) \quad (10) \]

By restricting variables,

(A) For $X = \{x_1, x_2, x_3\}$,

\[ f_{\text{restricted}}(X) = f_{\text{restricted}}(x_1, x_2, x_3) = f_{\text{restricted}}(x_1, x_2, x_3, 0, 0) = (x_1 \land x_2 \land x_3) \lor (x_2 \land 0 \land 1) = (x_1 \land x_2 \land x_3). \]

\[ \text{Aut}(f_{\text{restricted}}(x_1, x_2, x_3)) \cong S_3. \]

(B) For $X = \{x_1, x_2\}$,

\[ f_{\text{restricted}}(X) = f_{\text{restricted}}(x_1, x_2) = f_{\text{restricted}}(x_1, x_2, 0, 0, 0) = (x_1 \land x_2 \land 0) \lor (x_2 \land 0 \land 1) = 0. \]

\[ \text{Aut}(f_{\text{restricted}}(x_1, x_2)) \cong S_2. \]

(C) For $X = \{x_2, x_4, x_5\}$,

\[ f_{\text{restricted}}(X) = f_{\text{restricted}}(x_2, x_4, x_5) = f_{\text{restricted}}(0, x_2, 0, x_4, x_5) = (0 \lor x_2 \land 0) \lor (x_2 \lor x_4 \land \neg x_5) = (x_2 \land x_4 \land \neg x_5). \]

\[ \text{Aut}(f_{\text{restricted}}(x_2, x_4, x_5)) \cong S_2(x_2 \land x_4 \land \neg x_5). \]

(D) For $X = \{x_1, x_2, x_3, x_4\}$,

\[ f_{\text{restricted}}(X) = f_{\text{restricted}}(x_1, x_2, x_3, x_4) = f_{\text{restricted}}(x_1, x_2, x_3, x_4, 0) = (x_1 \land x_2 \land x_3) \lor (x_2 \land x_4 \land 1) = (x_1 \land x_2 \land x_3) \lor (x_2 \land x_4). \]

\[ \text{Aut}(f_{\text{restricted}}(x_1, x_2, x_3, x_4)) \cong S_2(x_1 \land x_2 \land x_3 \land x_4). \]

In order to measure the local symmetry of $f_k$, let $X_k = \{x_1, x_2, \ldots, x_{\binom{n}{2}}\} = \{\text{edges among a } k-\text{clique}\}$. As a graph with $n$ vertices has $\binom{n}{2}$ candidate $k$-cliques, $X_k$ is used as a representative of $\binom{n}{2}$ candidate $k$-cliques $X_k^1, X_k^2, \ldots, X_k^{\binom{n}{2}}$. For a Boolean function $f_k$ for the $k$-clique problem with $n$ vertices, the local symmetry of $f_k$ can be expressed as follows.

**Theorem 1.** $\text{Aut}(f_k_{\text{restricted}}(X_k)) \cong S_{\binom{n}{2}}$.

A proof of theorem 1 is shown later. In order to prove theorem 1, it is necessary to understand the relationship between the symmetry (or asymmetry) of variables and the structure of Boolean function. To reduce the number of possibilities of structures of Boolean functions, the following theorem is useful.

**Theorem 2.** For $f_k_{\text{restricted}}(X_k) = C_1 \lor C_2 \lor \ldots \lor C_m$,
each one of the clauses, \( C_i (1 \leq i \leq m) \), has to contain all of the variables in \( X_k \).

Proof. A method of proof by contradiction is used. If \( C_i (1 \leq i \leq m) \) contains only \( l (\leq (\binom{n}{2})) \) variables, then two cases are conceivable.
(1) \( C_i \) is satisfiable if the truth values of \( l \) variables are appropriately chosen.
(2) \( C_i \) is not satisfiable for any of the truth values.

In case (1), \( C_i = 1 \) for \( l \) variables with appropriately chosen truth values. So

\[
f_k^{\text{restricted}} (X_k) = C_1 \lor C_2 \lor \ldots \lor C_i \lor \ldots \lor C_m = 1 \quad (11)
\]

But if the variable not used in \( C_i \) takes 0, \( f_k^{\text{restricted}} (X_k) \) should return 0 as \( X_k \) does not form a \( k \)-clique. This contradicts with (11).

In case (2), as \( C_i = 0 \)

\[
f_k^{\text{restricted}} (X_k) = C_1 \lor C_2 \lor \ldots \lor C_i \lor \ldots \lor C_m = C_1 \lor C_2 \lor \ldots \lor C_{i-1} \lor 0 \lor C_{i+1} \lor \ldots \lor C_m = C_1 \lor C_2 \lor \ldots \lor C_{i-1} \lor C_{i+1} \lor \ldots \lor C_m
\]

\( C_i \) does not influence the return value and should be erased. Therefore each one of \( C_i \) has to contain all of the variables in \( X_k \).

By theorem 2, we just need to consider clauses, each one of which contains all of the variables in \( X_k \). Regarding the symmetry of variables in a clause, the following theorem follows.

Theorem 3. For a clause \( C(X_k) \) in which all of the variables are connected by \( \land \), it follows that

\[
\text{Aut}(C(X_k)) \cong S_{\binom{n}{2}}
\]

Therefore \( C(X_k) \) is \( (x_1 \land x_2 \land \ldots \land x_{\binom{n}{2}}) \) or \( (\neg x_1 \land \neg x_2 \land \ldots \land \neg x_{\binom{n}{2}}) \)

Proof. To show \( \Rightarrow \) is trivial. So it is necessary to show \( \Leftarrow \). If \( \text{Aut}(C(X_k)) \cong S_{\binom{n}{2}} \), then all of the variables in \( X_k \) are exchangeable. Based on the simply observation, \( x_i \) and \( x_j \) (\( x_i, x_j \in X_k, i \neq j \)) are exchangeable in \( C(X_k) \) if and only if \( x_i \) and \( x_j \) take the forms \( (x_i \land x_j) \) or \( (\neg x_i \land \neg x_j) \). Therefore in order for all of the variables in \( X_k \) to be exchangeable, \( C(X_k) = (x_1 \land x_2 \land \ldots \land x_{\binom{n}{2}}) \) or \( (\neg x_1 \land \neg x_2 \land \ldots \land \neg x_{\binom{n}{2}}) \)

Regarding the asymmetry, many possibilities can be considered. So I discuss the case where one transposition does not follow.

Theorem 4. For a clause \( C(X_k) \) in which all of the variables are connected by \( \land \), one transposition, say \( (x_1, x_2)(x_1, x_2 \in X_k) \), does not follow

\[
\text{Aut}(C(X_k)) = (x_1 \lor \neg x_2 \land \text{remaining variables}) \text{ or } (\neg x_1 \land x_2 \land \text{remaining variables})
\]

Proof. To show \( \Leftarrow \) is trivial. So it is necessary to show \( \Rightarrow \). Based on the simply observation, \( x_1 \) and \( x_2 \) are exchangeable if and only if one of them is connected to a NOT gate. As all of the variables in a clause are connected by \( \land \), \( C(X_k) = (x_1 \land x_2 \land \text{remaining variables}) \) or \( (\neg x_1 \land x_2 \land \text{remaining variables}) \)

Using these results, theorem 1 is shown here.

Theorem 1. \( \text{Aut}(f_k^{\text{restricted}} (X_k)) \cong S_{\binom{n}{2}} \).

Proof. A method of proof by contradiction is used. For \( f_k^{\text{restricted}} (X_k) = C_1 (X_k) \lor C_2 (X_k) \lor \ldots \lor C_m (X_k) \), suppose if \( \text{Aut}(C_i (X_k)) \not\cong S_{\binom{n}{2}} \), then one of the transpositions, say \( (x_1 x_2)(x_1, x_2 \in X_k) \), does not follow in \( C_i (X_k) \). To satisfy this exchangeability of \( x_1 \) and \( x_2 \),

\[
C_i (X_k) = (\neg x_1 \land x_2) \bigwedge_{3 \leq \binom{n}{2}} (\neg x_i) \bigwedge_{3 \leq \binom{n}{2}} x_w (z \neq w) \quad (12)
\]

or \( C_i (X_k) = (x_1 \land \neg x_2) \bigwedge_{3 \leq \binom{n}{2}} (\neg x_i) \bigwedge_{3 \leq \binom{n}{2}} x_w (z \neq w) \quad (13)
\]

However by assigning values \( x_1 = 0, x_2 = 1, x_3 = 0, x_w = 1 \) to (12). (12) returns 1 though \( f_k^{\text{restricted}} (X_k) \) should return 0. By assigning values \( x_1 = 1, x_2 = 0, x_3 = 0, x_w = 1 \) to (13). (13) returns 1 though \( f_k^{\text{restricted}} (X_k) \) should return 0. Therefore \( \text{Aut}(C_i (X_k)) \cong S_{\binom{n}{2}} \) and \( \text{Aut}(f_k^{\text{restricted}} (X_k)) \cong \text{Aut}(C_1 (X_k) \lor C_2 (X_k) \lor \ldots \lor C_m (X_k)) \cong S_{\binom{n}{2}} \).

Theorem 5. \( f_k^{\text{restricted}} (X_k) = (x_1 \land x_2 \land \ldots \land x_{\binom{n}{2}}) \)

Proof. By theorem 1, \( f_k^{\text{restricted}} (X_k) = C_1 (X_k) \lor C_2 (X_k) \lor \ldots \lor C_m (X_k) \) and \( \text{Aut}(C_i (X_k)) \cong S_{\binom{n}{2}} \) then for any input \( (X_k, y) \) the return value of \( f_k^{\text{restricted}} (X_k, y) \) does not change after the permutation on variables. However for two inputs \( (X_k, y) = (x_1, x_2, \ldots, x_{\binom{n}{2}}, y) = (1, 1, \ldots, 1, 1, 0) \) and \( (y, x_2, \ldots, x_{\binom{n}{2}}, x_1) = (0, 1, \ldots, 1, 1, 1)(x_1 \land y \text{ are exchanged}) \), the return values of each of these inputs need to be different.

\[
f_k^{\text{restricted}} (x_1, x_2, \ldots, x_{\binom{n}{2}}, y) \neq f_k^{\text{restricted}} (0, 1, \ldots, 1, 1) = 0
\]

Therefore \( \text{Aut}(f_k^{\text{restricted}} (X_k, y)) \not\cong S_{\binom{n}{2}}+1 \).

Based on these results, the local structure of Boolean function \( f_k \) can be specified. So next, it is necessary to specify its global structure based on its local structure. For \( f_k(x_1, x_2, \ldots, x_{\binom{n}{2}}) = C_1 \lor C_2 \lor \ldots \lor C_m \) \( (C_j (1 \leq j \leq m) \) is not the same as \( C_j \) used in the discussion above), the following theorem follows.

Theorem 7. \( (A) C_j (1 \leq j \leq m) \) has to contain at least \( \binom{n}{2} \) variables.
(\( (B) \) \( \binom{n}{2} \) variables in \( X_i (1 \leq i \leq \binom{n}{2}) \) have to be contained in one of \( C_j (1 \leq j \leq m) \).
\( (C) \) After reordering clauses, we can take \( C_i (X_i) = f_k^{\text{restricted}} (X_i) = (x_1 \land x_2 \land \ldots \land x_{\binom{n}{2}}) \leq \binom{n}{2} \). \( (i \leq \binom{n}{2}) \).
PROOF. (A) Like the discussion in theorem2, a method of proof by contradiction is used. If \( C_j (1 \leq j \leq m) \) contains only \( l (\leq \binom{m}{2}) \) variables, then two cases are conceivable. 
(1) \( C_j \) is satisfiable if the truth values of \( l \) variables are appropriately chosen. 
(2) \( C_j \) is not satisfiable for any of the truth values. 
In case (1), \( C_j = 1 \) for \( l \) variables with appropriately chosen truth values. So 
\[
f_k (x_1, x_2, \ldots, x_{(l)} ) = C_1 \lor C_2 \lor \ldots \lor C_m = 1 \quad (15)
\]
But if the variable not used in \( C_j \) and the remaining variables take 0, \( f_k (x_1, x_2, \ldots, x_{(l)} ) \) should return 0 as \( X_k \) does not form a \( k \)-clique. This contradicts with (15). 

In case (2), as \( C_j = 0 \) 
\[
f_k (x_1, x_2, \ldots, x_{(l)} ) = C_1 \lor C_2 \lor \ldots \lor C_{j-1} \lor C_{j+1} \lor \ldots \lor C_m
\]
\[
= C_1 \lor C_2 \lor \ldots \lor C_{j-1} \lor 0 \lor C_{j+1} \lor \ldots \lor C_m
\]
\[
= C_1 \lor C_2 \lor \ldots \lor C_{j-1} \lor C_{j+1} \lor \ldots \lor C_m
\]
\( C_j \) does not influence the return value and should be erased. Therefore each one of \( C_j \) has to contain at least \( \binom{l}{2} \) variables. 

(B) If \( \binom{l}{2} \) variables in \( X_k \) \((1 \leq i \leq \binom{l}{2})\) are not contained in any of \( C_j \), then \( f_k^{\text{restricted}} (X_k) \) does not contain \( \binom{l}{2} \) variables in \( X_k \), which contradicts with theorem 5. Therefore \( \binom{l}{2} \) variables in \( X_k \) \((1 \leq i \leq \binom{l}{2})\) have to be contained in one of \( C_j (1 \leq j \leq m) \).

(C) For clauses which contain \( \binom{l}{2} \) variables in \( X_k \) suppose if all of them have more than \( \binom{l}{2} \) variables. Then a clause \( C_j \) which satisfies the above condition takes the forms 
\[
C_j = (x_1^i \land x_2^i \land \ldots \land x_{(l)}^i \land y_1 \land Y) \quad (16)
\]
(\( Y \) is a clause in which variables are connected by \( \land \)). 
\[
\text{or} \quad C_j = (x_1^i \land x_2^i \land \ldots \land x_{(l)}^i \land \bigwedge_{i=1}^{(l)} \neg y_i) \quad (17)
\]
In (16), assigning false values to variables other than \( x_1^i, x_2^i, \ldots, x_{(l)}^i \) does not produce a clause 
\[
f_k^{\text{restricted}} (X_k) = (x_1^i \land x_2^i \land \ldots \land x_{(l)}^i )
\]
In (17), assigning false values to variables other than \( x_1^i, x_2^i, \ldots, x_{(l)}^i \) produces a clause 
\[
f_k^{\text{restricted}} (X_k) = (x_1^i \land x_2^i \land \ldots \land x_{(l)}^i )
\]
but assigning \( x_1 = x_2 = \ldots = x_{(l)} = 1, y_i = 1 (1 \leq i \leq l) \) to \( C_j \) returns 0, though \( C_j \) and \( f_k \) should return 1. 
Therefore neither (16) nor (17) follow. So \( C_j \) has exactly \( \binom{l}{2} \) variables in \( X_k \), and \( C_j = (x_1^i \land x_2^i \land \ldots \land x_{(l)}^i ) \). \( \Box \)

By theorem7, \( f_k \) can be expressed in the following form. 
\[
f_k (x_1, x_2, \ldots, x_{(l)} ) = \bigvee_{i=1}^{\binom{l}{2}} f_k^{\text{restricted}} (X_k) \bigvee_j C_j' \quad (18)
\]
It is necessary to specify the form of \( C_j' \). 

**Theorem 8.** \( C_j' \) has to contain \( \binom{l}{2} \) variables representing edges among \( k \) vertices: \( C_j' = C_j' (X_k, \ldots) \) for some \( i \)

**Proof.** If \( C_j' \) returns 0 for all the inputs, \( C_j' \) should be erased in \( f_k \). So it is necessary to consider the case where \( C_j' \) returns 1 for some input. If no \( \binom{l}{2} \) variables in \( C_j' \) represent edges among \( k \) vertices and \( C_j' \) returns 1 for some input,
adding double NOT gates such as $C$.

A combinatorial property $C$ cannot be erased when expressed in disjunctive normal form. Therefore the size of $f_k$ as a Boolean circuit is larger than $(\binom{n}{k} - 1) > (\frac{2}{3})^k - 1$. □

By theorem10, for $k$ in $3 < k < n^\tau$, this proves $P/\text{poly} \neq \text{NP}$ and $P \neq \text{NP}$.

3.1 Proof that this strategy is non-Naturalizable

In the paper [2], the proof is natural or naturalizable if it satisfies the following three conditions, constructivity, largeness, and usefulness.

"Formally, by a combinatorial property of Boolean functions we will mean a set of Boolean functions $\{C_n \subseteq F_n | n \in \omega\}$. Thus, a Boolean function $f_n$ will possess property $C_n$ if and only if $f_n \in C_n$. (Alternatively, we will sometimes find it convenient to use function notation: $C_n(f_n) = 1$ if $f_n \in C_n$; $C_n(f_n) = 0$ if $f_n \notin C_n$.) The combinatorial property $C_n$ is natural if it contains a subset $C_n^*$ with the following two conditions:

Constructivity. The predicate $f_n \in C_n^*$ is computable in P. Thus $C_n^*$ is computable in time which is polynomial in the truth table of $f_n$;

Largeness. $|C_n^*| \geq 2^{-O(n)}|F_n|$ A combinatorial property $C_n$ is useless against $P/\text{poly}$ if it satisfies:

Usefulness. The circuit size of any sequence of functions $f_1, f_2, \ldots, f_n, \ldots$, where $f_n \in C_n$, is super-polynomial; i.e., for any constant $k$, for sufficiently large $n$, the circuit size of $f_n$ is greater than $n^k$."

The proof strategy used in this paper is to specify the Boolean function $f_k$ as (26). Of course, a Boolean function

$$f_k(x_1, x_2, \ldots, x_{\binom{n}{k}}) = \bigvee_{i=1}^{\binom{n}{k}} f_k^{\text{restricted}}(X_i^k)$$

and not a clause in $f$ cannot be erased when expressed in the disjunctive normal form."

Boolean functions satisfying this property can be given by adding double NOT gates such as

$$f_k(x_1, x_2, \ldots, x_{\binom{n}{k}}) = \bigvee_{i=1}^{\binom{n}{k}} f_k^{\text{restricted}}(X_i^k) = (x_1^1 \land \ldots \land x_{\binom{n}{k}}^1) \bigvee_{i=1}^{\binom{n}{k}} f_k^{\text{restricted}}(X_i^k)$$

$$= ((-x_1^1) \lor (-x_2^1) \lor \ldots \lor (-x_{\binom{n}{k}}^1)) \bigvee_{i=1}^{\binom{n}{k}} f_k^{\text{restricted}}(X_i^k)$$

Therefore the total number of Boolean function satisfying this property is at most $2^{\binom{n}{k}}$. As the total number of Boolean functions with $n$ variables is $2^{2^n}$,

$$\frac{|C_n^*|}{2^n} = 2^{\binom{n}{k}} < 2^{-O(n)}$$

This violates the largeness condition in Natural proof. Therefore this strategy does not conflict with the widely believed conjecture on the existence of pseudorandom functions.

4. RELATIONSHIP TO OTHER OPEN PROBLEMS IN COMPUTATIONAL COMPLEXITY THEORY

As $P \neq \text{NP}$ and $NP \subseteq \text{PH} \subseteq \text{PSPACE}$, $P \neq \text{PH}$ and $P \neq \text{PSPACE}$. Among problems in $\text{NP}$, complexity classes of the integer factorization problem, the discrete logarithm problem and the graph isomorphism problem [10, 19, 27, 30] remain open for many years. It is already known that

(1) if the decision version of the integer factorization problem is in $\text{NP}$-complete, then $\text{NP}=\text{co-NP}$ and the polynomial hierarchy will collapse to its first level.

(2) if the graph isomorphism problem is in $\text{NP}$-complete, then the polynomial hierarchy will collapse to its second level.

As a collapse of polynomial hierarchy seems unlikely to happen under $P \neq \text{NP}$, they seem unlikely to be in $\text{NP}$-complete. Furthermore the circuit lower bounds of the integer factorization problem, the discrete logarithm problem and the graph isomorphism problem cannot be obtained by the proof strategy used in this paper, because

(1) the integer factorization problem and the discrete logarithm problem have neither global symmetry nor local symmetry to specify their structure.

(2) the global symmetry and local symmetry of graph isomorphism problem are hard to determine in general.

Whether or not $\text{NP}$-complete problems can be solved by quantum computers in polynomial time remains open.

5. CONCLUSIONS

By interpreting a Boolean circuit as a graph, the global symmetry and the local symmetry of variables in the circuit is discussed in this paper. The small number of global symmetry and the large number of local symmetry in the circuit which computes $f_k$ can establish the exponential circuit lower bound for a $\text{NP}$-complete problem, which means $P/\text{poly} \neq \text{NP}$ and $P \neq \text{NP}$.

Even if the same strategy is used, the computational complexity classes of the integer factorization problem, the discrete logarithm problem and the graph isomorphism problem remain open. Furthermore whether or not $\text{NP}$-complete
problems can be solved by quantum computers in polynomial time remain open.

As NP-complete problems turn out to be impossible to solve in polynomial time by a classical computer, heuristic approaches or algorithms for restricted types of inputs need to be developed for NP-complete problems.

6. REFERENCES

[1] A.A.Razborov. Lower bounds for the monotone complexity of some Boolean functions. Soviet Math. Dokl.31 (1985), 354-357; Dokl.Akad.Nauk SSSR 281, No.4(1985), 798-801.
[2] A.A.Razborov. Lower bounds of monotone complexity of the logical permanent function. Math. Notes Acad. Sci. USSR 37 (1985), 485-493; Mat. Zametki 37, No. 6(1985), 887-900.
[3] A.A.Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. Math. Notes Acad. Sci. USSR, 41 (1987), 333-338; Mat. Zametki 41, No. 4 (1987), 598-607.
[4] A.E.Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. Soviet Math. Dokl.31, No.3(1985), 530-534; Dokl.Akad. Nauk SSSR 282, No.5(1985), 1033-1037.
[5] A.E.Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of \( \pi \)-schemes. Moscow Univ. Math. Bull, 42(1):63–66, 1987.
[6] A.Hajnal, W.Maass, P.Pudlak, M.Szegedy, and G.Turan. Threshold circuits of bounded depth. In Proceedings, 28th Symposium on Foundations on Computer Science, pages 99–110, 1987.
[7] A. E. Andreev. On one method of obtaining effective lower bounds of monotone complexity. Algebra i Logika, 26(No.1):3–21, 1987.
[8] A.Razborov and S.Rudich. Natural proofs. Journal of Computer and System Sciences, 55(1):24–35, 1997.
[9] A.Yao. Separating the polynomial-time hierarchy by oracles. In Proceedings, 26th IEEE FOCS, pages 1–10, 1985.
[10] L. Babai, W. Kantor, and E. Luks. Computational complexity and the classification of finite simple groups. Proceedings of 24th FOCS, IEEE Computer Society, 25(1):162–171, November 1983.
[11] S. A. Cook. The complexity of theorem-proving procedures. Proceedings of the third annual ACM symposium on Theory of computing, pages 151–158, May 1971.
[12] S. A. Cook and D. G. Mitchell. Finding hard instances of the satisfiability problem: A survey. DIMACS Ser. Discr. Math. and Theor. Comp. Sci, 1997.
[13] D.A.Barrington. A Note on a Theorem of Razborov. Technical report, University of Massachusetts, 1986.
[14] J.Aspnes, R.Beigel, M.Furst, and S.Rudich. The expressive power of voting polynomials. Combinatorica, 14(2):135–148, 1994.
[15] J.Hästad. The shrinkage exponent is 2. in Proceedings of 34th IEEE FOCS 1993, 114-123, SIAM J.Comput.
[16] J.Hästad. Computational Limitations on Small Depth Circuits. Ph.D.thesis, Massachusetts Institute of Technology, 1986.
[17] R. M. Karp. Reducibility among combinatorial problems. In Complexity of Computer Computations, pages 85–103, 1972.
[18] E. Kranakis. Symmetry and Computational Complexity of Boolean Functions:Problems and Results, Graph Coloring and Symmetry, 2002.
[19] R. E. Ladner. On the structure of polynomial time reducibility. Journal of ACM (JACM), 22(1):155–171, January 1975.
[20] L. A. Levin. Universal sequential search problems. Problemy Peredachi Informatsii, 9(3):115–116, 1973.
[21] M.Ajtai. \( \Sigma_1 \)-formulae on finite structures. Ann. Pure Appl.Logic, 24:1–48, 1983.
[22] M.Furst, J.B.Saxe, and M.Sipser. Parity, circuits and the polynomial time hierarchy. Math.Syst.Theory, 17:13–27, 1984.
[23] M.Karchmer and A.Wigderson. Monotone circuits for connectivity require super-logarithmic depth. SIAM J.Disc. Math., 3(2):255–265, (May 1990).
[24] M.S.Paterson and U.Zwick. Shrinkage of de Morgan formulae under restriction. Random Structure and Algorithms, 4(2):135–150, 1993.
[25] N.Alon and R.Boppana. The monotone circuit complexity of Boolean functions. Combinatorica 7, No.1, pages 1–22, 1987.
[26] N.Nisan and R.Impagliazzo. The effect of random restrictions on formulae size. Random Structures and Algorithms, 4(2):121–134, 1993.
[27] R. C. Read and D. G. Corneil. The graph isomorphism disease. Journal of Graph Theory, 1(4):339–363, 1977.
[28] R.Raz and A.Wigderson. Monotone circuits for matching require linear depth. J.Assoc. Comput. Mach., 39:736–744, 1992.
[29] R.Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In Proceedings, 19th ACM Symposium on Theory of Computing, pages 77–82, 1987.
[30] U. Schöning. Graph isomorphism is in the low nonmonotone circuit complexity is exponential. Combinatorica, 8:141–142, 1988.