Journal of Global Optimization **7**: 279–295, 1995.

On the Longest Circuit in an Alterable Digraph*

KER-I KO** and CHIH-LONG LIN‡
Department of Computer Science, State University of New York at Stony Brook, Stony Brook, NY 11794, U.S.A.

(Received: 7 November 1994; accepted: 23 February 1995)

Abstract. An *alterable digraph* is a digraph with a subset of its edges marked alterable and their orientations left undecided. We say that an alterable digraph has an invariant of k on the length of the longest circuit if it has a circuit of length at least k regardless of the orientations over its alterable edges. Computing the maximum invariant on the length of the longest circuit in an alterable digraph is a *global optimization* problem. We show that it is hard to approximate the global optimal solution for the maximum invariant problem.

Key words: Alterable digraphs, global optimization, approximation, NP-hardness.

1. Introduction

We study the global optimization problems in dynamic environments modeled by *alterable digraphs*. An alterable digraph is a directed graph of which a subset of edges are marked alterable and have their orientations left undecided. An example of this kind of environment is the transportation system of a metropolitan area where one-way streets are dynamically directed to accommodate varying traffic requirements. Informally, an alterable digraph is a "succinct" description of a group of, potentially exponentially many, digraphs, and therefore testing whether a property Q is invariant over the group of digraphs would intuitively be more difficult than testing Q on a single digraph.

The specific problem to be studied in this paper is the *Longest Circuit* problem in alterable digraphs: given an alterable digraph G, find the maximum integer $lc(G)$ such that G has a simple circuit of length at least $lc(G)$, regardless of the orientations of the alterable edges of G. The longest circuit problem in *undirected graphss* is a well-known NP-hard problem and is closely related to the Travelling Salesman problem (TSP). Papadimitriou and Yannakakis showed in [11] that there exists a constant $c > 1$ such that the problem of *approximating* the optimal travelling salesman tour in a complete graph with edges of length one or two is NP-hard. Karger *et al.* [8] used this result to show that the problem of approximating the length of the *longest circuit* in an undirected graph to within *any* constant factor $c > 1$ is also NP-hard.

* Research supported in part by NSF grant CCR 9121472.
** email: keriko@sbcs.sunysb.edu.
‡ email: longlin@sbcs.sunysb.edu.

We study in this paper the effect of dynamic environment on the computational complexity of the longest circuit problem. Note that the value of $lc(G)$ for an alterable digraph G is required to remain the lower bound of the length of the longest circuit in G for all possible dynamic changes on the directions of alterable edges. This requirement corresponds to an extra level of nondeterminism, and potentially raises the complexity of the problem from NP to the second level of the polynomial-time hierarchy (PH) [12, 4] of which NP is the first level. Intuitively it is not hard to see that most *dynamic* optimization problems fall into this complexity category. Our main result here shows that *approximating* the value of $lc(G)$ is as hard as computing it *exactly*; specifically we show that there is a constant $c > 1$ such that the problem of computing a value $lc'(G)$ satisfying $lc(G)/c \leqslant lc'(G) \leqslant c \cdot lc(G)$ for any alterable digraph G is complete for the second level Π_2^P of PH. As a consequence, even if we have the access to an oracle capable of solving an NP-complete problem such as the Satisfiability problem, it is still hard to approximate the value of $lc(G)$ within a constant factor.

Our work continues the recent development on the intractability of approximating many NP-hard optimization problems including the problems MAX-CLIQUE and MIN-SET-COVER [2, 9], and of approximating some PSPACE-hard problems such as MAX-GEOGRAPHY [3]. These works suggest that for many intractable optimization problems, approximating the optimum solutions within a constant factor is essentially as hard as finding the exact optimum solutions. In [5, 7], this development has been extended to the second and higher levels of PH. Essentially it is shown in [5] that the approximation of MAX-3SAT$_2$ to within some constant factor $c > 1$ is Π_2^P-hard, where MAX-3SAT$_2$ is the following extension of the Maximum-Satisfiability problem: given a 3CNF boolean formula $F(X, Y)$ over two sets of variables X and Y, find the maximum integer k such that for any truth assignment t_x to X there exists a truth assignment t_y to Y satisfying at least k clauses of F. Similar results for other levels of PH are also shown.

For technical reasons, we introduce, in Section 3, a subproblem MAX-3SAT$_2$-B of MAX-3SAT$_2$ in which the number of occurrences of each boolean variable in a 3CNF input formula is bounded by some constant. We show that the approximation to this subproblem within a constant factor is also Π_2^P-hard. Our proof is a nontrivial extension of [10] where the problem MAX-3SAT-B, a subproblem of MAX-3SAT, is shown to be hard to approximate.

Finally we point out that our reductions use a more general notion of *gap-preserving reduction* than the *linear reduction* of [10]. This gap-preserving reduction is the most general type of reductions that preserve nonapproximability results and is necessary in our context. We present this notion formally in Section 2, along with the preliminary results in [5]. The main result is shown in Section 4.

2. Complexity of Approximation Problems

In this section, we review the notion of completeness in NP and Π_2^P, and define the notion of the gap-preserving reduction. We let Σ be the binary alphabet $\{0, 1\}$ and Σ^* be the set of finite strings over Σ. For any string x in Σ^*, we denote by $|x|$ the length of x. Let $\mathbf{Q}^+$ be the set of positive rationals and $\mathbf{R}^+$ the set of positive reals. In the following, we briefly review some basic complexity classes frequently mentioned in literature. For more details, the reader is referred to any standard text, for instance [4].

A decision problem A is just a language $A \subseteq \Sigma^*$. The class P is the class of decision problems that are solvable by deterministic Turing Machines (TMs) in polynomial time; that is, for each $A \in$ P, there exists a TM M_A such that for any $x \in \Sigma^*$, M_A on x halts in $p(|x|)$ steps for some polynomial $p > 0$, outputs 1 if $x \in A$, and 0 otherwise. The class NP is the class of decision problems solvable by nondeterministic TMs in polynomial time. It is easy to see that P $\subseteq$ NP. Whether P = NP is a major open question in complexity theory.

A decision problem A is *reducible* to a decision problem B if there is a polynomial-time computable function $f : \Sigma^* \to \Sigma^*$ such that for all $x \in \Sigma^*$, $x \in A$ if and only if $f(x) \in B$. For any complexity class $\mathcal{C}$, we say that the decision problem A is *$\mathcal{C}$-hard* if for any $B \in \mathcal{C}$, B is reducible to A; A is $\mathcal{C}$-complete if A is $\mathcal{C}$-hard and A is also in $\mathcal{C}$. The fact of A being NP-hard means that if $A \in$ P then P = NP, and so NP-hard problems are commonly considered *intractable*. A typical NP-complete problem is the Satisfiability problem (SAT) of deciding whether a given boolean formula is satisfiable. A corresponding optimization problem MAX-3SAT of finding the maximum number of satisfiable clauses in a given 3CNF boolean formula is known to be NP-hard. [4] contains hundreds of NP-complete problems.

Let $\langle x, y\rangle$ be any *pairing* function mapping strings x and y to a single string in polynomial time. A well-known characterization of the class NP is as follows: $A \in$ NP if and only if there exists some $B \in$ P such that for all $x \in \Sigma^*$,

$$x \in A \Leftrightarrow (\exists y, |y| \leqslant p(|x|))[\langle x, y\rangle \in B],$$

where $p(n)$ is some polynomial depending only on A. The complexity class Π_2^P is a natural extension of the class NP. We say that $A \in \Pi_2^P$ if there exists some $B \in$ P such that

$$x \in A \Leftrightarrow (\forall y, |y| \leqslant p(|x|))(\exists z, |z| \leqslant p(|x|))[\langle x, \langle y, z\rangle\rangle \in B].$$

It is obvious that NP $\subseteq \Pi_2^P$, and whether the inclusion is proper is also a major open question. A typical Π_2^P-complete problem, SAT_2, is that of deciding whether a given quantified boolean formula $(\forall x_1)\dots(\forall x_r)(\exists y_1)\dots(\exists y_s)\psi$ is true, where ψ is a quantifier free boolean formula over variables x_i's and y_j's. [6] contains more Π_2^P-complete problems.

We now formalize the notion of approximating *function evaluation problem* and its relation to the complexity of decision problems.

DEFINITION 1. Let $f, g : \Sigma^* \to \mathbf{Q}^+$ and $c : \mathbf{N} \to \mathbf{R}^+$, $c(n) > 1$ for all n, be given. We say that g approximates f to *within a factor of* c (c-approximates f in short) if for all $x \in \Sigma^*$, we have $f(x)/c(|x|) < g(x) < c(|x|) \cdot f(x)$. The *c-approximation problem* of f is to compute a function g that c-approximates f.

DEFINITION 2. Let $A, B \subseteq \Sigma^*$, $A \cap B = \emptyset$, and $\mathcal{C}$ be a decision problem class. We say $\langle A, B\rangle \in \mathcal{C} \times$ co-$\mathcal{C}$ if $A \in \mathcal{C}$ and $\bar{B} \in \mathcal{C}$. Given two pairs $\langle A, B\rangle$ and $\langle A', B'\rangle$ in $\mathcal{C} \times$ co-$\mathcal{C}$, we say that $\langle A, B\rangle$ is *G-reducible* to $\langle A', B'\rangle$ if there is a polynomial-time computable function f such that $f(A) \subseteq A'$ and $f(B) \subseteq B'$. We say that $\langle A, B\rangle$ is $\mathcal{C}$-hard if there exists a set C that is $\mathcal{C}$-hard and $\langle C, \bar{C}\rangle$ is G-reducible to $\langle A, B\rangle$.

The following definition relates the hardness of approximating function values to that of pairs of decision problems.

DEFINITION 3. Let $f : \Sigma^* \to \mathbf{Q}^+$ be a given function and $c : \mathbf{N} \to \mathbf{Q}^+$, $c(n) > 1$. We say that the c-approximation problem of f is $\mathcal{C}$-hard if there exist $s, l : N \to \mathbf{Q}^+$, $s(n) < l(n)$, such that

1. for all n, $c(n)s(n) < l(n)/c(n)$; and
2. $\langle\{x \mid f(x) \geqslant l(|x|)\}, \{x \mid f(x) \leqslant s(|x|)\}\rangle$ is $\mathcal{C}$-hard.

Given $s(n) < l(n)$, for the sake of simplicity, we shall write $\langle f : l, s\rangle$ for the pair of sets $\langle\{x \mid f(x) \geqslant l(|x|)\}, \{x \mid f(x) \leqslant s(|x|)\}\rangle$; further, we shall write only the constants for corresponding constant functions, e.g., 1 for $l(n) = 1$. The following proposition can be easily verified.

PROPOSITION 4. *Let $s < l$. If $\langle f : l, s\rangle$ is $\mathcal{C}$-hard and $\mathcal{C} \neq$ P, then the $(l/s)^{1/2}$-approximation problem of f is not polynomial-time computable.*

Boolean formulae will be very much involved in our reductions. Let u be a boolean variable; by a *literal* on u we mean u itself or its negation $\neg u$. For any boolean formula $F(U)$ over a set U of boolean variables, we say that $F(U)$ is in 3-conjunctive normal form (3CNF) if it is a conjunction of clauses and each clause is a disjunction of three literals over U. We say that *variable* u occurs in a clause C if C contains either u or $\neg u$ as one of its disjunct. A truth assignment t to a set of variable U is just a subset of U; for any $u \in U$, we say that t sets u to *true* (or *false*), written as $t(u) = 1$ (or, respectively $t(u) = 0$), if $u \in t$ (or, respectively, $u \notin t$). We let 2^U denote the set of truth assignments to U. For any $t \in 2^U$, we let $\#[F(t)]$ (and $\Pr[F(t)]$) denote the number (and, respectively, fraction) of satisfied clauses of F by t; for example, if $F(U) = (u_1 \vee u_2 \vee u_3) \wedge (\neg u_1 \vee \neg u_2 \vee \neg u_3)$ and $t = \emptyset$, then $\#[F(t)] = 1$ (and, respectively, $\Pr[F(t)] = 1/2$). Often the set U of variables is partitioned into disjoint sets, say X and Y, each dealt with differently; we then extend the above notations to, respectively, $F(X, Y)$, $\#[F(t_x, t_y)]$, and $\Pr[F(t_x, t_y)]$ for $t_x \in 2^X$ and $t_y \in 2^Y$. We call a variable in X an X-variable, and analogously call a literal defined on X an X-literal; Y-variables and Y-literals are defined similarly.

Our basis of reduction is the following standard Π_2^P-hard problem. It is proved in [5] that the corresponding approximation problem is also Π_2^P-hard.

MAX-3SAT$_2$
Input: A 3CNF boolean formula $F(X, Y)$ over two sets of variables X and Y.
Output: $f_{\text{MAX-3SAT}_2}(F) = \min_{t_x \in 2^X} \max_{t_y \in 2^Y} \Pr[F(t_x, t_y)]$.

PROPOSITION 5. [5]. $\langle f_{\text{MAX-3SAT}_2} : 1, 1 - \varepsilon\rangle$ *is* Π_2^P*-hard for some constant* $0 < \varepsilon < 1$.

In Section 3, we consider a subproblem MAX-3SAT$_2$-B of MAX-3SAT$_2$. Inputs to both problems are the same except that for MAX-3SAT$_2$-B the number of occurrences for each variable is bounded by some fixed constant b. The subscript "$_2$" is intended as a reminder of the *two* levels of optimization (min and max) involved in the definition above. Without the subscript, that is, MAX-3SAT and MAX-3SAT-B, we mean the versions of MAX-3SAT$_2$ and MAX-3SAT$_2$-B respectively with the restriction of $X = \emptyset$.

3. A Subproblem of Maximum Satisfiability$_2$

In this section we prove our main technical theorem, that is, MAX-3SAT$_2$-B is hard to approximate. The main theorem will be proved in two stages: given a 3CNF boolean formula $F(X, Y)$ for the MAX-3SAT$_2$ problem, we in the first stage consider instances having a constant bound on the number of occurrences for X-variables and then, in the second stage, consider those having a constant bound on the number of occurrences for both X- and Y-variables.

Papadimitriou and Yannakakis [10] used the fact that there exist polynomial-time constructible *expanders* of bounded degrees to show that MAX-3SAT is reducible to MAX-3SAT-B. In the first part of the proof, we will need an extended property of expanders. We first review the notion of the expanders. For any connected graph $G = (V, E)$ and any $u, v \in V$, let $dist_G(u, v)$ be the number of edges in a shortest path from u to v; for convenience, we let $dist_G(u, u) = 0$. Further, for any $S \subseteq V$, let $dist_G(u, S) = \min\{dist_G(u, v) \mid v \in S\}$.

DEFINITION 6. Let c be a constant with $0 < c < 1$. We call a graph $G = (V, E)$ a *c-expander* if for any subset S of V having at most $|V|/2$ vertices, $|\{u \in V \mid dist_G(u, S) = 1\}| \geqslant c|S|$.

We say a graph is of degree k if every vertex in G is of degree k.

LEMMA 7. *[1]. There exist a constant c, $0 < c < 1$, and an algorithm that, on input n, constructs a c-expander of size n and degree three in time polynomial in n.*

From the definition, we know that if G is a c-expander then for any subset $S \subseteq V$, with $|S| \leqslant |V|/2$, there exist sufficiently many vertices *outside* of S that are adjacent to some vertices inside S. The following lemma shows that G can be so augmented that a constant fraction of vertices *inside* of S are adjacent to some vertices outside of S.

LEMMA 8. *Let $0 < \delta < 1$ be any constant. Then for any n there exists a polynomial time constructible graph $\mathcal{G}[n, \delta] = (V, E)$ satisfying the following properties: (a) $|V| = n$, (b) $\mathcal{G}[n, \delta]$ is of degree b, where b is a constant depending only on δ, and (c) for any $S \subseteq V$, $|S| \leqslant n/2$,*

$$|\{u \in S \mid dist_{\mathcal{G}[n,\delta]}(u, V - S) = 1\}| > (1 - \delta)|S|.$$

Proof. Let $G' = (V, E')$ be a c-expander of size n constructed in Lemma 7. Let d be the least integer such that $(1 - c/3)^d < \delta$. Then we claim that $G = (V, E)$ with $E = \{\langle u, v\rangle \mid dist_{G'}(u, v) \leqslant d\}$ has the properties (a), (b) and (c). First, for property (b), we note that the degree of G is bounded by $b = 3^{d+1}$. Second, for property (c), we let $S \subseteq V$, $|S| \leqslant n/2$, and for each $i \geqslant 0$, define $S_i = \{u \in S \mid dist_{G'}(u, V - S) > i\}$. We argue that $|S_{i+1}| \leqslant (1 - c/3)|S_i|$. To see this, we observe that (1) by the definition, at least $c|S_i|$ vertices v of $V - S_i$ satisfy that $dist_{G'}(v, S_i) = 1$, and (2) at most three of them can be adjacent to a common vertex in S_i, since G' has degree 3. Therefore, at most $(1 - c/3)|S_i|$ vertices v of S_i satisfy that $dist_{G'}(v, V - S_i) \geqslant 2$, or equivalently, $dist_{G'}(v, V - S) > i + 1$. This shows that $|S_{i+1}| \leqslant (1 - c/3)|S_i|$ for each $i \geqslant 0$. Unwrap the recursion and we obtain $|S_i| \leqslant |S|(1 - c/3)^i$. Finally setting i to d satisfies property (c), and hence the claim. □

Now we are ready to prove the main result of this section. Let MAX-3SAT$_2$-XB be the version of MAX-3SAT$_2$-B without restricting the number of occurrences for each Y-variable; let $f_{\text{MAX-3SAT}_2\text{-XB}}$ be the correspondingly defined function.

LEMMA 9. *$\langle f_{\text{MAX-3SAT}_2\text{-XB}} : 1 - \varepsilon_1', 1 - \varepsilon_2'\rangle$ is Π_2^P-hard for some constants $0 < \varepsilon_1' < \varepsilon_2' < 1$.*

Proof. By Proposition 5, $\langle f_{\text{MAX-3SAT}_2} : 1, 1 - \varepsilon\rangle$ is Π_2^P-hard for some constant $0 < \varepsilon < 1$. We prove that $\langle f_{\text{MAX-3SAT}_2} : 1, 1 - \varepsilon\rangle$ is G-reducible to $\langle f_{\text{MAX-3SAT}_2\text{-XB}} : 1 - \varepsilon_1', 1 - \varepsilon_2'\rangle$ for some constants $0 < \varepsilon_1' < \varepsilon_2' < 1$; the actual values of ε_1' and ε_2' will be chosen later.

Let $F(X, Y)$ be a 3CNF boolean formula over two sets of variables X and Y. Assume that F has n clauses $C_1, \ldots, C_n$. Without loss of generality, we assume that no clause of F contains both v and its complement $\neg v$ for any variable $v \in X \cup Y$. We first construct a new boolean formula $F'(X', Y')$ from $F(X, Y)$; we note that it will only be in CNF, not in 3CNF as required. Converting $F'(X', Y')$ to an equivalent 3CNF formula is a routine task and is deferred till the end of the proof.

Construction. Assume that $X = \{x_1, \dots, x_r\}$. For each j, $1 \leqslant j \leqslant r$, let $d(j)$ be the number of clauses of $F(X,Y)$ that contain either x_j or $\neg x_j$, and let $C_{i_1}, C_{i_2}, \dots, C_{i_{d(j)}}$ be those clauses. Then, we define $2d(j)$ new variables and group them as $X_j = \{x_j^{i_1}, \dots, x_j^{i_{d(j)}}\}$ and $U_j = \{u_j^{i_1}, \dots, u_j^{i_{d(j)}}\}$; variables in X_j are called the *occurrence variables* of x_j. The two sets of variables in F' are $X' = X_1 \cup \dots \cup X_r$ and $Y' = Y \cup U_1 \cup \dots \cup U_r$.

For each j, $1 \leqslant j \leqslant r$, define a graph $G_j = \mathcal{G}[d(j), \varepsilon/3]$ and match arbitrarily $x_j^{i_1}, \dots, x_j^{i_{d(j)}}$ with the $d(j)$ vertices of G_j. We say that two occurrence variables x_j^i and $x_j^{i'}$ are *adjacent* if their corresponding vertices in G_j are adjacent. The clauses of F' are divided into the following two groups:

(1) *Major clause* C_i', for each $1 \leqslant i \leqslant n$: Each C_i' is obtained from C_i as follows: if x_j appears in C_i, then replace the occurrence of x_j by x_j^i and add an extra literal u_j^i. For example, let $C_i = (x_1 \vee \neg x_2 \vee y_3)$ and $C_j = (\neg x_1 \vee x_2 \vee y_3)$ be two clauses of $F(X,Y)$. Then $C_i' = (x_1^i \vee u_1^i \vee \neg x_2^i \vee u_2^i \vee y_3)$ and $C_j' = (\neg x_1^j \vee u_1^j \vee x_2^j \vee u_2^j \vee y_3)$. Note that Y-variables are not affected, and if C_i contains no X-variables, then $C_i' = C_i$.

(2) *Discrepancy-test clause* D_j^i, for each occurrence variable x_j^i: Assume that $x_j^{i_1'}, x_j^{i_2'}, \dots, x_j^{i_b'}$ are the b occurrence variables of x_j adjacent to x_j^i. Then, let $D_j^i = (\neg u_j^i \vee x_j^i \vee x_j^{i_1'} \vee x_j^{i_2'} \vee \dots \vee x_j^{i_b'})$ if C_i' contains literal x_j^i, and let $D_j^i = (\neg u_j^i \vee \neg x_j^i \vee \neg x_j^{i_1'} \vee \neg x_j^{i_2'} \vee \dots \vee \neg x_j^{i_b'})$ if C_i' contains literal $\neg x_j^i$. Thus for the above example, the discrepancy-test clause D_1^i will be of the first form, and D_2^i of the second form.

We note that these clauses are set up to have the following properties: First, if the truth value assigned to x_j^i is inconsistent with that to any of its neighbours in G_j, then the discrepancy-test D_j^i corresponding to x_j^i will be satisfied, regardless of the truth value of u_j^i. Thus, u_j^i can be set to *true* and the major clause C_i' is satisfied regardless of the truth values assigned to other liberals in C_i'. On the other hand, if the truth values assigned to x_j^i and any of its adjacent variables are consistent, then u_j^i has to be set to false to satisfy the discrepancy-test clause D_j^i, and thus has no effect on satisfying C_i'.

The above completes the construction of $F'(X',Y')$. It can be seen that F' has the following properties: each X-variable occurs only a constant number of times (say, two times the degree of $\mathcal{G}[m, \varepsilon/3]$, which depends only on ε), each clause contains only $O(1)$ literals, and there are n major and at most $3n$ discrepancy-test clauses in F', since there are $\sum_{j=1}^{r} d(j) \leqslant 3n$ occurrence variables. By adding dummy clauses, we can assume that F' has exactly $3n$ discrepancy-test clauses.

Correctness. To see that the reduction is indeed a G-reduction, we first exhibit an *error-confinement* property of the reduction; namely, for any $t_x \in 2^{X'}$ and

$t_y \in 2^Y$, we can always find $t_u \in 2^U$ such that

$$\begin{aligned} \#[F'(t_x, t_y \cup t_u)] &= \max_{s \in 2^U} \#[F'(t_x, t_y \cup s)], \quad \text{and} \\ \#[D^i_j(t_x, t_u)] &= 1 \text{ for all discrepancy-test clause } D^i_j. \end{aligned} \tag{1}$$

To see this, let $s_1 \in 2^U$ be such that $\#[F'(t_x, t_y \cup s_1)] = \max_{s \in 2^U} \#[F'(t_x, t_y \cup s)]$ but $\#[D^i_j(t_x, s_1)] = 0$ for some discrepancy-test clause D^i_j. Since D^i_j is either of the form $(\neg u^i_j \vee x^i_j \vee \cdots)$ or $(\neg u^i_j \vee \neg x^i_j \vee \cdots)$, we must have $s_1(u^i_j) = 1$. Now define $s_2 \in 2^U$ by $s_2(u) = s_1(u)$ for all $u \in U - \{u^i_j\}$, and let $s_2(u^i_j) = 0$. We claim that $\#[F'(t_x, t_y \cup s_2)] \geqslant \#[F'(t_x, t_y \cup s_1)]$. To justify, first we note that variable u^i_j only occurs in C'_i and D^i_j, and D^i_j contains no Y-variables. Let $\Delta_1 = \#[C'_i(t_x, t_y \cup s_2)] - \#[C'_i(t_x, t_y \cup s_1)]$ and $\Delta_2 = \#[D^i_j(t_x, s_2) - \#[D^i_j(t_x, s_2)]$. It is clear that $\#[F'(t_x, t_y \cup s_2)] - \#[F'(t_x, t_y \cup s_1)] = \Delta_1 + \Delta_2$. Since $\Delta_2 = 1$ and $\Delta_1 \geqslant -1$, the claim follows. We have just shown how to reduce by one the number of unsatisfied discrepancy-test clause without decreasing the total number of satisfied clauses in F'. Repeat this process and we can find a $t_u \in 2^U$ satisfying the error-confinement property.

REMARK 10. We summarize the following properties of the formula $F'(X', Y \cup U)$:

(1) $F'(X', Y \cup U)$ consists of the major clauses and the discrepancy-test clauses.
(2) Each X'- and U-literal occurs at most once in the major clauses and at most some constant number of times in the discrepancy-test clauses.
(3) Each Y-literal occurs only in the major clauses, and the number of its occurrences in F' is the same as that in F.
(4) For any $t_x \in 2^{X'}$, there exist $t_y \in 2^Y$ and $t_u \in 2^U$ such that the two equalities in equation (1) holds for F'. (In other words, we can have all the errors, i.e., unsatisfied clauses, occur only in the major clauses.) □

Let $m = \min_{s_x \in 2^X} \max_{s_y \in 2^Y} \#[F(s_x, s_y)]$ and $m' = \min_{s'_x \in 2^{X'}} \max_{\substack{s_y \in 2^Y \\ s_u \in 2^U}} \#[F(s'_x, s_y \cup s_u)]$. We claim that

$$m' \geqslant 3n + m - \varepsilon n/2, \quad \text{and} \tag{2}$$

$$m' \leqslant 3n + m. \tag{3}$$

First, we prove inequality (2). Consider any $t \in 2^{X'}$. Let $P(x_j) = \{x \in X_j \mid t(x) = 1\}$ and $N(x_j) = \{x \in X_j \mid t(x) = 0\}$. Define $t_x \in 2^X$ as follows: for any $1 \leqslant j \leqslant r$,

$$t_x(x_j) = \begin{cases} 1 & \text{if } |P(x_j)| \geqslant |N(x_j)|, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

By the definition of m, there must exist a $t_y \in 2^Y$ such that $\#[F(t_x, t_y)] \geqslant m$. By (1), we can always find a $t_u \in 2^U$ such that $\#[F'(t, t_y \cup t_u)] = \max_{s \in 2^U} \#[F'(t, t_y \cup s)]$ and that all $3n$ discrepancy-test clauses are satisfied by t and t_u; that is, by t, t_y and t_u, all unsatisfied clauses are confined to among major clauses. Let $\Delta_i = \#[C_i(t_x, t_y)] - \#[C'_i(t, t_y \cup t_u)]$. It is then clear that $\#[F'(t, t_y \cup t_u)] \geqslant 3n + m - \sum_{i=1}^{n} \Delta_i$. We will show that $\sum_{i=1}^{n} \Delta_i \leqslant \varepsilon n/2$ and inequality (2) follows.

Let us call an occurrence variable x_j^i *consistent* if $t(x_j^i) = t_x(x_j)$, and call x_j^i *frontal* if $t(x_j^i) \neq t(x_j^{i'})$ for some $x_j^{i'}$ adjacent to x_j^i. If C'_i contains a frontal occurrence variable x_j^i, then D_j^i, either of the form $(\neg u_j^i \vee x_j^i \vee x_j^{i'_1} \vee x_j^{i'_2} \vee \cdots \vee x_j^{i'_b})$ or $(\neg u_j^i \vee \neg x_j^i \vee \neg x_j^{i'_1} \vee \neg x_j^{i'_2} \vee \cdots \vee \neg x_j^{i'_b})$, is always satisfied by t, regardless of the truth value set to u_j^i; choose $t_u(u_j^i) = 1$ and then $C'_i = (\cdots \vee u_j^i \vee \cdots)$ is satisfied, leading us to $\Delta_i \leqslant 0$. If all occurrence variables x_j^i of C'_i are consistent, then it is clear that $\Delta_i \leqslant 0$ also. Therefore, $\Delta_i > 0$ only if C'_i contains an inconsistent, non-frontal occurrence variable. The number of such major clauses, as we will argue below, is at most $\varepsilon n/2$.

Let S_j be the set of major clauses containing some occurrence variable x_j^i of X_j. By Lemma 8, at most

$$(\varepsilon/3)\min\{|P(x_j)|, |N(x_j)|\} \leqslant (\varepsilon/6)|S_j|$$

clauses in S_j may contain inconsistent, non-frontal occurrence variable in X_j. In total, there are at most $\sum_{j=1}^{r}(\varepsilon/6)|S_j| = (\varepsilon/6)\sum_{j=1}^{r}|S_j|$ non-frontal occurrence variables. Since each major clause C'_i contains at most three X'-variables, we must have $|\{j \mid C'_i \in S_j\}| \leqslant 3$ for each i, $1 \leqslant i \leqslant n$. Therefore

$$\sum_{j=1}^{r}|S_j| = \sum_{i=1}^{n}|\{j \mid C'_i \in S_j\}| \leqslant 3n.$$

It follows that the number of major clauses containing inconsistent, non-frontal occurrence variables is at most $\varepsilon n/2$. Therefore, we have $\sum_{i=1}^{n} \Delta_i \leqslant \varepsilon n/2$, and inequality (2) follows.

Next, we show inequality (3). Let t_x witness that $m = \max_{s_y} \#[F(t_x, s_y)]$. Then define $t \in 2^{X'}$ by $t(x_j^i) = t_x(x_j)$ for all $1 \leqslant j \leqslant r$ and all occurrence variables x_j^i's of x_j. Let $t_y \in 2^Y$ and $t_u \in 2^U$ be such that

$$\#[F'(t, t_y \cup t_u)] = \max_{\substack{s_y \in 2^Y \\ s_u \in 2^U}} \#[F'(t, s_y \cup s_u)].$$

By (1), we can assume that t_u is so chosen that all $3n$ discrepancy-test clauses are satisfied by t and t_u. Let $\Delta_i = \#[C_i(t_x, t_y)] - \#[C'_i(t, t_y \cup t_u)]$, and we have $\#[F'(t, t_y \cup t_u)] = 3n + m - \sum_{i=1}^{n} \Delta_i$. We claim that $\Delta_i \geqslant 0$ for all $1 \leqslant i \leqslant n$, from which inequality (3) follows.

Suppose otherwise and let $\Delta_i < 0$ for some $1 \leqslant i \leqslant n$. Then $\#[C_i(t_x, t_y)] = 0$ and $\#[C_i'(t, t_y \cup t_u)] = 1$, and $t_u(u_j^i) = 1$ for some j such that either x_j^i or $\neg x_j^i$ occurs in C_i'. We consider only the former case; the argument for the latter is similar. Let $C_i' = (x_j^i \vee \cdots)$ and $D_j^i = (\neg u_j^i \vee x_j^i \vee \cdots)$; since $\#[C_i(t_x, t_y)] = 0$, we have $t_x(x_j) = t(x_j^i) = 0$. Since t is consistent, all occurrence variables of x_j are set to 0 and hence D_j^i is reduced to a single literal $\neg u_j^i$. But then $t_u(u_j^i) = 1$ and therefore $\#[D_j^i(t, t_u)] = 0$, contradicting equation (1).

Finally we transform F' to a 3CNF formula F''. To do this, we observe that there is a mapping that transforms a clause of B literals into a set of $O(B)$ 3-literal clauses with $O(B)$ additional variables. Further, if the original clause is satisfiable, so are the derived set of 3-literal clauses as a whole; if the original is not satisfiable, all but one of the new clauses are simultaneously satisfiable (see [4] for the NP-completeness of 3SAT). Since the clauses of F' are of bounded length, after the transformation, F' still has at most kn 3-literal clauses for some constant $k > 4$. By adding dummy clauses, we may assume that F'' has exactly kn 3-literal clauses for some constant k.

REMARK 11. We note that the properties mentioned in Remark 10 of F' are preserved by this transformation. That is, if we keep the name "major clauses" ("discrepancy-test clauses") for clauses in F'' that are generated from a major clause (a discrepancy-test clause, respectively) in F', then properties (1), (2) and (3) still hold for F''. In addition, let Z be the set of new variables introduced by this transformation. Then we can rephrase properties (2) and (4) as follows:

(2′) Each X'-, U- and Z-literal occurs at most once in the major clauses and at most some constant number of times in the discrepancy-test clauses.

(4′) For any $t_x \in 2^{X'}$, there exist $t_y \in 2^Y$ and $t_u \in 2^{U \cup Z}$ such that equation (1) holds for F''. □

Let $m'' = \min_{s_x} \max_{s_y} \#[F''(s_x, s_y)]$. From inequalities (2) and (3), we have

(a) if $m = n$, then $m' \geqslant 4n - \varepsilon n/2$, and so $m'' \geqslant kn - \varepsilon n/2 = (1 - \varepsilon/2k)kn$, and

(b) if $m \leqslant (1 - \varepsilon)n$, then $m' \leqslant 4n - \varepsilon n$ and $m'' \leqslant kn - \varepsilon n = (1 - \varepsilon/k)kn$.

Therefore set $\varepsilon_1' = \varepsilon/2k$ and $\varepsilon_2' = \varepsilon/k$ and the theorem is proven. □

With Lemma 9 proven, we can now apply the idea of the reduction of [10] to show that MAX-3SAT$_2$-B is hard to approximate. Their reduction, called the *linear reduction* or simply the *L-reduction*, is a restricted version of the G-reduction. More precisely, they proved that there exist a polynomial-time computable function f and a constant $\alpha > 1$ such that

(i) for each instance $H(U)$ of MAX-3SAT with m clauses, $f(H(U)) = H'(U')$ is a 3CNF boolean formula with $(\alpha + 1)m$ clauses such that each variable in U' occurs at most a constant number of times, and

(ii) $\max_{t' \in 2^{U'}} \#[H'(t')] = \alpha m + \max_{t \in 2^U} \#[H(t)]$.

THEOREM 12. $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1-\varepsilon_1, 1-\varepsilon_2\rangle$ *is* Π_2^P*-hard for some constants* $0<\varepsilon_1<\varepsilon_2<1$.

Proof. We prove that $\langle f_{\text{MAX-3SAT}_2\text{-XB}} : 1-\varepsilon_1', 1-\varepsilon_2'\rangle$ is G-reducible to $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1-\varepsilon_1, 1-\varepsilon_2\rangle$ for some ε_1 and ε_2 to be chosen later.

Let $F(X,Y)$ be a 3CNF boolean formula over two sets of variables X and Y. Assume that F has n clauses and that the number of occurrences of each X-variable is bounded by some constant. Recall that $f_{\text{MAX-3SAT}_2}(F) = \min_{s_x}\max_{s_y}\#[F(s_x,s_y)]$. Let f and α be the function and the constant satisfying properties (i) and (ii) above. Define $h(F(X,Y)) = f(F(X,Y))$, treating X-variables as constants. Assume that $h(F(X,Y)) = F'(X,Y')$. It is then clear that

$$\begin{aligned}\min_{s_x\in 2^X}\max_{s_y'\in 2^{Y'}}\#[F'(s_x,s_y')] &= \min_{s_x\in 2^X}\left\{\alpha n + \max_{s_y\in 2^Y}\#[F(s_x,s_y)]\right\}\\ &= \alpha n + \min_{s_x\in 2^X}\max_{s_y\in 2^Y}\#[F(s_x,s_y)].\end{aligned}$$

Now choose $\varepsilon_1 = \varepsilon_1'/(\alpha+1)$ and $\varepsilon_2 = \varepsilon_2'/(\alpha+1)$ and we can see that

(a) if $f_{\text{MAX-3SAT}_2\text{-XB}}(F) \geqslant (1-\varepsilon_1')n$ then $f_{\text{MAX-3SAT}_2\text{-B}}(h(F)) \geqslant (\alpha+1-\varepsilon_1')n = (1-\varepsilon_1)(\alpha+1)n$, and

(b) if $f_{\text{MAX-3SAT}_2\text{-XB}}(F) \leqslant (1-\varepsilon_2')n$ then $f_{\text{MAX-3SAT}_2\text{-B}}(h(F)) \leqslant (\alpha+1-\varepsilon_2')n = (1-\varepsilon_2)(\alpha+1)n$.

Note that $h(F)$ has $(\alpha+1)n$ clauses and this completes the proof. □

REMARK 13. The proof of Theorem 12 preserves a similar error-confinement property to $(4')$ of Remark 11: Assume that $F(X,Y)$ is a 3CNF formula satisfying properties $(1), (2'), (3)$, and $(4')$ (here, U and Z variables are considered as part of Y). Then after applying h of Theorem 12 to F, the transformed formula $h(F(X,Y)) = F_1(X,Y_1)$ satisfies the following properties:

(1) F_1 consists of the major clauses and the discrepancy-test clauses (which now include the discrepancy-test clauses of F and the ones generated by h); and the number of discrepancy-test clauses is at most some constant times that of the major clauses.

(2) Each X- and Y_1-literal occurs at most once in the major clauses and at most some constant number of times in the discrepancy-test clauses.

(3) For any $t_x \in 2^X$, there exists $t_y \in 2^{Y_1}$ such that $\#[F_1(t_x,t_y)] = \max_{s_y\in 2^Y}\#[F_1(t_x,s_y)]$, and that all discrepancy-test clauses of F_1 are satisfied by t_x and t_y. □

We will use these properties in Section 4.

4. Main Result

We first review some definitions on directed graphs. A digraph is a tuple $G=(V,A)$, where V is a set of vertices and $A\subseteq V\times V$ is the set of directed edges.

For convenience, we often write $|G|$ for $|V|$; we will say that G contains $u \to v$ if $u, v \in V$ and $\langle u, v\rangle \in A$. A *circuit* in G is a sequence of vertices $v_1, \ldots, v_l$ such that $\langle v_i, v_{i+1}\rangle \in A$ for $1 \leqslant i < l$, and $v_i \neq v_j$ for all $1 \leqslant i \neq j \leqslant l$, except that $v_1 = v_l$.

An alterable digraph is a pair $\langle G, S\rangle$, where $G = (V, A)$ is a digraph and $S \subseteq A$. Given $S' \subseteq S$, we let $G(S')$ be the digraph induced by reversing the orientations of the edges in S'. We note that although such a process potentially induces multi-graphs, our construction in the sequel ensures that such does not happen.

Recall that for any alterable digraph G, $lc(G)$ denotes the maximum integer k such that G has a simple circuit of length at least k, regardless of the orientations of the alterable edges of G. We will prove that approximating $lc(G)$ for any alterable digraph G to within some constant factor is as hard as solving any Π_2^P-complete problems and therefore fix its complexity at exactly the second level of PH.

THEOREM 14. *$\langle lc(G) : (1-\varepsilon_3)|G|, (1-\varepsilon_4)|G|\rangle$ is Π_2^P-hard for some constants $0 < \varepsilon_3 < \varepsilon_4 < 1$.*

Proof. We will show that $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1-\varepsilon_1, 1-\varepsilon_2\rangle$ is G-reducible to $\langle lc(G) : (1-\varepsilon_3)|G|, (1-\varepsilon_4)|G|\rangle$; the actual values of ε_3 and ε_4 will be determined later. As indicated in Remark 13, we can restrict input instance $F(X, Y)$ for the MAX-3SAT$_2$-B problem to those satisfying the following requirements:

(a) F is the conjunction of two 3CNF boolean formulae F_M and F_D having m and $n = O(m)$ clauses respectively, and each *literal* occurs at most once in F_M and at most B times in F_D for some constant B.

(b) For any truth assignment t_x to X, there always exists a truth assignment t_y to Y such that $\#[F(t_x, t_y)] = \max_{s_y \in 2^Y} \#[F(t_x, s_y)]$, and that $\Pr[F_D(t_x, t_y)] = 1$.

Let $F_M(X, Y) = C_1 \wedge \ldots \wedge C_m$ and $F_D(X, Y) = D_1 \wedge \ldots \wedge D_n$, where $X = \{x_1, \ldots, x_r\}$ and $Y = \{y_1, \ldots, y_s\}$, and C_i and D_j are 3-literal clauses over X and Y. We construct a digraph $G_F = (V_F, A_F)$ and $S \subseteq A_F$ as follows. Let "$u \rightleftharpoons v$" denote the digraph $(\{u, v\}, \{\langle u, v\rangle, \langle v, u\rangle\})$. G_F contains two groups of subgraphs.

(1) For each variable $v \in X \cup Y$, we have a *variable digraph* for v containing the following components: $v^* \to v$, $v^* \to \overline{v}$, $v \to v[1] \rightleftharpoons v[2] \rightleftharpoons \cdots \rightleftharpoons v[B] \rightleftharpoons \overline{v}$, and $v[1] \to v$ if $v \in Y$. We call v and $\overline{v}$ *boundary* vertices. Figure 1(a) shows the variable digraph for x_i; the symbol "$\times$" is to be explained later.

(2) For each clause C_i, there is a *clause digraph* consisting of a single vertex c_i; for each clause D_j, there is a clause digraph d_j of $6B + 6$ vertices. Each literal v (or $\neg v$) occurring in D_j corresponds to a path of length $2B + 2$ in d_j, and we refer to this path by v^j (or, respectively, $\overline{v}^j$); the first vertex on this path is labeled by $v^j[0]$ and the last by $v^j[1]$ (or, respectively, $\overline{v}^j[0]$ and $\overline{v}^j[1]$), and these are boundary vertices for d_j. We show in Figure 1(b) a clause digraph d_j and a path x_i^j (enclosed by dotted box) corresponding to the literal x_i in D_j. (In Figure 1(b), we use $\multimap$ to represent a partial path of length $B - 1$.) In addition to these paths,

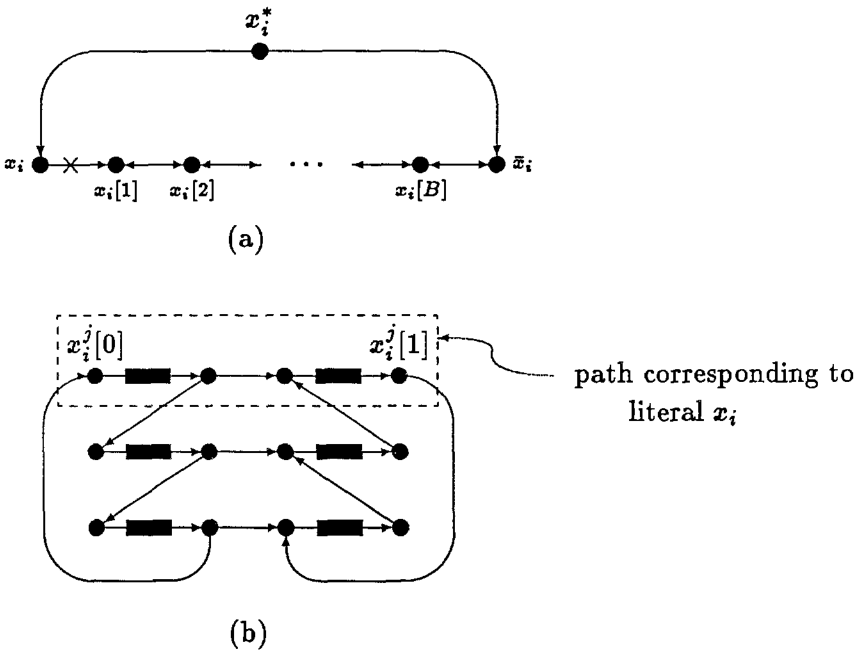

Fig. 1. Digraphs for variable x_i and clause D_j.

we also have in d_j three pairs of complementary *inter-literal* edges, as shown in Figure 1(b) by slanting and circled arrows. It is easy to verify that in order for a circuit h to visit d_j completely, h can pass d_j either one, two or three times, each time entering at some $v^j[0]$ and leaving via $v^j[1]$ for some literal $v \in D_j$, utilizing inter-literal edges to visit vertices on other paths if necessary.

These component digraphs are then interconnected by the following inter-component edges through boundary vertices: For each literal v occurring in clauses $D_{j_1}, D_{j_2}, \ldots, D_{j_b}$ and C_i, with $1 \leqslant j_1 < j_2 < \cdots < j_b \leqslant n$ and $1 \leqslant i \leqslant m$, we add the following edges:

(i) $v \to v^{j_1}[0]$,
(ii) $v^{j_k}[1] \to v^{j_{k+1}}[0], \quad 1 \leqslant k \leqslant b-1$,
(iii) $v^{j_b}[1] \to c_i$, and
(iv) $c_i \to u^*$, for all $u \in X \cup Y$.

If v does not occur in any C_i (or, instead, in any D_j), then we replace (iii) and (iv) by $v^{j_b}[1] \to u^*$ (or, respectively, replace (i), (ii), and (iii) by $v \to c_i$). For each literal $\neg v$, we do the same (i.e., replace the symbol v in the above by $\overline{v}$). For convenience, we shall call the inter-component edge $v \to v^{j_1}[0]$ the *positive* edge for v and $\overline{v} \to \overline{v}^{j_1'}[0]$ the *negative* edge for v. Furthermore, for any literal v, it is obvious that the above inter-component edges connect the paths $v^{j_1}, v^{j_2}, \ldots, v^{j_b}$

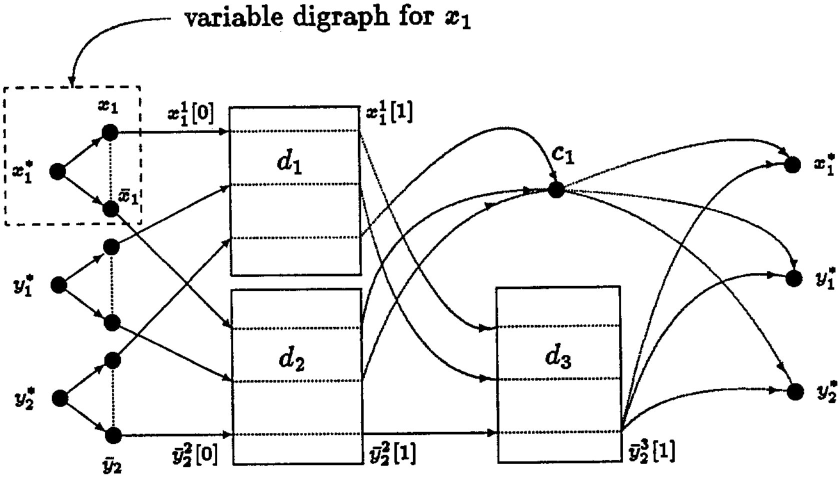

Fig. 2. Interconnections among component digraphs.

into a unique path from v to c_j;[1] we call this path $p(v)$. For each literal $\neg v$, the path $p(\overline{v})$ can be defined analogously. Shown in Figure 2 is the alterable digraph for $F(x_1, y_1, y_2) = C_1 \wedge D_1 \wedge D_2 \wedge D_3$, where $C_1 = (\neg x_1 \vee \neg y_1 \vee y_2)$, $D_1 = (x_1 \vee y_1 \vee y_2)$, $D_2 = (\neg x_1 \vee \neg y_1 \vee \neg y_2)$, $D_3 = (x_1 \vee y_1 \vee \neg y_2)$. To avoid clutter, some of the edges (e.g., $x_1^3[1] \to x_1^*$) and vertex-labels (e.g., y_2) are omitted. Also, x_1^*, y_1^* and y_2^* are duplicated to facilitate drawing.

We finish the construction by defining the set of alterable edges to be $S = \{\langle x_i, x_i[1]\rangle | x_i \in X\}$. In Figure 1(a), for example, the edge marked by "×" is alterable. The total number of vertices, $|V|$, is $O(mB)$: there are m clause digraphs c_i's, each of size one; $n = O(m)$ clause digraphs d_j's, each of size $O(B)$; and $O(m)$ variable digraphs, each of size $O(B)$.

We now show that the construction is indeed a G-reduction. First, we say that a subset $S' \subseteq S$ of alterable edges and a truth assignment $t_x \in 2^X$ are *consistent* if for all $x_i \in X, t_x(x_i) = 1$ if and only if $\langle x_i, x_i[1]\rangle \in S'$, i.e., $\langle x_i[1], x_i\rangle$ is in $G_F(S')$. We consider two cases.

Case I. Suppose that for any $t_x \in 2^X$, we have $\max_{s_y \in 2^Y} \#[F(t_x, s_y)] \geqslant (1 - \varepsilon_1)(m + n)$. Then we will show that for any $S' \subseteq S$, there exists a simple circuit in $G_F(S')$ that misses at most $\varepsilon_1(m + n)$ vertices in G_F.

More specifically, let $t_{S'} \in 2^X$ be consistent with S'. By our requirement (b) on F, we may assume that there exists $t_y \in 2^Y$ such that $\#[F(t_{S'}, t_y)] = \max_{s_y \in 2^Y} \#[F(t_{S'}, s_y)]$ and $\#[F_D(t_{S'}, t_y)] = n$, i.e., all unsatisfied clauses are among C_i's. We exhibit a simple circuit $H(t_{S'}, t_y)$, called the *standard traversal* by $t_{S'}$ and t_y, in $G_F(S')$, which misses at most $\varepsilon_1(m + n)$ vertices of G_F.

[1] We omit the case in which v does not occur in any C_i, and the case in which v does not occur in any D_j. In both cases, the path can be similarly defined.

Let $t = t_{S'} \cup t_y$ (recall that truth assignments are simply subsets of boolean variables). We first define a circuit h in $G_F(S')$ as follows, and later "expand" it to $H(t_{S'}, t_y)$. The circuit h consists of $r + s$ sub-paths, denoted by $h(v)$ for each $v \in X \cup Y$. If $t(v) = 1$, let $h'(v)$ be the path $v^* \to \overline{v} \to v[B] \to v[B-1] \to \cdots \to v[1] \to v$, and define $h(v) = h'(v) \cup p(v)$. If $t(v) = 0$, let $h'(v)$ be the path $v^* \to v \to v[1] \to v[2] \to \cdots \to v[B] \to \overline{v}$, and define $h(v) = h'(v) \cup p(\overline{v})$.

Note that the last vertex of each sub-path is always connected to v^* for any $v \in X \cup Y$, and therefore we can concatenate $h(x_1)$ through $h(y_s)$ into a simple circuit h. This completes the definition of h. It can be seen that h *visits* all variable digraphs and all clause digraphs to which the corresponding clauses are satisfied by t. Now for each clause digraph d_j visited by h fewer than three times, we expand h by using inter-literal edges so that all vertices in d_j are covered; let $H(t_{S'}, t_y)$ be the simple circuit so obtained from h. We can see that if h visits d_j, then $H(t_{S'}, t_y)$ visits all vertices of d_j. Since at most $\varepsilon_1(m+n)$ clauses of F_M are not satisfied by t, h then fails to visit at most $\varepsilon_1(m+n)$ clause digraphs c_i's, and so does $H(t_{S'}, t_y)$. As each c_i consists only of one vertex, *Case I* is proved.

Case II. Suppose that there exists some $t_x \in 2^X$ such that $k(t_x) = \max_{s_y \in 2^Y} \#[F(t_x, s_y)] \leqslant (1-\varepsilon_2)(m+n)$. Fix such a t_x, and let $S_{t_x} \subseteq S$ be consistent with t_x. We will show that any simple circuit in $G_F(S_{t_x})$ misses at least $\varepsilon_2(m+n)$ vertices.

By requirement (b), we can find $t_y \in 2^Y$ such that $\#[F(t_x, t_y)] = k(t_x)$ and $\Pr[F_D(t_x, t_y)] = 1$. Let h_0 be the standard traversal by t_x and t_y in $G_F(S_{t_x})$ as defined in *Case I*. Then h_0 misses $k_0 \geqslant \varepsilon_2(m+n)$ vertices, all among c_i's. We will show in the following that any simple circuit h in $G_F(S_{t_x})$ misses at least as many vertices as h_0 does.

Let t_h be the truth assignment corresponding to h; that is, for all $v \in X \cup Y$, $t_h(v) = 1$ if and only if the positive edge for v is in h. Let $t_{h,x} = t_h \cap X$, and $t_{h,y} = t_h \cap Y$. Let δ_h be the number of $x_i \in X$ such that $t_x(x_i) \neq t_{h,x}(x_i)$. We claim that for any $s_y \in 2^Y$,

$$\#[F(t_{h,x}, s_y)] \leqslant \#[F(t_x, t_y)] + \delta_h(B+1). \tag{4}$$

To justify the claim, we notice that each X-literal x_i or $\neg x_i$ occurs in at most $B+1$ clauses. So,

$$\#[F(t_{h,x}, s_y)] \leqslant \#[F(t_x, s_y)] + \delta_h(B+1).$$

The claim follows immediately from requirement (b) that $\#[F(t_x, s_y)] \leqslant \#[F(t_x, t_y)]$.

Now we say that h changes tracks (or, cheats) in a clause digraph d_j if h enters d_j at some $v^j[0]$ (or, $\overline{v}^j[0]$) but does not leave via $v^j[1]$ (or, respectively, $\overline{v}^j[1]$) for some literal v (or, respectively, $\neg v$) in D_j, as demonstrated in Figure 3. Let δ'_h be the number of clause digraph d_j in which h cheats. We observe that each time h cheats, it is able to visit at most B extra clause digraphs before it changes tracks

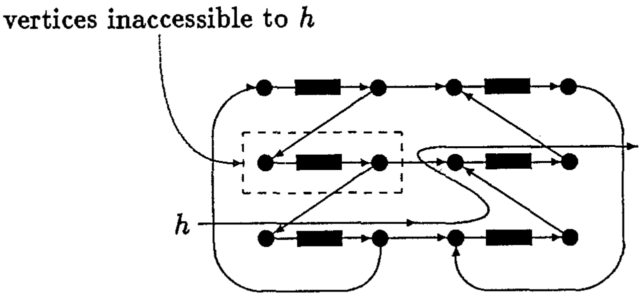

Fig. 3. h changes tracks.

again or visits a variable digraph. Therefore, the number of clause digraphs *ever* visited by h is

$$\mathcal{V}_h \leqslant \#[F(t_{h,x}, t_{h,y})] + \delta'_h \cdot B. \tag{5}$$

Combining (4) and (5), we have

$$\mathcal{V}_h \leqslant \#[F(t_x, t_y)] + \delta_h(B+1) + \delta'_h \cdot B.$$

In other words, h fails to visit at least $k_0 - (\delta_h + \delta'_h)(B+1)$ clause digraphs.

On the other hand, for each variable x_i such that $t_x(x_i) \neq t_{h,x}(x_i)$, h misses $B+1$ vertices in the variable digraph for x_i : $x_i[1], \ldots, x_i[B]$ and one of x_i and $\overline{x}_i$. In addition, each time h cheats in a clause digraph d_j, it loses access to at least $B+1$ vertices in d_j (see Figure 3). Therefore h misses additional $\delta_h(B+1) + \delta'_h(B+1)$ vertices. Altogether, h misses at least

$$\delta_h(B+1) + \delta'_h(B+1) + k_0 - (\delta_h + \delta'_h)(B+1) \geqslant k_0$$

vertices. *Case II* is therefore proved.

To conclude, if $\min_{s_x \in 2^X} \max_{s_y \in 2^Y} \#[F(X,Y)] \geqslant (1-\varepsilon_1)(m+n)$, then $l_c(G_G) \geqslant |G_F| - \varepsilon_1(m+n)$; and if $\min_{s_x \in 2^X} \max_{s_y \in 2^Y} \#[F(X,Y)] \leqslant (1-\varepsilon_2)(m+n)$, then $lc(G_F) \leqslant |G_F| - \varepsilon_2(m+n)$. Note that G_F is of size $O(mB)$, and $n = O(m)$. Therefore, to finish the proof, choosing $\varepsilon_3 = \varepsilon_1/c$ and $\varepsilon_4 = \varepsilon_2/c$ for some $c = O(B)$ suffices. □

5. Open Questions

Many NP-completeness proofs are built on reductions from the famous 3SAT problem – it reduces the complex operations of a Turing Machine to simple boolean operations. [10] further demonstrated the advantages of having the occurrences of the variables bounded by reducing it to many other NP-hard approximation problems. It appears that the problem MAX-3SAT$_2$-B might play a similar role in proving the nonapproximability of Π_2^P-hard functions. We have successfully extended their result to the case of MAX-3SAT$_2$-B. An important difference however needs to be pointed out. We observe that the reduction of [10] from MAX-3SAT to MAX-3SAT$_2$-B successfully preserves the property that the problems have only one-sided errors; that is, $\langle f_{\text{MAX-3SAT}} : 1, 1-\delta\rangle$ is G-reducible to $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1, 1-\delta'\rangle$. Our reduction from MAX-3SAT$_2$ to MAX-3SAT$_2$-B, however, does not preserve this property; we were only able to show that $\langle f_{\text{MAX-3SAT}_2} : 1, 1-\varepsilon\rangle$ is G-reducible to $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1-\varepsilon_1, 1-\varepsilon_2\rangle$, with $\varepsilon_1 > 0$. In other words, we have introduced a new error factor in our reduction. It is an interesting open question whether the one-sided version $\langle f_{\text{MAX-3SAT}_2\text{-B}} : 1, 1-\varepsilon_2\rangle$ is Π_2^P-hard for some $\varepsilon_2 > 0$.

Karger *et al.* [8] have proved, using a technique of amplifying the nonapproximability factors, that the c-approximation of the longest circuit problem of undirected graphs is NP-hard for all $c > 1$. A straightforward application of their technique to our case does not seem to work, since our graphs contain alterable edges and since our nonapproximability result on MAX-3SAT$_2$-B allows two-sided errors. Whether our main result Theorem 14 may be improved to an arbitrarily large gap c is another interesting question.

References

1. M. Ajtai (1987), Recursive Construction for 3-Regular Expanders, in *Proc. 28th IEEE Symp. on Foundations of Computer Science*, pp. 295–304.
2. A. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy (1992), Proof Verification and Hardness of Approximation Problems, in *Proc. 33rd IEEE Symp. on Foundations of Computer Science*, pp. 14–23.
3. A. Condon, J. Feigenbaum, C. Lund, and P. Shor (1993), Probabilistically Checkable Debate Systems and Approximation Algorithms for PSPACE-Hard Functions, in *Proc. 25th ACM Symp. on Theory of Computing*, pp. 305–314.
4. M.R. Garey and D.S. Johnson (1979), *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company.
5. K.-I. Ko and C.-L. Lin (1994), Non-Approximability in the Polynomial-Time Hierarchy, Technical Report 94-2, Dept. of Computer Science, SUNY at Stony Brook.
6. K.-I. Ko and C.-L. Lin (1995), On the Complexity of min-max Optimization Problems and Their Approximation, in D.-Z. Du and P.M. Pardalos, eds., *Minimax and Applications*, pp. 213–233, Kluwer, Boston.
7. M. Kiwi, C. Lund, A. Russell, D. Spielman, and R. Sundaram (1994), Alteration in Interaction, in *Proc. 9th IEEE Structure in Complexity Theory Conference*, pp. 294–303.
8. D. Karger, R. Motwani, and G.D.S. Ramkumar (1993), On Approximating the Longest Path in a Graph, in *Algorithms and Data Structures, Proc. of the 3rd Workshop, WADS'93, Lecture Notes in Computer Science 709*, pp. 421–432, Springer-Verlag.
9. C. Lund and M. Yannakakis (1993), On the Hardness of Approximating Maximization Problems, in *Proc. 25th ACM Symp. on Theory of Computing*, pp. 286–293.
10. C.H. Papadimitriou and M. Yannakakis (1991), Optimization, Approximation, and Complexity Classes, *J. Comput. System Sci.* **43**, 425–440.
11. C.H. Papadimitriou and M. Yannakakis (1993), The Traveling Salesman Problem with Distances One and Two, *Math. Oper. Res.* **18**(1), 1–11.
12. L.J. Stockmeyer (1977), The Polynomial-Time Hierarchy, *Theoret. Comput. Sci.* **3**, 1–22.