JOURNAL OF COMPUTER AND SYSTEM SCIENCES 37, 312–323 (1988)

Graph Isomorphism Is in the Low Hierarchy

UWE SCHÖNING

EWH Koblenz, Informatik, Rheinau 3–4,
D5400 Koblenz, West Germany

Received September 3, 1986; revised October 15, 1987

It is shown that the graph isomorphism problem is located in level L_2^p of the low hierarchy in NP. This implies that this problem is not NP-complete (even under weaker forms of polynomial reducibilities, like γ-reducibility) unless the polynomial-time hierarchy collapses to some finite level. © 1988 Academic Press, Inc.

1. INTRODUCTION

The problem of determining whether two given finite graphs are isomorphic is easily seen to belong to the class NP. But up to now, no polynomial time algorithm is known. On the other hand, no NP-completeness proof is known either. That is, GRAPH ISOMORPHISM is one of the few "open problems" in NP according to Garey and Johnson's terminology [9]. Note that GRAPH ISOMORPHISM is already mentioned in Karp's seminal paper [15]. Any attempt to prove NP-completeness of GRAPH ISOMORPHISM was frustrated by the apparently much more constrained nature of this problem as compared with the usual NP-complete problems. This led on the one hand to the conjecture that GRAPH ISOMORPHISM is not NP-complete (possibly in P); on the other hand that it might be NP-complete w.r.t. some weaker kind of reducibility (like γ-reducibility, see open problem 1 in [1]).

Since Ladner's work [18] we know that there exist problems in NP neither being NP-complete nor being in P, assuming $P \neq NP$. Hence, GRAPH ISOMORPHISM might very well be one of these "intermediate" problems.

It seems that there is more evidence that GRAPH ISOMORPHISM is in P (or at least "close" to P) than for NP-completeness. Indeed, some (nontrivial) special cases of the graph isomorphism problem have been shown to be in P and also some in NP $\cap$ co-NP (cf. [14]). Further, unlike the known NP-complete problems, the corresponding *counting* problem (given two graphs, compute the number of different isomorphisms between them) has been shown to be of the same "degree" of complexity as the mere decision problem [22, see also 14]. Note that the counting versions of many NP-complete problems are #P-complete, but #P is not even known to be included in the polynomial hierarchy [2].

Very recently, it has been observed that graph (non-) isomorphism can be

0022-0000/88 $3.00

"proved" in a certain probabilistic way in terms of so-called interactive proof systems [13, see also 12]. More formally, the complement of GRAPH ISOMORPHISM is an element of the class IP(2), where 2 indicates that the interactive proof protocol has 2 rounds between prover and verifier.

Further, in [12] it is shown that IP(k) is included in the class AM($k+2$) introduced by Babai [3] in terms of certain "Arthur versus Merlin" games. Babai also shows that the Arthur–Merlin hierarchy collapses to its second level, in symbols:

$$\bigcup_k \mathrm{AM}(k) = \mathrm{AM}(2).$$

To abuse the notation, we call this class AM in the following. In the papers [4, 7], direct proofs of the result that GRAPH ISOMORPHISM is in co-AM can be found.

Further, it can be shown [7] that if a set whose complement is in AM were NP-complete, then the polynomial-time hierarchy of Stockmeyer [28] collapses to its second level Σ_2^p, which seems to be a very unlikely event. Putting these results together, it follows that if GRAPH ISOMORPHISM is NP-complete, then the polynomial-time hierarchy collapses to Σ_2^p.

There is a totally different approach to these questions which will be exploited here. In [24], this author introduced a low and a high hierarchy of classes within the class NP. The definition is the analog of a definition in recursive function theory (cf. [20]), and has been designed to possibly being able to classify those "intermediate" problems in NP. The two bottom levels of the low hierarchy turned out to be P and NP $\cap$ co-NP, and the two bottom levels of the high hierarchy turned out to be the sets being NP-complete (under polynomial Turing and strong nondeterministic polynomial Turing reducibility, respectively) [24]. In fact, the entire high hierarchy may be considered as a generalized NP-completeness notion. Strong nondeterministic Turing reducibility was introduced by Long [21] as a generalization of γ-reducibility [1, see also 9]. Already in [24], Schöning has conjectured that GRAPH ISOMORPHISM is in L_2^p, the next level of the low hierarchy after $L_1^p = \mathrm{NP} \cap \text{co-NP}$. Exactly this is what will be shown in this paper.

There is still another view of the low and high hierarchies, because they give a uniform framework for understanding those polynomial hierarchy collapsing results, like the one mentioned above (for a more detailed discussion of this aspect see [17, 26]). This means in particular, that a set being in some level L_k^p of the low hierarchy cannot also be in the high hierarchy unless the polynomial-time hierarchy collapses. Since the high hierarchy also captures weaker forms of NP-completeness as the usual one, this implies that a low set cannot be NP-complete (with respect to any of a wide range of polynomial reducibilities) unless the polynomial hierarchy collapses. This paper shows that the framework developed in [24, 17] is applicable to the graph isomorphism problem.

This paper is organized as follows. First, we give a direct and much simpler proof of the fact that the complement of GRAPH ISOMORPHISM is in Babai's class AM without refering to the notion of interactive proof systems or games. Then it is shown that NP $\cap$ co-AM is included in L_2^p, which yields GRAPH ISOMORPHISM $\in L_2^p$. Hereby, two open problems are solved in this paper. Schöning's question in [24] whether GRAPH ISOMORPHISM is in L_2^p is answered positively, and Adleman and Mander's question in [1] whether GRAPH ISOMORPHISM is γ-complete is answered negatively (under the assumption that the polynomial hierarchy does not collapse to $\sum_2^p$.)

2. Preliminaries

All our sets will be languages over the fixed alphabet $\{0, 1\}$. For a set A, $\bar{A}$ will denote its complement $\{0, 1\}^* - A$, and for a class of sets $\mathscr{C}$, co-$\mathscr{C}$ denotes the set of complements of the elements in $\mathscr{C}$. For a set A and a natural number n, let $A_{=n}$ be $\{x \in A \mid |x| = n\}$, and similarly, $A_{\leqslant n} = \{x \in A \mid |x| \leqslant n\}$ where $|x|$ denotes the length of x.

Our model of computation is the multi-tape Turing machine (possibly equipped with an oracle). Let $L(M)$ denote the set accepted by Turing machine M, and $L(M, A)$ the set accepted by M when using oracle set A. The classes P and NP have their standard definition. Their A-relativized versions are denoted P(A) and NP(A), respectively. The classes of the (relativized) *polynomial-time hierarchy* [28] $\sum_k^p(A)$ and $\prod_k^p(A)$ are defined inductively as

$$\textstyle\sum_0^p(A) = \prod_0^p(A) = \mathrm{P}(A)$$

$$\textstyle\sum_{k+1}^p(A) = \mathrm{NP}(\sum_k^p(A)),$$

$$\textstyle\prod_{k+1}^p(A) = \text{co-NP}(\sum_k^p(A)) \qquad \text{for} \quad k \geqslant 0.$$

If A is the empty set, we just write $\sum_k^p$ or $\prod_k^p$, respectively. It is not known whether the polynomial hierarchy is a proper hierarchy or whether it "collapses"; i.e., for some k, $\sum_k^p = \sum_{k+1}^p = \cdots$. We will frequently use the alternating quantifier characterizations of these hierarchies: E.g., $L \in \sum_k^p(A)$ if and only if there is a set $B \in \mathrm{P}(A)$ such that $L = \{x \mid (\exists y_1)(\forall y_2) \cdots (Q_k y_k)(x, y_1, ..., y_k) \in B\}$. Here, the quantifiers are alternating (hence, $Q_k = \forall$ if k is even and $Q_k = \exists$, otherwise) and polynomially bounded; i.e., for some polynomial p, the range of the quantifications is such that $|y_i| \leqslant p(|x|)$. Further, $(x, y_1, ..., y_k)$ denotes some encoding of $(k+1)$-tuples into single strings over $\{0, 1\}$ such that coding and decoding can be done in polynomial time.

For the definition and properties of the class BPP (bounded error probabilistic polynomial time) we refer the reader to [10, 5, 19, 26].

A *NP-complete* set is a set A in NP with the property that for every set B in NP, $B \leqslant_m^p A$. Here $\leqslant_m^p$ denotes *polynomial-time many-one reducibility*: $B \leqslant_m^p A$ if and only if there is a polynomial time computable function f such that $f^{-1}(A) = B$. Other reducibilities which are mentioned in the text include: $\leqslant_T^p$-reducibility ($B \leqslant_T^p A$ iff $B \in \mathrm{P}(A)$), and γ-reducibility [1]: $B \leqslant_\gamma A$ if and only if there is a nondeterministic and polynomial time Turing machine with output device such that for every input x there is at least one computation that produces an output, and additionally, $x \in B$ iff $y \in A$ for every possible output y on input x.

We will consider finite, undirected graphs $G = (V, E)$ on the vertex set $V = \{1, ..., n\}$. A permutation p on $\{1, ..., n\}$ is an *automorphism* of graph G if $p(G) = G$. Here, $p(G)$ is the graph (V, E'), where $(p(u), p(v)) \in E'$ iff $(u, v) \in E$. The set of automorphisms of G is denoted $\mathrm{Aut}(G)$ (which is actually a group under the composition operation). A graph G' is *isomorphic* to G if $G' = p(G)$ for some permutation p. Note that the number of graphs being isomorphic to a given graph G is exactly $n!/|\mathrm{Aut}(G)|$ (see [14]).

The *graph isomorphism problem* is given as the set GRAPH ISOMORPHISM $=$ $\{(G_1, G_2) \mid G_1$ and G_2 are isomorphic$\}$ which is easily seen to belong to the class NP, but is currently not known to be NP-complete nor to be in P.

3. Graph Isomorphism Is in Co-AM

The class AM was introduced by Babai [3] in terms of certain "Arthur versus Merlin" games. Recently it has been shown by Goldwasser and Sipser [12] that the class AM is identical to the class IP (which stands for interactive proof systems with a constant number of rounds). The class IP was introduced in [11] with a motivation stemming from communication complexity and cryptographic protocols. The relevance to the graph isomorphism problem is that it is not very hard to see that GRAPH ISOMORPHISM (and in particular, its complement) belong to IP (see [12]).

We will now give a straightforward definition of AM without referring to proof systems or games, and then give a much simpler proof of the fact that GRAPH ISOMORPHISM is in co-AM.

Definition 3.1. A set A is in AM if there is a set B in NP and a polynomial q such that for each x, $|x| = n$, with probability at least 3/4,

$$(x, y) \in B \quad \text{iff} \quad x \in A,$$

where $y \in \sum^{q(n)}$ is randomly chosen under uniform distribution.

Note that the constant $\frac{3}{4}$ is arbitrary, and could be substituted by any constant c $(\frac{1}{2} < c < 1)$ without changing the defined class.

It is obvious that $\mathrm{NP} \subseteq \mathrm{AM}$ (by taking $B = A \times \sum^*$) and that $\mathrm{BPP} \subseteq \mathrm{AM}$ (by taking $B \in P$). Informally speaking, the definitional step from P to BPP is the same as from NP to AM, i.e., AM is the randomized version of NP, in the same sense as BPP can be considered as the randomized version of P.

The set inclusion structure between these classes and some other classes of the polynomial-time hierarchy is shown in Fig. 1.

To prepare the proof we need the following combinatorial definition and lemma.

DEFINITION 3.2. For two graphs G_1, G_2 with n vertices, define $\text{num}(G_1, G_2)$ to be the number of different triples of the form (H, p, i), where H ranges over all graphs being isomorphic to either G_1 or to G_2, and p is an automorphisms of G_i and $i \in \{1, 2\}$.

LEMMA 3.3. *If* G_1 *and* G_2 *are isomorphic, then* $\text{num}(G_1, G_2) = 2n!$, *otherwise* $\text{num}(G_1, G_2) \geq 4n!$

Proof. If the graphs are isomorphic, then $|\text{Aut}(G_1)| = |\text{Aut}(G_2)|$, and the set of isomorphic versions of G_1 is the same as for G_2. Therefore, $\text{num}(G_1, G_2) = (n!/|\text{Aut}(G_1)|) \cdot (|\text{Aut}(G_1)| + |\text{Aut}(G_2)|) = 2n!$ If the graphs are not isomorphic, then all $n!/|\text{Aut}(G_1)|$ isomorphic versions of G_1 are different from all $n!/|\text{Aut}(G_2)|$ isomorphic versions of G_2. Hence, $\text{num}(G_1, G_2) = (n!/|\text{Aut}(G_1)| + n!/|\text{Aut}(G_2)|) \cdot (|\text{Aut}(G_1)| + |\text{Aut}(G_2)|) \geq 4n!$ ∎

Note that the objects (H, p, i) that are counted in $\text{num}(G_1, G_2)$ can be nondeterministically guessed and verified in polynomial time. (On input (G_1, G_2), guess permutations p, q and $i, j \in \{1, 2\}$; and check whether $p(G_i) = G_i$. If so, output (H, p, i), where $H = q(G_j)$.) This observation will later become important.

The following lemma appears in a similar form in Sipser [27] and also in Goldwasser, Sipser [12], and can be considered as an application of universal hashing due to Carter and Wegman [8].

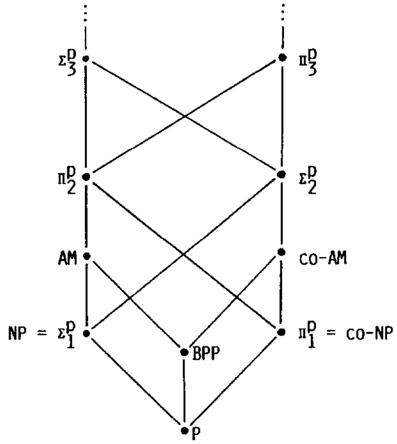

Fig. 1. Inclusion structure.

In the following, a random hash function $h: \sum^t \rightarrow \sum^m$ is given by a Boolean (t, m)-matrix whose elements $m_{ij} \in \{0, 1\}$ are picked uniformly at random and independently. Then the jth bit of $h(a_1 \cdots a_t) \in \sum^m$ is calculated by $(m_{1j} \wedge a_1) \oplus \cdots \oplus (m_{tj} \wedge a_t)$.

The lemma states that on every fixed "small" subset of $\sum^t$ a randomly selected collection of hash functions $\{h_i\}$ is very likely to be collision-free. But on a too "big" subset a collision is unavoidable. Here, a "collision" means the existence of an x such that for every h_i there is a $y \neq x$ with $h_i(x) = h_i(y)$.

HASHING LEMMA 3.4. (a) *If $X \subseteq \sum^t$ has cardinality at most 2^{m-1}, then for randomly selected hash functions $h_1, ..., h_{m+1}: \sum^t \rightarrow \sum^m$, with probability at most $\frac{1}{4}$,*

$$(\exists x \in X)(\forall i \leqslant m+1)(\exists y \in X)[y \neq x \text{ and } h_i(x) = h_i(y)]. \tag{*}$$

(b) *If X has more than $(m+1)\, 2^m$ elements, then the probability for $(*)$ is* 1.

Proof. (a) is essentially the same as the coding lemma of [27]. By the way a random hash function $h: \sum^t \rightarrow \sum^m$ is defined, it can be seen that for different $x, y \in \sum^t$, the probability that the jth bit of $h(x)$ equals the jth bit of $h(y)$ is $\frac{1}{2}$. Since the elements m_{ij} of the random Boolean matrix for h are picked independently, we get $\text{Prob}[h(x) = h(y)] = 2^{-m}$. Hence, for any given $x \in X$, the probability that there exists a $y \in X$ with $h(x) = h(y)$ is at most $|X| \cdot 2^{-m} \leqslant \frac{1}{2}$. The hash functions $h_1, ..., h_{m+1}$ are chosen independently. Hence, for any fixed $x \in X$, the probability that for every $i \leqslant m+1$ there is a $y \in X$ with $h_i(x) = h_i(y)$ is at most $(\frac{1}{2})^{m+1} = 2^{-m-1}$. Finally, the probability for a collision (i.e., the predicate $(*)$) is at most $|X| \cdot 2^{-m-1} \leqslant 2^{-2} = \frac{1}{4}$.

(b) Follows from the pidgeonhole principle (see also [29, Theorem 3.11]): Every hash function h_i can behave in an injective way only on domains up to size 2^m. If X has more than $(m+1)\, 2^m$ elements, then a collision must occur. Hence, $(*)$ is always true. ∎

Note that the probability bound $\frac{1}{4}$ in the hashing lemma can be improved to 2^{-q} by taking $m+q-1$ hash functions instead of $m+1$.

We like to apply the Hashing Lemma to the set $X = \{(H, p, i) \mid (H, p, i)$ as in Definition 3.2$\}$. Then, for G_1, G_2 not being isomorphic, we have $|X| \geqslant 4n!$, and for G_1, G_2 being isomorphic, $|X| = 2n!$. But unfortunately, the bounds 2^{m-1} and $(m+1) \cdot 2^m$ in the hashing lemma are too weak to distinguish between $2n!$ and $4n!$. But let us, instead, apply the hashing lemma to the set $Y = X^k$ (k depending on n). In this case, $|Y| \geqslant (4n!)^k$ for non-isomorphic graphs, and $|Y| = (2n!)^k$ for isomorphic graphs. Now the hashing lemma is applicable if we choose, e.g., $m = 1 + \lceil n \cdot \log(2n!) \rceil$ and $k = n$, and the following theorem can be established.

THEOREM 3.5. *There is a predicate B in* NP *and a polynomial p such that for every pair of graphs G_1, G_2 with n vertices, the following two assertions hold*:

(a) *If G_1, G_2 are isomorphic, then the probability for $(G_1, G_2, y) \in B$ is at most $\frac{1}{4}$.*

(b) *If G_1, G_2 are non-isomorphic, then the probability for $(G_1, G_2, y) \in B$ is* 1.

Hereby, $y \in \sum^{p(n)}$ is picked uniformly at random. Hence, GRAPH ISOMORPHISM $\in$ co-AM.

Proof. By the above discussion, the set B can be chosen as the collision predicate (*) where y is an encoding of the collection of hash functions, and X is the set of all triples as in Definition 3.2 (or rather, the set of all k-tuples of such triples where the choice of k is explained above.) The fact that $B \in \mathrm{NP}$ follows from the observation that the objects (H, p, i) can be nondeterministically generated, and thus the collision predicate (*) is in NP. (Note that the universal quantifier in (*) has polynomial range and can thus be eliminated.) ∎

There are several comments to be made. Note that Goldwasser and Sipser's proof [12] of this result is much more involved. The core of their proof is to show the equivalence of IP and AM. Here, we have avoided to introduce the class IP and the game-theoretic background connected to AM and IP. Recently, also other authors presented direct proofs of GRAPH ISOMORPHISM $\in$ co-AM [4, 6]. Note also that our result is actually a little stronger than what is expressed by GRAPH ISOMORPHISM $\in$ co-AM, because of the probability 1 on the one side.

Boppana and Hastad [7] show that $\mathrm{NP} \subseteq$ co-AM implies $\bigcup_k \sum_k^p \subseteq$ co-AM. Then, by the fact that $\mathrm{AM} \subseteq \prod_2^p$ (see [31]), it follows that GRAPH ISOMORPHISM cannot be NP-complete unless the polynomial hierarchy collapses to $\prod_2^p \cap \sum_2^p$. These observations will be superseded by our results of the next section, since we will additionally show that not even weaker forms of NP-completeness (like γ-completeness) are possible for GRAPH ISOMORPHISM unless the polynomial hierarchy collapses (cf. Open Problem 1 in [1].)

4. Lowness of Graph Isomorphism

A set A in NP is (polynomially) *low*$_k$ (in symbols: $A \in L_k^p$) if $\sum_k^p(A) \subseteq \sum_k^p$, and A is *high*$_k$ (in symbols: $A \in H_k^p$) if $\sum_{k+1}^p \subseteq \sum_k^p(A)$. This author [24] introduced this definition in the context of polynomial time computations as a straightforward translation of the definitions from recursive function theory (cf. [20]).

Obviously, $L_0^p \subseteq L_1^p \subseteq \cdots$ and $H_0^p \subseteq H_1^p \subseteq \cdots$. It is easy to see that L_0^p is the class P, and H_0^p is the class of NP-complete sets w.r.t. $\leqslant_T^p$-reducibility. Further, it can be shown [24] that $L_1^p = \mathrm{NP} \cap \text{co-NP}$, and H_1^p is the class of NP-complete sets w.r.t. the weaker $\leqslant_T^{sn}$-reducibility which was introduced and investigated by Long [21]. Note that the more familiar γ-reducibility ([1], see also [9]) is a special case of $\leqslant_T^{sn}$-reducibility, hence the γ-complete sets are in H_1^p. Adleman and Manders [1] presented some examples of γ-complete sets not known to be NP-complete in the usual sense, and they conjectured that GRAPH ISOMORPHISM might also be γ-complete.

In a sense, the high hierarchy generalizes the notion of NP-completeness, as shown by the following proposition.

PROPOSITION 4.1 [24]. *There exists a set being both low_k and high_k if and only if the polynomial hierarchy collapses to Σ_k^p, i.e. $\bigcup_n \Sigma_n^p = \Sigma_k^p$.*

Observe that the case $k=0$ means the well-known fact that an NP-complete set is in P if and only if $\mathrm{P}=\mathrm{NP}$.

Other known results about the low and high hierarchies are that the sparse sets in NP, the classes R, $\mathrm{NP} \cap \mathrm{BPP}$ are included in L_2^p, and the sets in NP having polynomial-size circuits are in L_3^p [17, 26]. The main result of this section will be that $\mathrm{NP} \cap$ co-AM is included in L_2^p. First we need to observe that, as in the case of the class BPP, a certain probability amplification property holds for the class AM.

PROPOSITION 4.2. *If A is in AM, then for every polynomial q there exists a set B in* NP *and a polynomial p such that for all x, $|x|=n$, if $w \in \Sigma^{p(n)}$ is chosen uniformly at random, then with probability at least $1-2^{-q(n)}$, $x \in A$ iff $(x, w) \in B$.*

Proof. As first noted by Bennett and Gill [5], by a "majority vote" argument, the error probability can be exponentially decreased. More formally, the set $B \in \mathrm{NP}$ from Definition 3.1 has to be substituted by B', where $(x, w_1 w_2 \cdots w_t) \in B'$ iff for more than $t/2$ many i's, $(x, w_i) \in B$. Here, t is a suitable polynomial in $|x|$. A detailed proof may be found, e.g., in [26, Chap. 2]. ∎

By choosing the bound for the error probability in Proposition 4.2 small enough, the following can be achieved: If A is in AM, then there is a set B in NP and a polynomial p such that for each n, with probability at least $1-2^{-n}$,

$$A_{\leqslant n} = \{x \mid (x, w) \in B\}_{\leqslant n},$$

where $w \in \Sigma^{p(n)}$ is picked randomly with uniform distribution.

This strong property of AM will be used in the following theorem. Furthermore, we need the following claim.

CLAIM 4.3 [25, p. 74f]. *If $E \subseteq \Sigma^{p(n)}$ is a set with $|E| > (1-2^{-n}) \cdot 2^{p(n)}$, then* (a) *and* (b) *hold.*

(a) $(\exists u = (u_1, ..., u_{p(n)}),\ |u_i| = p(n))\ (\forall v,\ |v| = p(n))$

$$[\textit{for some } i \leqslant p(n),\ u_i \oplus v \in E],$$

(b) $(\forall u = (u_1, ..., u_{p(n)}),\ |u_i| = p(n))\ (\exists v,\ |v| = p(n))$

$$[\textit{for all } i \leqslant p(n),\ u_i \oplus v \in E],$$

where $\oplus$ means bitwise addition modulo 2.

Proof of Claim 4.3. This is a modification of Lautemann's argument showing $\mathrm{BPP} \subseteq \Sigma_2^p$ [19]. Suppose (a) does not hold. Then for every $u = (u_1, ..., u_{p(n)})$ there is a v such that $u_1 \oplus v \notin E, ..., u_{p(n)} \oplus v \notin E$. Thus, the set U of all $p(n)$-tuples (of $p(n)$-tuples) can be partitioned into (not necessarily disjoint) subsets $U_1, ..., U_{2^{p(n)}}$, where $U_j = \{(u_1, ..., u_{p(n)}) \mid \text{for the } j\text{th string } v \text{ of size } p(n) \text{ according to lexicographical order, } u_1 \oplus v \notin E, ..., u_{p(n)} \oplus v \notin E\}$. Therefore, for some $j \leqslant 2^{p(n)}$, $|U_j| \geqslant |U|/2^{p(n)} = 2^{p^2(n) - p(n)}$. On the other hand, $|\bar{E}|^{p(n)} \geqslant |U_j|$, thus $|\bar{E}| \geqslant 2^{p(n)-1}$, which contradicts the assumption about the density of E.

Assume that (b) does not hold, and fix some $u = (u_1, ..., u_{p(n)})$ such that for every v of size $p(n)$ there is an $i \leqslant p(n)$ with $u_i \oplus v \notin E$. The set V of all strings v of size $p(n)$ is thus partitioned into $V_1, ..., V_{p(n)}$, where $V_j = \{v \in V \mid u_j \oplus v \notin E\}$. For some $j \leqslant p(n)$ we get the inequality $|\bar{E}| \geqslant |V_j| \geqslant 2^{p(n)}/p(n)$. This, again, contradicts the assumption about E's density. ∎

THEOREM 4.4. NP $\cap$ co-AM *is included in* L_2^p.

Proof. Let A be in NP $\cap$ co-AM and let $L \in \Sigma_2^p(A)$. Hence, by the quantifier characterization of the (relativized) polynomial hierarchy [28],

$$L = \{x \mid (\exists y)(\forall z)(x, y, z) \in L(M, A)\},$$

where M is deterministic and polynomial time bounded and the quantifiers range over strings up to some polynomial size (in $|x|$). By the fact that $A \in \mathrm{NP}$, we have $A = L(M')$ for some nondeterministic and polynomial time machine. Further, by the above discussion, there is a nondeterministic polynomial time Turing machine M'' such that with probability at least $1 - 2^{-n}$,

$$\bar{A}_{\leqslant n} = \{x \mid (x, w) \in L(M'')\}_{\leqslant n} \tag{**}$$

where w is chosen uniformly at random from $\Sigma^{p(n)}$, p being a suitable polynomial.

Now we apply Claim 4.3 with E being the set of all w that satisfy (**). Then we get the following characterization of L where all the quantifiers are appropriately polynomially bounded

$$L = \{x \mid (\exists u)(\exists y)(\forall v)(\forall z) \\ [\exists i \leqslant p(n), (u_i \oplus v, x, y, z) \notin K]\}.$$

Note that the existential quantifier within the brackets has polynomial range and thus is not essential. The set K is accepted by the following nondeterministic, polynomial time Turing machine (hence $K \in \mathrm{NP}$):

> On input (w, x, y, z):
> run M on input (x, y, z) and reject if M accepts, and accept if M rejects. Oracle queries of M are handled as follows: if s is the query string, first guess non-deterministically whether $s \in A$ or $s \notin A$. Verify the first case by running M' on s. Verify the second case by running M'' on (s, w). Continue with answer "yes" (or "no", respectively) if the verification succeeds, otherwise reject.

To see the correctness of this characterization, observe that for all w that satisfy $(**)$, $(w, x, y, z) \in K$ iff $(x, y, z) \notin L(M, A)$. The "if" part of this equivalence follows from Claim 4.3(a), and the "only if" part follows from Claim 4.3(b).

Now by the fact that $K \in \mathrm{NP}$, the expression in brackets is in co-NP, and thus L has a $\sum_2^p$-characterization. This shows that $A \in L_2^p$. ∎

Combining Theorems 3.5 and 4.4, we get the following corollary which answers Open Problem 3 in [24] positively.

COROLLARY 4.5. GRAPH ISOMORPHISM *is* low_2.

Using Proposition 4.1, we immediately obtain

COROLLARY 4.6. GRAPH ISOMORPHISM *cannot be in* H_k^p *unless the polynomial-time hierarchy collapses to* $\sum_{\max\{2,k\}}^p$.

A special case of this corollary is the following

COROLLARY 4.7. GRAPH ISOMORPHISM *cannot be* NP-*complete* (*under* $\leqslant_m^p$, $\leqslant_T^p$, $\leqslant_\gamma$, $\leqslant_R$, $\leqslant_T^{sn}$-*reducibility*) *unless the polynomial-time hierarchy collapses to* $\sum_2^p$.

For the case of γ-completeness, this answers Open Problem 1 in [1] negatively (assuming $\sum_2^p \neq \prod_2^p$).

This lowness result for GRAPH ISOMORPHISM could be taken as further evidence that this problem finally will turn out to be in P. But, on the other hand, notice that in [17] it is shown that under the hypothesis NEXPTIME, there exists sets in $L_2^p - \mathrm{P}$; and under the stronger hypothesis NEXPTIME $\neq$ co-NEXPTIME, there exist sets in $L_2^p - L_1^p$. Hence, the possibility still remains that the membership in L_2^p for GRAPH ISOMORPHISM cannot be improved.

5. CONNECTIONS TO NON-UNIFORM CLASSES

The purpose of this section is to point out some relationships to non-uniform complexity classes.

DEFINITION 5.1 [16]. For a class of sets $\mathscr{C}$, define $\mathscr{C}$/poly to be the class of all sets A for which there is a set $B \in \mathscr{C}$ and a polynomial p such that for each n there is a string w, $|w| \leqslant p(n)$, such that

$$A_{\leqslant n} = \{x \mid (x, w) \in B\}_{\leqslant n}.$$

Observe that P/poly is exactly the class of sets having polynomial size circuits (cf. [26]), and NP/poly is the class of sets having polynomial size "generators" [30, 25, 26].

From the definition of AM it is clear that $\mathrm{AM} \subseteq \mathrm{NP/poly}$. Hence, by the above results, GRAPH ISOMORPHISM $\in \mathrm{NP} \cap (\text{co-NP/poly})$.

As a generalization of a result by Meyer (stated in [8]), it is shown in [25, 26] that for any "reasonable" complexity class $\mathscr{C}$, $\mathscr{C}/\mathrm{poly} = \bigcup\{\mathscr{C}(S) \mid S$ is a sparse set$\} = \bigcup\{\mathscr{C}(T) \mid T$ is a tally set$\}$. This holds in particular for $\mathscr{C} = \mathrm{P}$, NP, $\sum_k^p$, $\prod_k^p$, PSPACE. This tells us that GRAPH ISOMORPHISM $\in \mathrm{NP} \cap \bigcup\{\text{co-NP}(T) \mid T$ is a tally set$\}$.

Yap shows in [30] that $\text{co-NP} \subseteq \mathrm{NP/poly}$ (or equivalently, $\mathrm{NP} \subseteq \text{co-NP/poly}$) implies that the polynomial hierarchy collapses to $\sum_3^p$. Hence, the conclusion that the polynomial hierarchy collapses (at least to level three), assuming that GRAPH ISOMORPHISM is NP-complete, can already be drawn from Yap's result without using Theorem 4.4 nor Boppana, Hastad, and Zachos' results [7].

In [17] it is shown that $A \in \mathrm{NP} \cap (\mathrm{P/poly})$ and A being self-reducible implies $A \in L_2^p$. Note that the graph isomorphism problem is (polynomially equivalent to) a self-reducible set [23]. Hence, another approach to show that GRAPH ISOMORPHISM is low_2 would be to show $A \in \mathrm{NP} \cap (\text{co-NP/poly})$ and A being self-reducible implies $A \in L_2^p$. But this is still an open problem.

Acknowledgments

The author thanks Oded Goldreich, Ker-I Ko, Kurt Mehlhorn, Silvio Micali, Osamu Watanabe, Klaus Wagner, and the anonymous referee for their constructive criticism on the earlier versions of this paper.

References

1. L. Adleman and K. Manders, Reducibility, randomness, and intractability, *in* "Proceedings, 9th ACM Symp. Theory of Comput. 1977," pp. 151–163.
2. D. Angluin, On counting problems and the polynomial time hierarchy, *Theor. Comput. Sci.* **12** (1980), 161–173.
3. L. Babai, Trading group theory for randomness, *in* "Proceedings, Symp. Theory of Comput. 1985," pp. 421–429.
4. L. Babai, Arthur–Merlin games: A randomized proof system, and a short hierarchy of complexity classes, manuscript, 1986.
5. C. H. Bennett and J. Gill, Relative to a random oracle A, $\mathrm{P}^A \neq \mathrm{NP}^A \neq \text{co-NP}^A$ with probability 1, *SIAM J. Comput.* **10** (1981), 96–113.
6. L. Berman and J. Hartmanis, On isomorphism and density of NP and other complete sets, *SIAM J. Comput.* **6** (1977), 305–327.
7. R. B. Boppana, J. Hastad, and S. Zachos, Does co-NP have short interactive proofs?, *Inform. Process. Lett.* **25** (1987), 127–132.
8. J. L. Carter and M. N. Wegman, Universal classes of hash functions, *J. Comput. System Sci.* **18** (1979), 143–154.
9. M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," Freeman, San Francisco, 1979.
10. J. Gill, Computational complexity of probabilistic Turing machines, *SIAM J. Comput.* **6** (1977), 675–695.

11. S. Goldwasser, S. Micali, and C. Rackoff, The knowledge complexity of interctive proofs, *in* "Proceedings, 17th ACM Symp. Theory of Computing, 1985," pp. 291–304.
12. S. Goldwasser and M. Sipser, Private coins versus public coins in interactive proof systems, *in* "Proceedings, 18th ACM Symp. Theory of Comput., 1986," pp. 59–68.
13. O. Goldreich, S. Micali, and A. Widgerson, Proofs that yield nothing but their validity and a methodology of crypographic protocol design, *in* "Proceedings, 27th Symp. Found. Comput. Sci., 1986."
14. C. M. Hoffmann, "Group-Theoretic Algorithms and Graph Isomorphism," Lecture Notes in Computer Science Vol. 136, Springer-Verlag, Heidelberg, 1982.
15. R. M. Karp, Reducibility among combinatorial problems, *in* "Complexity of Computer Computations," (Miller and Thatcher, Eds.), pp. 302–309, Plenum, New York, 1972.
16. R. M. Karp and R. J. Lipton, Some connections between non-uniform and uniform complexity classes, *in* "Proceedings, 12th ACM Symp. Theory of Comput., 1980," pp. 302–309.
17. K. Ko and U. Schöning, On circuit-size complexity and the low hierarchy in NP, *SIAM J. Comput.* **14** (1985), 41–51.
18. R. E. Ladner, On the structure of polynomial time reducibilities, *J. Assoc. Comput. Mach.* **22** (1975), 155–171.
19. C. Lautemann, BPP and the polynomial hierarchy, *Inform. Process. Lett.* **17** (1983), 215–217.
20. M. Lerman, "Degrees of Unsolvability," Springer-Verlag, Heidelberg, 1983.
21. T. J. Long, Strong nondeterministic polynomial-time reducibilities, *Theoret. Comput. Sci.* **21** (1982), 1–25.
22. R. Mathon, A note on the graph isomorphism counting problem, *Inform. Process. Lett.* **8** (1979), 131–132.
23. C. P. Schnorr, Optimal algorithms for self-reducible problems, *in* "Proceedings, 3rd Colloq. Automata. Languages, Programming," pp. 322-337, Edinburgh Univ. Press, Edinburgh, 1976.
24. U. Schöning, A low and a high hierarchy within NP, *J. Comput. System Sci.* **27** (1983), 14–28.
25. U. Schöning, On small generators, *Theoret. Comput. Sci.* **34** (1984), 337–341.
26. U. Schöning, "Complexity and Structure," Lecture Notes in Computer Science Vol. 211, Springer-Verlag, Heidelberg, 1986.
27. M. Sipser, A complexity theoretic approach to randomness, *in* "Proceedings, 15th ACM Symp. Theory of Comput., 1983," pp. 330-335.
28. L. J. Stockmeyer, The polynomial-time hierarchy, *Theoret. Comput. Sci.* **3** (1977), 1–22.
29. L. J. Stockmeyer, On approximation algorithms for $\#P$, *SIAM J. Comput.* **14** (1985), 849–861.
30. C. K. Yap, Some consequences of non-uniform conditions on uniform classes, *Theoret. Comput. Sci.* **26** (1983), 287–300.
31. S. Zachos, Probabilistic quantifiers, adversaries, and complexity classes: an overview, *in* "Proceedings, Structure in Complexity Theory Conference," Lecture Notes in Computer Science Vol. 223, pp. 383–400, Springer-Verlag, New York/Berlin, 1986.