Information Processing Letters 25 (1987) 127–132
North-Holland

6 May 1987

DOES co-NP HAVE SHORT INTERACTIVE PROOFS?

Ravi B. BOPPANA * and Johan HASTAD **

Department of Mathematics and Laboratory for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139, U S A

Stathis ZACHOS

Department of Computer and Information Science, Brooklyn College of The City University of New York, Brooklyn, NY 11210, U S A

Communicated by David Gries
Received 10 July 1986
Revised 15 September 1986 and 24 December 1986

Babai (1985) and Goldwasser, Micali and Rackoff (1985) introduced two probabilistic extensions of the complexity class NP The two complexity classes, denoted AM[Q] and IP[Q] respectively, are defined using randomized interactive proofs between a prover and a verifier Goldwasser and Sipser (1986) proved that the two classes are equal We prove that if the complexity class co-NP is contained in IP[k] for some constant k (i e, if every language in co-NP has a short interactive proof), then the polynomial-time hierarchy collapses to the second level As a corollary, we show that if the Graph Isomorphism problem is NP-complete, then the polynomial-time hierarchy collapses

Keywords Interactive proof, probabilistic computation, polynomial-time hierarchy, graph isomorphism

1. Introduction

For which languages is it possible to prove efficiently that a particular string is in the language? The standard notion of an efficient proof says that this is possible precisely for the languages in NP. The theory of efficient computing shows, however, that randomization can increase power, provided that we allow a small probability of error. Applying this idea to provability, Babai [2] and Goldwasser, Micali and Rackoff [9] recently proposed two, seemingly different, randomized generalizations of NP.

In both papers, a proof is an interaction between two players, a prover and a verifier. The verifier is a probabilistic polynomial-time Turing machine; the prover is all powerful. The two players interact by sending messages to each other and, after at most a polynomial number of interactions, the verifier performs a polynomial-time computation to determine if he or she is convinced that a particular string is in the language. A language L has an interactive proof if, for strings x in L, the prover can, with probability at least $1 - 2^{-|x|}$, convince the verifier that x is in L, for strings x not in L, the probability that the verifier can be convinced that x is in L, even by an optimal prover, is at most $2^{-|x|}$.

The key difference between the two models proposed by Babai [2] and Goldwasser, Micali and Rackoff [9] is in the way the verifier can use coin flips. Babai requires that coins flipped by the verifier be known to the prover, and so the only tasks of the verifier are to supply random bits to the prover and to decide, at the end of the interac-

* Research supported in part by the National Science Foundation with a Mathematical Sciences Postdoctoral Fellowship.
** Research supported in part by an IBM Fellowship

tion, if he or she is convinced that a particular string is in the language. In the model of Goldwasser, Micali and Rackoff, the verifier may perform computations based on the coins flipped, and may also hide the coins from the prover. Two reasons for such flexibility are first, it seems to give the broadest possible definition of what is efficiently provable, second, it is useful for cryptography Surprisingly, Goldwasser and Sipser [10] proved that for language recognition the two models are equivalent. In fact, they proved that every language recognizable by a protocol using hidden coins can be recognized by a protocol using only public coins with the same number of interactions Thus, from a language recognition viewpoint, public coins suffice, even though interactive proofs are simpler to construct using hidden coins

Babai [2] called the verifier Arthur and the prover Merlin. Babai let AM[Q] denote the class of languages recognizable by an interactive proof using only public coins with $Q(|x|)$ interactions between the two players. For example, in an AM[2] protocol, Arthur first sends a random string to Merlin, then Merlin responds, and finally Arthur decides whether to accept. For every constant $k \geqslant 2$, Babai proved that the class AM[k] equals AM[2]. In addition, Babai showed that the class AM[2] is contained in the polynomial-time hierarchy (in fact, he showed that AM[2] is contained in Π_2^P).

Both of the above results can also be obtained by quantifier-swapping techniques developed by Hinman and Zachos [11], Zachos [17] and Zachos and Fürer [18] Their approach is to characterize complexity classes (with or without probabilistic components) in terms of existential ($\exists$), universal ($\forall$) and probabilistic ($\exists^+$) quantifiers applied to polynomial-time predicates. The quantifier $\exists^+$ means 'for most', i.e., for at least a fraction $1 - 2^{-|x|}$ of the computation paths. In their notation, the class

$$\mathsf{NP} = (\exists/\forall), \qquad \mathsf{co\text{-}NP} = (\forall/\exists),$$

$$\mathsf{RP} = (\exists^+/\forall),$$

$$\mathsf{BPP} = (\exists^+/\exists^+) = (\exists^+\forall/\forall\exists^+) = (\forall\exists^+/\exists^+\forall),$$

$$\mathsf{MA} = (\exists\forall/\forall\exists^+) \subseteq (\forall\exists/\exists^+\forall) = \mathsf{AM},$$

and

$$\Pi_2^P = (\forall\exists/\exists\forall).$$

Languages with interactive proofs are the languages for which it is possible to prove, efficiently and probabilistically, membership of the language. An open question is whether this class contains co-NP, the class of languages having short deterministic proofs of nonmembership Fortnow and Sipser [6] gave evidence that this is false, by constructing an oracle A such that $\mathsf{co\text{-}NP}^A$ is not contained in $\mathsf{AM}^A[Q]$ for all polynomials Q Earlier, Hinman and Zachos [11] had constructed an oracle A such that $\mathsf{co\text{-}NP}^A$ is not contained in $\mathsf{AM}^A[2]$

The main result of this paper is another piece of evidence in the same direction. We prove that if co-NP is contained in AM[2], then the entire polynomial-time hierarchy collapses to its second level Our proof does not seem to carry over to the case where $\mathsf{co\text{-}NP} \subseteq \mathsf{AM}[Q]$ for unbounded Q. We still conjecture, however, that co-NP is not a subset of AM[Q] for all polynomials Q.

Our result provides a new type of evidence that a language L is not NP-complete. if the complement of a language L is in AM[2] and the language L itself is NP-complete, then the polynomial-time hierarchy collapses. Recently, Goldreich, Micali and Wigderson [8] showed that the complement of the Graph Isomorphism problem is in AM[2] Combining their result with ours, we deduce that if the Graph Isomorphism problem is NP-complete, then the polynomial-time hierarchy collapses

An open question is whether, for unbounded Q, the classes AM[Q] and AM[2] are equal. Aiello, Goldwasser and Hastad [1] showed that there is an oracle A such that $\mathsf{AM}^A[Q]$ is not equal to $\mathsf{AM}^A[2]$ for all unbounded Q. In fact, for their oracle A, they show that the class $\mathsf{AM}^A[Q]$, for all unbounded Q, is not contained in the polynomial-time hierarchy relative to A. A trivial extension of a result by Babai [2] shows that $\mathsf{AM}^A[2]$ is contained in $\Pi_2^{P,A}$ for all oracles A.

Schöning [14] has further investigated the complexity of the Graph Isomorphism problem, showing that it lies in the second level of the low hierarchy (for definition of the low hierarchy, see

[13]). Schoning observes that this result also implies that the Graph Isomorphism problem is not NP-complete, unless the polynomial-time hierarchy collapses to its second level.

Several other papers are related to our work. Condon and Ladner [5] describe a model of probabilistic game automata that generalizes interactive proof systems Zachos [17] and Zachos and Fürer [18] show how probabilistic complexity classes can be described using probabilistic quantifiers. Ko [12] and Zachos [17] prove that if the complexity class NP is contained in the class BPP, then the polynomial-time hierarchy is contained in BPP $\subseteq \Sigma_2^P$.

The outline of the paper is as follows In Section 2 we prove that if the class co-NP is contained in the class AM[2], then the polynomial-time hierarchy collapses. In Section 3 we deduce that if the Graph Isomorphism problem is NP-complete, then the polynomial-time hierarchy collapses.

2. The classes co-NP and AM[2]

In this section we show that the class co-NP is unlikely to be contained in AM[2]. In particular, we show that if co-NP is contained in AM[2], then the entire polynomial-time hierarchy collapses to AM[2] $\subset \Pi_2^P$.

First, we present a few definitions. A *string* is a finite sequence of 0's and 1's. Let $|x|$ denote the length of a string x. Given two strings x and y, let $x \circ y$ denote the concatenation of x and y. A *language* is a (finite or infinite) set of strings. The *complement* of a language is the set of strings that are not in the language. Given a class of languages $\mathscr{C}$, let co-$\mathscr{C}$ be the class of languages whose complements are in $\mathscr{C}$. For definitions of the complexity classes Σ_k^P and Π_k^P, see [16]. The *polynomial-time hierarchy* is the union, over $k \geqslant 1$, of the classes Σ_k^P.

Next, we formally define the randomized complexity class AM[2], which we simply refer to as AM. A language L is in AM iff there is a language M in NP and a polynomial p such that, for all strings x, the fraction of strings y of length $p(|x|)$ that satisfy $x \circ y \in M$ is (i) at least $\frac{2}{3}$ for x in L, and (ii) at most $\frac{1}{3}$ for x not in L Observe that NP is a subset of AM.

The class AM is a randomized version of NP in the same way that BPP (defined by Gill [7]) is a randomized version of P. Bennett and Gill [3] show that the $\frac{1}{3}-\frac{2}{3}$ separation of BPP can be amplified by taking a majority vote. The lemma below states the analogous result for the class AM; again, the amplification is achieved by a majority vote

2.1. Lemma. *For every language* L *in* AM *and every polynomial* q, *there is a language* M *in* NP *and a polynomial* p *such that, for all strings* x, *the fraction of strings* y *of length* $p(|x|)$ *that satisfy* $x \circ y \in M$ *is* (i) *at least* $1 - 2^{-q(|x|)}$ *for* x *in* L, *and* (ii) *at most* $2^{-q(|x|)}$ *for* x *not in* L.

Proof. Since L is in AM, there is a language M_0 in NP and a polynomial p_0 such that, for all strings x, the fraction of strings y of length $p_0(|x|)$ that satisfy $x \circ y \in M_0$ is (i) at least $\frac{2}{3}$ for x in L, and (ii) at most $\frac{1}{3}$ for x not in L.

We next explain how to reduce the error probability. Let M be the set of strings $x \circ y_1 \circ \cdots \circ y_{10q(|x|)}$ such that a majority of the strings $x \circ y_i$ are in M_0 Let $p(n) = 10q(n)p_0(n)$ If x is in L, then, by Chernoff's bound [4], the probability that a majority of strings $x \circ y_i$ will lie in M_0 is at least $1 - 2^{-q(|x|)}$. Similarly, for x not in L, the probability that a majority of strings $x \circ y_i$ will lie in M_0 is at most $2^{-q(|x|)}$.

It remains to show that M lies in NP. But the strings y_i that form a majority can be guessed and verified, since M_0 is in NP. Thus, M lies in NP. □

We now derive consequences of the (probably false) hypothesis that co-NP lies in AM. The next lemma shows that this hypothesis implies that co-AM is contained in AM. The main idea behind the proof is that two consecutive 'random quantifiers' can be collapsed into one.

2.2. Lemma. *If* co-NP *is contained in* AM, *then* co-AM *is contained in* AM.

Proof Suppose that co-NP is contained in AM Let L be a language in co-AM. By performing an amplification as in Lemma 2.1, there is a language M in co-NP and a polynomial p such that, for all strings x, the fraction of strings y of length $p(|x|)$ that satisfy $x \circ y \in M$ is (i) at least $\frac{9}{10}$ for x in L, and (ii) at most $\frac{1}{10}$ for x not in L

The hypothesis that co-NP is contained in AM implies that M lies in AM. Again, by performing an amplification, there is a language N in NP and a polynomial q such that, for all strings $x \circ y$, the fraction of strings z of length $q(|x| + |y|)$ that satisfy $x \circ y \circ z \in N$ is (i) at least $\frac{9}{10}$ for $x \circ y$ in M, and (ii) at most $\frac{1}{10}$ for $x \circ y$ not in M.

Finally, we combine the facts about languages M and N to show that L is in AM If x is in L, then the fraction of strings y of length $p(|x|)$ that satisfy $x \circ y \in M$ is at least $\frac{9}{10}$ For these strings y, the fraction of strings z of length $q(|x| + |y|)$ that satisfy $x \circ y \circ z \in N$ is at least $\frac{9}{10}$ Thus, for x in L, the fraction of strings $y \circ z$ such that y is of length $p(|x|)$ and z is of length $q(|x| + p(|x|))$ that satisfy $x \circ y \circ z \in N$ is at least $(\frac{9}{10})^2 \geqslant \frac{2}{3}$ Similar reasoning applies for x not in L to show that, for all strings x, the fraction of strings $y \circ z$ such that y has length $p(|x|)$ and z has length $q(|x| + p(|x|))$ that satisfy $x \circ y \circ z \in N$ is at most $1 - (\frac{9}{10})^2 \leqslant \frac{1}{3}$ for x not in L. Since the language N is in NP, the language L is in AM □

Using the quantifier notation of Hinman and Zachos [11], Zachos [17] and Zachos and Fürer [18], the above proof can be written as follows. The hypothesis is that co-NP = $(\forall/\exists)$ is contained in AM = $(\forall\exists/\exists^+\forall)$ Then, co-AM = $(\exists^+\forall/\forall\exists)$ is contained in $(\exists^+\forall\exists/\forall\exists^+\forall)$, which is contained in $(\forall\exists^+\exists/\exists^+\forall\forall)$ by a property of BPP, which is contained in $(\forall\exists/\exists^+\forall) =$ AM by collapsing two adjacent quantifiers.

We continue deriving consequences of the hypothesis that co-NP is contained in AM. The next theorem, the main result of this paper, shows that this hypothesis implies that the entire polynomial-time hierarchy collapses to $\mathsf{AM} \subseteq \Pi_2^p$. The proof uses a 'quantifier interchange' argument, similar to the one used by Babai [2]

2.3. Theorem. *If* co-NP *is contained in* AM, *then the polynomial-time hierarchy is contained in* $\mathsf{AM} \subseteq \Pi_2^p$.

Proof. Suppose that co-NP is contained in AM. We will show, for all $k \geqslant 1$, that the class Σ_k^p is contained in AM. The proof is by induction on k. The case k = 1 is trivial, since $\Sigma_1^p = \mathsf{NP} \subseteq \mathsf{AM}$.

Assume by induction, for some $k \geqslant 2$, that the class Σ_{k-1}^p is contained in AM, we will show that this implies that the class Σ_k^p is also contained in AM. Let L be a language in Σ_k^p. By the definition of Σ_k^p, there is a language M in Π_{k-1}^p and a polynomial p such that, for all strings x, the string x is in L iff there is a string y of length $p(|x|)$ that satisfies $x \circ y \in M$. The induction hypothesis implies that the language M belongs to co-AM, and thus, by Lemma 2.2, the language M also belongs to AM. Performing an amplification as in Lemma 2.1, there is a language N in NP and a polynomial q such that, for all strings $x \circ y$, the fraction of strings z of length $q(|x| + p(|x|))$ that satisfy $x \circ y \circ z \in N$ is (i) at least $1 - \frac{1}{3} \times 2^{-p(|x|)}$ for $x \circ y$ in M, and (ii) at most $\frac{1}{3} \times 2^{-p(|x|)}$ for $x \circ y$ not in M.

We combine the statements above to show that L is in AM Define the language N_0 so that $x \circ z \in N_0$ iff there is a string y of length $p(|x|)$ that satisfies $x \circ y \circ z \in N$. The statements above imply that, for all strings x, the fraction of strings z of length $q(|x| + p(|x|))$ that satisfy $x \circ z \in N_0$ is (i) at least $1 - \frac{1}{3} \times 2^{-p(|x|)} \geqslant \frac{2}{3}$ for x in L, and (ii) at most $2^{p(|x|)} \times (\frac{1}{3} \times 2^{-p(|x|)}) = \frac{1}{3}$ for x not in L. Since N is in NP, the language N_0 is also in NP. Hence, L is in AM. □

In the quantifier notation of Hinman and Zachos [11], Zachos [17] and Zachos and Fürer [18], this proof can be written as follows.

By induction, assume that Σ_{k-1}^p is contained in $\mathsf{AM} = (\forall\exists/\exists^+\forall)$. Then, Σ_k^p is contained in $(\exists\exists^+\forall/\forall\forall\exists)$, which is contained in $(\exists\forall\exists/\forall\exists^+\forall)$ by Lemma 2.2, which is contained in $(\forall\exists\exists/\exists^+\forall\forall)$ since $\mathsf{MA} \subseteq \mathsf{AM}$, which equals $(\forall\exists/\exists^+\forall) = \mathsf{AM}$ by collapsing two adjacent quantifiers

3. Is graph isomorphism NP-complete?

In this section we show that if the Graph Isomorphism problem is NP-complete, then the polynomial-time hierarchy collapses to the second level. The proof is a corollary of Theorem 2 3 and a result of Goldreich, Micali and Wigderson [8]

The *Graph Isomorphism* problem takes as input two graphs, and decides if the two graphs are isomorphic. The Graph Isomorphism problem is clearly in NP, but is not known to be in co-NP. Goldreich, Micali and Wigderson [8] proved that the Graph Isomorphism problem is in co-AM. They proved this result by constructing an interactive proof of two interactions, using hidden coins, for the complement of the Graph Isomorphism problem. Applying the main result of Goldwasser and Sipser [10], they deduce that Graph Isomorphism is in co-AM We prove the following corollary, which shows that the Graph Isomorphism problem is unlikely to be NP-complete

3.1. Corollary. *If the Graph Isomorphism problem is NP-complete, then the polynomial-time hierarchy is contained in* $\mathbf{AM} \subseteq \Pi_2^p$.

Proof. Suppose that the Graph Isomorphism problem is NP-complete. This assumption implies that all of NP will lie in co-AM, since Goldreich, Micali and Wigderson [8] showed that the Graph Isomorphism problem is in co-AM. But NP contained in co-AM is equivalent to co-NP contained in AM. By Theorem 2.3, this implies that the polynomial-time hierarchy collapses to $\mathbf{AM} \subseteq \Pi_2^p$ □

In the remainder, we show that the Graph Automorphism problem is in co-AM, as well as provide an alternative proof that the Graph Isomorphism problem is in co-AM. The *Graph Automorphism* problem takes as input a single graph and decides if the graph has a nontrivial automorphism.

We first introduce a new complexity class, called ANP (for 'approximate' NP). A language L is in ANP iff there is a language M in NP, a polynomial-time computable function θ from the nonnegative integers to the positive integers, and a polynomial p such that, for all strings x, the number of strings y of length $p(|x|)$ that satisfy $x \circ y \in M$ is (i) at least $2\theta(|x|)$ for x in L, and (ii) at most $\theta(|x|)$ for x not in L. Notice that the definition is in terms of the number of strings y instead of the fraction of strings y, but these are equivalent up to a scaling factor.

It is clear that AM is contained in ANP, by setting $\theta(n) = \lfloor \frac{1}{3} \times 2^{p(n)} \rfloor$. What is more surprising is that the converse is also true, i.e., the class ANP is contained in AM. The proof uses universal hash functions, first applied to complexity theory by Sipser [15] Babai [2] and Goldwasser and Sipser [10] applied universal hash functions to obtain approximate lower bounds on the cardinality of sets using Arthur–Merlin protocols

3.2. Theorem. *The complexity classes* ANP *and* AM *are equal.*

Proof. As observed above, the class AM is in ANP. To show the converse, let L be a language in ANP. Let M be as promised in the definition of ANP.

We now describe the hash function technique to show that L is in AM. Let $q(n) = \lceil \log_2 \theta(n) \rceil + 2$. Let h be a $q(|x|) \times p(|x|)$ matrix of 0's and 1's and let z be a string of length $q(|x|)$. Define the language N to be the set of strings $x \circ h \circ z$ such that there is a string y of length $p(|x|)$ that satisfies $x \circ y \in M$ and $hy = z$ (where the matrix multiplication is performed mod 2). It is clear that N is in NP.

Using the inclusion–exclusion principle from elementary probability, we can show that, for all strings x, the fraction of strings $h \circ z$ that satisfy $x \circ h \circ z \in N$ is (i) at least

$$2\theta(|x|)2^{-q(|x|)} - \binom{2\theta(|x|)}{2}4^{-q(|x|)}$$

for x in L, and (ii) at most $\theta(|x|)2^{-q(|x|)}$ for x not in L. These two bounds lead to a constant separation and, by performing an amplification, we can achieve a $\frac{1}{3}$–$\frac{2}{3}$ separation Thus, L is in AM. □

We now use the complexity class ANP to classify the Graph Automorphism and Graph Isomorphism problems. Goldreich, Micali and Wigderson

[8] showed that the Graph Isomorphism problem is in co-AM. Below we give an alternative proof, as well as observe that the Graph Automorphism problem is also in co-AM

3.3. Theorem. *The Graph Isomorphism and Graph Automorphism problems are in* NP ∩ co-AM.

Proof. Both problems are clearly in NP. To show that the Graph Automorphism problem is in co-AM, suppose the input graph is G, and consider the set $A = \{H : H \cong G\}$. If G has no nontrivial automorphisms, then A has cardinality n!; otherwise, it has cardinality at most $\frac{1}{2}$n! This observation shows that the Graph Automorphism problem is in co-ANP, and thus by Theorem 3.1 is also in co-AM.

To show that the Graph Isomorphism problem is in co-AM, suppose the two input graphs are G_1 and G_2, and consider the set

$$B = \{(H, \pi) . (H \cong G_1 \vee H \cong G_2) \wedge \pi \in \mathrm{Aut}(H)\}.$$

If the graphs G_1 and G_2 are not isomorphic, then B has cardinality 2n!; otherwise, it has cardinality n!. Hence, the Graph Isomorphism problem is in co-ANP = co-AM □

Acknowledgment

We thank Oded Goldreich and Shafi Goldwasser for useful discussions.

References

[1] W Aiello, S Goldwasser and J Hastad, On the power of interaction, Proc 27th IEEE Symp on Foundations of Computer Science (1986) 368–379

[2] L Babai, Trading group theory for randomness, Proc 17th ACM Symp on Theory of Computing (1985) 421–429

[3] C G Bennett and J Gill, Relative to a random oracle A, $P^A \neq NP^A \neq$ co-NP^A with probability 1, SIAM J Comput 10 (1) (1981) 96–113

[4] H Chernoff, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, Ann Math Statist 23 (1952) 493–509

[5] A Condon and R Ladner, Probabilistic game automata, Proc Structure in Complexity Theory Conf, Lecture Notes in Computer Science, Vol 223 (Springer, Berlin, 1986) 144–162

[6] L Fortnow and M Sipser, Private communication

[7] J Gill, Computational complexity of probabilistic Turing machines, SIAM J Comput 6 (4) (1977) 675–695

[8] O Goldreich, S Micali and A Wigderson, Proofs that yield nothing but their validity and a methodology of cryptographic protocol design, Proc 27th IEEE Symp on Foundations of Computer Science (1986) 174–187

[9] S Goldwasser, S Micali and C Rackoff, The knowledge complexity of interactive proof systems, Proc 17th ACM Symp on Theory of Computing (1985) 291–304

[10] S Goldwasser and M Sipser, Private coins versus public coins in interactive proof systems, Proc 18th ACM Symp on Theory of Computing (1986) 59–68

[11] P Hinman and S Zachos, Probabilistic machines, oracles and quantifiers, Proc Oberwolfach Recursion-Theoretic Week, Lecture Notes in Mathematics, Vol 1141 (Springer, Berlin, 1984) 159–192

[12] K Ko, Some observations on probabilistic algorithms and NP-hard problems, Inform Process Lett 14 (1) (1982) 39–43

[13] U Schoning, A low and high hierarchy within NP, J Comput System Sci 27 (1983) 14–28

[14] U Schoning, Graph isomorphism is in the low hierarchy, Preprint, 1986

[15] M Sipser, A complexity theoretic approach to randomness, Proc 15th ACM Symp on Theory of Computing (1983) 330–335

[16] L J Stockmeyer, The polynomial-time hierarchy, Theoret. Comput Sci 3 (1) (1976) 1–22

[17] S Zachos, Probabilistic quantifiers, adversaries, and complexity classes An overview, Proc Structure in Complexity Theory Conf, Lecture Notes in Computer Science, Vol 223 (Springer, Berlin, 1986) 383–400

[18] S Zachos and M Furer, Probabilistic quantifiers vs distrustful adversaries, Preprint, 1985