Electronic Colloquium on Computational Complexity, Report No. 151 (2011)

Is the Valiant-Vazirani Isolation Lemma Improvable?

Valentine Kabanets*
School of Computing Science
Simon Fraser University
Burnaby, BC
Canada V5A 1S6
kabanets@cs.sfu.ca

Osamu Watanabe
Tokyo Institute of Technology
Tokyo, Japan
watanabe@is.titech.ac.jp

November 9, 2011

Abstract

The Valiant-Vazirani Isolation Lemma [*TCS*, vol. 47, pp. 85–93, 1986] provides an efficient procedure for isolating a satisfying assignment of a given satisfiable circuit: given a Boolean circuit C on n input variables, the procedure outputs a new circuit C' on the same n input variables with the property that the set of satisfying assignments for C' is a subset of those for C, and moreover, if C is satisfiable then C' has exactly one satisfying assignment. The Valiant-Vazirani procedure is *randomized*, and it produces a uniquely satisfiable circuit C' with probability $\Omega(1/n)$.

Is it possible to have an efficient *deterministic* witness-isolating procedure? Or, at least, is it possible to improve the success probability of a randomized procedure to $\Omega(1)$? We argue that the answer is likely *'No'*. More precisely, we prove that

1. a non-uniform deterministic polynomial-time witness-isolating procedure exists *if and only if* $\mathsf{NP} \subseteq \mathsf{P/poly}$, and
2. if there is a randomized polynomial-time witness-isolating procedure with success probability bigger than 2/3, then $\mathsf{coNP} \subseteq \mathsf{NP/poly}$.

Thus, an improved witness-isolating procedure would imply the collapse of the Polynomial-Time Hierarchy. Finally, we consider a black-box setting of witness isolation (generalizing the setting of the Valiant-Vazirani Isolation Lemma), and give the upper bound $O(1/n)$ on the success probability for a natural class of randomized witness-isolating procedures.

1 Introduction

The Isolation Lemma of Valiant and Vazirani [VV86] (as well as the related Isolation Lemma of Mulmuley, Vazirani, and Vazirani [MVV87]) is a basic tool with many important applications in complexity theory (see, e.g., [Tod91, BDCGL92, RA00] for just a few such applications). This lemma provides an efficient randomized algorithm to "isolate" a single object from a collection of objects satisfying a given efficiently decidable property. More precisely, given a Boolean circuit $C(x_1, \ldots, x_n)$, the algorithm produces a new Boolean circuit $C'(x_1, \ldots, x_n)$ such that *(i)* with

*Most of this research was done during a visit to the Tokyo Institute of Technology in the Summer of 2011.

ISSN 1433-8092

probability one (over internal randomness of the algorithm), the satisfying assignments for C' also satisfy C, and *(ii)* if C is satisfiable, then, with probability $\Omega(1/n)$, C' has exactly one satisfying assignment. Thus, in case C is satisfiable, the unique satisfying assignment for C' is an "isolated" assignment from among the satisfying assignments for C.

The obvious question (raised already in [VV86]) is whether efficient *deterministic* isolation is possible. That is, is there a deterministic polynomial-time algorithm that would map an input circuit $C(x_1, \ldots, x_n)$ to an output circuit $C'(x_1, \ldots, x_n)$ so that *(i)* if C is unsatisfiable, then so is C', and *(ii)* if C is satisfiable, then C' has exactly one satisfying assignment, and moreover, the unique satisfying assignment for C' also satisfies C? Another natural question is whether the success probability $\Omega(1/n)$ for randomized isolation can be improved to, say, constant probability. We show that the answer to both questions is likely negative.

1.1 Our results

If $\mathsf{NP} = \mathsf{P}$, then efficient deterministic isolation is trivially possible: Given a circuit C, one can use the standard "search-to-decision" reduction to find in deterministic polynomial time some satisfying assignment w for C, and then construct a circuit C' so that C' accepts the single input w. Naively, it seems impossible to produce, efficiently deterministically, a circuit C' with exactly one satisfying assignment that also satisfies C, without actually finding such an assignment efficiently deterministically. In other words, naively it seems that *efficient deterministic isolation must be equivalent to* $\mathsf{NP} = \mathsf{P}$.

We show that such an equivalence is actually true in the *non-uniform* setting! We prove that if there is a non-uniform family of polynomial-size circuits that achieve deterministic isolation (in the sense defined above), then every language in NP can be decided by a non-uniform family of polynomial-size circuits, i.e., $\mathsf{NP} \subseteq \mathsf{P/poly}$. Since the standard "search-to-decision" reduction for NP can be run also in the non-uniform setting, we immediately get the other direction: if $\mathsf{NP} \subseteq \mathsf{P/poly}$, then non-uniform efficient deterministic isolation is possible.

Given that deterministic isolation is unlikely, what can we say about the existence of a better randomized isolation algorithm? A natural question is whether one can achieve randomized isolation with success probability better than $\Omega(1/n)$ achieved in [VV86]. For example, can one achieve (large) constant success probability?

We show that the answer is likely negative. We prove that if there is a randomized isolation algorithm with success probability greater than $2/3$, then $\mathsf{coNP} \subseteq \mathsf{NP/poly}$ (and, consequently, the polynomial-time hierarchy collapses).

Finally, we consider a natural black-box setting for isolation (generalizing the setting of [VV86]), and observe that $O(1/n)$ is an upper bound on success probability for randomized isolation in this black-box setting.

1.2 Related work

The problem of efficient deterministic isolation is related to the problem of multi-valued vs. single-valued NP-computable functions [Sel94], which received considerable attention in the 1990's. In fact, it easily follows from the work of Hemaspaandra et al. [HNOS96] that efficient deterministic isolation yields collapse of the polynomial-time hierarchy. More precisely, [HNOS96] implies that efficient deterministic isolation leads to $\mathsf{NP} \subseteq (\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$, which in turn is known to imply the collapse of the polynomial-time hierarchy to the second level (in fact, to $\mathsf{ZPP}^{\mathsf{NP}}$). In contrast, we

prove that the same assumption implies $\mathsf{NP} \subseteq \mathsf{P/poly}$. This conclusion is stronger, and, as observed above, is actually equivalent to the existence of efficient non-uniform deterministic isolation.

The problem of efficient deterministic isolation as defined above is different from the problem of *derandomizing* the Valiant-Vazirani Isolation Lemma as studied, e.g., in [KM02]. In the setting of [KM02], randomized isolation is defined via the existence of an efficient randomized algorithm that maps an input circuit C to a *list* of circuits $C_1, \ldots, C_m$ so that if C is unsatisfiable then every C_i is also unsatisfiable, and if C is satisfiable then, with high probability, at least one of the circuits C_i is uniquely satisfiable; this kind of randomized isolation also follows from the Valiant-Vazirani Isolation Lemma.

Derandomizing such isolation means designing an efficient deterministic algorithm that produces such a list $C_1, \ldots, C_m$. One of the results in [KM02] is that this kind of derandomization is likely (as it follows from some plausible circuit complexity assumptions). However, if we want to get a *single* circuit C' that is uniquely satisfiable if C is satisfiable, no better way is known other than to pick one of the circuits on the list at random. But then we end up with a randomized isolation procedure with inverse-polynomial success probability. Thus, while it may be possible to design an efficient deterministic algorithm mapping a given input circuit C to a *list* of circuits $C_1, \ldots, C_m$ achieving isolation in the sense of [KM02], it is unlikely that there is an efficient deterministic isolation mapping C to a *single* circuit C'. Also, by our results, it is unlikely that there is a [KM02]-style randomized isolation algorithm mapping a satisfiable circuit C to a list of circuits where more than $2/3$ of the circuits on the list are uniquely satisfiable.

The question of existence of efficient deterministic isolation is also related to the question whether every NP language can be decided by a nondeterministic polynomial-time Turing machine that has exactly one accepting computation for every string in the language, i.e., whether $\mathsf{NP} = \mathsf{UP}$. Clearly, if deterministic polynomial-time isolation is possible, then $\mathsf{NP} = \mathsf{UP}$. However, the converse is not necessarily true. It remains an open question whether the assumption $\mathsf{NP} = \mathsf{UP}$ yields any unexpected consequences, e.g., if it implies any collapse of the polynomial-time hierarchy.

1.3 Our techniques

Our proof arguments use the notion of p-selectivity [Sel79] and its generalizations [HNOS96], as well as some new ideas. Below we sketch the proof of one of our main results that efficient deterministic isolation implies $\mathsf{NP} \subseteq \mathsf{P/poly}$.

Suppose there is an efficient deterministic algorithm A achieving isolation for Boolean circuits. That is, given a circuit $C(x_1, \ldots, x_n)$, our algorithm outputs a circuit $C'(x_1, \ldots, x_n)$ such that *(i)* if C is unsatisfiable, then so is C', and *(ii)* if C is satisfiable, then C' has exactly one satisfying assignment w, and moreover, w is also a satisfying assignment for the original circuit C.

Such an algorithm yields a nondeterministic procedure that can "uniquely select" a satisfiable circuit from a pair of circuits $C_1(x_1, \ldots, x_n)$ and $C_2(x_1, \ldots, x_n)$ when $C_1 \vee C_2$ is satisfiable: Apply the algorithm A to the circuit $C_1 \vee C_2$, getting a circuit C'; nondeterministically guess the unique satisfying assignment w for C'; if w satisfies only one of the circuits C_1 and C_2, then select that circuit; if w satisfies both circuits, then select the lexicographically smaller one. Note that even though the described procedure is nondeterministic, it always produces the same answer, i.e., it is a *single-valued* NP-computable function.

The described selection procedure shows that the language Circuit SAT is essentially p-selective, with the only difference that the standard definition of p-selectivity [Sel79] requires that the selection procedure be deterministic. Ko [Ko83] showed that every p-selective language is in $\mathsf{P/poly}$. As

observed in [HNOS96], Ko's techniques can be applied also in the case of a single-valued NP-computable selection procedure. In our case, this yields that Circuit SAT is in $(\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$, where the advice is used to encode a polynomial number of satisfiable circuits. Since Circuit SAT is NP-complete, we get that $\mathsf{NP} \subseteq (\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$.

To get the deeper collapse of NP to P/poly, we apply our isolation algorithm A recursively! Our new selection procedure takes as input a pair of circuits C_1 and C_2, applies the algorithm A to both of them, getting the new circuits $C_1' = A(C_1)$ and $C_2' = A(C_2)$, and then runs the original selection procedure on C_1' and C_2'. As before, we get a non-uniform algorithm that needs a polynomial number of satisfiable circuits as advice. The crucial point is that all of these circuits are *uniquely* satisfiable (as they are of the form $A(C)$ for some satisfiable circuit C). Thus we can also include in the advice all satisfying assignments for these circuits. This new advice (still of polynomial size) turns out to be sufficient in order to decide Circuit SAT in *deterministic* polynomial time, yielding that $\mathsf{NP} \subseteq \mathsf{P}/\mathsf{poly}$.

Remainder of the paper. Section 2 contains basic definitions. We prove our conditional impossibility results for determinisitc and randomized isolation in Section 3. In Section 4, we prove our unconditional impossibility result for "black-box" randomized isolation. We give concluding remarks in Section 5.

2 Preliminaries

We use standard definitions and notation for complexity classes such as P, NP, and P/poly (see, e.g., [AB09]). By a slight abuse of notation, we extend the notation P, NP, and P/poly to not necessarily Boolean functions from $\{0,1\}^*$ to $\{0,1\}^*$. Thus, a function $f : \{0,1\}^* \to \{0,1\}^*$ is called P-computable if it is computable by some deterministic polynomial-time algorithm. It is NP-computable if there is a nondeterministic polynomial-time algorithm such that, for every $x \in \{0,1\}^*$, the algorithm on x has at least one accepting computation, and every accepting computation produces the same value $f(x)$. Finally, the function f is called P/poly-computable if it can be computed by a family of polynomial-size circuits.

We will use the NP-complete problem Circuit SAT: Given a Boolean circuit $C(x_1, \ldots, x_n)$, decide if it is satisfiable. We say that a circuit $C(x_1, \ldots, x_n)$ is uniquely satisfiable if there is exactly one binary string $w \in \{0,1\}^n$ that satisfies C. We will encode circuits using binary strings, and will use the length of such an encoding as the *size* of the circuit. For a circuit C, we will denote its size by $|C|$. We assume that the encoding is efficient so that, e.g., for circuits C and D of size m each, the size of the circuit $C \vee D$ is $O(m)$.

By *isolation*[1], we will mean an efficient algorithm mapping a given Boolean circuit $C(x_1, \ldots, x_n)$ to a Boolean circuit $C'(x_1, \ldots, x_n)$ such that *(i)* every satisfying assignment for C' also satisfies C, and *(ii)* if C is satisfiable then C' is uniquely satisfiable. Depending on the type of the algorithm, we have *deterministic* uniform (P-computable) or non-uniform (P/poly-computable) isolation. We also have *randomized* isolation where a randomized polynomial-time algorithm mapping C to C' must satisfy condition *(i)* with probability one (over its internal randomness), and condition *(ii)*

[1]It is possible to talk about witness isolation for arbitrary "witness-checking" predicates $R(x, y)$, where R is P-computable and $|y| \in |x|^{O(1)}$. However, for simplicity of presentation, we restrict ourselves to the generic NP-complete problem Circuit SAT.

holds with some probability $\delta = \delta(n)$; we shall call such a reduction a *randomized δ-isolation*. Note that Valiant and Vazirani [VV86] proved the existence of randomized $\Omega(1/n)$-isolation.

3 Isolation is unlikely to exist

Here we show that both deterministic (non-uniform) isolation and randomized isolation would lead to surprising complexity-theoretic consequences, and hence are unlikely to exist.

3.1 Deterministic isolation

Here we will prove the following.

Theorem 3.1. *If* $\mathsf{P/poly}$*-computable isolation exists, then* $\mathsf{NP} \subseteq \mathsf{P/poly}$.

By the standard "search-to decision" reduction for Circuit SAT, we get that the assumption $\mathsf{NP} \subseteq \mathsf{P/poly}$ implies the existence of the $\mathsf{P/poly}$-computable isolation. Together with Theorem 3.1, this yields the following.

Corollary 3.2. $\mathsf{P/poly}$*-computable isolation exists iff* $\mathsf{NP} \subseteq \mathsf{P/poly}$.

Proof of Theorem 3.1. We first show that Circuit SAT $\in (\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$ (using the techniques similar to those in [HNOS96]). and then we explain how to get the stronger conclusion that Circuit SAT $\in \mathrm{P}/\mathsf{poly}$.

Let F be the assumed $\mathsf{P/poly}$-computable isolation reduction for Circuit SAT. Consider the following $\mathsf{NP/poly}$-computable "selection" algorithm which selects one of the two given Boolean circuits C_1 and C_2, under the assumption that at least one of C_1 and C_2 is satisfiable.

> **Selection algorithm $\mathcal{A}$:** Given an unordered pair of distinct (encodings of) Boolean circuits $C_1(x_1, \ldots, x_n)$ and $C_2(x_1, \ldots, x_n)$ (where C_1 is lexicographically smaller than C_2) such that at least one of the circuits is satisfiable, define the satisfiable circuit $C(x_1, \ldots, x_n) = C_1(x_1, \ldots, x_n) \vee C_2(x_1, \ldots, x_n)$. Apply the isolation reduction F to C, getting a uniquely satisfiable circuit $C'(x_1, \ldots, x_n)$. Nondeterministically guess the unique satisfying assignment $w \in \{0,1\}^n$ for C'. If $C_1(w) = 1$, then output C_1; otherwise output C_2.

We make several observations about the described selection algorithm $\mathcal{A}$. First, given the advice necessary for the isolation reduction F, our nondeterministic selection algorithm has exactly one accepting computation (and so defines a single-valued NP function). Secondly, the selection algorithm always outputs a satisfiable circuit, since, by the definition of isolation, the unique satisfying assignment w of the circuit C' must be also satisfying for C, and so must be satisfying for at least one of C_1 and C_2; if w satisfies both C_1 and C_2, we break the tie by choosing the lexicographically smaller of the two circuits.

Claim 3.3. *Let S be any non-empty set of satisfiable circuits on n inputs. Then there is a circuit $C^* \in S$ such that $\mathbf{Pr}_{C \in S \setminus \{C^*\}}[\mathcal{A}(C^*, C) = C] \geqslant 1/2$.*

Proof. Consider the following random experiment: Pick a uniformly random circuit $C^0 \in S$, then a uniformly random circuit $C^1 \in S \setminus \{C^0\}$, and finally pick a uniformly random bit $b \in \{0,1\}$.

Observe that once C^0 and C^1 are chosen, the output of the selection algorithm $\mathcal{A}(C^0, C^1)$ is uniquely defined. Suppose that $\mathcal{A}(C^0, C^1) = C^r$ for some $r \in \{0, 1\}$. Then the probability that our random experiment chooses $b = r$ is exactly $1/2$. Since the choice of b is independent of the choice of C^0 and C^1, we get by an averaging argument that there is some fixed value $b^* \in \{0, 1\}$ such that $\mathbf{Pr}_{C^0,C^1}[\mathcal{A}(C^0, C^1) = C^{b^*}] \geqslant 1/2$. Define $\bar{b} = 1 - b^*$. We have that with probability at least $1/2$ over the choice of C^0 and C^1, the algorithm $\mathcal{A}$ does *not* choose $C^{\bar{b}}$.

Note that the random choices of C^0 and C^1 made by our random experiment induce the uniform distribution over all 2-element subsets of S. The same uniform distribution is induced if our random experiment were first to choose C^1 uniformly from S, and then C^0 uniformly from $S \setminus \{C^1\}$. Since the selection algorithm $\mathcal{A}$ treats its input as an unordered pair of circuits, we get $\mathbf{Pr}_{C^{\bar{b}} \in S, C^{b^*} \in S \setminus \{C^{\bar{b}}\}}[\mathcal{A}(C^{\bar{b}}, C^{b^*}) \neq C^{\bar{b}}] \geqslant 1/2$. By averaging, we can fix the choice of $C^{\bar{b}}$ to some circuit C^* so that $\mathbf{Pr}_{C \in S \setminus \{C^*\}}[\mathcal{A}(C^*, C) = C] \geqslant 1/2$, as required. □

Now suppose we wish to solve Circuit SAT for circuits $C(x_1, \ldots, x_n)$ of the binary encoding size m (for some $m \geqslant n$). Set S to be the set of all satisfiable circuits of size m on n inputs; note that $|S| \leqslant 2^m$. By Claim 3.3, there is a satisfiable circuit $C_1 \in S$ such that for at least $1/2$ of the remaining satisfiable circuits C in S, the algorithm $\mathcal{A}(C_1, C) = C$. Call such circuits C *covered* by C_1. Remove the covered circuits from S, and apply Claim 3.3 again, getting another satisfiable circuit C_2 that covers at least $1/2$ of the remaining circuits in S. After at most m iterations, we get $t \leqslant m$ satisfiable circuits $C_1, \ldots, C_t$ so that each satisfiable circuit $C(x_1, \ldots, x_n)$ of size m is covered by some C_i for $1 \leqslant i \leqslant t$. On the other hand, if a given circuit $C(x_1, \ldots, x_n)$ of size m is covered by some C_i, $1 \leqslant i \leqslant t$, then we know that C is satisfiable (since $\mathcal{A}$ always selects a satisfiable circuit). Thus, we get

$$C(x_1, \ldots, x_n) \text{ is satisfiable } \Leftrightarrow \exists i \in \{1, \ldots, t\}\ \mathcal{A}(C_i, C) = C. \tag{1}$$

Equivalently,

$$C(x_1, \ldots, x_n) \text{ is unsatisfiable } \Leftrightarrow \forall i \in \{1, \ldots, t\}\ \mathcal{A}(C_i, C) = C_i. \tag{2}$$

Let us fix the advice needed by the isolation reduction F for circuits of size m on n inputs; this advice has size polynomial in m. Let us add to the advice the list of circuits $C_1, \ldots, C_t$ defined above; note that the new advice is still of size polynomial in m since $t \leqslant m$ and each circuit C_i is of size at most m. Once the advice is fixed, we can use it to decide in NP both Circuit SAT by Eq. (1) and Circuit UNSAT by Eq. (2), since $\mathcal{A}$ is single-valued NP-computable function and $t \leqslant m$. So we get that Circuit SAT $\in (\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$.

Next, to show that Circuit SAT $\in \mathsf{P}/\mathsf{poly}$, we modify our argument as follows. Suppose we want to decide Circuit SAT for circuits $C(x_1, \ldots, x_n)$ of size m. For each such circuit $C(x_1, \ldots, x_n)$, define the circuit $C'(x_1, \ldots, x_n) = F(C_1)$. That is, C' is obtained from C by applying the isolation reduction F to C. Let m' be the size bound for the new circuits $C' = F(C)$; note that m' is polynomial in m.

Clearly, C is satisfiable iff C' is satisfiable. Moreover, we have that if C' is satisfiable, then it is uniquely satisfiable. Consider the set S of $D' = F(C)$ for all satisfiable circuits C. By applying Claim 3.3 to S for at most m' times, we get a list of satisfiable circuits $C'_1, \ldots, C'_t$, for $t \leqslant m'$, so that each satisfiable circuit C' in S is covered by some circuit C'_i on the list. Let $w_1, \ldots, w_t \in \{0, 1\}^n$ be the unique satisfying assignments for the respective circuits $C'_1, \ldots, C'_t$. As before, we have

$$C \text{ is satisfiable } \Leftrightarrow C' = F(C) \text{ is satisfiable } \Leftrightarrow \exists i \in \{1, \ldots, t\}\ \mathcal{A}(C', C'_i) = C'. \tag{3}$$

Claim 3.4. *Let C'_i be a fixed circuit of size m', with the unique satisfying assignment w_i. Then there is a* $\mathsf{P/poly}$ *algorithm to decide, for any uniquely satisfiable circuit C' of size m' (where C' is different from C'_i), whether $\mathcal{A}(C', C'_i) = C'$. Moreover, the advice needed by the algorithm is exactly the advice needed for the isolation reduction F on circuits of the size $\max\{|C'_i \vee C'|, |C' \vee C'_i|\} \in O(m')$ and the unique assignment w_i of C'_i.*

Proof. Let us first assume that C' is lexicographically smaller than C'_i. By definition, we have $\mathcal{A}(C', C'_i) = C'$ iff the unique satisfying assignment of the circuit $D = F(C' \vee C'_i)$ satisfies C'. We have two cases to consider. If $D(w_i) = 0$, then the unique satisfying assignment of D is some string other than w_i, and so it must satisfy C'. Hence, $\mathcal{A}$ will select C'. If, on the other hand, $D(w_i) = 1$, then for $\mathcal{A}$ to select C' it must be the case that $C'(w_i) = 1$. Overall, $\mathcal{A}(C', C'_i) = C'$ iff either $D(w_i) = 0$ or $D(w_i) = C'(w_i) = 1$.

Now let us assume that C'_i is lexicographically smaller than C'. Let $D = F(C'_i \vee C')$. By definition, $\mathcal{A}(C', C'_i) = C'$ iff the unique satisfying assignment of D satisfies C' but not C'_i, which happens iff $D(w_i) = 0$. Thus, in this case, $\mathcal{A}(C', C'_i) = C'$ iff $D(w_i) = 0$.

Finally, to decide if $\mathcal{A}(C', C'_i) = C'$, we do as follows: If C' is lexicographically smaller than C'_i, then output True iff $[D(w_i) = 0] \vee [D(w_i) = C'(w_i) = 1]$, where $D = F(C' \vee C'_i)$. If C' is lexicographically larger than C'_i, then output True iff $D(w_i) = 0$, where $D = F(C'_i \vee C')$.

Clearly, we can perform all the necessary computation in P, given the advice needed by F. □

Now, to decide if a an arbitrary circuit C of size m is satisfiable, we will use the algorithm suggested by Eq. (3). We need the advice that allows us to compute the isolation reduction F on circuits of size m (to compute $C' = F(C)$) and size $O(m')$ (to compute F inside Claim 3.4), as well as the circuits $C'_1, \ldots, C'_t$ and their unique satisfying assignments $w_1, \ldots, w_t$. Note that this whole advice is of size polynomial in m. Using Claim 3.4, we conclude that Circuit SAT is in $\mathsf{P/poly}$. □

Suppose we strengthen the requirement on the deterministic isolation to isolate a particular satisfying assignment, e.g., the lexicographically smallest one. Let us call isolation F *strong* if it satisfies the following additional property: if a given input circuit $C(x_1, \ldots, x_n)$ is satisfiable and if $w \in \{0,1\}^n$ is the lexicographically smallest satisfying assignment for C, then the circuit $F(C)$ is uniquely satisfied by w.

Theorem 3.5. *Suppose there is* P*-computable strong isolation F. Then* $\mathsf{NP} = \mathsf{P}$.

Proof. We can decide Circuit SAT in P as follows: Given a circuit $C(x_1, \ldots, x_n)$, check if $C(1, \ldots, 1) = 1$; if so, then output "Yes" and halt; otherwise, construct the circuit $\tilde{C}(x_1, \ldots, x_n) = [C(x_1, \ldots, x_n) \vee [x_1 = 1 \wedge \cdots \wedge x_n = 1]]$; apply F to $\tilde{C}$, getting the uniquely satisfiable circuit C'; if $C'(1, \ldots, 1) = 0$, then output "Yes", else output "No".

For correctness, suppose that $C(1, \ldots, 1) = 0$ yet C is satisfiable. Then $\tilde{C}$ has some satisfying assignment other than $1 \ldots 1$. Hence C' must be satisfiable by some assignment other than $1 \ldots 1$ (by our assumption that F isolates the lexicographically smallest satisfying assignment), and so $C'(1, \ldots, 1) = 0$ in that case (as C' is uniquely satisfiable). □

Since the standard "search-to-decision" reduction for Circuit SAT can produce the lexicographically smallest satisfying assignment for a given satisfiable circuit, we get the following.

Corollary 3.6. P*-computable strong isolation exists iff* $\mathsf{NP} = \mathsf{P}$.

3.2 Randomized isolation

Here we show that randomized isolation is also unlikely to exist.

Theorem 3.7. *If, for some constant $\delta > 2/3$, there exists randomized polynomial-time δ-isolation, then* $\mathsf{coNP} \subseteq \mathsf{NP/poly}$.

For the proof, we generally follow the same strategy as in the first half of the proof of Theorem 3.1, but with an important change in the definition of a selection algorithm, where we need to account for the fact that our isolation is no longer deterministic. We provide the details next.

Proof of Theorem 3.7. Suppose that, for some constant $\delta > 2/3$, there is a randomized polynomial-time algorithm F mapping a Boolean circuit $C(x_1, \ldots, x_n)$ to a Boolean circuit $C'(x_1, \ldots, x_n)$ so that *(i)* with probability one, the satisfying assignments of C' are also satisfying for C, and *(ii)* with probability δ, if C is satisfiable, then C' is uniquely satisfiable; here the probability is over the internal randomness of F. We would like to use this F in order to define a selection algorithm for choosing between a given pair of circuits $C_1(x_1, \ldots, x_n)$ and $C_2(x_1, \ldots, x_n)$, similarly to the proof of Theorem 3.1.

First we define a new randomized algorithm that, on a given circuit $C(x_1, \ldots, x_n)$ of size m, runs the reduction F for $t = O(m)$ times, and outputs a list of the obtained t circuits $C'_1(x_1, \ldots, x_n), \ldots, C'_t(x_1, \ldots, x_n)$. For a satisfiable circuit C, we expect to see at least $\delta \cdot t$ uniquely satisfiable circuits on our list. By the standard Chernoff bound, for some constant α, $2/3 < \alpha \leqslant \delta$, the probability of getting fewer than $\alpha \cdot t$ uniquely satisfiable circuits is less than 2^{-m}, if we choose $t = c \cdot m$ for a sufficiently large constant c. By averaging, there is a choice of randomness for the algorithm so that, for every satisfiable circuit $C(x_1, \ldots, x_n)$ of size m, the output list contains at least $\alpha \cdot t$ uniquely satisfiable circuits. With this good choice of randomness put into advice (of size polynomial in m), we get a $\mathsf{P/poly}$-computable deterministic algorithm F' mapping every satisfiable circuit C to a list of t circuits, where at least α fraction of the circuits on the list are uniquely satisfiable. Note that each of the remaining circuits on the list is either unsatisfiable or has more than one satisfying assignment.

Next we define the following $\mathsf{NP/poly}$-computable "selection" algorithm $\mathcal{A}'$.

> **Selection algorithm $\mathcal{A}'$:** Given an unordered pair of distinct (encodings of) Boolean circuits $C_1(x_1, \ldots, x_n)$ and $C_2(x_1, \ldots, x_n)$ (where C_1 is lexicographically smaller than C_2) such that at least one of the circuits is satisfiable, define the satisfiable circuit $C(x_1, \ldots, x_n) = C_1(x_1, \ldots, x_n) \vee C_2(x_1, \ldots, x_n)$. Apply the algorithm F' to C, getting a list of t circuits $C'_1(x_1, \ldots, x_n), \ldots, C'_t(x_1, \ldots, x_n)$. Nondeterministically guess a subset $I \subseteq \{1, \ldots, t\}$ of size $\alpha \cdot t$. For each $i \in I$, nondeterministically guess a string $w_i \in \{0,1\}^n$. If, for any $i \in I$, $C'_i(w_i) = 0$, then halt with rejection. Otherwise, if for every $i \in I$, $C_1(w_i) = 1$, then output C_1; else, if for every $i \in I$, $C_1(w_i) = 0$, output C_2; otherwise, output $\perp$.

We make several observations about the described selection algorithm $\mathcal{A}'$.

Claim 3.8. *On input C_1, C_2, with at least one of the circuits satisfiable, there is always an accepting computation of $\mathcal{A}'$ that outputs C_1, C_2, or $\perp$.*

Proof. There are at least α fraction of satisfiable (in fact, uniquely satisfiable) circuits on the list produced by F'. Thus there is a nondeterministic guess of the set I of satisfiable circuits. □

Claim 3.9. *If, on input (C_1, C_2), with at least one of circuits satisfiable, $\mathcal{A}'$ has an accepting computation that outputs C_i, for $i \in \{1, 2\}$, then C_i is satisfiable.*

Proof. If C_1 is output, then it is obviously satisfiable. If C_2 is output, we have w_i that satisfies $C_1 \vee C_2$ and falsifies C_1, and so w_i must satisfy C_2. □

Claim 3.10. *If, on input (C_1, C_2), with at least one of the circuits satisfiable, $\mathcal{A}$ has an accepting computation that outputs $\perp$, then both C_1 and C_2 are satisfiable.*

Proof. Note that, if $\perp$ is output, each w_i, for $i \in I$, must be satisfying for $C_1 \vee C_2$. If not all w_i's satisfy C_1, then some of them must satisfy C_2. Similarly, if not all w_i's falsify C_1, then some must satisfy C_1. □

Finally, the crucial property of the selection algorithm $\mathcal{A}'$ is given in the following claim.

Claim 3.11. *On input (C_1, C_2), with at least one the circuits satisfiable, $\mathcal{A}'$ cannot have two accepting computations such that one of them outputs C_1 and the other one outputs C_2.*

Proof. For a given list of circuits $C'_1, \ldots, C'_t$ produced by $F'(C_1 \vee C_2)$, let $U \subseteq \{1, \ldots, t\}$ be a subset of $\alpha \cdot t$ positions such that each C_i, $i \in U$, is uniquely satisfiable. Every set I nondeterministically chosen by $\mathcal{A}'$ must contain at least $(\alpha - (1 - \alpha))t = (2\alpha - 1)t > t/3$ positions from U, since $\alpha > 2/3$. For these positions, the satisfying assignments w_i are unique, and so will be the same on any nondeterministic computation.

Suppose that one accepting computation selects C_1. Then, by the above, more than $t/3$ circuits among $C'_1, \ldots, C'_t$ have unique satisfying assignments that all satisfy C_1. Thus there are fewer than $2t/3 < \alpha \cdot t$ circuits among $C'_1, \ldots, C'_t$ that may have satisfying assignments which all falsify C_1. The latter implies that there cannot exist an accepting computation that selects C_2. □

We have by Claims 3.8 and 3.11 that $\mathcal{A}'(C_0, C_1)$, for a satisfiable circuit $C_0 \vee C_1$, always has an accepting computation. Exactly one of the following happens:

1. each accepting computation of $\mathcal{A}'$ outputs some value from $\{C_0, \perp\}$, or
2. each accepting computation of $\mathcal{A}'$ outputs some value from $\{C_1, \perp\}$.

In the first case, we will say that $\mathcal{A}'$ *avoids* C_1; in the second case, we say that $\mathcal{A}'$ *avoids* C_0.

Claim 3.12. *Let S be any non-empty set of satisfiable circuits on n inputs. Then there is a circuit $C^* \in S$ such that $\mathbf{Pr}_{C \in S \setminus \{C^*\}}[\mathcal{A}'(C^*, C)$ avoids $C^*] \geqslant 1/2$.*

Proof. The same averaging argument as in the proof of Claim 3.3 applies. □

By repeatedly applying Claim 3.12, we get the existence of at most m satisfiable circuits $C^1(x_1, \ldots, x_n), \ldots, C^\ell(x_1, \ldots, x_n)$ such that, for every satisfiable circuit $C(x_1, \ldots, x_n)$ of size m, there is $i \in \{1, \ldots, \ell\}$ so that $\mathcal{A}'(C^i, C)$ avoids C^i. Also note that, by Claims 3.9 and 3.10, if $\mathcal{A}'(C^i, C)$ avoids C^i, then C must be satisfiable.

It follows that a given circuit $C(x_1, \ldots, x_n)$ of size m is unsatisfiable iff, for every $1 \leqslant i \leqslant \ell$, $\mathcal{A}'(C^i, C)$ avoids C. The latter can be checked in $\mathsf{NP/poly}$ (where the advice is whatever needed by F' on n-input circuits of size $O(m)$): simply guess an accepting computation of $\mathcal{A}'(C^i, C)$ that outputs C^i or $\perp$. Hence, we get that $\mathsf{coNP} \subseteq \mathsf{NP/poly}$, as required. □

Finally, suppose we strengthen our notion of randomized isolation so that there are no false negatives. That is, define a *zero-error* randomized δ-isolation to be a randomized algorithm that maps a circuit $C(x_1, \ldots, x_n)$ to a circuit $C'(x_1, \ldots, x_n)$ so that *(i)* if C is unsatisfiable then so is C' (always), *(ii)* if C is satisfiable then so is C' (always), and moreover, the set of satisfying assignments of C' is a subset of those of C, and *(iii)* if C is satisfiable then, with probability at least δ, the circuit C' is uniquely satisfiable.

This type of randomized isolation is unlikely to exist even for small $\delta = poly(1/n)$.

Theorem 3.13. *If , for some $\delta = poly(1/n)$, there is a zero-error randomized polynomial-time isolation for n-input Boolean circuits, then* $\mathsf{coNP} \subseteq \mathsf{NP/poly}$.

Proof. The proof is similar to that of Theorem 3.7. We first define a randomized algorithm that maps an input circuit C to a list of t circuits, for $t = poly(m, 1/\delta)$, so that, if C is satisfiable, then, with probability greater than $1 - 2^{-m}$, at least one circuit on the list is uniquely satisfiable (and, by definition of the zero-error isolation reduction, all circuits on the list are satisfiable). Then we fix the randomness to get a $\mathsf{P/poly}$-computable such mapping, which we denote by F'. Then we use the selection algorithm $\mathcal{A}'$ from the proof of Theorem 3.7 with the parameter $\alpha = 1$ (and so the set $I = \{1, \ldots, t\}$).

This modified selection algorithm $\mathcal{A}'$ still enjoys all the properties given in Claims 3.8-3.11. Claims 3.8-3.10 are proved in exactly the same way as before. To see why Claim 3.11 still holds, observe that the list of circuits produced by F' contains a uniquely satisfiable circuit with some w as its only satisfying assignment; if $\mathcal{A}'$ outputs C_1, then $C_1(w) = 1$; on the other hand, in order for $\mathcal{A}'$ to output C_2 on some other nondeterministic branch it must be the case that $C_1(w) = 0$, which is impossible.

The remainder of the proof is then exactly the same as that of Theorem 3.7. □

3.3 Generalized randomized isolation

Here we consider some generalizations of the definition of randomized isolation given earlier. Before we considered a randomized δ-isolation mapping $C(x_1, \ldots, x_n)$ to a new circuit $C'(x_1, \ldots, x_n)$ on the same variables. We can generalize this to allow $C'(x_1, \ldots, x_n, y_1, \ldots, y_k)$ to depend on more variables. Rather than requiring that all satisfying assignments of C' be also satisfying for C, we require that the projection of any satisfying assignment for C' to the first n coordinates be a satisfying assignment for C. The δ-isolation property changes as follows: if C is satisfiable, then, with probability at least δ, the circuit $C'(x_1, \ldots, x_n, y_1, \ldots, y_k)$ is such that

$$\exists! w \in \{0,1\}^n \; \exists z \in \{0,1\}^k \;\; C'(w, z) = 1.$$

That is, while C' may have many satisfying assignments, they all must have the same value on the first n variables. Let us call such a reduction a *generalized randomized δ-isolation.* The proof of Theorem 3.7 can be easily adapted to prove the following.

Theorem 3.14. *If, for some constant $\delta > 2/3$, there is generalized randomized polynomial-time δ-isolation, then* $\mathsf{coNP} \subseteq \mathsf{NP/poly}$.

Proof. The proof is essentially identical to that of Theorem 3.7, with the obvious modification of the selection procedure $\mathcal{A}'$ to use the projections to the first n coordinates of the satisfying assignments w for the circuits $C'_i(x_1, \ldots, x_n, y_1, \ldots, y_k)$, for $1 \leqslant i \leqslant t$. □

Consider now generalized *deterministic* polynomial-time isolation reductions (which are generalized randomized polynomial-time 1-isolation reductions). Note first that such a reduction exists if $\mathsf{NP} = \mathsf{coNP}$.

Lemma 3.15. *If* $\mathsf{coNP} = \mathsf{NP}$*, then there is a generalized deterministic polynomial-time isolation reduction.*

Proof. Indeed, given a circuit $C(x_1, \ldots, x_n)$ and an assignment $w \in \{0,1\}^n$, we can verify in coNP that w is the lexicographically smallest satisfying assignment of C. If $\mathsf{coNP} = \mathsf{NP}$, this verification can be performed in NP. Let $y_1, \ldots, y_k$ be a witness for this NP-computable verification. Let $C'(x_1, \ldots, x_n, y_1, \ldots, y_k)$ be a deterministic circuit that checks if $y_1 \ldots y_k$ is a valid witness certifying that $x_1 \ldots x_n$ is the lexicographically smallest satisfying assignment of C; such a circuit of polynomial size can be obtained from an NP-machine by standard methods. Clearly, if C is satisfiable, then the constructed circuit C' is also satisfiable and all of its satisfying assignment agree on the first n variables (which is the unique lexicographically smallest satisfying assignment of C). □

One can easily extend the proof of Lemma 3.15, to get the following.

Lemma 3.16. *If* $\mathsf{coNP} \subseteq \mathsf{NP/poly}$*, then there is a generalized deterministic* $\mathsf{P/poly}$*-computable isolation reduction.*

Using Theorem 3.14 and Lemma 3.16, we obtain the following "derandomization" result.

Corollary 3.17. *If, for some constant* $\delta > 2/3$*, there is generalized randomized polynomial-time* δ*-isolation, then there is also a* $\mathsf{P/poly}$*-computable generalized deterministic isolation.*

Obviously, Theorem 3.14 applies to the case of generalized deterministic polynomial-time isolation reductions. Moreover, it is easy to see that the same conclusion $\mathsf{coNP} \subseteq \mathsf{NP/poly}$ follows also from the assumption that there is generalized deterministic $\mathsf{P/poly}$-computable isolation reduction (the advice used by the isolation reduction is simply added to the advice of an $\mathsf{NP/poly}$-algorithm in the conclusion of Theorem 3.14). This generalization of Theorem 3.14 and Lemma 3.16 immediately yield the following.

Corollary 3.18. $\mathsf{P/poly}$*-computable generalized deterministic isolation exists iff* $\mathsf{coNP} \subseteq \mathsf{NP/poly}$.

Remark 3.19. It is possible to relax the definition of isolation even further as follows. We require that there be an efficient "witness-extraction" procedure G such that, when a satisfiable circuit C is mapped to a circuit C', the procedure G, given a satisfying assignment for C', outputs a satisfying assignment for C. The proof of Theorem 3.7 can be adapted to this generalization of isolation as well; we omit the details.

4 Black-Box Isolation

We consider a general situation where some randomized procedure is used to isolate one element in a given unknown set W in some specified family $\mathcal{W}$ of subsets of $\{0,1\}^n$. The randomized procedure can be designed depending on $\mathcal{W}$, but it is not given any information on which $W \in \mathcal{W}$ is chosen. The randomized procedure can check whether a given $w \in \{0,1\}^n$ is chosen or not; in other words, it is specified as a distribution $\mathcal{D}$ over subsets of $\{0,1\}^n$, where each $D \in \mathcal{D}$ is the set of strings

that the randomized procedure selects when its random seed is fixed. This leads to the following type of isolation. Below, for a distribution $\mathcal{D}$ and an element D from the support of $\mathcal{D}$, we denote by $D : \mathcal{D}$ the fact that D is chosen according to the distribution $\mathcal{D}$.

For any family $\mathcal{W}$ of nonempty subsets of $\{0,1\}^n$, its *blackbox isolation procedure* is a distribution $\mathcal{D}$ over subsets D of $\{0,1\}^n$. For any $D \in \mathcal{D}$ and any $W \in \mathcal{W}$, we say that D *succeeds on* W if $|D \cap W| = 1$. The *isolation probability* of $\mathcal{D}$ for $\mathcal{W}$ is defined as $\min_{W \in \mathcal{W}} \Pr_{D:\mathcal{D}}[\,|D \cap W| = 1\,]$.

While this is regarded as the "worst-case" isolation probability, we may also consider an average isolation probability. For this, we regard $\mathcal{W}$ as a distribution over subsets of $\{0,1\}^n$.

For any distribution $\mathcal{W}$ over subsets of $\{0,1\}^n$ and any blackbox isolation procedure $\mathcal{D}$, its *average isolation probability for* $\mathcal{W}$ is defined as $\mathrm{E}_{W:\mathcal{W}}[\,\Pr_{D:\mathcal{D}}[\,|D \cap W| = 1\,]\,]$. Clearly, the average isolation probability for a distribution $\mathcal{W}$ is an upper bound of the isolation probability for the corresponding subset family $\mathcal{W}$.

Below we show that the average isolation probability is $O(1/n)$ for some distribution $\mathcal{W}$. Here we first give a key relation for our analysis. Consider any distribution $\mathcal{W}$ over subsets of $\{0,1\}^n$ and any blackbox isolation procedure $\mathcal{D}$ for $\mathcal{W}$. For any $W \in \mathcal{W}$ and $D \in \mathcal{D}$, by $\mathrm{Iso}(D, W)$ we denote the indicator of the isolation; that is, $\mathrm{Iso}(D, W) = 1$ if $|D \cap W| = 1$ and $\mathrm{Iso}(D, W) = 0$ otherwise. Then we have that the average success probability $\mathrm{E}_{W:\mathcal{W}}[\,\Pr_{D:\mathcal{D}}[\,|D \cap W| = 1\,]\,]$ equals

$$\begin{aligned}\sum_{W:\mathcal{W}} \mathcal{W}(W) \cdot \left(\sum_{D \in \mathcal{D}} \mathcal{D}(D) \cdot \mathrm{Iso}(D, W)\right) &= \sum_{D:\mathcal{D}} \mathcal{D}(D) \cdot \left(\sum_{W \in \mathcal{W}} \mathcal{W}(W) \cdot \mathrm{Iso}(D, W)\right) \\ &= \sum_{D:\mathcal{D}} \mathcal{D}(D) \cdot \Pr_{W:\mathcal{W}}[\,|D \cap W| = 1\,]. \qquad (4)\end{aligned}$$

Below we let $N = 2^n$, and for any K, $1 \le K \le N$, let $p_K = K/N$. For defining a distribution over subsets of $\{0,1\}^n$, we consider a random procedure that generates a subset of $\{0,1\}^n$. First for any fixed $K = o(N)$, consider a simple procedure that chooses each $w \in \{0,1\}^n$ with probability p_K independently, and let $\mathcal{W}_K$ denote the distribution corresponding this procedure. Roughly, $W : \mathcal{W}_K$ has K strings on average. That is, we consider the isolation when we can approximate the target set size well. In this case, by the isolation technique by Valiant-Vazirani we can achieve $1/4$ isolation probability. The following theorem shows that one cannot go beyond $1/e$ by any black box isolation procedure.

Theorem 4.1. *Let $K = o(N)$. For any black-box isolation procedure $\mathcal{D}$, its average isolation probability for $\mathcal{W}_K$ is at most $e^{-1}(1 + o(N^{-1}))$.*

Proof. Consider any set D with H elements. Then its isolation probability for $\mathcal{W}_K$ is

$$\begin{aligned}\Pr_{W:\mathcal{W}_K}[\,|D \cap W| = 1\,] &= H \cdot p_K\,(1 - p_K)^{H-1} = \frac{HK}{N}\left(1 - \frac{K}{N}\right)^H \left(1 - o\left(\frac{1}{N}\right)\right)^{-1} \\ &\le \frac{HK}{N} e^{-HK/N}\left(1 + o\left(\frac{1}{N}\right)\right) \le e^{-1}(1 + o(N^{-1})),\end{aligned}$$

where this upper bound is obtained by choosing $H = N/K$. Since the bound is the same for any D, this gives an upper bound for any distribution $\mathcal{D}$ by (4). □

Next consider a more general case. We define a distribution $\mathcal{W}_0$ by the following random procedure generating a subset of $\{0,1\}^n$: First define $K = 2^k$ by choosing $k \in \{0,\ldots,n-1\}$ uniformly at random, and then generate W following $\mathcal{W}_K$, i.e., by selecting each string with probability p_K independently. For this distribution, we have the following bound.

Theorem 4.2. *For any black-box isolation procedure $\mathcal{D}$, its average isolation probability for $\mathcal{W}_0$ is $O(1/n)$.*

Proof. We again give a uniform isolation probability bound for any set D. Consider any D and let H be the number of its elements. Note that

$$\Pr_{W:\mathcal{W}_0}[\,|D\cap W|=1\,] \;=\; \sum_{k=0}^{n-1}\frac{1}{n}\cdot\Pr_{W:\mathcal{W}_{2^k}}[\,|D\cap W|=1\,] \;=\; \frac{1}{n}\cdot\sum_{K\in\mathcal{K}}\Pr_{W:\mathcal{W}_K}[\,|D\cap W|=1\,],$$

where $\mathcal{K} = \{1,2,4,\ldots,2^{n-1}\} = \{1,2,4,\ldots,N/2\}$. Thus, we estimate the above sum. From the previous proof, we have

$$\begin{aligned}
\sum_{K\in\mathcal{K}}\Pr_{W:\mathcal{W}_K}[\,|D\cap W|=1\,] &\le \sum_{K\in\mathcal{K}}\frac{2HK}{N}\left(1-\frac{K}{N}\right)^H\\
&\le \sum_{K\in\mathcal{K}\&K\le N/H}\frac{2HK}{N}\left(1-\frac{K}{N}\right)^H + \sum_{K\ge N/H}\frac{2HK}{N}e^{-HK/N}\\
&= \sum_{K\in\mathcal{K}\&K\le N/H}\frac{2HK}{N}\left(1-\frac{K}{N}\right)^H + O(1),
\end{aligned}$$

where the last bound is from the fact that $\sum_{x\ge 1} xe^{-x} = O(1)$. On the other hand, since we have that

$$\frac{2HK}{N}\left(1-\frac{K}{N}\right)^H \;\le\; \frac{2HK}{N}\left(1-\frac{HK}{N}+\frac{1}{2}\left(\frac{HK}{N}\right)^2\right),$$

and that

$$\sum_{K\in\mathcal{K}\&K\le N/H}\frac{2HK}{N}\left(1-\frac{HK}{N}+\frac{1}{2}\left(\frac{HK}{N}\right)^2\right) \;\le\; \frac{2H}{N}\cdot\frac{2N}{H}-\frac{2H^2}{N^2}\cdot\frac{4N^2}{3H^2}+\frac{2H^3}{2N^3}\cdot\frac{8N^3}{7H^3} \;\le\; 3,$$

we can conclude that

$$\sum_{K\in\mathcal{K}}\Pr_{W:\mathcal{W}_K}[\,|D\cap W|=1\,] \;\le\; \sum_{K\in\mathcal{K}\&K\le N/H}\frac{2HK}{N}\left(1-\frac{HK}{N}+\frac{1}{2}\left(\frac{HK}{N}\right)^2\right)+O(1) \;=\; O(1),$$

which proves the desired bound. □

5 Conclusions

We have considered different ways in which one might want to strengthen the Valiant-Vazirani isolation: deterministic isolation, randomized isolation with large constant success probability, or

zero-error randomized isolation with inverse-polynomial success probability. We showed that any such strengthening would lead to a collapse of the polynomial-time hierarchy, and thus, is unlikely. We also showed that a natural "black-box" isolation procedure (generalizing the one of Valiant and Vazirani) cannot have success probability better than $O(1/n)$ (achieved by the Valiant-Vazirani isolation).

Our result that an efficient deterministic isolation procedure would imply $\mathsf{NP} \subseteq \mathsf{P/poly}$ (Theorem 3.1) can be interpreted as saying that derandomizing the Isolation Lemma (in the strong sense, where the output of the isolation procedure is a *single* circuit) would imply circuit *upper* bounds for NP. This is in contrast to the previous results showing that derandomization would imply circuit *lower* bounds for NEXP [IKW02, KI04]. Also, while such strong derandomization of the Isolation Lemma seems unlikely, the derandomization in the weak sense, where a satisfiable circuit is mapped to a *list* of circuits with at least one being uniquely satisfiable, is likely to exist (under plausible complexity assumptions) [KM02].

There are several interesting open questions. While we have argued that an efficient randomized δ-isolation with success probability $\delta > 2/3$ is unlikely to exist, we do not know about intermediate values of δ, for $2/3 > \delta > \omega(1/n)$. Also, suppose one relaxes the isolation requirement so that a given satisfiable circuit is mapped to a circuit with an *odd number* of satisfying assignments (rather than unique satisfying assignment). Is it possible to rule out (under suitable complexity assumptions) an efficient deterministic algorithm for isolating an odd number of satisfying assignments? Note that for this version of isolation, there is an efficient randomized isolation procedure achieving *constant* success probability [Gup98]. Finally, it remains a very interesting open question whether the assumption $\mathsf{NP} = \mathsf{UP}$ would lead to any surprising consequences (e.g., a collapse of the polynomial-time hierarchy).

Acknowledgements We would like to thank Leslie Valiant for his insightful comments on the results presented in the paper.

References

[AB09] S. Arora and B. Barak. *Complexity theory: a modern approach.* Cambridge University Press, New York, 2009.

[BDCGL92] S. Ben-David, B. Chor, O. Goldreich, and M. Luby. On the theory of average-case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.

[Gup98] S. Gupta. Isolating an odd number of elements and applications in complexity theory. *Theory of Computing Systems*, 31:27–40, 1998.

[HNOS96] L. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. Computing solutions uniquely collapses the polynomial hierarchy. *SIAM Journal on Computing*, 25(4):697–708, 1996.

[IKW02] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

[KI04] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1–2):1–46, 2004.

[KM02] A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[Ko83] K. Ko. On self-reducibility and weak P-selectivity. *Journal of Computer and System Sciences*, 26:209–211, 1983.

[MVV87] K. Mulmuley, U. Vazirani, and V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987.

[RA00] K. Reinhardt and E. Allender. Making nondeterminism unambiguous. *SIAM Journal on Computing*, 29:1118–1131, 2000.

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Mathematical Systems Theory*, 13:55–65, 1979.

[Sel94] A. Selman. A taxonomy of complexity classes of functions. *Journal of Computer and System Sciences*, 48:357–381, 1994.

[Tod91] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.

[VV86] L. Valiant and V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de