THE ROBUSTNESS OF LWPP AND WPP, WITH AN APPLICATION TO GRAPH RECONSTRUCTION

EDITH HEMASPAANDRA, LANE A. HEMASPAANDRA, HOLGER SPAKOWSKI, AND OSAMU WATANABE

Abstract. We show that the counting class LWPP remains unchanged even if one allows a polynomial number of gap values rather than one. On the other hand, we show that it is impossible to improve this from polynomially many gap values to a superpolynomial number of gap values by relativizable proof techniques.

The first of these results implies that the Legitimate Deck Problem (from the study of graph reconstruction) is in LWPP (and thus low for PP, i.e., $PP^{\text{Legitimate Deck}} = PP$) if the weakened version of the Reconstruction Conjecture holds in which the number of nonisomorphic preimages is assumed merely to be polynomially bounded. This strengthens the 1992 result of Köbler, Schöning & Torán that the Legitimate Deck Problem is in LWPP if the Reconstruction Conjecture holds, and provides strengthened evidence that the Legitimate Deck Problem is not NP-hard.

We additionally show on the one hand that our LWPP robustness result also holds for WPP, and also holds even when one allows both the rejection and acceptance gap-value targets to simultaneously be polynomial-sized lists; yet on the other hand, we show that for the $\#$P-based analogue of LWPP the behavior much differs in that, in some relativized worlds, even two target values already yield a richer class than one value does. Despite that nonrobustness result for a $\#$P-based class, we show that the $\#$P-based “exact counting” class $C_=P$ remains unchanged even if one allows a polynomial number of target values for the number of accepting paths of the machine.
Keywords. Structural complexity theory, Robustness of counting classes, The legitimate deck problem, PP-lowness, The Reconstruction Conjecture

Subject classification. 68Q15

1. Introduction and related work

It is natural to want to better understand objects by asking what will and won’t change their behavior. Whether wondering if changing from premium to regular gas in one’s car will leave its acceleration unchanged, or wondering if \( \text{NP}^{\text{NP}} \) can without loss of generality be assumed to ask at most one question per nondeterministic path (as it indeed can, as is implicit in the quantifier characterization of \( \text{NP}^{\text{NP}} \), see Stockmeyer & Meyer 1973; Wrathall 1977), we both in life and as theoreticians want to find how robust things are.

We are often particularly happy when a class proves to be quite robust under definitional perturbations. Such robustness on the one hand suggests that perhaps there is something broadly natural about the class, and on the other hand such robustness often makes it easier to put the class to use.

This paper shows that the counting classes LWPP and WPP, which were first defined in work of Fenner, Fortnow & Kurtz (1994) on gap-based counting classes, are quite robust. Even though their definitions are in terms of having the gap function (the difference between the number of accepting and rejecting paths of a machine) hit a single target value, we prove (in Section 3) that one can allow a list of up to polynomially many target values without altering the descriptive richness of the class, i.e., without changing the class.

LWPP, WPP, and their cousins have been studied in a variety of contexts, e.g., Cox & Pay (2018) have defined and studied a variant of WPP whose analogous variant for the case of LWPP is the subject of our Section 6; Fenner (2003) has established the robustness of the counting class AWPP (Fenner et al. 2003); Rao, Rothe & Watanabe (1994) showed that LWPP has “upward
The Robustness of LWPP and WPP

Fenner, Fortnow & Kurtz (1994) showed the robustness result that $\text{SPP}^{\text{LWPP}} = \text{LWPP}$ and Rao et al. (1994) in their corrigendum extended that to a result about the “FEW” operator; Spakowski & Tripathi (2006) showed that LWPP and WPP are not uniformly gap-definable; Berthiaume & Brassard (1992, 1994) showed that the class $C_{=\leq}P[\text{half}]$, a subclass of LWPP, is a subset of quantum polynomial time; Fortnow & Rogers (1999) showed that an important quantum class is a subset of LWPP; Vinodchandran (2004) has shown that LWPP has natural problems that are related to black-box group issues; and as is quite important to this paper and will be further discussed soon, Köbler, Schöning & Torán (1992) showed an important connection between LWPP, the Legitimate Deck Problem, and the Reconstruction Conjecture.

We apply our LWPP robustness result—that allowing a list of up to polynomially many targets does not alter the descriptive richness of LWPP—to the question of whether the Legitimate Deck Problem is in LWPP.

The Legitimate Deck Problem (a formal definition will be given in Section 4) is the decision problem of determining whether, given a multiset of (unlabeled) graphs, there exists a graph $G$ such that the multiset is precisely (give or take isomorphisms) the multiset of one-node-deleted subgraphs of $G$ (a.k.a. the deck of $G$) (Kratsch & Hemaspaandra 1994). The Reconstruction Conjecture (Kelly 1942; Ulam 1960)—and in the wake of the resolution of the Four-Color Conjecture the editors of the Journal of Graph Theory in a lead editorial declared, “The foremost currently unsolved problem in graph theory is, in our considered opinion, the Reconstruction Conjecture” (Editors (of the Journal of Graph Theory) 1977)—states that every graph with three or more nodes is uniquely determined (give or take isomorphisms) by its multiset of one-node-deleted subgraphs. The Legitimate Deck Problem was defined in the 1970s by Nash-Williams (1978), in his paper that framed the algorithmic/complexity issues related to reconstructing graphs—such as telling whether a given deck is legitimate (i.e., is

---

1 The two equal signs in $C_{=\leq}P[\text{half}]$ are not a typo; $C_{=\leq}P[\text{half}]$ is precisely the class name given by Berthiaume & Brassard (1992, 1994).
the deck of some graph). For other work on the computational complexity of graph reconstruction problems, see, e.g., Hemaspaandra et al. (2007); Köbler et al. (1992); Kratsch & Hemaspaandra (1994).

Our application of our LWPP robustness result to the question of whether the Legitimate Deck Problem is in LWPP is the following. The strongest previous evidence of the simplicity of the Legitimate Deck Problem is the 1992 result of Köbler et al. (1992) that the Legitimate Deck Problem is in LWPP (and thus is PP-low, i.e., $\text{PP}^{\text{Legitimate Deck}} = \text{PP}$) if the Reconstruction Conjecture (i.e., that each deck whose elements all have at least two nodes has at most one preimage, give or take isomorphisms) holds. Using this paper’s LWPP robustness result as a tool, Section 4 proves that the Legitimate Deck Problem is in LWPP (and thus is PP-low) if a weakened version of the Reconstruction Conjecture holds, namely, that each deck has at most a polynomial number of nonisomorphic preimages.

This weakened version is not known to be equivalent to the Reconstruction Conjecture itself. And so our result for the first time gives a path to proving that the Legitimate Deck Problem is PP-low that does not require one to, on the way, resolve what as noted above the editors of the Journal of Graph Theory in 1977 dubbed “the foremost currently unsolved problem in graph theory.”

We started this section by noting that it is natural to want to know both flexibilities and limitations of classes. Our LWPP robustness result is about flexibility: going from one target gap to instead a polynomial number. But are we leaving money on the table? Could we extend our result to slightly superpolynomial numbers of target gaps, or even to exponential numbers of target gaps? In Section 5, we note that if the robustness of LWPP were to hold up to exponentially many target gaps, then NP would be in LWPP and so would be PP-low (i.e., $\text{PP}^{\text{NP}} = \text{NP}$); yet NP is widely suspected not to be PP-low. We also, by encoding nondeterministic oracle Turing machines by low-degree multivariate polynomials so as to capture the gap functions of those machines, show an oracle relative to which robustness fails for all superpolynomial numbers of target gaps; thus, no extension beyond this paper’s polynomial-number-of-target-gaps robustness result for LWPP can be proven.
by a relativizable proof. (In this paper, we generally use relativized results as a counterpoint to real-world (though clearly relativizable) positive results, to give evidence that the positive results are not blindly missing possible improvements—more specifically, to show that no relativizable technique can improve the given result.) And for the \#P-based analogue of LWPP, in Section 6 we show that even allowing two target values yields, in some relativized worlds, a richer class than one target value. For the important \#P-based counting class $C_\#P$, however, we prove in Section 7 that the class does not change even if one allows not just one but instead a polynomial number of target values for the number of accepting paths of the underlying \#P function. We also (in the final part of Section 3) extend our LWPP robustness result to show that the simultaneous expansion to polynomial-sized lists of both the acceptance and rejection target-gap lists still, for LWPP and WPP, yields the classes LWPP and WPP.

To summarize: In this paper, we prove that LWPP and WPP are robust enough that they remain unchanged when their single target gap is allowed to be expanded to a polynomial-sized list; we apply this new robustness of LWPP to show that the PP-lowness of the Legitimate Deck Problem follows from a weaker hypothesis than was previously known; we show that our polynomial robustness of LWPP is optimal with respect to relativizable proofs; and we prove a number of related results on limitations and extensions.

2. Preliminaries

We first present the definitions of many of the counting classes that we will be speaking of, taking the definitions from the paper of Fenner, Fortnow & Kurtz (1994). We use the notations $\mathbb{N} = \{0, 1, 2, \ldots \}$, $\mathbb{N}^+ = \{1, 2, \ldots \}$, and $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots \}$.

**Definition 2.1** (Fenner et al. 1994). (i) For each nondeterministic polynomial-time Turing machine $N$, the function $\text{acc}_N : \Sigma^* \rightarrow \mathbb{N}$ is defined such that for every $x \in \Sigma^*$, $\text{acc}_N(x)$ equals the number of accepting computation paths of $N$ on input $x$.

(ii) For each nondeterministic polynomial-time Turing machine
$N$, the function $\text{rej}_N : \Sigma^* \to \mathbb{N}$ is defined such that for every $x \in \Sigma^*$, $\text{rej}_N(x)$ equals the number of rejecting computation paths of $N$ on input $x$.

(iii) For each nondeterministic polynomial-time Turing machine $N$, the function $\text{gap}_N : \Sigma^* \to \mathbb{Z}$ is defined such that for every $x \in \Sigma^*$,
\[ \text{gap}_N(x) = \text{acc}_N(x) - \text{rej}_N(x). \]

**Definition 2.2 (Fenner et al. 1994).** $\text{GapP} = \{ \text{gap}_N \mid N \text{ is a polynomial-time nondeterministic Turing machine} \}$.

**Definition 2.3.** (Fenner et al. (1994), Ogiwara & Hemachandra 1993) $\text{SPP}$ is the class of all sets $A$ such that there exists a GapP function $g$ such that for all $x \in \Sigma^*$,
\[ x \in A \implies g(x) = 1 \]
\[ x \notin A \implies g(x) = 0. \]

The following class, WPP, is potentially larger than SPP. Instead of the “target value” 1 for the case $x \in A$, we allow a target value $f(x)$, where $f$ may be any polynomial-time computable function whose image does not contain 0. FP denotes the class of polynomial-time computable functions.

**Definition 2.4 (Fenner et al. 1994).** WPP is the class of all sets $A$ such that there exists a GapP function $g$ and a function $f \in \text{FP}$ that maps from $\Sigma^*$ to $\mathbb{Z} - \{0\}$ such that for all $x \in \Sigma^*$,
\[ x \in A \implies g(x) = f(x) \]
\[ x \notin A \implies g(x) = 0. \]

The class LWPP is the same as WPP except that the “target function” $f$ may depend on only the length of the input.
Definition 2.5 (Fenner et al. 1994). LWPP is the class of all sets $A$ such that there exists a GapP function $g$ and a function $f \in \text{FP}$ that maps from $0^*$ to $\mathbb{Z} - \{0\}$ such that for all $x \in \Sigma^*$,

$$x \in A \implies g(x) = f(0^{\left| x \right|})$$

$$x \notin A \implies g(x) = 0.$$ 

Above, as is usual, for any $x \in \Sigma^*$, $|x|$ denotes the length of $x$, and for any $n \in \mathbb{N}$, $0^n$ is the string consisting of exactly $n$ zeroes. To avoid any chance of confusion, we use the notation $\|A\|$ to denote the cardinality of set $A$.

Whenever we use the 2-argument notation $\langle \cdot, \cdot \rangle$, we are denoting a standard pairing function, namely a polynomial-time computable, polynomial-time invertible bijection between $\Sigma^* \times \Sigma^*$ and $\Sigma^*$. (When one or more of those three $\Sigma^*$ fields has an argument of the type, or its output is being treated as being of the type, $\mathbb{N}$, we view that as taking place via the standard correspondence in which the $i$th string in $\Sigma^*$ in lexicographic order is associated with the natural number $i - 1$.) In a slight abuse of notation that is typical and will be clear from context, whenever—and most of the uses in Section 4 are of this sort—we use $\langle \cdots \rangle$ on an argument whose number of arguments is not fixed a priori, e.g., $\langle G_1, G_2, \ldots, G_n \rangle$, the notation denotes a standard, complexity-theoretic, multi-arity, pairing function (e.g., the one built in Section 2 of Han et al. 1997).

We now generalize the definition of LWPP to the case of having the target of the GapP function be, for members of the set, not a single value but a collection of values.

One might expect us to formalize this by simply having the polynomial-time computable “what is the target” function output a list of the nonzero-integer targets. That would work fine and be equivalent to what we are about to do, as long as we are dealing with lists having at most a polynomial number of elements. However, to be able to speak of even longer lists—as will be important in our negative results offsetting our LWPP robustness result—we use an indexing approach, as follows.

Definition 2.6. Let $r$ be any function mapping from $\mathbb{N}$ to $\mathbb{N}$. The class $r$-LWPP is the class of all sets $A$ such that there exists
a GapP function $g$ and a function $f \in \text{FP}$ that maps from $\Sigma^*$ to $\mathbb{Z} - \{0\}$ such that for each $x \in \Sigma^*$,

$$x \in A \implies \text{there exists } i \in \{1, 2, \ldots, r(|x|)\} \text{ such that } g(x) = f(\langle 0^{\mid x \mid}, i \rangle)$$

$$x \notin A \implies g(x) = 0.$$

The following class, $\text{Poly-LWPP}$, will be central to this paper. In particular, we will prove that this class in fact equals LWPP.

The “+c” in Definition 2.7 may seem strange at first. But without it we would have a boundary-case pathology at $n = 0$, namely, the class could not contain any set that contains the empty string.\footnote{The “+c” in Definition 2.7 also, on its surface, would seem to make a difference at length 1, by allowing lists of size greater than one; however, one could work around that issue. In contrast, the exclusion of the empty string would not be avoidable if our class of polynomials were to be a class—such as $n^c$—such that all of its members evaluate to 0 at $n = 0$. In any case, our use of $n^c + c$ avoids any special worries at lengths 0 and 1. And since for every polynomial $p$ there is a $c$ such that $(\forall n \in \mathbb{N})[p(n) \leq n^c + c]$, using polynomials just of the form $n^c + c$ is in fact not a restriction.}

**Definition 2.7.** $\text{Poly-LWPP} = \bigcup_{c \in \mathbb{N}^+} (n^c + c)\text{-LWPP}$.

It is easy to see that $1\text{-LWPP} = \text{LWPP}$, and that, of course, more flexibility as to targets never removes sets from the class, i.e., speaking loosely for the moment as to notation (and the log case will not be defined or used again in this paper, but it is clear from context here what we mean by it; the exponential case will be defined only in Section 5), $1\text{-LWPP} \subseteq 2\text{-LWPP} \subseteq 3\text{-LWPP} \subseteq \cdots \subseteq \text{Log-LWPP} \subseteq \text{Poly-LWPP} \subseteq \text{Exp-LWPP}$. As mentioned above, in this paper we will prove that the first five of these “$\subseteq$”s are in fact all equalities. We will also prove that the sixth “$\subseteq$” cannot be an equality unless NP is PP-low.

The well-known counting class $\text{C}_\pm\text{P}$—to which the entire polynomial hierarchy is known to in a certain sense bounded-error probabilistically reduce (Toda & Ogiwara 1992)—has been described by Köbler et al. (1992) and others (e.g., Hemaspaandra & Ogihara 2002; Rothe 1999) as capturing the power of “exact counting.” As
Proposition 2.10, we will show that for every function \( r \), \( r \)-LWPP is contained in the class of complements of \( C_{=P} \) sets, namely in \( \text{co}C_{=P} = \{ L \mid \overline{L} \in C_{=P} \} \). First, though, we define \( C_{=P} \) and give a characterization of Fenner et al. (1994)—that we will use later in this paper—of \( C_{=P} \) in terms of \( \text{GapP} \) functions.

**Definition 2.8** (Simon 1975; Wagner 1986). \( C_{=P} \) is the class of all sets \( A \) such that there is a nondeterministic polynomial-time Turing machine \( N \) and a function \( f \in \text{FP} \) such that for each \( x \in \Sigma^* \),

\[
x \in A \iff \text{acc}_N(x) = f(x).
\]

**Theorem 2.9** (Fenner et al. 1994). For each \( A \subseteq \Sigma^* \), \( A \in C_{=P} \) if and only if there exists a function \( g \in \text{GapP} \) such that for all \( x \in \Sigma^* \),

\[
x \in A \iff g(x) = 0.
\]

We thus certainly have the following, which holds simply by taking the \( \text{GapP} \) function \( g \) required in Theorem 2.9 to be the same as the function \( g \) in Definition 2.6.

**Proposition 2.10.** For each function \( r : \mathbb{N} \to \mathbb{N} \), \( r \)-LWPP \( \subseteq \text{co}C_{=P} \).

**3. LWPP stays the same if for accepted inputs we allow polynomially many gap values instead of one**

We now state our main LWPP robustness result (though we will later extend it): LWPP altered to allow even a polynomial number of target gap values is still LWPP (i.e., with just one target gap value). (When, in this paper, we refer to “our LWPP robustness result,” we are referring to Theorem 3.1.)

**Theorem 3.1.** Poly-LWPP = LWPP.
For the proof, we need the following closure properties shown by Fenner et al. (1994).

**Closure Property 3.2 (Fenner et al. 1994).** \( \text{GapP} \circ \text{FP} = \text{GapP} \) and \( \text{FP} \subseteq \text{GapP} \).

**Closure Property 3.3 (Fenner et al. 1994).** If \( g \in \text{GapP} \), then \( -g \in \text{GapP} \).

**Closure Property 3.4 (Fenner et al. 1994).** If \( g \in \text{GapP} \) and \( q \) is a polynomial, then the function

\[
h(x) = \prod_{0 \leq i \leq q(|x|)} g(\langle x, i \rangle)
\]

is in \( \text{GapP} \).

**Closure Property 3.5 (Fenner et al. 1994).** \( \text{GapP} \) is closed under addition, subtraction, and multiplication.

**Proof of Theorem 3.1.** As mentioned previously, it is easy to see that \( \text{LWPP} \subseteq \text{Poly-LWPP} \).

To show \( \text{Poly-LWPP} \subseteq \text{LWPP} \), let \( A \) be a set in \( \text{Poly-LWPP} \) defined by \( g \in \text{GapP} \), \( f \in \text{FP} \), and polynomial \( r(n) = n^c + c \) according to Definition 2.6 and Definition 2.7.

Let \( h_1 \) be a function such that for all \( x \in \Sigma^* \) and \( i \in \mathbb{N}^+ \),

\[
h_1(\langle x, i \rangle) = f(\langle 0^{\lfloor |x| \rfloor}, i \rangle) - g(x).
\]

\(^3\) We mention in passing that, regarding Closure Property 3.4, if the polynomial \( q \) were allowed to have coefficients that are uncomputable—or that are extremely expensive to compute prefixes of the values of—real numbers, that claimed closure property might fail; we here are, as is typical in such settings, tacitly assuming that the polynomials have rational coefficients.
We have $h_1 \in \text{GapP}$ since $f \in \text{FP} \subseteq \text{GapP}$, $g \in \text{GapP}$, and \text{GapP} is closed under subtraction (Fenner et al. 1994). We define $h_2$ such that for all $x \in \Sigma^*$,

$$h_2(x) = \prod_{1 \leq i \leq r(\|x\|)} h_1(\langle x, i \rangle).$$

By Closure Property 3.4, $h_2 \in \text{GapP}$. Since for all $x \in \Sigma^*$, $h_2(x) = \prod_{1 \leq i \leq r(\|x\|)} (f(\langle |x|, i \rangle) - g(x))$, it follows that for every $x \in \Sigma^*$, (3.6)

$$h_2(x) = \begin{cases} 
0 & \text{if there exists } i \in \{1, 2, \ldots, r(\|x\|)\} \text{ such that } g(x) = f(\langle |x|, i \rangle) \\
\prod_{1 \leq i \leq r(\|x\|)} f(\langle |x|, i \rangle) & \text{if } g(x) = 0.
\end{cases}$$

Now we define the function $\hat{g}$ such that for all $x \in \Sigma^*$,

$$\hat{g}(x) = h_2(x) - \prod_{1 \leq i \leq r(\|x\|)} f(\langle |x|, i \rangle).$$

Using the closure properties, it is easy to see that $\hat{g} \in \text{GapP}$. Note that by Eqn. (3.6), we have that for all $x \in \Sigma^*$, (3.7)

$$\hat{g}(x) = \begin{cases} 
- \prod_{1 \leq i \leq r(\|x\|)} f(\langle |x|, i \rangle) & \text{if there exists } i \in \{1, 2, \ldots, r(\|x\|)\} \text{ such that } g(x) = f(\langle |x|, i \rangle) \\
0 & \text{if } g(x) = 0.
\end{cases}$$

Let $\hat{f}$ be a function such that for all $\ell \in \mathbb{N}$,

$$\hat{f}(0^\ell) = - \prod_{1 \leq i \leq r(\ell)} f(\langle 0^\ell, i \rangle).$$

It is easy to see that $\hat{f} \in \text{FP}$. Keeping in mind Eqn. (3.7), it follows from the above that for every $x \in \Sigma^*$,

$$\hat{g}(x) = \begin{cases} 
\hat{f}(\|x|) & \text{if } x \in A \\
0 & \text{otherwise}.
\end{cases}$$

Since $\hat{g} \in \text{GapP}$ and $\hat{f} \in \text{FP}$, this implies that $A \in \text{LWPP}$. \hfill \square
A theorem analogous to Theorem 3.1 also holds for the corresponding class—\( \text{Poly-WPP} \)—that allows the target values to depend on the actual input instead of only the length of the input.

**Definition 3.8.** Let \( r \) be any function mapping from \( \mathbb{N} \) to \( \mathbb{N} \). The class \( r \)-WPP is the class of all sets \( A \) such that there exists a \( \text{GapP} \) function \( g \) and a function \( f \in \text{FP} \) that maps from \( \Sigma^* \) to \( \mathbb{Z} - \{0\} \) such that for each \( x \in \Sigma^* \),

\[
\begin{align*}
x \in A &\implies \text{there exists } i \in \{1, 2, \ldots, r(|x|)\} \text{ such that } \\
g(x) &= f(\langle x, i \rangle) \\
x \notin A &\implies g(x) = 0.
\end{align*}
\]

**Definition 3.9.** \( \text{Poly-WPP} = \bigcup_{c \in \mathbb{N}^+} (n^c + c)\)-WPP.

**Theorem 3.10.** \( \text{Poly-WPP} = \text{WPP} \).

The proof is almost exactly the same as the proof of Theorem 3.1, simply taking into account the fact that for WPP the output of the “gap” function \( f \) can vary even among input strings \( x \) of the same length.

We remark that the robustness results stated in Theorem 3.1 and Theorem 3.10 do not seem to in any obvious way follow as corollaries to the closure of LWPP under polynomial-time Turing reductions (Fenner, Fortnow & Kurtz 1994) or of WPP under polynomial-time truth-table reductions (Spakowski, Thakur & Tripathi 2005).

Let us now see whether we can extend Theorem 3.1 and Theorem 3.10. Suppose we not only allow the target-gap set for acceptance to have polynomially many values, but in addition allow the target-gap set for rejection to have polynomially many values. (In contrast, both LWPP and \( r \)-LWPP allow rejection only when the gap’s value is 0.) Is LWPP so robust that even that class is no larger than LWPP? We will now build on Theorem 3.1 (and Theorem 3.10) to give the answer “yes” to that question, thus extending those results to this more symmetric case for LWPP and for WPP.
To this end, let us define \((r_A, r_R)\)-LWPP as follows. (One of course can in the clear, analogous way define a similarly loosened version of WPP, \((r_A, r_R)\)-WPP.)

**Definition 3.11.** Let \(r_A\) and \(r_R\) be any functions mapping from \(\mathbb{N}\) to \(\mathbb{N}\). Then \((r_A, r_R)\)-LWPP is the class of all sets \(B\) such that there exists a GapP function \(g\) and functions (each mapping to \(\mathbb{Z}\)) \(f_A \in \text{FP}\) and \(f_R \in \text{FP}\) such that both of the following hold:

(i) For each \(j \in \mathbb{N}\), \(A_j \cap R_j = \emptyset\), where

\[
A_j = \{n \mid (\exists i \in \{1, 2, \ldots, r_A(j)\})[f_A(\langle 0^j, i \rangle) = n]\}
\]

and

\[
R_j = \{n \mid (\exists i \in \{1, 2, \ldots, r_R(j)\})[f_R(\langle 0^j, i \rangle) = n]\}.
\]

(ii) For each \(x \in \Sigma^*\),

\[
x \in B \implies g(x) \in A_{|x|}
\]

\[
x \not\in B \implies g(x) \in R_{|x|}.
\]

It is not hard to see, via shifting gaps with dummy paths, that the class \((r, 1)\)-LWPP equals \(r\)-LWPP. However, what can be shown more generally about the classes \((r_A, r_R)\)-LWPP? For example, for which functions \(r_{1,A}, r_{1,R}, r_{2,A},\) and \(r_{2,A}\) does it hold that \((r_{1,A}, r_{1,R})\)-LWPP \(\subseteq\) \((r_{2,A}, r_{2,R})\)-LWPP? (There are some literature notions that, in different ways, have at least a somewhat similar flavor to our notion, namely, defining classes by being more flexible regarding acceptance types. In particular, the counting classes \(\mathcal{CP}_S\) of \(\mathcal{?}\) and the “C-class” framework of \(\mathcal{?}\) have such a flavor. But in contrast with those, our classes here are ones whose definitions are centered on the notion of gaps.) We do not here undertake that general study, but instead resolve what seems the most compelling question, namely, we prove that \((\text{Poly}, \text{Poly})\)-LWPP = LWPP.

**Definition 3.12.**

\[
(\text{Poly}, \text{Poly})\text{-LWPP} = \bigcup_{c \in \mathbb{N}^+} (n^c + c, n^c + c)\text{-LWPP}.
\]
Theorem 3.13. \((\text{Poly, Poly})\)-LWPP = \text{Poly-LWPP}.

The proof of Theorem 3.13 will be given in a moment. Together with Theorem 3.1, we get the following corollary.

Corollary 3.14. \((\text{Poly, Poly})\)-LWPP = LWPP.

An analogous statement can also be shown for the analogously defined class \((\text{Poly, Poly})\)-WPP:

Theorem 3.15. \((\text{Poly, Poly})\)-WPP = WPP.

We now give the briefly deferred proof of Theorem 3.13.

Proof of Theorem 3.13. It is easy to see that \text{Poly-LWPP} \subseteq \((\text{Poly, Poly})\)-LWPP.

To show \((\text{Poly, Poly})\)-LWPP \subseteq \text{Poly-LWPP}, let \(B\) be a set in \((\text{Poly, Poly})\)-LWPP defined by \(g \in \text{GapP}, f_A, f_R \in \text{FP}, \) and polynomials \(r_A(n) = n^c + c\) and \(r_R(n) = n^c + c\) according to Definition 3.11 and Definition 3.12.

Let \(h\) be a function such that for all \(x \in \Sigma^*\) and \(j \in \mathbb{N}^+\),

\[ h(\langle x, j \rangle) = f_R(\langle 0^{|x|}, j \rangle) - g(x). \]

We have \(h \in \text{GapP}\) since \(f_R \in \text{FP} \subseteq \text{GapP}, g \in \text{GapP},\) and as noted earlier \text{GapP} is closed under subtraction (Fenner et al. 1994). We define \(\hat{g}\) such that for all \(x \in \Sigma^*\),

\[ \hat{g}(x) = \prod_{1 \leq j \leq r_R(|x|)} h(\langle x, j \rangle). \]

By Closure Property 3.4, \(\hat{g} \in \text{GapP}\). Note that for all \(x \in \Sigma^*\),

\[ (3.16) \quad \hat{g}(x) = \prod_{1 \leq j \leq r_R(|x|)} (f_R(\langle 0^{|x|}, j \rangle) - g(x)). \]

Let \(\hat{f}\) be a function such that for all \(\ell \in \mathbb{N}\) and \(i \in \mathbb{N}^+\),

\[ (3.17) \quad \hat{f}(\langle 0^\ell, i \rangle) = \prod_{1 \leq j \leq r_R(\ell)} (f_R(\langle 0^\ell, j \rangle) - f_A(\langle 0^\ell, i \rangle)). \]

It is easy to see that \(\hat{f} \in \text{FP}\). It follows from Eqns. (3.16) and (3.17) that for every \(x \in \Sigma^*\), the following are true.
1. If there exists \( j \in \{1, 2, \ldots, r_R(|x|) \} \) such that \( g(x) = f_R(\langle 0|x|, j \rangle) \), then \( \hat{g}(x) = 0 \).

2. For each \( i \in \{1, 2, \ldots, r_A(|x|) \} \), it holds that \( g(x) = f_A(\langle 0|x|, i \rangle) \) implies that \( \hat{g}(x) = \hat{f}(\langle 0|x|, i \rangle) \).

Hence, for each \( x \in \Sigma^* \),

\[
x \notin B \implies \hat{g}(x) = 0,
\]

\[
x \in B \implies \text{there exists } i \in \{1, 2, \ldots, r_A(|x|) \} \text{ such that } \hat{g}(x) = \hat{f}(\langle 0|x|, i \rangle).
\]

Note also (see Eqn. (3.17)) that for all \( \ell \in \mathbb{N} \) and \( i \in \{1, 2, \ldots, r_A(\ell) \} \), it holds that \( \hat{f}(\langle 0^\ell, i \rangle) \neq 0 \) because for each \( \ell \in \mathbb{N} \), the sets \( A_\ell \) and \( R_\ell \) in Definition 3.11 are disjoint. By Definition 2.7, it thus follows that \( B \in Poly\text{-LWPP} \). \( \square \)

4. Applying our LWPP robustness result to graph reconstruction

**Definition 4.1.** Let \( \langle G_1, G_2, \ldots, G_n \rangle \) be a sequence of graphs and \( G = (V, E) \) a graph with \( V = \{1, 2, \ldots, n\} \). Suppose that there is a permutation \( \pi \in S_n \) such that for each \( k \in \{1, 2, \ldots, n\} \), the graph \( G_{\pi(k)} \) is isomorphic to the graph \( (V - \{k\}, E - \{\{k, \ell\} : \ell \in V\}) \) obtained by deleting vertex \( k \) from \( G \). Then \( \langle G_1, G_2, \ldots, G_n \rangle \) is called a deck of \( G \) and \( G \) is called a preimage of the sequence \( \langle G_1, G_2, \ldots, G_n \rangle \).

A sequence of graphs \( \langle G_1, G_2, \ldots, G_n \rangle \) is called a legitimate deck if there exists a graph \( G \) that is a preimage of \( \langle G_1, G_2, \ldots, G_n \rangle \).

The Reconstruction Conjecture (see, e.g., the surveys Bondy 1991; Bondy & Hemminger 1977; Manvel 1988 and the book Lauri & Scapellato 2003) says that each legitimate deck consisting of graphs with at least two vertices has exactly one preimage up to isomorphism. This conjecture is a very prominent conjecture in graph theory—as mentioned in Section 1 it is perhaps the most
important conjecture in that area—and has been studied for many decades.

Nash-Williams (1978), Mansfield (1982), Kratsch & Hemaspaandra (1994), and Hemaspaandra et al. (2007) introduced various decision problems related to the Reconstruction Conjecture, as part of a stream of work studying the algorithmic and complexity issues of reconstruction. We here are interested mainly in the Legitimate Deck Problem, which is defined as the following decision problem.

**Legitimate Deck (a.k.a. the Legitimate Deck Problem) (Mansfield 1982):** Given a sequence of graphs \(\langle G_1, G_2, \ldots, G_n\rangle\), is \(\langle G_1, G_2, \ldots, G_n\rangle\) a legitimate deck?

Mansfield (1982) showed that the Graph Isomorphism Problem, GI (given two graphs \(G_1\) and \(G_2\), are they isomorphic?), is polynomial-time many-one reducible to the Legitimate Deck Problem. However, to this day it remains open whether there is a polynomial-time many-one reduction from the Legitimate Deck Problem to GI.

**Example 4.2.** Let us give examples of decks that are and are not legitimate.

**A legitimate deck** Consider a deck consisting of a sequence of four graphs, each a 3-clique (\(\Delta\)). Clearly that deck is legitimate, since it is the deck generated by a 4-clique.

**A deck that is not legitimate** Consider a deck consisting of a sequence of four graphs, with three of them being 3-cliques and one being a graph that is simply three isolated points. There are many ways to quickly see that this is not a legitimate deck. For example, the fact that at least one of our four one-point-deleted subgraphs contains a 3-clique means that any underlying graph \(G\) whose deck this is contains a 3-clique as an induced subgraph, and thus every one-point-deleted subgraph of \(G\) must contain at least one edge; yet our fourth graph in the deck does not.
Note: The above type of observation gives an “easy” way to detect that certain decks are not legitimate, and many such tricks exist. It is natural to wonder whether one can put together a suite of easy tricks that perfectly capture all cases of decks that are not legitimate, or more generally, whether one can write a polynomial-time algorithm for the Legitimate Deck Problem. However, by the abovementioned result of Mansfield (1982)—that the Graph Isomorphism Problem is polynomial-time many-one reducible to the Legitimate Deck Problem—it follows that if one could write a polynomial-time algorithm for the Legitimate Deck Problem then as a side effect one would have proven that the Graph Isomorphism Problem in P, which itself would be an earthshaking advance, even in light of the recent work of Babai (2016) that put the Graph Isomorphism Problem into quasipolynomial time.

So how hard is the Legitimate Deck Problem? It is easy to see that the Legitimate Deck Problem is in NP. In the following, we will see that there is some evidence that the Legitimate Deck Problem is not NP-hard, and we will improve that evidence.

Let us define the following function problem.

**Preimage Counting** (Kratsch & Hemachandra 1991; Kratsch & Hemaspaandra 1994): Given a sequence of graphs \( \langle G_1, G_2, \ldots, G_n \rangle \), compute the number \( \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle) \) of all nonisomorphic preimages for the sequence \( \langle G_1, G_2, \ldots, G_n \rangle \).

Köbler, Schöning & Torán (1992) showed the following theorem.

**Theorem 4.3** (Köbler et al. 1992). There is a function \( h : 0^* \rightarrow \mathbb{N} - \{0\} \) in FP such that the function that maps every sequence \( \langle G_1, G_2, \ldots, G_n \rangle \) to \( h(0^n) \) times the number of nonisomorphic preimages, i.e.,

\[
\langle G_1, G_2, \ldots, G_n \rangle \mapsto h(0^n) \cdot \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle)
\]

is in GapP.
Köbler, Schöning & Torán (1992) also showed that Theorem 4.3 has the following corollary.

**Corollary 4.4 (Köbler et al. 1992).** If the Reconstruction Conjecture holds, then the Legitimate Deck Problem is in LWPP.

Since all LWPP sets are PP-low, that immediately gives some evidence that the Legitimate Deck Problem is not NP-hard.

**Corollary 4.5.** If the Reconstruction Conjecture holds, then the Legitimate Deck Problem is not $\text{NP}$-hard (or even $\text{NP}$-Turing-hard) unless $\text{NP}$ is PP-low.

Unfortunately, we do not know whether the Reconstruction Conjecture holds. However, perhaps we can prove the membership of the Legitimate Deck Problem in LWPP under a *weaker assumption* than the Reconstruction Conjecture, for instance, what if the number of nonisomorphic preimages is not always 0 or 1 (as holds under the Reconstruction Conjecture), but rather is merely relatively small, e.g., some constant or some polynomial in the number of vertices (as must hold for each graph class having bounded minimum degree)? We will now use the results of the previous section to prove that this indeed is the case.

**Conjecture 4.6 ($q$-Reconstruction Conjecture).** For each legitimate deck there exist at most $q(n)$ nonisomorphic preimages, where $n$ is the number of graphs in the deck.

For any function $r$, we now define a complexity class $r\text{-LWPP}$. This class may not seem very natural, but we will see that it is very well suited to helping us classify the problem Legitimate Deck. In some sense, it is a tool that we will use in our proof, and then will discard by noting that it in fact turns out to be a disguised version of $r\text{-LWPP}$.

**Definition 4.7.** Let $r$ be any function mapping from $\mathbb{N}$ to $\mathbb{N}$. The class $r\text{-LWPP}$ is the class of all sets $A$ such that there exists a GapP function $g$, and a function $f \in \text{FP}$ that maps from $0^*$ to
The Robustness of LWPP and WPP

**Theorem 4.8.** Let $q$ be any nondecreasing function from $\mathbb{N}$ to $\mathbb{N}$. The following holds.

If the $q$-Reconstruction Conjecture holds, then the Legitimate Deck Problem is in $q$-LWPP.

**Proof.** Suppose that the $q$-Reconstruction Conjecture holds. Let $\langle G_1, G_2, \ldots, G_n \rangle$ be an input to the Legitimate Deck Problem. By our assumption, we have that

$$\langle G_1, G_2, \ldots, G_n \rangle \in \text{Legitimate Deck} \implies \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle) \in \{1, 2, \ldots, q(n)\},$$

and

$$\langle G_1, G_2, \ldots, G_n \rangle \notin \text{Legitimate Deck} \implies \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle) = 0.$$

Let $h \in \text{FP}$ be the function discussed in Theorem 4.3. Then we have that the function $g$ such that, for each sequence $\langle G_1, G_2, \ldots, G_n \rangle,$

$$g(\langle G_1, G_2, \ldots, G_n \rangle) = h(0^n) \cdot \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle)$$

(and on all other inputs $g$’s output is 0) will be such that $g \in \text{GapP}$. It follows that

$$\langle G_1, G_2, \ldots, G_n \rangle \in \text{Legitimate Deck} \implies$$

for some $i \in \{1, 2, \ldots, q(n)\}$, $g(\langle G_1, G_2, \ldots, G_n \rangle) = h(0^n) \cdot i$

and

$$\langle G_1, G_2, \ldots, G_n \rangle \notin \text{Legitimate Deck} \implies$$

$$g(\langle G_1, G_2, \ldots, G_n \rangle) = 0.$$
This almost directly implies that Legitimate Deck ∈ q-LWPP. However, note that to satisfy Definition 4.7, function \( h \) must be a function that depends only on the length of the input \( \langle G_1, G_2, \ldots, G_n \rangle \). The problem here is that (depending on how exactly we decide to encode the graphs \( G_1, G_2, \ldots, G_n \) in the input \( \langle G_1, G_2, \ldots, G_n \rangle \)) even the value \( n \) (the number of graphs in the deck) may depend on the actual input \( \langle G_1, G_2, \ldots, G_n \rangle \) and not only on the length of the input \( \langle G_1, G_2, \ldots, G_n \rangle \).

Fortunately, there is a way to get around this problem. From the length of the input, we get at least an upper bound for \( n \) since we can certainly assume that

\[ n \leq |\langle G_1, G_2, \ldots, G_n \rangle|. \]

Define a function \( \hat{h} \) such that for each \( m \in \mathbb{N} \), \( \hat{h}(0^m) \) is the product of all “h-values” up to length \( m \). That is, define function \( \hat{h} \) such that for all \( m \in \mathbb{N} \),

\[ \hat{h}(0^m) = \prod_{0 \leq i \leq m} h(0^i). \]

Define function \( h' \) such that for all inputs \( \langle G_1, G_2, \ldots, G_n \rangle \),

\[ h'(\langle G_1, G_2, \ldots, G_n \rangle) = \prod_{0 \leq i \leq |\langle G_1, G_2, \ldots, G_n \rangle| \land i \neq n} h(0^i). \]

Now we can see that for all inputs \( \langle G_1, G_2, \ldots, G_n \rangle \),

\[ g(\langle G_1, G_2, \ldots, G_n \rangle) \cdot h'(\langle G_1, G_2, \ldots, G_n \rangle) = \]

\[ \begin{cases} 
0 & \text{if } \langle G_1, G_2, \ldots, G_n \rangle \notin \text{Legitimate Deck} \\
i \cdot \hat{h}(0^{|\langle G_1, G_2, \ldots, G_n \rangle|}) & \text{for some } i \in \{1, 2, \ldots, q(n)\} \text{ Legitimate Deck.}
\end{cases} \]

Note that \( \hat{h}, h' \in \text{FP} \) and \( g \in \text{GapP} \). By Closure Property 3.2 and Closure Property 3.5, the function

\[ x \mapsto g(x) \cdot h'(x) \]
is in GapP. Finally, note that since $q$ is nondecreasing and $n \leq |\langle G_1, G_2, \ldots, G_n \rangle|$, we have that $q(n) \leq q(|\langle G_1, G_2, \ldots, G_n \rangle|)$ in Eqn. (4.9). Thus by Definition 4.7—with the GapP function $g$ there being our $g(x) \cdot h'(x)$ and the function $f$ there being our $\hat{h}$—we have that Legitimate Deck $\in q$-LWPP.  

We want to show that if there exists a polynomial $q$ such that the $q$-Reconstruction Conjecture holds, then Legitimate Deck is in LWPP. To this end, we need the following inclusion.

**Theorem 4.10.** For each function $r : \mathbb{N} \rightarrow \mathbb{N}$, $r$-LWPP $\subseteq r$-LWPP.

**Proof.** Let $A$ be a set in $r$-LWPP via $f_1 \in \text{FP}$ and $g \in \text{GapP}$. Let $f_2 \in \text{FP}$ be the function defined such that for every $n, i \in \mathbb{N}^+$, $f_2((0^n, i)) = i \cdot f_1(0^n)$. Then $A$ is in $r$-LWPP via $f_2 \in \text{FP}$ and $g \in \text{GapP}$.  

**Corollary 4.11.** If the $q$-Reconstruction Conjecture holds for some polynomial $q$, then the Legitimate Deck Problem is in LWPP.

**Proof.** Suppose the $q$-Reconstruction Conjecture holds for nondecreasing polynomial $q$. (If $q$ is not nondecreasing but the $q$-Reconstruction Conjecture holds, then obviously we can replace $q$ with a nondecreasing polynomial $q'$ that on each input $n$ is greater than or equal to $q(n)$ and the $q'$-Reconstruction Conjecture will hold. So we may w.l.o.g. take it that $q$ is nondecreasing.) By Theorem 4.8 and Theorem 4.10, Legitimate Deck $\in q$-LWPP $\subseteq q$-LWPP and hence Legitimate Deck $\in \text{Poly-LWPP}$. With Theorem 3.1, it follows that Legitimate Deck $\in$ LWPP.

This gives us our new, more flexible—though still conditional—evidence that the Legitimate Deck Problem is not NP-hard.

**Corollary 4.12.** If the $q$-Reconstruction Conjecture holds for some polynomial $q$, then the Legitimate Deck Problem is not NP-hard (or even NP-Turing-hard) unless NP is PP-low.
Definition 4.13 (Kratsch & Hemaspaandra 1994). Let $\mathcal{H}$ be any class of graphs. The Legitimate Deck Problem restricted to $\mathcal{H}$ consists of all sequences of graphs $\langle G_1, G_2, \ldots, G_n \rangle$ such that $\langle G_1, G_2, \ldots, G_n \rangle$ is a legitimate deck and for each $i \in \{1, 2, \ldots, n\}$, $G_i$ is in $\mathcal{H}$.

Note that the above definition is so flexible that it allows even the case where the preimage(s) may not be in $\mathcal{H}$.

Theorem 4.14. Let $\mathcal{H}$ be any P-recognizable class of graphs such that decks consisting only of graphs in $\mathcal{H}$ have a number of nonisomorphic preimages that is bounded polynomially in the number of graphs in the deck. Then the Legitimate Deck Problem restricted to $\mathcal{H}$ is in LWPP.

Proof. Let $\mathcal{H}$ be any P-recognizable class of graphs and $q$ a nondecreasing polynomial such that decks consisting only of graphs in $\mathcal{H}$ have a number of nonisomorphic preimages that is bounded by $q(n)$, where $n$ is the number of graphs in the deck.

First, we show that the Legitimate Deck Problem restricted to $\mathcal{H}$ is in $q$-\textit{LWPP}. Let $\langle G_1, G_2, \ldots, G_n \rangle$ be an input to the Legitimate Deck Problem. Check if for every $i \in \{1, 2, \ldots, n\}$, $G_i$ is in $\mathcal{H}$. If this is not the case, then reject in the sense of $q$-\textit{LWPP}, i.e., produce a gap of zero. Otherwise, the deck $\langle G_1, G_2, \ldots, G_n \rangle$ has at most $q(n)$ preimages, i.e., \text{PCount}(\langle G_1, G_2, \ldots, G_n \rangle \leq q(n)$. Proceed as in the proof of Theorem 4.8.

Since by Theorem 4.10 and Theorem 3.1, $q$-\textit{LWPP} $\subseteq q$-\textit{LWPP} $\subseteq$ LWPP, it follows that the Legitimate Deck Problem restricted to $\mathcal{H}$ is in LWPP. □

In two of this section’s corollaries, we used the fact that all LWPP sets are PP-low. We mention that since all LWPP sets are also $C_{=}P$-low (Fenner et al. 1994; Köbler et al. 1992), the altered versions of Corollary 4.5 and Corollary 4.12 in which the conclusion is changed from “unless NP is PP-low” to “unless NP is $C_{=}P$-low” both hold, respectively due to Köbler, Schöning & Torán (1992) and the present paper. So, regarding the latter, we have the following.
Corollary 4.15. If the $q$-Reconstruction Conjecture holds for some polynomial $q$, then the Legitimate Deck Problem is not NP-hard (or even NP-Turing-hard) unless NP is $C\subseteq P$-low.

5. Optimality of our LWPP robustness result

It is easy to see that the proof of Theorem 3.1 breaks down if Poly-LWPP is replaced by the analogous class where the size of the set of allowed gap values can be larger than polynomial in the input length. This does not necessarily imply that the corresponding theorem does not hold. However, in this section, we establish that relativizable proof techniques are not sufficient to improve Theorem 3.1 from Poly-LWPP to $r$-LWPP for any function $r$ that is not polynomially bounded. That is, we show that Theorem 3.1 is optimal with respect to what can be proven by relativizable proof techniques. (We consider one of the most important uses of oracles to be, as a counterpoint to positive results, giving evidence that the positive results are not blindly missing possible improvements—more specifically, showing that no relativizable technique can improve the given result. That is the use to which we put oracles in this paper. Good discussions of the meaning and limitations of oracle results are available in the literature, see Fortnow 1994; Hartmanis et al. 1992, 1990.)

But first, let us briefly consider the “more extreme” case of the class $Exp$-LWPP, where the number of allowed gap values can be an exponential function.

Definition 5.1. $Exp$-LWPP = \( \bigcup_{c \in \mathbb{N}^+} 2^{n^c} + \cdot LWPP \).

We show that the whole class NP is contained in $Exp$-LWPP. This gives strong evidence that $Exp$-LWPP $\not\subseteq$ LWPP because LWPP is known to be low for PP (Fenner et al. 1994), but NP is widely believed not to be low for PP.

Theorem 5.2. $\text{coC}_P \subseteq Exp$-LWPP (and hence—since $\text{coC}_P \supseteq Exp$-LWPP is immediate from the definitions—$\text{coC}_P = Exp$-LWPP).
Since $NP \subseteq \text{coC} = P$ and all LWPP sets are PP-low, we obtain the following corollaries.

**Corollary 5.3.** $NP \subseteq \text{Exp-LWPP}$.

**Corollary 5.4.** If $\text{Exp-LWPP} = \text{LWPP}$, then $NP \subseteq \text{LWPP}$ and $PP^{NP} = PP$ (and, indeed, $PP^{\text{coC} = P} = PP$).

All of the above, except the PP-lowness claims, analogously hold for $\text{Exp-WPP}/\text{WPP}$, since $r$-$\text{LWPP}$ is a subset of $r$-$\text{WPP}$. So for example we have the following.

**Corollary 5.5.** If $\text{Exp-WPP} = \text{WPP}$, then $NP \subseteq \text{WPP}$ (and, indeed, $\text{coC} = P \subseteq \text{WPP}$).

**Proof of Theorem 5.2.** Let $A \in \text{coC} = P$. By Theorem 2.9, there exists a function $g \in \text{GapP}$ such that for every $x \in \Sigma^*$,

\[
x \in A \implies g(x) \neq 0, \quad \text{and} \quad x \notin A \implies g(x) = 0.
\]

Let $p$ be a polynomial such that for each $x \in \Sigma^*$, $-2^{p(|x|)} \leq g(x) \leq 2^{p(|x|)}$. Let $f \in \text{FP}$ be the function such that, for every $n \in \mathbb{N}^+$ and every $i \in \mathbb{N}^+$,

\[
f(\langle 0^n, i \rangle) = \begin{cases} 
\frac{i}{2} & \text{if } i \text{ is even} \\
-(i+1)/2 & \text{otherwise}
\end{cases}
\]

We can now see that according to Definition 2.6, $A \in 2^{p(n)+1}$-$\text{LWPP}$ and hence $A \in \text{Exp-LWPP}$. \hfill $\square$

**Theorem 5.6.** Let $r$ be any function from $\mathbb{N}$ to $\mathbb{N}$ such that for every $c \in \mathbb{N}$, $r \notin O(n^c)$. Then there exists an oracle $O$ such that $r$-$\text{LWPP}^O \nsubseteq \text{LWPP}^O$.

To prove Theorem 5.6, we will encode nondeterministic oracle Turing machines by low-degree multivariate polynomials. This framework is apparently folklore and has been used, for example, by de Graaf & Valiant (2002) to construct a relativized world
where the quantum complexity class EQP is not contained in the modularity-based complexity class MOD\(_{p^k}\)P. The general idea of replacing oracle machines by simpler combinatorial objects such as circuits, decision trees, or polynomials and then using properties of such combinatorial objects to show the existence of a desired oracle dates to the seminal work of Furst, Saxe & Sipser (1984), who made the connection between circuit lower bounds and the relativization of the polynomial hierarchy—a connection that has led to the resolution of many previously long-open relativized questions, such as the achievement of an oracle making the polynomial hierarchy infinite (Håstad 1987; Yao 1985) and of oracles making the polynomial hierarchy extend exactly \(k\) levels (Ko 1989). Our separation’s technique, although carried out within the above framework, is a quite different item than the framework itself. In particular, our separation is achieved by a somewhat fragile, diagonalization-based construction that is using issues regarding the density of the primes to yield its correctness argument about whether a certain required type of set exists.

As indicated above, in Definition 5.7 and Proposition 5.8 below, we will encode nondeterministic oracle Turing machines by low-degree multivariate polynomials, in order to obtain polynomials that compute the gap of polynomial-time nondeterministic oracle machines. Definition 5.7 and Proposition 5.8 appear implicitly in the proof of the result called “Lemma 2” in a paper by Beigel (1997). The earliest reference we know for a paper making the connection between numbers of computation paths in nondeterministic oracle machines and low-degree polynomials is a paper by Tarui (1991, Section 1.2).

**Definition 5.7** (Beigel 1997). Let \(N^{(c)}(x)\) be a nondeterministic polynomial-time oracle Turing machine with running time \(t(.)\) and input \(x \in \Sigma^*\). Let \(x_1, x_2, \ldots, x_m\) be an enumeration of all strings in \(\Sigma^*\) up to length \(t(|x|)\).

A polynomial encoding of \(N^{(c)}(x)\) is a multilinear polynomial \(p \in \mathbb{Z}[y_1, y_2, \ldots, y_m]\) defined as follows: Call a computation path \(\rho\) valid if \(\rho\) is a computation path of \(N^D(x)\) for some oracle \(D \subseteq \Sigma^*\). Let \(x_{i_1}, x_{i_2}, \ldots, x_{i_\ell}\) be the distinct queries along a valid computation path \(\rho\). Create a monomial \(\text{mono}(\rho)\) that is the product of
terms $z_{ik}, k \in \{1, 2, \ldots, \ell\}$, where $z_{ik} = y_{ik}$ if $x_{ik}$ is answered “yes” and $z_{ik} = (1 - y_{ik})$ if $x_{ik}$ is answered “no” along $\rho$. Define

$$p(y_1, y_2, \ldots, y_m) = \sum_{\rho: \rho \text{ is valid}} \text{sign}(\rho) \cdot \text{mono}(\rho).$$

Here, $\text{sign}(\rho) = 1$ if $\rho$ is an accepting path and $\text{sign}(\rho) = -1$ if $\rho$ is a rejecting path.

The next proposition states that the multilinear polynomial $p$ has low total degree, and contains all the necessary information about $N(\cdot)(x)$ to yield the value gap$_{NB}(x)$ for every oracle $B \subseteq \Sigma^*$. $\chi_B$ denotes the characteristic function of $B$.

**Proposition 5.8 (Beigel 1997).** The polynomial $p(y_1, y_2, \ldots, y_m)$ defined in Definition 5.7 has the following properties:

(i) $\deg(p) \leq t(|x|)$, and

(ii) for each $B \subseteq \Sigma^*$, $p(\chi_B(x_1), \chi_B(x_2), \ldots, \chi_B(x_m)) = \text{gap}_{NB}(x)$.

The expression $p \mid \text{val}$ means that $p$ is a divisor of $\text{val}$, i.e., $\text{val} \mod p = 0$.

**Lemma 5.9 (Spakowski et al. 2005).** Let $N, q \in \mathbb{N}$ be such that $q$ is a prime and $q \leq N/2$. Let $s \in \mathbb{Z}[y_1, y_2, \ldots, y_N]$ be a multilinear polynomial with total degree $\deg(s) < q$. If for some $\text{val} \in \mathbb{Z}$, it holds that

(i) $s(0, 0, \ldots, 0) = 0$, and

(ii) $s(y_1, y_2, \ldots, y_N) = \text{val}$, for every $y_1, y_2, \ldots, y_N \in \{0, 1\}$ with $q = \sum_{1 \leq i \leq N} y_i$,

then $q \mid \text{val}$.

Lemma 5.10 states a variant of the prime number theorem. We will need it in our oracle construction to get a lower bound for the number of primes in a given set of natural numbers (see set $S$ defined below).
Lemma 5.10 (Rosser & Schoenfeld 1962). For every $n \geq 17$, the number of primes less than or equal to $n$, $\pi(n)$, satisfies

$$\pi(n) > n / \ln n.$$ 

Proof of Theorem 5.6. First, we need a test language. For every set $B \subseteq \Sigma^*$, define $L_B$ as

$$L_B = \{0^n \mid \|B^n\| > 0\},$$

where $B^n$ denotes $B \cap \Sigma^n$. If $B$ satisfies the condition that for every length $n$ it holds that $\|B^n\| \leq r(n)$, then $L_B \in r$-LWPP$^B$. (To see this, note that the function $d(0^n) = \|B^n\|$ is a $\#P^B$ function, where $\#P$ is Valiant’s (1979) class of functions that count the number of accepting paths of nondeterministic polynomial-time Turing machines. But $\#P \subseteq \text{GapP}$ (Fenner et al. 1994), and that fact itself relativizes, so $d(0^n) \in \text{GapP}^B$. Since our test language $L_B$ should not contain $0^n$ when $d(0^n) = 0$ and should contain $0^n$ when $1 \leq d(0^n) \leq r(n)$, and since by our “if $B$ satisfies” those are the only possibilities, we have $L_B \in r$-LWPP$^B$.)

We will construct an oracle $B$ such that for each $n$, $\|B^n\| \leq r(n)$ and $L_B \notin \text{LWPP}^B$. Let $(N_j, M_j, p_j)_{j \geq 1}$ be an enumeration of all triples such that $N_j$ is a nondeterministic polynomial-time oracle Turing machine, $M_j$ is a deterministic polynomial-time oracle Turing machine computing a function, and $p_j$ is a monotonically increasing polynomial such that the running time of both $N_j$ and $M_j$ is bounded by $p_j$ regardless of the oracle. We construct the oracle $B$ in stages. In stage $j$, we decide the membership in $B$ of strings of length $n_j$ and extend the initial segment $B_{j-1}$ of $B$ to $B_j$. Initially, we set $B_0 = \emptyset$.

Stage $j$, where $j \geq 1$: Let $n_j$ be large enough that: (a) $n_j > p_m(n_{j-1})$ for all $m < j$ (to ensure that the previous stages are not affected), (b) $r(n_j) \geq p_j(n_j)^4$, (c) $(2^{n_j} - p_j(n_j)) / 2 \geq p_j(n_j)^4$, (d) $p_j(n_j)^3 - p_j(n_j) \geq p_j(n_j)^2$, and (e) $p_j(n_j) \geq 9$. Such an $n_j$ exists because $p_j$ is a monotonically increasing polynomial and for each $c \in \mathbb{N}$, $r \notin \mathcal{O}(n^c)$. We diagonalize against nondeterministic polynomial-time oracle Turing machine $N_j$ and deterministic polynomial-time oracle Turing machine $M_j$. That is, we make sure
that $L_B$ is not decided according to the definition of LWPP by GapP function $g$ computed by $N_j$ together with FP function $f$ computed by $M_j$ (see Definition 2.5). Let $val$ be the value computed by $M_j^{B_{j-1}}(0^{n_j})$. Because of the condition $0 \notin \text{range}(f)$ in the definition of LWPP, we can assume that $val \neq 0$. (If $val = 0$, then we can right away go to stage $j + 1$.)

Let 
$$T = \{ w \in \Sigma^{n_j} | M_j^{B_{j-1}}(0^{n_j}) \text{ queries } w \}.$$ 

Note that $\|T\| \leq p_j(n_j)$ since the computation time of $M_j^{B_{j-1}}(0^{n_j})$ is bounded by $p_j(n_j)$. In the following, we will never add any string from $T$ to the oracle. This ensures that the value $val$ computed by $M_j^{B_{j-1}}(0^{n_j})$ is never changed when we replace oracle $B_{j-1}$ by $B_j$.

(*) We choose a set $C \subseteq \Sigma^{n_j} - T$ such that

- $\|C\| \in \{1, 2, \ldots, r(n_j)\}$ and $\text{gap}_{N_j^{B_{j-1} \cup C}}(0^{n_j}) \neq val$, or
- $\|C\| = 0$ and $\text{gap}_{N_j^{B_{j-1} \cup C}}(0^{n_j}) \neq 0$.

Let $B_j = B_{j-1} \cup C$.

**End of Stage $j$.**

This construction guarantees that for each $n$, $\|B^{=n}\| \leq r(n)$ and $L_B \notin \text{LWPP}^B$. Thus our proof is complete if we can show that it is always possible to find a set $C$ satisfying (*). We state and prove that as the following claim and its proof.

**Claim 5.11.** For each $j \geq 1$, there exists a set $C$ satisfying (*).

**Proof of Claim 5.11.** Suppose that in stage $j$ no set $C$ satisfying (*) exists. Then, for every $C \subseteq \Sigma^{n_j} - T$, the following holds:

$$\|C\| \in \{1, 2, \ldots, r(n_j)\} \implies \text{gap}_{N_j^{B_{j-1} \cup C}}(0^{n_j}) = val,$$

and

$$\|C\| = 0 \implies \text{gap}_{N_j^{B_{j-1} \cup C}}(0^{n_j}) = 0.$$ 

Let $s' \in \mathbb{Z}[y_1, y_2, \ldots, y_m]$ be the polynomial encoding of $N_j^{(*)}(0^{n_j})$ as in Definition 5.7. W.l.o.g. assume that $x_1, x_2, \ldots, x_N$ enumerate the strings in $\Sigma^{n_j} - T$, and $x_{N+1}, x_{N+2}, \ldots, x_m$ enumerate the
remaining strings of length at most $p_j(n_j)$. By Proposition 5.8, polynomial $s'(y_1, y_2, \ldots, y_m)$ satisfies the following:

1. For all $C \subseteq \Sigma^{n_j} - T$, it holds that
   
   \[ s'(\chi_C(x_1), \chi_C(x_2), \ldots, \chi_C(x_N), \chi_{B_{j-1}}(x_{N+1}), \ldots, \chi_{B_{j-1}}(x_m)) = \text{gap}_{N^j B_{j-1} \cup C}(0^{n_j}), \text{ and} \]

2. $\deg(s') \leq p_j(n_j)$.

Note that the sets $B_{j-1}$ and $\Sigma^{n_j} - T$ are disjoint. The values $\chi_{B_{j-1}}(x_{N+1}), \ldots, \chi_{B_{j-1}}(x_m)$ do not depend on $C$. We fix these values and obtain the following polynomial $s(y_1, y_2, \ldots, y_N)$:

\[ s(y_1, y_2, \ldots, y_N) = s'(y_1, y_2, \ldots, y_N, \chi_{B_{j-1}}(x_{N+1}), \ldots, \chi_{B_{j-1}}(x_m)). \]

Hence the polynomial $s(y_1, y_2, \ldots, y_N)$ satisfies the following:

1. For all $C \subseteq \Sigma^{n_j} - T$, it holds that
   
   \[ s(\chi_C(x_1), \chi_C(x_2), \ldots, \chi_C(x_N)) = \text{gap}_{N^j B_{j-1} \cup C}(0^{n_j}), \text{ and} \]

2. $\deg(s) \leq \deg(s') \leq p_j(n_j)$.

Statements (5.13) and (5.12) imply that

- $s(0, 0, \ldots, 0) = 0$, and
- for all $z_1, z_2, \ldots, z_N \in \{0, 1\}$ such that $\sum_{1 \leq i \leq N} z_i \in \{1, 2, \ldots, r(n_j)\}$, we have $s(z_1, z_2, \ldots, z_N) = \text{val}$.

By Lemma 5.9, for every prime $q$ in

\[ S = \{\deg(s) + 1, \deg(s) + 2, \ldots, \min(N/2, r(n_j))\}, \]

it holds that $q \mid \text{val}$.

To obtain a lower bound for $\text{val}$, we determine a lower bound for the number of primes in $S$. First, note that at the beginning of stage $j$, we have taken $n_j$ large enough such that $N/2 \geq (2^{n_j} - p_j(n_j))/2 \geq p_j(n_j)^4$ and $r(n_j) \geq p_j(n_j)^4$, and thus
min(N/2, r(n_j)) \geq p_j(n_j)^4. In light of Lemma 5.10, we hence obtain (to support the second of the two “≥” signs here is why we earlier—as condition (e)—made certain to ensure that \( p_j(n_j) \geq 9 \)\)

\[
\pi(\min(N/2, r(n_j))) \geq \frac{p_j(n_j)^4}{\ln(p_j(n_j)^4)} \geq p_j(n_j)^3.
\]

Further, we obviously have

\[
\pi(\deg(s)) \leq \deg(s) \leq p_j(n_j).
\]

Hence the number of primes in \( S \) is at least \( p_j(n_j)^3 - p_j(n_j) \), which is (by the choice of \( n_j \) at the beginning of stage \( j \)) greater than or equal to \( p_j(n_j)^2 \). Since each prime in \( S \) is greater than or equal to \( 2 \) and \( \text{val} \neq 0 \), we have that the absolute value of \( \text{val} \) is at least \( 2^{p_j(n_j)^2} \). Yet we also must have that the absolute value of \( \text{val} \) is less than or equal to \( 2^{p_j(n_j)} \), since the running time of \( M_j^{(\epsilon)(0^{n_j})} \) is bounded by \( p_j(n_j) \) regardless of the oracle. This is a contradiction.

So for each \( j \geq 1 \), \( B_{j-1} \) can always be extended in stage \( j \) as required. This finishes the proof of Claim 5.11. \( \square \)

And since Claim 5.11 was all that remained in our proof of Theorem 5.6, that theorem itself is now proven. \( \square \)

Is the oracle constructed in the proof of Theorem 5.6 necessarily recursive? It might not be, since Theorem 5.6 put no complexity or computability restrictions on the function \( r \). However, we note that our construction is recursive in \( r \). Thus if \( r \) is a computable function, then the oracle that our construction builds is certainly recursive.

### 6. LWPP

In this section, we show that, despite the fact that LWPP is robust up to a polynomial number of targets (in the real world and every relativized world), the analogue of LWPP defined in terms of \(#P\) rather than \(\text{GapP} \) functions is in some relativized worlds not even robust for the case of two targets.
Recall that Theorem 3.1 of Section 3 established a robustness property of LWPP, namely, that Poly-LWPP = LWPP. That is, having one target value for acceptance and having a list of target values for acceptance yield the same class of languages, in the content of LWPP, which, recall, is defined in terms of the values of GapP functions. That robustness result is itself robust in the sense that it holds both in the real world and, it is easy to see, in every relativized world.

On the other hand, this equivalence for LWPP, in terms of descriptive richness, between one target value and a polynomial number of values, may not hold even for quite similar counting-class situations. In particular, in this section we prove that in some relativized worlds, for the analogue of LWPP defined in terms of \#P rather than GapP functions, having even two target values for acceptance yields a richer class of languages than having one value. So it is not the case that single targets and lists of targets inherently function identically as to descriptive richness for counting classes.

The analogue of WPP defined using \#P functions rather than GapP functions already exists in the literature, namely it is the class known as $F_{\#P}$ that was recently introduced in an interesting paper by Cox & Pay (2018). Similarly, we here define, and denote as LWPP$^+$, the analogue of LWPP except defined using \#P functions rather than GapP functions. Clearly, LWPP$^+ \subseteq$ LWPP, and it would be natural to guess that the containment is strict, although obviously proving that strictness is a much stronger result than proving $P \neq \text{PSPACE}$ and so seems beyond current techniques.

We now define LWPP$^+$ and then we prove as Theorem 6.3 that, unlike LWPP, there are relativized worlds where size-two target sets yield languages that cannot be obtained via any size-one target set.

**Definition 6.1.** LWPP$^+$ is the class of all sets $A$ such that there exists a function $g \in \#P$ and a function $f \in FP$ that maps from $0^*$ to $\mathbb{N}^+$ such that for all $x \in \Sigma^*$,

\[
x \in A \implies g(x) = f(0^{||x||})
\]

\[
x \notin A \implies g(x) = 0.
\]
**Definition 6.2.** 2-LWPP$^+$ is the class of all sets $A$ such that there exists a function $g \in \#P$ and functions $f_1, f_2 \in \text{FP}$ that map from $0^*$ to $\mathbb{N}^+$ such that for all $x \in \Sigma^*$,

\[
x \in A \implies g(x) \in \{f_1(0^{|x|}), f_2(0^{|x|})\}
\]

\[
x \notin A \implies g(x) = 0.
\]

**Theorem 6.3.** There exists an oracle $O$ such that $(\text{LWPP}^+)^O \subsetneq (2\text{-LWPP}^+)^O$.

We will soon prove this theorem, but we first state and prove a corollary, and will give some groundwork for the theorem’s proof, and also pointers to some related work.

**Corollary 6.4.** There exists an oracle $O$ such that \(\text{LWPP}^O \not\subseteq (\text{LWPP}^+)^O\).

**Proof.** It follows from the definitions that, for each oracle $Q$, $(2\text{-LWPP}^+)^Q \subseteq \text{Poly-LWPP}^Q$. Since, as mentioned above, Theorem 3.1 clearly relativizes, we have that for each oracle $Q$, $(2\text{-LWPP}^+)^Q \subseteq \text{LWPP}^Q$. Thus, for the oracle $O$ of Theorem 6.3, \(\text{LWPP}^O \not\subseteq (\text{LWPP}^+)^O\). \(\square\)

Let us give the definition of the complexity class $\text{UP}_{\leq 2}$.

**Definition 6.5 (Beigel 1989).** $\text{UP}_{\leq 2}$ is the class of all sets $A$ such that there exists a function $g \in \#P$ such that for all $x \in \Sigma^*$,

\[
x \in A \implies g(x) \in \{1, 2\}
\]

\[
x \notin A \implies g(x) = 0.
\]

Just as UP (Valiant 1976) (“unambiguous NP”) is the restriction of NP to allowing at most one accepting path, $\text{UP}_{\leq 2}$ is the restriction of NP to allowing at most two accepting paths. Analogously to the fact that 1-to-1 complexity-theoretic one-way functions exist if and only if $P \neq \text{UP}$ (Grollmann & Selman 1988; Ko 1985), it holds that 2-to-1 complexity-theoretic one-way functions exist if and only if $P \neq \text{UP}_{\leq 2}$ (Hemaspaandra & Zimand 1993).
Interestingly, $P \neq UP \iff P \neq UP_{\leq 2}$ (Watanabe 1988), and so these two class inequalities stand or fall together.

Note that $UP_{\leq 2} \subseteq 2$-LWPP$^+$. To prove Theorem 6.3, we will prove the stronger result that there is an oracle relative to which even $UP_{\leq 2}$ is not contained in LWPP$^+$.

**Theorem 6.6.** There exists an oracle $\mathcal{O}$ such that

$$UP_{\leq 2}^\mathcal{O} \not\subseteq (\text{LWPP}^+)^\mathcal{O}.$$  

**Proof.** First, we need a test language. For every set $B \subseteq \Sigma^*$, define $L_B$ as

$$L_B = \{0^n \mid \|B^n\| > 0\}.$$  

If $B$ satisfies the condition that for every length $n$ it holds that $\|B^n\| \leq 2$, then $L_B \in UP_{\leq 2}^B$.

In this proof, we are going to centrally use a slightly unusual notion of what is a computation path of a nondeterministic polynomial-time oracle Turing machine on a given oracle and a given input. Our notion of a computation path of such a machine relative to some oracle on some input will give not just the nondeterministic choices of the path, but will also contain a set of strings that ideally should capture exactly the set of queries that are asked and answered Yes along the path, and a set of strings that ideally should capture exactly the set of queries that are asked and answered No along the path. However, this is made a bit trickier as to formalizing it by the fact that we’ll be dealing with various hypothetical oracles in our proof, and so sometimes things will fail to be a computation path of a machine relative to an oracle due to the action of the machine on the given input under the given oracle not being in harmony with nondeterministic bits or sets that are specifying the computation path. So our basic notion of a path will not enforce much on the two sets; rather, only when asserting that the path is in harmony with a machine under some oracle on a certain input will we require that the “natural” things hold.

**Definition 6.7.** (i) A computation path is a triple: a binary string, a set of strings, and second set of strings that is disjoint from the first set of strings.
(ii) (This is implicit in the above part of the definition, but is here stated explicitly for clarity, as this will be tacitly but importantly drawn on in our proof.) Two computation paths \( \rho_1 \) and \( \rho_2 \) are equal if and only if, for each \( i \in \{1, 2, 3\} \), the \( i \)th component of \( \rho_1 \) is identical to the \( i \)th component of \( \rho_2 \).

(iii) A computation path \( \rho \) is said to be a computation path of a nondeterministic polynomial-time oracle Turing machine \( N \) with oracle \( \mathcal{O} \) on input \( x \) if \( \rho \) consists of a binary string \( \rho_{\text{choices}} \), a set of strings \( Q^+(\rho) \), and a set of strings \( Q^-(\rho) \), that satisfy all of the following conditions.

(a) \( Q^+(\rho) \subseteq \mathcal{O} \) and \( Q^-(\rho) \subseteq \mathcal{O}^c \).

(b) When one runs machine \( N \) with oracle \( Q^+(\rho) \) on input \( x \), there is a path \( r \) in that machine’s computation tree such that all the following hold:

1. the sequence of nondeterministic choices in \( r \) is precisely \( \rho_{\text{choices}} \) (i.e., the path makes precisely \(|\rho_{\text{choices}}|\) (binary) choices and \( \rho_{\text{choices}} \) lists those choices in the order they are made); and

2. the set of strings queried along path \( r \) is precisely \( Q^+(\rho) \cup Q^-(\rho) \).

(iv) If in the above we after (iii.b.1) and (iii.b.2) add a new third condition that states “the path \( r \) accepts,” then that is our definition of computation path \( \rho \) being an accepting computation path of nondeterministic polynomial-time oracle Turing machine \( N \) with oracle \( \mathcal{O} \) on input \( x \).

Given an oracle \( \mathcal{O} \), a computation path \( \rho \) may or may not be a computation path of a nondeterministic Turing machine \( N \) relative to \( \mathcal{O} \) on input \( x \). For example, if what the computation path specifies is inconsistent with the machine’s actions on that input and oracle, then it would not be a computation path of that machine with that oracle and that input. As other examples, if relative to the machine’s action on that oracle and that input the \( \rho_{\text{choices}} \) of a computation path has too few or too many bits to

---

4 Note that this implies that \( Q^+(\rho) \) and \( Q^-(\rho) \) are disjoint.
precisely yield a complete, actual path (in the traditional sense of the word, not the more encumbered one we are using here) through the machine’s action, or if one of the two \( Q \) sets contains a string that is not queried by the machine when it is run on \( x \) with oracle \( O \) making nondeterministic choices \( \rho_{\text{choices}} \), then the computation path would not be a computation path of that machine with that oracle and that input.

To make the previous paragraph more concrete, consider the case where the machine \( N \) on input \( x \), for any given oracle \( O \), asks without making any nondeterministic moves the oracle query \( \epsilon \) (i.e., the empty string) and if the answer is Yes makes a nondeterministic move but otherwise immediately halts. A computation path \( \rho_1 \) of course can have \( \epsilon \in Q^+(\rho_1) \). But no such \( \rho_1 \) can be a computation path of machine \( N \) with oracle \( O = \emptyset \) on input \( x \), since condition (iii.a) of Definition 6.7 is violated. As another example, though computation paths can, of course, have one or more nondeterministic choice bits in their first component, no such path \( \rho_2 \) can possibly be a computation path of nondeterministic polynomial-time oracle Turing machine \( N \) with oracle \( \emptyset \) on input \( x \). That is so because there will be zero nondeterministic choices made with oracle \( \emptyset \). And so either the path \( \rho_2 \) would need to violate conditions (iii.a) or (iii.b.2) of Definition 6.7 by not having \( \epsilon \in Q^-(\rho_2) \wedge \epsilon \notin Q^+(\rho_2) \) hold; or if that did hold, would violate condition (iii.b.1) of Definition 6.7 by having one or more nondeterministic bits listed for a path that should have had zero nondeterministic choice bits.

Let \((N_j, M_j, p_j)_{j \geq 1}\) be an enumeration of all triples such that \( N_j \) is a nondeterministic polynomial-time oracle Turing machine, \( M_j \) is a deterministic polynomial-time oracle Turing machine computing a function, and \( p_j \) is a monotonically increasing polynomial such that the running time of both \( N_j \) and \( M_j \) is bounded by \( p_j \) regardless of the oracle. We construct the oracle \( B \) in stages. In stage \( j \), we decide the membership in \( B \) of strings of length \( n_j \) and extend the initial segment \( B_{j-1} \) of \( B \) to \( B_j \). Initially, we set \( B_0 = \emptyset \).

**Stage \( j \), where \( j \geq 1 \):** Let \( n_j \) be large enough that: (a) \( n_j > p_m(n_{j-1}) \) for all \( m < j \) (to ensure that the previous stages are not
affected), and (b) \(2^{n_j} - p_j(n_j) > 6p_j(n_j) + 1\).

We diagonalize against nondeterministic polynomial-time oracle Turing machine \(N_j\) and deterministic polynomial-time oracle Turing machine \(M_j\). That is, we make sure that \(L_B\) is not decided according to the definition of LWPP\(^+\) by \#P function \(g\) computed by \(N_j\) together with FP function \(f\) computed by \(M_j\) (see Definition 6.1). Let \(val\) be the value computed by \(M_{j-1}^B(0^{n_j})\). Because of the condition \(0 \notin \text{range}(f)\) in the definition of LWPP\(^+\), we can assume that \(val \neq 0\). (If \(val = 0\), then we can right away go to stage \(j + 1\).)

Let

\[
T = \{w \in \Sigma^{n_j} | M_{j-1}^B(0^{n_j})\text{ queries } w\}.
\]

Note that \(\|T\| \leq p_j(n_j)\) since the computation time of \(M_{j-1}^B(0^{n_j})\) is bounded by \(p_j(n_j)\). In the following, we will never add any string from \(T\) to the oracle. This ensures that the value \(val\) computed by \(M_{j-1}^B(0^{n_j})\) is never changed when we replace oracle \(B_{j-1}\) by \(B_j\).

\((**)\) We (and we will soon prove that such a set must exist) choose a set \(C \subseteq \Sigma^{n_j} - T\) such that

- \(\|C\| \in \{1, 2\}\) and \(\text{acc}_{N_{j-1}^B \cup C}(0^{n_j}) \neq val\), or
- \(\|C\| = 0\) and \(\text{acc}_{N_{j-1}^B \cup C}(0^{n_j}) \neq 0\).

Let \(B_j = B_{j-1} \cup C\).

End of Stage \(j\).

Note that, assuming that it is always possible to find a set \(C\) satisfying \((**), the construction has guaranteed that, for each \(n\), \(\|B^n\| \leq 2\) (and thus clearly it holds that \(L_B \in \text{UP}^B_{\leq 2}\)), and has also guaranteed (in particular, by the choice of \(C\) and the proof’s diagonalization framework) that \(L_B \notin (\text{LWPP}^+)^B\). Thus our proof is complete if we can show that it is always possible to find a set \(C\) satisfying \((**). We now state and prove that as Claim 6.8.

**Claim 6.8.** For each \(j \geq 1\), there exists a set \(C\) satisfying \((**).**

**Proof of Claim 6.8.** Suppose that in stage \(j\) no set \(C\) satisfying \((**\) exists. Then, for every \(C \subseteq \Sigma^{n_j} - T\), the following
holds:

\[ \|C\| \in \{1, 2\} \implies \text{acc}_{N_j^{B_{j-1} \cup C}}(0^n_j) = \text{val}, \text{ and} \]

\[ \|C\| = 0 \implies \text{acc}_{N_j^{B_{j-1} \cup C}}(0^n_j) = 0. \]

It follows that (i) \(N_j^{B_{j-1}}(0^n_j)\) has no accepting computation paths, and (ii) for each string \(\alpha \in \Sigma^{n_j} - T\), \(N_j^{B_{j-1} \cup \{\alpha\}}(0^n_j)\) has exactly \(\text{val}\) distinct accepting computation paths \(\rho_1(\alpha), \rho_2(\alpha), \ldots, \rho_{\text{val}}(\alpha)\). For each \(\alpha \in \Sigma^{n_j} - T\), let \(A_\alpha\) be the set of accepting computation paths of \(N_j^{B_{j-1} \cup \{\alpha\}}(0^n_j)\). Then, for each \(\alpha \in \Sigma^{n_j} - T\), we have that \(\|A_\alpha\| = \text{val}\).

Since \(N_j^{B_{j-1}}(0^n_j)\) has no accepting computation paths, we claim that for each \(\alpha_1\) and \(\alpha_2\), \(\alpha_1 \neq \alpha_2\), in \(\Sigma^{n_j} - T\), it holds that \(A_{\alpha_1} \cap A_{\alpha_2} = \emptyset\). Why? Let \(\alpha_1, \alpha_2 \in \Sigma^{n_j} - T\) be distinct, and let \(\rho\) be any computation path in \(A_{\alpha_1}\). By definition, \(\rho\) is an accepting computation path of \(N_j^{B_{j-1} \cup \{\alpha_1\}}(0^n_j)\). Suppose \(\alpha_1 \notin Q^+(\rho)\). Note that in light of that supposition we either have that (a) \(\alpha_1 \in Q^-(\rho)\) and \(\alpha_1\) is queried by \(N_j^{B_{j-1} \cup \{\alpha_1\}}(0^n_j)\), or (b) \(\alpha_1 \notin Q^+(\rho) \cup Q^-(\rho)\) and \(\alpha_1\) is not queried by \(N_j^{B_{j-1} \cup \{\alpha_1\}}(0^n_j)\). In each of those two cases, however, it is clear that \(\rho\) will be an accepting computation path of \(N_j^{B_{j-1}}(0^n_j)\), which is a contradiction because by our assumption, \(N_j^{B_{j-1}}(0^n_j)\) has no accepting computation paths. Hence we have shown that \(\alpha_1 \in Q^+(\rho)\). But this implies that \(\rho\) is not a computation path of \(N_j^{B_{j-1} \cup \{\alpha_2\}}(0^n_j)\) (since \(\alpha_1 \in Q^+(\rho)\) yet \(\alpha_1 \notin B_{j-1} \cup \{\alpha_2\}\)), and therefore \(\rho \notin A_{\alpha_2}\).

Now we define, for each \(\alpha \in \Sigma^{n_j} - T\),

\[ \text{conflicting}(\alpha) = \{\beta \in \Sigma^{n_j} - T \mid \text{there are at least} \lfloor \text{val}/3 \rfloor + 1 \text{ computation paths in} A_\alpha \text{ that are not computation paths of} N_j^{B_{j-1} \cup \{\alpha, \beta\}}(0^n_j)\}. \]

We will show that for each \(\alpha \in \Sigma^{n_j} - T\),

\[ \|\text{conflicting}(\alpha)\| \leq 3p_j(n_j). \]

Fix some string \(\alpha \in \Sigma^{n_j} - T\). To get a contradiction, suppose that \(\|\text{conflicting}(\alpha)\| > 3p_j(n_j)\). Let \(\beta\) be some string in \(\text{conflicting}(\alpha)\).
Since adding $\beta$ to $B_{j-1} \cup \{\alpha\}$ causes at least $\lfloor val/3 \rfloor + 1$ computation paths in $A_\alpha$ to disappear from $N_{j}^{B_{j-1} \cup \{\alpha, \beta\}}(0^{n_j})$, this means that

$$\|\{\rho \in A_\alpha \mid \beta \in Q^-(\rho)\}\| \geq \lfloor val/3 \rfloor + 1.$$ 

Then

$$\sum_{\beta \in \text{conflicting}(\alpha)} \|\{\rho \in A_\alpha \mid \beta \in Q^-(\rho)\}\| > 3p_j(n_j)(\lfloor val/3 \rfloor + 1) \geq p_j(n_j) \cdot val,$$

and thus

$$\sum_{\beta \in \Sigma^{n_j} - T} \|\{\rho \in A_\alpha \mid \beta \in Q^-(\rho)\}\| > p_j(n_j) \cdot val.$$

Hence it follows that

\begin{equation}
(6.10) \quad \sum_{\rho \in A_\alpha} \|Q^-(\rho)\| > p_j(n_j) \cdot val.
\end{equation}

On the other hand, because each computation path $\rho$ in $A_\alpha$ is of length at most $p_j(n_j)$, for each $\rho \in A_\alpha$,

$$\|Q^-(\rho)\| \leq p_j(n_j).$$

Since $\|A_\alpha\| = val$, it follows that

$$\sum_{\rho \in A_\alpha} \|Q^-(\rho)\| \leq p_j(n_j) \cdot val,$$

which contradicts Eqn. (6.10). Hence we have shown that for each $\alpha \in \Sigma^{n_j} - T$,

\begin{equation}
(6.11) \quad \|\text{conflicting}(\alpha)\| \leq 3p_j(n_j).
\end{equation}

Because we have chosen $n_j$ large enough to ensure that $\|\Sigma^{n_j} - T\| > 6p_j(n_j) + 1$, Eqn. (6.11) implies that there exist distinct $\gamma_1, \gamma_2 \in \Sigma^{n_j} - T$ such that $\gamma_1 \notin \text{conflicting}(\gamma_2)$ and $\gamma_2 \notin \text{conflicting}(\gamma_1)$. Now consider

$$N_{j}^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^{n_j}).$$
Since $\gamma_2 \notin \text{conflicting}(\gamma_1)$, there are at most $\lfloor \text{val}/3 \rfloor$ computation paths in $A_{\gamma_1}$ that are not computation paths of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$. Further, because $\|A_{\gamma_1}\| = \text{val}$, it follows that there are at least $\text{val} - \lfloor \text{val}/3 \rfloor$ computation paths in $A_{\gamma_1}$ that are computation paths of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$.

The same reasoning applies when we swap the roles of $\gamma_1$ and $\gamma_2$: Since $\gamma_1 \notin \text{conflicting}(\gamma_2)$, there are at most $\lfloor \text{val}/3 \rfloor$ computation paths in $A_{\gamma_2}$ that are not computation paths of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$. Further, because $\|A_{\gamma_2}\| = \text{val}$, it follows that there are at least $\text{val} - \lfloor \text{val}/3 \rfloor$ computation paths in $A_{\gamma_2}$ that are computation paths of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$.

For the reasons given earlier in the proof, $A_{\gamma_1}$ and $A_{\gamma_2}$ are disjoint sets. Hence there are at least $2(\text{val} - \lfloor \text{val}/3 \rfloor)$ (distinct!) computation paths in $A_{\gamma_1} \cup A_{\gamma_2}$ that are computation paths of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$. Also, all computation paths in $A_{\gamma_1}$ and $A_{\gamma_2}$ are accepting. Note that for any two distinct computation paths $\rho_1, \rho_2 \in A_{\gamma_1} \cup A_{\gamma_2}$ of $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$, the corresponding nondeterministic choices $\rho_{1\text{choices}}$ and $\rho_{2\text{choices}}$ must be different. It follows that $N_j^{B_{j-1} \cup \{\gamma_1, \gamma_2\}}(0^n_j)$ has at least $2(\text{val} - \lfloor \text{val}/3 \rfloor) \geq 2\text{val} - \frac{2}{3}\text{val} = \frac{4}{3}\text{val}$ accepting computation paths. Since $\text{val} \in \mathbb{N}^+$, this contradicts Eqn. (6.9) if we set $C = \{\gamma_1, \gamma_2\}$. This completes the proof of Claim 6.8.

Since Claim 6.8 was all that remained in our proof of Theorem 6.6, that theorem itself is now proven. □

7. Multiple-target $C \equiv P$: A robustness result for $C \equiv P$

Section 3 showed robustness results for LWPP and WPP. The present section comes back to robustness, but now regarding the class $C \equiv P$ (see Definition 2.8). In particular, we will now show that $C \equiv P$ is robust in the sense that even if one changes the number of target values of acceptance-path cardinality from 1 to a polynomial, the class remains unchanged.
Definition 7.1. Let $r$ be any function mapping from $\mathbb{N}$ to $\mathbb{N}$. The class $r$-$C_{\geq} \mathbb{P}$ is the class of all sets $A$ such that there exists a non-deterministic polynomial-time Turing machine $N$ and a function $f \in \mathbb{F} \mathbb{P}$ that maps to $\mathbb{Z}$ such that for each $x \in \Sigma^*$,

\[(7.2) \quad x \in A \iff \text{there exists } i \in \{1, 2, \ldots, r(|x|)\} \text{ such that } \text{acc}_N(x) = f(\langle x, i \rangle).\]

Definition 7.3. $Poly$-$C_{\geq} \mathbb{P} = \bigcup_{c \in \mathbb{N}^+} (n^c + c)$-$C_{\geq} \mathbb{P}$.

Theorem 7.4. $Poly$-$C_{\geq} \mathbb{P} = C_{\geq} \mathbb{P}$.

We mention that Theorem 7.4 has already been applied in the study of a formal-languages notion. In particular, the $2$-$C_{\geq} \mathbb{P}$ special case of the theorem has been drawn on in the generalization of the notion of minimal words (see, e.g., Birget 1993; Berstel & Boasson 1997) to the notion of median words (Hemaspaandra, Královič, Královič & Ravikumar).

We have not yet proven Theorem 7.4. That is because we will instead prove a slightly stronger result (Theorem 7.8) that itself implies Theorem 7.4.

In particular, in Definition 7.1 and Definition 7.3, we generalized to the multiple-target case the standard definition of $C_{\geq} \mathbb{P}$, namely Definition 2.8. That definition is based on the number-of-accepting-paths function of a Turing machine.

However, recall that there is a potentially more flexible variant of $C_{\geq} \mathbb{P}$ that instead speaks of GapP functions. Fenner, Fortnow & Kurtz (1994) showed (in the result appearing as Theorem 2.9 in the present paper) that variant in fact itself yields precisely $C_{\geq} \mathbb{P}$.

\[\text{As to our “potentially more flexible variant” comment, one might worry that the $C_{\geq} \mathbb{P}$ variant used in Theorem 2.9, though potentially reaping increased flexibility due to the change from a $\#P$ function to a GapP function, might potentially suffer from decreased flexibility due to the target changing from being set by an arbitrary FP function to being fixed to be 0. However, since GapP is closed under subtraction of arbitrary FP—or even GapP—} \]
It is natural to similarly wonder whether, if we build a polynomially rich multi-target variant in terms of GapP functions, that will yield the same class as $\text{Poly-C}_{\leq} \text{P}$ (and so in particular itself will equal $C_{\leq} \text{P}$) or will yield a richer class (the yielded class certainly is at least as large). We show that the answer to that question is that the classes are the same, and hand-in-hand we implicitly also prove (since every $#P$ function belongs to GapP) Theorem 7.4. We do so below defining the GapP-characterization analogues of multi-target $C_{\leq} \text{P}$, denoted by adding a “prime” to the class names, and then proving that $\text{Poly-C}_{\leq} \text{P}' = C_{\leq} \text{P}$.

**Definition 7.5.** Let $r$ be any function mapping from $\mathbb{N}$ to $\mathbb{N}$. The class $r-C_{\leq} \text{P}'$ is the class of all sets $A$ such that there exists a nondeterministic polynomial-time Turing machine $N$ and a function $f \in \text{FP}$ that maps to $\mathbb{Z}$ such that for each $x \in \Sigma^*$,

\[ x \in A \iff \text{there exists } i \in \{1, 2, \ldots, r(|x|)\} \text{ such that } \text{gap}_N(x) = f(\langle x, i \rangle). \]

**Definition 7.7.** $\text{Poly-C}_{\leq} \text{P}' = \bigcup_{c \in \mathbb{N}^+} (n^c + c) - C_{\leq} \text{P}'$.

**Theorem 7.8.** $\text{Poly-C}_{\leq} \text{P}' = C_{\leq} \text{P}$.

**Proof.** It is easy to see that $C_{\leq} \text{P} \subseteq \text{Poly-C}_{\leq} \text{P}'$.

To show $\text{Poly-C}_{\leq} \text{P}' \subseteq C_{\leq} \text{P}$, let $A$ be a set in $\text{Poly-C}_{\leq} \text{P}'$ defined by nondeterministic polynomial-time Turing machine $N$, $f \in \text{FP}$, and polynomial $r(n) = n^c + c$ according to Definition 7.5 and Definition 7.7.

Let $h_1$ be a function such that for all $x \in \Sigma^*$ and $i \in \mathbb{N}^+$,

\[ h_1(\langle x, i \rangle) = f(\langle x, i \rangle) - \text{gap}_N(x). \]

functions, clearly Theorem 2.9 would remain true even if one changed its “$g(x) = 0$” target statement to instead require that $g(x)$ equals an FP function $f(x)$—thus making its only change relative to Definition 2.8 the change from a $#P$ function to a GapP function—or even to require that $g(x)$ equals a GapP function $f(x)$.
We have $h_1 \in \text{GapP}$ since $f \in \text{FP} \subseteq \text{GapP}$, $\text{gap}_N \in \text{GapP}$, and \text{GapP} is closed under subtraction (Fenner et al. 1994). We define $h_2$ such that for all $x \in \Sigma^*$,

$$h_2(x) = \prod_{1 \leq i \leq r(|x|)} h_1(\langle x, i \rangle).$$

By Closure Property 3.4, $h_2 \in \text{GapP}$. Note that for all $x \in \Sigma^*$,

$$h_2(x) = \prod_{1 \leq i \leq r(|x|)} (f(\langle x, i \rangle) - \text{gap}_N(x)).$$

From the above equality, for each $x \in \Sigma^*$ we have that $h_2(x) = 0$ if and only if $(\exists i \in \{1, 2, \ldots, r(|x|)\}[\text{gap}_N(x) = f(\langle x, i \rangle)])$. But the $\text{Poly}-C_{\equiv \text{P}}'$ structures defining $A$ specify that for each $x \in \Sigma^*$ we have $x \in A$ if and only if $(\exists i \in \{1, 2, \ldots, r(|x|)\}[\text{gap}_N(x) = f(\langle x, i \rangle)])$. So $x \in A$ if and only if the GapP function $h_2(x)$ equals 0, and so by Theorem 2.9 it follows that $A \in C_{\equiv \text{P}}$. □

8. Conclusions and open questions

In this paper, we proved that LWPP and WPP are robust enough that they remain unchanged when their single target gap is allowed to be expanded to polynomial-sized lists. We then applied this new robustness of LWPP to show that the PP-lowness of the Legitimate Deck Problem follows from a weaker hypothesis than was previously known. In doing so, we provided enhanced evidence that the Legitimate Deck Problem is not NP-hard or even NP-Turing-hard. We also showed: that the polynomial-target robustness of LWPP that we have established is optimal (i.e., cannot be extended to any superpolynomial number of targets) with respect to relativizable proofs; that for the $\#\text{P}$-based analogue of the (GapP-based) class LWPP, in some relativized worlds even two targets give more languages than one target; that our robustness of LWPP holds even when one simultaneously expands both the acceptance target-gap set and the rejection target-gap set to be polynomial-sized lists; that our main results also hold for WPP; and that $C_{\equiv \text{P}}$ is polynomial-target robust.
Regarding the Reconstruction Conjecture, we proved (in Corollary 4.12 and Corollary 4.15) as a consequence of our results that if there exists a polynomial $q$ such that the $q$-Reconstruction Conjecture holds, then the Legitimate Deck Problem is both PP-low and $C=P$-low. Since NP is widely believed not to be PP-low or $C=P$-low, this provides strengthened evidence that the Legitimate Deck Problem is not NP-hard or even NP-Turing hard (since otherwise even the “Poly”-Reconstruction Conjecture must fail, yet even the far stronger Reconstruction Conjecture is generally believed to hold).

A natural open problem is whether the Legitimate Deck Problem is $\Sigma_k^p$-low (Schöning 1983) for some $k$, i.e., whether for some $k$ it holds that $(\Sigma_k^p)^{\text{Legitimate Deck}} = \Sigma_k^p$. Proving that that holds—though we mention that this has been a known open issue for more than two decades (see Kratsch & Hemaspaandra 1994)—would imply that the Legitimate Deck Problem cannot be NP-complete (even with respect to more flexible reductions such as Turing reductions or the strong nondeterministic reductions of Long 1982) unless the polynomial hierarchy collapses.

**Acknowledgements**

A preliminary version of this paper appeared in MFCS 2018 (Hemaspaandra *et al.* 2018). We are very grateful to the anonymous conference and journal referees for many helpful corrections and comments, including suggesting that we strengthen our Theorem 7.4 into what now is Theorem 7.8.

This work was done in part while Edith and Lane A. Hemaspaandra were visiting the Tokyo Institute of Technology’s Department of Mathematical and Computing Sciences, ETH Zürich’s Department of Computer Science, and the University of Düsseldorf’s Department of Computer Science, and in part while Holger Spakowski was visiting the University of Rochester’s Department of Computer Science. We thank those departments for generously hosting these visits.

Edith Hemaspaandra was supported in part by a Renewed Research Stay award from the Alexander von Humboldt Foundation and sabbatical visit support from ETH Zürich’s Department of
Computer Science. Lane A. Hemaspaandra was supported in part by NSF grant CCF-2006496, a Renewed Research Stay award from the Alexander von Humboldt Foundation, and sabbatical visit support from ETH Zürich’s Department of Computer Science.

Publisher’s Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

L. Babai (2016). Graph Isomorphism in Quasipolynomial Time. In Proceedings of the 48th ACM Symposium on Theory of Computing, 684–697. ACM Press.

R. Beigel (1989). On the Relativized Power of Additional Accepting Paths. In Proceedings of the 4th Structure in Complexity Theory Conference, 216–224. IEEE Computer Society Press.

R. Beigel (1997). Closure Properties of GapP and #P. In Proceedings of the 5th Israeli Symposium on Theory of Computing and Systems, 144–146. IEEE Computer Society Press.

J. Berstel & L. Boasson (1997). The Set of Minimal Words of a Context-Free Language is Context-Free. Journal of Computer and System Sciences 55(3), 477–488.

A. Berthiaume & G. Brassard (1992). The Quantum Challenge to Structural Complexity Theory. In Proceedings of the 7th Structure in Complexity Theory Conference, 132–137. IEEE Computer Society Press.

A. Berthiaume & G. Brassard (1994). Oracle Quantum Computing. Journal of Modern Optics 41(12), 2521–2535.

J.-C. Birget (1993). Partial Orders on Words, Minimal Elements of Regular Languages and State Complexity. Theoretical Computer Science 119(2), 267–291.

J. Bondy (1991). A Graph Reconstructor’s Manual. In Surveys in Combinatorics, London Mathematical Society Lecture Notes Series 66, 221–252. Cambridge University Press.
J. Bondy & R. Hemminger (1977). Graph Reconstruction—A Survey. *Journal of Graph Theory* 1(3), 227–268.

D. Bovet, P. Crescenzi & R. Silvestri (1992). A Uniform Approach to Define Complexity Classes. *Theoretical Computer Science* 104(2), 263–283.

J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner & G. Wechsung (1989). The Boolean Hierarchy II: Applications. *SIAM Journal on Computing* 18(1), 95–111.

J. Cox & T. Pay (2018). An Overview of Some Semantic and Syntactic Complexity Classes. Technical Report arXiv:1806.03501 [cs.CC], Computing Research Repository, arXiv.org/corr/.

The Editors (of the Journal of Graph Theory) (1977). Editorial Note. *Journal of Graph Theory* 1(3), 191.

S. Fenner (2003). PP-Lowness and a Simple Definition of AWPP. *ACM Transactions on Computer Systems* 36(2), 199–212.

S. Fenner, L. Fortnow & S. Kurtz (1994). Gap-Definable Counting Classes. *Journal of Computer and System Sciences* 48(1), 116–148.

S. Fenner, L. Fortnow, S. Kurtz & L. Li (2003). An Oracle Builder’s Toolkit. *Information and Computation* 182(2), 95–136.

Fortnow & Rogers (1999). Complexity Limitations on Quantum Computation. *Journal of Computer and System Sciences* 59(2), 240–252.

L. Fortnow (1994). The Role of Relativization in Complexity Theory. *Bulletin of the European Association for Theoretical Computer Science* 52, 229–244.

M. Furst, J. Saxe & M. Sipser (1984). Parity, Circuits, and the Polynomial-Time Hierarchy. *Mathematical Systems Theory* 17, 13–27.

M. de Graaf & P. Valiant (2002). Comparing EQP and MOD_p^k using polynomial degree lower bounds. Technical Report arXiv:quant-ph/0211179, arXiv.org.

J. Grollmann & A. Selman (1988). Complexity Measures for Public-Key Cryptosystems. *SIAM Journal on Computing* 17(2), 309–335.
Y. Han, L. Hemaspaandra & T. Thierauf (1997). Threshold Computation and Cryptographic Security. *SIAM Journal on Computing* **26**(1), 59–78.

J. Hartmanis, R. Chang, S. Chari, D. Ranjan & P. Rohatgi (1992). Relativization: A Revisionistic Retrospective. *Bulletin of the European Association for Theoretical Computer Science* **47**, 144–153.

J. Hartmanis, R. Chang, D. Ranjan & P. Rohatgi (1990). Structural Complexity Theory: Recent Surprises. In *Proceedings of the 2nd Scandinavian Workshop on Algorithm Theory*, 1–12. Springer-Verlag Lecture Notes in Computer Science #447.

J. Håstad (1987). *Computational Limitations of Small-Depth Circuits*. MIT Press.

E. Hemaspaandra, L. Hemaspaandra, S. Radziszowski & R. Tripathi (2007). Complexity Results in Graph Reconstruction. *Discrete Applied Mathematics* **155**(2), 103–118.

E. Hemaspaandra, L. Hemaspaandra, H. Spakowski & O. Watanabe (2018). The Robustness of LWPP and WPP, with an Application to Graph Reconstruction. In *Proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science*, volume 117, 51:1–51:14. Leibniz International Proceedings in Informatics (LIPIcs).

L. Hemaspaandra, R. Královič, R. Královič & B. Ravikumar. The Complexity of Culling: Decimation, Minimal Words, and Many-One Reductions. In preparation.

L. Hemaspaandra & M. Ogihara (2002). *The Complexity Theory Companion*. Springer.

L. Hemaspaandra & M. Zimand (1993). Strong Forms of Balanced Immunity. Technical Report TR-480, Department of Computer Science, University of Rochester, Rochester, NY. Revised, May 1994.

P. Kelly (1942). *On Isometric Transformations*. Ph.D. thesis, University of Wisconsin, USA.

K. Ko (1985). On Some Natural Complete Operators. *Theoretical Computer Science* **37**(1), 1–30.
K. Ko (1989). Relativized Polynomial-Time Hierarchies Having Exactly $k$ Levels. *SIAM Journal on Computing* 18(2), 392–408.

J. Kölner, U. Schöning & J. Torán (1992). Graph Isomorphism is low for PP. *Computational Complexity* 2, 301–330.

D. Kratsch & L. Hemachandra (1991). On the Complexity of Graph Reconstruction. In *Proceedings of the 8th Conference on Fundamentals of Computation Theory*, 318–328. Springer-Verlag *Lecture Notes in Computer Science* #529.

D. Kratsch & L. Hemaspaandra (1994). On the Complexity of Graph Reconstruction. *Mathematical Systems Theory* 27(3), 257–273.

J. Lauri & R. Scapellato (2003). *Topics in Graph Automorphisms and Reconstruction*. Cambridge University Press.

T. Long (1982). Strong Nondeterministic Polynomial-Time Reducibilities. *Theoretical Computer Science* 21, 1–25.

A. Mansfield (1982). The Relationship between the Computational Complexities of the Legitimate Deck and Isomorphism Problems. *Quart. J. Math. Ser.* 33(2), 345–347.

B. Manvel (1988). Reconstruction of Graphs: Progress and Prospects. *Congressus Numerantium* 63, 177–187.

C. St. J. A. Nash-Williams (1978). The Reconstruction Problem. In *Selected Topics in Graph Theory*, L. Beineke & R. Wilson, editors, 205–236. Academic Press.

M. Ogiwara & L. Hemachandra (1993). A Complexity Theory for Feasible Closure Properties. *Journal of Computer and System Sciences* 46(3), 295–325.

R. Rao, J. Rothe & O. Watanabe (1994). Upward Separation for FewP and Related Classes. *Information Processing Letters* 52(4), 175–180. Corrigendum in *Information Processing Letters*, 74(1–2), 89.

J. Rosser & L. Schoenfeld (1962). Approximate Formulas for Some Functions of Prime Numbers. *Illinois Journal of Mathematics* 6, 64–94.
J. Rothe (1999). Immunity and Simplicity for Exact Counting and Other Counting Classes. *RAIRO Theoretical Informatics and Applications* 33(2), 159–176.

U. Schöning (1983). A Low and a High Hierarchy within NP. *Journal of Computer and System Sciences* 27, 14–28.

J. Simon (1975). *On Some Central Problems in Computational Complexity*. Ph.D. thesis, Cornell University, Ithaca, N.Y. Available as Cornell Department of Computer Science Technical Report TR75-224.

H. Spakowski, M. Thakur & R. Tripathi (2005). Quantum and Classical Complexity Classes: Separations, Collapses, and Closure Properties. *Information and Computation* 200(1), 1–34.

H. Spakowski & R. Tripathi (2006). LWPP and WPP Are Not Uniformly Gap-Definable. *Journal of Computer and System Sciences* 72(4), 660–689.

L. Stockmeyer & A. Meyer (1973). Word Problems Requiring Exponential Time. In *Proceedings of the 5th ACM Symposium on Theory of Computing*, 1–9. ACM Press.

J. Tarui (1991). Degree Complexity of Boolean Functions and Its Applications to Relativized Separations. In *Proceedings of the 6th Annual Conference on Structure in Complexity Theory*, 382–390. IEEE Computer Society Press.

S. Toda & M. Ogiwara (1992). Counting Classes are at Least as Hard as the Polynomial-Time Hierarchy. *SIAM Journal on Computing* 21(2), 316–328.

S. Ulam (1960). *A Collection of Mathematical Problems*. Interscience Publishers, New York.

L. Valiant (1976). The Relative Complexity of Checking and Evaluating. *Information Processing Letters* 5(1), 20–23.

L. Valiant (1979). The Complexity of Computing the Permanent. *Theoretical Computer Science* 8(2), 189–201.

N. Vinodchandran (2004). Counting Complexity of Solvable Black-Box Group Problems. *SIAM Journal on Computing* 33(4), 852–869.
K. Wagner (1986). The Complexity of Combinatorial Problems with Succinct Input Representations. *Acta Informatica* 23(3), 325–356.

O. Watanabe (1988). On Hardness of One-Way Functions. *Information Processing Letters* 27(3), 151–157.

C. Wrathall (1977). Complete Sets and the Polynomial-time Hierarchy. *Theoretical Computer Science* 3, 23–33.

A. Yao (1985). Separating the Polynomial-Time Hierarchy by Oracles. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, 1–10.

Manuscript received 11 February 2019

**Edith Hemaspaandra**  
Department of Computer Science  
Rochester Institute of Technology  
Rochester, NY 14623, USA  
eh@cs.rit.edu

**Lane A. Hemaspaandra**  
Department of Computer Science  
University of Rochester  
Rochester, NY 14627, USA  
https://www.cs.rochester.edu/u/lane

**Holger Spakowski**  
Department of Mathematics and Applied Mathematics  
University of Cape Town  
Rondebosch 7701, South Africa  
Holger.Spakowski@uct.ac.za

**Osamu Watanabe**  
Department of Mathematical and Computing Sciences  
Tokyo Institute of Technology  
Tokyo 152-8552, Japan  
watanabe@c.titech.ac.jp