

# Universal Languages and the Power of Diagonalization

Alan Nash

Department of Mathematics  
anash@math.ucsd.edu

Russell Impagliazzo\*

Department of Computer Science and Engineering  
russell@cs.ucsd.edu

Jeff Remmel

Department of Mathematics  
remmel@math.ucsd.edu

University of California, San Diego, CA 92093

## Abstract

*We define and study strong diagonalization and compare it to weak diagonalization, implicit in [7]. Kozen's result in [7] shows that virtually every separation can be recast as weak diagonalization. We show that there are classes of languages which can not be separated by strong diagonalization and provide evidence that strong diagonalization does not relativize. We also define two kinds of indirect diagonalization and study their power.*

*Since we define strong diagonalization in terms of universal languages, we study their complexity. We distinguish and compare weak and strict universal languages. Finally we analyze some apparently weaker variants of universal languages, which we call pseudouniversal languages, and show that under weak closure conditions they easily yield universal languages.*

## 1. Introduction

Baker, Gill, and Solovay [1] showed that there are oracles  $A, B$  so that  $P^A \neq NP^A$  and  $P^B = NP^B$ . While this shows that *relativizing techniques* won't settle  $P \stackrel{?}{=} NP$ , some have interpreted this result to show that *diagonalization* won't settle  $P \stackrel{?}{=} NP$ . Indeed, the diagonalization techniques that we are familiar with seem to relativize. Perhaps as a reaction to such overly broad interpretation of the results in [1], Kozen proved a result (theorem 6.2 in [7], adapted below as theorem 7) which he interpreted as follows: "if  $P \neq NP$  is provable at all, then it is provable by diagonalization" and "there is an effective method for going from proofs of  $P \neq NP$  to diagonalization proofs of

$P \neq NP$  (i.e., those in which a specific diagonal function in  $NP - P$  is exhibited)."

One difficulty with this interpretation is that there is no generally-accepted definition of "separation by diagonalization." Regarding Kozen's result, Fortnow has written [6] "I believe this result says more about the difficulty of exactly formalizing the notion of a 'diagonalization proof' than of actually arguing the diagonalization technique is the only technique we have for class separation."

Some obvious questions are:

1. What is "separation by diagonalization"?
2. Are there separations that it can't do?
3. Does it always relativize?
4. Can it prove  $P \neq NP$ ?

In this paper we first highlight the difference between "separation by diagonalization" and "nonmembership by diagonalization" and formalize two notions of "separation by direct diagonalization": weak and strong. In [7], Kozen showed that separation by weak diagonalization is essentially equivalent to constructive separation (he didn't give a definition of "separation by diagonalization"). Therefore, if  $P$  can be separated from  $NP$  constructively, then it can be separated by weak diagonalization.

We define "separation by strong diagonalization" roughly as follows: a proof of separation by strong diagonalization of  $\mathcal{B} \neq \mathcal{A}$  is one that shows that  $\mathcal{A}$  has a weak universal language for  $\mathcal{B}$  (we define "universal language" and "weak universal language" in the next section). Obvious examples of separation by strong diagonalization are the time and space hierarchy theorems. We show that there are pairs of classes of languages that can not be separated by strong diagonalization, including  $P$  and a strict superset of

---

\*Research supported by NSF Award CCR-0098197

P. Thus, even if  $P \neq NP$ , it is possible that P can't be separated from NP by strong diagonalization. Since we will see that P and EXP can be separated by strong diagonalization, providing a definitive answer to whether P and NP can be separated by strong diagonalization is hard: a positive answer would give  $P \neq NP$  and a negative answer would give  $NP \neq EXP$ . We show that there are oracles  $A$  and  $B$  such that  $P^A$  and  $NP^A$  can be separated by strong diagonalization and such that  $P^B = NP^B$ , and we provide additional evidence that strong diagonalization doesn't relativize.

We are primarily interested in the situation where two classes of languages  $\mathcal{A}$  and  $\mathcal{B}$  are given to us to separate, rather than the situation where one or both of  $\mathcal{A}$  and  $\mathcal{B}$  are constructed during the "diagonalization" process. That is, we are interested in diagonalization used to separate, rather than to construct. When  $\mathcal{A}$  is closed under linear-time many-one reducibility, strong diagonalization shows  $\mathcal{B} \subsetneq \mathcal{A}$  rather than just  $\mathcal{B} \neq \mathcal{A}$ . Therefore we are primarily interested in the situation where  $\mathcal{B} \subseteq \mathcal{A}$ . The time and space hierarchy theorems as well as the open problem of whether  $P \neq NP$  satisfy these restrictions. We primarily consider the situation where diagonalization is the main technique for separation, rather than just one piece of a more complex argument. This is what we mean by "separation by *direct* diagonalization." We briefly discuss various forms of indirect use of diagonalization arguments.

Since universal languages play a key role in our investigations, we study their complexity. For any class of languages  $P^B$ , Kozen's result (theorem 7 below) implies that there are universal languages of  $P^B$  of arbitrarily high complexity. We show that there are no universal languages of  $P^B$  of arbitrarily low complexity (greater than or equal to that of  $B$ ), show a density result for universal languages, and show that there can be arbitrarily large gaps in complexity between weak universal languages and strict universal languages of the same fixed class of languages.

While it seems hopeless to define "separation by indirect diagonalization" in general, we isolate two related kinds of "separation by indirect diagonalization" where diagonalization is only part of the proof that  $\mathcal{A} \neq \mathcal{B}$ . We show that they are strictly stronger than "separation by strong diagonalization."

To simplify the presentation of the results, we focus on many-one reducibilities  $\leq_m^{\text{lin}}$  (linear time) and  $\leq_m^{\text{P}}$  (polynomial time) and their corresponding Turing reducibilities  $\leq_T^{\text{lin}}$  (linear time) and  $\leq_T^{\text{P}}$  (polynomial time). Clearly most of our results can be easily adapted to a variety of different reducibilities.

The outline of this paper is as follows. In section 2, we review diagonalization, show how it can be used to establish nonmembership, introduce universal languages and most of the notation we will need, and present some basic properties, including Kozen's result. In section 3, we introduce

our definitions of separation by strong and weak diagonalization, show how the space and time hierarchy theorems fit the definition of separation by strong diagonalization, compare the two definitions (showing they are not equal), and discuss their consequences, referring to structural results from the next section. In section 4, we present our main results: structural results on universal languages for a fixed class of languages. In section 5, we discuss pseudouniversal languages and show that, in general, it is easy to obtain universal languages from them. In section 6, we present relativization results for separation by strong diagonalization.

## 2. Diagonalization, non-membership, and universal languages

We start by reviewing what diagonalization is and how it is used to prove non-membership. We will see later that there is an extra twist when it is used to prove separations and that it is this extra twist that leads to the possible range of definitions of "separation by diagonalization." Diagonalization was introduced by Georg Cantor [4] in his famous proof that the set of real numbers is uncountable.

The basic idea is that given

- a (usually infinite) matrix  $[a_{ij}]$  and
- a function  $g: \mathbb{N} \rightarrow \mathbb{N}$  with no fixed points,

the  $g$ -diagonal  $d_j := g(a_{g(j)j})$  is different from every row in  $[a_{ij}]$ . If the matrix is 0,1-valued (the case we are interested in), we must have  $g(x) = \eta(x)$  where  $\eta$  is the negation function given by  $\eta(x) := 1 - x$ .

We may use something other than  $\text{diag}_U$ . That is, given

- an infinite matrix  $[a_{ij}]$ ,
- a function  $g: \mathbb{N} \rightarrow \mathbb{N}$  with no fixed points, and
- a surjective function  $h: \mathbb{N} \rightarrow \mathbb{N}$ ,

the  $g$ - $h$ -diagonal  $d_j := g(a_{h(j)j})$  is different from every row in  $[a_{ij}]$ .

This is the format in which many diagonalization results are presented. We start with some enumeration of the languages in the set  $\mathcal{C}$  and then we build the function  $h$ . Very sophisticated techniques have been developed in computability and complexity theory to define such a function  $h$ . However, here we will be concerned only with the resulting complexity of such a function  $h$ , not with the techniques used to define it.

Throughout this paper, we identify strings with natural numbers, for example, by the correspondence

$$\hat{\sigma} := 2^{|\sigma|} + \sum_{i < |\sigma|} \sigma_i \cdot 2^i - 1.$$

That is, we append 1 to the string, read off the resulting string as a binary number and subtract 1. We identify languages with characteristic functions of subsets of  $\mathbb{N}$ . We define a linear-time computable pairing function  $\langle \cdot, \cdot \rangle$  and the corresponding projection functions  $\pi_1, \pi_2$  satisfying  $\pi_i(\langle x_1, x_2 \rangle) = x_i$  as follows: to get  $\langle a, b \rangle$  we insert zeros between bits of  $a$  and ones between bits of  $b$ . That is,  $\langle a, b \rangle$  is the string  $a_0 0 a_1 0 a_2 0 \dots b_0 1 b_1 1 b_2 1 \dots$  where  $a_0, a_1, a_2, \dots$  are the bits of  $a$  and  $b_0, b_1, b_2, \dots$  are the bits of  $b$ .

In the case where the matrix  $[a_{ij}]$  is 0,1-valued, our identification of strings with natural numbers and languages with characteristic functions allows us to view the  $i$ -th row of  $[a_{ij}]$  as a language  $U_i$ . If the set  $\{U_i : i \in \mathbb{N}\}$  is a class of languages  $\mathcal{C}$ , then we can view the matrix  $[a_{ij}]$  as a universal language for  $\mathcal{C}$ . We say that  $U$  is a *universal language* for a class of languages  $\mathcal{C}$ , which we denote by  $U \dashv\vdash \mathcal{C}$ , if and only if

$$\mathcal{C} = \{U(\langle e, \bullet \rangle) : e \in \mathbb{N}\}.$$

That is,

$$L \in \mathcal{C} \iff \exists e \forall x (L(x) = U(\langle e, x \rangle)).$$

We say that that  $U$  is a *weak universal language* for  $\mathcal{C}$ , which we denote by  $U \dashv\vdash \mathcal{C}$  if and only if

$$\mathcal{C} \subseteq \{U(\langle e, \bullet \rangle) : e \in \mathbb{N}\}.$$

That is,

$$L \in \mathcal{C} \implies \exists e \forall x (L(x) = U(\langle e, x \rangle)).$$

Sometimes we will refer to universal languages as *strict* universal languages, when we need to emphasize that they are not weak.

Thus if  $U \dashv\vdash \mathcal{C}$ , then every language in  $\mathcal{C}$  corresponds to a row in the infinite 0,1-valued matrix  $[u_{ex}]$  where  $u_{ex} = U(\langle e, x \rangle)$ . If  $U \dashv\vdash \mathcal{C}$ , then the matrix  $[u_{ex}]$  may have rows which do not correspond to languages in  $\mathcal{C}$ . We will write  $U_e$  for the language  $U(\langle e, \bullet \rangle)$ , that is,  $U_e(x) = U(\langle e, x \rangle)$ . In this notation, we have

$$U \dashv\vdash \mathcal{C} \iff \mathcal{C} = \{U_e : e \in \mathbb{N}\}$$

and

$$U \dashv\vdash \mathcal{C} \iff \mathcal{C} \subseteq \{U_e : e \in \mathbb{N}\}.$$

The  $\eta$ -diagonal of  $U$ , which we will denote  $\text{diag}_U$ , is given by  $\text{diag}_U(x) := 1 - U_x(x)$ .

**Proposition 1.** *If  $U \dashv\vdash \mathcal{B}$ , then  $\text{diag}_U \notin \mathcal{B}$ .*

*Proof.*  $\text{diag}_U \in \mathcal{B}$  implies  $\text{diag}_U = U_e$  for some  $e$ , which gives the contradiction  $U_e(e) = \text{diag}_U(e) = 1 - U_e(e)$ .  $\square$

Notice that the  $g$ - $h$ -diagonal of  $U$  is equal to the  $g$ -diagonal of  $V$  where  $V$  is given by  $V_e(x) = U_{h(e)}(x)$  since  $g(U_{h(e)}(e)) = g(V_e(e))$ . If  $h$  is surjective, we have  $V \dashv\vdash \mathcal{C}$ . Since we will be concerned with the complexity of the languages  $V$  and  $\text{diag}_V$ , we will not need to consider  $U$  and  $h$  separately, only their composition  $V$ . As a result, we can limit ourselves to studying universal languages  $U$  and their ‘‘straight’’ diagonals  $\text{diag}_U$ .

A frequently used universal language for  $P^Q$  is given by  $U_e(x) = \psi_{e;i}^Q(x)$  where  $\psi_{e;i}^Q$  denotes the  $e$ th oracle Turing machine with time bound  $|x|^i + i$  on input  $x$ . This machine returns 0 or 1 (i.e., rejects or accepts) and, in particular, returns 0 if the time bound is reached. Similarly, a frequently used universal language for  $PSPACE^Q$  is given by  $U_e(x) = \psi_{e;sp:e}^Q(x)$  where  $\psi_{e;sp:e}^Q$  denotes the  $e$ th oracle Turing machine with space bound  $|x|^i + i$  on input  $x$ . This machine returns 0 or 1 (i.e., rejects or accepts) and, in particular, returns 0 if the space bound is reached or if a loop is detected (which can be done effectively).

We use the following notation for universal languages and corresponding notation for weak universal languages.

- $A \dashv\vdash^U \mathcal{B}$  for  $U \dashv\vdash \mathcal{B}$  and  $U \in \mathcal{A}$ .
- $A \dashv\vdash \mathcal{B}$  for  $\exists U (A \dashv\vdash^U \mathcal{B})$ .
- $A \not\dashv\vdash \mathcal{B}$  for  $\neg \exists U (A \dashv\vdash^U \mathcal{B})$ .

**Proposition 2.** *If  $U \dashv\vdash \mathcal{B}$  and  $L \in \mathcal{B}$ , then  $L \leq_m^{\text{lin}} U$ .*

*Proof.* We must have  $L = U_e$  for some  $e$  and therefore  $f$  given by  $f(x) := \langle e, x \rangle$  witnesses  $L \leq_m^{\text{lin}} U$ .  $\square$

**Corollary 3.** *If  $\mathcal{A}$  is closed under  $\leq_m^{\text{lin}}$  and  $\mathcal{A} \dashv\vdash \mathcal{B}$ , then  $\mathcal{B} \subseteq \mathcal{A}$ .*

Given a countable class of languages  $\mathcal{C}$ , there is always some universal language  $U$  for it. Therefore we are not interested in the existence of universal languages for  $\mathcal{C}$ , but rather in their complexity. We have the following two easy facts.

**Proposition 4.**  $\text{diag}_U \leq_T^{\text{lin}} U$ .

*Proof.* By definition,  $\text{diag}_U(x) = 1 - U_x(x) = 1 - U(\langle x, x \rangle)$  which can be computed in linear time with a single query to the oracle  $U$ .  $\square$

**Proposition 5.** *If  $\mathcal{B}$  is closed under  $\leq_T^{\text{lin}}$ , then  $\mathcal{B} \dashv\vdash \mathcal{B}$ .*

*Proof.* If we had  $\mathcal{B} \dashv\vdash^U \mathcal{B}$ , then by proposition 4 and the fact that  $\mathcal{B}$  is closed under  $\leq_T^{\text{lin}}$  we would have  $\text{diag}_U \in \mathcal{B}$ , contradicting proposition 1.  $\square$

We now present two results, the second one due to Kozen, which we will need in the next section to analyze separations by strong and weak diagonalization.

Let  $E$  be the empty language (i.e.,  $\forall n(E(n) = 0)$ ) and  $F$  be its complement (i.e.,  $\forall n(F(n) = 1)$ ). Notice that  $E \dashv\circ \{E\}$  and  $F \dashv\circ \{F\}$  and that  $\{E\}$  and  $\{F\}$  are closed under  $\leq_m^{\text{lin}}$ .

**Theorem 6.** *If  $\mathcal{B}$  is a set of computable languages closed under  $\leq_m^{\text{lin}}$  and  $\mathcal{B} \neq \{E\}$  and  $\mathcal{B} \neq \{F\}$ , then  $\mathcal{B} \not\text{-}\dashv\circ \mathcal{B}$ .*

This shows that  $\mathcal{B} \not\text{-}\dashv\circ \mathcal{B}$  even when  $\mathcal{B}$  is not closed under complements. The proof technique comes from [9] and is interesting because it shows that there are separations such as the nondeterministic time hierarchy theorems in [9] that seem to be considered to be “by diagonalization” even though no diagonal  $\text{diag}_U$  is explicit or even implicit in the proof.

*Proof.* Assume to get a contradiction that  $\mathcal{B} \dashv\circ \mathcal{B}$ , so  $U$  is computable by some Turing machine  $M$ . Define  $g$  by

$$g(\langle e, x \rangle) = \begin{cases} u_1 & \text{if } M(\langle e, 0 \rangle) = 0 \text{ in } |x| \\ & \text{steps or less} \\ u_0 & \text{if } M(\langle e, 0 \rangle) = 1 \text{ in } |x| \\ & \text{steps or less} \\ \langle e, x' \rangle & \text{otherwise} \end{cases}$$

where  $x'$  is  $x$  extended by appending a zero (so  $|x'| = |x| + 1$ ) and where  $u_0$  and  $u_1$  are two constants so that  $U(u_0) = 0$  and  $U(u_1) = 1$  ( $u_0$  and  $u_1$  exist because  $\mathcal{B} \neq \{E\}, \{F\}$ ). Notice that  $g(z)$  can be computed in linear time since our pairing function  $\langle \cdot, \cdot \rangle$  and the corresponding projections (described in an earlier section) are computable in linear time.

Now set  $L = U \circ g$  so  $L \leq_m^{\text{lin}} U$ . Since  $\mathcal{B}$  is closed under  $\leq_m^{\text{lin}}$  and  $U \in \mathcal{B}$ , we must have  $L \in \mathcal{B}$  and

$$L(\langle e, x \rangle) = \begin{cases} 1 - U_e(\langle e, 0 \rangle) & \text{if } M(\langle e, 0 \rangle) = y \text{ in} \\ & |x| \text{ steps or less} \\ U_e(\langle e, x' \rangle) & \text{otherwise.} \end{cases}$$

Since  $L \in \mathcal{B}$ , we must have  $L = U_e$  for some  $e$  and therefore

$$L(\langle e, x \rangle) = \begin{cases} 1 - L(\langle e, 0 \rangle) & \text{if } M(\langle e, 0 \rangle) = y \text{ in} \\ & |x| \text{ steps or less} \\ L(\langle e, x' \rangle) & \text{otherwise.} \end{cases}$$

We define  $x^{(s)}$  to be  $x$  extended by appending  $s$  zeros. By our correspondence of strings and numbers,  $0$  denotes the empty string and so  $0^{(s)}$  denotes a string of  $s$  zeros.  $M(\langle e, 0 \rangle)$  must halt, so say it halts in  $s$  steps. Then we have the following contradiction

$$L(\langle e, 0 \rangle) = L(\langle e, 0' \rangle) = \dots = L(\langle e, 0^{(s)} \rangle) = 1 - L(\langle e, 0 \rangle),$$

which shows that  $\mathcal{B} \not\text{-}\dashv\circ \mathcal{B}$ .  $\square$

A set  $\mathcal{B}$  of languages is closed under finite variations iff  $(\forall B \in \mathcal{B})(A =^* B \in \mathcal{B} \implies A \in \mathcal{B})$  where  $A =^* B$  means that  $A$  and  $B$  differ on at most finitely many positions.

**Theorem 7 (Kozen).** *If  $\mathcal{B}$  is closed under finite variations and there is  $V$  so that  $V \dashv\circ \mathcal{B}$ , then, for every  $L \notin \mathcal{B}$ , there is  $U$  computable in  $V$  and  $L$  so that  $U \dashv\circ \mathcal{B}$  and  $L = \text{diag}_U$ .*

*Proof.* We construct  $U$  in stages, one row at a time. Start with  $J_0 := \mathbb{N}$ . At stage  $e$ , set  $U_e := V_j$  and  $J_e := J_{e-1} - \{j\}$  where  $j$  is the smallest index in  $J_{e-1}$  satisfying  $V_j(e) = 1 - L(e)$ . Since  $\mathcal{B}$  is closed under finite variations and  $V \dashv\circ \mathcal{B}$ , at every stage  $e$  there is such  $j$ . Conversely, for every  $j$ , there is some stage  $e$  at which we set  $U_e = V_j$ . Otherwise there would be a smallest  $j$  which would be in  $J_i$  for all  $i$ . At some stage  $e$ , this  $j$  would become the smallest index in  $J_e$  and, since it would be never removed, for every  $e' > e$  we would have  $V_j(e') = L(e')$ . But this would imply  $L =^* V_j \in \mathcal{B}$ . Since  $\mathcal{B}$  is closed under finite variations, we would have  $L \in \mathcal{B}$ , contradicting the hypothesis. The construction gives  $U \dashv\circ \mathcal{B}$  satisfying, for all  $e$ ,  $L(e) = 1 - U_e(e) = \text{diag}_U(e)$  as desired and clearly  $U$  is computable in  $V$  and  $L$ .  $\square$

Kozen’s formulation (theorem 6.2 in [7]) is somewhat different and in particular refers to computable  $V$  and  $L$ , in which case  $V$  is computable.

This result shows that if  $\mathcal{B}$  is closed under finite variations, then any language  $L \notin \mathcal{B}$  is the diagonal of some universal language for  $\mathcal{B}$ . That is, any proof that  $L \notin \mathcal{B}$  can be recast as a *diagonalization* proof that  $L \notin \mathcal{B}$ . Notice that the proof that  $U$  is a universal language for  $\mathcal{B}$  requires the proof that  $L \notin \mathcal{B}$  inside of it and so it does not yield a new proof of  $L \notin \mathcal{B}$  independent of the original one.

### 3. Diagonalization and separation

In general, it is clear how to use diagonalization to establish non-membership, but for a constructive separation we need a little more:

- a definition of a language  $L$ ,
- a proof of  $L \notin \mathcal{B}$ , and
- a proof of  $L \in \mathcal{A}$ .

We call such  $L$  a *separating language* for  $\mathcal{B} \neq \mathcal{A}$ . Given  $U \dashv\circ \mathcal{B}$ , clearly  $L := \text{diag}_U$  satisfies the first two requirements. Theorem 6 shows that if  $\mathcal{B}$  is a class of computable languages closed under  $\leq_m^{\text{lin}}$ , then  $L := U$  also satisfies the first two requirements.

Regarding the last requirement, if we set  $L := \text{diag}_U$ , the least we can ask for is a proof that  $\text{diag}_U \in \mathcal{A}$ . We can

also ask for more: a proof that  $U \in \mathcal{A}$ , which under some closure conditions on  $\mathcal{A}$  (for example, closure under  $\leq_T^{\text{lin}}$ ) yields  $\text{diag}_U \in \mathcal{A}$ .

These considerations lead us to the following two definitions. A *separation by weak diagonalization* of  $\mathcal{A}$  and  $\mathcal{B}$  (implicitly used in Kozen’s interpretation of his results) requires:

1. a definition of  $U$ ,
2. a proof that  $U \dashv\vdash \mathcal{B}$ , and
3. a proof that  $\text{diag}_U \in \mathcal{A}$ .

These conditions imply  $\mathcal{A} \neq \mathcal{B}$  by proposition 1. A *separation by strong diagonalization* of  $\mathcal{A}$  and  $\mathcal{B}$  requires:

1. a definition of  $U$ ,
2. a proof that  $U \dashv\vdash \mathcal{B}$ ,
3. a proof that  $U \in \mathcal{A}$ , and
4. a proof that (a)  $\mathcal{A}$  is closed under  $\leq_T^{\text{lin}}$  or (b)  $\mathcal{B}$  is closed under  $\leq_m^{\text{lin}}$ .

By proposition 4, 4a implies  $\text{diag}_U \in \mathcal{A}$ . By theorem 6, 4b implies  $\mathcal{B} \not\text{-} \dashv\vdash \mathcal{B}$  (so  $U \notin \mathcal{B}$ ) and therefore both 4a and 4b imply  $\mathcal{A} \neq \mathcal{B}$ .<sup>1</sup>

If we limit ourselves to the case where  $\mathcal{B}$  is closed under  $\leq_m^{\text{lin}}$ , then using theorem 7, we have that  $\mathcal{A}$  and  $\mathcal{B}$  can be separated by weak or strong diagonalization iff there is a definition of a separating language  $L$  and proof of any kind that  $L \in \mathcal{A} - \mathcal{B}$  where

- for weak diagonalization,  $L$  is arbitrary, and
- for strong diagonalization,  $L \dashv\vdash \mathcal{B}$ .

This explains the use of the name “weak” and “strong” and shows that separation by strong diagonalization implies separation by weak diagonalization. These two definitions seem to provide two natural extremes to any definition of “separation by direct diagonalization.”

When  $\mathcal{A}$  and  $\mathcal{B}$  are given to us to separate (rather than constructed by us), it is hard to see how we can show that  $\text{diag}_U \in \mathcal{A}$  other than by proving first that  $U \in \mathcal{A}$ , unless we simply “retrofit”  $U$  to get a specific  $\text{diag}_U$  (theorem 7 shows this can be done), once we have proved  $L := \text{diag}_U \in \mathcal{A} - \mathcal{B}$  using any kind of proof method. This is in marked contrast to the situation where we construct classes which we then show to be distinct. In such a case, we may have  $U$  much harder to compute than  $\text{diag}_U$ .

<sup>1</sup>If you do not consider the proof of theorem 6 to be “by diagonalization” you may prefer the more conservative definition which instead of 4b requires (4b’)  $\mathcal{B}$  is closed under  $\leq_T^{\text{lin}}$ . The proof of theorem 6 goes beyond the framework outlined in the previous section, but we include condition 4b because that proof seems to fit within what is in practice considered to be diagonalization.

If  $P \neq NP$ , then using Ladner’s delayed diagonalization we can build languages  $A$  and  $B$  in NP which are Turing-incomparable and are neither in P nor NP-complete and so  $P^A$  and  $P^B$  are distinct. Here some kind of “diagonalization” is used to show  $A$  and  $B$  satisfy the properties mentioned above, but note that  $A$  and  $B$  are constructed during the process. Since we have  $P^A \not\subseteq P^B$  and  $P^B \not\subseteq P^A$ , strong diagonalization can’t separate  $P^A$  from  $P^B$ , but weak diagonalization can. (Using theorem 7 we can get  $U \dashv\vdash P^A$  and  $V \dashv\vdash P^B$  such that  $A = \text{diag}_V$  and  $B = \text{diag}_U$ .) It easily follows from the definition that if several classes  $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_k$  can’t be separated from another class  $\mathcal{B}$  by strong diagonalization, then the union  $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \dots \cup \mathcal{A}_k$  can’t be so separated from  $\mathcal{B}$ , since none of  $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_k$  contain a weak universal language for  $\mathcal{B}$ . Therefore  $P^B$  can’t be separated from  $P^A \cup P^B$  by strong diagonalization, even though  $P^B \subsetneq P^A \cup P^B$ . This provides an example of two classes, one a subset of the other, which can’t be separated by strong diagonalization. Since these classes can be separated by weak diagonalization, the two notions (strong and weak) are different.

Furthermore, theorem 10 (in the next section) shows that for any languages  $B$  and  $L$  with  $B <_T^P L$ , there is a language  $A$  such that  $B <_T^P A \leq_T^P L$  and  $P^A \not\text{-} \dashv\vdash P^B$ . That is,  $P^A$  and  $P^B$  can’t be separated by strong diagonalization, even though  $P^B \subsetneq P^A$ .

Note that theorem 10 includes the case where  $B = 0$  and  $L$  is such that  $\text{SAT} \leq_T^P L$ . Hence there is an  $A \notin P$  such that  $NP \subseteq P^L$ ,  $P^A \subseteq P^L$ , and  $P^A \not\text{-} \dashv\vdash P$ . If, moreover, we had  $\text{SAT} \leq_T^P A$ , we could deduce  $NP \not\text{-} \dashv\vdash P$ , i.e., NP would not be separable from P by strong diagonalization.

The space and time hierarchy theorems are separations by strong diagonalization.

**Proposition 8.** *If  $f, g$  are space-constructible,  $\log \leq f$ , and  $f \in o(g)$ , then  $\text{SPACE}[g] \dashv\vdash \text{SPACE}[f]$ .*

*Proof.* (outline) We know we can simulate Turing machine  $e$  on input  $x$  limited to space  $f(|x|)$ , including detection of an infinite loop with only a linear loss of space efficiency. We compute  $U_{\langle i, j \rangle}(x)$  as follows. We simulate  $\phi_i(x)$  limited to space  $f(|x|)$  using at most  $g(|\langle i, j, x \rangle|)$  space. If the computation succeeds, we return its value; otherwise we return 0. Since  $f \in o(g)$ , for every  $i$  and  $j$  one of two things can happen: (a) the simulation succeeds for all  $x$ , in which case  $U_{\langle i, j \rangle} \in \text{SPACE}[f]$ , or (b) the simulation runs out of space for finitely many  $x$ , in which case  $U_{\langle i, j \rangle}$  is a finite variation of a language in  $\text{SPACE}[f]$  and so it is in  $\text{SPACE}[f]$ . On the other hand, for every  $i$ , there is a  $j$  so that (a) above happens and therefore  $U \dashv\vdash \text{SPACE}[f]$ . Since we limit the space used in the simulation by  $g$ , we have  $U \in \text{SPACE}[g]$ . We are using the space-constructibility of  $f$  and  $g$  to efficiently limit the computations to space  $f$  and  $g$  and the fact that  $g \geq \log$  to

detect infinite loops and (together with  $f \in o(g)$ ) to ensure that for any value of  $c$  and  $d$  there is some  $j$  such that

$$\forall i, x(c \cdot f(|x|) + d \leq g(|\langle i, j, x \rangle|)).$$

□

**Proposition 9.** *If  $f, g$  are time-constructible and  $f \in o(g/\log g)$ , then  $\text{TIME}[g] \dashv\vdash \text{TIME}[f]$ .*

*Proof.* (outline) We know we can simulate Turing machine  $e$  on input  $x$  limited to time  $f(|x|)$ , in time  $f(|x|) \log(|x|)$ . We compute  $U_{\langle i, j \rangle}(x)$  as follows. We simulate  $\phi_i(x)$  limited to time  $f(|x|)$  using at most  $g(|\langle i, j, x \rangle|)$  time. If the computation succeeds, we return its value; otherwise we return 0. Since  $f \in o(g/\log g)$ , for every  $i$  and  $j$  one of two things can happen: (a) the simulation succeeds for all  $x$ , in which case  $U_{\langle i, j \rangle} \in \text{TIME}[f]$ , or (b) the simulation runs out of time for finitely many  $x$ , in which case  $U_{\langle i, j \rangle}$  is a finite variation of a language in  $\text{TIME}[f]$  and so it is in  $\text{TIME}[f]$ . On the other hand, for every  $i$ , there is a  $j$  so that (a) above happens and therefore  $U \dashv\vdash \text{TIME}[f]$ . Since we limit the time used in the simulation by  $g$ , we have  $U \in \text{TIME}[g]$ . We are using the time-constructibility of  $f$  and  $g$  to efficiently limit the computations to time  $f$  and  $g$ . Since  $f \in o(g/\log g)$  and so  $g > \log$  we have that for any value of  $c$  and  $d$  there is some  $j$  so that

$$\forall i, x(c \cdot f(|x|) \log(|x|) + d \leq g(|\langle i, j, x \rangle|)).$$

□

As a consequence we have

- $\text{EXP} \dashv\vdash \text{P}$ ,
- $\text{EXPSPACE} \dashv\vdash \text{PSPACE}$ , and
- $\text{PSPACE} \dashv\vdash \text{NLOGSPACE}$ .

All these results relativize.

Next we consider, as an example, a separation which is an easy consequence of the space hierarchy theorem. Its proof uses strong diagonalization and some additional logic. Since we know from above that  $\text{PSPACE} \dashv\vdash \text{LSPACE}$ , we can deduce that  $\text{P}^{\text{LSPACE}} = \text{PSPACE} \neq \text{LSPACE}$ . That is,  $\text{LSPACE}$  is not closed under  $\leq_m^{\text{P}}$ , in contrast to  $\text{P}$  and therefore  $\text{LSPACE} \neq \text{P}$ .

Since we know that  $\text{EXP} \dashv\vdash \text{P}$ , if we could show  $\text{NP} \not\leq_m^{\text{P}} \text{P}$  we would be able to deduce  $\text{NP} \neq \text{EXP}$ . This not only shows that proving  $\text{NP} \not\leq_m^{\text{P}} \text{P}$  is hard, but also suggests a formalization of two very related kinds of separation by indirect diagonalization (for lack of better name, we simply call them “indirect A” and “indirect B”:  $A \neq B$  if there is a  $\mathcal{C}$ , closed under  $\leq_m^{\text{lin}}$ , so that either

1.  $A \dashv\vdash \mathcal{C}$  but  $B \not\leq_m^{\text{P}} \mathcal{C}$  (indirect A) or
2.  $\mathcal{C} \not\leq_m^{\text{P}} A$  but  $\mathcal{C} \dashv\vdash B$  (indirect B)

We call  $\mathcal{C}$  a *separating class*. Notice that we allow arbitrary methods to prove  $B \not\leq_m^{\text{P}} \mathcal{C}$  or to prove  $\mathcal{C} \not\leq_m^{\text{P}} A$ . If we allowed arbitrary classes  $\mathcal{C}$ , then separation by weak diagonalization would fall under indirect B diagonalization. That is, if we had

1. a proof that  $U \dashv\vdash B$  and
2. a proof that  $\text{diag}_U \in A$ ,

we could simply set  $\mathcal{C} = \{U\}$ . Then the first item would imply  $\mathcal{C} \dashv\vdash B$  and the second item (together with proposition 1) would imply  $\mathcal{C} \not\leq_m^{\text{P}} A$ .

In theorem 13 we show that if  $U \dashv\vdash P^B$ , then

$$\exists V(B \leq_m^{\text{P}} V \leq_m^{\text{P}} U \text{ and } P^U \not\leq_m^{\text{P}} P^V \not\leq_m^{\text{P}} P^B).$$

This shows that separations by indirect diagonalization are more powerful than separations by strong diagonalization, even when the separating class is closed under  $\leq_m^{\text{P}}$ . That is, take  $A = P^U$  and  $B = P^V$  and  $\mathcal{C} = P^B$  with  $U \dashv\vdash P^B$ ; then

$$A \not\leq_m^{\text{P}} B \not\leq_m^{\text{P}} \mathcal{C} \text{ and } A \dashv\vdash \mathcal{C}.$$

Similarly, take  $\mathcal{C} = P^U$  and  $A = P^V$  and  $B = P^B$  with  $U \dashv\vdash P^B$ ; then

$$\mathcal{C} \not\leq_m^{\text{P}} A \not\leq_m^{\text{P}} B \text{ and } \mathcal{C} \dashv\vdash B.$$

That is,  $A$  and  $B$  can't be separated by strong diagonalization, but can be separated by indirect diagonalization A or B.

## 4. Structural results

The following three theorems give structural results for universal languages of a fixed, arbitrary class of languages  $P^B$ .

**Theorem 10.** *For all  $B$  and  $L$ ,*

$$B <_T^{\text{P}} L \implies \exists A(B <_T^{\text{P}} A \leq_m^{\text{P}} L \text{ and } P^A \not\leq_m^{\text{P}} P^B)$$

and

$$B <_m^{\text{P}} L \implies \exists A(B <_m^{\text{P}} A \leq_m^{\text{P}} L \text{ and } P^A \not\leq_m^{\text{P}} P^B).$$

*Proof.* We prove the first statement. The proof of the second statement is similar. We use Ladner's delayed diagonalization technique (see [8] and [2]) to construct a set  $A$  which meets a certain list of requirements  $\{R_k: k \in \mathbb{N}\}$ . We will have a function  $\rho$  which, for every stage  $s$ , will give the number  $\rho(s)$  of the requirement that we will try to

satisfy at stage  $s$ . We will use the length of the input  $z$  to indicate the stage; i.e.,  $s = |z|$ .  $\rho$  will be increasing, but will usually grow very, very slowly.

To ensure that  $A \not\leq_T^P B$ , we shall satisfy the following requirements for all  $e$ :

$$R_{\langle e,0 \rangle}: \quad \exists x (A(x) \neq \psi_{e;e}^B(x)).$$

To satisfy requirement  $R_{\langle e,0 \rangle}$ , we use the fact that  $L \notin P^B$ . At stage  $s = |z|$ , when we want meet requirement  $R_{\langle e,0 \rangle}$ , we set  $A(z) = L(z)$ . We keep trying to meet requirement  $R_{\langle e,0 \rangle}$  at every stage  $|z'| = s' > s$  by setting  $A(z') = L(z')$  until we find  $A(z') = L(z') \neq \psi_{e;e}^B(z')$ . We must eventually find such a  $z'$  since otherwise we would have  $L =^* A =^* \psi_{e;e}^B$ , contradicting  $L \notin P^B$ .

To ensure that  $P^A \not\text{-} \text{f-} \text{o} P^B$  we shall satisfy the following requirements for all  $e$  and  $i$ :

$$R_{\langle e,i+1 \rangle}: \quad \exists x (L_e(x) \neq \psi_{e;e}^A(\langle i, x \rangle)).$$

To satisfy requirement  $R_{\langle e,i+1 \rangle}$ , we use the language  $L_e = \text{diag}_V$  with  $\text{TIME}^B [n^{e+1}] \xrightarrow{V} \text{TIME}^B [n^e]$  given by the time hierarchy theorem, which we compute as we proceed. At stage  $s = |z|$ , when we want to meet requirement  $R_{\langle e,i+1 \rangle}$ , we set  $A(z) = B(z)$ . We keep trying to meet requirement  $R_{\langle e,i+1 \rangle}$  at every stage  $|z'| = s' > s$  by setting  $A(z') = B(z')$  until we find  $z'$  such that  $L_e(z') \neq \psi_{e;e}^A(z')$ . Such  $z'$  must eventually appear since otherwise we would have  $A =^* B$  and  $L_e =^* \psi_{e;e}^A$ , contradicting  $L_e \notin \text{TIME}^B [n^e]$ .

To ensure  $B \leq_T^P A \leq_T^P L$  we use Ladner's delayed diagonalization technique as follows. We define a function  $\rho$  satisfying the following conditions.

1. If  $x < y$ , then  $\rho(x) \leq \rho(y)$ .
2. If  $x + 1 = y$ , then  $\rho(x) + 1 \geq \rho(y)$ .
3.  $\rho(0) = 0$ .
4.  $\rho(s)$  is computable in time  $O(s)$ .

We compute  $\rho(s)$  as follows:

- For  $s$  steps, compute  $\rho(0), \rho(1), \rho(2), \dots$ . Say  $\rho(s') = r$  (with  $s' < s$ ) is the last value for which this computation can be completed.
- For  $s$  steps try to satisfy requirement  $R_r$ .
- Set  $\rho(s) := \begin{cases} r + 1 & \text{if } R_r \text{ has been satisfied} \\ r & \text{otherwise.} \end{cases}$

Thus  $\rho$  determines, within the given time limit, which requirement we were last trying to satisfy and then determines whether that requirement has been satisfied. While we try to satisfy requirement  $R_r$  in the computation of  $\rho(s)$ , we

may need to compute  $\rho(w)$  recursively and we handle this as follows. If we need the value  $\rho(w)$  for  $w \geq s$ , we simply abort that computation and set  $\rho(s) = r$ . If we need the value  $\rho(w)$  for  $w < s$ , we go ahead and compute it.

Using  $\rho$  we define  $A$  as follows:

$$A(z) := \begin{cases} L(z) & \text{if } \pi_2(\rho(|z|)) = 0 \\ B(z) & \text{otherwise.} \end{cases}$$

Then we have  $A \leq_m^P L \oplus B$  by  $h$  given by

$$h(z) := \begin{cases} 2z & \text{if } \pi_2(\rho(|z|)) = 0 \\ 2z + 1 & \text{otherwise.} \end{cases}$$

That is, if  $\pi_2(\rho(|z|)) = 0$ , we have  $A(z) = L(z) = (L \oplus B)(2z)$  and if  $\pi_2(\rho(|z|)) \neq 0$ , we have  $A(z) = B(z) = (L \oplus B)(2z + 1)$ . Since  $B \leq_T^P L$ , we get  $A \leq_T^P L$ .

Furthermore  $B \leq_T^P A$  since if  $\pi_2(\rho(|z|)) \neq 0$ , then  $B(z)$  is equal to  $A(z)$  and otherwise  $B(z)$  can be computed from  $A(z) = L(z)$  because  $B \leq_T^P L$ .

Since we attempt to satisfy requirements in sequence, if there is an unsatisfied requirement  $R_r$ , we would have  $\rho$  eventually constant with value  $r$ , and we have shown above that this leads to a contradiction.  $\square$

**Theorem 11.** *If  $U \text{---} \text{o} P^B$  and  $W <_T^P U$ , then*

$$\exists V (W <_T^P V <_T^P U) (V \text{---} \text{o} P^B).$$

*Proof.* By Ladner's density theorem, we know there is a language  $L$  such that  $W <_T^P L <_T^P U$ . We build  $V$  in stages, row by row, using Ladner's delayed diagonalization to get  $V \leq_m^P U$ . We set

$$V_j(x) := \begin{cases} U_{\rho(|j|)/2}(x) & \text{if } \rho(|j|) \text{ is even} \\ L(\lfloor \rho(|j|)/2 \rfloor) & \text{if } \rho(|j|) \text{ is odd and } x = 0 \\ 0 & \text{if } \rho(|j|) \text{ is odd and } 0 < x \end{cases}$$

where  $\rho$  is as in the proof of theorem 10.

To ensure  $V \text{---} \text{o} P^B$  we shall satisfy the following requirements for all  $e$ :

$$R_{2e}: \quad \exists j (V_j = U_e).$$

At stage  $s$  when we want to meet requirement  $R_{2e}$ , we look for some  $s' < s$  with  $\rho(s') = \rho(s) = 2e$ . If there is such  $s'$ , then we have  $V_j = U_e$  for all  $j$  for which  $|j| = s'$ , satisfying the requirement. There must be some  $s$  large enough to give us enough time to find  $s' < s$  as desired.

To ensure  $U \not\leq_T^P V$  we shall satisfy the following requirements for all  $e$ :

$$R_{2e+1}: \quad \exists z (U(z) \neq \psi_{e;e}^V(z)).$$

At stage  $s$  when we want to meet requirement  $R_{2e+1}$  (i.e., when  $\rho(s) = 2e + 1$ ), we search for  $z$  satisfying  $U(z) \neq$

$\psi_{e;e}^V(z)$ . We must find some such  $z$  at some stage  $s' \geq s$  since otherwise we would have  $\rho$  eventually constant with value  $2e + 1$  and then  $V$  would index only finitely many languages of  $P^B$  and we would have  $V \leq_T^P B$ . So if we had  $U \leq_T^P V$ , we would have  $U \in P^B$  contradicting  $U \not\rightarrow P^B$  by proposition 5.

Since every row we add to  $V$  is either a finite variation of the empty language or a language in  $P^B$ , if we meet all requirements, we get  $V \rightarrow P^B$ . We can recover  $L$  from the first position of the odd rows of  $V$ :  $L(j) := V_{\lfloor \rho(|j|)/2 \rfloor}(0)$  and so we have  $W <_T^P L \leq_m^P V$ , which gives  $W <_T^P V$ .  $\square$

Notice it is not essential to have  $\mathcal{B} = P^B$  in the proof; it is enough to have  $\mathcal{B}$  closed under  $\leq_m^{\text{lin}}$  and a computable  $W \rightarrow \mathcal{B}$ . It is not clear whether there are infinite descending chains of weak universal languages, since if  $U \dashrightarrow P^B$  it may be, for example, that  $U_e = U$  for some  $e$  or that  $U_e$  is a language of high complexity, and there seems to be no easy way of detecting this. Notice that we have used the fact that  $U \rightarrow P^B$  in the proof only to conclude that  $U \not\leq_T^P U_1 \oplus U_2 \oplus \dots \oplus U_k$  for any  $k$  and therefore the same proof gives the following corresponding theorem for weak universal languages.

**Theorem 12.** *If  $U \dashrightarrow P^B$  and  $W <_T^P U$  and  $U$  is such that for any  $k$ ,  $U \not\leq_T^P U_1 \oplus U_2 \oplus \dots \oplus U_k$ , then*

$$\exists V (W <_T^P V <_T^P U) (V \dashrightarrow P^B).$$

**Theorem 13.** *If  $U \dashrightarrow P^B$ , then  $\exists V (B \leq_m^P V \leq_m^P U$  and  $P^U \not\rightarrow P^V \not\rightarrow P^B$ ).*

*Proof.* We build  $V$  in stages, cell by cell (not row by row), using Ladner's delayed diagonalization to get  $V \leq_m^P U$ . We set

$$V(z) := \begin{cases} U(z) & \text{if } \pi_1(\rho(|z|)) \text{ is even} \\ B(z) & \text{if } \pi_1(\rho(|z|)) \text{ is odd} \end{cases}$$

where  $\rho$  is as in the proof of theorem 10.

We use the languages  $L_e^B := \text{diag}_{W_e}$  and  $L_e^V := \text{diag}_{W'_e}$  where  $W_e$  and  $W'_e$  satisfy

$$\text{TIME}^B [n^{e+1}] \xrightarrow{W_e} \text{TIME}^B [n^e]$$

and

$$\text{TIME}^V [n^{e+1}] \xrightarrow{W'_e} \text{TIME}^V [n^e].$$

The existence of such languages  $W_e$  and  $W'_e$  follows from the time hierarchy theorem. We will use  $L_e^V$  to witness  $\psi_{e;e}^U \not\rightarrow P^V$  and  $L_e^B$  to witness  $\psi_{e;e}^V \not\rightarrow P^B$ .

To ensure  $P^U \not\rightarrow P^V$ , we satisfy the requirements

$$R_{\langle 2e, i \rangle}: \quad \exists x (L_e^V(x) \neq \psi_{e;e}^U(\langle i, x \rangle)).$$

At stage  $s = |z|$  when we want to meet requirement  $R_{\langle 2e, i \rangle}$  (i.e., when  $\rho(s) = \langle 2e, i \rangle$ ), we set  $V(z) = U(z)$ . There

must be some  $z$  for which we satisfy the requirement, since otherwise we would have  $V =^* U$  and  $L_e^V =^* \psi_{e;e}^U(\langle i, \bullet \rangle)$  contradicting  $L_e^V \notin \text{TIME}^V [n^e]$ .

To ensure  $P^V \not\rightarrow P^B$  we satisfy the requirements

$$R_{\langle 2e+1, i \rangle}: \quad \exists x (L_e^B(x) \neq \psi_{e;e}^V(\langle i, x \rangle)).$$

At stage  $s = |z|$  when we want to meet requirement  $R_{\langle 2e+1, i \rangle}$  (i.e., when  $\rho(s) = \langle 2e + 1, i \rangle$ ), we set  $V(z) = B(z)$ . There must be some  $z$  for which we satisfy the requirement, since otherwise we would have  $V =^* B$  and  $L_e^B =^* \psi_{e;e}^V(\langle i, \bullet \rangle)$  contradicting  $L_e^B \notin \text{TIME}^B [n^e]$ .  $\square$

Theorem 13 shows that  $\not\rightarrow$  is "strongly" non-transitive. In particular, the binary relation on  $\mathcal{A} \times \mathcal{B}$  defined by  $\mathcal{A} \not\rightarrow \mathcal{B}$  is not an equivalence relation and so we can not define equivalence classes of languages for which  $\rightarrow$  gives a partial order.

By definition, if  $U \rightarrow \mathcal{B}$ , then  $U \dashrightarrow \mathcal{B}$ . On the other hand, we have

**Theorem 14.** *For every  $L$ , there is  $\mathcal{B}_L \subseteq \text{TIME}[n]$  closed under finite variations so that there is  $P \dashrightarrow \mathcal{B}_L$ , but for any  $V \rightarrow \mathcal{B}_L$ ,  $L$  is  $\Sigma_2^0$  in  $V$ .*

*Proof.* Set

$$Z_k(n) := \begin{cases} 1 & \text{if } k \text{ divides } n \\ 0 & \text{otherwise} \end{cases}$$

and  $\mathcal{B}'_L := \{Z_k: L(k) = 1\}$ . Take  $\mathcal{B}_L$  to be the closure under finite variations of  $\mathcal{B}'_L$ . Then  $U'$  given by  $U'_e(n) := Z_e(n)$  is a weak universal language of  $\mathcal{B}'_L$  and so a weak pseudouniversal language for  $\mathcal{B}_L$ . Proposition 16 (below) shows that there is  $U \leq_m^{\text{lin}} U'$  so that  $U \dashrightarrow \mathcal{B}_L$ . Since all we need to do to compute  $U'_e(n)$  is test whether  $e$  divides  $n$ , we have  $U \in P$ . On the other hand, given  $V \rightarrow \mathcal{B}_L$ , we have  $k \in L$  iff  $Z_k \in \mathcal{B}'_L$  iff

$$\exists e, m (\forall n \geq m) (V_e(n) = 1 \iff k \text{ divides } n).$$

$\square$

If we take  $L \notin \Sigma_2^0$ , then  $\mathcal{B}_L$  has a weak universal language in  $P$ , but no recursive universal language.

To summarize, our results tell us the following for universal languages of a fixed arbitrary class  $P^B$ .

1. There are universal languages of arbitrarily high complexity (theorem 7).
2. Below them, there are dense chains of universal languages (under  $\leq_T^P$ ) (theorem 11).
3. None of these chains come arbitrarily close to  $P^B$ . Instead, there is a strict superset of  $P^B$  which has no universal languages (theorem 10).

4. Furthermore, no matter how high the complexity of a universal language  $U$ , we can find a superset  $P^V$  of  $P^B$  so that  $P^U \not\leq_{\text{m}} P^V \not\leq_{\text{m}} P^B$ .

We have the same results for weak universal languages, except for item 2 (but see theorem ??). Finally, theorem 14 shows that there are classes of languages which have weak universal languages of low complexity, but no universal languages below any arbitrarily high fixed complexity.

## 5. Variants of universal languages

We have seen that  $U \dashv\vdash \mathcal{B}$  implies  $\text{diag}_U \notin \mathcal{B}$ . It is natural to ask how much further we can weaken  $U$  and still get  $\text{diag}_U \notin \mathcal{B}$ , and how such weaker notions relate to one another.

A language  $U$  has padding iff for all  $e$  the set  $\{i: U_e = U_i\}$  is infinite. It is computationally very easy to obtain a universal language  $V$  with padding from an arbitrary universal language  $U$ : define  $V_e = U_{\pi_1(e)}$ . This gives  $V$  with padding and  $V \leq_m^{\text{lin}} U$ .

**Proposition 15.** *If  $U \dashv\vdash \mathcal{B}$  and  $U$  has padding, then  $\forall e(\text{diag}_U \neq^* U_e)$ .*

*Proof.* By definition of  $\text{diag}_U$ , we have

$$\forall i(\text{diag}_U(i) \neq U_i(i))$$

and so

$$\begin{aligned} \{i: U_i = U_e\} &\subseteq \{i: U_i(i) = U_e(i)\} \\ &= \{i: \text{diag}_U(i) \neq U_e(i)\} \end{aligned}$$

Since  $U$  has padding, the set on the left is infinite and so the one on the right is infinite too, that is  $\text{diag}_U \neq^* U_e$ .  $\square$

Therefore, when  $U$  has padding, we can require less than a weak universal language in order to get  $\text{diag}_U \notin \mathcal{B}$ . When  $U$  has padding, it is enough to have  $U$  give, for every language  $B$  in  $\mathcal{B}$ , one row  $U_e$  that is almost equal to it ( $U_e =^* B$ ). Accordingly, we say that  $U$  is a *weak pseudouniversal language* for  $\mathcal{B}$  iff it satisfies

$$(\forall B \in \mathcal{B}) \exists e (U_e =^* B).$$

Similarly, we say that  $U$  is a *pseudouniversal language* for  $\mathcal{B}$  if it also satisfies

$$\forall e (\exists B \in \mathcal{B})(U_e =^* B).$$

It is easy to get (weak) universal languages from (weak) pseudouniversal languages, as the following two propositions show.

Let  $E$  be the empty language (i.e.,  $\forall n(E(n) = 0)$ ) and  $F$  be its complement (i.e.,  $\forall n(F(n) = 1)$ ).

**Proposition 16.** *If  $U$  is a weak pseudouniversal language for  $\mathcal{B}$  and  $\mathcal{B} \neq \{E\}$  and  $\mathcal{B} \neq \{F\}$ , then there is  $V \dashv\vdash \mathcal{B}$  with  $V \leq_m^{\text{lin}} U$ .*

*Proof.* Define  $V \dashv\vdash \mathcal{B}$  by

$$V_{\langle e, \hat{\sigma} \rangle}(x) = \begin{cases} \sigma(x) & \text{if } x < |\sigma| \\ U_e(x) & \text{otherwise} \end{cases}$$

(remember that  $\hat{\sigma}$  is the number associated with the string  $\sigma$ ). We let  $V_j$  be all zeros if there are no  $e, \hat{\sigma}$  so that  $j = \langle e, \hat{\sigma} \rangle$ . That is,  $V$  is obtained from  $U$  as follows. Every row  $U_e$  appears infinitely often in  $V$  as  $V_{\langle e, \hat{0} \rangle}, V_{\langle e, \hat{1} \rangle}, V_{\langle e, \hat{00} \rangle}, V_{\langle e, \hat{01} \rangle}, \dots$ , except that its initial portion is replaced by the strings  $0, 1, 00, 01$ , and so on. This ensures that  $V$  indexes all finite variations of  $U_e$  and therefore we have  $V \dashv\vdash \mathcal{B}'$  where  $\mathcal{B}'$  is the closure of  $\mathcal{B}$  under finite variations. Since  $\mathcal{B} \neq \{E\}, \{F\}$  there are constants  $u_0$  and  $u_1$  such that  $U(u_0) = 0$  and  $U(u_1) = 1$ . The function  $g$  given by

$$g(\langle e, \hat{\sigma}, x \rangle) := \begin{cases} u_{\sigma(x)} & \text{if } x < |\sigma| \\ \langle e, x \rangle & \text{otherwise} \end{cases}$$

witnesses  $V \leq_m^{\text{lin}} U$ .  $\square$

This proof also shows the following.

**Proposition 17.** *If  $U$  is a pseudouniversal language for  $\mathcal{B}$  and  $\mathcal{B}$  is closed under finite variations, then there is  $V \dashv\vdash \mathcal{B}$  with  $V \leq_m^{\text{lin}} U$ .*

## 6. Relativization and strong diagonalization

In this section, we consider the problem of whether our notion of separation by strong diagonalization relativizes. The following propositions provide evidence that separations by strong diagonalization do not relativize.

**Proposition 18.** *There is a computable oracle  $A$  so that  $\text{NP}^A \not\leq \text{P}^A$ .*

*Proof.* There is a computable oracle  $A$  so that  $\text{EXP}^A = \text{NP}^A$  [5] (see also [3]) and we know that  $\text{EXP}^A \not\leq \text{P}^A$  (by the time hierarchy theorem).  $\square$

**Proposition 19.** *There is a computable oracle  $B$  so that  $\text{NP}^B \not\leq \text{P}^B$ .*

*Proof.* Take  $B$  a PSPACE-complete language. Then

$$\text{NP}^B \subseteq \text{PSPACE}^B \subseteq \text{PSPACE} \subseteq \text{P}^B \subseteq \text{NP}^B$$

and therefore, by proposition 5 we have  $\text{NP}^B = \text{P}^B \not\leq \text{P}^B$ .  $\square$

The two propositions above show that it can't be established whether  $NP \dashv\vdash P$  or not using techniques that relativize. Also, while we do not know whether  $NP \dashv\vdash P$  or not, one of the two, existence or nonexistence of a weak universal language for  $P$  in  $NP$ , doesn't relativize.

**Proposition 20.** *There are computable languages  $A$  and  $C$  such that  $NP^A \dashv\vdash P^A$ , yet  $NP^{A\oplus C} = P^{A\oplus C}$  and therefore  $NP^{A\oplus C} \not\dashv\vdash P^{A\oplus C}$ .*

*Proof.* Take  $A$  as in theorem 18 and  $C$  to be a  $PSPACE^A$ -complete language. Then  $NP^A \dashv\vdash P^A$  by theorem 18 and

$$P^{A\oplus C} \subseteq NP^{A\oplus C} \subseteq PSPACE^{A\oplus C} = P^C \subseteq P^{A\oplus C}.$$

□

If you accept the definition of  $(P^A)^C$  to be  $P^{A\oplus C}$  and that of  $(NP^A)^C$  to be  $NP^{A\oplus C}$ , then theorem 20 shows that separations by strong diagonalization do not relativize.

## 7. Conclusion

“Separation by direct diagonalization” is not clearly defined, but there are two natural bounds on what such a definition might be. Kozen showed that separation by weak diagonalization is essentially equivalent to constructive separation. We showed that separation by strong diagonalization, which includes the time and space hierarchy theorems, is a more restrictive notion and that there are classes of languages it can not separate. We have shown evidence that strong diagonalization doesn't relativize.

Turning to universal languages, we have shown that, for a fixed arbitrary class of languages  $P^B$ , there are universal languages of arbitrarily high complexity, there are dense descending chains under every strict universal language, and there is a non-trivial lower bound to every descending chain of universal languages. We have shown that apparent weakenings of the notion of universal language suitable for diagonalization are, in fact, not much weaker.

We have shown that a universal language can be used as a separating language when we have closure under linear-time many-one reducibility. Finally, we have shown that two kinds of indirect diagonalization,  $A$  and  $B$ , are stronger than strong diagonalization even limited to separating classes closed under  $\leq_T^P$ .

## Acknowledgments

Thanks to Lance Fortnow for bringing our attention to the technique in [9], to Valentin Kabanets for an informative discussion, and to the referees for their helpful comments.

## References

- [1] Baker, Gill, and Solovay. Relativizations of the  $P = ?$  NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.
- [2] J. L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I*. Springer, 2nd edition, 1995.
- [3] R. Beigel, H. Buhrman, and L. Fortnow. NP might not be as easy as detecting unique solutions. In *Proc. of 30-th ACM Symp. on the Theory of Computing*, pages 203–208, New York, 1998. ACM.
- [4] G. Cantor. Ueber eine eigenschaft des inbegriffes aller reellen algebraischen zahlen. *Crelle's Journal*, 77:258–262, 1874.
- [5] M. Dekhtiar. On the relativization of deterministic and non-deterministic classes. *Lecture Notes in Computer Science*, 45:255–259, 1976.
- [6] L. Fortnow. Diagonalization. *Bulletin of the European Association for Theoretical Computer Science*, 71:102–112, 2000.
- [7] D. Kozen. Indexings of subrecursive classes. *Theoretical Computer Science*, 11:277–301, 1980.
- [8] R. E. L. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, 1975.
- [9] S. Žák. A turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, 1983.