Strong isomorphism reductions in complexity theory

Sam Buss
sbuss@math.ucsd.edu

Yijia Chen
yijia.chen@cs.sjtu.edu.cn

Jörg Flum
joerg.flum@math.uni-freiburg.de

Sy Friedman
sdf@logic.univie.ac.at

Moritz Müller
mmueller@crm.cat

1. Introduction

In many areas of computational complexity, polynomial time reduction is the appropriate notion for comparing the complexity of problems. However, suppose that we face, for example, the problem of comparing the complexity of the isomorphism problem for two classes C and D of graphs. Here

$$\mathrm{Iso}(C) := \big\{ (\mathcal{A}, \mathcal{B}) \mid \mathcal{A}, \mathcal{B} \in C \text{ and } \mathcal{A} \cong \mathcal{B} \big\}$$

is the isomorphism problem for C (more precisely, the set of positive instances of this problem) and $\mathrm{Iso}(D)$ is defined analogously. Probably we would not accept a polynomial time computable function $f : C \times C \to D \times D$ with

$$(\mathcal{A}, \mathcal{B}) \in \mathrm{Iso}(C) \iff f(\mathcal{A}, \mathcal{B}) \in \mathrm{Iso}(D)$$

as the right notion of reduction in this context but we would seek a *strong isomorphism reduction*, that is, a polynomial time computable function $f : C \to D$ with

$$\mathcal{A} \cong \mathcal{B} \iff f(\mathcal{A}) \cong f(\mathcal{B}). \tag{1}$$

This paper is devoted to the study of this type of reduction. For us the motivation for this study came from various areas:

Computational complexity: The isomorphism relation (on a class C) is an equivalence relation. In the context of arbitrary equivalence relations a notion of reduction defined analogously as in (1) (and that for the isomorphism relation coincides with our notion) has been introduced in [7]. However that paper is mainly devoted to other problems (see the end of Section 7 for some more details); concerning the notion of reduction only some open problems are stated in [7], problems we address in our paper.

Descriptive set theory: For the isomorphism relation our notion of reduction was first considered by the fourth author (see [8]) inspired by the analogous notion from descriptive set theory (see [9]). In descriptive set theory, C and D denote classes of structures with universe $\mathbb{N}$ and the function f satisfying (1) is required to be Borel (in the topology generated by the first-order definable classes).

Descriptive complexity: The existence of a logic capturing polynomial time remains the central open problem of descriptive complexity theory. For many classes C of graphs (or of other types of structures), one shows that a logic L captures polynomial time *on C* by defining in L an invariantization for C. From the definition of invariantization (given in Section 4), one immediately gets that if C is strongly isomorphism reducible to D, then C has an invariantization if D has one.

This paper contains the first systematic study of strong isomorphism reductions. In Section 3 and Section 4 we introduce our framework, derive some basic properties of strong isomorphism reductions, and explain via invariantizations and canonizations the relationship to logics capturing polynomial time mentioned above. At various places of our analysis, invariantizations and canonizations will be valuable tools. Their relationship and the computational complexity of problems related to these notions have been studied in [2, 3, 7, 11, 13, 14].

We denote by $\leq_{\mathrm{iso}}$ the partial ordering on the set of degrees induced by strong isomorphism reductions. While already in Section 3 we observe that (the degree of) the class of graphs is the $\leq_{\mathrm{iso}}$ maximum element, by Theorem 11 we see that some "basic algebraic classes of structures" all have the same strong isomorphism degree. In Section 5 we show that the structure of $\leq_{\mathrm{iso}}$ is rich already when restricting to classes with an invariantization.

Assume that C is strongly isomorphism reducible to D. Since such reductions are computable in polynomial time we know that for some polynomial $p \in \mathbb{N}[X]$ and all $n \in \mathbb{N}$ the number of isomorphism types of structures in C with at most n elements is at most the number of isomorphism types of structures in D with $\leq p(n)$ elements. If this condition is satisfied, then following [8] we say that C is potentially reducible to D. Already in Section 5 this concept is the main tool to demonstrate the richness of the partial ordering $\leq_{\mathrm{iso}}$. We believe that the notions of strong isomorphism reducibility and that of potential reducibility are distinct but can only show this under the hypothesis U2EXP $\cap$ co-U2EXP $\neq$ 2EXP (see Section 6). It turns out in Section 7 that we would get P $\neq$ #P if we could separate the two notions without any complexity-theoretic assumption.

The isomorphism relation is an equivalence relation in NP. In Section 8 we study reductions (defined in analogy to (1)) between arbitrary equivalence relations in NP. In particular, we show that there is a maximum element in the corresponding partial ordering if and only if there is an effective enumeration of these equivalence relations by means of clocked Turing machines. Even if we restrict to equivalence relations in P (= PTIME), we cannot show that a maximum element exists; we can guarantee its existence if a p-optimal propositional proof system exists. The existence of a maximum element for equivalence relations in P was addressed in [7, Open Question 4.14].

2. Some preliminaries

Throughout the paper Σ denotes the alphabet $\{0, 1\}$ and we let Σ^* be the set of strings over this alphabet. For $n \in \mathbb{N}$ we denote by 1^n the string $11\ldots 1$ of length n. An ordered pair (x, y) with $x, y \in \Sigma^*$ is coded (identified) with a string in Σ^* of length $2 + 2\max\{|x|, |y|\}$. Sometimes statements containing a formulation like "there is a $d \in \mathbb{N}$ such that for all $x \in \Sigma^*$: $\ldots \leq |x|^d$" can be wrong for $x \in \Sigma^*$ with $|x| \leq 1$ (here $|x|$ denotes the length of the string x). We trust the reader's common sense to interpret such statements reasonably.

2.1. Structures and classes of structures. A *vocabulary* τ is a finite set of relation symbols, function symbols, and constant symbols. The universe of a τ-structure $\mathcal{A}$ will be denoted by the corresponding Latin letter A, the cardinality of the set A by $|A|$, and the interpretation of a symbol $s \in \tau$ in $\mathcal{A}$ by $s^{\mathcal{A}}$.

> *All structures in this paper are assumed to be finite and to have $[n] := \{1, 2, \ldots, n\}$ as universe for some $n \in \mathbb{N}$.*

For a structure $\mathcal{A}$ we denote by $\|\mathcal{A}\|$ the size of $\mathcal{A}$, that is, the length of a reasonable encoding of $\mathcal{A}$ by a nonempty string $\mathrm{enc}(\mathcal{A}) \in \Sigma^*$ (e.g., cf. [6]). In particular, we assume that the mappings

$\mathcal{A} \mapsto \operatorname{enc}(\mathcal{A})$ and $\operatorname{enc}(\mathcal{A}) \mapsto \mathcal{A}$ are computable in polynomial time and that for every vocabulary τ there is a polynomial $q_\tau \in \mathbb{N}[X]$ such that $|A| \leq |\operatorname{enc}(\mathcal{A})| \leq q_\tau(|A|)$ for every τ-structure $\mathcal{A}$ (here $|A|$ denotes the cardinality of the set A and $|\operatorname{enc}(\mathcal{A})|$ the length of the string $\operatorname{enc}(\mathcal{A})$).

Clearly, a class C of τ-structures is in P if the set

$$\big\{ \operatorname{enc}(\mathcal{A}) \mid \mathcal{A} \in C \big\}$$

is in P. The class C is *closed under isomorphism* if for all structures $\mathcal{A}$ and $\mathcal{B}$

$$\mathcal{A} \in C \text{ and } \mathcal{A} \cong \mathcal{B} \text{ imply } \mathcal{B} \in C$$

(recall that we restrict to structures with universe $[n]$ for some $n \in \mathbb{N}$).

> *In the rest of the paper C (and D) will always denote a class of structures, all of the same vocabulary τ, which is in* P*, is closed under isomorphism, and contains arbitrarily large (finite) structures.*

Examples of such classes are:

- The classes SET, BOOLE, FIELD, GROUP, ABELIAN, and CYCLIC of sets (structures of empty vocabulary), Boolean algebras, fields, groups, abelian groups, and cyclic groups, respectively.

- The class GRAPH of (undirected and simple) graphs. We view graphs as τ_{GRAPH}-structures, where $\tau_{\text{GRAPH}} := \{E\}$ for a binary relation symbol E.

- The class ORD of linear orderings. Here we use the vocabulary $\tau_{\text{ORD}} := \{<\}$ with a binary relation symbol $<$.

- The class LOP of *L*inear *O*rderings with a distinguished *P*oint and the class LOU of *L*inear *O*rderings with a *U*nary relation. Let $\tau_{\text{LOP}} := \tau_{\text{ORD}} \cup \{c\}$ with a constant symbol c and $\tau_{\text{LOU}} := \tau_{\text{ORD}} \cup \{P\}$ with a unary relation symbol P. Then LOP (LOU) is the class of all τ_{LOP}-structures (τ_{LOU}-structures) $\mathcal{A}$ with $(A, <^\mathcal{A}) \in$ ORD.

Structures in LOU correspond to strings in Σ^* and vice versa; in fact, a structure $\mathcal{A} \in$ LOU corresponds to the string of length $|A|$ whose ith bit is 1 if and only if the ith element of A (according to $<^\mathcal{A}$) is in $P^\mathcal{A}$.

3. Strong isomorphism reductions

We define the notion of strong isomorphism reduction already indicated in the Introduction and present first examples.

Definition 1. Let C and D be classes. We say that C is *strongly isomorphism reducible to D* and write $C \leq_{\text{iso}} D$, if there is a function $f : C \to D$ computable in polynomial time such that for all $\mathcal{A}, \mathcal{B} \in C$

$$\mathcal{A} \cong \mathcal{B} \iff f(\mathcal{A}) \cong f(\mathcal{B}).$$

We then say that f is a *strong isomorphism reduction* from C to D and write $f : C \leq_{\text{iso}} D$. If $C \leq_{\text{iso}} D$ and $D \leq_{\text{iso}} C$, denoted by $C \equiv_{\text{iso}} D$, then C and D *have the same strong isomorphism degree*.

Examples 2. The mapping sending a field to its multiplicative group witnesses that FIELD $\leq_{\text{iso}}$ CYCLIC. Furthermore, ORD $\equiv_{\text{iso}}$ CYCLIC and CYCLIC $\leq_{\text{iso}}$ ABELIAN.

Remark 3. We can reduce the notion of strong isomorphism reduction to a central notion of computational complexity, namely to the notion of polynomial time reduction. For this, we introduce the problem

$$
\begin{array}{ll}
\text{Iso}(C) & \\
\quad \textit{Instance:} & \mathcal{A}, \mathcal{B} \in C. \\
\quad \textit{Problem:} & \text{Is } \mathcal{A} \cong \mathcal{B}?
\end{array}
$$

A function $f : C \to D$ induces the function $\hat{f} : C \times C \to D \times D$ with $\hat{f}(\mathcal{A}, \mathcal{B}) := \big(f(\mathcal{A}), f(\mathcal{B})\big)$. Then

$$f : C \leq_{\text{iso}} D \iff \hat{f} : \text{Iso}(C) \leq_p \text{Iso}(D),$$

where $\hat{f} : \text{Iso}(C) \leq_p \text{Iso}(D)$ means that $\hat{f}$ is a polynomial time reduction from $\text{Iso}(C)$ to $\text{Iso}(D)$.

Of course, it is easy to construct polynomial time reductions from $\text{Iso}(C)$ to $\text{Iso}(D)$ that are not of the form $\hat{f}$ for some $f : C \leq_{\text{iso}} D$. Moreover, in Remark 14 we shall present classes C and D such that

$$\text{Iso}(C) \leq_p \text{Iso}(D) \text{ but not } C \leq_{\text{iso}} D.$$

This answers [7, Open Question 4.13].

As already mentioned in the Introduction one of our goals is to study the relation $\leq_{\text{iso}}$. First we see that this relation has a maximum element:

Proposition 4. $C \leq_{\text{iso}} \text{GRAPH}$ *for all classes C.*

Proof: Let τ be a vocabulary and S be the class of all τ-structures. It is well-known that there is a strong isomorphism reduction from S to GRAPH (even a first-order interpretation, e.g. see [6, Proposition 11.2.5 (i)]). In particular, its restriction to a class C of τ-structures shows that $C \leq_{\text{iso}} \text{GRAPH}$. $\square$

4. Invariantizations and canonizations

One of the central aims of algebra and of model theory is to describe the isomorphism type of a structure by means of an invariant. The underlying notion of invariantization is also relevant in our context. We use it (and the related notion of canonization) to show that most classes of structures mentioned in Section 2.1 have the same strong isomorphism degree (cf. Corollary 12).

Definition 5. An *invariantization for C* is a polynomial time computable function $\text{Inv} : C \to \Sigma^*$ such that for all $\mathcal{A}, \mathcal{B} \in C$

$$\mathcal{A} \cong \mathcal{B} \iff \text{Inv}(\mathcal{A}) = \text{Inv}(\mathcal{B}).$$

Lemma 6. *If $C \leq_{\text{iso}} D$ and D has an invariantization, then also C has an invariantization.*

Proof: If Inv is an invariantization for D and $f : C \leq_{\text{iso}} D$, then $\text{Inv} \circ f$ is an invariantization for C. $\square$

LOU is a maximum class among those with an invariantization:

Proposition 7. *For a class C the following are equivalent.*

(1) C has an invariantization.

(2) $C \leq_{\text{iso}} \text{LOU}$.

4

(3) There is a class D of ordered structures such that $C \leq_{\mathrm{iso}} D$.

Here, a class D is a class of ordered structures *if its vocabulary contains a binary relation symbol which in all structures of D is interpreted as a linear ordering of the universe.*

Proof: (1) implies (2) by the natural correspondence between strings in Σ^* and structures in LOU. That (2) implies (3) is trivial. To see that (3) implies (1) assume that there is a class D of ordered structures such that $C \leq_{\mathrm{iso}} D$. As ordered structures have no nontrivial automorphisms, every ordered structure $\mathcal{A}$ is isomorphic to a unique structure $\mathcal{A}'$ with universe $\{1, 2, \ldots, |A|\}$ and with its natural linear ordering on it. Thus the mapping on D defined by $\mathcal{A} \mapsto \mathrm{enc}(\mathcal{A}')$ is an invariantization of D (recall that $\mathrm{enc}(\mathcal{B})$ is the encoding of $\mathcal{B}$ by a string in Σ^*). Now we apply Lemma 6. $\qquad\square$

It is open whether the class GRAPH has an invariantization or equivalently (by Proposition 4 and Proposition 7) whether LOU is a maximum element of $\leq_{\mathrm{iso}}$. Moreover, it is known [11, 13] that an invariantization for GRAPH yields a canonization.

Definition 8. A function $\mathrm{Can} : C \to C$ is a *canonization for C* if it is polynomial time computable and

(1) for all $\mathcal{A}, \mathcal{B} \in C$: $\quad \big(\mathcal{A} \cong \mathcal{B} \iff \mathrm{Can}(\mathcal{A}) = \mathrm{Can}(\mathcal{B})\big)$;

(2) for all $\mathcal{A} \in C$: $\mathcal{A} \cong \mathrm{Can}(\mathcal{A})$.

Every class C of ordered structures, in particular LOU, has a canonization. In fact, the mapping $\mathcal{A} \mapsto \mathcal{A}'$ defined for all ordered structures in the previous proof is a canonization for C.

We do not define the notion of a *logic capturing* P *on a class C* (e.g. see [6]). However we mention that canonizations and invariantizations are important in descriptive complexity theory as:

Proposition 9. *(1) If C has a canonization, then there is a logic capturing* P *on C.*

(2) If GRAPH *has an invariantization, then there is a logic capturing* P *(on all finite structures).*

If a class C has a canonization, then it also has an invariantization. In fact, by property (1) of Definition 8, if $\mathrm{Can} : C \to C$ is a canonization, then $\mathcal{A} \to \mathrm{enc}(\mathrm{Can}(\mathcal{A}))$ is an invariantization. Often the invariantizations we encounter in mathematics yield canonizations. For example, consider the class FIELD of fields. Then an invariant for a field $\mathcal{K}$ is the pair $(p_\mathcal{K}, n_\mathcal{K})$, where $p_\mathcal{K}$ is its characteristic and $n_\mathcal{K}$ its dimension over the prime field. As for every invariant (p, n) one can explicitly construct a canonical field $\mathcal{F}_{p^n}$ of this invariant, we see that the mapping $\mathcal{K} \mapsto \mathcal{F}_{p_\mathcal{K}^{n_\mathcal{K}}}$ is a canonization. This canonization has a further property, it is a canonization that has a polynomial time enumeration:

Definition 10. Let Can be a canonization for the class C. The *enumeration induced by* Can is the enumeration

$$\mathcal{A}_1, \mathcal{A}_2, \ldots$$

of $\mathrm{Can}(C)$ such that $\mathrm{enc}(\mathcal{A}_i) <_{\mathrm{lex}} \mathrm{enc}(\mathcal{A}_j)$[1] for $i < j$. If the mappings $\mathcal{A}_n \mapsto 1^n$ and $1^n \mapsto \mathcal{A}_n$ are computable in polynomial time, then Can *has a polynomial time enumeration*.

Note that the mapping $\mathcal{A}_n \mapsto 1^n$ is computable in polynomial time if and only if we get an invariantization Inv of C by setting

$$\mathrm{Inv}(\mathcal{A}) := 1^n \iff \mathrm{Can}(\mathcal{A}) = \mathcal{A}_n.$$

---

[1] By $<_{\mathrm{lex}}$ we denote the standard (length-)lexicographic ordering on Σ^*.

Moreover, if the mapping $\mathcal{A}_n \mapsto 1^n$ is computable in polynomial time, then

$$\{\mathrm{enc}(\mathrm{Can}(\mathcal{A})) \mid \mathcal{A} \in C\}$$

is a sparse set.

The classes SET, FIELD, ABELIAN, CYCLIC, ORD, and LOP have canonizations with polynomial time enumerations (for ABELIAN see [12], for example). The classes BOOLE and LOU have canonizations but none with a polynomial time enumeration (for BOOLE the function $1^n \mapsto \mathcal{A}_n$ will not be computable in polynomial time, as there are, up to equivalence, "too few" Boolean algebras of cardinality $\leq n$, namely $\lfloor \log n \rfloor$; for LOU the function $\mathcal{A}_n \mapsto 1^n$ won't be computable in polynomial time, as there are "too many" structures in LOU of cardinality $\leq n$, namely $2^{n+1} - 1$).

Theorem 11. *Assume that the classes C and D have canonizations with polynomial time enumerations. Then*

$$C \equiv_{\mathrm{iso}} D.$$

Corollary 12. *The classes* SET*,* FIELD*,* ABELIAN*,* CYCLIC*,*ORD*, and* LOP *all have the same strong isomorphism degree.*

Proof of Theorem 11: Let C and D be classes with canonizations Can_C and Can_D which have polynomial time enumerations

$$\mathcal{A}_1, \mathcal{A}_2, \ldots$$

and

$$\mathcal{B}_1, \mathcal{B}_2, \ldots$$

respectively. We define a strong isomorphism reduction f from C to D by:

$$f(\mathcal{A}) = \mathcal{B}_n \iff \mathrm{Can}_C(\mathcal{A}) = \mathcal{A}_n.$$

Hence, $C \leq_{\mathrm{iso}} D$; by symmetry we get $D \leq_{\mathrm{iso}} C$. $\qquad\qquad\square$

An analysis of the previous proof shows that we already obtain $C \leq_{\mathrm{iso}} D$ if the mappings $\mathcal{A}_n \mapsto 1^n$ and $1^n \mapsto \mathcal{B}_n$ are computable in polynomial time. By this, we get, for example, BOOLE$\leq_{\mathrm{iso}}$ CYCLIC.

5. On $\leq_{\mathbf{iso}}$ below LOP

As we have seen that the structure of $\leq_{\mathrm{iso}}$ between LOU and GRAPH is linked with central open problems of descriptive complexity, we turn our attention to the structure below LOU even below LOP. In this section we show that there the structure is quite rich. In fact, this section is devoted to a proof of the following result:[2]

Theorem 13. *The partial ordering of the countable atomless Boolean algebra is embeddable into the partial ordering induced by $\leq_{\mathrm{iso}}$ on the degrees of strong isomorphism reducibility below* LOP*. More precisely, let $\mathcal{B}$ be a countable atomless Boolean algebra. Then there is a one-to-one function $b \mapsto C_b$ defined on B such that for all $b, b' \in B$*

- *C_b is a subclass of* LOP*;*
- *$b \leq b' \iff C_b \leq_{\mathrm{iso}} C_{b'}$.*

---

[2]Recall that up to isomorphism there is a unique countable atomless Boolean algebra (e.g. see [10]).

Recall that the partial ordering of an atomless Boolean algebra has infinite antichains and infinite chains, even chains of ordertype the rationals.

Remark 14. By the preceding result, we see that there exist an infinite $\leq_{\text{iso}}$-antichain of classes C below LOP, whose problems $\text{ISO}(C)$ are pairwise equivalent under usual polynomial time reductions. Indeed, even $\text{ISO}(C) \in P$ for all $C \subseteq \text{LOP}$.

The reader not interested in the details of the proof of Theorem 13 should read till Lemma 17 and can then skip the rest of this section. We obtain Theorem 13 by comparing the number of isomorphism types of structures with universe of bounded cardinality in different classes. First we introduce the relevant notations and concepts.

For a class C we let $C(n)$ be the subclass consisting of all structures in C with universe of cardinality $\leq n$ and we let $\#C(n)$ be the number of isomorphism types of structures in $C(n)$, more formally

$$C(n) := \{\mathcal{A} \in C \mid |A| \leq n\} \quad \text{and} \quad \#C(n) := |C(n)/_{\cong}|$$

Here, for a class of structures S we denote by $S/_{\cong}$ the set of isomorphism classes in S.

Examples 15. (1) $\#\text{BOOLE}(n) = \lfloor \log n \rfloor$ and $\#\text{CYCLIC}(n) = n$.

(2) $\#\text{LOP}(n) = \sum_{i=1}^{n} i = (n+1) \cdot n/2$ and $\#\text{LOU}(n) = \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$.

(3) For every vocabulary τ there is a polynomial $p_\tau \in \mathbb{N}[X]$ such that $\#C(n) \leq 2^{p_\tau(n)}$ for all $n \in \mathbb{N}$.

(4) (e.g. see [1]) $\#\text{GROUP}(n)$ is superpolynomial but subexponential (more precisely, $\#\text{GROUP}(n) \leq n^{O(\log^2 n)}$).

Definition 16. A class C is *potentially reducible* to a class D, written $C\leq_{\text{pot}} D$, if there is some polynomial $p \in \mathbb{N}[X]$ such that $\#C(n) \leq \#D(p(n))$ for all $n \in \mathbb{N}$. Of course, by $C \equiv_{\text{pot}} D$ we mean $C\leq_{\text{pot}} D$ and $D\leq_{\text{pot}} C$.

The following lemma explains the term potentially reducible.

Lemma 17. *If* $C\leq_{\text{iso}} D$, *then* $C\leq_{\text{pot}} D$.

Proof: Let $f : C\leq_{\text{iso}} D$. As f is computable in polynomial time, there is a polynomial p such that for all $\mathcal{A} \in C$ we have $|f(A)| \leq p(|A|)$, where $f(A)$ denotes the universe of $f(\mathcal{A})$. As f strongly preserves isomorphisms, it therefore induces a one-to-one map from $\{\mathcal{A} \in C \mid |A| \leq n\}/_{\cong}$ to $\{\mathcal{B} \in D \mid |B| \leq p(n)\}/_{\cong}$. $\square$

We state some consequences of this simple observation:

Corollary 18. *(1)* CYCLIC $\not\leq_{\text{iso}}$ BOOLE *and* LOU $\not\leq_{\text{iso}}$ LOP.

(2) $C\leq_{\text{pot}}$ LOU *for all classes C and* LOU $\equiv_{\text{pot}}$ GRAPH.

(3) *The strong isomorphism degree of* GROUP *is strictly between that of* LOP *and* GRAPH, *that is,*

$$\text{LOP}\leq_{\text{iso}} \text{GROUP}\leq_{\text{iso}} \text{GRAPH}, \quad but \quad \text{LOP} \not\equiv_{\text{iso}} \text{GROUP} \text{ and } \text{GROUP} \not\equiv_{\text{iso}} \text{GRAPH}.$$

(4) *The potential reducibility degree of* GROUP *is strictly between that of* LOP *and* LOU, *that is,*

$$\text{LOP}\leq_{\text{pot}} \text{GROUP}\leq_{\text{pot}} \text{LOU}, \quad but \quad \text{LOP} \not\equiv_{\text{pot}} \text{GROUP} \text{ and } \text{GROUP} \not\equiv_{\text{pot}} \text{LOU}.$$

Proof: Using the previous lemma we see that

- (1) follows by Examples 15 (1), (2);

- (2) from Examples 15 (2), (3) and Proposition 4;

- GROUP$\leq_{\mathrm{iso}}$ GRAPH holds by Lemma 4 and LOP$\leq_{\mathrm{iso}}$ CYCLIC$\leq_{\mathrm{iso}}$ GROUP by Corollary 12; the remaining claims in (3) follow from (4) as LOU $\equiv_{\mathrm{pot}}$ GRAPH;

- the first claim follows from the first claim in (3) as LOU $\equiv_{\mathrm{pot}}$ GRAPH; the remaining claims follow from Examples 15 (2), (4).

$\square$

The following concepts and tools will be used in the proof of Theorem 13. We call a function $f : \mathbb{N} \to \mathbb{N}$ *value-polynomial* if it is increasing and $f(n)$ can be computed in time $f(n)^{O(1)}$. Let VP be the class of all value-polynomial functions.

Lemma 19. *The images of the functions in* VP *and the finite subsets of* $\mathbb{N}$ *are the elements of a countable Boolean algebra* $\mathcal{V}$ *(under the usual set-theoretic operations). The factor algebra* $\mathcal{V}/_{\equiv}$, *where for* $b, b' \in V$

$$b \equiv b' \iff (b \setminus b') \cup (b' \setminus b) \text{ is finite},$$

is a countable atomless Boolean algebra.

Proof: For a function $f : \mathbb{N} \to \mathbb{N}$ we denote by $\mathrm{im}(f)$ the image of f. Using the definition of value-polynomial function it is easy to verify that for $f, g \in$ VP the sets

$$\mathbb{N} \setminus \mathrm{im}(f), \quad \mathrm{im}(f) \cap \mathrm{im}(g), \quad \text{and} \quad \mathrm{im}(f) \cup \mathrm{im}(g)$$

are images of value-polynomial functions provided they are infinite. We leave the rest of the proof to the reader. $\square$

For $f \in$ VP the set

$$C_f := \big\{ \mathcal{A} \in \mathrm{LOP} \mid |A| \in \mathrm{im}(f) \big\}$$

is in P and is closed under isomorphism. As in cardinality $f(k)$ there are exactly $f(k)$ pairwise nonisomorphic structures in LOP, we get

$$\#C_f(n) = \sum_{k \,\in\, \mathbb{N} \text{ with } f(k) \,\leq\, n} f(k).$$

The following observation contains an essential idea used in the proof of Theorem 13.

Proposition 20. *Let* $f \in$ VP *and assume that for every polynomial* $p \in \mathbb{N}[X]$ *there is* $n \in \mathbb{N}$ *such that*

$$\sum_{k \,\in\, \mathbb{N} \text{ with } f(2k) \,\leq\, n} f(2k) > \sum_{k \,\in\, \mathbb{N} \text{ with } f(2k+1) \,\leq\, p(n)} f(2k+1). \tag{2}$$

Then C_{g_0} *is not potentially reducible to* C_{g_1}, *where* $g_0, g_1 : \mathbb{N} \to \mathbb{N}$ *are defined by* $g_0(n) := f(2n)$ *and* $g_1(n) := f(2n+1)$.

Proof: By contradiction, assume that there is some polynomial $p \in \mathbb{N}[X]$ such that $\#C_{g_0}(n) \leq \#C_{g_1}(p(n))$ for all $n \in \mathbb{N}$. Choose n such that (2) holds. Then

$$\#C_{g_0}(n) = \sum_{f(2k) \leq n} f(2k) > \sum_{f(2k+1) \leq p(n)} f(2k+1) = \#C_{g_1}(p(n)),$$

a contradiction. □

The reader will easily verify that the following function $h : \mathbb{N} \to \mathbb{N}$ defined by recursion is value-polynomial:

$$\begin{aligned} h(0) &= 0, \\ h(n+1) &= (h(0) + \cdots + h(n))^n. \end{aligned}$$

For $f, g \in \text{VP}$ set

$$f \subseteq^* g \iff \text{im}(f) \setminus \text{im}(g) \text{ is finite.}$$

By the homogeneity properties of atomless countable Boolean algebras, to prove Theorem 13 it suffices to find a corresponding embedding defined only on the nonzero elements of $\mathcal{V}/_\equiv$. In general $f \subseteq^* g$ and $g \subseteq^* f$ do not imply $C_{h \circ f} = C_{h \circ g}$. However, by the following lemma we get an embedding of $\mathcal{V}/_\equiv$ into the partial ordering of the $\leq_{\text{iso}}$-degrees as required by Theorem 13 by defining the mapping on a set of representatives, more precisely on a set $R \subseteq \text{VP}$ such that

– for every $f \in \text{VP}$ there is exactly one $g \in R$ with $f \subseteq^* g$ and $g \subseteq^* f$.

Lemma 21. *The mapping $f \mapsto C_{h \circ f}$ from $(\text{VP}, \subseteq^*)$ to $(\{C \subseteq \text{LOU} \mid C \text{ a class}\}, \leq_{\text{iso}})$ is one-to-one and for all $f, g \in \text{VP}$:*

(1) if $C_{h \circ f} \leq_{\text{iso}} C_{h \circ g}$, then $f \subseteq^* g$;

(2) if $f \subseteq^* g$ and $g \not\subseteq^* f$, then $C_{h \circ f} \leq_{\text{iso}} C_{h \circ g}$.

For the proof of the Lemma 21 we need an appropriate way to invert increasing functions $f : \mathbb{N} \to \mathbb{N}$. We define $f^{-1} : \mathbb{N} \to \mathbb{N}$ by

$$f^{-1}(n) := \max\{i \mid f(i) \leq n\},$$

where we set $\max \emptyset := 0$. We collect some properties of this inverse in the following lemma, whose simple proof we omit. We denote by $\text{id}_\mathbb{N}$ the identity function on $\mathbb{N}$.

Lemma 22. *(1) If $f : \mathbb{N} \to \mathbb{N}$ is increasing, then f^{-1} is nondecreasing, $f^{-1} \leq \text{id}_\mathbb{N}$, $f^{-1} \circ f = \text{id}_\mathbb{N}$ and $f(f^{-1}(n)) \leq n$ for all $n \geq f(0)$.*

(2) If $f, g : \mathbb{N} \to \mathbb{N}$ are increasing, then $(f \circ g)^{-1} = g^{-1} \circ f^{-1}$.

(3) If $f \in \text{VP}$, then f^{-1} is computable in polynomial time.

A further notation is useful: For $f : \mathbb{N} \to \mathbb{N}$ let $f^\Sigma : \mathbb{N} \to \mathbb{N}$ be defined by

$$f^\Sigma(n) := \sum_{i \leq n} f(i).$$

Lemma 23. *Let $f, g : \mathbb{N} \to \mathbb{N}$ be functions and assume g is increasing. Then $(f \circ g)^\Sigma \leq f^\Sigma \circ g$.*

Proof: This is seen by direct calculation:

$$(f \circ g)^\Sigma(n) = \sum_{i \leq n} f(g(i)) = \sum_{\substack{i \leq g(n) \\ i \in \mathrm{im}(g)}} f(i) \leq \sum_{i \leq g(n)} f(i) = f^\Sigma \circ g(n);$$

here the second equality uses that g is increasing. $\square$

Furthermore observe that:

Lemma 24. *If $f \in$ VP, then for all $n \in \mathbb{N}$ we have $\#C_f(n) = (f^\Sigma \circ f^{-1})(n)$.*

Proof of Lemma 21: The mapping $f \mapsto C_{h \circ f}$ is one-to-one: Assume $C_{h \circ f} = C_{h \circ g}$. Then $\mathrm{im}(h \circ f) = \mathrm{im}(h \circ g)$ and thus, $\mathrm{im}(f) = \mathrm{im}(g)$ as h is one-to-one. Since f and g are both increasing, this yields $f = g$. We prove the remaining statements of Lemma 21 by the following two claims.

Claim 1: Let $f, g \in$ VP and $f \subseteq^* g$ and $g \not\subseteq^* f$. Then $C_{h \circ f} \leq_{\mathrm{iso}} C_{h \circ g}$.

Proof of Claim 1: By our assumptions, the set $\mathrm{im}(h \circ f) \setminus \mathrm{im}(h \circ g)$ is finite (as $f \subseteq^* g$ implies $h \circ f \subseteq^* h \circ g$) and (by injectivity of h) the set $\mathrm{im}(h \circ g) \setminus \mathrm{im}(h \circ f)$ is infinite. Then $C_{h \circ f} \leq_{\mathrm{iso}} C_{h \circ g}$ is witnessed by a function sending the (up to $\cong$) finitely many structures in $C_{h \circ f} \setminus C_{h \circ g}$ to $C_{h \circ g} \setminus C_{h \circ f}$ and which is the identity on all other structures in $C_{h \circ f}$.

Claim 2: Let $f, g \in$ VP and $f \not\subseteq^* g$. Then $C_{h \circ f} \not\leq_{\mathrm{iso}} C_{h \circ g}$.

Proof of Claim 2: By contradiction assume $C_{h \circ f} \leq_{\mathrm{iso}} C_{h \circ g}$. Then $C_{h \circ f}$ is potentially reducible to $C_{h \circ g}$ by Lemma 17. Hence there is $p \in \mathbb{N}[X]$ such that $\#C_{h \circ f}(n) \leq \#C_{h \circ g}(p(n))$ for all $n \in \mathbb{N}$. We show that this is wrong for some n. For this purpose we choose k such that

$$g(0) < f(k), \quad p(h(f(k))) < h(f(k) + 1), \quad \text{and} \quad f(k) \in \mathrm{im}(f) \setminus \mathrm{im}(g) \tag{3}$$

(by the definition of h and the assumption $f \not\subseteq^* g$ such a k exists). Then we get

$\#C_{h \circ g}(p(h(f(k))))$

$\quad = (h \circ g)^\Sigma \circ (h \circ g)^{-1}(p(h(f(k))))$ (by Lemma 24)

$\quad = (h \circ g)^\Sigma \circ (g^{-1} \circ h^{-1})(p(h(f(k))))$ (by Lemma 22(2))

$\quad \leq (h \circ g)^\Sigma \circ g^{-1}(f(k))$ (by $p(h(f(k))) < h(f(k) + 1)$ (see (3)) and by definition of h^{-1})

$\quad = (h \circ g)^\Sigma \circ g^{-1}(f(k) - 1)$ (as $f(k) \notin \mathrm{im}(g)$)

$\quad \leq h^\Sigma \circ g \circ g^{-1}(f(k) - 1)$ (by Lemma 23)

$\quad \leq h^\Sigma(f(k) - 1)$ (by Lemma 22(1) as $g(0) < f(k)$)

$\quad < h(f(k))$ (by definition of h)

$\quad \leq \#C_{h \circ f}(h(f(k)))$ (by definition of $\#C_{h \circ f}$).

$\square$

6. Strong isomorphism reducibility and potential reducibility

We know that GRAPH$\leq_{\text{pot}}$ LOU (cf. Corollary 18 (2)) while GRAPH$\leq_{\text{iso}}$ LOU is equivalent to GRAPH having an invariantization (cf. Proposition 7). However, so far in all concrete examples of classes C and D, for which we know the status of $C\leq_{\text{iso}} D$ and of $C\leq_{\text{pot}} D$, we had that

$$C\leq_{\text{iso}} D \iff C\leq_{\text{pot}} D.$$

So the question arises whether the relations of strong isomorphism reducibility and of potential reducibility coincide. Recall that we require the classes C and D to be closed under isomorphism and decidable in polynomial time. Generalizing the proof idea of Theorem 11, we shall see in the next section that indeed the relations $\leq_{\text{iso}}$ and $\leq_{\text{pot}}$ coincide if P $=$ #P. We believe that they are distinct but could only show:

Theorem 25. *If* U2EXP $\cap$ co-U2EXP $\neq$ 2EXP, *then the relations of strong isomorphism reducibility and that of potential reducibility are distinct.*

Recall that

$$2\text{EXP} := \text{DTIME}\left(2^{2^{n^{O(1)}}}\right) \quad \text{and} \quad \text{N2EXP} := \text{NTIME}\left(2^{2^{n^{O(1)}}}\right)$$

The complexity class U2EXP consists of those $Q \in$ N2EXP for which there is a nondeterministic Turing machine of type N2EXP that for every $x \in Q$ has exactly one accepting run. Finally, co-U2EXP $:= \{\Sigma^* \setminus Q \mid Q \in \text{U2EXP}\}$.

The rest of this section is devoted to a proof of this result. We explain the underlying idea: Assume $Q \in$ U2EXP $\cap$ co-U2EXP. We construct classes C and D which contain structures in the same cardinalities and which contain exactly two nonisomorphic structures in these cardinalities. Therefore they are potentially reducible to each other. While it is trivial to exhibit two nonisomorphic structures in C of the same cardinality, from any two concrete nonisomorphic structures in D we obtain information on membership in Q for all strings of a certain length. If $C\leq_{\text{iso}} D$, we get concrete nonisomorphic structures in D (in time allowed by 2EXP) by applying the strong isomorphism reduction to two nonisomorphic structures in C and therefore obtain $Q \in$ 2EXP.

Proof of Theorem 25: Let $Q \in$ U2EXP $\cap$ co-U2EXP. Then there exists a nondeterministic Turing machine $\mathbb{M}$ and a constant $d \geq 2$ such that (M1)–(M5) hold:

(M1) The machine $\mathbb{M}$ has three terminal states *'yes,' 'no,'* and *'maybe.'*

(M2) For $x \in \Sigma^*$, every run of $\mathbb{M}$ on input x stops after *exactly* $2^{2^{|x|^d}}$ many steps.

(M3) For $x \in Q$ exctly one run of $\mathbb{M}$ on x stops in 'yes' and none in 'no.'

(M4) For $x \notin Q$ exactly one run of $\mathbb{M}$ on x stops in 'no' and none in 'yes.'

(M5) The machine $\mathbb{M}$ has exactly two different choices for the next step in every nonterminal state.

We say that a run of $\mathbb{M}$ *takes a decision* if it ends in 'yes' or in 'no.'

For $n \in \mathbb{N}$ we set $\ell(n) := 2^{2^{n^d}}$. For $x \in \Sigma^n$, by (M2) and (M5), every run of $\mathbb{M}$ on input x can be identified with a binary string $r \in \{0,1\}^{\ell(n)}$. Conversely, from such a string r we can determine a run of $\mathbb{M}$ on x.

Let $m(n) := 2^n$ and
$$x_1, x_2, \ldots, x_{m(n)}$$
be the enumeration of all strings of Σ^n in the lexicographic ordering. We call a binary string s of length $m(n) \cdot \ell(n) = 2^n \cdot 2^{2^{n^d}}$ a *decision string* if for $i \in [m(n)]$ the ith substring of s of length $\ell(n)$ corresponds to a run of $\mathbb{M}$ on x_i taking a decision; more precisely, if we have $s = s_1\hat{\ }s_2\hat{\ }\cdots\hat{\ }s_{m(n)}$ with $|s_i| = \ell(n)$ for $i \in [m(n)]$, then s_i corresponds to a run of $\mathbb{M}$ on x_i taking a decision. By our assumptions (M3) and (M4) we get:

$$\text{for every } n \in \mathbb{N} \text{ there is exactly one decision string of length } m(n) \cdot \ell(n). \qquad (4)$$

We turn every string s of length $m(n) \cdot \ell(n)$ into a structure $\mathcal{A}(s)$ over the vocabulary $\tau = \{P, R\}$, where P is a unary relation symbol and R is a binary one. Let

$$A(s) := [m(n) \cdot \ell(n)],$$
$$R^{\mathcal{A}(s)} := \{(j, j+1) \mid j \in [m(n) \cdot \ell(n) - 1]\},$$
$$P^{\mathcal{A}(s)} := \begin{cases} \{j \mid j \in [m(n) \cdot \ell(n)] \text{ and the } j\text{th bit of } s \text{ is one}\}, & \text{if } s \text{ is a decision string} \\ \emptyset, & \text{otherwise.} \end{cases}$$

By (4) for every $s, s' \in \{0, 1\}^{m(n) \cdot \ell(n)}$

$$\mathcal{A}(s) \not\cong \mathcal{A}(s') \iff \text{exactly one of } s \text{ and } s' \text{ is a decision string.} \qquad (5)$$

Let D_n be the class containing, up to isomorphism, the structures $\mathcal{A}(s)$ with $s \in \{0, 1\}^{m(n) \cdot \ell(n)}$. The following is straightforward.

(D1) The universe of every structure in D_n has cardinality $m(n) \cdot \ell(n)$.

(D2) $|D_n/_\cong| = 2$.

We set

$$D := \bigcup_{n \in \mathbb{N}} D_n.$$

Finally, we let

$$C := \bigcup_{n \in \mathbb{N}} C_n,$$

where for $n \in \mathbb{N}$ every structure in the class C_n is isomorphic to the complete graph $K_{m(n) \cdot \ell(n)}$ on $m(n) \cdot \ell(n)$ vertices or to its complement $\bar{K}_{m(n) \cdot \ell(n)}$. Then:

(C1) The universe of every structure in C_n has cardinality $m(n) \cdot \ell(n)$.

(C2) $|C_n/_\cong| = 2$.

Hence, $C \leq_{\mathrm{pot}} D$.

Claim: Assume $f : C \leq_{\mathrm{iso}} D$. Then there is $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$

$$f(C_n/_\cong) = D_n/_\cong. \qquad (6)$$

By this equality we mean:

– $f(\mathcal{A}) \in D_n$ for every $\mathcal{A} \in C_n$;

– for every $\mathcal{B} \in D_n$ there exists an $\mathcal{A} \in C_n$ such that $f(\mathcal{A}) \cong \mathcal{B}$.

Proof of the Claim: First observe that by (C2) and (D2) it suffices to show for all sufficiently large $n \in \mathbb{N}$

$$f(C_n) \subseteq D_n. \tag{7}$$

As f is computable in polynomial time there is $c \in \mathbb{N}$ such that for every $n \in \mathbb{N}$ and $\mathcal{A} \in C_n$

the universe of $f(\mathcal{A})$ has $\leq \left(2^n \cdot 2^{2^{n^d}}\right)^c$ elements.

We choose $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$

$$\left(2^n \cdot 2^{2^{n^d}}\right)^c < 2^{n+1} \cdot 2^{2^{(n+1)^d}}.$$

Hence, for $n \geq n_0$

$$f\left(\bigcup_{q \leq n} C_q\right) \subseteq \bigcup_{q \leq n} D_q.$$

As $\bigcup_{q \leq n} C_q$ and $\bigcup_{q \leq n} D_q$ contain, up to isomorphism, the same number of structures the Claim follows. $\dashv$

Now assume that $f : C \leq_{\mathrm{iso}} D$. Then the following algorithm $\mathbb{A}$ witnesses that $Q \in 2\mathrm{EXP}$. Let n_0 be as in the Claim. For $x \in \Sigma^n$ with $n \geq n_0$ the algorithm $\mathbb{A}$ computes the structures

$$f\left(K_{m(n) \cdot \ell(n)}\right) \quad \text{and} \quad f\left(\bar{K}_{m(n) \cdot \ell(n)}\right);$$

they are nonisomorphic and in D_n by the Claim. In particular, by (5) we get a run of $\mathbb{M}$ on input x taking a decision; the algorithm $\mathbb{A}$ answers accordingly. $\square$

7. Strong isomorphism reducibility and potential reducibility coincide under P = #P

In the previous section we have seen that under some complexity-theoretic assumption the two notions of reduction (strong isomorphism reducibility and potential reducibility) are distinct. One might wonder whether we can separate them without any such complexity-theoretic assumption. We show in this section that this would settle some open problem in complexity theory; more precisely, we show the statement of the title of this section.[3] In particular, by Corollary 18 (2), the assumption P = #P implies that LOU is a maximum element of $\leq_{\mathrm{iso}}$. We prove the result in a more general setting.

For a class C consider the equivalence relation $E(C)$ on Σ^* induced by the isomorphism relation, that is,

$$E(C) := \big\{(\mathrm{enc}(\mathcal{A}), \mathrm{enc}(\mathcal{B})) \mid \mathcal{A}, \mathcal{B} \in C \text{ and } \mathcal{A} \cong \mathcal{B}\big\}$$
$$\cup \big\{(x, y) \mid x, y \in \Sigma^* \text{ and } x \text{ and } y \text{ are not encodings of structures in } C\big\}.$$

---

[3]Recall that P = #P means that for every polynomial time nondeterministic Turing machine $\mathbb{M}$ the function $f_{\mathbb{M}}$ such that $f_{\mathbb{M}}(x)$ is the number of accepting runs of $\mathbb{M}$ on $x \in \Sigma^*$ is computable in polynomial time. The class #P consists of all the functions $f_{\mathbb{M}}$.

Of course, $E(C)$ is in NP. In this section we consider arbitrary such equivalence relations on Σ^* and show that the corresponding two notions of reduction coincide if P $=$ #P. We start by introducing all relevant concepts; we do not restrict ourselves to equivalence relations in NP, but consider equivalence relations in an arbitrary complexity class (for an equivalence relation E on Σ^* we also write xEy for $(x, y) \in E$).

Definition 26. (1) Let CC be an arbitrary complexity class. Then we denote by CC(eq) the set of equivalence relations E on Σ^* with $E \in$ CC.

(2) Let E and E' be equivalence relations on Σ^*. We say that E is *strongly equivalence reducible to* E' and write $E \leq_{\mathrm{eq}} E'$, if there is a function $f : \Sigma^* \to \Sigma^*$ computable in polynomial time such that for all $x, y \in \Sigma^*$

$$xEy \iff f(x)E'f(y).$$

We then say that f is a *strong equivalence reduction* from E to E' and write $f : E \leq_{\mathrm{eq}} E'$.

Clearly, $E(C) \in$ NP(eq) for every class C of structures; furthermore, $E(\textsc{Lou}) \in$ P(eq). Let Prop and Taut denote the set of all formulas of propositional logic and the set of tautologies, respectively. Note that $E_{\mathrm{equiv}} \in$ co-NP(eq), where

$$E_{\mathrm{equiv}} := \{(\alpha, \beta) \mid \alpha, \beta \in \textsc{Prop} \text{ and } (\alpha \leftrightarrow \beta) \in \textsc{Taut}\} \cup \{(x, y) \mid x, y \in \Sigma^* \setminus \textsc{Prop}\}.$$

Clearly, if C and D are classes of structures as in the previous sections, then

$$C \leq_{\mathrm{iso}} D \iff E(C) \leq_{\mathrm{eq}} E(D).$$

We generalize the notion of potential reducibility to equivalence relations.

Definition 27. Let E and E' be equivalence relations on Σ^*. We say that E is *potentially reducible to* E' and write $E \leq_{\mathrm{pot}} E'$ if there is a $p \in \mathbb{N}[X]$ such that for all $n \in \mathbb{N}$ the number $|\Sigma^{\leq n}/E|$ of E-equivalence classes containing a string in $\Sigma^{\leq n}$ is at most $\left|\Sigma^{\leq p(n)}/E'\right|$.

Due to our definition of $E(C)$, the new notion coincides with the old one for equivalence relations of the form $E(C)$:

Proposition 28. *Let C and C' be classes. Then*

$$C \leq_{\mathrm{pot}} C' \iff E(C) \leq_{\mathrm{pot}} E(C').$$

Proof: Recall that the empty string is not the encoding of a structure. Let C be a class of τ-structures and C' a class of τ'-structures. By our assumption on the encoding of structures, there are polynomials $p_\tau, p_{\tau'} \in \mathbb{N}[X]$ such that for every τ-structure $\mathcal{A}$

$$|A| \leq |\mathrm{enc}(\mathcal{A})| \leq p_\tau(|A|)$$

and for every τ'-structure $\mathcal{B}$

$$|B| \leq |\mathrm{enc}(\mathcal{B})| \leq p_{\tau'}(|B|).$$

Assume first that $C \leq_{\mathrm{pot}} C'$, say $\#C(n) \leq \#C'(p(n))$ for some polynomial p. Then

$$|\Sigma^{\leq n}/E(C)| \leq \#C(n) + 1 \leq \#C'(p(n)) + 1 \leq |\Sigma^{\leq p_{\tau'}(p(n))}/E(C')|.$$

14

Conversely, assume that $E(C) \leq_{\text{pot}} E(C')$, say $\left|\Sigma^{\leq n}/E(C)\right| \leq \left|\Sigma^{\leq p(n)}/E(C')\right|$ with $p \in \mathbb{N}[X]$. Then

$$\#C(n) + 1 \leq \left|\Sigma^{\leq p_\tau(n)}/E(C)\right| \leq \left|\Sigma^{\leq p(p_\tau(n))}/E(C')\right| \leq \#C'(p(p_\tau(n))) + 1. \qquad \square$$

Along the lines of the proof of Lemma 17, one shows that $E \leq_{\text{eq}} E'$ implies $E \leq_{\text{pot}} E'$. For equivalence relations we can show that $\leq_{\text{eq}}$ is finer than $\leq_{\text{pot}}$ under weaker assumptions than that of Theorem 25:

Proposition 29. *If* $\text{NP} \neq \text{P}$*, then the relations of strong equivalence reduction and that of potential reducibility do not coincide on* NP(eq)*.*

Proof: Assume $Q \in \text{NP} \setminus \text{P}$. We define E_Q by

$$x E_Q y \quad \Longleftrightarrow \quad \Big(x = y \text{ or } \big(x = b\hat{\ }z \text{ and } y = (1 - b)\hat{\ }z \text{ for some } z \in Q \text{ and } b \in \Sigma \big) \Big).$$

By our assumptions on Q, we have $E_Q \in \text{NP(eq)}$. We let E be the identity on Σ^*. Clearly, $E_Q \leq_{\text{pot}} E$. As $Q \notin \text{P}$, we get $E_Q \not\leq_{\text{eq}} E$, as any $f : E_Q \leq_{\text{eq}} E$ would yield a polynomial time decision procedure for Q. $\qquad \square$

Generalizing the proof idea of Theorem 11 we show:

Theorem 30. *If* $\text{P} = \#\text{P}$*, then* $\leq_{\text{eq}} = \leq_{\text{pot}}$ *on* P(eq)*, that is, the relations of strong equivalence reducibility and that of potentially reducibility coincide on* P(eq)*.*

To prove this theorem we first generalize the notions of canonization and of enumeration induced by a canonization.

Definition 31. Let $E \in \text{CC(eq)}$. A function $\text{Can} : \Sigma^* \to \Sigma^*$ is a *canonization for* E if it is polynomial time computable and

(1) for all $x, y \in \Sigma^*$: $\big(x E y \iff \text{Can}(x) = \text{Can}(y) \big)$;

(2) for all $x \in \Sigma^*$: $x E \, \text{Can}(x)$.

Let Can be a canonization of E. The *enumeration induced by* Can is the enumeration

$$x_1, x_2 \ldots$$

of $\text{Can}(\Sigma^*)$ such that $x_i <_{\text{lex}} x_j$ for $i < j$.

If E has a canonization, then $E \in \text{P}$: to decide whether $x E y$ we compute $\text{Can}(x)$ and $\text{Can}(y)$ and check whether $\text{Can}(x) = \text{Can}(y)$.

Now it is easy to explain the idea underlying the proof of Theorem 30. First we show that (under the assumption $\text{P} = \text{NP}$) every $E \in \text{P(eq)}$ has a canonization Can_E. Then, given $E, E' \in \text{P(eq)}$, we define a strong equivalence reduction $f : \Sigma^* \to \Sigma^*$ from E to E' as follows: Let $x \in \Sigma^*$. If $\text{Can}_E(x)$ is the ith element in the enumeration induced by Can_E, then we let $f(x)$ be the ith element in the enumeration induced by $\text{Can}_{E'}$. By the properties of canonizations it should be clear that

$$x E y \iff f(x) E' f(y)$$

(we can even replace $f(x) E' f(y)$ by $f(x) = f(y)$). So it remains to show (under suitable assumptions) that f is computable in polynomial time and to show that every equivalence relation has a canonization.

The following lemma was already proven in [2].

Lemma 32. *If* P = NP, *then every* $E \in$ P(eq) *has a canonization; in fact, then the mapping sending each* $x \in \Sigma^*$ *to the* $\leq_{\mathrm{lex}}$ *-first member of the* E-equivalence class of x is a canonization.

Proof: Let $E \in$ P(eq) and assume P = NP. Then we know that the polynomial hierarchy collapses, P = PH. So it suffices to show that the mapping defined in the statement of this lemma can be computed by an alternating polynomial time algorithm $\mathbb{A}$ with a constant number of alternations. This is easy: on input $x \in \Sigma^*$ the algorithm $\mathbb{A}$ guesses existentially $y \in \Sigma^*$ with $|y| \leq |x|$ and xEy; then $\mathbb{A}$ guesses universally a further $z \in \Sigma^*$ with $|z| \leq |x|$ and xEz; if $y \leq_{\mathrm{lex}} z$, then $\mathbb{A}$ outputs y otherwise it rejects. $\qquad\square$

Lemma 33. *Let* $E \in$ P(eq) *be an equivalence relation with a canonization* Can. *Then the following problem is in* #P:

> *Instance:* $x \in \Sigma^*$.
> *Problem:* Compute i (in binary) such that $\mathrm{Can}(x)$ is the ith element in the enumeration induced by Can.

Proof: Consider a nondeterministic polynomial time algorithm $\mathbb{A}$ which on input $x \in \Sigma^*$ runs as follows: It first computes the string $y := \mathrm{Can}(x)$. Then $\mathbb{A}$ guesses a string $z \in \Sigma^*$ with $|z| \leq |y|$. Finally it accepts if $\mathrm{Can}(z) = z$ and $z \leq_{\mathrm{lex}} y$. It should be clear that the number of accepting runs of $\mathbb{A}$ on x is

$$|\{z \mid z \leq_{\mathrm{lex}} \mathrm{Can}(x) \text{ and } \mathrm{Can}(z) = z\}|. \qquad\square$$

Proof of Theorem 30: Assume that P = #P. Let $E, E' \in$ P(eq) be equivalence relations and assume that $E \leq_{\mathrm{pot}} E'$, that is, $|\Sigma^{\leq n}/E| \leq |\Sigma^{\leq p(n)}/E'|$ for some polynomial p and all $n \in \mathbb{N}$. We show $E \leq_{\mathrm{eq}} E'$.

As P = #P, there are canonizations Can_E of E and $\mathrm{Can}_{E'}$ of E' and there are polynomial time algorithms $\mathbb{A}$ and $\mathbb{A}'$ that solve the problem of the preceding lemma for E and E', respectively. The following nondeterministic polynomial time algorithm computes an $f : E \leq_{\mathrm{eq}} E'$. On input $x \in \Sigma^*$, it computes $\mathrm{Can}_E(x)$ and $n := |\mathrm{Can}_E(x)|$ and guesses a string $x' \in \Sigma^{\leq p(n)}$ with $\mathrm{Can}_{E'}(x') = x'$. Simulating $\mathbb{A}$ and $\mathbb{A}'$, it checks whether $\mathrm{Can}_E(x)$ and x' are at the same position in the enumeration induced by Can_E and in the enumeration induced by $\mathrm{Can}_{E'}$, respectively; in the positive case it outputs x', otherwise it rejects. As $|\Sigma^{\leq n}/E| \leq |\Sigma^{\leq p(n)}/E'|$ such an $x' \in \Sigma^{\leq p(n)}$ with $\mathrm{Can}_{E'}(x') = x'$ at the same position as $\mathrm{Can}_E(x)$ exists. As P = NP, the function f is computable in polynomial time. $\qquad\square$

We briefly point to the papers [2, 3, 7] that deal with related problems. Let Inv(eq) be the class of equivalence relations having an invariantization (defined in analogy to Definition 5), Can(eq) the class of equivalence relations having a canonization and finally, Lexfirst(eq) the class of equivalence relations having a canonization that maps every string to the $\leq_{\mathrm{lex}}$ -first element of its equivalence class. Clearly

$$\text{Lexfirst(eq)} \subseteq \text{Can(eq)} \subseteq \text{Inv(eq)} \subseteq \text{P(eq)}. \tag{8}$$

Lemma 32 shows that Lexfirst(eq) = Can(eq) = Inv(eq) = P(eq) if P = #P. Blass and Gurevich [2], for example, prove that Lexfirst(eq) $\neq$ Can(eq) unless the polynomial hierarchy collapses, and Fortnow and Grochow [7] show that Can(eq) = Inv(eq) would imply that integers can be factored in probabilistic polynomial time. Blass and Gurevich [2, 3] compare the complexity of the "problems

underlying the definition of the sets in (8)." Finally, the book [14], among other things, deals with the question whether two propositional formulas are logically equivalent up to a permutation of their variables. It is not hard to see that the isomorphism problem for a class C can be rephrased in these terms; however no analogue of $\leq_{\mathrm{iso}}$ is considered in [14].

8. On maximum elements in P(eq) and NP(eq)

In this section we study whether there is a maximum element with respect to strong equivalence reductions in the classes P(eq) and NP(eq), that is, in the classes of deterministic and nondeterministic polynomial time equivalence relations. We already mentioned that the existence of a maximum element in P(eq) is mentioned as [7, Open Question 4.14]; the notion of strong equivalence reduction was already introduced in that paper and called kernel reduction there.

Let SAT be the set of satisfiable propositional formulas. Consider the NP-equivalence relation

$$E_{\mathrm{sat}} := \big\{ (\alpha, \beta) \mid \alpha, \beta \in \mathrm{PROP} \text{ and } \big(\alpha = \beta \text{ or } \alpha, \beta \in \mathrm{SAT} \big) \big\};$$

more precisely, to get an equivalence relation on Σ^*, we should write

$$E_{\mathrm{sat}} := \big\{ (\alpha, \beta) \mid \alpha, \beta \in \mathrm{PROP} \text{ and } \big(\alpha = \beta \text{ or } \alpha, \beta \in \mathrm{SAT} \big) \big\} \cup \big\{ (x, y) \mid x, y \in \Sigma^* \setminus \mathrm{PROP} \big\}.$$

However, henceforth if we speak of an equivalence relation E whose field $\mathrm{Fld}(E)$ is a proper subset of Σ^*, we identify it with the equivalence relation $E \cup \big\{ (x, y) \mid x, y \in \Sigma^* \setminus \mathrm{Fld}(E) \big\}$. We use E_{sat} to show:

Proposition 34. *If the polynomial hierarchy* PH *does not collapse, then* $E(\mathrm{GRAPH})$ *is not a maximum element in* $(\mathrm{NP(eq)}, \leq_{\mathrm{eq}})$; *in fact, then* $E_{\mathrm{sat}} \not\leq_{\mathrm{eq}} E(\mathrm{GRAPH})$.

Proof: For $\alpha \in \mathrm{PROP}$ and a propositional variable X we have $(\alpha \in \mathrm{SAT} \iff \alpha E_{\mathrm{sat}} X)$. By contradiction, assume that $f : E_{\mathrm{sat}} \leq_{\mathrm{eq}} E(\mathrm{GRAPH})$. We have $f(X) \in \mathrm{GRAPH}$; otherwise, $\mathrm{SAT} \in \mathrm{P}$, which contradicts our assumption that the polynomial hierarchy does not collapse. Then for every $\alpha \in \mathrm{PROP}$

$$\alpha \in \mathrm{SAT} \iff f(\alpha) \cong f(X).$$

Thus $E(\mathrm{GRAPH})$ would be NP-complete. It is well-known [4] that this implies $\Sigma_2^p = \mathrm{PH}$. $\square$

We show that the existence of a maximum element in $(\mathrm{NP(eq)}, \leq_{\mathrm{eq}})$ is equivalent to the existence of an effective enumeration of NP(eq). This result is also true for P(eq) and co-NP(eq). To state the precise result we introduce some notions. A deterministic or nondeterministic Turing machine $\mathbb{M}$ is *clocked* (more precisely, *polynomially time-clocked*), if (the code of) $\mathbb{M}$ contains a natural number $\mathrm{time}(\mathbb{M})$ such that $n^{\mathrm{time}(\mathbb{M})}$ is a bound for the running time of $\mathbb{M}$ on inputs of length n. So, by this definition, all runs of a clocked machine are of polynomial length. Of course, the function $\mathbb{M} \mapsto \mathrm{time}(\mathbb{M})$, defined on the set of clocked machines, is computable in polynomial time.

Definition 35. Let $\mathrm{CC} \in \{\mathrm{P}, \mathrm{NP}, \mathrm{co\text{-}NP}\}$. Let L be a set of languages L with $L \subseteq \Sigma^*$. We say that

$$L_0, L_1, \ldots$$

is a CC-*enumeration of* L *by clocked Turing machines*, if $\mathrm{L} = \{L_0, L_1, \ldots\}$ and there is a computable function $\mathbb{M}$ defined on $\mathbb{N}$ such that $\mathbb{M}(i)$ for $i \in \mathbb{N}$ is (the code of) a clocked Turing machine of type CC accepting L_i.

Proposition 36. *Let* $CC \in \{P, NP, co\text{-}NP\}$. *Then the following are equivalent:*

(1) $(CC(eq), \leq_{eq})$ *has a maximum element.*

(2) *There is a* CC-*enumeration* $E_0, E_1, \ldots$ *of* CC(eq*) by clocked Turing machines.*

Proof: $(1) \Rightarrow (2)$: Assume that E is a maximum element in $(CC(eq), \leq_{eq})$ and let $\mathbb{M}_{max}$ be a Turing machine of type CC accepting E. Of course, there is a computable function $\mathbb{M}'$ such that $\mathbb{M}'(i)$ for $i \in \mathbb{N}$ is a deterministic clocked Turing machine computing a function $f_i : \Sigma^* \to \Sigma^*$ such that $f_0, f_1, \ldots$ is an enumeration of all polynomial time computable functions from Σ^* to Σ^*. We define the machine $\mathbb{M}_{max} \circ \mathbb{M}'(i)$ in a straightforward manner such that it decides

$$E_i := \big\{ (x,y) \mid (f_i(x), f_i(y)) \in E \big\}.$$

We let $\mathbb{M}$ be the function defined on $\mathbb{N}$ with $\mathbb{M}(i) := \mathbb{M}_{max} \circ \mathbb{M}'(i)$. As from a polynomial bounding $\mathbb{M}_{max}$ and $time(\mathbb{M}'(i))$ we get a time bound for $\mathbb{M}(i)$, we can assume that $\mathbb{M}(i)$ is clocked. It should be clear that $E_0, E_1, \ldots$ has the desired properties.

$(2) \Rightarrow (1)$: Let $E_0, E_1, \ldots$ be as in (2) and let $\mathbb{M}$ be a corresponding computable function. By padding if necessary, we may assume that the graph $\{(1^i, 1^{|\mathbb{M}(i)|}) \mid i \in \mathbb{N}\}$ is decidable in polynomial time and that $i \leq |\mathbb{M}(i)|$ for all $i \in \mathbb{N}$. We define the relation E as follows (for better reading we denote here, and in the proof of Lemma 39, the string 1^ℓ, that is the string $11 \ldots 1$ of length ℓ, by $\langle \ell \rangle$):

$$E := \Big\{ \Big((\mathbb{M}(i), x, \langle (2+2|x|)^{time(\mathbb{M}(i))} \rangle), (\mathbb{M}(i), y, \langle (2+2|y|)^{time(\mathbb{M}(i))} \rangle) \Big) \mid i \in \mathbb{N} \text{ and } (x,y) \in E_i \Big\}.$$

By the effectivity properties of $\mathbb{M}$, we have $E \in CC(eq)$ (more precisely $E \cup \{(x,y) \mid x,y \in \Sigma^* \setminus Fld(E)\} \in CC(eq)$. Clearly, for $i \in \mathbb{N}$ the mapping $x \mapsto (\mathbb{M}(i), x, \langle (2+2|x|)^{time(\mathbb{M}(i))} \rangle)$ is a strong equivalence reduction from E_i to E, hence E is a maximum element. $\qquad\square$

Below we will show that $(NP(eq), \leq_{eq})$ has a maximum element if $NP = co\text{-}NP$. Note that we do not even know whether $(P(eq), \leq_{eq})$ has a maximum element. The main result concerning this problem that we have reads as follows (later we recall the definition of p-optimal proof system):

Theorem 37. *If* TAUT *has a p-optimal proof system, then* $(P(eq), \leq_{eq})$ *has a maximum element.*

The following observations will lead to a proof of this result.

Definition 38. Let $\mathbb{M}$ be a deterministic or nondeterministic Turing machine and $n \in \mathbb{N}$. The machine $\mathbb{M}$ *defines an equivalence relation on* $\Sigma^{\leq n}$ if the set

$$\big\{ (x,y) \mid x,y \in \Sigma^{\leq n} \text{ and } \mathbb{M} \text{ accepts } (x,y) \big\}$$

is an equivalence relation on $\Sigma^{\leq n} := \big\{ x \in \Sigma^* \mid |x| \leq n \big\}$.

An analysis of the complexity of the first of the following problems will be crucial for our purposes.

EQUIV(P)
>*Instance:* A deterministic clocked Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$.
>*Problem:* Does $\mathbb{M}$ define an equivalence relation on $\Sigma^{\leq n}$?

EQUIV(NP)
    Instance: A nondeterministic clocked Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$.
    Problem: Does $\mathbb{M}$ define an equivalence relation on $\Sigma^{\leq n}$?

Lemma 39.   *(1) If $(\mathbb{M}, n) \in$ EQUIV(P) is solvable by a deterministic algorithm in time $n^{f(\|\mathbb{M}\|)}$ for some function $f : \mathbb{N} \to \mathbb{N}$, then P(eq) has a maximum element.[4]*

    (2) If $(\mathbb{M}, n) \in$ EQUIV(NP) is solvable by a nondeterministic algorithm in time $n^{f(\|\mathbb{M}\|)}$ for some function $f : \mathbb{N} \to \mathbb{N}$, then NP(eq) has a maximum element.

Proof: Let $\mathbb{A}$ be an algorithm, deterministic for (1) and nondeterministic for (2), witnessing that $(\mathbb{M}, n) \in Q$ is solvable in time $n^{f(\|\mathbb{M}\|)}$ for some $f : \mathbb{N} \to \mathbb{N}$. We define the equivalence relation E_0 on Σ^* by: for $u, v \in \Sigma^*$

$$u E_0 v$$

if and only if

$$u = v \text{ or } \Big(u = \big(\mathbb{M}, x, (2 + 2 \cdot |x|)^{\text{time}(\mathbb{M})}, 1^t \big) \text{ and}$$
$$v = \big(\mathbb{M}, x', (2 + 2 \cdot |x'|)^{\text{time}(\mathbb{M})}, 1^{t'} \big) \text{ and } (1) - (3) \text{ are fulfilled} \Big),$$

where

- $\mathbb{M}$ is a clocked Turing machine of type CC, where CC = P for (1) and CC = NP for (2);

- $\mathbb{A}$ accepts $(\mathbb{M}, |x|)$ in at most t steps and $(\mathbb{M}, |x'|)$ in at most t' steps;

- $\mathbb{M}$ accepts (x, x').

Clearly, $E_0 \in$ CC(eq). We show that E_0 is a maximum element. Let $E \in$ CC(eq) be arbitrary and let $\mathbb{M}$ be a clocked Turing machine deciding E. Then

$$x \mapsto \big(\mathbb{M}, x, (2 + 2 \cdot |x|)^{\text{time}(\mathbb{M})}, \langle |x|^{f(\|\mathbb{M}\|)} \rangle \big)$$

is computable in polynomial time and hence a strong equivalence reduction from E to E_0. $\square$

Theorem 40.   *(1) If E = NE, then P(eq) has a maximum element.*

    (2) If NP = co-NP, then NP(eq) has a maximum element.

Proof: (1) We may assume that n is written in binary in the instances $(\mathbb{M}, n)$ of EQUIV(P). We consider the following nondeterministic algorithm $\mathbb{A}$ accepting the complement of EQUIV(P). On input $(\mathbb{M}, n)$, it guesses one of the three axioms of an equivalence relation, say, the transitivity axiom; then $\mathbb{A}$ guesses $x, y, z \in \Sigma^n$, it simulates $\mathbb{M}$ on input (x, y), on input (y, z), and on input (x, z) and accepts if $\mathbb{M}$ accepts the first two inputs but not the third one. As we may assume that $\|\mathbb{M}\| \geq \text{time}(\mathbb{M})$, the algorithm $\mathbb{A}$ runs in time $\|\mathbb{M}\| \cdot n^{O(\text{time}(\mathbb{M}))} = 2^{O(\|\mathbb{M}\| \cdot \log n)}$. By the assumption E = NE, there is a deterministic algorithm deciding the complement of EQUIV(P) and hence EQUIV(P) itself in time $2^{O(\|\mathbb{M}\| \cdot \log n)}$. Now our claim follows from the preceding lemma.

---

[4]By $\|\mathbb{M}\|$ we denote the length of a reasonable encoding of $\mathbb{M}$ by a string of Σ^*.

(2) The following alternating algorithm $\mathbb{A}$ decides the complement of $\text{EQUIV}(\text{NP})$: On input $(\mathbb{M}, n)$ (again we may assume that $\|\mathbb{M}\| \geq \text{time}(\mathbb{M})$), it existentially guesses one of the three axioms of an equivalence relation, say, the transitivity axiom; then $\mathbb{A}$ existentially guesses $x, y, z \in \Sigma^n$ and runs of $\mathbb{M}$ accepting (x, y) and (y, z); furthermore it yields the string $\langle n^{\|\mathbb{M}\|} \rangle$. Finally $\mathbb{A}$ universally simulates $\mathbb{M}$ on input (x, z) and accepts if $\mathbb{M}$ rejects. The algorithm $\mathbb{A}$ has one alternation. By our assumption $\text{NP} = \text{co-NP}$, its universal part (an algorithm of type co-NP with inputs $\mathbb{M}$, (x, z), and $\langle n^{\|\mathbb{M}\|} \rangle$) can be simulated by a nondeterministic algorithm running in time $n^{O(\|\mathbb{M}\|)}$. Altogether we get a nondeterministic algorithm accepting (the complement of) $\text{EQUIV}(\text{NP})$ in time $n^{O(\|\mathbb{M}\|)}$. Now our claim follows from the preceding lemma. $\qquad\square$

We consider the *acceptance problem for nondeterministic Turing machines*:

$\text{ACC}_{\leq}$
>   *Instance:*   A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$.
>   *Problem:*   Does $\mathbb{M}$ accept the empty input tape in $\leq n$ steps?

Lemma 41. *The following are equivalent:*

(1) $(\mathbb{M}, n) \in \text{ACC}_{\leq}$ *is solvable deterministically in time* $n^{f(\|\mathbb{M}\|)}$ *for some* $f : \mathbb{N} \to \mathbb{N}$.

(2) $(\mathbb{M}, n) \in \text{EQUIV}(\text{P})$ *is solvable deterministically in time* $n^{f(\|\mathbb{M}\|)}$ *for some* $f : \mathbb{N} \to \mathbb{N}$.

Proof: $(1) \Rightarrow (2)$: Assume that $(\mathbb{M}, n) \in \text{ACC}_{\leq}$ (where $\mathbb{M}$ is a nondeterministic machine and $n \in \mathbb{N}$) can be solved by an algorithm $\mathbb{A}$ in time $n^{f(\|\mathbb{M}\|)}$ for some $f : \mathbb{N} \to \mathbb{N}$. Then the following algorithm $\mathbb{B}$ will witness that $\text{EQUIV}(\text{P})$ is decidable in the time claimed in (2). Let $(\mathbb{M}, n)$ be an instance of $\text{EQUIV}(\text{P})$, in particular $\mathbb{M}$ is a deterministic clocked Turing machine. We may assume that $\mathbb{M}$ on input (x, y) runs for exactly $(2 + 2 \cdot \max\{x, y\})^{\text{time}(\mathbb{M})}$ steps. Let $\widetilde{\mathbb{M}}$ be the nondeterministic Turing machine that on empty input tape, in the first phase guesses one of the three axioms of an equivalence relation, say, the transitivity axiom; then in the second phase $\widetilde{\mathbb{M}}$ guesses $x, y, z \in \Sigma^*$; finally in the third phase it simulates $\mathbb{M}$ on input (x, y), on input (y, z), and on input (x, z) and accepts if $\mathbb{M}$ accepts the first two inputs but not the third one. We can assume that $\widetilde{\mathbb{M}}$ does this simulation in such a way that it runs for exactly $(2 + 2 \cdot \max\{x, y, z\})^{\text{time}(\mathbb{M})}$ steps on each of the tuples (x, y), (y, z), and (x, z).

Let k_1, $k_2(x, y, z)$, and $k_3(x, y, z)$ be the exact time $\widetilde{\mathbb{M}}$ uses for the first phase, the second phase and the third phase, respectively. As indicated for the third phase we may arrange things in such a way that there are (nonconstant) polynomials k_2', k_3' such that

$$k_2(x, y, z) = k_2'(\max\{|x|, |y|, |z|\}) \text{ and } k_3(x, y, z) = k_3'(\max\{|x|, |y|, |z|\})$$

and such that if for example $\widetilde{\mathbb{M}}$ has chosen the symmetry axiom and $x, y \in \Sigma^*$, then $k_2(\max\{|x|, |y|\})$ is also the exact number of steps $\widetilde{\mathbb{M}}$ uses for the second phase. As k_2' and k_3' are increasing functions, we get

$$(\mathbb{M}, n) \notin \text{EQUIV} \iff (\widetilde{\mathbb{M}}, k + k_2'(n) + k_3'(n)) \in \text{ACC}_{\leq},$$

which gives the desired bound.

$(2) \Rightarrow (1)$: For a nondeterministic Turing machine $\mathbb{M}$ let $\widehat{\mathbb{M}}$ be the deterministic Turing machine that on input (x, y) with $x, y \in \Sigma^*$ first checks whether $x \neq y$; if so, it accepts; if $x = y$, it simulates the $|x|$ steps of a run of $\mathbb{M}$ on empty input tape, namely the steps corresponding to (the bits in) x and rejects if in these $|x|$ steps $\mathbb{M}$ accepts; otherwise $\widehat{\mathbb{M}}$ accepts. Thus for every $n \in \mathbb{N}$

$$(\mathbb{M}, n) \in \text{ACC}_{\leq} \iff \widehat{\mathbb{M}} \text{ does not define an equivalence relation on } \Sigma^{\leq n}.$$

As from the definition of $\widehat{\mathbb{M}}$ we immediately get a polynomial time bound, we can assume that $\widehat{\mathbb{M}}$ is clocked, so that the preceding equivalence immediately gives the claim. $\square$

A *proof system* for TAUT is a surjective function $S : \Sigma^* \to$ TAUT computable in polynomial time. The proof system S for TAUT is *p-optimal* if for every proof system S' for TAUT there is a polynomial time computable $T : \Sigma^* \to \Sigma^*$ such that for all $w \in \Sigma^*$

$$S(T(w)) = S'(w).$$

It is not known whether there is a p-optimal proof system for TAUT, even though it is conjectured there is no such p-optimal proof system. In [5] it has been shown that:

Proposition 42. *The following are equivalent:*

(1) There is a p-optimal proof system for TAUT*.*

(2) $(\mathbb{M}, n) \in \text{ACC}_\leq$ *is solvable in time* $n^{f(\|\mathbb{M}\|)}$ *for some function* $f : \mathbb{N} \to \mathbb{N}$*.*

Proof of Theorem 37: If there is a p-optimal proof system for TAUT, by the previous proposition and Lemma 41 we see that $(\mathbb{M}, n) \in \text{EQUIV}(P)$ is solvable in time $n^{f(\|\mathbb{M}\|)}$ for some function $f : \mathbb{N} \to \mathbb{N}$. Now the claim follows from Lemma 39.

References

[1] H.U. Besche, B. Eick and E.A. O'Brien. The groups of order at most 2000, Electronic Research announcements of the American Mathematical Society, 7:1–4, 2001.

[2] A. Blass and Y. Gurevich. Equivalence relations, invariants, and normal forms. SIAM Journal of Computing, 13:682–689, 1984.

[3] A. Blass and Y. Gurevich. Equivalence relations, invariants, and normal forms, II. Lecture Notes in Computer Science, 171:24–42, 1984.

[4] R. B. Boppana, J. Hastad and S. Zachos. Does co-NP have short interactive proofs? Information Processing Letters, 25(2):127-132, 1987.

[5] Y. Chen and J. Flum. On p-optimal proof systems and logics for PTIME. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP'10)*, Lecture Notes in Computer Science 6199, pages 321–332, Springer, 2010.

[6] H.-D. Ebbinghaus and J. Flum. Finite Model Theory. Perspectives in Mathematical Logic, Second Edition, Springer 1999.

[7] L. Fortnow and J. Grochow. Complexity classes of equivalence problems revisited, arXiv:0907.4775v1 [cs.CC], 2009.

[8] S. Friedman Descriptive set theory for finite structures, Lecture at the Kurt Gödel Research Center, 2009, Available at http://www.logic.univie.ac.at/~sdf/papers/wien-spb.pdf

[9] H. Friedman and L. Stanley. A Borel reducibility theory for classes of countable structures, Journal Symbolic Logic 54, (1989), 894–914.

[10] S. Givant and P. Halmos. Introduction to Boolean algebras, Springer, 2008.

[11] Y. Gurevich. From invariants to canonization. Bull. Europ. Assoc. Theor. Comp. Sci 63:115–119, 1997.

[12] T. Kavithal. Efficient algorithms for abelian group isomorphism and related problems. In *Proceedings of the 23rd Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'02)*, Lecture Notes in Computer Science 2914, pages 277–288, Springer, 2003.

[13] G. Miller. Isomorphism testing for graphs of bounded genus. In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC'80)*, 225–235, 1980.

[14] T. Thierauf. The computational complexity of equivalence and isomorphism problems. Lecture Notes in Computer Science, 1852, Springer, 2000.