A First-Order Isomorphism Theorem*

Eric Allender†

Department of Computer Science, Rutgers University
New Brunswick, NJ, USA 08903
`allender@cs.rutgers.edu`

José Balcázar‡

U. Politècnica de Catalunya, Departamento L.S.I.
Pau Gargallo 5, E-08071 Barcelona, Spain
`balqui@lsi.upc.es`

Neil Immerman§

Computer Science Department, University of Massachusetts
Amherst, MA, USA 01003
`immerman@cs.umass.edu`

Abstract

We show that for most complexity classes of interest, all sets complete under first-order projections (fops) are isomorphic under first-order isomorphisms. That is, a very restricted version of the Berman-Hartmanis Conjecture holds. Since "natural" complete problems seem to stay complete via fops, this indicates that up to first-order isomorphism there is only one "natural" complete problem for each "nice" complexity class.

1 Introduction

In 1977 Berman and Hartmanis noticed that all NP complete sets that they knew of were polynomial-time isomorphic, [BH77]. They made their now-famous isomorphism conjecture: namely that all NP complete sets are polynomial-time isomorphic. This conjecture has engendered a large amount of work (cf. [KMR90, You] for surveys).

The isomorphism conjecture was made using the notion of NP completeness via polynomial-time, many-one reductions because that was the standard definition at the time. In [Coo],

*A preliminary version of this work appeared in Proc. 10th Symposium on Theoretical Aspects of Computer Science, 1993, Lecture Notes in Computer Science 665, pp. 163–174.

†Some of this work was done while on leave at Princeton University; supported in part by National Science Foundation grant CCR-9204874.

‡Supported in part by ESPRIT-II BRA EC project 3075 (ALCOM) and by Acción Integrada Hispano-Alemana 131 B

§Supported by NSF grant CCR-9207797.

Cook proved that the Boolean satisfiability problem (SAT) is NP complete via polynomial-time Turing reductions. Over the years SAT has been shown complete via weaker and weaker reductions, e.g. polynomial-time many-one [Kar], logspace many-one [Jon], one-way logspace many-one [HIM], and first-order projections (fops) [Dah]. These last reductions, defined in Section 3, are provably weaker than logspace reductions. It has been observed that *natural* complete problems for various complexity classes including NC^1, L, NL, P, NP, and PSPACE remain complete via fops, cf. [I87, IL, SV, Ste, MI].

On the other hand, Joseph and Young, [JY] have pointed out that polynomial-time, many-one reductions may be so powerful as to allow *unnatural* NP-complete sets. Most researchers now believe that the isomorphism conjecture as originally stated by Berman and Hartmanis is false.[1]

We feel that the choice of polynomial-time, many-one reductions in the statement of the Isomorphism Conjecture was made in part for historical rather than purely scientific reasons. To elaborate on this claim, note that the class NP arises naturally in the study of logic and can be defined entirely in terms of logic, without any mention of computation [Fa]. Thus it is natural to have a notion of NP-completeness that is formulated entirely in terms of logic. On another front, Valiant [Val] noticed that reducibility can be formulated in algebra using the natural notion of a projection, again with no mention of computation. The sets that are complete under fops are complete in *all* of these different ways of formulating the notion of NP-completeness.

Since natural complete problems turn out to be complete via very low-level reductions such as fops, it is natural to modify the isomorphism conjecture to consider NP-complete reductions via fops. Motivating this in another way, one could propose as a slightly more general form of the isomorphism conjecture the question: is completeness a sufficient structural condition for isomorphism? Our work answers this question by presenting a notion of completeness for which the answer is yes. Namely for every nice complexity class including P, NP, etc, any two sets complete via fops are not only polynomial-time isomorphic, they are first-order isomorphic.

There are additional reasons to be interested in first-order computation. It was shown in [BIS] that first-order computation corresponds exactly to computation by uniform AC^0 circuits under a natural notion of uniformity. Although it is known that AC^0 is properly contained in NP, knowing that a set A is complete for NP under polynomial-time (or logspace) reductions does not currently allow us to conclude that A is not in AC^0; however, knowing that A is complete for NP under first-order reductions does allow us to make that conclusion.

First-order reducibility is a uniform version of the constant-depth reducibility studied in [FSS, CSV]; sometimes this uniformity is important. For a concrete example where first-order reducibility is used to provide a circuit lower bound, see [AG92].

Preliminary results and background on isomorphisms follow in Section 2. Definitions and background on descriptive complexity are found in Section 3. The main result is stated and proved in Section 4, and then we conclude with some related results and remarks about the structure of NP under first-order reducibilities.

[1] One way of quantifying this observation is that since Joseph and Young produced their unnatural NP-complete sets, Hartmanis has been referring to the isomorphism conjecture as the "Berman" conjecture.

2 Short History of the Isomorphism Conjecture

The Isomorphism Conjecture is analogous to Myhill's Theorem that all r.e. complete sets are recursively isomorphic, [Myh]. In this section we summarize some of the relevant background material. In the following FP is the set of functions computable in polynomial time.

Definition 2.1 For $A, B \subseteq \Sigma^*$, we say that A and B are *p-isomorphic* ($A \overset{p}{\cong} B$) iff there exists a bijection $f \in \mathrm{FP}$ with inverse $f^{-1} \in \mathrm{FP}$ such that A is many-one reducible to B ($A \leq_{\mathrm{m}} B$) via f (and therefore $B \leq_{\mathrm{m}} A$ via f^{-1}). □

Observation 2.2 ([BH77]) *All the NP complete sets in [GJ] are p-isomorphic.*

How did Berman and Hartmanis make their observation? They did it by proving a polynomial-time version of the Schröder-Bernstein Theorem. Recall:

Theorem 2.3 ([Kel, Th. 20]) *Let A and B be any two sets. Suppose that there are 1:1 maps from A to B and from B to A. Then there is a 1:1 and onto map from A to B.*

Proof Let $f : A \to B$ and $g : B \to A$ be the given 1:1 maps. For simplicity assume that A and B are disjoint. For $a, c \in A \cup B$, we say that c is an *ancestor* of a iff we can reach a from c by a finite (non-zero) number of applications of the functions f and/or g. Now we can define a bijection $h : A \to B$ which applies either f or g^{-1} according as whether a point has an odd number of ancestors or not:

$$h(a) = \begin{cases} g^{-1}(a) & \text{if } a \text{ has an odd number of ancestors} \\ f(a) & \text{if } a \text{ has an even or infinite number of ancestors} \end{cases}$$

□

The feasible version of the Schröder-Bernstein theorem is as follows:

Theorem 2.4 ([BH77]) *Let $f : A \leq_m B$ and $g : B \leq_m A$, where f and g are 1:1, length-increasing functions. Assume that $f, f^{-1}, g, g^{-1} \in \mathrm{FP}$ where f^{-1}, g^{-1} are inverses of f, g. Then $A \overset{p}{\cong} B$.*

Proof Let the ancestor chain of a string w be the path from w to w's parent, to w's grandparent, and so on. Ancestor chains are at most linear in length because f and g are length-increasing. Thus they can be computed in polynomial time. The theorem now follows as in the proof of Theorem 2.3. □

Consider the following definition:

Definition 2.5 ([BH77]) We say that the language $A \subseteq \Sigma^*$ has *p-time padding functions* iff there exist $e, d \in \mathrm{FP}$ such that

1. For all $w, x \in \Sigma^*$, $w \in A \Leftrightarrow e(w, x) \in A$

2. For all $w, x \in \Sigma^*$, $d(e(w,x)) = x$

3. For all $w, x \in \Sigma^*$, $|e(w,x)| \geq |w| + |x|$

□

As a simple example, the following is a padding function for SAT,

$$e(w,x) \quad = \quad (w) \wedge c_1 \wedge c_2 \wedge \cdots \wedge c_{|x|}$$

where c_i is $(y \vee \bar{y})$ if the i^{th} bit of x is 1 and $(\bar{y} \vee y)$ otherwise, where y is a Boolean variable numbered higher than all the Boolean variables occurring in w.

Then the following theorem follows from Theorem 2.4:

Theorem 2.6 ([BH77]) *If A and B are NP complete and have p-time padding functions, then $A \overset{p}{\cong} B$.*

Finally, Observation 2.2 now follows from the following:

Observation 2.7 ([BH77]) *All the NP complete problems in [GJ] have p-time padding functions.*

Hartmanis also extended the above work as follows: Say that A has *logspace-padding functions* if there are logspace computable functions as in Definition 2.5.

Theorem 2.8 ([Har]) *If A and B are NP complete via logspace reductions and have logspace padding functions, then A and B are logspace isomorphic.*

Proof Since A and B have logspace padding functions, we can create functions f and g as in Theorem 2.4 that are length squaring and computable in logspace. Then, the whole ancestor chain can be computed in logspace because each successive iteration requires half of the previous space. □

Here, we show that sets complete under a very restrictive notion of reducibility are isomorphic under a very restricted class of isomorphisms. This result is incomparable to a recent result of [AB], showing that all sets complete under one-way logspace reductions (1-L reductions) are isomorphic under polynomial-time computable isomorphisms. (This work of [AB] improves an earlier result of [A88].) Note that it is easy to prove that the class of 1-L reductions is incomparable with the class of first-order projections. Other interesting results concerning 1-L reductions may be found in [BH90, HH].

3 Descriptive Complexity

In this section we recall the notation of Descriptive Complexity which we will need to state and prove our main results. See [I89] for a survey and [IL] for an extensive discussion of the reductions we use here including first-order projections.

We will code all inputs as finite logical structures. The most basic example is a binary string w of length $n = |w|$. We will represent w as a logical structure:

$$\mathcal{A}(w) = \langle \{0, 1, \ldots, n{-}1\}, R \rangle$$

where the unary relation $R(x)$ holds in $\mathcal{A}(w)$ (in symbols, $\mathcal{A}(w) \models R(x)$) just if bit x of w is a 1. As is customary, the notation $|\mathcal{A}|$ will be used to denote the universe $\{0, 1, \ldots, n{-}1\}$ of the structure $\mathcal{A}$. We will write $\|\mathcal{A}\|$ to denote n, the cardinality of $|\mathcal{A}|$.

A *vocabulary* $\tau = \langle R_1^{a_1} \ldots R_r^{a_r}, c_1, \ldots, c_s \rangle$ is a tuple of input relation and constant symbols. We call the R_i's "input relations" because they correspond to the input bits to a boolean circuit. In the case of binary strings, the input relation tells us which bits are 0 and which are 1. In the case of graphs, the input relation E tells us which edges are present.

Let STRUC$[\tau]$ denote the set of all finite structures of vocabulary τ. We define a complexity theoretic *problem* to be any subset of STRUC$[\tau]$ for some τ.

For any vocabulary τ there is a corresponding first-order language $\mathcal{L}(\tau)$ built up from the symbols of τ and the numeric relation symbols and constant symbols[2]: $=, \leq, \mathrm{BIT}, 0, \mathbf{m}$, using logical connectives: $\wedge, \vee, \neg$, variables: $x, y, z, \ldots$, and quantifiers: $\forall, \exists$.

First-Order Interpretations and Projections

In [Val], Valiant defined the *projection,* an extremely low-level many-one reduction.

Definition 3.1 Let $S, T \subseteq \{0,1\}^\star$. A k-ary *projection* from S to T is a sequence of maps $\{p_n\}$, $n = 1, 2, \ldots$, that satisfy the following properties. First, for all n and for all binary strings s of length n, $p_n(s)$ is a binary string of length n^k and,

$$s \in S \quad \Leftrightarrow \quad p_n(s) \in T \; .$$

Second, let $s = s_0 s_1 \ldots s_{n-1}$. Then each map p_n is defined by a sequence of n^k literals: $\langle l_0, l_1, \ldots, l_{n^k-1} \rangle$ where

$$l_i \in \{0,1\} \cup \{s_j, \bar{s}_j \mid 0 \leq j \leq n{-}1\} \; .$$

Thus as s ranges over strings of length n, each bit of $p_n(s)$ depends on at most one bit of s,

$$p_n(s)[[i]] \quad = \quad l_i(s) \; .$$

□

[2]Here $\leq$ refers to the usual ordering on $\{0, \ldots, n-1\}$, "BIT(i,j)" means that the i^{th} bit of the binary representation of j is 1, and 0 and $\mathbf{m}$ refer to 0 and $n-1$, respectively. For simplicity we will assume throughout that $n > 1$ and thus $0 \neq \mathbf{m}$. These relations are called "numeric" as opposed to the input relations because, for example, "BIT(i,j)" and "$i \leq j$" depend only on the numeric values of i and j and do not refer to the input.

Projections were originally defined as a non-uniform sequence of reductions – one for each value of n. That is, a projection can be viewed as a many-one reduction produced by a family $\{C_n\}$ of circuits of depth one. The circuits consist entirely of wires connecting input bits or negated input bits to outputs. If the circuit family $\{C_n\}$ is sufficiently *uniform*, we arrive at the class of *first-order projections*. (Recall that first-order corresponds to uniform AC^0 [BIS].) We find it useful to work in the framework of first-order logic rather than in the circuit model. The rest of this section presents the necessary definitions of first-order reductions.

The idea of the definition is that the choice of the literals $\langle l_0, l_1, \ldots, l_{n^k-1}\rangle$ in Definition 3.1 is given by a first-order formula in which no input relation occurs. Thus the formula can only talk about bit positions, and not bit values. The choice of literals depends only on n. In order to make this definition, we must first define first-order interpretations. These are a standard notion from logic for translating one theory into another, cf. [End], modified so that the transformation is also a many-one reduction, [I87]. (For readers familiar with databases, a first-order interpretation is exactly a many-one reduction that is definable as a first-order query.)

Definition 3.2 (First-Order Interpretations) Let σ and τ be two vocabularies, with $\tau = \langle R_1^{a_1}, \ldots, R_r^{a_r}, c_1, \ldots, c_s\rangle$. Let $S \subseteq \mathrm{STRUC}[\sigma]$, $T \subseteq \mathrm{STRUC}[\tau]$ be two problems. Let k be a positive integer. Suppose we are given an r-tuple of formulas $\varphi_i \in \mathcal{L}(\sigma)$, $i = 1, \ldots, r$, where the free variables of φ_i are a subset of $\{x_1, \ldots, x_{k\cdot a_i}\}$. Finally, suppose we are given an s-tuple of constant symbols[3] $t_1, \ldots, t_s$ from $\mathcal{L}(\sigma)$. Let $I = \lambda_{x_1 \ldots x_d}\langle \varphi_1, \ldots, \varphi_r, t_1, \ldots, t_s\rangle$ be a tuple of these formulas and constants. (Here $d = \max_i(ka_i)$.)

Then I induces a mapping also called I from $\mathrm{STRUC}[\sigma]$ to $\mathrm{STRUC}[\tau]$ as follows. Let $\mathcal{A} \in \mathrm{STRUC}[\sigma]$ be any structure of vocabulary σ, and let $n = ||\mathcal{A}||$. Then the structure $I(\mathcal{A})$ is defined to be:

$$I(\mathcal{A}) \;=\; \langle \{0, \ldots, n^k - 1\}, R_1, \ldots, R_r, t_1, \ldots, t_s\rangle$$

where the relation R_i is determined by the formula φ_i, for $i = 1, \ldots, r$ as follows. Let the function $\langle \cdot, \cdots, \cdot\rangle : |\mathcal{A}|^k \to |I(\mathcal{A})|$ be given by

$$\langle u_1, u_2, \ldots, u_k\rangle \;=\; u_k + u_{k-1}n + \cdots + u_1 n^{k-1}$$

Then,

$$R_i \;=\; \{(\langle u_1, \ldots, u_k\rangle, \ldots, \langle u_{1+k(a_i-1)}, \ldots, u_{ka_i}\rangle) \mid \mathcal{A} \models \varphi_i(u_1, \ldots u_{ka_i})\}$$

If the structure $\mathcal{A}$ interprets some variables $\bar{u}$ then these may appear freely in the the φ_i's and t_j's of I, and the definition of $I(\mathcal{A})$ still makes sense.

Suppose that I is a many-one reduction from S to T, i.e. for all $\mathcal{A}$ in $\mathrm{STRUC}[\sigma]$,

$$\mathcal{A} \in S \quad \Leftrightarrow \quad I(\mathcal{A}) \in T$$

Then we say that I is a k-ary *first-order interpretation* of S to T. $\square$

[3] More generally, we could use closed terms which are expressions involving constants and function symbols. An even more general way to interpret constants and functions is via a formula φ such that $\vdash (\forall \bar{x})(\exists! y)\varphi(\bar{x}, y)$. However, in this paper the simpler definition involving constant symbols suffices.

We are now ready to define first-order projections, a syntactic restriction of first-order interpretations. If each formula in the first-order interpretation I satisfies this syntactic condition then it follows that I is also a projection in the sense of Valiant. In this case we call I a first-order projection.

Definition 3.3 [First-Order Projections] Let $I = \langle \varphi_1, \ldots, \varphi_r, t_1, \ldots, t_s \rangle$. be a k-ary first-order interpretation from S to T as in Definition 3.2. Suppose further that the φ_i's all satisfy the following *projection condition:*

$$\varphi_i \equiv \alpha_1 \vee (\alpha_2 \wedge \lambda_2) \vee \cdots \vee (\alpha_e \wedge \lambda_e) \qquad (3.4)$$

where the α_j's are mutually exclusive formulas in which no input relations occur, and each λ_j is a literal, i.e. an atomic formula $P(x_{j_1}, \ldots x_{j_a})$ or its negation.

In this case the predicate $R_i(\langle u_1, \ldots, u_k \rangle, \ldots, \langle \ldots, u_{ka_i} \rangle)$ holds in $I(\mathcal{A})$ if $\alpha_1(\bar{u})$ is true, or if $\alpha_j(\bar{u})$ is true for some $1 < j \leq e$ and the corresponding literal $\lambda_j(\bar{u})$ holds in $\mathcal{A}$. Thus each bit in the binary representation of $I(\mathcal{A})$ is determined by at most one bit in the binary representation of $\mathcal{A}$. We say that I is a *first-order projection* (fop). Write $S \leq_{\text{fop}} T$ to mean that S is reducible to T via a first-order projection. □

Example 3.5 To help the reader grasp an intuition of the way an fop reduction behaves, let us describe an example. We present here the reduction from 3-SAT, satisfiability of CNF Boolean expressions with exactly three literals per clause, to 3-COL, the problem of coloring the vertices of a graph with 3 colors under the constraint that the endpoints of all edges get different colors. We use the same reduction as described in section 11.4.5 of [Man], so that the reader in need of additional help can consult it there.

The respective vocabularies for the input and output structures are as follows. To describe instances of 3-SAT, clauses and Boolean variables are each numbered from 0 through $n-1$. There are six predicates: $P_i(x, c), N_i(x, c)$, i=1,2,3, indicating that variable x occurs positively or negatively in the i^{th} position of the clause c. The vocabulary for the output structures is simply a binary predicate E standing for the Boolean adjacency matrix of the output graph. Thus $E(u, v)$ is true exactly when the edge (u, v) is present in the output graph.

The output graph consists of 6 vertices per clause and two vertices per Boolean variable, plus three additional vertices usually named T, F, and R (standing for true, false, and red). Let an arbitrary 3CNF formula be coded by an input structure,

$$\mathcal{A} \quad = \quad \langle \{0, 1, \ldots, n-1\}, P_1, P_2, P_3, N_1, N_2, N_3 \rangle$$

The output structure will be a graph with $8n + 3$ relevant vertices. The easiest way for us to code this is to use an fop of arity 2. We will assume for simplicity that n is always greater than or equal to 9.

$$I(\mathcal{A}) \;=\; \langle \{\langle a, b \rangle \,:\, 0 \leq a, b < n\}, E \rangle \;=\; \langle \{0, \ldots n^2 - 1\}, E \rangle$$

$$\text{where,} \qquad E \;=\; \{(\langle x_1, x_2 \rangle, \langle y_1, y_2 \rangle) \mid \mathcal{A} \models \varphi(x_1, x_2, y_1, y_2)\}$$

It remains to write down the first-order projection, φ. To do this, we need some nitty gritty coding. We will let the vertices T, F, and R be the elements $\langle 0, 0 \rangle, \langle 1, 0 \rangle$, and $\langle 2, 0 \rangle$ of $I(\mathcal{A})$ respectively. The formula φ will have three pieces:

$$\varphi(x_1, x_2, y_1, y_2) \;=\; \alpha(x_1, x_2, y_1, y_2) \,\vee\, \beta(x_1, x_2, y_1, y_2) \,\vee\, \beta(y_1, y_2, x_1, x_2) \vee$$

$$\gamma(x_1, x_2, y_1, y_2) \vee \gamma(y_1, y_2, x_1, x_2)$$

Where α says that there are edges between T, F, and R; β says that vertices $\langle x, 1\rangle$ and $\langle x, 2\rangle$ representing variable x and its negation are connected to each other and to R; and; γ says that for clause $C = (a \vee b \vee d)$, vertices $\langle C, 6\rangle$, $\langle C, 7\rangle$, $\langle C, 8\rangle$ are connected to each other, and the following edges exist: $(\langle C, 3\rangle, \langle C, 6\rangle)$, $(\langle C, 4\rangle, \langle C, 7\rangle)$, $(\langle C, 5\rangle, \langle C, 8\rangle)$, as well as the edges $(a, \langle C, 3\rangle), (T, \langle C, 3\rangle)$,
$(b, \langle C, 4\rangle)$, $(T, \langle C, 4\rangle)$, and $(d, \langle C, 5\rangle), (T, \langle C, 5\rangle)$.

In case anyone really wants to see them, here are the formulas written out:

$$\begin{aligned}
&\alpha(x_1, x_2, y_1, y_2) \equiv (x_2 = y_2 = 0) \wedge (x_1 \neq y_1) \wedge (x_1 \leq 2) \wedge (y_1 \leq 2) \\
&\beta(x_1, x_2, y_1, y_2) \equiv (x_2 = 1 \wedge y_2 = 2 \wedge x_1 = y_1) \vee (x_1 = 2 \wedge x_2 = 0 \wedge (1 \leq y_2 \leq 2)) \\
&\gamma(x_1, x_2, y_1, y_2) \equiv (x_1 = x_2 = 0 \wedge (3 \leq y_1 \leq 5) \wedge (3 \leq y_2 \leq 5) \\
&\qquad \vee \quad (x_1 = y_1 \wedge (3 \leq x_2 \leq 5) \wedge (y_2 = x_2 + 3)) \\
&\qquad \vee \quad [(x_2 = 1 \wedge y_2 = 3) \wedge P_1(x_1, y_1)] \vee [(x_2 = 2 \wedge y_2 = 3) \wedge N_1(x_1, y_1)] \\
&\qquad \vee \quad [(x_2 = 1 \wedge y_2 = 4) \wedge P_2(x_1, y_1)] \vee [(x_2 = 2 \wedge y_2 = 4) \wedge N_2(x_1, y_1)] \\
&\qquad \vee \quad [(x_2 = 1 \wedge y_2 = 5) \wedge P_3(x_1, y_1)] \vee [(x_2 = 2 \wedge y_2 = 5) \wedge N_3(x_1, y_1)] \quad \vee \\
&\qquad (x_1 = y_1 \wedge x_2 \neq y_2 \wedge (6 \leq x_2 \leq 8) \wedge (6 \leq y_2 \leq 8))
\end{aligned}$$

□

4 Main Theorem and Proof

Theorem 4.1 *Let $\mathcal{C}$ be a nice complexity class, e.g.,* L, NL, P, NP, *etc. Let S and T be complete for $\mathcal{C}$ via first-order projections. Then S and T are isomorphic via a first-order isomorphism.*

To prove Theorem 4.1 we begin with the following lemma. Note the similarity between Lemma 4.2 and the proofs of Theorems 2.4 and 2.8. For simplicity in this lemma we are assuming that I is a single fop that maps STRUC$[\sigma]$ to itself. The proof for the case with two fops and two vocabularies as in Lemma 4.5 is similar.

Lemma 4.2 *Let I be an fop that is 1:1 and of arity greater than or equal to two (i.e. it at least squares the size). Then the following two predicates are first-order expressible concerning a structure $\mathcal{A}$:*

a. IE$(\mathcal{A})$, *meaning that $I^{-1}(\mathcal{A})$ exists.*

b. #Ancestors$(\mathcal{A}, r)$, meaning that the length of $\mathcal{A}$'s maximal ancestor chain is r.

Proof Let $I = \lambda_{x_1 \ldots x_d}\langle \varphi_1, \ldots, \varphi_r, t_1, \ldots, t_s\rangle$, where each φ_i is in the form of Equation 3.4. To prove (a) just observe that each bit of the relation R_i of $\mathcal{A}$ either (1) depends on exactly one bit of some pre-image $\mathcal{B}$ (specified by an occurrence of a literal λ_{ij} in φ_i), or (2) it doesn't depend on any bit of a pre-image. In case (2) a given bit of $\mathcal{A}$ is either "right" or "wrong." Thus, $\mathcal{A}$ has an inverse iff no bit of $\mathcal{A}$ is wrong, and no pair of bits from $\mathcal{A}$ are determined by the same bit of $\mathcal{A}$'s preimage in conflicting ways. We can check this in a first-order way by checking that for all pairs of bits from $\mathcal{A}$: $R_i(\bar{a})$ and $R_{i'}(\bar{b})$, either they do not depend on the same bit from $\mathcal{B}$, or the same value of that bit gives the correct answer for $R_i(\bar{a})$ and $R_{i'}(\bar{b})$. Furthermore, the preimage $\mathcal{B}$ if it exists can be described uniquely by a first-order formula that chooses the correct bits determined by entries of $\mathcal{A}$. **N.B.** Since we have assumed that I is 1:1 every bit of $I^{-1}(\mathcal{A})$ is determined by some bit of $\mathcal{A}$.

(b) To express $\#Ancestors(\mathcal{A}, r)$, we want to describe the existence of an Ancestor Chain:

$$\mathcal{A}_r \xrightarrow{I} \mathcal{A}_{r-1} \xrightarrow{I} \cdots \xrightarrow{I} \mathcal{A}_1 \xrightarrow{I} \mathcal{A}_0 = \mathcal{A} \tag{4.3}$$

We will then assert that this is the maximal length such chain, i.e.,

$$\neg\mathrm{IE}(\mathcal{A}_r) \wedge (\forall k < r)\mathrm{IE}(\mathcal{A}_k) \tag{4.4}$$

Equation 4.4 expresses the existence of the ancestor chain 4.3 inductively in the following sense: Once we know that $\mathcal{A}_k$ exists, we can ascertain the value $\mathcal{A}_k[[p_k]]$ of the bit at position p_k of $\mathcal{A}_k$, by exhibiting a certificate:

$$C(k, p_k) \;=\; \langle(\mathcal{A}_k[[p_k]], p_k), (\mathcal{A}_{k-1}[[p_{k-1}]], p_{k-1}), \ldots, (\mathcal{A}_0[[p_0]], p_0)\rangle$$

We can say in a first-order sentence that $C(k, p_k)$ is internally consistent. That is, for all i with $k > i \geq 0$, bit p_{i+1} of $\mathcal{A}_{i+1}$ is determined correctly via I by bit p_i of $\mathcal{A}_i$.[4] Note that because each structure $\mathcal{A}_{i+1}$ is of size at most the square root of the size of $\mathcal{A}_i$, the certificate requires only $O(\log n)$ bits, i.e., a constant number of variables, to express.

Thus, in Equation 4.4, we refer to bit p_k of the structure $\mathcal{A}_k$ by existentially quantifying an internally consistent certificate $C(k, p_k)$. We know inductively, that since $\mathrm{IE}(\mathcal{A}_{k-1})$, the bit value determined by $C(k, p_k)$ is unique and correct. $\square$

Lemma 4.5 *If S and T are interreducible via 1:1 fops I and J each of arity at least two, then S and T are isomorphic via first-order isomorphisms.*

Proof Let $\mathcal{A}$ be a structure in the vocabulary of S, and, as in the proof of Theorem 2.3 define the length of the ancestor chain of $\mathcal{A}$ to be the length of the longest sequence of the form $J^{-1}(\mathcal{A})$, $I^{-1}(J^{-1}(\mathcal{A}))$, $J^{-1}(I^{-1}(J^{-1}(\mathcal{A})))$, ... The argument given in Lemma 4.2 shows that there is a formula $\#Ancestors(\mathcal{A}, r)$ that evaluates to true iff $\mathcal{A}$'s ancestor chain has length r. Lemma 4.2 also shows that there is a formula computing J^{-1}. The desired isomorphism is now the function b such that the i-th bit of $b(\mathcal{A})$ is one iff the following first-order formula is true:

$$(\exists r)(\#Ancestors(\mathcal{A}, r) \wedge (\mathrm{BIT}(0, r) \wedge I(i)) \vee (\neg\mathrm{BIT}(0, r) \wedge J^{-1}(i)))$$

[4] The reader who is more familiar with bit hacking on Turing machines than with first-order formulas, could instead convince herself that this can be done by an alternating Turing machine running in logarithmic time and making $O(1)$ alternations; first-order expressibility follows by [BIS].

(Note that this first-order isomorphism b is not, strictly speaking, a first-order interpretation, since it maps some inputs to strictly shorter outputs, which is impossible for an interpretation.) $\square$

It now remains to show,

Lemma 4.6 *Suppose that a problem S is complete via fops for a nice complexity class, $\mathcal{C}$. Then S is complete for $\mathcal{C}$ via fops that are 1:1 and of arity at least two.*

Proof Of course it remains to define "nice", but here is the proof. Every nice complexity class has a universal complete problem:

$$U_{\mathcal{C}} \;=\; \{M\$w\#^r \mid M(w)\downarrow \text{ using resources } f_{\mathcal{C}}(r)\} \tag{4.7}$$

Here $f_{\mathcal{C}}(r)$ defines the appropriate complexity measure, e.g. r nondeterministic steps for NP, deterministic space $\log r$, for L, space 2^r for EXPSPACE, etc.

We claim that $U_{\mathcal{C}}$ is complete for $\mathcal{C}$ via fops that are 1:1 and of arity at least two . In order to make this claim, we need to agree on an encoding of inputs to $U_{\mathcal{C}}$ that allows us to interpret them as structures over some vocabulary. Since all of our structures are encoded in binary, we will encode \$ and # by 10 and 11 respectively, and the binary bits 0 and 1 constituting M and w will be encoded by 00 and 01 respectively. Now, as in, for example, [I87], we consider a binary string of length n to be a structure with a single unary predicate over a universe of size n. Now for any given problem $T \in \mathcal{C}$ accepted by machine M, we show that T is reducible to $U_{\mathcal{C}}$ via a fop that is 1:1 and of arity at least two. The fop simply maps input w to the string $M\$w\#^r$, for an appropriate r which we can always take to be at least $|w|^2$. The fop checks that if $i \leq 2|M|$ then the odd-numbered bits are 0 and if i is even, then the i^{th} bit is 1 iff the $i/2$-th bit of M is 1. Similarly, if $2|M|+2 < i \leq 2(|M|+|w|+1)$ then the odd-numbered bits are 0 and the even numbered bits are the corresponding bit of w, etc.

To complete the proof of the lemma, let T be any problem in $\mathcal{C}$ and let S be as above. Then we reduce T to S via a 1:1, length squaring fop as follows. First reduce T to $U_{\mathcal{C}}$ as above. Next reduce $U_{\mathcal{C}}$ to S via the fop promised in the statement of the lemma.

It is easy to verify that, using the encoding we have chosen for $U_{\mathcal{C}}$, it holds that for every length n, for all $i \leq n$, there are two strings x and y of length n, differing only in position i, such that $x \in U_{\mathcal{C}}$ and $y \notin U_{\mathcal{C}}$.

Thus the fop from $U_{\mathcal{C}}$ cannot possibly ignore any of the bits in its input. But an fop cannot process several bits into one, it can only either ignore a bit or copy it, or negate it; and this choice is made independently of the values of any of the bits.

It follows that the composition of these two fops is the 1:1 length squaring fop that we desire. (Note that an fop by definition must have arity at least one and thus cannot be length decreasing on Boolean strings.) $\square$

From the above three lemmas we have a first-order version of Theorem 2.3 and thus Theorem 4.1 follows.

We can inspect the proof of Lemma 4.6 to get a definition of "nice". A complexity class is "nice" if it has a universal complete problem via fops as in Equation 4.7. It is easy to check that the following complexity classes, among many others, are nice and thus meet the conditions of Theorem 4.1.

Proposition 4.8 *The following complexity classes are nice:* NC^1, L, NL, LOG(CFL), NC^2, P, NP, PSPACE, EXPTIME, EXPSPACE.

Proof This is immediate for the Turing machine based classes: L, NL, P, NP, PSPACE, EXPTIME, EXPSPACE. It similarly follows for the other three classes using the definitions: $\mathrm{NC}^i = \mathrm{ASPACE}[\log n]-\mathrm{TIME}[(\log n)^i]$, and $\mathrm{LOG(CFL)} = \mathrm{ASPACE}[\log n]-\forall\mathrm{TIME}[\log n]$. $\square$

5 More on the Relationship between Isomorphisms and Projections

There are several questions about isomorphisms among complete sets that can be answered in the setting of first-order computation but are open for general polynomial-time computation. It is not known whether one-way functions exist since their existence would imply that $\mathrm{P} \neq \mathrm{NP}$. However, if one-way functions exist (i.e., if $\mathrm{P} \neq \mathrm{UP}$) then there exists a one-way function f such that $f(\mathrm{SAT})$ is polynomial-time isomorphic to SAT [Ga].

Here we can be more definitive: the bijection $f(x) = 3x \pmod{2^{|x|}}$ was shown in [BL] to be one-way for first-order computation, in the sense that f is first-order expressible, but f^{-1} is not. (See also [Hås] for other examples.) However, it is not too hard to show that for this choice of f, $f(SAT)$ is complete for NP under first-order projections, and thus it is first-order isomorphic to SAT.

The next result shows that the class of sets complete under first-order projections is not closed under first-order isomorphisms. (This also seems to be the first construction of a set that is complete for NP under first-order (or even poly-time) many-one reductions, that is not complete under first-order projections.)

Theorem 5.1 *There is a set first-order isomorphic to SAT that is not complete for NP under first-order projections.*

Proof Let $g(x)$ be a string of $|x|^2$ bits, with bit $x_{i,j}$ representing the logical AND of bits i and j of x. Let $A = \{\langle x, g(x)\rangle : x \in \mathrm{SAT}\}$. By an extension of the techniques used in proving Theorem 4.1, it can be shown that A is first-order isomorphic to SAT. However a direct argument shows that there cannot be any projection (even a nonuniform projection) from SAT to A. (Sketch: For all n, one can find bit positions i and j that are independent of each other and are independent of every other bit position, in the sense that for any setting b of bit j there are two words that differ only in bit i, having b in position j, such that one of the words is in SAT and one is not. No projection reducing SAT to another language can "ignore" either i or j. But since i and j are independent of all other bit positions, no projection can encode the AND of bits i and j.) $\square$

A natural question that remains open is the question of whether every set complete for NP under first-order many-one reductions is first-order isomorphic to SAT. A related question is whether one can construct a set complete for NP under poly-time many-one reductions that is not first-order isomorphic to SAT. Since so many tools are available for proving the limitations of first-order computation, we are optimistic that this and related questions about sets complete under first-order reductions should be tractable.[5] Furthermore, we hope that insights gleaned in answering these questions will be useful in guiding investigations of the polynomial-time degrees.

Acknowledgments The authors wish to thank the organizers of the 1992 Seminar on Structure and Complexity Theory at Schloß Dagstuhl, where this work was initiated. We also thank Richard Beigel, Jose Antonio Medina, and two anonymous referees for comments on an earlier draft.

References

[AB] Manindra Agrawal and Somenath Biswas, "Polynomial Isomorphism of 1-L-Complete Sets," *Eighth Annual Structure in Complexity Theory Symp.* (1993), 75-79.

[A88] Eric Allender, "Isomorphisms and 1-L Reductions," *J. Computer Sys. Sci.* **36** (1988), 336-350.

[A89] Eric Allender, "P-Uniform Circuit Complexity," *JACM* **36** (1989), 912-928.

[AG91] Eric Allender and Vivek Gore, "On Strong Separations from AC^0," in *Advances in Computational Complexity Theory*, Jin-Yi Cai, ed., DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 13, AMS Press, 1993, pp. 21–37.

[AG92] Eric Allender and Vivek Gore, "A Uniform Circuit Lower Bound for the Permanent," *SIAM J. Comput.* **23** (1994) 1026–1049.

[BIS] David Mix Barrington, Neil Immerman, Howard Straubing, "On Uniformity Within NC^1," *J. Computer Sys. Sci.* **41** (1990), 274-306.

[BH77] Len Berman and Juris Hartmanis, "On Isomorphism and Density of NP and Other Complete Sets," *SIAM J. Comput.* **6** (1977), 305-322.

[BL] Ravi Boppana and Jeff Lagarias, "One-Way Functions and Circuit Complexity," *Information and Computation* **74**, (1987), 226-240.

[BH90] Hans-Jörg Burtschick and Albrecht Hoene, "The degree structure of 1-L reductions," *Proc. Math. Foundations of Computer Science*, Lecture Notes in Computer Science 629, Springer-Verlag, 1992, 153–161.

[CSV] Ashok Chandra, Larry Stockmeyer, and Uzi Vishkin, "Constant Depth Reducibility," *SIAM J. Comput.* **13**, (1984), 423-439.

[5] One possible approach might be to attempt to construct a first-order analog of the "scrambling" and "annihilating" functions studied in [KMR89]. However, we suspect that this particular approach is likely to be difficult, since this would involve constructing sets having a sort of "immunity" property relative to AC^0. Related problems (although not precisely this problem) were shown in [AG91] to imply the solution to some longstanding open questions in complexity theory.

[Coo] Stephen Cook, "The Complexity of Theorem Proving Procedures," *Proc. Third Annual ACM STOC Symp.* (1971), 151-158.

[Dah] Elias Dahlhaus, "Reduction to NP-Complete Problems by Interpretations," in *Logic and Machines: Decision Problems and Complexity,* Börger, Rödding, and Hasenjaeger eds., Lecture Notes In Computer Science 171, Springer-Verlag, 1984, 357-365.

[End] Herbert Enderton, *A Mathematical Introduction to Logic,* Academic Press, 1972.

[Fa] Ron Fagin, "Generalized First-Order Spectra and Polynomial-Time Recognizable Sets," in *Complexity of Computation,* ed. R. Karp, *SIAM-AMS Proc.,* 7 (1974) 43–73.

[FSS] Merrick Furst, James Saxe, and Michael Sipser, "Parity, Circuits, and the Polynomial-Time Hierarchy," *Math. Systems Theory* **17** (1984), 13-27.

[Ga] K. Ganesan, "One-way Functions and the Isomorphism Conjecture," *Theoret. Comp. Sci.* , 129 (1994), 309-321.

[GJ] Michael R. Garey and David S. Johnson, *Computers and Intractability,* Freeman, 1979.

[Hås] Johan Håstad, "One-Way Permutations in NC^0," *Information Processing Letters* **26** (1987), 153-155.

[Har] Juris Hartmanis, "On the logtape isomorphism of complete sets,*Theoret. Comp. Sci.* 7 (1978), 273-286.

[HIM] Juris Hartmanis, Neil Immerman, and Stephen Mahaney, "One-Way Log Tape Reductions," *19th IEEE FOCS Symp.* (1978), 65-72.

[HH] Lane Hemachandra and A. Hoene, "Collapsing Degrees Via Strong Computation," *J. of Computer Sys. Sci.* **46** (1993), 363–380.

[I87] Neil Immerman, "Languages That Capture Complexity Classes," *SIAM J. Comput.* **16**, (1987), 760-778.

[I89] Neil Immerman, "Descriptive and Computational Complexity," *Computational Complexity Theory,* ed. J. Hartmanis, *Proc. Symp. in Applied Math.,* 38, American Mathematical Society (1989), 75-91.

[IL] Neil Immerman, Susan Landau, "The Complexity of Iterated Multiplication," *Information and Computation* (116:1) (1995), 103-116.

[Jon] Neil Jones, "Space-Bounded Reducibility among Combinatorial Problems," *J. Computer Sys. Sci.* **11** (1975), 68–85.

[JY] Deborah Joseph and Paul Young, "Some Remarks on Witness Functions for Non-polynomial and Non-complete sets in NP," *Theoretical Computer Science* **39** (1985), 225-237.

[Kar] Richard Karp, "Reducibility Among Combinatorial Problems," in *Complexity of Computations,* R.E.Miller and J.W.Thatcher, eds. (1972), Plenum Press, 85-104.

[Kel] John L. Kelley, *General Topology,* 1955, Van Nostrand Reinhold.

[KMR89] Stuart Kurtz, Stephen Mahaney, James Royer, "The Isomorphism Conjecture Fails Relative to a Random Oracle," *21st ACM STOC Symp.* (1989), 157-166.

[KMR90] Stuart Kurtz, Stephen Mahaney, James Royer, "The Structure of Complete Degrees," in *Complexity Theory Retrospective* (Alan Selman, Ed.), Springer-Verlag, 1990, pp. 108-146.

[Man] Udi Manber, *Introduction to Algorithms: A Creative Approach,* Addison-Wesley, 1989.

[MI] J. Antonio Medina and Neil Immerman, " A Syntactic Characterization of NP-Completeness," *IEEE Symp. Logic In Comput. Sci.* (1994), 241-250.

[Myh] John Myhill, "Creative Sets," *Zeitschrift für mathematische Logik und Grundlagen der Mathematik,* **1** (1955), 97-108.

[SV] Sven Skyum and Leslie Valiant, "A Complexity Theory Based on Boolean Algebra," *JACM,* 32, No. 2, April, 1985, (484-502).

[Ste] Iain Stewart, "Using the Hamiltonian Operator to Capture NP," *J. Comput. Sys. Sci. , 45* (1992), 127–151.

[Val] Leslie Valiant, "Reducibility By Algebraic Projections," *L'Enseignement mathématique,* **28**, 3-4 (1982), 253-68.

[You] Paul Young, "Juris Hartmanis: Fundamental Contributions to Isomorphism Problems," in *Complexity Theory Retrospective,* Alan Selman, ed., Springer-Verlag (1990), 28-58.