A first-order isomorphism theorem

E. Allender
J. Balcázar
N. Immerman

Report LSI-92-31-R
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 01003
immerman@cs.umass.edu

CLASSIFICATION: Computational Complexity

Abstract

We show that for most complexity classes of interest, all sets complete under first-order projections are isomorphic under first-order isomorphisms. That is, a very restricted version of the Berman-Hartmanis Conjecture holds.

1 Introduction

In 1977 Berman and Hartmanis noticed that all NP complete sets that they knew of were polynomial-time isomorphic, [BH]. They made their now-famous

*Supported in part by National Science Foundation grant CCR-9204874.
1Supported in part by ESPRIT-ll BRA EC project 3075 (ALCOM) and by Acción Integrada Hispano-Alemana 131 B
2Supported by NSF grants CCR-9008416 and CCR-9207797.
isomorphism conjecture: namely that all NP complete sets are polynomial-time isomorphic. This conjecture has engendered a large amount of work (cf. [KMR, 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 [Coo71], 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 remain complete via fops, cf. [Imm87, IL, Ste].

On the other hand, Joseph and Young, [JY] have pointed out that polynomial-time, many-one reductions are 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.\footnote{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.}

We feel on the other hand, that the problem is simply that the choice of reduction at the time was made for historical rather than scientific reasons. 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 structural sufficient 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 [AG].

Preliminary results and background on isomorphisms follows in Section 2. Definitions and background on descriptive complexity is 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.

2 Short History of the Isomorphism Conjecture

Definition 2.1 For $A, B \subseteq \Sigma^*$, we say that $A$ and $B$ are p-isomorphic ($A \cong B$) iff there exists a bijection $f \in FP$ with inverse $f^{-1} \in FP$ such that $A \leq_m B$ via $f$ (and therefore $B \leq_m A$ via $f^{-1}$).

Observation 2.2 ([BH]) 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 (Schröder-Bernstein Theorem) 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 $f : A \rightarrow B$ and $g : B \rightarrow A$ 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 $a$ is an ancestor of $c$ iff we can reach $c$ by a finite number of applications of the functions $f$ and/or $g$. Now we can define a bijection $h : A \rightarrow B$ 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 ([BH]) Let $f : A \leq_m B$ and $g : B \leq_m A$, where $f$ and $g$ are 1:1, length-increasing functions. Assume that $f, f^{-1}, g, g^{-1} \in FP$ where $f^{-1}, g^{-1}$ are inverses of $f, g$. Then $A \cong B$.

Proof This follows from the proof of Theorem 2.3 because in this case all ancestor chains are at most linear in length.

Consider the following definition:

Definition 2.5 ([BH]) We say that the language $A \subseteq \Sigma^*$ has p-time padding functions iff there exist $e, d \in FP$ such that
1. For all \( w, x \in \Sigma^* \), \( w \in A \iff e(w, x) \in A \)

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

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

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

\[
e(w, x) = (w) \land c_1 \land c_2 \land \cdots \land c_{|x|}
\]

where \( c_i \) is \( (y \lor y) \) if the \( i \)th bit of \( x \) is 1 and \( (y \lor z) \) otherwise, where \( y \) and \( z \) are boolean variables numbered higher than all the boolean variables occurring in \( w \).

Then the following theorem follows from Theorem 2.4:

**Theorem 2.6 ([BH])** If \( A \) and \( B \) are \( NP \) complete and have \( p \)-time padding functions, then \( A \models B \).

Finally, Observation 2.2 now follows from the following:

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

Berman and Hartmanis also extended the above work as follows: Say that \( A \) has \textit{logspace-padding functions} if there are logspace computable functions as in Definition 2.5.

**Theorem 2.8 ([BH])** 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.

There is not space here to give a more extensive survey of work that has been done on the isomorphism conjecture; the interested reader should consult [KMR]. We should, however, mention one paper that is particularly relevant.

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 very 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 [All88].) Note that it is easy to prove that the class of 1-L reductions is incomparable with the class of first-order projections.
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 [Imm89] 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:

\[
\mathcal{A}(w) = (\{0, 1, \ldots, n-1\}, R)
\]

where the unary relation \( R(x) \) holds in \( \mathcal{A}(w) \) (in symbols, \( \mathcal{A}(w) \models R(x) \)) just if \( x \) 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 \( \tau = (R_1^1, \ldots, R_r^r, c_1, \ldots, c_s) \) is a tuple of input relation and constant symbols. Let \( \text{STRUC}[\tau] \) denote the set of all finite structures of vocabulary \( \tau \). We define a complexity theoretic problem to be any subset of \( \text{STRUC}[\tau] \) for some \( \tau \).

For any vocabulary \( \tau \) there is a corresponding first-order language \( \mathcal{L}(\tau) \) built up from the symbols of \( \tau \) and the logical relation symbols and constant symbols\(^2\): \( =, \leq, s, \text{BIT}, 0, \text{m} \), using logical connectives: \( \land, \lor, \neg \), 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 A \( k \)-ary projection from \( S \) to \( T \) is a sequence of maps \( \{p_n\} \), \( n = 1, 2, \ldots \), such that for all \( n \) and for all binary strings \( s \) of length \( n \), \( p_n(s) \) is a binary string of length \( n^k \) and,

\[
s \in S \iff p_n(s) \in T.
\]

Let \( s = s_0s_1 \ldots s_{n-1} \). Then each map \( p_n \) is defined by a sequence of \( n^k \) literals:

\[
\{I_0, I_1, \ldots, I_{n^{k-1}}\}
\]

where

\[
I_i \in \{0, 1\} \cup \{s_j, \overline{s}_j \mid 0 \leq j \leq n-1\}.
\]

\(^2\)Here \( \leq \) refers to the usual ordering on \( \{0, \ldots, n-1\} \), \( s \) is the successor relation, \( \text{BIT} \) is described below, \( m \) and \( n \) refer to \( 0, n-1 \), respectively. Some of these are redundant, but useful for quantifier-free interpretations. For simplicity we will assume throughout that \( n > 1 \) and thus \( 0 \neq m \). Sometimes the logical relations are called "numeric" relations. For example, "\( \text{BIT}(i,j) \)" (meaning the \( i \)th bit of the binary representation of \( j \) is 1) and "\( i \leq j \)" describe the numeric values of \( i \) and \( j \) and do not refer to any input predicates.
Thus as $s$ ranges over strings of length $n$, each bit of $p_n(s)$ depends on at most one bit of $s$,

$$p_n(s)[i] = l_i(s)$$

Projections were originally defined as a non-uniform sequence of reductions — one for each value of $n$. We now define first-order projections, which are a uniform version of Valiant’s projections. The idea of our definition is that the choice of the literals $\{l_0, l_1, \ldots, l_{n+1}\}$ 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, [Imm97]. (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 $\sigma$ and $\tau$ be two vocabularies, with $\tau = \langle R^{\sigma}_1, \ldots, R^{\sigma}_r, c_1, \ldots, c_k \rangle$. Let $S \subseteq \text{STRUC}[\sigma]$, $T \subseteq \text{STRUC}[\tau]$ 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 $\varphi_i$ are a subset of $\{x_1, \ldots, x_{k+n}\}$. Finally, suppose we are given an $s$-tuple of closed terms $t_1, \ldots, t_s$ from $\mathcal{L}(\sigma)$. Let $I = \lambda x_1, \ldots, x_s (\varphi_1, \ldots, \varphi_r, t_1, \ldots, t_s)$ be a tuple of these formulas and closed terms. (Here $d = \max_i (ka_i)$.)

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

$$\bar{I}(A) = (\{0, \ldots, n^k - 1\}, R_1, \ldots, R_r)$$

where the relation $R_i$ is determined by the formula $\varphi_i$, for $i = 1, \ldots, r$. More precisely, let the function $\langle \cdot, \ldots, \cdot \rangle: |A|^k \to |\bar{I}(A)|$ be given by

$$\langle u_1, u_2, \ldots, u_k \rangle = u_k - u_{k-1}n + \cdots + u_1n^{k-1}$$

Then,

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

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

---

A closed term is an expression involving constants and function symbols. This is as opposed to an open term which also has free variables. In this paper, since we do not have function symbols, closed terms are synonymous with constant symbols. Note that a more general way to interpret constants and functions is via a formula $\varphi$ such that $\mathcal{F} \rightarrow (\forall \bar{x})(\exists \bar{y})\varphi(\bar{x}, \bar{y})$. However, in this paper the simpler definition using closed terms suffices.
Suppose that \( \hat{I} \) is a many-one reduction from \( S \) to \( T \), i.e. for all \( A \) in \( \text{STRUC}[\sigma] \),
\[
A \in S \implies \hat{I}(A) \in T
\]
Then we say that \( \hat{I} \) is a \( k \)-ary first-order interpretation of \( S \) to \( T \).

Note that \( I \) induces a map which we will also call \( I \) from \( L(\tau) \) to \( L(\sigma) \). For \( \varphi \in L(\tau) \), \( I(\varphi) \) is the result of replacing all relation and constant symbols in \( \varphi \) by the corresponding formulas and close terms in \( I \). Note that if \( I \) is a \( k \)-ary interpretation then each variable in \( \varphi \) is replaced by a \( k \)-tuple of variables. Furthermore, the logical relations \( \leq, = \) are replaced by the corresponding quantifier-free formulas on \( k \)-tuples ordered lexicographically. For example, with \( k = 2 \), an occurrence of the successor relation, \( s(x, y) \), would be replaced by
\[
I(s(z, y)) = (x_1 = y_1 \land s(z_2, y_2)) \lor (x_2 = m \land y_2 = 0 \land s(x_1, y_1))
\]
The logical constants, \( 0, m \), are replaced by \( k \)-tuples of the same constants.

Note that the logical relation \( \text{BIT} \) when mapped to \( k \)-tuples cannot be easily replaced by a quantifier-free formula. However, \( \text{BIT} \) on tuples is definable in \( \text{FO} \) (with \( \text{BIT} \)), cf. [Lin].

It follows immediately from the definitions that:

**Proposition 3.3** Let \( \sigma, \tau, \text{ and } I \) be as in Definition 3.2. Then for all sentences \( \varphi \in L(\tau) \) and all structures \( A \in \text{STRUC}[\sigma] \),
\[
A \models I(\varphi) \iff \hat{I}(A) \models \varphi
\]

We are now ready to define first-order projections, a syntactic restriction of first-order interpretations.

If each formula in the first-order interpretation \( \hat{I} \) satisfies this syntactic condition then it follows that \( \hat{I} \) is also a projection in the sense of Valiant. In this case we call \( \hat{I} \) a first-order projection.

**Definition 3.4 (First-Order Projections)** Let \( \hat{I} \) be a \( k \)-ary first-order interpretation from \( S \) to \( T \) as in Definition 3.2. Let \( I = (\varphi_1, \ldots, \varphi_t, t_1, \ldots, t_s) \).

Suppose further that the \( \varphi_i \)'s all satisfy the following projection condition:
\[
\varphi_i \equiv \alpha_1 \lor (\alpha_2 \land \lambda_2) \lor \cdots \lor (\alpha_s \land \lambda_s)
\]
where the \( \alpha_j \)'s are mutually exclusive formulas in which no input relations occur, and each \( \lambda_j \) is a literal, i.e. an atomic formula \( P(x_{j_1}, \ldots, x_{j_s}) \) or its negation.

In this case the predicate \( R_i((u_{i_1}, \ldots, u_{i_t}), (x_{j_1}, \ldots, x_{j_s})) \) holds in \( \hat{I}(A) \) if \( \alpha_1(u) \) is true, or if \( \alpha_j(u) \) is true for some \( 1 < j \leq t \) and the corresponding literal \( \lambda_j(u) \) holds in \( A \). Thus each bit in the binary representation of \( \hat{I}(A) \) is determined by at most one bit in the binary representation of \( A \). We say that \( \hat{I} \) is a first-order projection. Write \( S \leq_{\text{fo}} T \) to mean that \( S \) is reducible to \( T \) via a first-order projection.
4 Main Theorem and Proof

Theorem 4.1 Let $C$ be a nice complexity class, e.g., $L$, $NL$, $P$, $NP$, etc. Let $S$ and $T$ be complete for $C$ 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 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 are expressible concerning a structure $A$:

a. $IE(A)$, i.e., $\tilde{I}^{-1}(A)$ exists.

b. $\#\text{Ancestors}(A, r)$, i.e., the length of $A$'s maximal ancestor chain is $r$.

Proof. Let $I = \lambda_{x_1, \ldots, x_r}(\varphi_1, \ldots, \varphi_r, t_1, \ldots, t_r)$, where each $\varphi_i$ is in the form of Equation 3.5. To prove (a) just observe that each bit of the relation $R_i$ of $A$ either (1) depends on exactly one bit of some pre-image $B$ (specified by an occurrence of a literal $\lambda_{1j}$ in $\varphi_i$), or (2) it doesn't depend on any bit of a pre-image. In case (2) a given bit of $A$ is either "right" or "wrong." Thus, $A$ has an inverse iff no bit of $A$ is wrong, and no pair of bits from $A$ are determined by the same bit of $A$'s preimage in conflicting ways. We can check this in a first-order way by checking that for all pairs of bits from $A$: $R_i(\bar{a})$ and $R_i(\bar{b})$, either they do not depend on the same bit from $B$, or the same value of that bit gives the correct answer for $R_i(\bar{a})$ and $R_i(\bar{b})$. Furthermore, the preimage $B$ if it exists can be described uniquely by a first-order formula that chooses the correct bits determined by entries of $A$. Since we have assumed that $\tilde{I}$ is 1:1 every bit of the preimage is determined. This observation is important enough that is should be repeated: every bit of $\tilde{I}^{-1}(A)$ is determined by some bit of $A$.

(b) In order to define $\#\text{Ancestors}(A, r)$, we will first define the related predicate $\#\text{Ancestors}(A, r)$ which will evaluate to true iff the implication

$A$ has at least $r$ ancestors implies $A$ has at least $r + 1$ ancestors.

is true. Note that $\#\text{Ancestors}(A, r)$ is equivalent to

$(r = 0 \land \neg IE(A)) \lor$

$(r > 0 \land IE(A) \land (\forall t \leq r - 1)\#\text{Ancestors}(A, t) \land \neg\#\text{Ancestors}(A, r))$.

Let us use the notation $A_i(\bar{b})$ to denote the $b$-th bit of the string encoding the structure $A_i$. If the assumption that the number of ancestors of $A$ is at least $r$ is true, this means that there is a chain:

$A_r \leftarrow A_{r-1} \ldots \leftarrow A_1 \leftarrow A_0 = A$
where each $A_t$ is a structure over a domain of size $n^{1/k^t}$. If this chain exists, that means that for each bit $b_t$ of $A_t$ there is a certificate of the form

$$C = \langle (A_r(b_r), b_r), (A_{r-1}(b_{r-1}), b_{r-1}), \ldots, (A_0(b_0), b_0) \rangle$$

where this certificate has the meaning that for each $t$, bit $b_t$ of $A_t$ is determined by bit $b_{t-1}$ of $A_{t-1}$, and furthermore bit $A_0(b_0)$ agrees with bit $b_0$ of the input structure $A$.

The predicate $\#Anc.induc(A, r)$ is equivalent to saying that if this sequence exists, then $\exists b_r$. Just as in part (a), this is equivalent to saying that for all $b_r$, $\leq$ the number of bits in the description of $A_r$, either (1) there exists a certificate for bit $b_r$, showing that it is "right", or (2) there exists a certificate of the form

$$C = \langle (A_{r+1}(b_{r-1}), b_{r+1}), (A_r(b_r), b_r), \ldots, (A_0(b_0), b_0) \rangle$$

such that bit $\bar{A}$ defines bit $b_r$ of $A_r$ in terms of bit $b_{r+1}$ of $A_{r+1}$, and for all pairs of bits $b_r, b_r$, there do not exist certificates showing that bits $A_r(b_r)$ and $A_r(b_r)$ depend on $A_{r+1}(b_{r-1})$ in contradictory ways.

The fact that $\bar{A}$ squares the size of structures implies that the size of the certificate $C$, fits in $O(\log n)$ bits, i.e., a bounded number of first-order variables. Furthermore, it is easy to see that the problem of determining if $C$ is a legal certificate for $A$ is first-order expressible. (All that is required is check that for all $j \leq |C|$, if $j$ is the start of a field of $C$, then the information in this field is locally consistent with $\bar{A}$. It is clear 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].) It now follows that $\#Anc.induc(A, r)$ is expressible in first-order.

It now remains to show,

**Lemma 4.3** Suppose that a problem $S$ is complete via fops for a nice complexity class, $C$. Then $S$ is complete via 1:1 length squaring fops for $C$.

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

$$U_C = \{ M \uparrow \downarrow, M(w) \downarrow \text{ using resources } f_c(r) \}$$

here $f_c(r)$ defines the appropriate complexity measure, e.g. $r$ nondeterministic steps for NP, deterministic space $\log r$, for L, etc.

We claim that $U_C$ is complete via 1:1 length squaring fops for $C$. In order to make this claim, we need to agree on an encoding of inputs to $U_C$ 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, [Imm87], 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 $T \in C$ accepted by machine $M$, the fop simply checks that if $i \leq 2|M|$ then the odd-numbered bits are 0 and bit number $2i$ is the $i$-th bit of $M$, and 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 $C$ 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 $U_C$ as per the claim. Next reduce $U_C$ 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 $U_C$, it holds that for every length $n$, for all $i \leq n$, there are two strings $z$ and $y$ of length $n$, differing only in position $i$, such that $z \in U_C$ and $y \notin U_C$. Thus the fop from $U_C$ cannot possibly ignore any of the bits in its input. 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.)

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

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 that are open for general polynomial-time computation. For example, given a one-way function $f$, it is an open question if $f(SAT)$ can be poly-time isomorphic to SAT. In part, this remains open because there is no function $f$ that can be proven to be one-way at this time. On the other hand, the bijection $f(x) = 3x \pmod{2^{|x|}}$ was shown in [BL] to be a one-way for first-order computation, in the sense that $f$ is first-order expressible, but $f^{-1}$ is not. (See also [Há] for other examples.) However, it 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 $A = \{(x, g(x)) : x \in \text{SAT}\}$. 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 $A$.

A natural question that remains open is the question of whether every set complete for $\text{NP}$ under first-order many-one reductions is first-order isomorphic to SAT. A related question is whether one can construct a set complete for $\text{NP}$ 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. 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.

References


