DSPACE(n) $\stackrel{?}{=}$ NSPACE(n): A Degree Theoretic Characterization

Manindra Agrawal
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur 208016, INDIA
email: manindra@iitk.ernet.in

Abstract

It is shown that the following are equivalent.

1. DSPACE(n) = NSPACE(n).
2. There is a non-trivial $\leq_m^{1-NL}$-degree that coincides with a $\leq_m^{1-L}$-degree.
3. For every class $\mathcal{C}$ closed under log-lin reductions, the $\leq_m^{1-NL}$-complete degree of $\mathcal{C}$ coincides with the $\leq_m^{1-L}$-complete degree of $\mathcal{C}$.

1 Introduction

The structure of complete degrees has been extensively investigated over the years. Particular attention has been paid to the polynomial-time many-one complete degrees (in short, $\leq_m^p$-complete degrees) of well known classes, e.g., NP, EXP, NEXP etc. For the class EXP, it has been shown that the $\leq_m^p$-complete degree of EXP coincides with the $\leq_{1,li}^p$-complete degree [5, 22, 10] (the definitions of $\leq_{1,li}^p$ and other restricted reducibilities are given in Section 2). A weaker result is known for the class NEXP—the $\leq_m^p$-complete degree of NEXP coincides with the $\leq_1^p$-complete degree [10]. For the $\leq_m^p$-complete degree of NP, although no absolute result is known, there exist two contradictory conjectures about its structure [6, 16]. While the *isomorphism conjecture*, proposed in [6], states that the $\leq_m^p$-complete degree of NP coincides with a p-isomorphism type, the *encrypted complete set conjecture*, proposed in [16, 18], (effectively) states that the $\leq_{1,li}^p$-complete degree of NP does not coincide with the $\leq_{1,li,i}^p$-complete degree of NP. Moreover, it has also been shown that the answer to either of the above two conjectures cannot be obtained via relativizable techniques [8, 19]. This prompted the investigation of the following more general question: does there exist *some* $\leq_{1,li}^p$-degree that does not coincide with a $\leq_{1,li,i}^p$-degree? Ko, Long, and Du [17], in a surprising result, showed that the answer to the general question is subtle—every $\leq_{1,li}^p$-degree coincides with a $\leq_{1,li,i}^p$-degree if and only if P = UP if and only if every $\leq_{1,li}^p$-degree within the 2-tt-complete degree of EXP coincides with a $\leq_{1,li,i}^p$-degree. Since every non-empty $\leq_{1,li,i}^p$-degree coincides with a p-isomorphism type [6], the above result implies that every non-empty $\leq_{1,li}^p$-degree coincides with a p-isomorphism type if and only if P = UP.

In the same spirit as above, the relationship between $\leq_1^p$- and p-isomorphism types has been characterized by Fenner, Kurtz, and Royer [9]—every $\leq_1^p$-degree coincides with a p-isomorphism type if and only if P = PSPACE if and only if every $\leq_1^p$-degree within the 2-tt-complete degree of EXP coincides with a p-isomorphism type. An excellent survey of results concerning polynomial-time many-one degrees and its functional restrictions can be found in [18].

While the structure of $\leq_m^p$-complete degrees is not well understood at present—as is evident from the (lack of) results above—there has been success in describing the structure of complete degrees under reducibilities that are much weaker than polynomial-time. For example, the structure of $\leq_m^{1-L}$- and $\leq_m^{1-NL}$-complete degrees has been investigated and described in [1, 3, 4, 7, 10, 12, 13, 14], where 1-L and 1-NL reductions are computed essentially by deterministic and nondeterministic logspace TMs respectively with a one-way input head. It was shown in [1] that $\leq_m^{1-L}$-complete degrees of classes closed under log-lin reductions (see next section for the definition of log-lin reductions) coincide with $\leq_{1,li,i}^{1-L}$-complete degrees but not with 1-L-isomorphism types. Also, it was shown that the $\leq_m^{1-NL}$-complete degrees of classes closed under log-lin reductions coincide with 1-NL-isomorphism types.

Sometimes, the complete degrees exhibit another interesting property: the complete degrees under some (many-one) reducibility r coincide with the complete degrees under a (many-one) reducibility *weaker* than r. For example, the (nonuniform) AC^0-complete degrees coincide with the (nonuniform) NC^0-complete degrees for the class NC^1 [2] (another such example is given in [1]). This gives rise to the following interesting question: do complete degrees under polynomial-time and, say, logspace reductions coincide? Or, those under 1-NL and 1-L reductions? In this paper, we study these two questions and obtain some surprising connections of these with the separations of complexity classes.

For 1-NL and 1-L reducibilities, we show that there is a $\leq_m^{1-NL}$-degree (apart from the degrees of Σ^* and $\emptyset$) that coincides with a $\leq_m^{1-L}$-degree if and only if DSPACE(n) = NSPACE(n) if and only if for every class $\mathcal{C}$ closed under log-lin reductions, the $\leq_m^{1-NL}$-complete degree of $\mathcal{C}$ coincides with the $\leq_m^{1-L}$-complete degree (since 1-L reductions are not closed under composition [4], $\leq_m^{1-L}$-complete degrees have to be carefully defined: see next section).

The above result implies that under the assumption DSPACE(n) $\neq$ NSPACE(n), all $\leq_m^{1-NL}$-degrees (except for the degrees of Σ^* and $\emptyset$) are distinct from all $\leq_m^{1-L}$-degrees. Moreover, we get the following corollary for every class $\mathcal{C}$ closed under nondeterministic logspace reductions: there is a set $A_{\mathcal{C}} \in \mathcal{C}$ such that DSPACE(n) $\neq$ NSPACE(n) if and only if $R_m^{1-L}(A_{\mathcal{C}}) \neq \mathcal{C}$ ($R_m^{1-L}(A)$ denotes the class of sets that are reducible to A via 1-L reductions). For any class $\mathcal{C}$ that is closed under nondeterministic logspace reductions and properly contained in DSPACE(n), the above corollary gives us an upward separation result. For example, letting $\mathcal{C}$ = NLOG we get: if $R_m^{1-L}(A_{\mathrm{NLOG}}) \neq$ NLOG then DSPACE(n) $\neq$ NSPACE(n). We get a curious result with $\mathcal{C}$ = EXP: if DSPACE(n) = NSPACE(n) then the class $R_m^{1-L}(A_{\mathrm{EXP}})$ coincides with EXP; on the other hand, if DSPACE(n) $\neq$ NSPACE(n) then the class $R_m^{1-L}(A_{\mathrm{EXP}})$ does not contain *even* DLOG.

As an aside, we are able to show that for all classes closed under 1-NL reductions, the $\leq_m^{1-L}$-complete degree of the class coincides with the $\leq_{1,li,i}^{1-L}$-complete degree. This result is incomparable to a similar result shown for classes closed under log-lin reductions in [1].

For polynomial-time and logspace reducibilities, perhaps not surprisingly, we can only show a much weaker result: if E $\neq$ DSPACE(n) then the $\leq_m^{log}$-complete degree of PSPACE (and deterministic classes above it) is properly contained in the $\leq_m^{p}$-complete degree.

2 Preliminaries

All strings that we consider here are over $\Sigma = \{0, 1\}$. For a string s, $|s|$ denotes its length. The set $\Sigma^{=n}$ denotes the set of all strings of length n. For any string s and for any number i, $1 \leq i \leq |s|$, $s[i]$ denotes the i^{th} bit of s.

Our model of computation is Turing machines with a read-only input tape, a write-only output tape and a read-write work tape.

For a resource bound r on TMs, we denote by $\mathcal{F}(r)$ the class of total functions computed by TMs within the resource bound of r. For the class of functions $\mathcal{F}(r)$, we say that f is an *r-computable function*, or simply, an *r function*, if $f \in \mathcal{F}(r)$; and f is *r-invertible* if there is a function $g \in \mathcal{F}(r)$ such that $g(f(x)) = x$ for every x. We say that set $A \le_m^r$ ($\le_1^r$; $\le_{1,li}^r$; $\le_{1,li,i}^r$) B if there is a many-one (one-one; one-one, length-increasing; one-one, length-increasing and r-invertible) r-computable function f reducing A to B. The $\le_m^r$*-degrees* are the strongly connected components of the relation $\le_m^r$. Similarly, one can define the $\le_1^r$-, $\le_{1,li}^r$- and $\le_{1,li,i}^r$-degrees. For sets Σ^* and $\emptyset$, their $\le_m^r$-degrees contain no other set and thus are trivial. We shall refer to all other $\le_m^r$-degrees as *non-trivial* $\le_m^r$-degrees.

Set A is $\le_m^r$*-hard* for class $\mathcal{C}$ if for every $B \in \mathcal{C}$, $B \le_m^r A$. Set A is $\le_m^r$*-complete* for class $\mathcal{C}$ if A is $\le_m^r$-hard for $\mathcal{C}$ and $A \in \mathcal{C}$. The $\le_m^r$*-complete degree* of $\mathcal{C}$ is defined to be the $\le_m^r$-degree of $\le_m^r$-complete sets for $\mathcal{C}$. The $\le_1^r$-, $\le_{1,li}^r$-, and $\le_{1,li,i}^r$-complete degrees are similarly defined. Set A is r*-isomorphic* to set B, $A \equiv^r B$, if there exists a bijection f between A and B with both f and f^{-1} being r-computable. The r*-isomorphism types* are the smallest equivalence classes induced by the relation $\equiv^r$.

Whenever we talk of a function computed by a nondeterministic TM, we assume that the TM, on any input, outputs the same string on *all* accepting paths.

A *1-L TM* is a deterministic Turing machine with a read-only input tape, a write-only output tape and a logspace-bounded work tape such that its input head is *one-way*, i.e., it moves from left to right only. Further, at the beginning of the computation, $1^{\lceil \log n \rceil}$ is written on the work tape, where n is the length of the input. The class $\mathcal{F}$(1-L) denotes the class of total functions computed by these TMs. These functions were first defined in [12] for studying complete sets for DLOG. The class of $\le_m^{1-L}$-complete sets for NP is a fairly large one: it was shown in [12] that all known *natural* $\le_m^p$-complete sets for NP are $\le_m^{1-L}$-complete as well.

In general, 1-L functions are not closed under composition as 1-L TMs have $1^{\lceil \log n \rceil}$ written on the work tape at the beginning of the computation [4]. Therefore, it is possible that there are two sets in a $\le_m^{1-L}$-degree that do not reduce to each other via a 1-L reduction. Nevertheless, it can be shown that the $\le_m^{1-L}$-complete degree of any class closed under logspace reductions consists entirely of $\le_m^{1-L}$-complete sets and therefore, every pair of sets in such degrees reduce to each other via 1-L reductions (this result follows directly from results in [1, Theorem 4.2 & Corollary 4.4], however, it has not been proved explicitly since the $\le_m^{1-L}$-complete degrees were defined there to be just the class of all $\le_m^{1-L}$-complete sets).

A *1-NL TM* is a nondeterministic Turing machine with the rest of the conditions being same as for a 1-L TM. Class $\mathcal{F}$(1-NL) contains total functions that are computed by 1-NL TMs (as these TMs are nondeterministic, they output the same string on all accepting paths). These functions are closed under composition—to compute $f \circ g$ on input x (both f and g are 1-NL functions computed by TMs M_f and M_g respectively), a 1-NL TM first guesses the length of $g(x)$, then simulates M_g on x and M_f on its output with the guessed length and accepts iff the guessed length turns out to be correct. It is obvious that every $\le_m^{1-L}$-degree is contained in a $\le_m^{1-NL}$-degree.

Finally, function f is a *log-lin* function [20] if it can be computed by logspace bounded DTMs and for all x: $|f(x)| = O(|x|)$.

3 The characterization

The structure of $\le_m^{1-L}$- and $\le_m^{1-NL}$-complete degrees for various classes has been investigated intensively (see [1, 3, 4, 7, 10, 14]). As a culmination of this research, the structure of these degrees (for classes closed under log-lin reductions) was completely described in [1]. It was shown there that for every class closed under log-lin reductions, the $\le_m^{1-L}$-complete degree coincides with the $\le_{1,li,i}^{1-L}$-complete degree, and sets in this degree are 2-L-isomorphic (2-L reductions are computed by logspace DTMs with a one-way input head that are allowed to make two left-to-right scans of the input) but not 1-L-isomorphic. For 1-NL reducibility, it was shown that for any class closed under log-lin reductions, the $\leq_m^{1\text{-}NL}$-complete degree coincides with the $\leq_{1,li,i}^{1\text{-}NL}$-complete degree, which, in turn, is a 1-NL-isomorphism type. In the proof of our results below, we shall be using several results and ideas of [1]. We shall also need the following definitions about 1-NL TMs.

For the purpose of definitions below, let M be a 1-NL TM computing a total function f. Without loss of generality, we can assume that M has a set of special states called the *transit* states such that M enters one of these states when and only when it moves the input head. Also, we can assume that once M reads all the input bits, it behaves deterministically.

Definition 3.1 A *configuration of M of size n* is a partial ID of M on the input strings of size n. It is written as a 5-tuple $\langle st, in, out, wk, tape \rangle$ where st denotes the state of M; in, out and wk denote respectively the input head, output head and work tape head positions; and $tape$ denotes the contents of the work tape. We refer to the starting configuration of M of size n as C_{init}^n and an accepting configuration as C_{accept}^n (there may be more than one accepting configuration).

A *transit* configuration of M of size n is a configuration of M of size n whose state is a transit state.

A 1-NL TM, on any input of size n, and on any of its nondeterministic paths, passes through exactly n transit configurations.

Forgetful TMs were defined in [1]. There, a slightly different, but essentially equivalent notion of transit configurations was used.

Definition 3.2 A 1-NL TM M is *forgetful* if, for every n, there is a sequence $\mathcal{S}$ of transit configurations of M of size n such that, for every input x of size n there is an accepting path of M on input x such that $\mathcal{S}$ is the sequence of transit configurations that M goes through on this path. A function computed by a forgetful 1-NL TM is called a *forgetful* 1-NL function.

In [1, Theorem 5.3], the following is proved for forgetful 1-NL TMs.

Lemma 3.3 *For any class $\mathcal{C}$ closed under log-lin reductions, any $\leq_m^{1\text{-}NL}$-hard set for $\mathcal{C}$ is also hard under forgetful 1-NL reductions.*

We do not prove the lemma here: its proof is similar to the proof of Claim 4.3.3 in the next section. The above lemma will be the key to our proof of the theorem below, which gives the first implication of our characterization.

Theorem 3.4 *If* DSPACE(n) = NSPACE(n), *then for every class $\mathcal{C}$ closed under log-lin reductions, the $\leq_m^{1\text{-}NL}$-complete degree of $\mathcal{C}$ coincides with the $\leq_m^{1\text{-}L}$-complete degree.*

Proof. Let A be a $\leq_m^{1\text{-}NL}$-complete set for $\mathcal{C}$ and $B \in \mathcal{C}$. By Lemma 3.3, there is a forgetful 1-NL reduction f of B to A, computed by, say, TM M. We show that, assuming DSPACE(n) = NSPACE(n), f can be computed by a 1-L TM too.

Denote by $p(D)$ the input head position in the configuration D of M. We define three sets:

$O_1 = \{(m, i, D_1, D_2) \mid D_1$ and D_2 are configurations of M of size m with D_2 being a transit configuration, and $p(D_1) = i$, and $p(D_2) = i + 1$, and for any bit written on the i^{th} cell of the input tape, there is a guess path on which the TM M moves from D_1 to $D_2\}$,

$O_2 = \{(m, i, D) \mid D$ is a configuration of M of size m, and $p(D) = i$, and there exist transit configurations $D_{i+1}, D_{i+2}, \ldots, D_{m+1}$ of M such that for every j, $i \leq j \leq m$: $(m, D_j, D_{j+1}) \in O_1$ where $D_i = D$, and M moves from D_{m+1} to an accepting configuration$\}$,

$$O_3 = \{(m, i, D_1, D_2, b, D') \mid D_1, D_2, \text{ and } D' \text{ are configurations of } M \text{ of size } m \text{ with } D_2 \text{ being a transit configuration, and } b \in \{0,1\}, \text{ and } p(D_1) = i, \text{ and } p(D_2) = i+1, \text{ and there is a guess on which the TM } M \text{ moves from } D_1 \text{ to } D' \text{ in a single step and there is a guess path on which the TM } M \text{ moves from } D' \text{ to } D_2 \text{ when the bit } b \text{ is written at the } i^{th} \text{ input cell of the input tape}\}.$$

All the above three sets are easily seen to be in NSPACE(n). Thus, by our assumption, they belong to DSPACE(n). Using these sets, we can define a 1-L TM which computes f.

> Input x, $|x| = n$. Let $D_1 = C^n_{init}$, $i = p(D_1) = 1$, and repeat the following until $i = n+1$.
>
> > Find D_2 such that $(n, i, D_1, D_2) \in O_1$ and $(n, i+1, D_2) \in O_2$. Repeat the following until $D_1 = D_2$.
> >
> > > Let D' be a configuration such that $(n, i, D_1, D_2, x[i], D') \in O_3$. Find out the output o, if any, of M while it moves from D_1 to D' (as D' is just one step away from D_1 it can be easily done). Output o, let $D_1 = D'$, and continue.
> >
> > Let $i = i + 1$.
>
> Finally, output the string output by M while moving from D_1 to an accepting configuration (as M behaves deterministically after reading the input, this is easily simulated).

The above procedure is logspace computable as the input queries to the sets O_1, O_2, and O_3 are of size $O(\log n)$. It is also easy to see that the above procedure can be carried out by a 1-L TM. We now show that the procedure computes $f(x)$.

Lemma 3.3 guarantees that $(n, 1, C^n_{init}) \in O_2$. And by the accepting criteria of O_2, it is clear that a configuration D_2, as required inside the first loop of the above procedure, always exists. Inside the second loop, the procedure simulates M between configuration D_1 and D_2 along one guess path. Since M is constrained to output the same string on every accepting path, it must output the same string along every path from D_1 to D_2 (recall that a configuration also stores the position of the output head). The accepting criteria of O_2 also guarantees that, after scanning all the input bits, the configuration of M, D_1, must lead to an accepting configuration. Thus, the procedure computes $f(x)$ correctly.

Therefore, we have $B \leq_m^{1\text{-}L} A$ via h. This completes the proof. ■

We now show the other non-trivial direction of the characterization.

Theorem 3.5 *If there is a non-trivial $\leq_m^{1\text{-}NL}$-degree that coincides with a $\leq_m^{1\text{-}L}$-degree then* DSPACE(n) = NSPACE(n).

Proof. Let B be a set in the $\leq_m^{1\text{-}NL}$-degree that coincides with a $\leq_m^{1\text{-}L}$-degree. Define $\hat{B}$ to be the set $\{0x \mid x \in B\} \cup 1^*$. Clearly, $\hat{B}$ is in the same $\leq_m^{1\text{-}NL}$-degree as B.

Let L be the following 'universal' set for NSPACE(n):

$$L = \{i \mid \text{ NTM } M_i \text{ accepts } i \text{ within } |i| \text{ space}\}.$$

Here we assume that each TM M_i works over binary alphabet Σ. It is easy to see that $L \in$ DSPACE(n) iff DSPACE(n) = NSPACE(n). We show that $L \in$ DSPACE(n).

Let χ^m_L denote the characteristic vector of L for the first m bits (in the lexicographic ordering) of Σ^*, i.e., $\chi^m_L[i] = 1$ iff the i^{th} string of Σ^* is in L. Define the set $\hat{B}_L$ as:

$x \in \hat{B}_L$ iff (1) $x = yz$ with $|y| = |z|$, (2) $y \in \hat{B}$, and (3) $z = \chi_L^{|z|}$.

Claim 3.5.1 *Set $\hat{B}_L$ is in the same $\leq_m^{1\text{–}NL}$-degree as $\hat{B}$.*

Proof of Claim 3.5.1. To reduce $\hat{B}$ to $\hat{B}_L$, a 1-NL TM first outputs the input string x and then computes and outputs $\chi_L^{|x|}$ (it can be done within nondeterministic logspace by the TM as every such string has length at most $\log |x|$ and NSPACE(n) is closed under complement [15, 21]).

To reduce $\hat{B}_L$ to $\hat{B}$, a 1-NL TM works in the following way: first it guesses n to be the length of the input (note that the string written on the worktape gives it an upper bound on the input length). If n is odd then it outputs a fixed string not in $\hat{B}$. It then checks if the input length equals n. If yes, it accepts, else aborts. If n is even then it branches into two paths.

On the first path, it outputs the first $n/2$ bits of the input, then checks if the next $n/2$ bits equal $\chi_L^{n/2}$. If yes, and the input length equals n, it accepts. In all other cases, it aborts.

On the second path, it outputs a fixed string not in $\hat{B}$, skips the first $n/2$ bits of the input, then checks if the next $n/2$ bits equal $\chi_L^{n/2}$. If no, and the input length equals n, it accepts. In all other cases, it aborts. □

Using ideas from [1], we can prove the following lemma (see next section).

Lemma 4.3 *If a non-trivial $\leq_m^{1\text{–}NL}$-degree coincides with a $\leq_m^{1\text{–}L}$-degree then any two sets in the degree reduce to each other via size-increasing 1-L reductions.*

By our assumption, the $\leq_m^{1\text{–}NL}$-degree of $\hat{B}$ coincides with the $\leq_m^{1\text{–}L}$-degree of $\hat{B}$. Therefore, by the above lemma, there exists a size-increasing 1-L reduction, say f, of $\hat{B}$ to $\hat{B}_L$. Now, the following procedure recognizes L.

> On input x, let x be the n^{th} string in the lexicographic ordering of strings of Σ^*. Compute the length of the string $f(1^{2n})$. Let it be m. Now, compute the $(m/2+n)^{th}$ bit of $f(1^{2n})$ and accept iff this bit is 1.

The above procedure is deterministic and works within linear space. To see that it recognizes L, we note that $1^{2n} \in \hat{B}$ and f is a size-increasing reduction of $\hat{B}$ to $\hat{B}_L$. Therefore, $f(1^{2n})$ must belong to $\hat{B}_L$, and its length, m, must be at least $2n$. Since $f(1^{2n}) \in \hat{B}_L$, m is even and the last $m/2$ bits of $f(1^{2n})$ represent the characteristic vector of L for the first $m/2 \geq n$ strings of Σ^*. Thus, the $(m/2+n)^{th}$ bit of $f(1^{2n})$ is 1 iff $x \in L$.

Therefore, $L \in$ DSPACE(n), which implies NSPACE(n) = DSPACE(n). ■

Combining the results of the above two theorems, we get:

Theorem 3.6 *The following three statements are equivalent.*

1. DSPACE(n) = NSPACE(n).
2. *There is a non-trivial $\leq_m^{1\text{–}NL}$-degree that coincides with a $\leq_m^{1\text{–}L}$-degree.*
3. *For every class $\mathcal{C}$ closed under log-lin reductions, the $\leq_m^{1\text{–}NL}$-complete degree of $\mathcal{C}$ coincides with the $\leq_m^{1\text{–}L}$-complete degree of $\mathcal{C}$.*

Proof. ($2 \Rightarrow 1$): from Theorem 3.5.
($1 \Rightarrow 3$): from Theorem 3.4.
($3 \Rightarrow 2$): Follows from the fact that there are classes having non-empty $\leq_m^{1\text{–}L}$-complete degrees, e.g., NLOG [12]. ■

Let $R_m^{1\text{–}L}(D)$ denote the class of sets that are reducible to D via 1-L reductions. We get the following corollary.

Corollary 3.7 *For any class $\mathcal{C}$ closed under both 1-NL and log-lin reductions that has a $\leq_m^{1\text{–}NL}$-complete set the following holds. There exists a set $A_{\mathcal{C}} \in \mathcal{C}$ such that:*

$$\mathrm{DSPACE}(n) \neq \mathrm{NSPACE}(n) \textit{ iff } R_m^{1\text{–}L}(A_{\mathcal{C}}) \neq \mathcal{C}.$$

Proof. Let B be a $\leq_m^{1\text{–}NL}$-complete set for $\mathcal{C}$. Take $A_{\mathcal{C}}$ to be the set $\hat{B}_L$ defined in the proof of Theorem 3.5. Then, since $\mathcal{C}$ is closed under 1-NL reductions and $\hat{B}_L \leq_m^{1\text{–}NL} B$, $A_{\mathcal{C}}$ is $\leq_m^{1\text{–}NL}$-complete for $\mathcal{C}$. If $A_{\mathcal{C}}$ is also $\leq_m^{1\text{–}L}$-complete for $\mathcal{C}$ then, by [1, Theorem 4.2] and the fact that $\mathcal{C}$ is closed under log-lin reductions, $A_{\mathcal{C}}$ is $\leq_{li}^{1\text{–}L}$-complete as well. Now, as proved in Theorem 3.5, DSPACE(n) = NSPACE(n). On the other hand, if DSPACE(n) = NSPACE(n) then indeed $A_{\mathcal{C}}$ is $\leq_m^{1\text{–}L}$-complete for $\mathcal{C}$ (by Theorem 3.4). ■

When $\mathcal{C}$ is any class closed under both 1-NL and log-lin reductions and is properly contained in DSPACE(n), e.g., NLOG, NSPACE($\log^2 n$) etc., the above corollary gives us an upward separation result.

A curious result follows by setting $\mathcal{C}$ = EXP.

Corollary 3.8 *There is a set* $A_{\mathrm{EXP}} \in \mathrm{EXP}$ *such that if* DSPACE(n) = NSPACE(n) *then* $R_m^{1\text{–}L}(A_{\mathrm{EXP}}) = \mathrm{EXP}$*, and if* DSPACE($n$) $\neq$ NSPACE(n) *then* DLOG $-$ $R_m^{1\text{–}L}(A_{\mathrm{EXP}})$ *is non-empty.*

Proof. Again, we let A_{EXP} to be the set $\hat{B}_L$ defined in the proof of Theorem 3.5 for a $\leq_m^{1\text{–}NL}$-complete set B of EXP. The case when DSPACE(n) = NSPACE(n) follows from the Corollary 3.7 above. If DLOG $\subseteq R_m^{1\text{–}L}(A_{\mathrm{EXP}})$ then A_{EXP} is $\leq_m^{1\text{–}L}$-hard for DLOG. So, by [1, Theorem 4.2], A_{EXP} is also $\leq_{li}^{1\text{–}L}$-hard for DLOG. Therefore, there is a size-increasing 1-L reduction of 1^* to A_{EXP}. Now, observe that the only fact required in the proof of Theorem 3.5 is that $1^* \leq_{li}^{1\text{–}L} \hat{B}_L$. So, the same proof yields here that $L \in$ DSPACE(n) and therefore DSPACE(n) = NSPACE(n). ■

We can easily obtain the following result for complete degrees under logspace and polynomial-time reductions, using the proof idea of Theorem 3.5.

Theorem 3.9 *If* E $\neq$ DSPACE(n) *then the $\leq_m^p$-complete degree of* PSPACE *properly contains the $\leq_m^{log}$-complete degree of* PSPACE.

Proof. Suppose that the $\leq_m^p$-complete degree of PSPACE coincides with the $\leq_m^{log}$-complete degree. Then, as in the proof of Theorem 3.5, we take a set A in the $\leq_m^p$-complete degree of PSPACE and a $\leq_m^{log}$-complete set L of E, and define the set A_L. Set A_L is $\leq_m^p$-complete for PSPACE and by our assumption, is also $\leq_m^{log}$-complete. For PSPACE, it is known that the $\leq_m^{log}$-complete degree coincides with the $\leq_{1,li}^{log}$-complete degree [11]. Therefore, set A_L is $\leq_{1,li}^{log}$-complete for PSPACE. Let f be a size-increasing logspace reduction of 1^* to A_L. Again, as in the proof of Theorem 3.5, we get that $L \in$ DSPACE(n). ■

Results analogous to the above theorem can be shown for any class whose $\leq_m^{log}$-complete degree coincides with the $\leq_{1,li}^{log}$-complete degree, e.g., the deterministic classes above PSPACE.

4 $\leq_m^{1\text{–}L}$-Complete sets

In this section, we first give a proof of Lemma 4.3 and then prove a result on the structure of $\leq_m^{1\text{–}L}$-complete degrees of classes closed under 1-NL reductions.

As for 1-NL functions, one can define the notion of *forgetful* 1-L TMs and functions.

Definition 4.1 A 1-L TM M is *forgetful* if, for every n, the sequence of transit configurations of M on every input of size n is identical. A *forgetful* 1-L function is one computed by a forgetful 1-L TM.

Function g is said to be a *length-restricted* 1-L function if it is a 1-L function satisfying the property that for every x, $|g(x)| = |g(1^{2^{\lceil \log |x| \rceil}})|$. Such 1-L functions satisfy the following property which we shall make use of.

Proposition 4.2 *Let f be a 1-L function and g a length-restricted 1-L function. Then, $f \circ g$ is also a 1-L function. Further, if both f and g are forgetful, then $f \circ g$ is also forgetful.*

Proof. Let M_1 and M_2 be 1-L TMs computing f and g respectively. A 1-L TM M can compute $f \circ g$ by simulating M_2 on input x and M_1 on its output in parallel. However, to start the computation of M_1, it needs to have $1^{\lceil \log |g(x)| \rceil}$ written on the worktape. Since g is length-restricted, this string can be computed without scanning the input. The TM M just runs M_2 on the input $1^{2^{\lceil \log |x| \rceil}}$ and calculates the length of the output ($= |g(1^{2^{\lceil \log |x| \rceil}})|$). Using this, $\lceil \log |g(x)| \rceil$ can be easily computed.

If both M_1 and M_2 are forgetful, then M would also be forgetful as M needs to store only the configurations of these two TMs on the worktape (besides some input independent information). ∎

Lemma 4.3 *If a non-trivial $\leq_m^{1\text{–}NL}$-degree coincides with a $\leq_m^{1\text{–}L}$-degree then any two sets in the degree reduce to each other via size-increasing 1-L reductions.*

Proof. It follows from [1, Theorem 4.2 & Corollary 4.4] that, for every class closed under log-lin reductions, the $\leq_m^{1\text{–}L}$-complete degree of the class coincides with the $\leq_{1,li,i}^{1\text{–}L}$-complete degree. A somewhat easier proof can be given to show that the $\leq_m^{1\text{–}L}$-complete degree coincides with the $\leq_{li}^{1\text{–}L}$-complete degree. We adopt this proof for our purposes. However, we need to modify the proof in several places to make it work here.

The outline of the proof in [1] is as follows. Given a $\leq_m^{1\text{–}L}$-complete set A of class $\mathcal{C}$ that is closed under log-lin reductions, first it is shown that the set is also complete under forgetful 1-L reductions. And then, given any set $B \in \mathcal{C}$, a coded version of the set B, D, is constructed such that the forgetful 1-L reduction of D to A composed with a straightforward reduction of B to D is a size-increasing 1-L reduction of B to A.

While adopting the above proof, we have to take care of the following two points. Firstly, we have only a $\leq_m^{1\text{–}NL}$-degree instead of a class closed under log-lin reductions. So the proof has to be modified to work for $\leq_m^{1\text{–}NL}$-degrees. Secondly, although the $\leq_m^{1\text{–}NL}$-degree under consideration coincides with a $\leq_m^{1\text{–}L}$-degree (under the hypothesis), it is not clear if every pair of sets in the degree are reducible to each other via 1-L reductions. Instead, for two sets A_1 and A_2 in the degree, we can only say that there is a finite sequence of sets B_1, B_2, …, B_t such that $A_1 \leq_m^{1\text{–}L} B_1 \leq_m^{1\text{–}L} B_2 \leq_m^{1\text{–}L} \cdots \leq_m^{1\text{–}L} B_t \leq_m^{1\text{–}L} A_2$ (since a $\leq_m^{1\text{–}L}$-degree is defined as the strongly connected components of $\leq_m^{1\text{–}L}$ relation). Thus, to show that $A_1 \leq_{li}^{1\text{–}L} A_2$, we must do some more work.

Let $\mathbf{d}$ be the $\leq_m^{1\text{–}NL}$-degree that coincides with a $\leq_m^{1\text{–}L}$-degree. We now describe our construction which is in two stages. In the first stage, we show that every pair of sets in $\mathbf{d}$ are reducible to each other via forgetful 1-L reductions (under the assumption that the degree coincides with a $\leq_m^{1\text{–}L}$-degree), and in the second stage we show that these reductions can also be made length-increasing.

Stage 1. Let $B \in \mathbf{d}$. We first define a somewhat complicated looking partial function based on B which will play a crucial role in obtaining a forgetful 1-L reduction from B to any set in the degree. The function is given by the following procedure.

Function $l(y)$.

If $|y|$ is not an exact power of two then reject.

If $y = 0y'$ then output y.

If $y = 1y'$ then

If $|y| \neq n^2$ for any n then reject.

Else, let $y' = 1^b 0 y'' 0 1^r$ with $|y'| = n^2 - 1$.

If $|y''| \neq 2bn$ then reject.

Else, let $y'' = u_1 u_2 \cdots u_n$ with $|u_i| = 2b$ for each i.

If for some i, u_i is not of the form $v_i 0^{j_i - 1} 1 0^{b - j_i}$ for any j_i then reject.

Else, let string x, $|x| = n$, be such that $x[i] = 1$ iff $u_i = v_i 0^{j_i - 1} 1 0^{b - j_i}$ and the ${j_i}^{th}$ bit of v_i, $v_i[j_i]$, is 1. Output x.

Recall that a partial function p is computed by a *strong* NTM if the TM, on input x, outputs $p(x)$ on all the accepting paths whenever $p(x)$ is defined, otherwise the TM rejects. Further, the TM never both accepts and rejects on two different paths on the same input (it may abort on some of the paths though).

Claim 4.3.1 *Function l can be computed by a strong 1-NL TM.*

Proof of Claim 4.3.1. The following 1-NL TM computes l. On input y, it begins by checking if $y = 0y'$. If yes, it outputs y and then verifies if $|y|$ is an exact power of two. If yes, then accepts else rejects. Otherwise, if $y = 1y'$, it scans the initial part of the input ($1^b 0$) to compute the value of b.

Now the TM scans the remaining string $y'' 0 1^r$. To detect that it has read entire y'' it employs the following 'delayed processing' strategy: after reading any zero, the TM scans the input for the next zero and keeps a count of the number of ones read. On finding a zero it concludes that it is within y'', and if there is no zero then the number of ones give the value of r.

The TM, while scanning y'', does the following. It processes the input in blocks of $2b$ bits and also keeps a count of the number of blocks read so far. For every such block, before reading it, the TM branches into $b+1$ paths. On the j^{th} path, $1 \leq j \leq b$, it assumes the last b bits of the block to be of the form $0^{j-1} 1 0^{b-j}$, while on the $(b+1)^{th}$ path it assumes the last b bits to be not of the above form. On the j^{th} path, $1 \leq j \leq b$, the TM outputs the j^{th} bit of the block while disregarding the rest of the first b bits, and then goes on to verify if the last b bits are indeed of the form $0^{j-1} 1 0^{b-j}$. If yes, the TM continues otherwise aborts. On the $(b+1)^{th}$ path, the TM disregards the first b bits and then verifies if the last b bits do not have exactly one one. If yes, the TM halts in a rejecting state, otherwise aborts.

Once the TM has scanned y'' completely, it computes the number of blocks in y'', say n, and the length of the input, say m, and checks that m is an exact power to two, and $m = n^2$. If any of these conditions is not satisfied, the TM rejects, otherwise accepts.

It is easy to see that the above TM computes l correctly and is a strong 1-NL TM. □

Define the function p by $p(x 0 1^r) = x$. The function p is a 1-L function: a 1-L TM computing p employs the 'delayed processing' strategy as above. Specifically, the TM, on reading a zero in the input, counts the number of successive ones, and outputs a zero followed by the number of ones counted only if it reads another zero.

We now define a the following set using B, l, and p:

$$C = \{y \mid (\exists i \geq 1)(p \circ l)^i(y) = 0x \ \wedge \ x \in B\}$$

(here $(p \circ l)^i(y)$ denotes $(\underbrace{p(l(p(l(\cdots p(l}_{i \ times}(y))\cdots)))))$.

Claim 4.3.2 *The set C is in degree* **d**.

Proof of Claim 4.3.2. $B \leq_m^{1-L} C$ via mapping $h_0(x) = 0x01^{2^{\lceil \log |x| \rceil + 2} - |x| - 2}$.

$C \leq_m^{1-NL} B$ via the mapping h, where $h(y) = x$ if there is an $i \geq 1$ such that $(p \circ l)^i(y) = 0x$; $h(y) = z_2$ (for some fixed $z_2 \notin B$) otherwise. To see that h is a 1-NL function we first note that $p \circ l$ can be computed by a strong 1-NL TM; for any z, $|p(z)| < |z|$; and if $l(z)$ is defined then either $l(z) = z$ (this happens exactly when z begins with a 0) or $|l(z)| \leq |z|^{1/2}$. So, to compute $h(y)$, a 1-NL TM branches out in two paths. In the first path, it starts the computation of $p \circ l$ on y, and the computation of $p \circ l$ on $(p \circ l)(y)$, and so on. It also keeps checking if the first bit of the output of $(p \circ l)^i(y)$ for any i, is 0. If yes, then it outputs $(p \circ l)^i(y)$ except for the first bit. It can simulate all the copies within the available space as the space requirements are halved for every next copy, essentially as described in [11, 1]. Eventually, all the computations halt, and if any of them aborts or rejects, the TM aborts; and if all of them accept, the TM accepts. On the second path also the TM does the same simulation but without outputting any string. If any of the computations aborts or all the computations accept, the TM aborts; otherwise it outputs z_2 and accepts. □

The set C has the following interesting property:

Claim 4.3.3 *Let $C \leq_m^{1-L} A$ for any set A. Then, $C \leq_m^{1-L} A$ via a forgetful, length-restricted 1-L reduction.*

Proof of Claim 4.3.3. Let the TM M compute a 1-L reduction f of C to A. Let $q_M(n)$ be the polynomial bounding the number of configurations of M of size n. Define a reduction g of C to itself as given by the following stage-wise procedure:

> On input x, let n_0 be the smallest number such that $n_0^2 \geq (1 + 2 \cdot n_0) \cdot (\lceil \log q_M(n_0^2) \rceil + 1) + 3$. Let $n = \max\{2^{\lceil \log n_0 \rceil}, 2^{\lceil \log |x| \rceil}\}$, and $b = \lceil \log q_M(n^2) \rceil + 1$.
>
> **Stage** 0 : Let C_0 be the configuration of size n^2 such that M moves from $C_{init}^{n^2}$ to C_0 on reading the string 11^b0 written on the first $b + 2$ bits positions of the input string. Output 11^b0.
>
> **Stage** i, $1 \leq i \leq n$: Find the smallest configuration C of size n^2 and the smallest two strings v and w of length b with $v > w$ and the j_i^{th} bit being the first one where v and w differ, such that M moves from C_{i-1} to C when either of the strings $v0^{j_i-1}10^{b-j_i}$ and $w0^{j_i-1}10^{b-j_i}$ is written on the bit positions $b + 2 + (i - 1) \cdot 2b + 1$ thru $b + 2 + i \cdot 2b$ (the i^{th} block) of the input string. Let $C_i = C$. If $i \leq |x|$, output $v0^{j_i-1}10^{b-j_i}$ if $x[i] = 1$, $w0^{j_i-1}10^{b-j_i}$ otherwise. If $i = |x| + 1$, output $w0^{j_i-1}10^{b-j_i}$. And if $i > |x| + 1$, outout $v0^{j_i-1}10^{b-j_i}$. Goto next stage.
>
> **Stage** $n + 1$: Output 01^r where $r = n^2 - (b + 2bn + 3)$. By the choice of n we have that $r \geq 0$.

The above procedure is clearly computable within logspace as $b = \lceil \log q_M(n^2) \rceil + 1 = O(\log n)$. We now show that the two strings u and v as required in Stage i of the procedure, exist for every i, $1 \le i \le n$. Since $|v| = |w| = b$, there are a total of 2^b such strings. Since there are at most $q_M(n^2)$ configurations of M on inputs of size n^2, there are at least $2^b/q_M(n^2) \ge 2$ such strings, say v and w, $v > w$, and a configuration D such that M moves from C_{i-1} to D when either of v and w is written on the first half of the i^{th} block of input. Now, M will enter the $(i+1)^{th}$ block in the same configuration on both vs and ws for any string s of size b. It is easy to verify that $p(l(g(x))) = x$. Thus, function g is a 1-L reduction of C to itself.

We now show that the function $f \circ g$ is a forgetful, length-restricted 1-L reduction of C to A. The TM that computes $f \circ g$ works as follows. On input x, it first computes the output of the Stage 0 of the above procedure computing g, and then simulates M on this output. Next, for each i, $1 \le i \le n$, it computes the output of the Stage i of the procedure (it would need to read the bit $x[i]$ if $i \le |x|$), and simulates the TM M on the output. After completing the simulation, it erases the output from the worktape and only keeps the configuration of M stored. Finally, it computes the output of Stage $n+1$ and simulates M on it. Therefore, any time the TM moves the input head, it has only the configuration of M written on the worktape, and by the construction of g, this configuration is independent of the bits read so far. Also, the TM needs to scan the input only once. Thus, $f \circ g$ is a forgetful 1-L function. To see that it is also length-restricted, it is sufficient to observe that $|g(x)| = |g(1^{2^{\lceil \log |x| \rceil}})|$, and the TM M, on inputs of size n from the range of g, halts in the same configuration. Therefore, $|f(g(x))| = |f(g(1^{2^{\lceil \log |x| \rceil}}))|$.

□

Using the set C, we can construct a forgetful and length-restricted 1-L reduction of B to any set in the degree $\mathbf{d}$. Let $A \in \mathbf{d}$. Since $C \in \mathbf{d}$, and $\mathbf{d}$ coincides with a $\le_m^{1\text{-}L}$-degree, we have that there exist sets $B_1, B_2, \ldots, B_t$ such that $C \le_m^{1\text{-}L} B_1 \le_m^{1\text{-}L} B_2 \le_m^{1\text{-}L} \cdots \le_m^{1\text{-}L} B_t \le_m^{1\text{-}L} A$. By Claim 4.3.3, $C \le_m^{1\text{-}L} B_1$ via a forgetful, length-restricted 1-L function. Therefore, by Proposition 4.2, $C \le_m^{1\text{-}L} B_2$. Iterating this process $t+1$ times, we get $C \le_m^{1\text{-}L} A$ via a forgetful, length-restricted 1-L function. The reduction of B to C in the proof of Claim 4.3.2 is also a forgetful and length-restricted 1-L function. Therefore, by Proposition 4.2, $B \le_m^{1\text{-}L} A$ via forgetful 1-L function. This completes the Stage 1.

Stage 2. Let $B \in \mathbf{d}$. In this stage we show that B reduces to every set in $\mathbf{d}$ via a length-increasing 1-L function. As in the previous stage, we construct an intermediate set which will be used to obtain a length-increasing reduction of B. Define the set D as given by the following procedure:

> On input y, let $y = v01^r$. Reject if $|v|$ is not even. Otherwise, let $v = xw$ with $|x| = |w|$. Accept iff either $w = 1^{|w|}$ and $x \in B$, or for some j, $w = 1^{j-1}01^{|w|-j}$ and $x[j] = 1$.

Set D also belongs to the degree $\mathbf{d}$: $g(x) = x1^{|x|}01^{2^{\lceil \log |x| \rceil + 2} - 2|x| - 1}$ is a length-restricted 1-L reduction of B to D, and a 1-NL reduction of D to B is computed by a TM as follows. Let $z_1 \in B$ and $z_2 \notin B$ be two fixed strings. On input $y = v01^r$, guess the length of the string v to be n (the upper bound on n is given by $2^{\lceil \log |y| \rceil}$ which can be computed without scanning the input). If n is odd, then output z_2 and scan the input to compute $|v|$ (the TM computes $|v|$ by using the usual delayed processing strategy described above). If $|v| = n$ then accept otherwise abort. If n is even, then branch into $n/2 + 2$ paths. On the first path, output the first $n/2$ bits of the input, and then check if the next $n/2$ bits are all ones. If yes, and $|v| = n$ then accept, otherwise abort. On the second path, output z_2, ignore first $n/2$ bits, and check if the next $n/2$ bits have at least two zeroes. If yes, and $|v| = n$ then accept, otherwise abort. On the $(j+2)^{th}$ path, $1 \le j \le n/2$, output z_1 if $y[j] = 1$, output z_2 otherwise. Check if the second block of $n/2$ bits is of the form $1^{j-1}01^{n/2-j}$. If yes, and $|v| = n$ then accept, otherwise abort.

Let $A \in \mathbf{d}$ be any other set. From Stage 1, we know that there is a forgetful 1-L reduction, say f computed by TM M, of D to A. By Proposition 4.2, function $h = f \circ g$ is a forgetful 1-L reduction of B to A since g is both forgetful and length-restricted. Consider two different strings x and y with $|x| = |y| = n$. Since h is forgetful, $|h(x)| = |h(y)|$. Suppose $h(x) = h(y)$. By definition, $g(x) = x1^n01^r$, $g(y) = y1^n01^r$ for some r. Clearly, $|g(x)| = |g(y)|$, and $g(x) \neq g(y)$. Let j^{th} bit be the first one where x and y differ. Let $x' = x1^{j-1}01^{n-j}01^r$ and $y' = y1^{j-1}01^{n-j}01^r$. By the definition of D, exactly one of x' and y' belongs to D. Consider $f(x')$ and $f(y')$. The TM M, after reading the first n bits of either x' or y' would end up in the same configuration since it is forgetful. Since the first n bits of x' are y' are identical to those of $g(x)$ and $g(y)$ respectively, and $f(g(x)) = f(g(y))$, M would output the same string while reading the first n bits of either of x' or y'. And since the remaining bits of x' and y' are the same, the output of M on either of x' or y' is identical. Therefore, $f(x') = f(y')$. However, this is a contradiction since f is a reduction of D to A. Therefore, $h(x) \neq h(y)$. Since $|h(x)| = |h(y)|$, we get that for every x, $|h(x)| \geq |x|$. Now, a simple padding (see the proof of Theorem 4.4) yields a size-increasing reduction of B to A. ■

It was shown in [1] that for every class closed under log-lin reductions, the $\leq_m^{1-L}$-complete degree of the class coincides with the $\leq_{1,li,i}^{1-L}$-complete degree. We now show that this result holds for all classes closed under 1-NL reductions too. Our result is not comparable to the one in [1] as there are classes closed under 1-NL reductions but not under log-lin reductions (e.g., the class of sets recognized by 1-NL TMs), and if DSPACE(n) $\neq$ NSPACE(n) then there are classes closed under log-lin reductions but not under 1-NL reductions (e.g., DLOG).

Theorem 4.4 *For any class $\mathcal{C}$ closed under 1-NL reductions, any set $\leq_m^{1-L}$-hard for $\mathcal{C}$ is also $\leq_{1,li,i}^{1-L}$-hard for $\mathcal{C}$.*

Proof. Let A be a $\leq_m^{1-L}$-hard set for $\mathcal{C}$ and B be any set in $\mathcal{C}$. We need to show that $B \leq_{1,li,i}^{1-L} A$. Define set E as:

$$E = \{x01^l \mid l \geq 0 \ \wedge \ x \in B\}.$$

Proceed exactly as in the proof of Lemma 4.3 to obtain a forgetful 1-L reduction h of E to A (as in Stage 2). The only point to note is that the sets C and D as constructed there, are reducible to E via 1-NL reductions, and so it suffices for the class $\mathcal{C}$ to be closed under 1-NL reductions.

Function h, as shown in Stage 2, is size-nondecreasing. In fact, it is also shown to be one-one on Σ^n for every $n > 0$. To get a reduction of B to A that is one-one everywhere, we just need to do some padding. Let $|h(x)| \leq 2^{c \cdot \lceil \log |x| \rceil + c}$ for some constant c and for all x. Define $r(m) = c \cdot r(m-1) + c + 1$, $r(0) = 1$. Let $g(x) = x01^l$ where $l = 2^{r(m)} - |x| - 1$, and m is the smallest number such that $2^{r(m)} \geq |x| + 1$. Function g is clearly a forgetful and length-restricted 1-L reduction of B to E. Let $\hat{h} = h \circ g$. Function $\hat{h}$, by Proposition 4.2, is a forgetful 1-L reduction of B to A. It is clearly also length-increasing. We now show that it is one-one as well. For any two different strings x and y, if $|g(x)| = |g(y)|$ then $\hat{h}(x) \neq \hat{h}(y)$ since h is one-one on equal length strings. Consider the case when $|g(x)| = 2^{r(m)} < |g(y)| = 2^{r(n)}$. Then, $|\hat{h}(x)| \leq 2^{c \cdot r(m) + c} = 2^{r(m+1)-1} < 2^{r(n)} = |g(y)| \leq |\hat{h}(y)|$. Therefore, $\hat{h}$ is one-one.

To complete the proof, we need to show that $\hat{h}$ is 1-L-invertible too. We provide only a sketch of this, a complete proof can be found in [1]. The TM M computing the inverse of $\hat{h}$ exploits the fact that h is a forgetful 1-L function. TM M consists of two TMs: the first one, say M_1, computes the inverse of h on strings that are in the range of $\hat{h}$, and the second one, say M_2, computes the inverse of g on the output of M_1. On input z, M_1 first calculates the possible size of $h^{-1}(z)$ (the TM must do this accurately and without scanning the input; this can be achieved since the strings in the range of g are 'widely spaced'). Then, it checks to see which of the two outputs of M_h (the forgetful TM computing h) is a prefix of z. These two outputs must be of the same length and different since M_h is forgetful and h is one-one on equal sized strings. If none match, it rejects, otherwise it outputs the bit whose output matches. Continuing this way, the TM can compute the inverse of h if it exists. The TM M_2, computing g^{-1}, simply deletes trailing ones and the last zero using the delayed processing strategy. ■

Following [1, Corollary 8 & Theorem 10], the following corollary can be easily shown.

Corollary 4.5 *For any class $\mathcal{C}$ closed under 1-NL reductions, the sets in the $\leq_m^{1-L}$-complete degree of $\mathcal{C}$ are 2-L-isomorphic to each other but not 1-L-isomorphic.*

5 Concluding Remarks

Our result has a different flavor from the ones in [17, 9]: while [17] and [9] show that if P $\neq$ UP (or, P $\neq$ PSPACE) then *some* $\leq_{1,li}^p$-degree (or, $\leq_1^p$-degree) does not coincide with any p-isomorphism type, we show that if DSPACE(n) $\neq$ NSPACE(n) then *no* $\leq_m^{1-NL}$-degree coincides with a $\leq_m^{1-L}$-degree. Also, we are able to relate complete degrees.

We have also shown that if DSPACE(n) = NSPACE(n) then 'several' $\leq_m^{1-NL}$-degrees coincide with $\leq_m^{1-L}$-degrees. Can this result be strengthened to show that if DSPACE(n) = NSPACE(n) then *every* $\leq_m^{1-NL}$-degree coincides with a $\leq_m^{1-L}$-degree? Unfortunately, proving such a result is as hard as proving DSPACE(n) $\neq$ NSPACE(n). This is because there exists at least one $\leq_m^{1-NL}$-degree that does not coincide with any $\leq_m^{1-L}$-degree. This degree is the smallest $\leq_m^{1-NL}$-degree, i.e., the $\leq_m^{1-NL}$-degree containing the finite sets. By Lemma 4.3, if this degree coincides with a $\leq_m^{1-L}$-degree then any two sets in the degree would reduce to each other via size-increasing reductions. However, this is not possible as the degree contains finite sets.

An important corollary of our result is the upward separation given in Corollary 3.7. It is easy to obtain downward separation results, i.e., to show that if two classes $\mathcal{C}_1$ and $\mathcal{C}_2$ are distinct then some other two classes that are properly contained in $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively are also distinct. An example is that if E $\neq$ NE then P $\neq$ NP. However, it has been hard to show an upward separation, i.e., if two classes $\mathcal{C}_1$ and $\mathcal{C}_2$ are distinct then two classes that properly contain $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively are also distinct. Corollary 3.7 provides several such results—one each for every class closed under nondeterministic logspace reductions and properly contained in DSPACE(n), e.g., NLOG, NSPACE($\log^2 n$), NSPACE($\log^3 n$), ... etc.

For logspace and polynomial-time complete degrees, we have been able to show only one direction of a possible characterization, and that too for some classes only. It would be interesting to know if there is a class $\mathcal{C}$ such that the $\leq_m^p$-complete degree of $\mathcal{C}$ coincides with the $\leq_m^{log}$-complete degree of $\mathcal{C}$ iff E = DSPACE(n).

Acknowledgments

The author wishes to thank the two anonymous referees whose thoughtful comments helped improve the paper. Also, thanks are due to Eric Allender for going through the paper carefully, and suggesting improvements.

References

[1] M. Agrawal. On the isomorphism problem for weak reducibilities. To appear in *J. Comput. Sys. Sci.*. A preliminary version appeared in *Proc. 9th Structure in Complexity Theory Conference*, pages 338–355, 1994.

[2] M. Agrawal and E. Allender. An isomorphism theorem for circuit complexity. In *Proc. 11th Conference on Computational Complexity*, pages 2–11, 1996.

[3] M. Agrawal and S. Biswas. Polynomial isomorphism of 1-L-complete sets. In *Proc. 8th Structure in Complexity Theory Conference*, pages 75–80, 1993.

[4] E. W. Allender. Isomorphisms and 1-L reductions. *J. Comput. Sys. Sci.*, 36(6):336–350, 1988.

[5] L. Berman. *Polynomial Reducibilities and Complete Sets.* PhD thesis, Cornell University, 1977.

[6] L. Berman and J. Hartmanis. On isomorphism and density of NP and other complete sets. *SIAM J. Comput.*, 1:305–322, 1977.

[7] H. Burtschick and A. Hoene. The degree structure of 1-L reductions. In *Proc. Math. Foundation of Computer Science*, pages 153–161. Springer Lecture Notes in Computer Science 629, 1992.

[8] S. Fenner, L. Fortnow, and S. Kurtz. The isomorphism conjecture holds relative to an oracle. In *Proc. 33rd IEEE Sypm. on Foundations of Computer Science*, pages 30–39, 1992. To appear in SIAM J. Comput.

[9] S. Fenner, S. Kurtz, and J. Royer. P = PSPACE iff every 1-degree collapses. In *Proc. 30th IEEE Sypm. on Foundations of Computer Science*, pages 624–629, 1989.

[10] K. Ganesan and S. Homer. Complete problems and strong polynomial reducibilities. In *Proc. Symposium on Theoretical Aspects of Computer Science*, pages 240–250. Springer Lecture Notes in Computer Science 349, 1988.

[11] J. Hartmanis. On log-tape isomorphisms of complete sets. *Theoretical Computer Science*, pages 273–286, 1978.

[12] J. Hartmanis, N. Immerman, and S. Mahaney. One-way log-tape reductions. In *Proc. 19th IEEE Sypm. on Foundations of Computer Science*, pages 65–72, 1978.

[13] J. Hartmanis and S. Mahaney. Languages simultaneously complete for one-way and two-way log-tape automata. *SIAM J. Comput.*, 10(2):383–390, 1981.

[14] L. A. Hemchandra and A. Hoene. Collapsing degrees via strong computation. In *Proc. International Colloquium on Automata, Languages and Programming*, pages 393–404. Springer Lecture Notes in Computer Science 510, 1991.

[15] N. Immerman. Nondeterministic space is closed under complementation. *SIAM J. Comput.*, 17:935–938, 1988.

[16] D. Joseph and P. Young. Some remarks on witness functions for nonpolynomial and noncomplete sets in NP. *Theoretical Computer Science*, 39:225–237, 1985.

[17] K. Ko, T. Long, and D. Du. A note on one-way functions and polynomial-time isomorphisms. *Theoretical Computer Science*, 47:263–276, 1987.

[18] S. Kurtz, S. Mahaney, and J. Royer. The structure of complete degrees. In A. Selman, editor, *Complexity Theory Retrospective*, pages 108–146. Springer-Verlag, 1988.

[19] S. Kurtz, S. Mahaney, and J. Royer. The isomorphism conjecture fails relative to a random oracle. In *Proc. 21st ACM Sypm. on Theory of Computing*, pages 157–166, 1989.

[20] L. J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic*. PhD thesis, Massachusetts Institute of Technology, 1974.

[21] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26:279–284, 1988.

[22] O. Watanabe. On one-one polynomial time equivalence relations. *Theoretical Computer Science*, 38:157–165, 1985.