NP-Creative Sets: A New Class of Creative Sets in NP[*]

Manindra Agrawal

School of Mathematics
SPIC Science Foundation
Madras - 600 017, INDIA
email : manindra@ssf.ernet.in

&

Somenath Biswas

Department of Computer Science and Engineering
Indian Institute of Technology
Kanpur - 208 016, INDIA
email : sb@iitk.ernet.in

[*]A preliminary version of this paper was presented at the Eleventh Conference on Foundation of Software Technology and Theoretical Computer Science, India.

Abstract

We obtain a new definition of creativeness for NP, called NP-creativeness. We show that all NP-creative sets are NP-complete and provide strong evidence that all known NP-complete sets are NP-creative. We also show that all NP-creative sets are complete under exponentially honest reductions and contain an infinite NP subset in their complement (in other words, they are not NP-simple).

1 Introduction

The notion of *creative* sets plays an important role in the recursion theory: it provides an alternative characterization of many-one completeness for the class of recursively enumerable sets, and this characterization, in turn, helps in demonstrating that all many-one r.e.-complete sets are recursively isomorphic [7]. Naturally then, the question arises: are there such alternative characterizations of complete sets for bounded complexity classes as well? And if yes, then do these characterizations help us to obtain new properties of complete sets for bounded complexity classes? The most obvious way of obtaining such a characterization would be to translate the definition of creativeness to the polynomial settings and then see if the complete sets are equivalent to such creative sets (of course, one has to get a different definition for every class). This approach has been taken in several papers, e.g., [6, 5, 10] etc. In [10], definitions of creativeness for classes E, EXP, NE and NEXP were obtained and it was shown that these creative sets are equivalent to the complete sets for E, EXP, NE and NEXP respectively. Thus, for these classes, creativeness indeed provides an alternative definition of completeness and it has been used to obtain some interesting properties of complete sets for EXP and NEXP in [11], and identify interesting connections with the isomorphism question in [9] (also see the survey paper by Selman [8]). For the class NP, Joseph and Young obtained the definition of *k-completely creative* sets and showed that such sets are all NP-complete.[†] In [10], Wang defined *k-creative* sets and showed that k-completely creative sets are also k-creative. However, neither of the above two definitions, viz., k-completely creative and k-creative, have been able to successfully capture NP-complete sets. In fact, most of the *natural* complete sets for NP are not known to be even k-creative (one notable exception is the Bounded Tiling problem, which was shown to be 1-completely creative by Homer [4]). Also, it is not known if all k-creative sets are NP-complete.

Wang [10] has observed that the hardness of creative sets defined for bounded complexity classes depends on the particular indexing chosen to present the underlying language class. This observation helped him to obtain the definitions of creativeness for EXP and NEXP in [10].

[†]Joseph and Young called these sets k-creative sets while Wang [10] called these sets k-completely creative sets. We shall use Wang's nomenclature throughout.

In this paper, we extend this observation and propose a uniform scheme of obtaining the definitions of creativeness for various time and space bounded complexity classes. The definitions of creativeness, under this scheme, for classes EXP and NEXP turn out to be equivalent to the existing ones. However, for NP, our definition, called NP-*creativeness*, is a more general one. We show that all NP-creative sets are NP-complete and though the converse is hard to prove or disprove (it relativizes both ways even under the assumption P $\neq$ NP), we show that all k-completely creative sets are NP-creative, as well as provide strong evidence that all natural NP-complete sets are also NP-creative. However, our definition appears incomparable to the notion of k-creativeness defined in [10] as there are k-creative sets that are not known to be NP-creative and vice-versa.

To show that natural complete sets are NP-creative, we define a stronger notion of completeness, called *strong completeness*, such that all strongly complete sets for NP are NP-creative and then observe that natural complete sets are strongly complete. Strongly complete sets form an interesting subclass of NP-complete sets: for any set B in NP and any strongly complete set A, there is a reduction of B to A such that the TM computing the reduction must output bits 'regularly'.

We also prove an interesting analogue of a result about creative sets in recursion theory: no NP-creative set is NP-*simple*, where NP-simple sets are those sets in NP whose complements do not contain any infinite NP-subset. Homer [4] showed that the complements of k-completely creative sets that have *honest* productive functions contain an infinite NP-subset and asked if the assumption of honesty can be eliminated. The above result answers this in positive as all k-completely creative sets are NP-creative. On the other hand, our result appears incomparable to a result shown in [11]: the complements of all k-creative sets that have honest productive functions are *p-levelable* and therefore, contain an infinite polynomial-time subset. Finally, we show that all NP-creative sets are also complete under *exponentially honest* reductions.

The paper is organized as follows. Section 2 contains some basic definitions and notations used. In section 3, we define and study NP-creativeness. In section 4, we define the strongly complete sets and show that such sets in NP are NP-creative. We also observe that natural NP-complete sets are strongly complete for NP. In section 5, we obtain some properties of NP-creative sets and consider the question of all NP-complete sets being NP-creative. In section 6, we propose a uniform scheme of obtaining creativeness and using it, obtain the definitions for various other classes which turn out to be equivalent to earlier definitions. The last section, section 7, contains some concluding remarks.

2 Preliminaries

2.1 Notations

Languages are defined over the alphabet $\Sigma = \{0, 1\}$. Sometimes we refer to strings over $\{0, 1, \#\}$, which should be taken as being over $\{00, 01, 11\}$. We use a standard isomorphism between strings and natural numbers to remove any distinction between them. For any string s, $|s|$ denotes the length of s.

Multiple work-tape Turing machines (TMs) with a read-only input tape and a one-way write-only output tape are our model for computation. Unless otherwise stated, our machines are non-deterministic. We consider a fixed encoding scheme of such machines into strings with the proviso that the *information bearing part of the code of a TM begin and end with a 1.* This property of the indexing scheme is formally stated as follows,

$$(\forall i)(\forall m)(\forall n)[\phi_i = \phi_{0^m i 0^n} \text{ and } (\forall x)[T_{M_{0^m i 0^n}}(x) = T_{M_i}(x)]]$$

using the notation given below. We shall refer to this property as the *paddability* property of the indexing scheme.

Remark. Although we make use of this property extensively, the proofs do not critically depend on this indexing scheme. It has been chosen only because it makes many proofs simpler. All the theorems in this paper would hold for any reasonable indexing scheme as the required property can be established by padding the index by non-reachable states also.

TMs are indexed by the strings that encode them. For any string i,

- M_i denotes the TM whose code is i.
- $T_{M_i}(x)$ is the computation time (as defined in [1]) of M_i on input x ($T_{M_i}(x) = 0$ if M_i does not halt on x).

- $T^k_{M_i}(x)$ denotes the time taken by M_i to output k bits on input x (if the length of the output of M_i on x is less than k, then $T^k_{M_i}(x)$ is assumed to be *zero*).

- $S_{M_i}(x)$ is the space taken by M_i on the work tape(s) during the computation on input x.

- W_i is the subset of Σ^* accepted by M_i (M_i *accepts* input x if it halts in an accepting state on x). Clearly, $\{W_i\}_{i \in N}$ is an enumeration of all r.e. sets (N is the set of natural numbers).

- ϕ_i denotes the partial function from Σ^* to Σ^* computed by M_i.

For a set of TM indices I, let $\mathcal{L}(I) = \{W_i \mid i \in I\}$. $\mathcal{L}(I)$ is called the class of languages *presented* by I.

p_1, p_2, p_3, ... is the sequence of polynomial functions with $p_i(n) = n^{|i|} + 2^{|i|}$. Note that this definition is different from the standard one which defines $p_i = n^i + i$. This definition was first used by Wang [10] to ensure that p_i does not grow faster than exponential in the length of i.

2.2 Basic Definitions

Set A is *hard* for class $\mathcal{C}$ under polynomial-time reductions (or simply, $\mathcal{C}$-hard) if for every $B \in \mathcal{C}$ there is a polynomial-time many-one function f such that for every x, $x \in B$ iff $f(x) \in A$. A is *complete* for class $\mathcal{C}$ under polynomial-time reductions (or simply, $\mathcal{C}$-complete) if A is hard for $\mathcal{C}$ under polynomial-time reductions and $A \in \mathcal{C}$.

Classes P, NP^k ($k > 0$), NP, E, EXP, NEXP are defined in the usual way with NP^k = $NTIME(n^k)$, E = $DTIME(2^{O(n)})$ and EXP = $DTIME(2^{n^{O(1)}})$.

The following TM index sets present classes P, NP and NP^k ($k > 0$) respectively.

$$\begin{aligned}
PM &= \{i \mid M_i \text{ a DTM and } (\forall x)[T_{M_i}(x) \le p_i(|x|)]\}, \\
NPM &= \{i \mid M_i \text{ a NDTM and } (\forall x)[T_{M_i}(x) \le p_i(|x|)]\}, \\
NPM^k &= \{i \mid M_i \text{ a NDTM and } (\forall x)[T_{M_i}(x) \le |i| \cdot |x|^k + |i|]\}, \text{ for any } k > 0.
\end{aligned}$$

Productive sets over a language class $\mathcal{C}$ are those sets that have a function witnessing the fact that the set does not belong to the class $\mathcal{C}$. Creative sets are defined to be the complements of productive sets. While defining creativeness for bounded complexity classes, one encounters the following problem, as first noted by Wang [10] — depending on the index set chosen to present a language class, the complexity of the creative set defined over this class may vary, e.g., while Ko and Moore [6] show that there are no creative sets over P in EXP, Wang [10] shows, using a different indexing of polynomial-time languages, that there *are* creative sets over P in EXP (in fact, even in E). Therefore, we use the index set, instead of the language class, to define creativeness and productiveness.

Definition 2.1 For a given set of functions F and a set of TM indices I, language A is (F, I)*-productive* if there is a function $f \in F$ such that for every $i \in I$, $f(i) \in W_i$ iff $f(i) \in \bar{A}$. Function f is called (F, I)*-productive function* for A.

Definition 2.2 Set A is (F, I)*-creative* if A is r.e. and $\bar{A}$ is (F, I)-productive.

It is clear by the above definitions that if A is an (F, I)-productive function then A does not belong to the class $\mathcal{L}(I)$. Thus such sets can be viewed as effectively diagonalizable sets over the class $\mathcal{L}(I)$. When the meaning is clear by the context, we shall refer to (F, I)-productive functions simply as productive functions. When F is the class of polynomial-time functions, we shall write (p, I)-creative for (F, I)-creative.

Following are the standard definitions of creativeness for some classes in our notation.

- For the class r.e. [7]: (rec, N)-creative (referred to as *completely creative* in [7]), where *rec* is the class of total recursive functions and N is the set of natural numbers.
- For the classes EXP and NEXP [10]: (p, PM)-creative and (p, NPM)-creative (referred to as *completely creative over* P and *completely creative over* NP in [10]) respectively.
- For NP [5]: (p, NPM^k)-creative (referred to as *k-creative* in [5] and *k-completely creative* in [10]) for $k > 0$.

Remark. Our notions of creativeness and productiveness are referred to as *completely* creativeness and *completely* productiveness in [7] and [10]. According to [7], set A is (F, I)-productive if there is a function $f \in F$ such that for every $i \in I$, $W_i \subseteq A \Rightarrow f(i) \in A - W_i$. Creative sets are, as usual, complements of the productive sets. For the classes EXP, NEXP and r.e., the two different notions of creativeness are known to be equivalent (see [7] and [10]). For NP, as far as k-creativeness is concerned, the question of their equivalence is open although it has been proved that under certain conditions k-creativeness implies k-complete creativeness [10]. However, surprisingly, for our definition, they are different if P $\neq$ NP (as shown in the next remark).

For our definition of creativeness, we shall require the class of languages recognized by TMs working in *constant time*.

$$\text{CONST} = \{W_i \mid (\exists c)(\forall x)[T_{M_i}(x) \leq c]\}.$$

The following proposition lists some basic properties of the class CONST.

Proposition 2.3 *1.* CONST *is the smallest class of languages containing all finite sets and closed under union, complementation and right concatenation with* Σ^**.*

2. CONST *is a strict subclass of regular languages.*

3 NP-creative sets

As we shall see, by considering different index sets presenting CONST, one can obtain complete sets for widely different classes: for NP as well as r.e. In this section, we obtain and study the definition for the class NP. We can make use of any of the following presentations of CONST for NP.

$$CM_{NP(r)} = \{i \mid M_i \text{ a NDTM and } (\forall x)[T_{M_i}(x) \leq |i|^{r+1}]\} \text{ for } r \geq 0.$$

Of these, we choose the set $CM_{NP(0)}$. We make the zero implicit and write it as simply CM_{NP}. It shall be shown later (Lemma 3.5) that this choice does not change the class of creative sets.

Definition 3.1 Set A is NP-*creative* if $A \in$ NP and A is (p, CM_{NP})-creative.

We take NP-creativeness to be the definition of creativeness for NP. For this definition, we immediately note that

Proposition 3.2 *If A is* NP*-creative then $A \notin$* CONST.

Remark. If we use the definition of creativeness (and productiveness) given in the previous remark, then some trivial sets in P can be shown (p, CM_{NP})-productive. An example is the set TALLY $= \{1^n \mid n \geq 1\}$, which is (p, CM_{NP})-productive with productive function $h(i) = 1^{|i|+1}$—Given any index $i \in CM_{NP}$, it is easy to see that either W_i is not a subset of TALLY or W_i contains strings of size at most $|i|$. We need not bother about the first case and if the second case is true then $h(i) \in$ TALLY $-W_i$.

3.1 NP-creative sets are NP-complete

At a first glance, the NP-creativeness defined above appears to be a trivial definition since the (p, CM_{NP})-productive sets diagonalize only over the class CONST which is a very trivial class. However, it is the chosen indexing of the language class over which creativeness is defined that makes the creative set difficult or easy, not the language class itself. In fact, using a different indexing of the class CONST (see section 6), it can be shown that creative sets over this indexing are r.e.-complete! The reason for this seemingly strange behavior is not too difficult to see. It shall be made clear by the following theorem which shows that NP-creative sets are NP-complete.

Theorem 3.3 *Every* NP*-creative set is* NP*-complete.*

Proof. For $B \in$ NP, let M_B be the NDTM recognizing B with running time bounded by $p_i(n)$ for some polynomial p_i. Define NDTM $M_{g(x)}$ as—

> On input z, run M_B on x for at most $p_i(|x|)$ steps and accept if M_B accepts x.

Note that the function g can be computed in polynomial-time and that $M_{g(x)}$ works *independently of all inputs.* It is clear by construction that, $T_{M_{g(x)}}(z) \leq p_k(|x|)$ for some polynomial p_k.

Let $q(x) = g(x)0^{p_k(|x|)}$. From the paddability property of the indexing scheme we have that,

$$\phi_{q(x)} = \phi_{g(x)} \text{ and } (\forall z)[T_{M_{q(x)}}(z) = T_{M_{g(x)}}(z) \leq |q(x)|].$$

Therefore, $q(x) \in CM_{NP}$. Further,

$$W_{q(x)} \;=\; \begin{cases} \emptyset & \text{if } x \notin B \\ \Sigma^* & \text{if } x \in B \end{cases} \tag{1}$$

Let set A be NP-creative with h the productive function for $\bar{A}$. Since $q(x) \in CM_{NP}$ we have, by the definition of creativeness,

$$h(q(x)) \in A \;\;\Leftrightarrow\;\; h(q(x)) \in W_{q(x)} \tag{2}$$

Now using (1) & (2) we get, $x \in B \Leftrightarrow h(q(x)) \in W_{q(x)} \Leftrightarrow h(q(x)) \in A$. Therefore, $f = \lambda x.\ h(q(x))$ reduces B to A. ■

To reduce any language in NP to the given NP-creative language, the above proof constructs a TM index for each instance of the NP language and then uses the productive function on these indices to yield a reduction. TMs coded by these indices work independently of the input and therefore these TMs need work only for a constant time. This is the reason why the creativeness over such trivial class of languages yields NP-complete sets. Moreover, it is easy to see that if we increase the bound on the time of TMs in CM_{NP} from linear in index length to, say, exponential, the corresponding creative language can be shown NEXP-complete. Thus, a variation in the bound on the time of TMs would vary the complexity of the corresponding creative language, while the language class over which creativeness is defined remains the same, i.e., CONST. This gives us a uniform scheme of obtaining definitions of creativeness for various classes. In section 6, we consider it in more detail.

Remark. Our definition of NP-creativeness fails to satisfy an important property of creative sets: the complements of creative sets (i.e., the productive sets) must diagonalize over an 'interesting' class of languages. However, our motivation is to obtain a definition that characterizes NP-complete sets; and our definition succeeds in this to a large extent.

There exist NP-creative sets in NP. The following 'universal' sets are the most fundamental NP-creative sets.

$K_r = \{i \mid M_i \text{ is an NDTM and accepts } i \text{ in at most } |i|^{r+1} \text{ steps}\}$, for $r \geq 0$.

For these sets we note that,

Proposition 3.4 *For every $r \geq 0$, $K_r \in$ NP and is NP-creative with identity productive function.*

One may wonder as to why have we chosen (p, CM_{NP})-creativeness as the definition for NP instead of $(p, CM_{NP(r)})$-creativeness for some $r > 0$. Theorem 3.3 holds for $(p, CM_{NP(r)})$-creative sets as well and such sets do exist in NP (the set K_r is, in fact, $(p, CM_{NP(r)})$-creative with identity productive function). The following lemma shows the robustness of our definition by proving that every (p, CM_{NP})-creative set is also $(p, CM_{NP(r)})$-creative for any $r > 0$.

Lemma 3.5 *For any $r \geq 1$, A is (p, CM_{NP})-creative iff A is $(p, CM_{NP(r)})$-creative.*

Proof. $(\Leftarrow)$. trivial.
$(\Rightarrow)$. Let A be (p, CM_{NP})-creative with productive function h. Define function h_r as—$h_r(i) = h(i0^{|i|^{r+1}})$. Now, by the paddability property of the indexing scheme, for every $i \in CM_{NP(r)}$, $i0^{|i|^{r+1}} \in CM_{NP}$. So, for every $i \in CM_{NP(r)}$, $h_r(i) \in A$ iff $h(i0^{|i|^{r+1}}) \in A$ iff $h(i0^{|i|^{r+1}}) \in W_{i0^{|i|^{r+1}}}$ iff $h_r(i) \in W_i$. ■

3.2 K-completely creative and NP-creative sets

In this subsection, we compare our notion of creativeness with k-complete creativeness defined in [5]. The following result is immediate.

Proposition 3.6 *If $A \in$ NP and is k-completely creative then A is NP-creative.*

Proof. Since a k-completely creative set is (p, NPM^k)-creative and CM_{NP} is contained in NPM^k, the result follows. ■

However, the converse does not appear to be true. To prove an NP-creative set k-completely creative for some $k > 0$, one seems to require constraints on the length of the productive function. The following theorem is the best we could obtain.

Theorem 3.7 *If A is $(p, CM_{NP(r)})$-creative with productive function h such that for some $t \leq r$, $|h(x)| = O(|x|^t)$, then A is $\lfloor r/t \rfloor$-completely creative.*

Proof. Let $k = \lfloor r/t \rfloor$. Define TM $M_{g(i)}$ as—

On input x, accept iff M_i accepts x in $|i| \cdot |i|^{2 \cdot t \cdot k + 1/2} + |i|$ steps.

Function g can be chosen such that $|i|^2 \leq |g(i)| = O(|i|^2)$. Clearly g is a polynomial-time function and $T_{M_{g(i)}}(x) = O(|i|^{2 \cdot t \cdot k + 3/2}) \leq |i|^{2 \cdot t \cdot k + 2}$(a.e. i) $\leq |g(i)|^{t \cdot k + 1}$(a.e. i). Therefore, $g(i) \in CM_{NP(r)}$ (a.e. i). Further, $|h(g(i))| = O(|i|^{2 \cdot t}) \leq |i|^{2 \cdot t + 1/(2k)}$(a.e. i) and therefore, for almost all $i \in NPM^k$, TM $M_{g(i)}$ accepts $h(g(i))$ iff M_i accepts $h(g(i))$. Therefore, we have that for every $i \in NPM^k$ and $i > i_0$ for some large enough i_0,

$$h(g(i)) \in W_i \Leftrightarrow h(g(i)) \in W_{g(i)} \Leftrightarrow h(g(i)) \in A.$$

Thus $h' = h \circ g$ is a productive function for all $i > i_0$ and this can be easily modified to yield a total productive function. E.g., $h'' = \lambda i.\ h'(i0^{i_0}))$ is a total (p, NPM^k)-productive function. ■

As for the notion of k-creativeness defined in [10], our notion appears to be incomparable to it. There are k-creative sets, given in [10], that are neither known to be NP-creative nor NP-complete while natural NP-complete sets like SAT are NP-creative (see next section) but are not known to be k-creative.

4 A strong version of NP-completeness

Are all NP-complete sets NP-creative? The answer to this question appears difficult as it relativizes both ways (see section 5). The goal of this section is to see what subclass of NP-complete sets is NP-creative. We develop a new notion of completeness, called strong completeness, and show that all such complete sets for NP are NP-creative. In the subsection 4.2 below, we show that this notion is possibly general enough to capture all natural NP-complete sets.

We begin by trying to prove that all NP-complete sets are NP-creative using a standard method (see, e.g., [7], [10]) and explain why it does not work.

Let A be NP-creative with productive function h and for some $B \in$ NP, $A \leq_m^p B$ via f. Given any i, construct TM $M_{g(i)}$ as—On input x, compute first $|i|$ bits of $f(x)$ and accept iff M_i accepts these $|i|$ bits in $|i|$ steps.

Now, since for all $i \in CM_{NP}$, M_i works only for $|i|$ steps on every input and $M_{g(i)}$ provides it with the first $|i|$ bits of $f(x)$, M_i would accept these $|i|$ bits iff it accepts $f(x)$. So, for $x = h(g(i))$, $h(g(i)) \in W_{g(i)}$ iff $f(h(g(i))) \in W_i$. If we can ensure that $g(i) \in CM_{NP}$ then by creativeness, we shall have $h(g(i)) \in W_{g(i)}$ iff $h(g(i)) \in A$ and by reduction, $h(g(i)) \in A$ iff $f(h(g(i))) \in B$. Combining these, we get, $f(h(g(i))) \in W_i$ iff $f(h(g(i))) \in B$ and therefore, $f \circ h \circ g$ would be the required productive function for $\bar{B}$, showing B to be NP-creative. But to have $g(i) \in CM_{NP}$, $M_{g(i)}$ must be able to compute the first $|i|$ bits of $f(x)$ for $x = h(g(i))$ in at most $|g(i)|$ steps. This imposes restrictions on f and/or h that are not likely to hold in general.

The above discussion makes us look for a more restricted class of reductions for which the method might work. It is immediately clear that the TM $M_{g(i)}$ needs to generate only the first $|i|$ bits of $f(h(g(i)))$. If this can be done quickly without computing all of $f(h(g(i)))$ then the proof would go through. This motivates the following definition.

Definition 4.1 Function f is a *k-bounded* function, for $k \geq 0$, if there is a polynomial-time DTM M computing f, a constant c and a polynomial p such that

$$(\forall x)(\forall b) T_M^b(x) \leq c \cdot |x|^k + p(b).$$

The idea behind k-bounded functions is that the DTMs computing them would have to output bits regularly after $c \cdot n^k$ steps, i.e., if $c \cdot n^k + t$ steps of computation are over then the number of bits written on the output tape must be at least $p^{-1}(t)$ for some polynomial p. Of course, every polynomial-time computable function is k-bounded for some suitable k. The interesting use of these functions is in the definition of strong completeness.

Definition 4.2 Set A is *k-strongly complete* for the class NP if $A \in$ NP and every set in NP reduces to A via a k-bounded reduction. We say that A is *strongly complete* for NP if there is a k, $k \geq 0$, such that A is k-strongly complete for NP.

Now one can see how the k-bounded reductions are used to restrict the definition of NP-completeness: to a usual NP-complete set, different sets in NP are reducible via k-bounded reductions with *different* k's.

There are sets that are k-strongly complete for NP, in fact, as we show in the subsection 4.2 below, all the natural NP-complete sets appear to be 0-strongly complete. As an example, we show that K_r is 0-strongly complete.

Proposition 4.3 *For any $r \geq 0$, K_r is 0-strongly complete.*

Proof Sketch. Function q, as defined in the proof of Theorem 3.3, is a reduction of set B to K_r. We modify it as: $\tilde{q}(x) \overset{\text{def}}{=} 0^{|x|}q(x)$. The function $\tilde{q}$ also reduces B to K_r (by the paddability property of the indexing scheme) and is a 0-bounded function. ■

K-bounded functions do not appear to be closed under composition. Is there a fixed l such that a composition of any two k- and k'-bounded functions is l-bounded? The answer to this is also probably no. However, a very weak form of closure does hold for these functions.

Lemma 4.4 *Let f_1 be a 0-bounded and f_2 be a k-bounded function. Then $f_1 \circ f_2$ is also a k-bounded function.*

Proof. Let M_1 and M_2 be the DTMs computing f_1 and f_2 respectively. We define the DTM M, computing $f_1 \circ f_2$, such that, on input x, it starts computing f_1 on $f_2(x)$ and generates bits of $f_2(x)$ as and when required. We know that $T^b_{M_1}(x) \leq p(b)$ and $T^b_{M_2}(x) \leq c \cdot |x|^k + p(b)$ for some polynomial p and constant c. Therefore, to compute b bits of $f_1(f_2(x))$, M needs to simulate at most $p(b)$ steps of M_1. This implies that at most $p(b)$ bits of $f_2(x)$ need be generated and given as input to M_1. To compute $p(b)$ bits, M_2 takes at most $c \cdot |x|^k + p(p(b))$ steps. Therefore, $T^b_M(x) = O(|x|^k + p(p(b)) + p(b)) \leq d \cdot |x|^k + p'(b)$ for some polynomial p' and constant d. Therefore, $f_1 \circ f_2$ is a k-bounded function. ■

4.1 Strongly complete sets are NP-creative

Our motivation in defining strongly complete sets was to show that they are all NP-creative. The following theorem accomplishes this.

Theorem 4.5 *If A is strongly complete for* NP *then A is* NP*-creative.*

Proof. The proof proceeds along the lines of the method outlined at the beginning of the section. Let A be k-strongly complete. Let f be the k-bounded reduction of the set K_k to A. Let M be the DTM computing f such that it halts within $p_r(.)$ steps and $T^b_M(x) \leq c.|x|^k + p_r(b)$. Define TM $M_{g(i)}$ as—

> On input x, if $|x| > p_r(|i|)$ then reject. Otherwise, compute first $|i|$ bits of $f(x)$ and accept iff M_i accepts $f(x)$ within $|i|$ steps.

Therefore, $T_{M_{g(i)}}(x) = O((p_r(|i|))^k + p_r(|i|))$. Choosing g and p_r such that $|g(i)| = p_r(|i|)$ for every i, we get, $T_{M_{g(i)}}(x) = O(|g(i)|^k) \leq |g(i)|^{k+1}$ (a.e. i).

Now, by the definition of K_k, we get that for large enough $i \in CM_{NP}$, $g(i) \in W_{g(i)}$ iff $g(i) \in K_k$. By the definition of $M_{g(i)}$, we have that $g(i) \in W_{g(i)}$ iff $f(g(i)) \in W_i$. Combining the two we get, $f(g(i)) \in W_i$ iff $g(i) \in W_{g(i)}$ iff $g(i) \in K_k$ iff $f(g(i)) \in A$ for large enough i.

Thus $f \circ g$ is the required productive function for sufficiently large inputs and from this a productive function that is correct on all inputs can be easily obtained, as was done in the proof of Theorem 3.7. ∎

4.2 Strongly complete and NP-complete sets

Several questions arise concerning the relationship between NP-complete and strongly complete sets. We consider the following three.

- Are all natural NP-complete sets strongly complete?
- Are all NP-complete sets strongly complete?
- Is there a k such that all NP-complete sets are k-strongly complete?

In this subsection, we provide evidence that the answer to the first question is positive and to the other two questions, negative. We begin with showing that several natural NP-complete sets are 0-strongly complete. Towards this, we first define a new padding function, then show that every NP-complete set possessing this function is 0-strongly complete and finally show that standard natural complete sets do posses this function.

For every set A, define

$$LP_A = \{y\#x \mid y \in \{0,1,\#\}^* \wedge x \in A\}$$

Definition 4.6 A is *left-paddable* if there exists a 0-bounded reduction lp_A of LP_A to A.

The following Lemma gives some elementary results about left-paddable sets.

Lemma 4.7 *1. $LP_A \leq_m^p A$.*

2. For every B, if $B \leq_m^p A$ then $B \leq_m^p LP_A$ via a 0-bounded function.

3. A is left-paddable iff for every B, if $B \leq_m^p A$ then $B \leq_m^p A$ via a 0-bounded function.

4. A is NP*-complete and left-paddable iff A is 0-strongly complete for* NP*.*

Proof. 1. & 2. Immediate.
3. ($\Rightarrow$) Let $B \leq_m^p A$ via f. Define, $\tilde{f}$ as: $\tilde{f}(x) = lp_A(1^{|x|}\#f(x))$. By Lemma 4.4, function $\tilde{f}$ is 0-bounded as it is a composition of two 0-bounded functions: lp_A and g with $g(x) = 1^{|x|}\#f(x)$.
($\Leftarrow$) Immediate from 1.
4. Immediate from 3. ∎

Corollary 4.8 *Every left paddable* NP*-complete set is* NP*-creative.*

Remark. We cannot prove the above corollary using the symmetric counterpart of left-padding functions. In fact, *every* set has a *right-padding* function which is 0-bounded: $rp_A(x\#y) = x$. This asymmetry arises because we are using TMs that start scanning from the *left* of the input. It can be eliminated by using random access machines, however, it makes the definition of k-bounded functions and the related proofs messy. Therefore, we have preferred the asymmetric way though it suffers from the defect of being strongly dependent on the computational model.

Left-paddability provides a nice way of exhibiting NP-creativeness of NP-complete sets. For example,

Proposition 4.9 *The following sets are left-paddable.*

1. SAT*, the set of all satisfiable boolean formulas.*

2. HAM*, the set of all directed Hamiltonian graphs.*

3. CLQ*,* CLQ $= \{x\$r \mid$ *graph x contains an r-clique* $\}$ $(\$ \notin \Sigma)$*.*

4. KNP*,* KNP $= \{n_1\$n_2\$\cdots\$n_k\$m \mid$ *there is a subset of first k numbers whose sum is m*$\}$ $(\$ \notin \Sigma)$*.*

Proof.

1. lp_{SAT} is defined as follows —

   On $y\#x$, initially output $|y| + |x|$ trivial clauses, clause i having the form: $v_i \vee \bar{v}_i$, v_i being the i^{th} variable and then output clauses of x.

2. lp_{HAM} is defined as follows —

   On $y\#x$, initially output a directed path containing $|y| + |x|$ vertices. Then output the following modified version of the graph x: pick *some* vertex v of x, split it into two vertices v_1 and v_2, replace all edges $\langle u, v\rangle$ by $\langle u, v_1\rangle$, all edges $\langle v, u\rangle$ by $\langle v_2, u\rangle$, and finally, add an edge from v_1 to the first vertex of the above path, and an edge from the last vertex of the path to v_2.

3. lp_{CLQ} is defined as follows —

   On $y\#x\$r$, initially output $|y| + |x| + |r|$ isolated vertices followed by graph x and then $\$r$.

4. lp_{KNP} is defined as follows —

   On $y\#n_1\$n_2\$\cdots\$n_k\$m$, output $l_1\$l_2\$\cdots\$l_r\$n_1\$\cdots\$n_k\$m$, where $l_i = 0$ for all $1 \le i \le r$ and $r = |y| + |n_1| + \cdots + |n_k| + |m|$.

It is easy to see that all the above functions are 0-bounded and preserve the membership in their respective languages. ∎

Corollary 4.10 *For any $L \in \{\mathrm{SAT}, \mathrm{HAM}, \mathrm{CLQ}, \mathrm{KNP}\}$, L is* NP*-creative.*

For natural NP-complete sets the left-padding function seems to be directly obtainable from the normal padding function of the set. And since all natural NP-complete sets are believed to be paddable, it is not unreasonable to assume that they are left-paddable as well. Are all paddable NP-complete sets left-paddable? The answer is no (see Corollary 4.14). Conversely, are all left-paddable NP-complete sets paddable as well? There is no apparent reason to believe so. The left-padding function need not be one-one or invertible in polynomial-time. However, it is not too difficult to see that left-padding functions of NP-complete sets can always be made size-increasing.

Proposition 4.11 *Any left-paddable* NP*-complete set A has a size-increasing left-padding function.*

Proof. Let lp_A be the left-padding function for A. It can be made size-increasing very simply: $lp'_A(y\#x) = lp_A(1^{p(|y|+|x|+2)}\#x)$ where polynomial p is such that the first b bits of $lp_A(z)$ are computable within $p(b)$ steps. The new function is clearly size-increasing as lp_A while scanning the sequence of $1's$ in the prefix would output at least $|y| + |x| + 2$ bits. ∎

Corollary 4.12 *If A is a left-paddable* NP*-complete set then A is complete for* NP *under* 0*-bounded size-increasing reductions.*

There are NP-complete sets that are *not* 0-complete. We show this by constructing a set whose first few bits are very difficult to compute. For this, we need the definition of p-immune sets: a set is *p-immune* if it is infinite and does not contain any infinite polynomial-time subset [1]. The following p-immune set B will be used in the proof: $x \in B$ iff for every $i \leq \log |x|$, x is not the lexicographically smallest string of length $|x|$ that is accepted by DTM M_i in $p_i(|x|)$ steps. One can easily show that $B \in \mathrm{E}$, is p-immune, and contains at least one string of size n for every n.

Theorem 4.13 *There is an* NP*-complete set A which is not* 0*-complete.*

Proof. Let B be the set defined as above. Define set A as:

$$A = \{y\#x \mid |y| = \lceil \log |x| \rceil \wedge y \in B \wedge x \in K_0\}$$

Clearly, $A \in \mathrm{NP}$ and $K_0 \leq_m^p A$ via f, where $f(x) = y\#x$, y is the smallest string of length $\lceil \log |x| \rceil$ in B (such a y can be generated in polynomial-time by enumerating all elements of B up to length $\lceil \log |x| \rceil$). Suppose A is 0-complete for NP. Let g be the 0-bounded reduction of K_0 to A and j be the index of a TM that accepts all strings in a single step. So, $\{j0^n \mid n \geq 0\} \subset K_0$. Function g can be taken to be size-increasing by Corollary 4.12. Let $g(j0^n) = y\#x$ with $y \in B$ and $x \in K_0$. DTM M_g, on input $j0^n$, would output y within $p(|y|)$ steps (for some polynomial p) and since g is size-increasing, there would be infinitely many such y's. This gives an infinite polynomial-time subset of B which is not possible as B is p-immune. Therefore, A is not 0-strongly complete. ■

Corollary 4.14 *There is a paddable* NP*-complete set which is not left-paddable.*

Proof. Set A constructed above is paddable, since K_0 is. The corollary follows. ■

Remark. We could have proved Theorem 4.13 more easily by utilizing the fact that a 0-bounded function must output a large number of bits without knowing the length of the input. One may find this condition very restrictive and may generalize the 0-bounded functions by supplying the length of the input at the beginning of their computation. Our proof goes through for this more general definition as well.

Is the set A constructed above k-strongly complete for some $k > 0$? We do not know an answer to this. Is it at least NP-creative? The answer is yes which is shown by the following proposition.

Proposition 4.15 *Set A defined in the proof of Theorem 4.13 is* NP*-creative.*

Proof. Define TM $M_{g(i,y)}$ as: on input x, accept iff M_i accepts $y\#0^{|i|}$ within $|i|$ steps. Choose g such that $|g(i,y)| = c \cdot (|i| + |y|)$ for some constant c. Let $m(i)$ be the smallest number such that $m(i) = \lceil \log(|i| + c \cdot (|i| + m(i))) \rceil$. Clearly, $m(i) = O(\log |i|)$.

Define function h as: $h(i) = y_i\#0^{|i|}g(i, y_i)$, where y_i is the smallest string in B of length $m(i)$. Function h is polynomial-time computable as $m(i) =$

$O(\log |i|)$ and $B \in \mathrm{E}$. We now show that h is the required productive function for $\bar{A}$. For every $i \in CM_{NP}$, $h(i) \in W_i$ iff $y_i \# 0^{|i|} \in W_i$ iff $0^{|i|} g(i, y_i) \in W_{g(i,y_i)}$ (by the definition of $M_{g(i,y_i)}$) iff $0^{|i|} g(i, y_i) \in W_{0^{|i|} g(i,y_i)}$ (by the paddability property of the indexing scheme) iff $0^{|i|} g(i, y_i) \in K_0$ iff $y_i \# 0^{|i|} g(i, y_i) \in A$ (by the definition of A) iff $h(i) \in A$. ■

Are all NP-complete or at least all NP-creative sets strongly complete? It does not appear to be so. Consider the following class of k-creative sets defined in [5].

$$K_k^f = \{f(i) \mid M_i \text{ accepts } f(i) \text{ within } |i| \cdot |f(i)|^k + |i| \text{ steps}\}$$

There is no apparent way of showing K_k^f to be l-complete for some l with f being an arbitrary polynomial-time function except when f is 0-bounded. As is shown by Theorem 4.13, there are polynomial-time functions that are not 0-bounded. So, the sets K_k^f for such f's may be the sets that are not l-complete for any l. However, all of these sets are NP-complete as well as NP-creative. Thus the answer to the last two questions raised at the beginning of the subsection is probably negative.

One may wonder about the usefulness of the concept of strongly complete sets apart from the fact that they are all NP-creative. We feel that they provide a further insight into the difference between the two kinds of NP-complete sets defined in the literature, viz., natural complete sets (all such sets are probably 0-strongly complete) and complete sets defined by structural considerations, e.g., k-creative sets (many of these may not be strongly complete at all).

5 NP-creative and NP-complete sets

So far, we have defined the notion of NP-creativeness for the class NP and seen that all the known NP-complete sets appear to be NP-creative. Proving that all NP-complete sets are indeed NP-creative would be difficult as it immediately implies P $\neq$ NP (follows from Proposition 3.2). Can we at least say that all paddable NP-complete sets are NP-creative? This question turns out to be as difficult as the previous one.

Proposition 5.1 *If all paddable* NP*-complete sets are* NP*-creative then all* NP*-complete sets are* NP*-creative.*

Proof. Suppose that all paddable NP-complete sets are NP-creative. Let A be any NP-complete set. Consider set $RP_A = \{x\#y \mid x \in A \wedge y \in \{0, 1, \#\}^*\}$. RP_A is a paddable NP-complete set and therefore NP-creative. RP_A reduces to A via function f, $f(x\#y) = x$. Define TM $M_{g(i)}$ as: on input x, compute the first $|i|$ bits of $f(x)$ and accept iff M_i accepts these bits within $|i|$ steps. We can choose g such that $T_{M_{g(i)}}(x) \leq |g(i)|$. Let h be the productive function for $\overline{RP}_A$. Now, $f \circ h \circ g$ would be the productive function for $\bar{A}$ as $f(h(g(i))) \in W_i$ iff $h(g(i)) \in W_{g(i)}$ iff $h(g(i)) \in RP_A$ iff $f(h(g(i))) \in A$. ■

This is an interesting property of NP-creative sets showing that if the property of NP-creativeness is invariant under polynomial isomorphism then it is also invariant under polynomial-time many-one equivalence.

Are there relativized worlds where the class of NP-creative sets is equal to or properly contained in the class of NP-complete sets? Such oracles already exist in the literature, though they were not constructed for this purpose. We first note that all our definitions and theorems relativize. To make the two classes equal, we choose the oracle A for which $\mathrm{NP}^A = \mathrm{EXP}^A$, and then since all EXP-complete sets are EXP-creative (see Proposition 6.1 below), and therefore NP-creative, the result follows. Using the following theorem one can exhibit an oracle that separates these two classes while also separating P and NP (if P = NP then obviously the two classes do not coincide).

We say that the function f is *exponentially honest* if there is a polynomial p such that

$$(\forall x) p^{-1}(|f(x)|) \leq |x| \leq 2^{p(|f(x)|)}.$$

Remark. This definition is slightly different from the one given by Ganesan and Homer [2], who define exponentially honest functions as satisfying the property

$$(\forall x) p^{-1}(|f(x)|) \leq |x| \leq 2^{|f(x)|}.$$

Theorem 5.2 *Let A be an* NP*-creative set. Then A is complete for* NP *under exponentially honest reductions.*

Proof. Since $A \in \mathrm{NP}$, there is a DTM M_A recognizing the set A in $O(2^{p(n)})$ time for some suitable polynomial p. Define NDTM $M_{g(x)}$ as: on input z, if $p(|z|) \geq \log |x|$ then accept z iff $x \in K_0$ else accept z iff $z \notin A$ (using M_A).

It can be seen that $T_{M_{g(x)}}(z) \leq p_r(|x|)$ for some polynomial p_r. Choosing g such that $|g(x)| \geq p_r(|x|)$, we get that $g(x) \in CM_{NP}$ for every x and

$$W_{g(x)} = \begin{cases} \{\bar{A}\}_{<n} \cup \Sigma_{\geq n} & \text{if } x \in K_0 \\ \{\bar{A}\}_{<n} & \text{otherwise} \end{cases}$$

where $n = p^{-1}(\log|x|)$. Now, let h be the productive function for $\bar{A}$. By creativeness we have, $h(g(x)) \in W_{g(x)}$ iff $h(g(x)) \in A$. If for some x, $|h(g(x))| < p^{-1}(\log|x|)$ then $h(g(x)) \in A$ iff $h(g(x)) \in W_{g(x)}$ iff $h(g(x)) \in \bar{A}$, a contradiction. Therefore for every x, $|h(g(x))| \geq p^{-1}(\log|x|)$ and then we have that $x \in K_0$ iff $h(g(x)) \in W_{g(x)}$ iff $h(g(x)) \in A$. Since K_0 is complete for NP under size-increasing reductions (recall that it is 0-strongly complete), it follows that A is complete for NP under exponentially honest reductions.

■

In [3], Hartmanis and Hemachandra have constructed an oracle A such that $\mathrm{P}^A \neq \mathrm{NP}^A$ and NP^A contains complete sets that have arbitrarily large 'gaps'. In other words, there are complete sets that do not contain any string of length between n and 2^{2^n} for infinitely many n's. Such sets cannot be exponentially honest complete for NP^A and therefore there are complete sets in NP^A that are not NP^A-creative.

Theorem 5.2 is a rather weak result about the honesty of reductions to NP-creative sets. Can one show that all NP-creative sets are complete under size-increasing reductions? We do not have an answer to this, however, what we can show is that the complements of all these sets have infinite NP subsets. Note that if NP-creative sets were complete under size-increasing reductions, the result would immediately follow.

Theorem 5.3 *If A is* NP*-creative then $\bar{A}$ contains an infinite subset in* NP.

Proof. Let h be the productive function for $\bar{A}$. Define TM $M_{g(x)}$ as: on input z, if $|z| \geq |x|$ then reject; otherwise, accept if $x \in A$. As in the proof of Theorem 3.3, one can choose a polynomial-time computable g such that $T_{M_{g(x)}}(z) \leq |g(x)|$. Consider the sets $T = \{x \mid |x| \leq |h(g(x))|\}$ and $S = \{z \mid (\exists x) x \in T \wedge z = h(g(x))\}$. It is clear that $T \in \mathrm{P}$ and $S \in \mathrm{NP}$. Further, since $W_{g(x)}$ contains strings of length less than $|x|$ only, for every $x \in T$, $h(g(x)) \notin W_{g(x)}$, and therefore, by creativeness of A, $h(g(x)) \notin A$. Thus we get that $S \subseteq \bar{A}$.

We now show that if S is finite then P = NP. First of all, we note that if S is finite then T is also finite. Now the following polynomial-time procedure recognizes the set A.

1. Input x.
2. $y = x$.
3. while $y \notin T$ do: $y = h(g(y))$.
4. Accept iff $y \in A$.

The above procedure works in polynomial-time since in every iteration of the while loop, the length of y decreases by at least one and since $T \cap A$ is a finite set. We now show that $y \in A$ iff $x \in A$ after every iteration of the while loop. If $y \notin T$ then $|h(g(y))| < |y|$. Now by the definition of TM $M_{g(y)}$, $h(g(y)) \in W_{g(y)}$ iff $y \in A$. And by creativeness, $h(g(y)) \in W_{g(y)}$ iff $h(g(y)) \in A$. Combining these we get: if $|h(g(y))| < |y|$ then $h(g(y)) \in A$ iff $y \in A$. This proves the correctness of the above procedure.

Thus, if P $\neq$ NP, then S must be infinite and then it would be an infinite NP subset of $\bar{A}$. On the other hand, if P = NP, then $\bar{A}$ would itself be an infinite polynomial-time set as no co-finite set can be NP-creative (Proposition 3.2). Therefore, $\bar{A}$ contains an infinite subset in NP. ■

From the above theorem, it follows that no NP-creative set can be NP-*simple*, a set whose complement does not contain any infinite subset in NP. This result is analogous to the one for creative sets in recursion theory [7] and answers a question posed in [4].

6 A uniform scheme of defining creativeness

As observed in subsection 3.1, the creativeness definitions for the higher classes can be obtained by simply changing the bound on the time of TMs in the set presenting the class CONST. So, the general scheme of obtaining the definition of creativeness for any class $\mathcal{C}$ is: identify a *suitable* TM presentation $CM_{\mathcal{C}}$ of CONST and a suitable class of functions F; define a set A to be $\mathcal{C}$*-creative* if $A \in \mathcal{C}$ and A is $(F, CM_{\mathcal{C}})$-creative. The suitable presentation of CONST for a class containing languages recognized in time (space) $T(n^{O(1)})$

$(S(n^{O(1)}))$ would be the set of TMs M_is with $T(|i|)$ $(S(|i|))$ being the bound on their time (space).

Using this scheme, we obtain the creativeness definitions for the classes PSPACE, EXP, NEXP and r.e. For EXP, NEXP and r.e. the definitions we obtain are equivalent to the ones already defined in the literature. For PSPACE, our definition is the only one available, and for this we show the equivalence between creative and complete sets (under logspace reductions).

We first define the suitable TM presentations of CONST for classes EXP, NEXP, r.e. and PSPACE.

$$\begin{aligned}
CM_E &= \{i \mid M_i \text{ is a DTM and } (\forall x) T_{M_i}(x) \leq 2^{|i|}\} \\
CM_{NE} &= \{i \mid M_i \text{ is an NDTM and } (\forall x) T_{M_i}(x) \leq 2^{|i|}\} \\
CM_{RE} &= \{i \mid (\forall x)(\forall y) T_{M_i}(x) = T_{M_i}(y)\} \\
CM_{PSPACE} &= \{i \mid M_i \text{ is a DTM and } (\forall x) S_{M_i}(x) \leq |i|\}
\end{aligned}$$

We shall choose the class of polynomial-time functions for EXP and NEXP, the class of recursive functions for r.e. and the class of logspace functions for PSPACE and define the creative sets for these classes as explained above.

For the classes EXP, and NEXP, our definitions can be seen as obtained from those in [10] by dropping the non-constant term in the time bound. Exactly the same proof as in [10] can be used to show that all EXP-creative (NEXP-creative) sets are EXP-complete (NEXP-complete). The equivalence of the two definitions follows as all creative sets in the sense of [10] for EXP and NEXP are equivalent to respectively EXP- and NEXP-complete sets. The same can be shown to hold for r.e. sets as well as for logspace complete sets for PSPACE using the same proof idea. Therefore, we have

Proposition 6.1 1. *A is r.e.-creative iff A is (rec, N)-creative iff A is r.e.-complete.*

2. *A is* NEXP*-creative iff A is (p, NPM)-creative and $A \in$* EXP *iff A is* NEXP*-complete.*

3. *A is* EXP*-creative iff A is (p, PM)-creative and $A \in$* EXP *iff A is* EXP*-complete.*

4. *A is* PSPACE*-creative iff A is complete for* PSPACE *under logspace reductions.*

7 Concluding remarks

Our aim was twofold: first to obtain the definition of creativeness for NP that characterizes the complete sets and then, using such creative sets, study the properties of the complete sets. Although we have given a new, and more general, definition of creativeness for NP, we have not been completely successful on either accounts. Firstly, we could not show that all NP-complete sets are NP-creative and secondly, we have not been able to obtain very significant new results about NP-complete sets. Nevertheless, our definition appears to cover all known NP-complete sets and provides some interesting new results, e.g., complements of all k-completely creative sets contain infinite NP-subsets.

An interesting aspect of our definitions is that all the presentations of CONST that we have considered in this paper are productive over themselves. For example, the set CM_{NP} is (p, CM_{NP})-productive, CM_E is (p, CM_E)-productive, CM_{RE} is (p, CM_{RE})-productive. We do not know if this property of 'self-productiveness' has any deeper reasons behind it.

One can see, therefore, that there is much we do not know about creative sets. We believe, and there are strong reasons to do so, that a study of these sets would increase our understanding of complete sets.

Acknowledegment: We thank two anonymous referees for many helpful suggestions and improvements.

References

[1] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1988.

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

[3] J. Hartmanis and L. Hemachandra. One-way functions and the non-isomorphism of NP-complete sets. *Theoretical Computer Science*, 81(1):155–163, 1991.

[4] S. Homer. On simple and creative sets in NP. *Theoretical Computer Science*, 47:169–180, 1986.

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

[6] K. Ko and D. Moore. Completeness, approximation and density. *SIAM Journal on Computing*, 10:787–796, 1981.

[7] H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, New York, 1967.

[8] A. L. Selman. A survey of one-way functions in complexity theory. *Mathematical Systems Theory*, 25:203–221, 1992.

[9] J. Wang. Some remarks on polynomial time isomorphisms. In *Lecture Notes in Computer Science 468*, pages 144–153, 1990.

[10] J. Wang. On p-creative sets and p-completely creative sets. *Theoretical Computer Science*, 85:1–31, 1991.

[11] J. Wang. Polynomial time productivity, approximations, and levelability. *SIAM Journal on Computing*, 21:1100–1111, 1992.