Kolmogorov Complexity and Instance Complexity
of Recursively Enumerable Sets

Martin Kummer*
Universität Karlsruhe

Abstract

We study in which way Kolmogorov complexity and instance complexity affect properties of r.e. sets. We show that the well-known $2 \log n$ upper bound on the Kolmogorov complexity of initial segments of r.e. sets is optimal and characterize the T-degrees of r.e. sets which attain this bound. The main part of the paper is concerned with instance complexity of r.e. sets. We construct a nonrecursive r.e. set with instance complexity logarithmic in the Kolmogorov complexity. This refutes a conjecture of Ko, Orponen, Schöning, and Watanabe. In the other extreme, we show that all wtt-complete set and all Q-complete sets have infinitely many hard instances.

Key words: Kolmogorov complexity, instance complexity, recursively enumerable sets, complete sets.

AMS (MOS) subject classification: 03D15, 03D32, 68Q15.

1 Introduction

Intuitively, Kolmogorov complexity measures the “descriptional complexity” of a string $x$. It is defined as the length of the shortest program that computes $x$ from the empty input. Accordingly, the Kolmogorov complexity of initial segments of a set $A$ is considered as a measure of the “randomness” of $A$. It is well-known that for r.e. sets the Kolmogorov complexity of initial segments of length $n$ is bounded by $2 \log n$. We show that this bound is optimal and characterize the Turing degrees of r.e. sets which attain this bound as the array nonrecursive degrees of Downey, Jockusch, and Stob [4].

Ko, Orponen, Schöning, and Watanabe [7,12] have recently introduced the notion of instance complexity as a measure of the complexity of individual instances of $A$. Informally, $ic(x : A)$, the instance complexity of $x$ with respect to $A$, is the length of the shortest total program which correctly computes $\chi_A(x)$ and does not make any mistakes on other inputs, but it is permitted to output “don’t know” answers. It is easy to see that the Kolmogorov complexity of $x$ is an upper bound for the instance complexity $ic(x : A)$.
complexity of \( x \) (up to a constant). A set \( A \) has hard instances if for infinitely many \( x \) the instance complexity of \( x \) w.r.t. \( A \) is at least as high as the Kolmogorov complexity of \( x \) (up to a constant which may depend on \( A \)), i.e., the trivial upper bound is already optimal.

Orponen et al. conjectured in [11, 12] that every nonrecursive r.e. set has hard instances ("Instance Complexity Conjecture (ICC)"). Buhrman and Orponen [2] proved ICC for m-complete sets. Tromp [14] proved that the instance complexity of \( x \) w.r.t. any nonrecursive set \( A \) is infinitely often at least logarithmic in the Kolmogorov complexity of \( x \). We construct an r.e. nonrecursive set which attains this lower bound for all \( x \). In particular, this is a counterexample to ICC. On the positive side, we show that ICC holds for wtt-complete sets, Q-complete sets, and hyperhypersimple sets. But ICC fails for a T-complete set, since it fails for an effectively simple set. However, ICC holds for all strongly effectively simple sets. We also investigate a weak version of instance complexity, where programs may not halt instead of giving "don’t know" answers.

The resource-bounded version of instance complexity is also well-studied; we refer the reader to [2, 5, 6, 12].

**Notation and Definitions:**

The notation generally follows [8]. For further recursion theoretic background we refer the reader to [10, 13]. For \( p \in \{0, 1\}^* \), \( l(p) \) denotes the length of \( p \); \( \lambda \) is the empty string. We use the special symbol \( \bot \) to denote the "don’t know" output. \( \chi_A \) is the characteristic function of \( A \). We identify \( \mathbb{N} \) and \( \{0, 1\}^* \) via the canonical correspondence as in [8, p. 11].

**Definition 1.1** (Chaitin, Kolmogorov, Solomonoff)

For any partial recursive mapping \( U : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\} \cup \{\bot\} \) and any \( x \in \{0, 1\}^* \) we define \( C_U(x) = \min\{l(p) : U(p, \lambda) = x\} \), the Kolmogorov complexity of \( x \) in \( U \). If no such \( p \) exists then \( C_U(x) = \infty \).

It is helpful to think of \( U \) as an interpreter which takes a program \( p \) and an input \( z \) and produces the output \( U(p, z) \).

Instance complexity was introduced in [4] in order to study the complexity of single instances of a decision problem.

**Definition 1.2** (Ko, Orponen, Schöning, Watanabe, 1986)

Let \( A \subseteq \{0, 1\}^* \). A function \( f : \{0, 1\}^* \rightarrow \{0, 1, \bot\} \) is called \( A \)-consistent if \( f(x) = \chi_A(x) \lor f(x) = \bot \), for all \( x \in \text{dom}(f) \). The instance complexity of \( x \) with respect to \( A \) in \( U \) is defined as

\[
i_{C_U}(x : A) = \min\{l(p) : \lambda z. U(p, z) \text{ is a total } A\text{-consistent function such that } U(p, x) = \chi_A(x)\}.
\]

If no such \( p \) exists then \( i_{C_U}(x : A) = \infty \).

If we drop in the definition of \( i_{C_U} \) the requirement that \( \lambda z. U(p, z) \) is total, then we obtain a weaker notion of instance complexity, which we denote by \( \overline{i_{C_U}}(x : A) \). Note that \( \overline{i_{C_U}}(x : A) \leq i_{C_U}(x : A) \) for all \( x, A \).
It is well-known (see [5]) that there exist “optimal” partial recursive functions \( U \) such that, for every partial recursive mapping \( U' \), there is a constant \( c \) with \( C_U(x) \leq C_{U'}(x) + c \), \( ic_U(x : A) \leq ic_{U'}(x : A) + c \), and \( \overline{ic}_U(x : A) \leq \overline{ic}_{U'}(x : A) + c \), for all \( x, A \).

For the following we fix an optimal mapping \( U \) and write \( C(x) \), \( ic(x : A) \), and \( \overline{ic}(x : A) \), for \( C_U(x) \), \( ic_U(x : A) \), and \( \overline{ic}_U(x : A) \), respectively. We also write \( U_\sigma(p, z) \) for the result, if any, after \( s \) steps of computation of \( U \) with input \((p, z)\). \( C^s(x) \) denotes the approximation to \( C(x) \) after \( s \) steps of computation (i.e., with \( U_\sigma \) in place of \( U \) in the definition of \( C(x) \)). Clearly, \( C^{s+1}(x) \leq C^s(x) \) and \( C^t(x) = C(x) \) for all sufficiently large \( t \).

The instance complexity of \( x \) can be bounded by the Kolmogorov complexity of \( x \) in the sense that for every set \( A \) there is a constant \( c \) such that \( ic(x : A) \leq C(x) + c \) for all \( x \). Informally, \( x \) is a hard instance of \( A \) if this upper bound is also a lower bound. This was the motivation for the following definition (which is independent of the choice of the optimal \( U \)).

**Definition 1.3** (Ko, Orponen, Schöning, Watanabe, 1986)

A set \( A \) has **hard instances** if there is a constant \( c \) such that
\[
ic(x : A) \geq C(x) - c \quad \text{for infinitely many } x.
\]

If the condition holds with \( \overline{ic} \) in place of \( ic \), we say that \( A \) has hard instances with respect to \( \overline{ic} \).

**Remark:** The difference between \( ic \) and \( \overline{ic} \) is perhaps best explained by an example:

Suppose that \( A \) is an r.e. set and we want to define a program \( p \) such that it witnesses \( ic(x : A) \leq |p| \) for all \( x \) with \( C(x) < n \). Since \( p \) has to be total we have to define it for every input \( z \) at some step \( s \). If \( z \) has already appeared in \( A \) there is no problem, we set \( U(p, z) = 1 \). If \( z \) has not yet appeared in \( A \) and \( C^s(z) \geq n \), we could try to define \( U(p, z) = \bot \), but this can later become incorrect if it turns out that \( C(z) < n \). If we set \( U(p, z) = 0 \) and \( z \) later appears in \( A \), then \( p \) is also incorrect.

In the case of \( \overline{ic} \) we have more freedom: We may leave \( U(p, z) \) undefined until \( C^s(z) < n \) at some stage \( s \). If this never happens, then \( U(p, z) \) is undefined and \( C(z) \geq n \), which is fine. Still the second source of error remains: If \( C^s(z) < n \) and \( z \) has not yet appeared in \( A \) at stage \( s \), we have to define \( U(p, z) \), and the best we can do is to set \( U(p, z) = 0 \). But this may later turn out to be incorrect.

## 2 A version of Barzdin’s Lemma

In this section we consider the Kolmogorov complexity of initial segments of r.e. sets. For \( A \subseteq \mathcal{N} \) and \( n \in \mathcal{N} \) we write \( \chi_A \upharpoonright n \) for the string \( \chi_A(0) \ldots \chi_A(n) \).

Let us first recall what was already known. The conditional complexity of a string \( \sigma \) of length \( n \) is defined as \( C(\sigma|n) = \min \{ l(p) : U(p, n) = \sigma \} \). We write \( C(\chi_A|n) \) for \( C(\chi_A \upharpoonright (n - 1)|n) \). Barzdin ([1], see [3, Theorem 2.18]) characterized the worst case of the conditional complexity for initial segments of r.e. sets:

- For every r.e. set \( A \) there is a constant \( c \) such that for all \( n \):
  \[
  C(\chi_A|n) \leq \log n + c.
  \]
There is an r.e. set $A$ such that $C(\chi_A|n) \geq \log n$ for all $n$.

Now we look at the standard Kolmogorov complexity $C(\chi_A|n)$. Utilizing a result of Meyer [4, p. 525], Chaitin proved that, if there is constant $c$ such that for all $n$, $C(\chi_A|n) \leq \log n + c$, then $A$ is recursive [3, Theorem 6], [5, Exercise 2.43].

For every r.e. set $A$ there is a constant $c$ such that $C(\chi_A|n) \leq 2\log n + c$ for all $n$ (see [3, Exercise 2.59]). On the other hand, there is no r.e. set $A$ such that $C(\chi_A|n) \geq 2\log n - O(1)$ for almost all $n$. This follows from the argument in [8, Exercise 2.58].

In [8, Exercise 2.59] it is stated as an open question (attributed to Solovay) whether the upper bound $2\log n$ is optimal. The following result shows that this is indeed the case. For ease of conversation, we say that $A$ is complex if there is a constant $c$ such that $C(\chi_A|n) \geq 2\log n - c$ for infinitely many $n \in \mathcal{N}$.

**Theorem 2.1** There is an r.e. complex set.

**Proof:** Let $t_0 = 0, t_{k+1} = 2^{t_k}$, and $I_k = (t_k, t_{k+1}]$, for all $k \geq 0$. $(I_k)$ is a sequence of exponentially increasing half-open intervals.

Let $f(k) = \sum_{i=t_k+1}^{t_{k+1}}(i - t_k + 1), g(k) = \max\{|l : 2^{|l|} - 1 < f(k)|\}$. Note that $f(k) = \frac{1}{2}t_{k+1}^2 - o(1)$ and $g(k) = 2\log t_{k+1} - 2 - o(1)$, for $k \to \infty$.

We enumerate an r.e. set $A$ in steps as follows:

**Step 0:** Let $A_0 = \emptyset$.

**Step $s + 1$:** Let $A_{s+1} = A_s$. For $k = 0, \ldots, s$ do: If $C^s(\chi_{A_s}|n) \leq g(k)$ for all $n \in I_k$ then enumerate $\min(A_s \cap I_k)$ into $A_{s+1}$.

Let $A = \cup_{s \geq 0} A_s$. Suppose for a contradiction that $C(\chi_A|n) \leq g(k)$ for all $n \in I_k$. Then we eventually enumerate every $n \in I_k$ into $A$. Note that for fixed $n$ there are at least $n - t_k + 1$ different strings $\sigma = \chi_{A_s}|n$ with $l(\sigma) = n + 1$ and $C(\sigma) \leq g(k)$. (The suffix of $\chi_{A_s}|n$ runs through $1^{x_0 n - k - x}$ for $x = 0, \ldots, n - t_k$.) Thus, there are at least $f(k)$ many different strings which all have Kolmogorov complexity at most $g(k)$. This contradicts the definition of $g(k)$.

So for every $k$ there exists $n \in I_k$ with $C(\chi_A|n) > g(k)$, i.e., $C(\chi_A|n) > g(k) \geq 2\log n - 2 - o(1)$. Thus, $A$ is complex.

We now characterize the degrees of r.e. complex sets. Downey, Jockusch, and Stob [4] introduced the notion of an array nonrecursive set. This captures precisely those r.e. sets that arise in multiple permitting arguments. In [4] several other natural characterizations of this degree class are given.

An r.e. set $A$ is called array nonrecursive with respect to $\{F_k\}_{k \in \mathcal{N}}$ if

$$(\forall e)(\exists \alpha k)[W_e \cap F_k = A \cap F_k].$$

Here $\{F_k\}_{k \in \mathcal{N}}$ denotes a very strong array. This means that $\{F_k\}_{k \in \mathcal{N}}$ is a strong array of pairwise disjoint sets which partition $\mathcal{N}$ and satisfy $|F_k| < |F_{k+1}|$ for all $k \in \mathcal{N}$.

An r.e. set is array nonrecursive if it is array nonrecursive with respect to some very strong array $\{F_k\}_{k \in \mathcal{N}}$. A degree is called array nonrecursive if it contains an r.e. array nonrecursive set. Not every r.e. nonrecursive degree is array nonrecursive [4, Theorem 2.10].
Theorem 2.2 The degrees containing an r.e. complex set coincide with the array nonrecursive degrees. In addition, if \( A \) is r.e. and not of array nonrecursive degree, then for every unbounded, nondecreasing, total recursive function \( f \) there is a constant \( c \) such that
\[
C(\chi_A \upharpoonright n) \leq \log n + f(n) + c \quad \text{for all } n \in \mathbb{N}.
\]

Proof: Note that, in order to make \( A \) complex, we only need to complete the construction from the previous theorem for infinitely many intervals. It follows that every r.e. set \( A \), that is array nonrecursive with respect to \( \{I_k\}_{k \in \mathbb{N}} \), is also complex.

In [4, Theorem 2.5] it is shown that every array nonrecursive degree contains such a set, i.e., it contains an r.e. complex set.

For the converse we use [4, Theorem 4.1]. It states that if \( A \) is r.e. and does not have array nonrecursive degree, then for every total function \( g \leq_T A \) there is a total recursive approximation \( g(x,s) \) such that \( \lim s g(x,s) = g(x) \) and \( |\{s : g(x,s) \neq g(x,s+1)\}| \leq x \), for all \( x \in \mathbb{N} \). Actually, in [4] this is only stated for 0/1-valued \( g \), but the proof provides the more general version.

Let \( A \) be r.e. and not of array nonrecursive degree. Assume we are given any total recursive, nondecreasing, unbounded function \( f \). Let \( m(x) = 1 + \max\{n : f(n) \leq x\}; m \) is total recursive. Let \( g(x) = \chi_A \upharpoonright m(x) \). Since \( g \) is recursive in \( A \), there is a total recursive approximation \( g(x,s) \) as above.

How can we describe \( \chi_A \upharpoonright n \)? Given \( n \) we compute \( n' = \min\{x : m(x) > n\} \). Then we simulate \( g(n',s) \) until it outputs \( g(n') \), which gives us \( \chi_A \upharpoonright n \). In order to perform the simulation we only need to know the exact number \( x \leq n' \) of mindchanges of \( g(n',s) \). Thus, \( \chi_A \upharpoonright n \) is specified by the pair \( \langle x,n' \rangle \) which can be encoded by a string of length \( \log n + 2 \log(x+1) + O(1) \). Since \( m(x-1) \leq n \) we have \( f(n) \geq x \), by the definition of \( m \). Thus, we get
\[
C(\chi_A \upharpoonright n) \leq \log n + 2 \log(x+1) + O(1) \leq \log n + f(n) + O(1).
\]

This completes the proof of the theorem.

Note that Theorem 2.2 entails the following curious gap phenomenon. For every r.e. degree \( \alpha \) there are only two cases:

1. There is an r.e. set \( A \in \alpha \) such that
   \[
   (\exists^\infty n)[C(\chi_A \upharpoonright n) \geq 2 \log n - O(1)].
   \]
2. There is no r.e. set \( A \in \alpha \) and \( \epsilon > 0 \) such that
   \[
   (\exists^\infty n)[C(\chi_A \upharpoonright n) \geq (1 + \epsilon) \log n - O(1)].
   \]

3 The Instance Complexity Conjecture fails

In this section we determine the least possible instance complexity of nonrecursive r.e. sets. Here it is convenient to take \( A \) as a subset of \( \{0,1\}^* \). Clearly, if \( A \) is recursive then \( ic(x : A) \) is bounded by a constant for all \( x \). The next result (another gap theorem) shows that, for infinitely many \( x \), \( ic(x : A) \) must be at least logarithmic in \( C(x) \) if \( A \) is nonrecursive.\footnote{This result was previously announced by Tromp [14].}
Theorem 3.1 If \( \text{ic}(x : A) \leq \log C(x) - 1 \) for almost all \( x \), then \( A \) is recursive.

Proof: Let \( P_k = \{0, 1\}^{\leq k} \) and let \( \mathcal{P}(P_k) \) denote the set of all subsets of \( P_k \). Uniformly in \( k \) we enumerate a finite set \( B_k \subseteq \{0, 1\}^* \).

Step 0: Let \( S_k = \mathcal{P}(P_k) \) and \( B_k = \emptyset \).

Step \( n + 1 \): Search via dovetailing for \( I \in S_k \), \( x \in \{0, 1\}^* \) and \( s \in \mathcal{N} \) such that \( U_s(p, x) = \perp \) for all \( p \in I \). If such an \( I \) is found, then enumerate \( x \) into \( B_k \), remove \( I \) from \( S_k \) and go to step \( n + 2 \). □

Note that \( B_k \) is nonempty, since \( I = \emptyset \) trivially satisfies the condition for all \( x, s \). Also, at most \( |\mathcal{P}(P_k)| = 2^{|P_k|} \) elements are enumerated into \( B_k \) and \( B_k \) is uniformly r.e. Thus, there is a partial recursive function \( \psi : \{0, 1\}^* \times \{0, 1\}^* \to \{0, 1\}^* \) such that \( \psi(\{0, 1\}^{|P_k|}, \lambda) = B_k \) for all \( k \). In particular, \( C_\psi(x) \leq |P_k| = 2^{k+1} - 1 \) for all \( x \in B_k \).

Choose a constant \( c \) such that \( C(x) \leq C_\psi(x) + c \) for all \( x \).

Let \( A \subseteq \{0, 1\}^* \) be given and suppose that \( \text{ic}(x : A) \leq \log C(x) - 1 \) for almost all \( x \). Then \( \text{ic}(x : A) \leq \log(C_\psi(x) + c) - 1 \), so \( \text{ic}(x : A) \leq \log C_\psi(x) \) for almost all \( x \). Since \( \lfloor \log C_\psi(x) \rfloor \leq \lfloor \log(2^{k+1} - 1) \rfloor = k \) for all \( x \in B_k \), we can choose \( k \) large enough such that for all \( x \in B_k \) we have \( \text{ic}(x : A) \leq k \).

Thus, for each \( x \in B_k \) there is \( p \in P_k \) such that \( \lambda z. U(p, z) \) is a total \( A \)-consistent function with \( U(p, x) = \chi_A(x) \). Let \( I_0 = \{p \in P_k : \lambda z. U(p, z) \) is a total \( A \)-consistent function\}. This set is nonempty since \( B_k \) is nonempty.

Now consider the construction of \( B_k \): Note that \( I_0 \) cannot be removed from \( S_k \). Otherwise there exists \( x \in B_k \) such that \( C_\psi(x) \leq 2^{k+1} - 1 \) and \( U(p, x) = \perp \) for all \( p \in I_0 \), i.e., \( \text{ic}(x : A) > k \), contradicting the choice of \( k \). Since \( I_0 \) is never removed from \( S_k \), it follows from the construction of \( B_k \) that for every \( x \) there is \( p \in I_0 \) with \( U(p, x) = \chi_A(x) \). Thus, if we amalgamate all of the functions \( U(p, -) \) with \( p \in I_0 \), we get a recursive characteristic function of \( A \), i.e., \( A \) is recursive. □

We prove that the lower bound of Theorem 3.1 is tight even for nonrecursive r.e. sets. This refutes the Instance Complexity Conjecture of Orponen et al. [11, 12, 8, Exercise 7.41], stating that every nonrecursive r.e. set has hard instances. In contrast, our result together with Theorem 3.1 shows that the true threshold between the instance complexity of recursive and nonrecursive sets is \( \log C(x) \) instead of \( C(x) \).

Theorem 3.2 There is a nonrecursive r.e. set \( A \) and a constant \( c \) such that \( \text{ic}(x : A) \leq \log C(x) + c \) for all \( x \).

Proof: It suffices to construct a nonrecursive r.e. set \( A \) and a partial recursive function \( \psi \) such that \( \text{ic}_\psi(x : A) \leq \log C(x) + 2 \) for almost all \( x \). In the following we write \( \psi_p \) for \( \lambda z. \psi(p, z) \).

Let \( E_k = \{x : C(x) < 2^k - 2\} \) for \( k \geq 1 \). We want to establish that \( \text{ic}_\psi(x : A) \leq k \) for all \( x \in E_k \). Let \( M_k = \{p_{k,1}, \ldots, p_{k,2^{k-2}}\} \) denote the set of the first \( 2^k - 2 \) strings of length \( k \). The idea is that every \( \psi_{p_{k,i}} \) is \( A \)-consistent and for each \( x \in E_k \) there is \( p \in M_k \) such that \( \psi_p \) witnesses that \( \text{ic}_\psi(x : A) \leq k \). There is, however, some difficulty to combine this with the requirement to make \( A \) nonrecursive.

The basic idea to satisfy the latter requirement is as follows: For each \( e \geq 1 \) we establish a unique diagonalization value \( d_e \), then we wait until \( d_e \) is enumerated into \( W_e \), if this ever happens we enumerate \( d_e \) into \( A \). Here \( \{W_e\}_{e \in \mathcal{N}} \) is the standard r.e.
listing of all r.e. sets of strings. Hence, this strategy makes sure that \( \overline{T} \) is not r.e., so
A is a nonrecursive r.e. set.

Suppose that \( d_e \) appears in \( E_k \) before it appears in \( W_e \). If we define \( \psi_{p_k,i}(d_e) = 0 \)
for some \( i \), then, since \( \psi_{p_k,i} \) should be \( A \)-consistent, we can no longer enumerate \( d_e \)
into \( A \). This threatens our diagonalization strategy. On the other hand, we certainly
should make sure that \( ic_{\psi}(d_e : A) \leq k \).

This conflict is solved by a finite-injury priority argument:

If \( e \geq k \) and we are forced to define \( \psi_{p_k,i}(d_e) = 0 \), then we assign a new much larger
value to \( d_e \) and try to diagonalize at this new value. Note that \( d_e \) is changed only
finitely often, because there are only finitely many values which may appear in \( E_k \)
for some \( k \leq e \). Thus, the value of \( d_e \) eventually stabilizes and the \( e \)-th diagonalization
strategy goes through with this final value.

If \( e < k \) then we do not use \( \psi_{p_k,i} \) to ensure that \( ic_{\psi}(d_e : A) \leq k \). Thus, we define
\( \psi_{p_k,i}(d_e) = \perp \), which certainly maintains the \( A \)-consistency. Instead we will have two
special programs \( \tau_{e,1}, \tau_{e,2} \) of length \( e \) (which are not in \( M_e \); this is the reason why we
have left out two strings) to witness that \( ic_{\psi}(d_e : A) \leq k \). More precisely, if the
final \( d_e \)-value is not enumerated into \( A \), then \( \psi_{\tau_{e,1}} \) will be the correct function. If the
final \( d_e \)-value is enumerated into \( A \), then \( \psi_{\tau_{e,2}} \) will not be \( A \)-consistent but \( \psi_{\tau_{e,2}} \)
is used as a back-up function.

It remains to explain how only \(|M_k|\) many programs can take care of all the elements
in \( E_k \), which may be up to \( 2^{1|M_k|} - 1 \) many. We show in an example how two programs
\( p_1, p_2 \) can take care of \( 3 = 2^2 - 1 \) elements (for simplicity, we drop the distinction
between numbers and strings): At the beginning \( \psi_{p_1}, \psi_{p_2} \) are undefined. Now in step
\( s_1 \) the first element \( x_1 < s_1 \) appears. We let \( \psi_{p_1}(x) = \chi_A(x) \) for all \( x \leq s_1 \). In the
following steps \( s \) we define \( \psi_{p_1}(s) = \perp \) until the second element \( x_2 \) appears, say at
step \( s_2 > x_2 \). If \( x_2 \leq s_1 \) we do nothing. If \( x_2 > s_1 \) then we define \( \psi_{p_2}(x) = \chi_A(x) \)
for all \( x \leq s_2 \) and in the following steps \( t \) we define \( \psi_{p_2}(t) = \perp \). The point is that
\( \psi_{p_2} \) also takes care of \( x_1 \), thus we suspend the definition of \( \psi_{p_1} \) until a third element
\( x_3 \) appears at step \( s_3 > x_3 \). If \( x_3 > s_2 \) then we resume the definition of \( \psi_{p_1} \) and let
\( \psi_{p_1}(x) = \chi_A(x) \) for all \( s_2 < x \leq s_3 \). For arguments \( t > s_3 \) we define both function
equal to \( \perp \). Note that now \( p_1 \) and \( p_2 \) together take care of \( x_1, x_2, x_3 \).

This idea is easily generalized: Let \( succ(\sigma) \) denote the lexicographical successor of
\( \sigma \), i.e., if \( \sigma = b_1 \ldots b_n \neq \perp \) then \( succ(\sigma) = 0^{i-1}1b_{i+1} \ldots b_n \) where \( i = \min\{j : b_j = 0\} \).
Then the programs \( p_{k,i} \in M_k \) with \( succ^{(m)}(0^{1|M_k|})(i) = 1 \) take care, if exactly \( m \)
are enumerated into \( E_k \). (In the implementation below we count only those elements which are not \( d_e \)-values for some \( e < k \).) Note that, since \( m \leq 2^{1|M_k|} - 1 \), \( succ \) is never applied to \( 1^{1|M_k|} \).

We now turn to the detailed implementation. First we fix some additional notation
and conventions. Let \( \langle \ldots \rangle \) denote a recursive pairing function which is increasing
in its second argument. We assume that elements of \( E_k \) are enumerated in steps such
that in each step at most one new element is enumerated; also if \( x \) is enumerated in
step \( s \) then \( l(x) < s \). \( W_e,s \) is the finite set of strings which are enumerated into \( W_e \) in
at most \( s \) steps of computation.

In the construction the variables \( e, i, j, k, n, s, t \) denote numbers, and \( p, x, z \) denote
strings. In addition, the following variables are used: \( \psi_{p,s} \) the finite portion of \( \psi_p \)
constructed up to stage \( s \); the \( i \)-th bit of \( \sigma_{k,s} \in \{0,1\}^{1|M_k|} \) tells us whether \( \psi_{p_k,i} \) is
currently assigned to take care of the elements in $E_k$; $len(k, s)$ is the greatest length $n$ such that our set-up at stage $s$ guarantees that $ic_p(x : A) \leq \log C^s(x) + 2$ for all $x \in E_{k,s}$ with $l(x) < n$; $d_e(s)$ is the current value of the $e$-th diagonalization point. We call $e$ “active” as long as no $d_e$-value has been enumerated into $A$, otherwise we call $e$ “passive”. So, if $e$ is “passive”, then we know that we have explicitly satisfied the $e$-th diagonalization requirement. $A_s$ denotes the finite set of elements which have been enumerated into $A$ up to stage $s$.

Let $R(k, s) = \{d_e(s') : e < k \land s' \leq s\}$. As explained above, the programs in $M_k$ do not need to take care of the elements in $R(k, s)$.

If one of the variables $v(s)$ is not explicitly changed at stage $s + 1$, then we assume without further mentioning that $v(s + 1) = v(s)$.

We first describe the construction of $\psi_p$ for $p \in M_k$, $k \geq 1$. Then we define $\psi_p$ for the two special values $p = \tau_{e,1}, \tau_{e,2}$ of each length $e$.

Construction:

Stage 0: Let $\psi_p = \lambda x. \uparrow$, for all $p \in \{0, 1\}^*$. For all $k \geq 1$: $\sigma_{k,0} = 0|M_k|; len(k, 0) = 0$; $d_k(0) = 0(k,0)$; declare $k$ as “active”. Let $A_0 = \emptyset$.

Stage $s + 1$:
Case I: $s$ is even.
For $e = 0, \ldots, s$: If $e$ is active and $d_e(s) \in W_{e,s} - A_s$, then enumerate $d_e(s)$ into $A$ and declare $e$ “passive”.
Case II: $s$ is odd, $s = 2(k, t) + 1$.
Let $\psi_{p_{k,i}}(x) = \bot$, for all $i$ with $\sigma_{k,s}(i) = 1$ and all $x$ with $l(x) = t$.
If a new element $x, l(x) < t$, enters $E_k$ after exactly $t$ steps, then act according to the following cases:

a.) If $x \in R(k, s)$ or $l(x) < len(k, s)$ then go to stage $s + 2$.
b.) Otherwise do the following:
   Let $\sigma_{k,s+1} = succ(\sigma_{k,s})$ and $i = \min\{j : \sigma_{k,s+1}(j) = 1\}$. (At at most $2|M_k| - 1$ elements are enumerated in $E_k$, so we get $\sigma_{k,s+1} \in \{0,1\}^{|M_k|} - \{0^{|M_k|}\}$.)
   Let $n = \min\{l(z) : z \notin dom(\psi_{p_{k,i}})\}$.
   Define $\psi_{p_{k,i}}(z) = \chi_{A_1}(z)$, for all $z \notin R(k, s)$ such that $n \leq l(z) \leq t$.
   Let $\psi_{p_{k,i}}(z) = \bot$, for all $z \notin R(k, s)$ such that $n \leq l(z) < t$.
   Let $len(k, s + 1) = t + 1$. For all active $e \geq k$, let $d_e(s + 1) = 0(e,s+1)$.
   Go to stage $s + 2$. □

For each $e \geq 1$ we define

$$
\psi_{\tau_{e,1}}(x) = \begin{cases} 
0 & \text{if } x \in \text{range}(d_e); \\
\bot & \text{otherwise.}
\end{cases}
$$

If $e$ is active at all stages then let $\psi_{\tau_{e,2}} = \lambda x. \uparrow$. Otherwise let $s_e$ be the (unique) stage where $e$ is declared “passive” and let

$$
\psi_{\tau_{e,2}}(x) = \begin{cases} 
0 & \text{if } (\exists t < s_e)[x = d_e(t) \neq d_e(s_e)]; \\
1 & x = d_e(s_e); \\
\bot & \text{otherwise.}
\end{cases}
$$

End of Construction.
**Verification:**
Most of the following claims are standard. The crucial one is Claim 3, b.), c.).

**Claim 1** For all $e \geq 1$:

a.) $l(d_e)$ is nondecreasing, and for all $s$: If $d_e(s) \neq d_e(s+1)$ then $l(d_e(s+1)) > s$.

b.) $\text{range}(d_e)$ is a uniformly recursive finite set; $\text{range}(d_e) \cap \text{range}(d_{e'}) = \emptyset$ for all $e' \neq e$.

c.) If $A \cap \text{range}(d_e)$ contains an element $x$ then $x = \lim_{s \to \infty} d_e(s)$.

d.) For all $x, s$: If $l(x) \leq s$ and, for all $e$, $x \neq d_e(s)$, then $x \neq d_e(s')$ for all $e$ and all $s' \geq s$.

e.) $A$ is r.e. and nonrecursive.

**Proof:** a.) If $d_e(s) \neq d_e(s+1)$ then for some $s' \leq s$: $l(d_e(s)) = (e, s') < (e, s+1) = l(d_e(s+1))$. Note that $(e, s+1) > s$, since $(\cdot, \cdot)$ is monotone in the second argument.

b.) It follows from a.) that $\text{range}(d_e)$ is uniformly recursive. It is a finite set, because $d_e(s)$ changes only if a new element is enumerated in some set $E_k$, $k < e$ which happens only finitely often. So $\lim_{s \to \infty} d_e(s)$ exists and is finite. $\text{range}(d_e)$ and $\text{range}(d_{e'})$ are disjoint for $e \neq e'$, since $(\cdot, \cdot)$ is injective.

c.) If $d_e(s)$ is enumerated into $A$ at stage $s+1$, then $e$ is declared “passive”, so $d_e(s)$ is fixed at all later stages.

d.) This follows from a.).

e.) Clearly $A$ is r.e. Suppose for a contradiction that $A$ is recursive. Then there exists $e$ with $\overline{A} = W_e$. By a.), b.), there is a stage $s$ such that $d_e(s') = d_e(s)$ for all $s' \geq s$. By construction, $d_e(s)$ is enumerated into $A$ iff it is enumerated into $W_e$. This contradicts the hypothesis $\overline{A} = W_e$. □

**Claim 2** For all $e \geq 1$:

a.) $\psi_{\tau,e,1}, \psi_{\tau,e,2}$ are uniformly partial recursive.

b.) If $e$ is always “active” then $\psi_{\tau,e,1}$ witnesses that $i\psi(x : A) \leq e$ for all $x \in \text{range}(d_e)$.

c.) If $e$ is eventually “passive” then $\psi_{\tau,e,2}$ witnesses that $i\psi(x : A) \leq e$ for all $x \in \text{range}(d_e)$.

**Proof:** a.) This follows from Claim 1, b.)

b.) If $e$ is always “active” then $\text{range}(d_e) \cap A = \emptyset$, thus $\psi_{\tau,e,1}$ is $A$-consistent and $\psi_{\tau,e,1}(x) = 0 = \chi_A(x)$ for all $x \in \text{range}(d_e)$.

c.) If $e$ is declared “passive” at stage $s+1$ then $A \cap \text{range}(d_e) = \{d_e(s)\}$. Thus $\psi_{\tau,e,2}$ is $A$-consistent and $\psi_{\tau,e,2}(x) = \chi_A(x)$ for all $x \in \text{range}(d_e)$. □

Let $\psi^s_e$ denote the finite portion of $\psi_e$ defined at the end of stage $s$.

**Claim 3** For all $s = 2(k, t) + 1$:

a.) For all $i$, $1 \leq i \leq |M_k|$: $\psi^s_{p,k,i}$ is an $A$-consistent function.

b.) For all $i$, $1 \leq i \leq |M_k|$: If $\sigma_{k,s+1}(i) = 1$ then $\text{dom}(\psi^{s+1}_{p,k,i}) = \{x : l(x) \leq t\}$.

c.) For all $x$, $l(x) < \text{len}(k, s+1)$: If $x \notin R(k, s)$ then there exists $i$, $1 \leq i \leq |M_k|$, with $\sigma_{k,s+1}(i) = 1$ and $\psi^{s+1}_{p,k,i}(x) = \chi_{A_{s+1}}(x)$. 

### Kolmogorov Complexity and Instance Complexity

9
Proof: a.) We use Claim 1, d.) and the fact that \( \psi_{p_k,i} \) is defined at stage \( s + 1 \) only for arguments less than \( s \). If \( e < k \) and \( \psi_{p_k,i}(d_e(s')) \) is defined then \( \psi_{p_k,i}(d_e(s')) = \bot \), so there is no problem with consistency. If \( e \geq k \) and \( \psi_{p_k,i}(d_e(s')) = \chi_{A_{e+1}}(d_e(s')) \) is defined at stage \( s + 1 > s' \), then either \( e \) is already “active”, so \( \chi_{A_{e+1}}(d_e(s')) = \chi_A(d_e(s')) \), or \( e \) is “active” and we define \( d_e(s + 1) \) at stage \( s + 1 \) such that \( l(d_e(s + 1)) > l(d_e(s')) \). In the latter case we get \( \psi_{p_k,i}(d_e(s')) = \chi_{A_{e+1}}(d_e(s')) = \chi_A(d_e(s')) = 0 \).

b.) and c.) are shown by induction on \( s \). Consider stage \( s + 1 = 2(k, t) + 2 \). If no new element is enumerated in \( E_k \) after exactly \( t \) steps then \( \sigma_{k,s+1} = \sigma_{k,s} \) and b.), c.) follow from the induction hypothesis and the definition of \( \psi_{p_k,i} \) at stage \( s + 1 \).

Now assume that \( x \) enters \( E_k \) after exactly \( t \) steps. If case a.) occurs, the claim follows from the induction hypothesis. If case b.) occurs, we have \( x \notin R(k, s) \) and \( \text{len}(k, s) \leq l(x) < t \). We have \( \sigma_{k,s+1} = \text{succ}(\sigma_{k,s}) \), so \( \sigma_{k,s+1}(i') = \sigma_{k,s}(i') \) for all \( i' > i = \min\{j : \sigma_{k,s+1}(j) = 1\} \).

If \( \sigma_{k,s+1}(i') = 0 \) for all \( i' > i \) then \( s + 1 \) is the first stage \( s' \) where \( \sigma_{k,s'}(i) = 1 \). This means that \( \psi_{p_k,i}^s = \lambda x. \uparrow \) and \( \psi_{p_k,i}^{s+1}(z) = \chi_A(z) = \chi_{A_{s+1}}(z) \), for all \( z \) such that \( l(z) \leq t \) and \( z \notin R(k, s) \).

If there is \( i' > i \) with \( \sigma_{k,s+1}(i') = 1 \), then there exists a greatest stage \( s' < s \) with \( \sigma_{k,s'}(i) = 1 \land \sigma_{k,s'+1}(i) = 0 \). Then we have \( \sigma_{k,s'+1}(i') = \sigma_{k,s'+1}(i') \) for all \( i' > i \) and we have \( \sigma_{k,s'+1}(i') = \sigma_{k,s+1}(i') = 0 \) for all \( i' < i \). By induction hypothesis, we get for all \( x \) with \( l(x) < \text{len}(k, s' + 1) \): If \( x \notin R(k, s') \) then there exists \( j \) with \( \sigma_{k,s'+1}(j) = 1 \) and \( \psi_{p_k,i}^{s+1}(x) = \chi_{A_{s+1}}(x) = \chi_A(x) \) (the second equality holds by part a.)).

Since \( R(k, s') \subseteq R(k, s) \), it only remains to consider \( x \) with \( \text{len}(k, s' + 1) \leq l(x) < \text{len}(k, s + 1) \). As \( \sigma_{k,s'}(i) = 1 \) it follows, by induction hypothesis, that \( \text{dom}(\psi_{p_k,i}^s) = \text{dom}(\psi_{p_k,i}^{s'}) = \{x : l(x) < \text{len}(k, s' + 1)\} \). Thus, \( n = \min\{l(z) : z \notin \text{dom}(\psi_{p_k,i}^s)\} = \text{len}(k, s' + 1) \) and at stage \( s + 1 \) we define \( \psi_{p_k,i}^{s+1}(z) = \chi_A(z) \), for all \( z \notin R(k, s) \) such that \( \text{len}(k, s' + 1) = n \leq l(z) < t + 1 = \text{len}(k, s + 1) \). For \( z \in R(k, s) \) and \( l(z) \leq t \) we have \( \psi_{p_k,i}^{s+1}(z) = \bot \). This completes the proof of b.), c.). □

Claim 4 For almost all \( x \): \( \text{ic}_\psi(x : A) \leq \log C(x) + 2 \).

Proof: Let \( k \geq 1 \) be minimal such that \( x \in E_k \). If \( x \in R(k, s) \) for some \( s \) then, by Claim 2, we get \( \text{ic}_\psi(x : A) < k \). If \( x \notin R(k, s) \) for all \( s \) then let \( \sigma_k = \lim_{s \to \infty} \sigma_{k,s} \). By Claim 3, there exists \( i, 1 \leq i \leq |M_k| \) such that \( \sigma_k(i) = 1 \land \psi_{p_k,i}(x) = \chi_A(x) \). Furthermore, \( \psi_{p_k,i} \) is total recursive and \( A \)-consistent, so \( \text{ic}_\psi(x : A) \leq k \). Since \( E_1 = \emptyset \), we have \( k > 1 \) and \( x \notin E_{k-1} \), so \( 2^{k-1} - 2 \leq C(x) \), i.e., \( k \leq \log(C(x) + 2) + 1 \leq \log C(x) + 2 \) for all \( x \) with \( C(x) \geq 2 \). □

What happens for \( \overline{\text{ic}} \)? Of course, the instance complexity conjecture also fails for \( \overline{\text{ic}} \). It even fails in a much stronger way, because, in contrast to Theorem 3.3, \( \overline{\text{ic}} \) can be arbitrary small, as we now show.

**Theorem 3.3** For every recursive function \( f \) there is an r.e. nonrecursive set \( A \) such that \( f(\overline{\text{ic}}(x : A)) \leq C(x) \) for almost all \( x \).

**Proof sketch:** We may assume that \( f \) is strictly increasing. As above it suffices to define a partial recursive function \( \psi(p, x) \) such that \( f(\overline{\text{ic}_\psi(x : A)}) \leq C(x) \) for almost all \( x \) and \( A \) is nonrecursive. This leads to the following requirements for all \( i \geq 1 \):
\[(N_i) \ (\forall x) [C(x) < f(i + 1) \Rightarrow (\exists p \in \{0, 1\}^i)[\chi_A \text{ extends } \psi_p \text{ and } \psi_p(x) = \chi_A(x)]].\]

\[(P_i) \ W_i \neq \overline{A}.\]

These can be satisfied by an easy finite-injury construction. Fix an enumeration of \(E_i = \{x : C(x) < f(i + 1)\}\) for all \(i\).

During the construction we have for each \(i\) a current \(p_i \in \{0, 1\}^i\) which satisfies \((N_i)\) for all \(x\) that have been currently enumerated into \(A\). If some \(x\) with \(\psi_{p_i}(x) = 0\) is later enumerated into \(A\), then \(\psi_{p_i}\) is no longer \(A\)-consistent and we have to choose a new \(p_i\). Since we have \(2^i\) candidates for \(p_i\), we can afford \(2^i - 1\) injuries.

Therefore, we allow to enumerate a diagonalization witness \(x\) into \(A\) at stage \(s\) for the sake of \((P_i)\), only if \(x\) has not yet appeared in any \(E_j\) with \(j \leq i\). Clearly, \((P_i)\) can still be satisfied. Furthermore, \((N_i)\) is injured at most \(i\) times. Since \(i \leq 2^i - 1\) for all \(i \geq 1\), every \((N_i)\) will be eventually satisfied. \(\blacksquare\)

**Remark:** In the course of the construction at most \(2^{f(i+1)} - 1\) elements are not allowed to be enumerated into \(A\) by \((P_i)\). Hence, we can fix in advance a set \(J_i\) of \(2^{f(i+1)}\) witnesses for \((P_i)\) and guarantee that one of them will be successful. Therefore, we can also modify the construction and satisfy the following requirements \((P'_i)\) instead of \((P_i)\) for any fixed r.e. set \(B\)

\[(P'_i) \ i \in B \Leftrightarrow J_i \cap A \neq \emptyset.\]

Then we get \(B \leq_d A\). If we choose \(B = K\), this shows that there is a \(d\)-complete set which satisfies the condition of the theorem. Since we need to enumerate at most one element of \(J_i\) into \(A\), we get that \(A \leq_{wtt(1)} B\). Thus, every r.e. \(wtt\)-degree contains a set \(A\) as in the theorem. It can be shown that this does not hold for r.e. \(tt\)-degrees.

## 4 R.e. sets having hard instances

While we have shown in the last section that ICC fails for some nonrecursive r.e. sets, it is interesting to find out whether there are properties of r.e. sets which imply the existence of hard instances. We consider this question for classes of complete sets and of simple sets. Indeed, in most cases it turns out that such sets must have hard instances, which is a partial resurrection of ICC.

Buhrman and Orponen [2, 5, Exercise 7.40] proved that the set of all random strings \(R = \{x : C(x) \geq l(x)\}\) satisfies \(ic(x : R) \geq l(x) - O(1)\) for all \(x \in R\). (Actually, their result also holds for \(ic\) instead of \(ic\).) Using the observation

\[(*) \text{ If } A \leq_m B \text{ via } f, \text{ then } ic(x : A) \leq ic(f(x) : B) + O(1) \text{ for all } x.\]

and the fact that \(R\) is co-r.e., they conclude that every m-complete set \(A\) has hard instances in its complement. They asked whether the hard instances can be chosen from \(A\) instead of \(\overline{A}\). (This is of course impossible in the \(ic\)-version.) The next result gives a positive answer.

**Theorem 4.1** There is an r.e. set \(A\) with \(ic(x : A) \geq l(x)\) for infinitely many \(x \in A\).
**Proof:** Uniformly in \( n \) we enumerate \( A \cap \{0,1\}^n \) as follows: Let \( x_1, \ldots, x_{2^n} \) be a listing of all strings of length \( n \) in lexicographical order.

**Step 0:** Enumerate \( x_1 \) into \( A \), let \( i = 1 \), \( I = \{0,1\}^{\leq n-1} \), \( J = \{1, \ldots, 2^n\} \). **Step \( s + 1 \):** If there is a string \( p \in I \) such that

(a) \( U_s(p, x_j) \in \{0,1\} \) for some \( j \in J \), or

(b) \( U_s(p, x_j) = \bot \) for all \( j \in J \),

then choose the least such \( p \), let \( I = I - \{p\} \), and do the following:

In case (a): Enumerate \( x_j \) into \( A \) iff \( U_s(p, x_j) = 0 \). Let \( J = J - \{j\} \).

In case (b): Let \( i = \text{min}(J) \). Enumerate \( x_i \) into \( A \) and let \( J = J - \{i\} \). \( \square \)

At the end of Step 0 we have \(|I| = |J| = 2^n - 1\). In all later steps an element of \( I \) is removed iff an element of \( J \) is removed. Thus, at the end of each step we have \(|I| = |J| \). Also, if case (b) occurs then \( \text{min}(J) \) exists (since at that point \(|J| > 0\)). Note that the value of \( \chi_A(x_j) \) is fixed when \( j \) is removed from \( J \).

Let \( i_0, I_0, J_0 \) be the final values of \( i, I, J \) in the above construction and choose \( s_0 \) such that \( i = i_0, I = I_0, J = J_0 \) in all steps \( t \geq s_0 \). Suppose for a contradiction that \( \text{ic}(x_{i_0} : A) < n \) via \( p \in \{0,1\}^{\leq n-1} \).

If \( p \not\in I_0 \) then there is a stage \( s \leq s_0 \) when \( p \) was removed from \( I \). If \( p \) was removed in case (a) via \( j \), then \( U(p, x_j) \neq \chi_A(x_j) \). If \( p \) was removed in case (b) then \( U(p, x_{i_0}) = \bot \). Hence, \( p \) does not witness that \( \text{ic}(x_{i_0} : A) < n \), a contradiction.

If \( p \in I_0 \) then \( |J_0| = |I_0| \geq 1 \) and there is \( t > s_0 \) such that \( U_t(p, x) \in \{0,1,\bot\} \) for all \( x \in J_0 \). Hence, at stage \( t+1 \) either case (a) or case (b) occurs and \( |I_0| \) decreases, contradicting the choice of \( s_0 \).

Thus, we have \( \text{ic}(x_{i_0} : A) \geq n = l(x_{i_0}) \) and clearly \( x_{i_0} \in A \). Since this holds for all \( n \), the theorem is proved. \( \blacksquare \)

Using (*) we get the following corollary.

**Corollary 4.2** For every \( m \)-complete set \( A \) there is a constant \( c \) such that

\[
\text{ic}(x : A) \geq C(x) - c \text{ for infinitely many } x \in A.
\]

This result also holds for a much weaker reducibility, as we now show.

**Theorem 4.3** For every wtt-complete set \( A \) there is a constant \( c \) such that

\[
\text{ic}(x : A) \geq C(x) - c \text{ for infinitely many } x \in A.
\]

**Proof:** Suppose that \( A \) is a wtt-complete set. We enumerate an auxiliary r.e. set \( B \) and a uniformly r.e. sequence \( \{E_n\}_{n \in \mathbb{N}} \) with \(|E_n| \leq 2^n \). Then there is a partial recursive function \( \psi : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N} \) such that \( \psi(\{0,1\}^n, \lambda) = E_n \). Hence, \( C_\psi(x) \leq n \) for all \( x \in E_n \) and there is a constant \( c \), independent of \( n \), such that \( C(x) \leq n + c \) for all \( x \in E_n \). Thus, it suffices to satisfy the following requirement for all \( n \)

\[
(R_n) \quad (\exists x \in E_n \cap A)[\text{ic}(x : A) \geq n - 1].
\]

By the recursion theorem and the fact that \( A \) is wtt-complete, we can assume that we are given in advance the index of a wtt-reduction from \( B \) to \( A \), i.e., a Turing reduction.
Φ and a total recursive use-bound g such that, for all x, χ_B(x) = Φ^A(x) and in the computation of Φ^A(x) every query is less than g(x).

Each (R_n) is satisfied independently from the other requirements; so for the following fix n and let x_1 = ⟨n, 1⟩, ..., x_2^n = ⟨n, 2^n⟩, m = max{g(x_i) : 1 ≤ i ≤ 2^n}, and I = {p : l(p) < n − 1}. We enumerate E_n and B \cap \{x_1, ..., x_{2^n}\} in steps i = 0, ..., 2^n as follows:

Step 0: Let s_0 = 0, E_n = ∅.
Step i + 1: Search for the least s ≥ s_i such that

1. Φ^A_s(x_j) = 1 with use less than g(x_j) for j = 1, ..., i and Φ^A_s(x_j) = 0 with use less than g(x_j) for j = i + 1, ..., 2^n.

2. For each x ∈ E_n there is p ∈ I such that

   (2.1) U_s(p, z) is defined for all z ≤ m.
   (2.2) (∀z ≤ m)[U_s(p, z) ≠ ⊥ ⇒ U_s(p, z) = χ_{A_s}(z)].
   (2.3) U_s(p, x) = 1.

Let s_{i+1} = s. Enumerate x_{i+1} into B and compute some x ≤ m with x ∈ A − A_{s_{i+1}}. (Note that x exists because otherwise Φ^A(x_{i+1}) = 0 ≠ 1 = χ_B(x_{i+1}). We can find x by enumerating A.) Let CONS be the set of all p ∈ I which satisfy conditions (2.1) and (2.2) for s = s_{i+1}. If U_s(p, x) = ⊥ for all p ∈ CONS, then enumerate x into E_n. Goto step i + 2. □

By construction, we have E_n ⊆ A. We want to argue that in some step of the construction the search does not terminate. Since χ_B(x) = Φ^A(x), this can only happen if condition (2) is not satisfied for any sufficiently large s. But this means that ic(x : A) ≥ n − 1 for some x ∈ E_n.

Consider the value of CONS ⊆ I after each terminating step: We show that a new element enters CONS or an element is removed forever from CONS. Since there are at most |I| < 2^{n-1} strings which may at some point become a member of CONS, it follows that there are less than 2 : 2^{n-1} = 2^n terminating steps, which completes the proof.

Note that if a string p is removed from CONS at some stage s, then there is no x such that U_s(p, x) = 0 and χ_{A_s}(x) = 1. Thus, x cannot enter CONS again at any later stage.

Suppose that step i + 1 terminates and consider the current value of CONS and of x at the end of this step. There are two cases:

(a) U_s(p, x) = ⊥ for all p ∈ CONS. Then x is enumerated into E_n, so in the next step a new string must enter CONS such that condition (2.3) is satisfied for x.

(b) U_s(p, x) ≠ ⊥ for all p ∈ CONS. Hence, U_s(p, x) = 0 and, since χ_{A_{s+2}}(x) = 1, p is removed from CONS if the next step terminates. □

By a similar proof, one can show that every btt-complete set has hard instances w.r.t. ic. We have noticed in the remark following Theorem 3.3 that this is no longer true for d-complete sets. But we can show that it still holds for Q-complete sets.
Recall that $A$ is Q-complete if it is r.e. and there is a recursive function $g$ such that for all $x$:
\[ x \in K \iff W_{g(x)} \subseteq A. \]
See [10, p. 281 f.] for more information on Q-reducibility.

**Theorem 4.4** Every Q-complete set $A$ has hard instances, even w.r.t. $\overline{ic}$.

**Proof:** Suppose that $A$ is Q-complete. As in the previous proof we enumerate an auxiliary r.e. set $B$ and an r.e. sequence of finite sets $\{E_n\}_{n \in \mathbb{N}}$ such that $|E_n| \leq 2^n$. It suffices to get infinitely many $n$ such that there is $y \in E_n$ with $\overline{ic}(y : A) \geq n - 2$.

By the recursion theorem and the Q-completeness of $A$, we may assume that we are given in advance a recursive function $g$ such that $B \subseteq Q A$ via $g$, i.e., for all $x$, $x \in B \iff W_{g(x)} \subseteq A$.

The first idea is to run a version of the previous construction: We keep a number $x$ out of $B$ and find $y \in W_{g(x)}$ which has not yet been enumerated into $A$. Then we enumerate $y$ into $E_n$ and wait until some $A$-consistent program $p$ with $l(p) < n - 2$ shows up and $U(p,y) = 0$. Then we enumerate $x$ into $B$, which forces $y$ into $A$ and diagonalizes $p$.

However, this approach does not work, because it might happen that after we enumerate $y$ into $E_n$, $y$ is also enumerated into $A$, and after that $U(p,y) = 1$ is defined. Then we cannot diagonalize $p$ by enumerating $x$ into $B$, but we have incremented $|E_n|$. Since this can happen an arbitrary finite number of times, we run into conflict with the requirement $|E_n| \leq 2^n$.

Therefore, we use the following modification: For each $n$, if $E_n \neq \emptyset$ then we enumerate $y$ into $E_n$ only if $y$ has been previously enumerated into $E_{n+1}$, and then we proceed according to the first idea. If later $y$ is enumerated into $A$ we get a diagonalization for $n + 1$ instead of $n$, which is also fine.

Now we turn to the formal details: Let $I_n = \{p : l(p) < n - 2\}$. $p \in \{0,1\}^*$ is called $A$-consistent at stage $s + 1$ if, for all $z \leq s$, either $U_s(p,z)$ is undefined or $U_s(p,z) = \chi_{A_s}(z)$.

We maintain the following invariant for all $n,s,y$:
If $E_n \neq \emptyset$ at stage $s + 1$ then enumerate $y$ into $E_n$ only if $P(n,s,y)$ holds, where:
\[
P(n,s,y) \iff y \in E_{n+1} - A_s, E_n \subseteq A_s, \text{ and there is } p \in I_{n+1} \text{ which is } A\text{-consistent at stage } s + 1 \text{ and } U_s(p,y) = 0.\]

As a consequence of this invariant it already follows that $|E_n| \leq 2^n$: Suppose that $E_n \neq \emptyset$ and we enumerate $y$ into $E_n$ at stage $s + 1$. Then we enumerate the next element into $E_n$ only after $y$ has been enumerated into $A$, and hence the program $p \in I_{n+1}$ which had witnessed the condition $P(n,s,y)$ is diagonalized and can never be $A$-consistent again. Since $|I_{n+1}| < 2^{n-1}$, it follows that we will enumerate at most $1 + 2^{n-1}$ programs into $E_n$. In particular, $|E_n| \leq 2^n$ for all $n$.

We say that $n$ is saturated at stage $s + 1$ if, for every $y \in E_n$, there is $p \in I_n$ such that $p$ is $A$-consistent at stage $s + 1$ and $U_s(p,y) = \chi_{A_s}(y)$. The goal of the construction is to produce infinitely many $n$ which are almost always not saturated. This implies at once that there are infinitely many $y \in E_n$ with $\overline{ic}(y : A) \geq n - 2$, and we are done.
To achieve this goal we construct a sequence \(d_0 < d_1 < d_2 < \cdots\) and satisfy the following requirements

\((R_i)\) The interval \([d_i, d_{i+1})\) contains an \(n\) which is almost always not saturated.

The \(d_i\)'s are constructed by recursive approximation: The value of \(d_i\) may change finitely often and eventually stabilizes. Some additional variables are needed for book-keeping: For each \(i\) there is a finite set \(T_i\) containing the set of all \(x\) which may be enumerated into \(B\) for the sake of \((R_i)\). For each \(n\) we have three variables \(\text{active}(n), \text{cand}(n), \text{source}(n)\). \(\text{active}(n)\) is a Boolean flag which indicates if there is some \(y \in E_n - A_s\) to be enumerated into \(E_{n-1}\); in this case \(\text{cand}(n) = y\) and \(\text{source}(n) = x\) such that \(x \notin B_s\) and \(y \in W_{g(x),s}\).

We say that \(i\) requires attention at stage \(s + 1\) if one of the following conditions holds at the beginning of stage \(s + 1\).

1. \(d_{i+1}\) is undefined.

2. \(d_{i+1}\) is defined and every \(n \in [d_i, d_{i+1})\) is saturated at stage \(s + 1\).

Construction:

Stage 0: Let \(d_0 = 0, d_{i+1} = \uparrow, T_i = \emptyset\) for all \(i\). Let \(\text{active}(n) = 0, E_n = \emptyset\) for all \(n\).

Stage \(s + 1\): For every \(n\) such that \(\text{active}(n) = 1\) and \(\text{cand}(n) \in A_s\) let \(\text{active}(n) = 0\). Let \(i\) be the least number which requires attention at stage \(s + 1\). If it requires attention through (1) then let \(d_{i+1} = s + 1\).

If it requires attention through (2) then we distinguish two cases:

(a) If there is a least \(n \in (d_i, d_{i+1})\) such that \(\text{active}(n) = 1\) and \(E_{n-1} \subseteq A_s\), then enumerate \(\text{cand}(n)\) into \(E_{n-1}\) and let \(\text{active}(n) = 0\). If \(n - 1 = d_i\) then enumerate \(\text{source}(n)\) into \(B\), else let \(\text{active}(n-1) = 1\), \(\text{cand}(n-1) = \text{cand}(n)\), and \(\text{source}(n-1) = \text{source}(n)\).

(b) Otherwise put \(s+1\) into \(T_i\) and let \(x = \min(T_i - B_s)\). Find the least \(s'\) such that \(W_{g(x),s'} - A_s \neq \emptyset\) and let \(y = \min(W_{g(x),s'} - A_s)\). Let \(\text{active}(s+1) = 1\), \(\text{cand}(s+1) = y\), \(\text{source}(s+1) = x\), and enumerate \(y\) into \(E_{s+1}\).

In both cases let \(T_i = T_i \cup \bigcup_{j > i} T_j\) and let \(T_j = \emptyset, d_j = \uparrow\), for all \(j > i\).

End of Construction.

It easily follows by induction on \(s\) that our invariant is satisfied: Note that before we enumerate a new number into \(E_{n-1}\) via step (a), we require that \(E_{n-1} \subseteq A_s\). If we enumerate a number via step (b) then the corresponding set was previously empty. Therefore, at each stage \(s + 1\) every \(E_n\) contains at most one number which is not in \(A_s\). Now suppose that \(E_{n-1} \neq \emptyset\) at the end of stage \(s\) and we enumerate a number \(y\) into \(E_{n-1}\) at stage \(s + 1\). Then case (a) occurred and \(y = \text{cand}(n) \notin A_s\) (since \(\text{active}(n) = 1\)). By the previous remarks, we have \(E_{n-1} \subseteq A_s\). Since \(n\) is saturated at stage \(s + 1\), there is an \(A\)-consistent \(p \in I_n\) such that \(U_p(p, y) = \chi_{A_s}(y) = 0\). Thus, \(P(n-1, s, y)\) holds.

Hence, it only remains to verify that requirement \((R_i)\) is satisfied for all \(i\). This is done by induction on \(i\). By induction hypothesis, there is a least stage \(s_0\) such that \(d_i = s_0\) is defined at stage \(s_0\) and no \(i' < i\) requires attention at any stage \(s > s_0\). At the end of stage \(s_0\) we have \(E_{d_i} = \emptyset\) and \(T_i = \emptyset\). We have shown above that the
cardinality of $E_{d_i}$ is always bounded by $2^{d_i}$. Hence, there exists $s_1 \geq s_0$ such that $E_{d_i}$ does not change after stage $s_1$. Note that $E_{d_i} \subseteq A$, because each time when we enumerate $y$ into $E_{d_i}$, we enumerate some $x$ into $B$ such that $x \in B \iff W_{g(x)} \subseteq A$ and $y \in W_{g(x)}$; thus we force $y$ into $A$. So we can choose $s_1$ large enough such that $E_{d_i} \subseteq A_s$ for all $s \geq s_1$.

Suppose for a contradiction that $i$ requires attention infinitely often. We will argue that at some stage $s_2 > s_1$ a new element is enumerated into $E_{d_i}$, which contradicts the choice of $s_1$. There is a first stage $s + 1 > s_1$ where $i$ requires attention through (2); let $x_0 = source(s + 1)$. If $y = cand(s + 1) \notin A$ then there is a stage $s' > s$ such that $s + 1$ is the least $n > d_i$ with $active(n) = 1$ and $E_{n-1} \subseteq A_s$. In the following stages when $i$ requires attention, $y$ will be enumerated into $E_{s_2}, E_{s_2-1}, \ldots$, and finally into $E_{d_i}$, which gives the desired contradiction. If $y \in A$, it might happen that $y$ is enumerated into $A$ before it arrives in $E_{d_i}$. Then a new candidate $y'$ from $W_{g(x_0)}$ is chosen and a new attempt is started to bring $y'$ into $E_{d_i}$. Again, it might happen that $y'$ is enumerated into $A$ before it arrives in $E_{d_i}$. However, this process cannot repeat infinitely often, because otherwise $x_0 \notin B$ and hence there is some $y \in W_{g(x_0)} - A$. This $y$ would in some iteration be chosen as a candidate which cannot be enumerated into $A$. So, at some stage $s_2 + 1 > s_1$ some $y$ is enumerated into $E_{d_i}$. Since $y \notin A_{s_2}$ and $E_{d_i} \subseteq A_{s_2}$, this implies that $E_{d_i}$ increases, a contradiction.

Thus, $i$ requires attention only finitely often and $(R_i)$ is satisfied. This completes the proof of the inductive step.

Recall that $A$ is strongly effectively simple if it is a coinfinite r.e. set and there is a total recursive function $f$ such that for all $e$,

$$W_e \subseteq \overline{A} \Rightarrow \max(W_e) < f(e).$$

Since every strongly effectively simple set is Q-complete \cite[Exercise III.6.21, a)]{10} we get the following corollary.

**Corollary 4.5** Every strongly effectively simple set has hard instances, even w.r.t. $\overline{ic}$.

It is known that hyperhypersimple sets are not Q-complete \cite[Theorem III.4.10]{10}, but we can still show that they have hard instances.

**Theorem 4.6** Every hyperhypersimple set has hard instances, even w.r.t. $\overline{ic}$.

**Proof:** The basic idea of this proof is similar to the previous one. Assume that $A$ is hyperhypersimple. We enumerate an r.e. sequence of finite sets $\{E_n\}_{n \in \mathbb{N}}$ such that $|E_n| \leq 2^n$. It suffices to get infinitely many $n$ such that there is $y \in E_n$ with $\overline{ic}(y : A) \geq n - 2$.

Let $I_n = \{p : l(p) < n - 2\}$. We initialize $E_n = \{n\}$ and may later enumerate numbers from $E_n$ into $E_{n-1}$. This time we ensure that at any stage $s$ at most two numbers of $E_n$ belong to $A_s$. We never enumerate a number twice into the same set. Furthermore, we enumerate $x$ into $E_n$ at stage $s + 1$ only if there is $p \in I_{n+1}$ which is $A$-consistent at stage $s + 1$ and $U_s(p, x) = 0$. 


From this invariant it already follows that $|E_n| \leq 2^n$: It is easy to see, by induction on $k$, that we enumerate the $(2k+1)$-st number into $E_n$ at stage $s+1$ only if there are at least $k$ programs $p$ from $I_{n+1}$ which were $A$-consistent at some previous stage and are now diagonalized (i.e., for each such $p$ there is $z \in E_n \cap A_s$ such that $U_s(p, z) = 0$). Since there are less than $2^{n-1}$ programs in $I_{n+1}$, it follows that $|E_n| < 2 \cdot 2^{n-1} + 1 = 2^n + 1$.

As in the previous proof, we say that $n$ is saturated at stage $s+1$ if for every $y \in E_n$ there is $p \in I_n$ such that $p$ is $A$-consistent at stage $s+1$ and $U_s(p, y) = \chi_{A_s}(y)$. We want to produce infinitely many $n$ which are almost always not saturated.

To this end we construct for each $e$ a sequence $d_e^0 < d_e^1 < \cdots$ such that for each $i$, $|\overline{A} \cap E_{d_e^i}| \geq 1$ or there is $n \in [d_e^i, d_e^{i+1})$ which is almost always not saturated. Suppose we have constructed at the end of stage $s$ an initial segment of this sequence, say $d_e^0 < \cdots < d_e^{m+1}$. Let $\text{count}(n, s) = |\overline{A} \cap E_{n,s}|$. We extend this initial segment at stage $s+1$ only if $\text{count}(d_e^i, s) \geq 1$ for all $i \leq m$. In the end we shall be able to argue that if the sequence is infinite then there is a weak array which witnesses that $A$ is not hyperhypersimple. Thus, the sequence must be finite, say $d_e^0 < \cdots < d_e^{m(e)+1}$, and there is $n \in [d_e^{m(e)}, d_e^{m(e)+1})$ which is almost always not saturated. Also, since the strategy to extend the $e$-th sequence is active at only finitely many stages, we can build an $(e+1)$-st sequence with $d_e^{m(e)+1} > d_e^{m(e)+1}$, which will also be finite and gives us another number that is almost always not saturated, etc.

We assign priorities as follows: The definition of the $e$-th sequence has higher priority than the definition of the $e'$-th sequence if $e < e'$. The definition of the $i$-th member of the $e$-th sequence has higher priority than the definition of the $i'$-th member if $i < i'$. Hence, we take the lexicographical ordering $<_\text{lex}$ on $\mathcal{N} \times \mathcal{N}$ as our priority ordering.

For technical reasons we enumerate for each $e$ a set $M_e$. When we are working on the $e$-th sequence we try to establish for each $d_e^i$ a number $x \in E_{d_e^i} - A$. In $M_e$ we enumerate the current candidate for $x$.

We say that $(e, i)$ requires attention at stage $s+1$ if one of the following conditions holds at the beginning of stage $s+1$.

1. $d_e^i$ is undefined and for all $j \in [0, i-1)$: $\text{count}(d_e^j, s) \geq 1$ and every $n \in [d_e^{j}, d_e^{j+1})$ is saturated at stage $s+1$.

2. $d_e^i, d_e^{i+1}$ are both defined, $\text{count}(d_e^{i}, s) = 0$, and every $n \in [d_e^{i}, d_e^{i+1})$ is saturated at stage $s+1$.

Construction:
Stage 0: Let $d_e^0 = \uparrow$ and $M_e = \emptyset$ for all $e, i$, and let $E_n = \{n\}$ for all $n$.
Stage $s+1$: Choose the lexicographically least $(e, i)$ which requires attention at stage $s+1$.

If it requires attention through (1) then let $d_e^i = s+1$, enumerate $s+1$ into $M_e$, and let $d_e^j = \uparrow$ for all $(e', j) >_{\text{lex}} (e, i)$.

If it requires attention through (2) and there is a least $n \in (d_e^i, d_e^{i+1})$, such that $\text{count}(n-1, s) \leq 1$ and there is a least $x \in E_{n,s} - (A_s \cup M_e \cup E_{n-1,s})$, then enumerate $x$ into $E_{n-1}$. If in addition $n-1 = d_e^i$ then enumerate $x$ into $M_e$. In any case, let
\[ d'_e = \uparrow \text{ for all } (e', j) >_{\text{lex}} (e, i) \]

End of Construction.

It easily follows by induction on \( s \) that \( \text{count}(n, s) \leq 2 \) for all \( n, s \), in particular, \(|E \cap \overline{A}| \leq 2\). Also, we enumerate at stage \( s + 1 \) a number \( x \) from \( E_n \) into \( E_{n-1} \) only if it does not yet belong to \( E_{n-1} \cap A \) and \( n \) is saturated. In particular, there is a program \( p \in I_{n+1} \) which is \( A \)-consistent at stage \( s + 1 \) and \( U_s(p, x) = 0 \).

Claim: For every \( e \), there are only finitely many stages where \( (e, i) \) requires attention for some \( i \).

Proof: Suppose for a contradiction that there exists a least \( e \) and infinitely many \( s \) such that \( (e, i) \) requires attention at stage \( s + 1 \) for some \( i \). Then we argue that \( A \) is not hyperhypersimple. First, there is a least stage \( s_0 \geq 1 \) such that no \( (e', i') \) with \( e' < e \) requires attention at any stage \( s \geq s_0 \). Then we define \( d_0^e = s_0 \) at stage \( s_0 \) and we enumerate \( s_0 \) into \( M_e \). By the choice of \( s_0 \), the value of \( d_0^e \) has stabilized. Note that all numbers which have been previously enumerated into \( M_e \) are less than \( s_0 \) and so they do not matter for the following. By induction on \( s \geq s_0 \), it follows that \( E_n, s \) contains at most one number from \( M_{e, s} - A \) for all \( n \geq d_0^e \).

Now we distinguish two cases:

(a) If there is a least \( i \) such that \( (e, i) \) requires attention infinitely often then there is a stage \( s_1 \geq s_0 \) where all \( d_j^e \) with \( j \leq i \) have stabilized. Thus, \( (e, i) \) infinitely often requires attention through (2) and \( d_{s+1}^e \) tends to infinity. But then it follows similarly as in the previous proof that unboundedly many numbers are eventually enumerated into \( E_{\infty}^e \) which contradicts the fact that the cardinality of \( E_{\infty}^e \) is bounded:

If \( (e, i) \) requires attention through (2) at any stage \( s \geq s_1 \) then \( \text{count}(d_i^e, s) = 0 \), thus an \( (e, j) \) with \( j > i \) cannot require attention through (2) at any later stage \( s' > s \), until a new number is enumerated into \( E_{\infty}^e \) and \( \text{count}(d_i^e, s') = 1 \). During that time \( M_e \) does not change. This guarantees that eventually a new number is enumerated into \( E_{\infty}^e \), since there exist numbers \( z \in (\bigcup_{n \geq d_i^e} E_{n, s}) - (A \cup M_e \cup E_{d_i^e, s}) \). Since \( |E_{n, s} \cap (M_{e, s} - A)| \leq 1 \) for \( n \geq d_i^e \), it causes no problems to maintain the constraint that a number \( x \) is enumerated from \( E_n \) into \( E_{n-1} \) at stage \( s + 1 \), only if \( x \not\in (M_{e, s} \cup A) \).

(b) If for every \( i \) there are only finitely many stages (but at least one stage) where \( (e, i) \) requires attention, then it follows that the values \( d_i^e \) stabilize and form an infinite increasing sequence. Let \( d_i^e \) denote the final value. Since the sequence is infinite it follows that \( \lim_s \text{count}(d_i^e, s) \geq 1 \), thus \( |E_{\infty}^e \cap \overline{A}| \geq 1 \). From the actual construction we get \( |E_{\infty}^e \cap \overline{A}| = 1 \) and \( E_{\infty}^e \subseteq M_e \).

Uniformly in \( i \) we enumerate an r.e. set \( U_i \) as follows: If there is a stage \( s + 1 \geq s_0 \) where \( (e, i) \) is the least pair which requires attention through (2) and a number \( x \) is enumerated into \( E_{\infty}^e \), then enumerate \( x \) into \( U_i \).

Since each such \( x \) is also enumerated into \( M_e \) and is therefore blocked for the other sets, it follows that the \( U_i \)'s are pairwise disjoint. By the remarks above, each \( U_i \) intersects \( \overline{A} \). Thus, \( A \) is not hyperhypersimple. This contradiction completes the proof of the claim. \( \square \)

Thus, for each \( e \) there exists a maximal \( m(e) \geq 0 \) such that the value of \( d_{m(e)+1}^e \) stabilizes and no \( (e, j) \) with \( j > m(e) + 1 \) requires attention at any sufficiently large stage. This means that there exists \( n \in [d_{m(e)}^e, d_{m(e)+1}^e) \) which is almost always not
saturated. Thus, there is \( y \in E_n \) with \( \overline{c}(y : A) \geq n - 2 \). Clearly, we get infinitely many pairwise different such \( y \)'s. This completes the proof.

The previous result does not hold for hypersimple sets, since one can construct a hypersimple set that does not have hard instances. This can be done, e.g., by a direct modification of the proof of the next theorem.

Recall that \( A \) is effectively simple if it is a coinfinite r.e. set and there is a recursive function \( f \) such that for all \( e \),

\[
W_e \subseteq \overline{A} \Rightarrow |W_e| \leq f(e).
\]

It is known that every effectively simple set is T-complete [10, Proposition III.2.18].

**Theorem 4.7** There is an effectively simple set which does not have hard instances. In particular, there is a T-complete set which does not have hard instances.

**Proof sketch:** The construction in the proof of Theorem 3.2 is not combinable with the requirement of making \( A \) effectively simple. Therefore, we use a modified version were we do not attempt to have the instance complexity as low as possible.

In the following we outline the construction. \( A \) will be effectively simple for some \( f \) to be determined later. As in the proof of Theorem 3.2 we are given a uniformly r.e. sequence \( \{E_k\}_{k \in \mathbb{N}} \) and we build a partial recursive function \( \psi \) such that for almost all \( k \) and for each \( x \in E_k \) there is some \( p \in \{0, 1\}^k \) witnessing that \( ic_{\psi}(x : A) \leq k \).

How do we define \( \psi_p \)? We will keep a list \( S_S = S_k \) of programs of length \( k \). The length of \( S \) will be fixed (depending on \( k \)). Furthermore, we have a pool \( P = P_k \) of unused programs of length \( k \). At the beginning \( |S| + |P| = 2^k \). During the construction some of the programs in \( S \) may become inconsistent with \( A \), in which case they are removed from \( S \) and new programs from \( P \) are inserted into \( S \). There may also exist a “back-up program” chosen from \( P \).

The programs in \( S \) will be defined at \( x \) with a 0/1-value only if \( x \) was enumerated into \( E_k \). The definition proceeds in a round-robin fashion: The first program in \( S \) takes care of the first number which is enumerated into \( E_k \), the second program takes care of the second number, and so on. In this way we handle the first \( |S| \) numbers. Ideally, we would like that again the first program takes care of the \((|S|+1)\)-st number, etc. However, this does not work, because as soon as a program was brought into play we have to define it for larger and larger inputs. So it might happen that all of our programs are already defined (with output \( \perp \)) at \( x \) when \( x \) is enumerated as the \( |S| + 1 \)-st number at stage \( s \).

Thus, we are using a program \( q \) from \( P \) which is still everywhere undefined and define it as \( \chi_{A,S}(z) \) for all \( z < s \), in particular this covers all numbers currently in \( E_k \). For all larger values we output \( \perp \). \( q \) is called the current back-up. We also suspend defining the programs in \( S \) until new numbers \( x \geq s \) are enumerated into \( E_k \). Then we continue as above for the next \( |S| \) such numbers. After that a new program from \( P \) is defined as the current back-up in a similar way as \( q \), and so on.

What is the advantage of that scheme? It is more robust against injuries which may happen when a number \( x \) with \( \psi_p(x) = 0 \) is later enumerated into \( A \). In that case only one \( p \in S \) is destroyed. Also, only the \( x \in E_k \) are critical because for \( x \not\in E_k \)
we have $\psi_p(x) = \bot$. If $p$ is destroyed then we assign a new program from $P$ as a substitute.

A crucial part in this process is the definition of the new back-up $q$ when a round has been completed at the beginning of stage $s$. Before we define $\psi_q$, we enumerate all $x < s$ into $A$ which do not belong to any $E_n$ with $n < g(k)$: This defines the current $A_s$. Here $g$ is some fast growing function to be determined later. Then we define $\psi_q(x) = \chi_{A_s}(x)$ for all $x < s$, and $\psi_q(x) = \bot$ otherwise.

We use the following strategy to make $A$ effectively simple. If at the end of some stage $s$ we have $W_{e,s} \subseteq A_s$ and $|W_{e,s}| > f(e)$, then choose an $x \in W_{e,s}$ which does not belong to any $E_n$ with $n \leq g(e)$ and enumerate it into $A$. Note that $x$ exists if we choose $f$ large enough such that $f(e) \geq |E_0| + |E_1| + \cdots + |E_{g(e)}|$.

This completes the description of the construction. It remains to choose the parameters such that it works. We first count how many of the $\psi_q$ with $l(p) = k$ are used. Then we choose $|E_k|$ and $g$ in such a way that the number of used programs is less than $2^k$.

Let $m = \max\{n : g(n) \leq k\}$. Then for each $i \leq m$ there can be $|[E_i]|/|S_i|)$ many rounds and after each round all programs in $S_k$ may be destroyed (and have to be replaced by new ones from $P_k$). At this time it is important that after the action of $i$ we immediately define the new programs that replace the former ones which have been destroyed. We can do this without any further enumeration of elements into $A$. There is no cascading effect which could blow up the number of injuries. Thus, at most $|S_k|\sum_{i=1}^{m} |[E_i]|/|S_i|)$ many programs in $S_k$ are ever injured.

How many of the back-up functions are destroyed? Note that this may happen each time when some $i < k$ acts, i.e., whenever $i$ completes a round. Thus, at most $\sum_{i=1}^{k} |[E_i]|/|S_i|)$ many back-up functions are destroyed.

The number of injuries from making $A$ effectively simple can be bounded by $m + k$: If we act for the sake of $W_{e,s} \cap A \neq \emptyset$ (which happens at most once), then a program from $S_k$ may be destroyed only if $e < m$, and a current back-up program can be destroyed only if $e < k$. To see the latter, note that if the current $q$ is defined at $x \not\in E_0 \cup \cdots \cup E_{g(k)}$, then $\psi_q(x) \in \{1, \bot\}$ because of the additional enumeration of numbers into $A$ which was performed when $q$ was brought into play.

Thus, we need to ensure that for almost all $k$:

$$2^k > |S_k|\sum_{i=1}^{m} |[E_i]|/|S_i|) + \sum_{i=1}^{k} |[E_i]|/|S_i|) + m + k.$$

Let $|S_k| = [2^k/k]$, $g(k) = 2^k$, and $E_k = \{x : C(x) < 3k/2\}$, so $|E_k| < 2^{3k/2}$. Define the recursive function $f$ by $f(e) = [\sum_{i=0}^{g(e)} 2^{3i/2}]$. The right hand side of (+) is bounded above by

$$(2^k/k)(\log k)^2 \sqrt{k} + k^22^{k/2} + \log k + k$$

which is less than $2^k$ for all sufficiently large $k$.

With this choice of parameters we get for almost all $x$, $C(x) \geq (3/2)(i \psi_{x} : A) - 1$, i.e., $i \psi_{x} : A \leq (2/3)C(x) + 1$. Thus, $A$ does not have hard instances.

The previous results characterize the reducibilities $\leq_r$ with $r \in \{m, \text{btt}, c, d, p, \text{tt}, \text{wtt}, Q, T\}$ (cf. the figure in [11]), p. 341) such that every $r$-complete set has hard instances, for both $i \psi$ and $\overline{i \psi}$. In the following table we have marked the possible combinations.
Remark: The T-degrees of r.e. sets with hard instances do not coincide with any of the known degree classes. It can be shown that they form a proper subclass of the r.e. nonrecursive degrees and that they properly extend the array nonrecursive degrees.

Acknowledgments: I would like to thank Lance Fortnow, Bill Gasarch, and Paul Vitányi for comments on a preliminary version of this paper.

References

[1] J. Barzdin. Complexity of programs to determine whether natural numbers not greater than $n$ belong to a recursively enumerable set. Soviet Math. Dokl., 9:1251–1254, 1968.

[2] H. Buhrman, P. Orponen. Random strings make instances hard. To appear in: Proceedings Structure in Complexity Theory, Ninth Annual Conference, 1994.

[3] G. J. Chaitin. Information-theoretic characterizations of recursive infinite strings. Theoretical Computer Science, 2:45–48, 1976.

[4] R. Downey, C. Jockusch, M. Stob. Array nonrecursive sets and multiple permitting arguments. In: Recursion Theory Week, Proceedings, Oberwolfach 1989 (Eds. K. Ambos-Spies, G. H. Müller, G. E. Sacks), pp. 141–174, Lecture Notes in Mathematics 1432, Springer-Verlag, 1990.

[5] L. Fortnow, M. Kummer. Resource-bounded instance complexity. Manuscript, 1994.

[6] K. Ko. A note on the instance complexity of pseudorandom sets. In: Proceedings Structure in Complexity Theory, Seventh Annual Conference, pp. 327–337, 1992.

[7] K. Ko, P. Orponen, U. Schöning, O. Watanabe. What is a hard instance of a computational problem? In: Structure in Complexity Theory (Ed. A. Selman), pp. 197–217, Lecture Notes in Computer Science 223, Springer-Verlag, Berlin, 1986.

[8] M. Li, P. Vitányi. An introduction to Kolmogorov complexity and its applications. Springer-Verlag, New-York, 1993.

[9] D. W. Loveland. A variant of the Kolmogorov concept of complexity. Information and Control, 15:510–526, 1969.

[10] P. Odifreddi. Classical recursion theory. North-Holland, Amsterdam, 1989.
[11] P. Orponen. On the instance complexity of NP-hard problems. In: Proceedings Structure in Complexity Theory, Fifth Annual Conference, pp. 20–27, 1990.

[12] P. Orponen, K. Ko, U. Schöning, O. Watanabe. Instance complexity. J. of the ACM, 41:96–121, 1994.

[13] R. I. Soare. Recursively enumerable sets and degrees. Springer-Verlag, Berlin, 1987.

[14] J. Tromp. On a conjecture by Orponen + 3. In: Descriptive Complexity. (Eds. R. V. Book, E. Pednault, D. Wotschke), p. 20, Dagstuhl-Seminar-Report; Vol. 63, May 1993.