STRENGTHENING PROMPT SIMPLICITY

DAVID DIAMONDSTONE, KENG MENG NG

Abstract. We introduce a natural strengthening of prompt simplicity, and study its relationship with existing lowness classes. We show that this notion is intimately related to superlow cuppability. This highlights the relationship between the computational and dynamical properties of a c.e. set. We also study the effect that lowness properties have on the behaviour of a set under the join operator. In particular we construct an array noncomputable c.e. set, whose join with every low c.e. set is low.

1. Introduction

Two of the most influential concepts in the study of computably enumerable sets are that of lowness and prompt simplicity. Lowness is concerned with the intrinsic information content of a set (or rather the lack thereof). The most well-studied notion of lowness is that of computational lowness: A set $A$ is low if $A' \equiv_T \emptyset'$. Thus $A$ is unhelpful as an oracle in computations. There have been many results highlighting the fact that low c.e. sets resemble computable sets. A well-known variation on lowness is superlowness introduced by Bickford and Mills [2]: A c.e. set $A$ is superlow if $A' \equiv_{wtt} \emptyset'$. Here $A \leq_{wtt} B$ means that $A$ is Turing reducible to $B$ with a computable bound on the use. Recent developments in the theory of algorithmic randomness have shown that the theory of low and superlow c.e. sets are much deeper than originally thought.

Prompt simplicity was introduced by Maass [10] in connection with the automorphisms of the lattice of c.e. sets. This is a dynamic property which describes the speed which elements are enumerated into the set. A promptly simple set $A$ resembles $\emptyset'$ in its dynamic properties. Ambos-Spies, Jockusch, Shore and Soare [1] proved the beautiful result which linked the dynamic property of a set with a degree theoretic property: $A$ is promptly simple iff $A$ is low cuppable. Here we say that $A$ is $C$ cuppable if there is a c.e. set $B \in C$ such that $A \oplus B \equiv_T \emptyset'$. They also proved that this class is exactly the class of c.e. sets which is not half of a minimal pair.

The result of Ambos-Spies, Jockusch, Shore and Soare demonstrated a certain robustness in the class of low cuppable sets. The subclass of the superlow cuppable sets was also studied. Nies asked if these two classes were equal, and Diamondstone [5] answered this by constructing a promptly simple set which is not superlow cuppable. The result of Diamondstone suggests a natural question: Is there a strengthening of prompt simplicity that characterizes superlow cuppability?

The primary aim of this paper is to introduce a natural strengthening of prompt simplicity in order to understand the question above. We will examine classical theorems demonstrating the interplay between prompt simplicity and lowness, and see to what extent the analogy carries over to the effective versions. Our choice is motivated by the following three considerations:

1. To understand the dynamic properties of the superlow cuppable sets.
2. To further examine the relationship between lowness and promptness.
3. To investigate how best to strengthen prompt simplicity.
We will discuss each of these in turn.

Our first motivation is to understand the intricate relationship between the dynamic properties of a c.e. set and its computational characteristics from the point of view of cupping. In particular we want to find an analogue of the theorem of Ambos-Spies, Jockusch, Shore and Soare. We introduce a dynamic notion which implies being of promptly simple degree which we call strongly prompt. We are able to show that this implies being superlow cuppable, and that in fact some suitable variation of superlow cuppability characterizes strong promptness. This demonstrates another relationship between the dynamic and computational properties of a set.

The second motivation was inspired by recent developments in the theory of algorithmic randomness, particularly in the study of different lowness notions. Our aim is to show that if a set was sufficiently feeble in terms of its computational aspects, then its dynamic properties would also be affected. Heuristically a notion of lowness is a property indicating feebleness in the operations concerned. For instance a set $A$ is low for random if every Martin-Łöf real is Martin-Łöf relative to $A$. This property asserts that $A$ is useless in derandomizing other reals. Another fundamental lowness class is the $K$-trivial reals. $A$ is said to be $K$-trivial if for all $n$, $K(A | n) \leq K(0^n) + O(1)$. Here $K(\sigma)$ denotes the prefix-free Kolmogorov complexity of the finite string $\sigma$. That is, $K$-trivial reals have minimal initial segment complexity which indicates that $A$ contains little algorithmically useful information. The c.e. $K$-trivial sets form a natural $\Sigma^0_3$-ideal in the c.e. degrees. Despite the fact that the notion of $K$-triviality has been around for many years, it was only recently that $K$-triviality became well-understood. Several difficult techniques were invented by Downey, Hirschfeldt, Nies and Stephan [7] to handle this class. Adapting these techniques, Nies [13] showed that $K$-triviality was equivalent to low for randomness. He also showed that every $K$-trivial set was superlow demonstrating a pleasing connection between classical lowness and Kolmogorov lowness.

Another central lowness subclass is the class of strongly jump traceable sets, introduced by Figueira, Nies and Stephan [8]. A is strongly jump traceable if for every computable non-decreasing and unbounded function $h$ (also called an order), there is a computable function $g$ such that for every $x$, the value of the partial $A$-computable function $\Phi^A_{x}(x) \in W_{g(x)}$ and $|W_{g(x)}| < h(x)$. This property says that the value of $\Phi^A_{x}(x)$ can be effectively traced with arbitrarily slow growing bounds (on the size of the trace). Cholak, Downey and Greenberg [3] showed that every c.e. strongly jump traceable set is $K$-trivial, and the former class forms a $\Pi^0_4$-ideal in the c.e. degrees. A recent result of Greenberg and Nies [9] revealed an unexpected connection between traceability and a notion from classical computability. Cholak, Groszek and Slaman [4] introduced the class of almost deep degrees, where they called a c.e. degree $a$ almost deep if for every low c.e. degree $b$, $a \cup b$ is low. They showed that this is a non-trivial ideal in the c.e. degrees. Another non-trivial ideal was studied by Ng [12]: a c.e. degree $a$ is almost superdeep if for every superlow c.e. degree $b$, $a \cup b$ is superlow. Each almost (super)deep degree is a fortiori (super)low. Greenberg and Nies showed that every strongly jump traceable c.e. set is almost superdeep.

The abovementioned are examples of a large collection of results in literature demonstrating that the strongly jump traceable c.e. sets greatly resemble the computable sets. We might expect that at this level of computational lowness, the strongly jump traceable sets would also resemble the computable sets in their dynamic properties. However Figueira, Nies and Stephan [8] showed that there is a promptly simple c.e. set which is strongly jump traceable. By partially relativizing strong jump traceability, Ng [11] studied a further subclass of the strongly jump traceable c.e. sets, called the hyper jump traceable sets. He showed that no hyper jump traceable c.e. set is promptly simple. Hence the hyper jump traceable c.e. sets are computationally so weak that they cannot
have a prompt enumeration. We prove in Section 3 that no strongly jump traceable c.e. set can be strongly prompt.

Our third motivation for introducing strong promptness comes from a desire to understand what it means to “have an effective bound on prompt simplicity”. Specifically, a set will be strongly prompt if there is a suitably effective bound \( g(e) \) on the size of \( W_e \) which guarantees a prompt permission. If this bound were computable, we would have the notion of effective simplicity, and such sets are known to be complete. In fact, we show that allowing \( g \) to be \( n \)-c.e. forces completeness. We instead require that \( g \) be \( \omega \)-c.e. Recall that a function (or a set) being \( \omega \)-c.e. is equivalent to it being wtt-computable from \( \emptyset' \). We imagine that we have a size bound \( \hat{g}(e, s) \) that we think guarantees a prompt permission. Then, a prompt permission is denied for \( W_e \) if it exceeds the size bound \( \hat{g}(e, s) \) at stage \( s \), but \( A \) has not changed below \( x \) by stage \( p(s) \). The fact that \( g \) is \( \omega \)-c.e. ensures that we have a computable bound on the number of times this can occur. This definition of strong promptness is arguably the most natural way to strengthen the notion of prompt simplicity, and captures a large number of prompt constructions. To wit, many constructions of promptly simple c.e. sets already give (or can be modified to give) a strongly prompt c.e. set. For instance the standard construction of a promptly simple superlow c.e. set \( A \) also makes \( A \) strongly prompt, since the \( e^{th} \) prompt requirement may only be denied from acting at most \( 2^e \) times. Other notable examples from algorithmic randomness include the following results:

- The cost function construction of a promptly simple \( K \)-trivial set.
- Every Martin-Löf random \( Z \) which is \( \omega \)-c.e. computes a promptly simple set.

In contrast, no strongly prompt set can be computed from every \( \omega \)-c.e. Martin-Löf random (Greenberg, Nies [9]).

The paper is organized as follows. In Section 2 we lay down the basic definitions and notations. We then prove a few fundamental properties about the strongly prompt sets. We also consider the structure of the Turing and the wtt-degrees containing a strongly prompt set. In Section 3 we consider the interactions between strong promptness and superlow cuppability. In Theorem 3.1 we prove that every strongly prompt c.e. set is superlow cuppable. One is motivated to ask whether these two properties are equivalent. Unfortunately, as we will see, strong promptness is not even a Turing degree notion, but instead applies naturally to weak truth-table degrees. However, it may still be the case that being superlow cuppable is equivalent to having the same Turing degree as a strongly prompt set. We leave this as a question. In Theorem 3.3 we provide a characterization of strong promptness in terms of a cupping property: A c.e. set \( A \) is strongly prompt iff there is a superlow c.e. set \( B \) such that \( \emptyset' \leq_T A \oplus B \) with an \( \omega \)-c.e. bound on the use.

In Section 4 we show that with the possible exception of the containment of (3) in (1), no one of the following three ideals in the c.e. degrees are included in any of the others: (1) \( K \)-trivial degrees, (2) Almost deep degrees and (3) Almost superdeep degrees. We do not yet know if (3) is contained in (1), or if every almost superdeep degree is strongly jump traceable. In Theorem 4.3 we show that for any benign cost function \( c \), there is a strongly prompt set \( A \) satisfying \( c \) (the terminologies will be explained in Section 4). As a corollary we derive that \( K \)-trivial sets can be superlow cuppable. This also gives an alternative proof of Greenberg and Nies’ result [9] that no single cost function serves to define the strongly jump traceable c.e. sets. In Theorem 4.6, we construct an almost deep c.e. degree which is not totally \( \omega \)-c.e. Downey, Greenberg and Weber [6] introduced the totally \( \omega \)-c.e. degrees to capture the multiple permitting required in a certain constructions in classical degree theory. Here a c.e. degree \( a \) is totally \( \omega \)-c.e. if every \( f \leq_T a \) is \( \omega \)-c.e. As a corollary to
Theorem 4.6, we obtain an almost deep degree which is array noncomputable, and therefore not superlow.

2. The degrees of superprompt sets

Henceforth, every set mentioned is c.e. unless otherwise stated. Throughout the paper, \( \Phi_e \) will denote the \( e \)th Turing functional in some computable enumeration, and \( \varphi_e \) its use. An expression depending on one or more c.e. sets, computable functions, and/or computable sequences may have \([s]\) appended to denote that every stage-dependent object is taken at stage \( s \). When we say \( x \) enters \( W_{e, \text{at } s} \), we refer to the (any fixed) standard enumeration of the c.e. sets. We assume that \( W_{e,\emptyset} = \emptyset \).

Recall that a c.e. set \( A \) is of promptly simple degree, if there is an enumeration \( \{A_s\}_{s \in \omega} \) of \( A \) and an increasing computable function \( p : \omega \to \omega \) such that for every \( e \) for which \( W_e \) is infinite, there is some \( x \) and \( s \) such that \( x \in W_{e, \text{at } s} \land A_s \upharpoonright x \neq A_{p(s)} \upharpoonright x \).

**Definition 2.1.** A set \( B \) is strongly prompt if there is an enumeration \( \{B_s\}_{s \in \omega} \) of \( B \), an increasing computable function \( p : \omega \to \omega \), called the “promptness function”, and an \( \omega \)-c.e. function \( g : \omega \to \omega \), such that the following holds:

\[(1) \quad |W_e| \geq g(e) \to (\exists x \exists s)(x \in W_{e, \text{at } s} \land B_s \upharpoonright x \neq B_{p(s)} \upharpoonright x)\]

That is, a strongly prompt set has a c.e. enumeration where there is a computable bound on the number of times a request for a prompt change can be denied. We illustrate this with examples. Imagine we want to build a strongly prompt set \( A \) satisfying certain negative requirements (such as lowness). Each time a promptness requirement demands that we change \( A \) (because \( |W_{e,s}| \) has grown beyond \( g(e,s) \)), then we have to either change \( A \) promptly at \( s \), or deny the prompt change by increasing \( g(e,s+1) > \max W_{e,s} \). Since \( g \) must be \( \omega \)-c.e., the number of times we can deny each prompt requirement (due to other negative constraints) must be computably bounded.

Suppose on the other hand, we were given a strongly prompt set \( A \) with enumeration \( \{A_s\}_{s \in \omega} \), bound \( g \) and promptness function \( p \). To test for prompt changes we enumerate an auxiliary c.e. set \( W_i \). Whenever we want to force an \( A \)-change below \( g(i,s) \) we enumerate \( \{0, \ldots, g(i,s)\} \) into \( W_i \). The result is either a prompt \( A \)-change below \( g(i,s) \), or a refusal in the form of an increase in \( g(i,s) \). The difference here between a prompt set and one which is strongly prompt is that with a prompt set we do not know ahead of time how many times \( W_i \) may be refused a change. However with the strongly prompt set \( A \), we know this can only happen at most \( k \) times (where \( k \) is the bound on the number of mind changes in \( g(i) \)). This fact is exploitable and enables us to prove Theorem 3.1. For another application, we invite the reader to work out a direct proof of the following fact: If a set \( A \) is strongly prompt, then \( A \) is not strongly jump traceable.

Classically the notions of prompt simplicity and prompt permitting are slightly different (though they are the same up to Turing degree). Since the latter is more useful, we have defined strong promptness as an analogue of “prompt permitting”. One can also consider the analogue of a promptly simple set: A set \( B \) is strongly promptly simple if it satisfies Equation (2), immediately below, for some enumeration \( \{B_s\}_{s \in \omega} \) of \( B \), a computable \( p \) and some \( g \leq_{wtt} \emptyset' \). In Proposition 2.4 we show that every strongly prompt set is wtt-equivalent to a strongly promptly simple set.

\[(2) \quad |W_e| \geq g(e) \to (\exists x \exists s)(x \in W_{e, \text{at } s} \land B_{p(s)})\]

We show that \( \omega \text{-c.e.} \) is the first place in the Ershov difference hierarchy where such a restriction on \( g \) makes sense. It is not hard to verify that if we replace the condition that “\( g \) is \( \omega \text{-c.e.} \)” in Definition...
2.1 by “g is $\Delta^0_2$”, then the resulting notion coincides with being of promptly simple degree. If, on the other hand, we required g to be computable, we would have “effective promptness” (by analogy with effective simplicity), and such sets are complete, just as effectively simple sets are known to be complete. In fact, the same would be true if we only required g to be n-c.e.

**Theorem 2.2.** For any $n \in \omega$, if you replace $\omega$-c.e. by n-c.e. in Definition 2.1, then any set satisfying the resulting definition is Turing complete.

**Proof.** Suppose $\{B_s\}_{s \in \omega}, p$ and g are as in Definition 2.1, but g is n-c.e. for some fixed n (instead of $\omega$-c.e.). We prove the case $n = 1$, and note that the proof generalizes easily to any fixed $n > 1$ with a higher level of nonuniformity involved.

The basic idea is as follows. We make two separate attempts at computing $\emptyset'$ from B. The intuition is that B can only deny our request for a prompt change at most once. To request a prompt change we build the auxiliary c.e. sets $U_0, U_1, \ldots$. Whenever k enters $\emptyset'$ we use $U_k$ to request B to change promptly (by enumerating a number of elements into $U_k$). B has to either change promptly, or it has to deny us this change (this will be reflected by an increase in g). If B changes promptly at almost every $k$, then we can easily define a reduction $\emptyset' \leq_T B$ using the promptness function $p$ for B. If there are infinitely many k for which the prompt permission is denied by B, then for this infinite c.e. set of k’s, a second round of requests for a prompt change cannot be denied. In the construction below these numbers are called “good”.

By the recursion theorem and the usual slowdown lemma we may assume that $U_x = W_{r(x)}$ for some computable r, and that if z is enumerated into $U_x$ at stage s of the construction then z is enumerated into $W_{r(z)}$ under the standard enumeration at some $t > s$. In fact we can require that $t > p(s')$ for the least stage s’ such that every number enumerated into $U_x$ before stage s of the construction has already showed up in $W_{r(z)}$. When we say we enumerate a number z into $U_x$, we mean that we enumerate every number not larger than z (which has not already been enumerated) into $U_x$ at s. Fix a 1-1 computable enumeration $\{k_n\}$ of $\emptyset'$ where at every stage s, exactly one element $k_s$ is enumerated into $\emptyset'$. We say that a number n is good at s if n has not yet been permitted (this will be defined during the construction), and $g(r(n), s) \neq g(r(n), t)$ for some $t < s$; i.e. $g(r(n))$ has already changed once. Informally we say that n is good if we had already put numbers into $U_n$, and a prompt change in B was denied.

**Construction:** Let c(i) denote the i$^{th}$ good number (larger than c(i − 1)) to be found in the construction. At stage 0, declare every number as “not permitted”. At stage $s > 0$ if c($k_s$) 1, we enumerate $g(r(c(k_s)), s)$ into $U_{c(k_s)}$. Additionally if $k_s$ is not good at s we enumerate $g(r(c(k_s)), s)$ into $U_{k_s}$. Run the given enumeration of B and check if B changes below $g(r(k_s), s)$ between s and $p(s')$, where $s' > s$ is such that $W_{r(k_s)},s \supseteq \{0, \cdots g(r(c(k_s)), s)\}$. That is, $W_{r(k_s)}$ has caught up with $U_{k_s}$. If B has changed then we declare that $k_s$ has been permitted. Go to stage $s + 1$ of the construction.

We now explain briefly what the construction does. We have two separate attempts at computing $\emptyset'$ from B. The primary attempt tries to code $\emptyset' \mid z$ into $B \upharpoonright g(r(z))$, provided z is larger than the biggest good number found. Since a good number c(i) denotes a denial of prompt permission, this means that if only finitely many good numbers are found during the construction, then our primary reduction successfully reduces B to $\emptyset'$ (at almost every input). On the other hand if infinitely many good numbers are found, then our backup strategy will try and decide if $z \in \emptyset'$ via $B \upharpoonright g(r(c(z)))$.

The first action in the construction enumerates $g(r(c(k_s)), s)$ into $U_{c(k_s)}$ for the sake of the backup
strategy. The second action in the construction enumerates \( g(r(k_s), s) \) into \( U_{k_s} \) for the sake of the primary strategy.

**Verification**: Assume that \( c \) is not total, i.e. let \( i_0 \) be the largest such that \( c(i_0) \downarrow \). Then for any \( z > c(i_0) \) it is easy to see that \( z \not\in \emptyset' \text{ iff } z \not\in \emptyset'_s \) where \( s \) is a stage large enough so that \( B \uparrow 1 + g(r(z), 0) \) is stable (on the given enumeration). To see this assume that \( z = k_t \) for some \( t > s \). At stage \( t \) of the construction we would have enumerated \( g(r(z), t) \) into \( U_z \). It is clear that \( U_z \) had been empty up till stage \( t \), and so \( z \) must be permitted at \( t \). Otherwise \( z \) is never permitted at all in the construction and we must have \( g(r(z), t) < \lim g(r(z), -) \). Hence at some later stage in the construction we will discover that \( z \) is good, a contradiction. Hence \( z \) has to be permitted at stage \( t \) of the construction, which contradicts that \( B \uparrow 1 + g(r(z), 0) \) is stable at \( s < t \). Hence \( \emptyset' \leq_T B \).

Now assume that \( c \) is total (and clearly computable). Fix \( z \) and let \( t \) be the least stage in the construction where \( c(z) \downarrow \). Let \( s \) be the least stage \( > t \) where \( B \uparrow 1 + g(r(c(z)), t) \) is stable with respect to the given enumeration. Now we claim that \( z \in \emptyset' \text{ iff } z \not\in \emptyset'_s \). Suppose for a contradiction that \( z = k_u \) for some \( u > s > t \). At stage \( u \) of the construction we would enumerate \( \lim g(r(c(z)), -) = g(r(c(z)), u) \) into \( U_{c(z)} \). It is easy to verify that \( c(z) \) is good at every stage \( \geq t \). The only other batch of numbers which could be in \( U_{c(z)} \) at stage \( u \), can only have been enumerated at some stage \( u' < t \). However if this is the case then these numbers must have been enumerated earlier at \( u' \) for the sake of the primary strategy, and we would already have obtained a prompt \( B \)-denial, which means that \( g(r(c(z)), u) \) is larger than these numbers. By the fact that \( |U_{c(z)}| > \lim g(r(c(z)), -) \), some number enumerated into \( U_{c(z)} \) must have caused a \( B \)-change. Since \( c(z) \) cannot be permitted at \( u' \), this number cannot be in the first batch. Hence a number enumerated in the second batch at stage \( u \) had caused a \( B \)-change, which contradicts the stability of \( B \uparrow 1 + g(r(c(z)), t) \) at \( s \). Hence we have \( \emptyset' \leq_T B \). 

We show that strong promptness is not a Turing degree notion. That is, strong promptness is not closed under Turing equivalence. In fact, the following theorem shows that there is a Turing complete c.e. set which is not strongly prompt.

**Theorem 2.3.** There is a c.e. set \( A \equiv_T \emptyset' \) which is not strongly prompt.

**Proof.** We construct a c.e. set \( A \) by specifying an enumeration \( \{A_e\}_{e \in \omega} \). Let \( \overline{A}_e = \{a_0^e < a_1^e < \cdots \} \). We ensure that \( \emptyset' \leq_T A \) by enumerating \( a_e^e \) into \( A_{e+1} \) whenever \( e \) enters \( \emptyset' \) at \( s \). Since each (possible) computable approximation for \( A \) is of the form \( \{D_{\eta(n)}\}_{n \in \omega} \) for a computable \( h \), we will denote the \( e^{th} \) possible approximation by \( \{V_s^e\}_{e \in \omega} \). We say that \( V_s^e \) converges, if \( \varphi_e(r) \downarrow \) for all \( r \leq x \). We adopt the convention that if \( V_s^e \) converges at stage \( s \) the construction, then \( e, x < s \).

We let \( p_e \) be the \( e^{th} \) partial computable function which is increasing, and \( g_e \) be the \( e^{th} \) pair \( (g_e(x, s), b_e(x)) \) where \( g_e \) is total computable, and \( b_e \) is partial computable. Every \( \omega \)-c.e. function is approximated by one of these pairs (with \( b_e \) bounding the number of mind changes in \( g_e \)). Note that we do not require that \( \{g_e\} \) lists all total computable functions of two variables, which is impossible. Rather we only require that \( \{g_e\} \) is large enough to simulate every \( \omega \)-c.e. function in the limit. To kill off all possible witnesses to the strong promptness of \( A \), we ensure that

\[
\varnothing_e : \text{If } \{V_s^e\}_{s \in \omega} \text{ and } p_e \text{ are total, and } g_e \text{ is } \omega \text{-c.e. via } b_e, \text{ and } \bigcup_s V_s^e = A, \\
\text{then for some } k, |W_k| > g_e(k) \text{ and } \forall x \forall s(x \in W_k, \text{ at } s \rightarrow V_s^e \upharpoonright x = V_{p_e(s)}^e \upharpoonright x). 
\]
By the recursion theorem and the slowdown lemma, we again assume that we are building $W_k$ for $k$ in an infinite computable set of indices, and that any number $z$ we enumerate into $W_k$ at stage $s$ of the construction appears strictly later in the standard enumeration of $W_k$. Each $R_e$ has a parameter $k_e$, whose value may change during the construction, and is picked to be one of these indices $k$ for which we build $W_k$. At stage $s$ of the construction we say that $V^e_t$ agrees with $A$ if $V^e_t \upharpoonright a^e_s = A \upharpoonright a^e_s$. At stage $s$ of the construction, when we say that we lift $a^e_s$ to a fresh value, we mean that we pick a fresh number $z$ and enumerate $a^e_s, \ldots, a^e_z$ into $A$ (so that now $a_e > z$).

The action of each $R_e$ is finitary, and moves the marker $a^e_s$ finitarily. $R_e$ is initialized each time $a^e_s$ is moved for $i < e$. This means that the $R_e$-state is reset back to 0 and $k_e$ is made undefined. The $R_e$-state can be 0, 1 or 2 corresponding to the following steps in the basic strategy:

1. **Unstarted phase**: Pick a fresh value for $k_e$ and wait for $b_e(k_e) \downarrow$.
2. **Preparation phase**: Enumerate $a^e_s$ into $A$, and lift it to a fresh value (in particular, above $g_e(k_e,s)$). Wait for $t$ such that $V^e_t$ converge and agree with $A$.
3. **Diagonalization phase**: Enumerate every number less than $a^e_s$ into $W_{k_e}$. Assume this is stage $s$ of the construction. Hence there is some larger $t > s$ where this change is reflected in $W_{k_e,t}$. Wait for $p_e(t)$ and $V^e_{p_e(t)}$ to converge and the latter to agree with $A$. Wait for $g_e(k_e) > a_e$. Go back to state (1).

**Construction**: At $s = 0$ initialize every $R_e$. $R_e$ is said to be inactive at $s$, if there is some $k < s$ such that $b_{k,s}(k) \downarrow$ and the number of mind changes in $g_{k_e}(k_e,s)$ up till $s$ is larger than $b_{k,s}(k)$. Otherwise we say $R_e$ is active. At stage $s > 0$, we search for the least $e < s$ where $R_e$ is active and requires attention, defined as follows: If $R_e$ is in state $0$ then either $k_e \uparrow$ or $b_e(k_e) \downarrow$. If $R_e$ is in state 1, then either $a^e_s \leq g_e(k_e,s)$, or $V^e_t$ converges and agrees with $A$ for some $t > a^e_s$. If $R_e$ is in state 2, then both conditions below hold:

1. **(C1)** $p_e(t) \downarrow$ and $V^e_{p_e(t)}$ converges and agrees with $A$, where $t$ is the least number such that $W_{k_e,t}$ is currently correct, and
2. **(C2)** $g_e(k_e,s) \geq a^e_s$.

Here we say that $W_{k_e,t}$ is currently correct if it agrees (at $s$) with the enumeration we are building for it. If no $e < s$ requires attention, do nothing and go directly to the coding phase. Otherwise pick the least $e$, and attend to $R_e$ by the following.

Suppose $R_e$ is in state 0. If $k_e \uparrow$, pick a fresh value for it. Otherwise set the $R_e$-state to 1 and go to the coding phase. If $R_e$ is in state 1 and $a^e_s \leq g_e(k_e,s)$, we lift $a^e_s$ to a fresh number, and initialize every $R_j$ for $j > e$. Otherwise we enumerate every number $< a^e_s$ into $W_{k_e}$, and declare that $R_e$ is in state 2. Go to the coding phase. If $R_e$ is in state 2 then both the conditions above hold and we declare that $R_e$ is back in state 1. Go to the coding phase.

**Coding phase**: Let $k \in \emptyset'_{at,s}$ (we fix a 1-1 enumeration of the Halting problem). Lift $a^e_k$ to a fresh number, and initialize $R_j$ for every $j > k$.

**Verification**: It is straightforward to verify that each $R_e$ is initialized finitarily often, and is attended to finitarily often. Hence $\lim a^e_s < \infty$ for each $e$, and so $\emptyset' \leq_T A$. Note that there is no computable bound on the number of times each $a_e$ is lifted. Indeed this is necessary by Theorem 3.3. Now we show each $R_e$ is met. Suppose that $\{V^e_s\}_{s<\omega}$, $p_e$ and $b_e$ are total, $g_e$ is $\omega$-c.e. via $b_e$, and $\cup_s V^e_s = A$. Let $k$ be the final value of $k_e$. Since $k$ was picked fresh, the only numbers that are in $W_k$ are those enumerated for the sake of $R_e$. The final $R_e$-state clearly cannot be 0 or 1. Hence the final $R_e$ state is 2, and once $R_e$ enters its final state 2, say at stage $s'$, it will never enumerate anything new into $W_k$. We claim that condition (C1) above will eventually be met: At
strong promptness is closed under wtt-reducibility. We next consider the appropriate analogue of a some enumeration
and $x$ A a cupping notion (see Corollary 3.4). For completeness, we give a very brief sketch of a direct proof:

Proof. (i) follows directly from the characterization of strong promptness in Theorem 3.3 in terms of
Proposition 2.4. Let $s$ we have $A \upharpoonright a_e = A_{s'} \upharpoonright a_e = V^c_{s'} \upharpoonright a_e$ for every
time $t' > t$, from which the claim immediately follows. Hence $g_e(k) = \lim_{u} g_e(k, u) < \lim_{u} a^u_e = |W_k|$. It
remains to verify that there is no $x, s$ such that $x \in W_k, at s$ and $V^c_s \upharpoonright x \neq V^c_{p_e(s)} \upharpoonright x$.

Suppose for a contradiction that $x$ and $s$ witnesses the above. $x$ has to be enumerated into $W_k$
during the construction, say at $s_0$. Note that $s_0$ is no earlier than the first state 1 stage since the
last initialization. Let $t$ be the least such that $W_{k,t}$ is correct at $s_0$. Then $t \geq s > s_0$, and let $u$ be a stage in the construction where $p_e(t) \downarrow$ and $V^c_{p_e(t)}$ converges and agrees with $A$. This number $u$ exists because: (i) If $R_e$ is ever attended to after $s_0$, then $u$ is the first such stage. (ii) If $R_e$ is
never attended to after $s_0$ then we can choose $u$ large enough so that (C1) holds, similar to the
argument in the preceding paragraph.

By convention $u > p_e(t)$ and furthermore we did not attend to $R_e$ between $s_0$ and $u$. Since
$x < a^{s_0}_e$ it follows that $A_{s_0} \upharpoonright x = A_u \upharpoonright x$. At $u$ we had $V^c_{p_e(t)} \upharpoonright x = A_u \upharpoonright x$ and at stage $s_0$ by the
construction, we had $V^c_s \upharpoonright x = A_{s_0} \upharpoonright x$ for some $t'< s_0 < t$. This contradicts the assumption that
$x$ and $s$ is counterexample to the failure of $R_e$. \hfill \Box

Even though strong promptness is not a Turing degree notion, the next proposition shows that
strong promptness is closed under wtt-reducibility. We next consider the appropriate analogue of a
promptly simple set: Recall that a set $B$ is strongly promptly simple if it satisfies Equation (2) for
some enumeration $\{B_s\}_{s<\omega}$ of $B$, a computable $p$ and some $g \leq_{wtt} B'$. We show that every strongly
prompt set is wtt-equivalent to a strongly promptly simple set.

**Proposition 2.4.** Let $A$ and $B$ be c.e. sets.

(i) If $A \leq_{wtt} B$ and $A$ is strongly prompt, then $B$ is also strongly prompt.

(ii) If $A$ is strongly prompt then $A \equiv_{wtt} B$ where $B$ is strongly promptly simple.

**Proof.** (i) follows directly from the characterization of strong promptness in Theorem 3.3 in terms of
a cupping notion (see Corollary 3.4). For completeness, we give a very brief sketch of a direct proof:
Suppose $A = \Gamma^B$ where $\gamma$ is computable, and $A$ is strongly prompt via $p$ and $g$. We will enumerate
auxiliary c.e. sets $\{W_{r(x)}\}$, and show that $B$ is strongly prompt via some $\bar{g}(x) > \gamma(g(r(x)))$. To
meet a single instance (for some $e$) of Equation (1) we need to do the following. Each time we need
to change $B$ below $\bar{g}(e,s) > \gamma(g(r(e)))|s|$, we increase the cardinality of $W_{r(e)}$ beyond $g(r(e), s)$.
Since $A$ is strongly prompt, this means that either $g(r(e), s) = \gamma(g(r(e), t))$ for some $t > s$, or else
$A$ has to change below $\max W_{r(e)}$ (which means $B$ has to change below $\bar{g}(e,s)$, since $\gamma$ does not
depend on $s$). If the latter holds, then we consider $B$ to have permitted promptly, otherwise we
increase $\bar{g}(e) > \max \{g(r(e), t), \max W_{r(e)}\}$.

(ii) Suppose $A$ is strongly prompt via $\{A_s\}_{s<\omega}, p$ and $g$. Let the computable function $r$ be such
that $W_{r(e)} = W_e \cap \{x : x > 3e\}$. We assume as before that if $x \in W_e, at s$ then $x \notin W_{r(e),s}$. We
build a computable enumeration $\{E_s\}_{s<\omega}$ for $E \equiv_{wtt} A$ satisfying for each $e$,

$$R_e : |W_e| \geq g(r(e)) + 3e \rightarrow (\exists x\exists s)[x \in W_e, at s \cap E_s].$$

At stage $s$ of the construction, check if the following holds for each $e < s$:

(i) The conclusion in $R_e$ fails, and
(ii) There is some $x > 3e$ where $x \in W_e, at s$, and $A_t \upharpoonright x \neq A_{p(t)} \upharpoonright x$ where $t > s$ is the stage
where $x \in W_{r(e), at t}$.

If (i) and (ii) holds for $e$, we enumerate $x$ into $E_s$. Also if $y \in A_{at s}$, we enumerate $e^s_{3y}$ into $E_s$,
where $E_s = \{e^s_0 < e^s_1 < \cdots\}$. 

Clearly $E \leq_{wtt} A$ via the identity use. At most $n$ elements less than $3n$ enters $E$, due to $R$-requirements, and another $n$ due to the coding of $A$. Hence $|E \cap \{x : x < 3n\}| \leq 2n$ for every $n$, hence $c^e_n \leq 3n$ for every $s, n$. Hence $A \leq_{wtt} E$, with use bounded by $9n$. $R_e$ is satisfied for each $e$, because if $|W_e| \geq g(r(e)) + 3e$ then $|W_{r(e)}| \geq g(r(e))$.

3. The analogues of promptness and cupping

In this section we investigate the analogue of the following result of Ambos-Spies, Jockusch, Shore and Soare [1]: A set is low cuppable iff it is of promptly simple degree. In particular we will investigate the interaction between strong promptness and superlow cuppability. We begin by showing that every strongly prompt set is superlow cuppable.

**Theorem 3.1.** If $B$ is strongly prompt, then $B$ is superlow cuppable.

**Proof.** Given a strongly prompt set $B$, we will construct a superlow set $A$ which cups with $B$. During the construction, we will also (uniformly) define a sequence of c.e. sets $U_e$. Again by the recursion theorem and the slowdown lemma, there is a computable function $q$ such that for all $e$, $W_{q(e)} = U_e$, and every element enumerated into $U_e$ appears strictly later in $W_{q(e)}$. Let $p$ and $g$ be as in equation 1, and let $g$ be $\omega$-c.e. via the approximation $\{g_s\}_{s<\omega}$. Moreover, let $h$ be a computable bound on the number of mind changes made by the approximation $\{g_s\}_{s<\omega}$. Let $\vartheta_e^s$ be an enumeration of $\vartheta_e$. In order to make $A \oplus B$ complete, we will keep track of a list of “coding markers” $\Gamma_n$, and let $\Gamma_n^s$ denote the position of the coding marker at stage $s$. During the construction, we will ensure the following:

1. If $n \in \vartheta^s_{s+1} \setminus \vartheta^s_s$, then $A_s \oplus B_s \upharpoonright \Gamma_n^s + 1 \neq A \oplus B \upharpoonright \Gamma_n^s + 1$
2. For all $n$ and $s$, we have $\Gamma_n^s \leq \Gamma_{n+1}^s$ and $\Gamma_n^s < \Gamma_{n+1}^s$
3. For all $n$ and $s$, we have $\Gamma_n^s \geq 2g_s(q(n))$
4. If $\Gamma_n^s < \Gamma_{n+1}^s$, then $A_s \oplus B_s \upharpoonright \Gamma_n^s + 1 \neq A \oplus B \upharpoonright \Gamma_n^s + 1$
5. For all $n$, the sequence $(\Gamma_n^s)_s$ is bounded.

These conditions suffice to guarantee that $\vartheta_e^s \leq T A \oplus B$, which will be demonstrated in the verification. For convenience, we will additionally ensure that $\Gamma_n^s$ is even, so that $A_s \oplus B_s \upharpoonright \Gamma_n^s$ depends on the same number of places of $A$ and $B$, which is $\frac{1}{2}\Gamma_n^s$.

In order to ensure $A'$ is $\omega$-c.e., we will make use of the fact that $B$ is strongly prompt to occasionally “clear” the use of computations $\Phi^A_e(e)$ of all markers $\Gamma_n^s$ for $n > e$. We do this by attempting to use the promptness of $B$ to force $B \upharpoonright \Gamma_n^s$ to change so that we can move the markers (and still follow the fourth requirement above). To show that $A'$ is $\omega$-c.e., we will exhibit a computable approximation $\lim_s f_s = A'$, with computably bounded mind-changes.

Construction of $A$.

**Stage $s = 0$:** Set $A_0 = \emptyset$. For all $n \in \omega$, set

$$\Gamma_0^0 = 2n + 2 \sup_{i \leq n} g_0(q(i)).$$

**Stage $s + 1$:**

- **Step 1.** Find the least $e$ (not marked unavailable) such that $\Phi^A_e(e)[s]$ converges, and

$$\frac{1}{2}\Gamma_e^s \leq \varphi^A_e(e)[s]$$
We say that $e$ acts at stage $s + 1$. Enumerate $0, 1, 2, ..., g_s(q(e))$ in $U_e$. Let $t$ be the least stage such that $0, 1, 2, ..., g_s(q(e)) \in W_{q(e)}$. Note that since $q$ was obtained from the slowdown lemma, $t > s$. During the verification, it will be shown that either

$$B_s \upharpoonright \frac{1}{2} \Gamma^s_e \neq B_{p(t)} \upharpoonright \frac{1}{2} \Gamma^s_e$$

or there is a least $t' > s$ such that $g_{t'}(q(e)) > g_s(q(e))$.

- Case 1: free clear. If $B_s \upharpoonright \frac{1}{2} \Gamma^s_e \neq B_{p(t)} \upharpoonright \frac{1}{2} \Gamma^s_e$, move all markers $\Gamma_m, m \geq e$ to new even positions which are not double of a number in $A$ and greater than their old positions, $2\varphi^A_e(e)[s]$, and $2g_s(q(i))$, preserving their order. We say that $e$ has a free clear.

- Case 2: capricious destruction. If case 1 fails to hold, let $t' > s$ be minimal such that $g_{t'}(q(e)) > g_s(q(e))$. Enumerate $\frac{1}{2} \Gamma^s_e$ into $A$. Move all markers $\Gamma_m, m \geq e$ to new even positions which are not yet in $A$ and greater than both their old positions and $2g_{t'}(q(e))$, preserving their order. Mark $e$ as unavailable until stage $t'$.

- Step 2. For each $n \in \emptyset_{s+1} \setminus \emptyset_s$, enumerate $\frac{1}{2}$ the current position of $\Gamma_n$ into $A$, and move all markers $\Gamma_m, m \geq n$, to new even positions which are double of a number in $A$ and greater than both their old positions and $2g_{s+1}(n)$, preserving their order.

- Step 3. For each $n \leq s$ with $g_{s+1}(q(n)) > g_s(q(n))$, enumerate $\frac{1}{2}$ the current position of $\Gamma_n$ into $A$, and move all markers $\Gamma_m, m \geq n$, to new even positions which are not yet in $A$ and greater than both their old positions and $2g_{s+1}(q(n))$, preserving their order.

- Step 4. For each $e$ which has a free clear, if $A\upharpoonright \varphi^A_e(e)[s + 1] \neq A\upharpoonright \varphi^A_e(e)[s]$, cancel $U_e$ and replace it with a new set (which will again be called $U_e$, and all references to $U_e$ will apply to the new set rather than the old set).

This ends the construction. We remark that in Step 4, when we say that we cancel $U_e$, we really mean that we move on to another $\tilde{U}_e$ with a different $\Sigma^0_1$-index. Since each $e$ will only need to cancel $U_e$ a computably bounded number of times, we simply recycle the notation $U_e$, $g(e)$ and $h(e)$.

Verification. We need to prove I) The construction is well defined; in particular, in step 1, if case 1 fails to hold, then there is some $t' > s$ such that $g_{t'}(q(e)) > g_s(q(e))$. II) During the construction, the positions of the markers $\Gamma_n$ satisfy properties (1)–(4) above. III) For all $n$, the sequence $(\Gamma^s_n)_n$ is bounded (property (5) above). IV) The join $A \oplus B$ is complete. V) The set $A$ is superlow, i.e. $A'$ is $\omega$-c.e.

We will start by proving that I) and II) hold by simultaneous induction on the stage number $s$; first we prove that the construction has well-defined behavior at stage $s + 1$ provided that II) holds through stage $s$, and then we prove that II) holds through stage $s + 1$ if the construction has well-defined behavior at that stage.

Proof of I). Suppose that in step 1, case 1 fails to hold, and furthermore, for all $t' > s$, we have $g_{t'}(q(e)) \leq g_s(q(e))$. Then the limit $g(q(e)) = \lim_s g_s(q(e)) \leq g_{t'}(q(e))$. Furthermore, we know that this stage of the construction is never completed, since in step 1, case 2, the construction searches for a $t'$ that doesn’t exist, which means that $U_e$ is exactly that portion of $U_e$ which was constructed up through stage $s + 1$, step 1, i.e. $\{0, 1, 2, ..., g_s(q(e))\}$. Since

$$|W_{q(e)}| - |U_e| = g_s(q(e)) + 1 > g_s(q(e)) \geq g(q(e)),$$
by the definition of strongly prompt (cf. equation 1), there must be some \( x, s' \) with
\[
(7) \quad x \in W_{q(e)}, \text{ at } s' \land B_{s'}| x \neq B_{p(s')}| x.
\]

By the slowdown lemma, the elements of \( W_{q(e)} \) are all enumerated in \( W_{q(e)} \) after being enumerated in \( U_e \); that is, after stage \( s + 1 \). Moreover, by choice of \( t \), they have all been enumerated by stage \( t \), so \( s < s' \leq t \). Furthermore, the \( x \) in question is an element of \( U_e \), so is at most \( g_e(q(e)) \). By induction, we may assume that property (3) holds at stage \( s \); that is, we have \( \Gamma^s_e \geq 2g_e(q(e)) \), so \( x \leq \frac{1}{2}\Gamma^s_e \). Therefore, \( B_{s'}| \frac{1}{2}\Gamma^s_e \neq B_{p(s')}| \frac{1}{2}\Gamma^s_e \), and since \( s < s' < p(s') \leq p(t) \), we must have
\[
(8) \quad B_{s'}| \frac{1}{2}\Gamma^s_e \neq B_{p(t)}| \frac{1}{2}\Gamma^s_e,
\]
which contradicts the fact that case 1 failed to hold. Hence the construction is well-defined.

Proof of II). We want to demonstrate that properties (1)–(4) hold. Property (1) was ensured in step 2 of the construction. Property (2) holds at stage 0, and in every step where the markers \( \Gamma_n \) are moved, we ensure that (2) continues to hold. Property (3) was ensured in step 3 of the construction. For (4), note that while the markers may be moved in any of step 1 case 2, step 2, or step 3, we always enumerate some element into \( A \) which ensures that (4) continues to hold in any of those cases. The only time we move some marker(s) without enumerate an element into \( A \) permitting the move is in step 1 case 1, free clear, but in this case,
\[
(9) \quad B_{s'}| \frac{1}{2}\Gamma^s_e \neq B_{p(t)}| \frac{1}{2}\Gamma^s_e
\]
so that a \( B \) change permits the move.

Proof of III). We want to show that the sequence of locations of the \( n \)th movable marker \( \Gamma_n \) is bounded. By examining the construction, we see that \( \Gamma_n \) is moved only during steps 1, 2, and 3. The marker \( \Gamma_n \) is moved during step 1 at a stage when the active requirement \( e \) is at most \( n \); during step 2 when an element at most \( n \) is enumerated into \( \emptyset \); and during step 3 when the approximation \( g_{n+1}(q(m)) \) has just made a change, for some \( m \leq n \). The step 2 change can clearly occur at most \( n + 1 \) times. The step 2 change can occur at most \( \sum_{m=0}^{n} h(q(m)) \) times, since \( h(q(m)) \) bounds the number of mind-changes of the approximation \( (g_s) \) on argument \( (q(m)) \). Finally, the step 1 change occurs when a requirement \( e \leq n \) acts.

If \( e \) has a free clear in step 1, and later the marker \( \Gamma_e \) is moved, we will say that the free clear on \( e \) was canceled. The marker \( \Gamma_e \) is moved any time an element \( x \leq \frac{1}{2}\Gamma^s_e \) is enumerated into \( A \) at a stage \( s \), so if requirement \( e \) has a free clear at stage \( t \), then \( A \) must not have changed below \( \Gamma^t_e \) since stage \( t \). At stage \( t \), we had \( \frac{1}{2}\Gamma^t_e > \varphi^t_e(e)[t] \), and neither side can have changed without the free clear being canceled, requirement \( e \) will not act again unless the free clear is canceled. Thus the total number of times requirement \( e \) acts is the number of times requirement \( e \) acts with capricious destruction, plus the number of times requirement \( e \) acts, achieving a free clear which is later canceled, possibly plus one if requirement \( e \) acts a final time and achieves a free clear which is never canceled. Each capricious destruction corresponds to a change of \( g_s(q(e)) \), so there are at most \( h(q(e)) \) of those. Each canceled free clear corresponds to either an occurrence in step 2 or 3 moving \( \Gamma_e \), of which there are at most \( \sum_{m=0}^{e} h(q(m)) + e + 1 \), as we accounted in the previous paragraph, or else an action of a requirement of greater priority. Thus the number of times requirement \( e \) can act is at most \( h(q(e)) + \sum_{m=0}^{e} h(q(m)) + e + 2 \) plus the number of times all requirements of greater priority can act. One can apply this formula recursively to bound the
number of times requirements \( e \leq n \) can act, and then use that to bound the number of times the marker \( \Gamma_n \) is moved during the construction. (In fact, the bound obtained will be computable.) Thus the sequence of locations of \( \Gamma_n \) is bounded, and in fact \( \lim_s \Gamma_n^s \) is \( \omega\text{-c.e.} \)

Proof of IV). We want to show that \( \emptyset' \leq_T A \oplus B \). Let \( \gamma \) be defined by \( \gamma(n) = \lim_n \Gamma_n^s \), which we know must exist from III). We must have \( \gamma \leq_T A \oplus B \), since (by properties (4) and (5)) every change to \( \Gamma_n^s \) is permitted by a change in \( A \oplus B \). For each \( n \), to compute whether \( n \in \emptyset' \), we need only search for an \( \epsilon \) such that

\[
\epsilon > s \implies \Gamma_{\epsilon+1} \not= \Gamma_{\epsilon},
\]

Thus the sequence of locations of \( \Gamma \) will remain so unless a new element at most \( \phi \) for some \( \epsilon \), which wants to act at stage \( \epsilon + 1 \), the construction will never act to change it. Since the least \( \epsilon \) is superlow. (In fact, the bound obtained will be computable.)

Then, by (1), \( n \in \emptyset' \) if and only if \( n \in \emptyset'_{\emptyset} \).

Proof of V). We want to show that \( A' \) is \( \omega\text{-c.e.} \). We define the following approximation:

\[
f_s(x) = \begin{cases} 1 & \text{if } \Phi_{e,s} A(e) \downarrow \text{ and } \varphi_{e,s} A(e) < \frac{1}{2} \Gamma_e^s \\ 0 & \text{otherwise} \end{cases}
\]

First, we observe that since the markers only move to the right, if \( \Phi_{e,s} A(e) \downarrow \) and \( \varphi_{e,s} A(e) \leq \frac{1}{2} \Gamma_e^s \), it will remain so unless a new element at most \( \varphi_{e,s} A(e) \) is enumerated into \( A \). This can occur only when there is a capricious destruction on some \( i < e \), when some \( g_s(i) \neq g_{s+1}(i) \) for some \( i < e \), or when some \( i < e \) is enumerated into \( \emptyset' \). Capricious destructions occur only when \( g_s(i) \neq g_{t'}(i) \) for some \( t' > s \) and cannot occur again for the same \( i \) until stage \( t' \). Additionally, remember that \( g_s(i) \) can change at most \( h(i) \) times. So the number of times the approximation \( f_s(e) \) can go from a 1 to a 0 is at most

\[
e + 2 \sum_{i=0}^{e-1} h(i),
\]

resulting in a total number of changes which is at most

\[
2e + 1 + 4 \sum_{i=0}^{e-1} h(i).
\]

Hence there is a computable bound on the number of changes made by this approximation.

It remains to show only that \( \lim_s f_s = A' \). Suppose \( e \) is minimal such that \( \lim_s f_s(e) = 0 \) and \( \Phi_{e,s} A(e) \downarrow \) infinitely often. Then there is some \( S \) so that \( \emptyset'_{\emptyset} \models e = \emptyset'_{\emptyset} \downarrow \epsilon \) and if \( i < e \) and \( s > S \), then \( \Phi_{e,s} A(e) \downarrow \) or \( \varphi_{e,s} A(i) < \frac{1}{2} \Gamma_e^s \), meaning that \( i \) does not want to act at stage \( s + 1 \). However, infinitely often \( \Phi_{e,s} A(e) \downarrow \) and \( \varphi_{i,s} A(i) \geq \frac{1}{2} \Gamma_e^s \), so infinitely often \( e \) wants to act. Since the least \( n \) which wants to act at stage \( s \) always acts, \( e \) must act infinitely often. There can be at most finitely many capricious destructions, so we have an \( s > S \) so that \( e \) acts at stage \( s \) and receives a free clear. After stage \( S + 1 \), the construction will never act to change \( A \), \( \varphi_{e,s} A_{S+1}(e) \), and \( \frac{1}{2} \Gamma_e^{S+1} \) will be greater than \( \varphi_{e,s} A_{S+1}(e) \), which means that \( f_s(e) = 1 \) for all \( s > S + 1 \), contradicting \( \lim_s f_s(e) = 0 \).

Thus if \( \lim_s f_s(e) = 0 \), then \( \Phi_{e,s} A(e) \downarrow \) infinitely often, of \( A'(e) = 0 = \lim_s f_s(e) \). If \( \lim_s f_s(e) = 1 \), then infinitely often \( \Phi_{e,s} A(e) \downarrow \) and \( \varphi_{e,s} A(e) \leq \frac{1}{2} \Gamma_e^s \). Since it remains so unless a new element at most \( \varphi_{e,s} A(e) \) is enumerated into \( A \), which can happen at most finitely often, it must be the case that \( A'(e) = 1 \) as well. Thus \( \lim_s f_s = A' \), and \( A \) is superlow.

\[\square\]
Remark 3.2. This construction guarantees that every strongly prompt set is superlow cuppable, but in fact it guarantees something slightly stronger: that the use of the reduction $\emptyset' \leq_T A \oplus B$ is an $\omega$-c.e. function, because the use of the computation of $\emptyset'(n)$ is $\gamma(n) = \lim_s \Gamma^s_n$, and we showed in the proof of III) that this is $\omega$-c.e.

The lack of closure of strong promptness under Turing reducibility is somewhat unfortunate, since it immediately forbids this notion from characterizing the sets which are superlow cuppable. However we are still able to characterize the strongly prompt sets in terms of a cupping notion: They are the c.e. sets which are superlow cuppable with $\omega$-c.e. use. Note that the latter cupping property is wtt-degree invariant.

Theorem 3.3. A c.e. set $A$ is strongly prompt iff there is a superlow c.e. set $B$ which cups with $A$ to $\emptyset'$ via an $\omega$-c.e. bound on the use. That is, there is a reduction $\emptyset' \leq_T A \oplus B$ where the use is bounded by an $\omega$-c.e. function.

Proof. By Remark 3.2 we only need to prove the “if” direction. Suppose $A$ and $B$ are c.e. sets with enumerations $\{A_s\}$ and $\{B_s\}$, such that $F = \Gamma^{A \oplus B}$ and $\gamma(e) \leq \lim_s u(e, s)$ with fewer than $h_u(e)$ many mind changes. Here $F$ is a c.e. set we build during the construction. We note that since the c.e. sets are uniformly reducible to $\emptyset'$, the recursion theorem will give us $\Gamma, u$ and $h_u$. We may assume (by speeding up the relevant approximations) that $F_i(f) = \Gamma^{A \oplus B}(f)[t]$ holds for every $f, t$, and that $\gamma_i(f) < u(f, t)$. If we enumerate $f$ into $F$ at $t$, so that $\Gamma^{A \oplus B}(f)[t] = 0$ is now incorrect, we assume that recovery occurs at $t + 1$.

We also use the recursion theorem to obtain a computable sequence of indices $v$ for which we control $J^B(v) := \Phi_v^B(v)$. To elaborate, we build a sequence of Turing functionals $\Phi_{v_1}, \Phi_{v_2}, \cdots$ and by the recursion theorem the function $\lambda iv_i$ is computable. Into each of these functionals we will enumerate computations with certain use (in the construction the use will be current initial segments of $B_s$). This allows us to control, say $J^B(v)[s]$, where we will make $J^B(v)[s] \downarrow$ by enumerating an axiom of the form $B_s \downarrow u$ for some $u$. We think of $J^B(v)$ as convergent as long as $B$ does not change below $u$. If $B_t \downarrow u \neq B_s \downarrow u$ at some future stage $t > s$, then $J^B(v)[t] \uparrow$ and (since $B$ is c.e.) stays divergent until we choose to enumerate a new computation for it.

Since $B$ is superlow, the eventual convergence or divergence of $J^B(v)$ is approximated by $b(v, s)$ with fewer than $h_b(v)$ many mind changes ($b(v, s) = 0$ means it is guessing divergence). To simplify notation we drop $v$ and assume that we control $J^B(0), J^B(1), J^B(2), \cdots$. This is permitted since $v_i$ is computable and everything mentioned in the construction gets transformed in a computable way. We partition $\omega$ into infinitely many intervals, where $I_n$ has length $h_b(n)$. To show that $A$ is strongly prompt, we construct another approximation $\{U_s\}_{s \in \omega}$ for $A$, and show that $A$ is strongly prompt via $\{U_s\}_{s \in \omega}$ and the identity function as the prompt function. That is, we will ensure that $\{U_s\}_{s \in \omega}$ is a strong array such that $\cup_s U_s = A$, and also build an $\omega$-c.e. function $g(e)$ with mind-change bound given by $h_g(e) = h_b(e) + \sum \{h_u(f) : f \in I_e\}$. We also ensure that $R_e : \text{Either max } W_e \leq g(e), \text{ or } \exists x \exists y (x \in W_{e \circ d s} \land U_s \uparrow x \neq U_{s+1} \uparrow x)$.

We now describe the actions of $R_e$. The actions of different $R_e$ do not interfere directly with one another. Each $R_e$ uses $J^B(e)$ to challenge the lowness of $B$, and will only enumerate numbers into $F$ from the partition $I_e$. $R_e$ has a parameter $f_e$, which is a number picked from the interval $I_e$ and targeted for $F$. At each stage $s$ of the construction we look at numbers $x$ and $e$ where $x > g(e, s)$ has entered $W_e$ at $s$, and we want to respond by enumerating some number $z < x$ into $U_{s+1}$. Of course to ensure that $\cup_s U_s = A$, we can only do this if we know $z \in A_t$ for some $t > s$. To force
this to happen, we run a procedure called an $\mathcal{R}_e$-challenge. This procedure begins by enumerating a jump computation (with $B$-use $\gamma(f_e)$) to make $J^B(e) \downarrow$. Assume that $g(e)$ is currently larger than $u(f_e)$. One of the following three things has to happen at some $t > s$:

(i) $A \uparrow \gamma(f_e)$ changes,
(ii) $B \uparrow \gamma(f_e)$ changes, or
(iii) $b(e, t) = 1$.

Let $t$ be the first stage found where one of (i)-(iii) holds. If (i) applies then we can safely enumerate some $z < \gamma(f_e)$ into $U_{s+1}$, since we know that $z \in A$. In this case $\mathcal{R}_e$ is satisfied (and remains satisfied forever). If (ii) holds then we check if $u(f_e, t)$ has increased beyond $g(e, s)$. If it has, then $u(f_e)$ has used up one more mind change. In this case we increase $g(e, s+1)$ to a fresh value well beyond $u(f_e, t)$, and end the challenge (note that in this case $J^B(e)[t] \downarrow$). On the other hand if $u(f_e, t)$ has not increased, then we define $J^B(e)[t] \downarrow$ with the new $B \uparrow \gamma(f_e)$-configuration and wait for a further $t' > t$ where (i), (ii) or (iii) happens. Finally if (iii) holds then we enumerate the agitator $f_e$ into $F$ and wait for a further $t' > t$ such that either (i) or (ii) holds at $t'$. Again if (i) holds then $\mathcal{R}_e$ is satisfied forever. Otherwise if (ii) holds then we end the challenge with $J^B(e)[t'] \downarrow$. We will also have made progress since $b(e)$ has used up one more change.

It is clear that once the $\mathcal{R}_e$-challenge is started, there are only one of three outcomes. The first outcome is that $A$ changes at some $t > s$ allowing us to satisfy $\mathcal{R}_e$ forever. The second is that the challenge is ended with $u(f_e)$ having used up one more mind change. The last possibility is that the challenge is ended with $b(e)$ having used up one more mind change. In the latter two cases when the challenge is ended we have $J^B(e)[t] \downarrow$ - this is necessary to run the next $\mathcal{R}_e$-challenge at some later stage of the construction. Also note that if (ii) holds with no movement of $u(f_e)$, then we will keep repeating the challenge until either (i) or (iii) happens, or $u(f_e)$ moves. Since we only move $g(e)$ when we find (iii) holds or $u(f_e)$ moves, it is easy to see that the number of $g(e, -)$ mind changes will be bounded by $h_g(e)$. To help organize the construction, we keep a global parameter called $M$. The intention is for $M_s$ to record during stage $s$ of the construction how far ahead we have looked. In the above example $M$ will be increased to the largest number $(t$ or $t')$ searched during the $\mathcal{R}_e$-challenge. Any other $\mathcal{R}_j$-challenge begun later will limit the scope of their search to numbers larger than $M$. This helps to prevent double accounting of mind changes. During the formal construction, several challenges may be run (sequentially) in a single stage. Each challenge may increase $M$, so a challenge started later will use the updated value for $M$.

Formal construction. At stage 0 we set $M = 0$ and $U_1 = \emptyset$. We set $f_s = \min I_e$ and $g(e, 0) = u(f_s, 0) + 1$ for every $e$. At stage $s > 0$, we say that $\mathcal{R}_e$ is satisfied at $s$ if $\exists x \exists t < s (x \in W_{e,s+1} \land U_t \uparrow x \neq U_{t+1} \uparrow x)$. For each $e < s$ such that $\mathcal{R}_e$ is not yet satisfied, and there is some $x > g(e, s)$ and $x \in W_{e,s}$, at $s$, we run the $\mathcal{R}_e$-challenge by the following.

$\mathcal{R}_e$-challenge: Find $t_0 > M$ such that $b(e, t_0) = 0$. If $u(f_e, t_0) \geq g(e, s)$ then we set $g(e, s+1)$ to a fresh value and end the challenge. Otherwise run the following cycle with $n$, starting with $n = 0$: Enumerate $J^B(e)[t_n] \downarrow$ with use $\gamma(f_e)[t_n]$. Search for the first $t_{n+1} > t_n$ where (i), (ii) or (iii) above holds. If (i) happens then some $z$ enters $A \uparrow \gamma(f_e)$ at $t_{n+1}$. Enumerate $z$ into $U_{s+1}$, and end the $\mathcal{R}_e$-challenge. If (ii) holds then we check if $u(f_e, t_{n+1}) \geq g(e, s)$. If so we set $g(e, s+1)$ to a fresh value and end the challenge. If not then restart the cycle again with $n+1$. If (iii) holds then $b(e, t_{n+1}) = 1$ and we enumerate $f_e$ into $F$. At $t_{n+1} + 1$ either $A \uparrow \gamma_{t_n}(f_e)$ or $B \uparrow \gamma_{t_n}(f_e)$ will be different. In the former case we enumerate $z$ into $U_{s+1}$ where $z$ enters $A \uparrow \gamma(f_e)$ at $t_{n+1} + 1$, and end the challenge. In the latter case we let $f_e$ be the next unused number in $I_e$, move $g(e, s+1)$ to
a fresh value and end the challenge. Finally, regardless of how the $R_e$-challenge was ended, we set $M = 1 + t_{n+1}$ where $n$ was the last cycle run.

At a single stage several challenges may be run (for different $R$). Before concluding the stage we need to ensure that $U_{s+1}$ is up to date: For every $z \in A_M$ we enumerate $z$ into $U_{s+1}$. This concludes the description of the construction.

Verification: It is clear that $\cup U_s = A$. Fix $e$, and we prove by induction the following claim: For an $R_e$-challenge run at stage $s$,

(i) It will eventually be completed.

(ii) Suppose that at least one cycle is run, and $k$ is the final cycle run. If the outcome of cycle $k$ is an $A$-change, then $R_e$ is satisfied and will remain satisfied forever.

(iii) If the outcome of cycle $k$ is not an $A$-change (or if no cycle was run), then when the challenge is ended $J^B(e)[t_{k+1} + 1] \uparrow$ and $g(e, s + 1)$ is picked fresh. In this case either $u(f_e, t_{k+1} + 1) \neq u(f_e, m')$, or $b(e, t_{k+1} + 1) \neq b(e, m')$, where $m'$ is the value of $M$ when the previous $R_e$-challenge was concluded.

Suppose an $R_e$-challenge is started at stage $s$. Then the outcome of the previous challenge is not an $A$-change, and $J^B(e)[M] \uparrow$. Hence $t_0 > M$ will be found. It is clear that we will never get stuck forever in an $n$-cycle (searching for $t_{n+1} > t_n$). Hence the only way for the challenge not to be completed, is to run infinitely many cycles. Hence every $n$-cycle must finish with a $B$-change and $u(f_e, t_{n+1}) < g(e, s)$. Since $\gamma(f_e) < g(e, s)$ this means that once $B \uparrow g(e, s)$ is stable we get a contradiction. This shows (i).

Now suppose that at least one cycle was run in the challenge. Let $k$ be the final cycle run. Hence $\gamma_{t_k}(f_e) < u(f_e, t_k) < g(e, s)$. Suppose the final cycle $k$ finished with an $A$-change. Then some $z < \gamma_{t_k}(f_e) < x$ with $z \in A_{t_{n+1} + 1} - A_{t_{n+1} - 1}$ is found (where $x$ is the element which entered $W_e$ at $s$). Hence $z \in U_{s+1} - U_s$ since $M < t_{n+1} - 1$. Then $R_e$ is satisfied at every stage larger than $s$, showing (ii). If no cycle was run in the challenge then clearly $J^B(e)[t_0 + 1] \uparrow$ and $g(e, s + 1)$ was given a fresh value when the challenge was ended. In this case we clearly have $u(f_e, t_0 + 1) > u(f_e, m')$. Suppose at least one cycle was run, and the outcome of cycle $k$ was not an $A$-change. At cycle $k$ we can only have either a $B \uparrow \gamma_{t_k}(f_e)$ change at $t_{k+1}$, or $b(e, t_{k+1} + 1) = 1$ followed by a $B \uparrow \gamma_{t_k}(f_e)$ change at $t_{k+1} + 1$. In the former case we have $u(f_e, t_{k+1}) \geq g(e, s) > u(f_e, t_0)$ and in the latter case we have $b(e, t_{k+1} + 1) \neq b(e, t_0)$. In either case we have $J^B(e)[t_{k+1} + 1] \uparrow$ and $g(e, s + 1)$ chosen fresh. This proves (iii).

It is easy to check (using (iii) of the claim) that $f_e$ can always be picked from $I_e$. It also follows easily from the claim that the number of mind changes we make to $g(e, \_)$ is bounded by $h_g(e)$. It remains to verify that $R_e$ is satisfied. Suppose for a contradiction that there is some $x \in W_{e, \uparrow s}$ and $x \not> \lim g(e, \_)$, and that $R_e$ is never satisfied. By convention $s > e$ so we will run the $R_e$-challenge at $s$. By the claim above this challenge will end with either the satisfaction of $R_e$, or with $g(e, s + 1)$ picked fresh (hence will be larger than $x$). This gives our desired contradiction. 

**Corollary 3.4.** The strongly prompt c.e. sets are closed upwards under wtt-reducibility.

We say that a Turing degree (or a wtt-degree) $a$ is strongly prompt if it contains a strongly prompt c.e. set. It is worth pointing out the following facts about the strongly prompt wtt-degrees:

**Proposition 3.5.** The strongly prompt wtt-degrees form a strong filter in the c.e. wtt-degrees. However their complement do not form an ideal in the c.e. wtt-degrees.
Proof (Sketch). To prove the first statement, suppose we are given c.e. sets $A, B$ which are strongly prompt via $\{A_s\}_{s \in \omega}, p, g$ and $\{B_s\}_{s \in \omega}, g, h$ respectively. We want to build a strongly prompt c.e. set $C \leq_{wtt} A, B$ via identity use. We need to satisfy requirements of the form $|W| \geq r \rightarrow (\exists x)(\exists s)[x \in W_{at s} \land C_s \upharpoonright x \neq C_{s+1} \upharpoonright x]$. To meet this requirement we enumerate auxiliary c.e. sets $W_{\ell_0}, W_{\ell_1}, \ldots, W_{\ell_m}$ where $m = \text{bound on the mind changes of } h(\ell_0)$. We begin by testing $A$ with $W_{\ell_1}$ and $B$ with $W_{\ell_0}$. Every time $|W| \geq r_s$, we try to force $A \upharpoonright r_s$ to change by dumping $\{0, \ldots, r_s\}$ into $W_{\ell_1}$. If the $A$-change was denied promptly, then we increase $r$ to match $g(\ell_1)$. If $A$ changes promptly, we proceed to request a prompt $B$-change by dumping $\{0, \ldots, r_s\}$ into $W_{\ell_0}$. If the $B$-change was denied promptly we increase $r$ to match $h(\ell_0)$ and $g(\ell_2)$, and from now on use $W_{\ell_3}$ to test $A$. The number of times this can happen before $A$ and $B$ are simultaneously permitted is at most $m + \sum_{i \leq m} \# \text{ mind changes in } g(\ell_i)$, which is computable.

To prove the second statement in the proposition, it suffices to construct c.e. sets $A$ and $B$ such that $A \cup B = \emptyset$ (that is, $A \cup B = \emptyset'$ and $A \cap B = \emptyset$), and ensure that $A$ and $B$ are not strongly prompt. The strategy to make $A$ and $B$ not strongly prompt was described in Theorem 2.3. The basic strategy for a requirement $\mathcal{R}^A$ (and similarly for $\mathcal{R}^B$) will seek to diagonalize a possible enumeration $\{V_s\}_{s \in \omega}$ of $A$, a (partial) computable function $p$, and an $\omega$-c.e. function $g$. To do this we must ensure that for some $k$, $|W_k| > g(k)$ and everytime we dump elements into $W_{k,s}$ we must also ensure that $A$ (and hence $V$) does not change before $p(s)$ converges. This basic strategy sets up a restraint on $A$ everytime it dumps new elements into $W_k$, and this restraint may be increased finitely many times whenever $g(k)$ increases. On the other hand every time this restraint is breached by an $A$-change (due to higher priority action), it will restart with a new $k$. This can easily be arranged as a finite injury construction. To ensure $A \cup B = \emptyset'$ we proceed as in Sacks’ Splitting Theorem. We remark that the main difference between the $\mathcal{R}$-requirements in this construction and the lowness requirements in the Sacks’ Splitting Theorem, is that each $\mathcal{R}$-requirement may increase its restraint (finitely often) even if $\mathcal{R}$ is not injured. 

It is natural to ask if the analogue of the statement “$A$ is capable $\iff A$ is not of promptly simple degree” holds. Non-prompt simplicity can be expressed as sharing no nontrivial common information with some noncomputable c.e. set. Is there some weaker condition that can be used to characterize strong promptness which talks about the ability of a set $A$ to share nontrivial information with other sets? A reasonable definition has so far been elusive.

Since strong promptness do not characterize the sets which are superlow cuppable, one is motivated to ask if there is still an equivalence up to Turing degrees:

**Question 3.6.** Are the strongly prompt Turing degrees the same as the superlow cuppable degrees?

### 4. RELATIONSHIP WITH OTHER CLASSES

In this section we investigate the connections between strong promptness and different classes exhibiting computational weakness. An important notion here is cost functions, introduced by Nies. The reader should also refer to Nies [14], and Greenberg and Nies [9]. The standard cost function $c_K$ is given by

\[ c_K(x, s) = \sum_{x < w \leq s} 2^{-K(w)}. \]
We say that a computable approximation $\{A_s\}_{s \in \omega}$ obeys a cost function $c$ if the sum
\begin{equation}
S = \sum_{(x,s) \in D} c(x,s)
\end{equation}
is finite, where $D = \{(x,s) : x < s$ and $x$ is least such that $A_{s-1}(x) \neq A_s(x)\}$. If $A$ has an approximation which obeys $c$, we write $A \models c$. It is easy to see that the standard cost function $c_K$ satisfies the limit condition
\begin{equation}
\limsup_{x,s > x} c_K(x,s) = 0.
\end{equation}

We will need the following result on cost function constructions, taken from [14], where it appears as Theorem 5.3.10:

**Lemma 4.1.** If there is a computable approximation $\{X_s\}_{s \in \omega}$ of a set $X$ which obeys the standard cost function, then $X$ is $K$-trivial.

**Definition 4.2.** We say a computable cost-function $c$ is benign if $c$ is monotonic (that is, $c(x+1,s) \leq c(x,s) \leq c(x,s+1)$ for each $x < s$), and there is a computable function $h$ such that for any sequence $x_0 < x_1 < \cdots < x_k$ with $c(x_i, x_{i+1}) \geq 2^{-n}$ for each $i < k$, then we must have $k \leq h(n)$.

The idea is that if a set $A \models c$ and $c$ is a (benign) cost function, then $A$ is computationally weak because changes in its enumeration are restricted by the cost function. For instance every superlow c.e. set satisfies a benign cost function. Generally the more slow-growing the bound, the more restrictions the cost function will place on $A$. We prove that no matter how slow-growing the bound is on a benign cost function $c$, one can always construct a superprompt set obeying $c$.

**Theorem 4.3.** If $c$ is a benign cost function, then there is a strongly prompt set $B$ such that $B \models c$.

**Proof.** Let $c$ be a cost function, which is benign via some function $h$. We enumerate a promptly simple set
\[ B_s = B_{s-1} \cup \{x : \exists \epsilon \]
\[ W_{e,s} \cap B_{s-1} = \emptyset \quad \text{We have not met the $e$-th simplicity requirement.} \]
\[ x \in W_{e,s} \quad \text{We can meet it with } x. \]
\[ x \geq 2\epsilon \quad \text{We make } B \text{ co-infinite.} \]
\[ c(x,s) \leq 2^{-\epsilon} \quad \text{We ensure that } B \text{ obeys } c. \]

Since benign cost functions necessarily obey the limit condition in (16), $c(x,s)$ will eventually be small for large enough $x$, so if $W_{e}$ is infinite, eventually some $x$ in $W_{e}$ will meet the four conditions and be enumerated into $B$. Moreover, it will be enumerated in $B_s$ at the same stage $s$ it appears in $W_{e}$, so $B$ is prompt via the identity.

We now ask how large the set $W_{e}$ must be in order for some $x \in W_{e}$ to be enumerated into $B$ to meet the simplicity requirement. We initially guess that $|W_{e}| > 2\epsilon$ is sufficient, since that will ensure some element of $W_{e}$ will meet the first three requirements. Any time some $x$ meets the first three requirements but not the fourth, and is bigger than our current guess for the minimum size of $W_{e}$, we revise our guess upwards to the stage at which this occurs. Let $s_1$ be our first guess, $s_2$ our second, and so on. The $n$th time we revise our guess upwards, there is some $x > s_n$ such that $c(x,s_{n+1}) > 2^{-\epsilon}$, so (by monotonicity) the sequence of guesses satisfies $c(s_i, s_{i+1}) > 2^{-\epsilon}$ for $1 \leq i < k$, where $k$ is the total number of guesses. Thus (because $c$ is benign via $h$), the total number of times we change our guess is bounded by $h$, and so $B$ is strongly prompt. \[ \square \]
As a corollary, we also see that $K$-triviality is compatible with strong promptness.

**Corollary 4.4.** There is a $K$-trivial c.e. set which is strongly prompt. Hence $K$-trivial sets can be superlow cuppable.

We have an alternative proof of Greenberg and Nies’ [9] result that no single benign cost function serves to characterize the strongly jump traceable sets: Here we see that for any benign cost function, there is a c.e. set obeying that cost function and is strongly prompt (hence superlow cuppable and not strongly jump traceable).

The following theorem gives another example of a set which is promptly simple and not superlow cuppable. Currently the only known sets in this class are the promptly simple strongly jump traceable sets. One can also find such examples outside of the class of superlow sets. We note that degree theoretically there is no difference between promptness and prompt simplicity.

**Theorem 4.5.** There is a promptly simple c.e. set $B$ which is not superlow cuppable, and moreover is not superlow.

**Proof (Sketch).** We modify the original construction of a set $B$ which is prompt and non-superlow-cuppable, from [5]. In order to make $B$ non-superlow, we construct a computable functional $Ψ$, giving us (via the recursion theorem) a computable function $f$ such that $J_B(f(i)) = Ψ_B(i)$. Thus by changing the use of $Ψ_B(i)[s]$ over the course of the construction, we can diagonalize against all $ω$-c.e. approximations which might give $B'$. To defeat a single $ω$-c.e. approximation, we first choose a follower $i$, and wait for the computable bound on the number of mind changes to converge. (Until it converges, the supposed “$ω$-c.e. approximation” does not appear to be one, so waiting forever on a single requirement constitutes a win.) Once it does, we look at the approximation, and every time it appears to give the right answer on $i$, we change $B↾ψ(i)[s]$ (where $ψ$ is the use of $Ψ$) if it previously converged, or create a new computation with large use if it previously diverged. This constitutes an additional finite (bounded by the maximum number of mind changes of the approximation) injury, which does not interfere with the original construction.

We now consider the ideals of the almost deep and the almost superdeep degrees. Recall that a c.e. degree $a$ is almost (super)deep if $a ∪ b$ is (super)low for every (super)low c.e. degree $b$. Each promptly simple strongly jump traceable set is almost superdeep but certainly not almost deep. We show that not every almost deep degree is almost superdeep. In fact we are able to construct an almost deep degree which is not totally $ω$-c.e.

**Theorem 4.6.** There is an almost deep c.e. degree which is not totally $ω$-c.e.

**Proof.** We build $A$ satisfying the almost deep requirements:

$N_e$: If $W_e$ is low via $g_e$, then make $W_e ⊕ A$ low.

Here $g_e$ is the $e^{th}$ function from a list of total computable functions, with the property that for any low c.e. set $W$, there is some $e$ such that for every $x$, $\lim_{s \to \infty} g_e(x, s)$ exists and equal $W'(x)$. To test for the lowness of $W_e$ we enumerate Turing functionals and assume by the recursion theorem we know these indices, given to us by a computable function. To simplify notation we assume we control $J_X(0), J_X(1), \cdots$ instead.

To make $A$ not totally $ω$-c.e. we need to build a total $Φ^A$ which is not $ω$-c.e. We do not build $Φ$ directly in the construction. Rather we define $Φ$ implicitly using the c.e. approximation $A_s$ to $A$ given by the construction. Let $\overline{A}_s = \{a^0_s < a^1_s < \cdots\}$ be the elements of $\overline{A}_s$ listed in increasing
order. During a single stage of the construction we will have several numbers entering $A_s$. As is customary when we refer to $a_0^e < a_1^e < \cdots$ (or any other parameter) we mean the respective values within a stage $s$ where the parameter is mentioned. We say that $a_e$ is moved at stage $s$ if $a_e^s$ is enumerated at (some substage of) stage $s$. We always assume that when this happens, $a_{e+1}^s, \ldots, a_k^s$ are also enumerated into $A$ for some fresh number $k$. We have to ensure that each $a_e$ is moved finitely often, and that we satisfy the requirements

$$P_e: \text{If } f_e(x) = \omega \text{-c.e. via } h_e, \text{ then for some } x > e, a_x \text{ is moved at some stage } t$$

after which there are no more mind changes in $f_e(x, -)$. As usual $\{f_e\}$ and $\{h_e\}$ are effective lists of total computable functions, and $f_e$ is $\omega$-c.e. via $h_e$ if for every $x$, there are only $h_e(x)$ many mind changes in $f_e(x, -)$. Any $\omega$-c.e. function (by the standard trick) is the limit of $f_e$ for some $e$ where $f_e = \omega$-c.e. via $h_e$. To define $\Phi^A(x)$ we $A$-compute the least stage $s$ after which $a_0, \ldots, a_x$ are never moved. Output $1 + \max\{f_0(x, s), \ldots, f_x(x, s)\}$. If every $P$ is satisfied then $\Phi^A$ cannot be $\omega$-c.e.

4.1. Construction tree. The construction takes place on the following tree. Nodes of even length $2e$ are assigned the requirement $P_e$. Nodes of length $2(e, 0) + 1$ are assigned the requirement $N_e$, while nodes of length $2(e, k + 1) + 1$ are assigned the subrequirement $N_{e,k}$ of $N_e$, which are devoted to preserving the computation $J_{W_e \oplus A}(k)$. A node is said to be a $P_e, N_e$ or $N_{e,k}$-node if they are assigned the corresponding requirement. If $\tau \subset \alpha$ where $\tau$ is an $N_e$-node and $\alpha$ is a $N_{e,k}$-node, then we also say that $\tau$ is the top node of $\alpha$, which we denote as $\tau = \tau(\alpha)$. $\alpha$ is called a $k$-subnode of $\tau$. The subrequirements and the subnodes of a top node $\tau$ are used to split a potentially infinite restraint function into infinitely many finite restraints (of differing priority). The $P$-nodes have only a single outcome, denoted as 0, while the top and subnodes have two outcomes: \infty to the left of $f$. The actions of the subnodes are coordinated by the top node, with the help of tokens. Each subnode $\alpha$ has a token (which we call an $\alpha$-token), which is a marker indicating the state of the substrategy $\alpha$. Each token can either be marked (indicating that $\alpha$ wants to hold restraint on $A$), or unmarked (indicating that $\alpha$ wants to drop all $A$-restraint for the sake of lower priority nodes). An $\alpha$-token can be moved between the top node $\tau = \tau(\alpha)$ and $\alpha$. If the $\alpha$-token is currently held by $\tau$, then the next time $\tau$ sees $J_{W_e \oplus A}(k) \downarrow$, $\tau$ will test the superlowness of $W$ using the index $i_\alpha$. Upon a successful test, $\tau$ will mark the tokens it has used to test and return the tokens to the subnodes. These tokens serve the same purpose as the “connections” in Cholak, Groszek and Slaman [4]. These connections between the top and subnodes do not cause the construction to skip over any intermediate nodes. To make $W_e \oplus A$ low each top node $\tau$ defines a computable function $t_\tau(x, s)$ which is intended to predict the value of $(W_e \oplus A')(x)$. The only parameter associated with a $P_e$-node $\alpha$ is $x_\alpha$ denoting its follower.

4.2. Strategy for making $A$ almost deep. The construction of an almost deep c.e. set $A$ first appeared in Cholak, Groszek and Slaman [4]. Since it is not well-known, we will describe it here. Suppose we only wanted to make $A$ non-computable. Hence each $P$-node enumerates at most once. Suppose $\tau$ is an $N_e$-node. The basic strategy for ensuring $t_\tau(k, -)$ predicts $(W_e \oplus A')$ correctly is straightforward. Each time we see $J_{W_e \oplus A}(k)[s_0] \downarrow$ (with use $u[s_0]$) we test for the correctness of $W_e$ by enumerating $J_{W_e}(i) \downarrow$ with the same use $u[s_0]$. We freeze $A$ until either $W_e$ changes, or $g_e(i) = 1$. If neither happens then we get a global win for $\tau$, so the $A$-restraint imposed during this testing phase will have priority $\tau$. It is important that we allow no $A$-change below $u[s_0]$ during
the testing phase, otherwise the index $i$ has to be abandoned, and hence all previous progress on $g_e(i)$ is undone.

If $W_e$ changes then we drop the restraint $A$ and wait for the next time $J^{W_e \oplus A}(k) \downarrow$. We can test $J^{W_e}(i)$ again using the same index $i$. On the other hand if $g_e(i)$ flips to 1, then we also set $t_\tau(k) = 1$, and continue to restrain $A$. This means that if later $J^{W_e \oplus A}(k) \uparrow$, then this has to be due to a $W_e \uparrow u[s_0]$-change, which means that $J^{W_e}(i) \uparrow$. We then wait for $g_e(i) = 0$ before repeating. In this simple case, if $W_e$ is low via $g_e$, then we will only flip $t_\tau(k)$ finitely many times.

It is clear that the simple strategy above will not work, because $\tau$ may implement an increasing amount of $A$-restraint, since there are infinitely many different $k$ which want to restrain $A$. The obvious thing to do is to let $\tau$ hold the $A$-restraint only at testing phases (since if this $A$-restraint is never dropped then $\tau$ gets a global win). When each $k$-module gets successfully tested, we transfer the $A$-restraint down to the level of the $k$-subnodes. This ensures that each positive node extending $\tau$ only has to obey a finite amount of restraint. This means that during the second phase of the basic $k$-strategy, we might have an $A$-change resulting in $t_\tau(k)$ to be incorrect, and for the index $i$ to be abandoned. For each level $k$ this happens only finitely many times, so this does not cause additional problems.

The unpleasant situation is that there might be infinitely many stages where $J^{W_e \oplus A}(k)$ is successfully tested on $J^{W_e}(i)$, and the restraint is transferred to level $k$. Although this means that $W_e$ is not low via $g_e$, we must still allow for the positive nodes below level $k$ to act. To arrange for this we let each $k$-subnode have two outcomes. The intention is that every time we visit a $k$-subnode $\beta$ at some stage where the $A$-restraint was recently increased, we will drop restraint on $\beta$ by letting $\beta^\sim \infty$ be accessible. Suppose we only had a single testing index $i$ for the entire level $k$ (in the discussion so far, $i$ only needed to be reset finitely often due to positive action between $\tau$ and the $k$-subnodes of $\tau$). The problem with allowing $k$-subnodes to drop restraint (infinitely often) is the following: Consider a $k$-subnode $\beta_0$ on the true path; it could be the case that restraint is transferred to level $k$ from $\tau$ infinitely often, but $\beta_0$ is only visited at those stages where $J^{W_e}(i)$ has been tested successfully and restraint is currently held by level $k$. This means that $\beta_0$ has to drop restraint at infinitely many of these stages, which would cause $i$ to be reset infinitely often, and hence causing $\mathcal{N}_e$ not to be met.

It is clear now what we need to do. We need to allow different $k$-subnodes $\beta$ to have different testing indices $i_\beta$. If $\beta_1 <_L \beta_1$, $\beta_0$ is visited and it feels the need to drop $A$-restraint, it will reset $i_{\beta_1}$ (and cause $t_\tau(k)$ to be wrong). To meet $\mathcal{N}_e$ we need to restrict the number of mind changes to $t_\tau(k)$. In the above example when $\beta_0$ was visited, we allowed for $t_\tau(k)$ to be injured without accounting for it against any $g_e(i_\beta)$. This is fine as long as we test $i_{\beta_0}$ before next believing that $J^{W_e \oplus A}(k) \downarrow$ and consequently flipping $t_\tau(k) = 1$. If $\beta_0$ is the leftmost $k$-subnode visited infinitely often, then $i_{\beta_0}$ is reset finitely often, and we cannot have infinitely many mind changes in $t_\tau(k)$ (unless $g_e(i_{\beta_0})$ has infinitely many). In the formal construction this is coordinated via tokens - the tokens will tell $\tau$ which of its $k$-subnodes are ready to be tested again.

**Implementing a finite amount of positive injury:** In the discussion above we only considered positive requirements making $A$ non-computable. Consequently each $t_\tau(k)$ can only be injured by a positive requirement finitely often. In this construction, if a $\mathcal{P}$-node $\eta$ is not on the true path then $\eta$ may change $A$ infinitely often (since $\lim x_\eta = \infty$). This may now cause $t_\tau(k)$ to be injured infinitely often. To get around this, whenever $\tau$ successfully tests $J^{W_e}(i_{\beta_0}), \ldots, J^{W_e}(i_{\beta_m})$, and transfers restraint to level $k$, we will also initialize every positive node $\eta$ which lies to the right of
some $\beta_i$, $i \leq m$. Hence any restraint held by level $k$ which have been successfully tested on $J_{W_c}(i\beta)$ for $\beta$ on true path, can only be later taken down by some positive node $\eta \subset \beta$. This is good since $x_0$ (for these $\eta$) will be eventually stable. We mention a final point: even though a positive node $\eta$ may be initialized (at some stage $s_1$) even when we do not visit left of $\eta$, the only time this can happen to $\eta$ again is if we really do visit left of $\eta$ after $s_1$.

4.3. **Construction.** At stage $s = 0$, initialize every node and do nothing. To initialize a $\mathcal{P}$-node $\alpha$ means that we set $x_\alpha \uparrow$. To initialize a subnode $\alpha$ means we unmark the $\alpha$-token, and if $i_\alpha \downarrow$ we reset it to a fresh value. To initialize a top node $\tau$ means that for every subnode $\alpha$ of $\tau$ we initialize $\alpha$ and return to $\alpha$ its token. At $s > 0$ we define the stage $s$ approximation to the true path, $\delta_\alpha$ of length $s$ by induction on $u < s$: Assume that $\alpha = \delta_k \uparrow u$ has been defined. We say $\alpha$ has been visited at $s$; in this case we let $s^-$ be the previous stage where $\alpha$ was visited. There are three cases:

1. $\alpha$ is a $\mathcal{P}_c$-node: If $x_\alpha \uparrow$, we set it to be a fresh number. Otherwise if $h_\alpha(x_\alpha) \downarrow$ and $f_\alpha(x_\alpha, s) \neq f_\alpha(x_\alpha, s^*)$ (with at most $h_\alpha(x_\alpha)$ mind changes), we move $a_{x_\alpha}$, and initialize every node extending $\alpha$ and to the right of $\alpha$. Here $s^* = s^-$ if $h_\alpha(x_\alpha)[s^-] \downarrow$, and $s^* = 0$ otherwise. Let $\delta_k(u) = 0$.

2. $\alpha$ is an $\alpha_c$-node: Let $t < s$ be the previous $\alpha^-\infty$-stage ($t = 0$ if this does not exist). For every $k < t$ such that $\alpha$ is holding a $k$-token (i.e. a $\beta$-token for some $k$-subnode $\beta$), we set $t_\alpha(k, s) = 0$. Also for every $k < t$ such that for some $k$-subnode $\beta$, the $\beta$-token is marked and $J_{W_c}(i\beta) \uparrow$, we unmark the $\beta$-token.

Next check if for every $k < t$ and every $k$-subnode $\beta$ for which $i_\beta \downarrow$, we have $A_\beta(i_\beta, s)$ predicting $J_{W_c}(i\beta)$ correctly. If not do nothing. Otherwise if every $k$-subnode for $k < t$ is predicted correctly, we find the least $k < t$ such that the $k$-module (of $\alpha$) requires attention: that is, $\alpha$ is holding at least one $k$-token. $J_{W_c \oplus A}(k)t \downarrow$ and this computation currently applies. For the least $k < t$ found there are two possibilities:

i. There is a $k$-subnode $\beta$ where $\alpha$ is holding a $\beta$-token, and $J_{W_c}(i\beta) \uparrow$. For each such $k$-subnode $\beta$ we enumerate $J_{W_c}(i\beta) \downarrow$ with the same $W_c$-use in $J_{W_c \oplus A}(k)t$.

ii. Not (i). Let $B = \{\beta \mid \beta$ is a $k$-subnode and $\alpha$ is holding a $\beta$-token$\}$. Then for every $\beta \in B$, $J_{W_c}(i\beta) \downarrow$ and $A_\beta(i_\beta, s) = 1$. Let $\beta_0 = \alpha$ be the leftmost node in $B$. Mark the $\beta_0$-token and for every $\beta \in B$ return $\beta$ its token. Set $t_\alpha(k, s) = 1$. Initialize every node to the right of $\beta_0^-\infty$.

In all the cases above we define $\delta_k(u) = f$ and initialize every node to the right of $\alpha$, and go to the next stage. Finally if no such $k < t$ is found we define $\delta_k(u) = \infty$, initialize every node to the right of $\alpha$, and go to the next substage.

3. $\alpha$ is an $\alpha_{c,k}$-node: If $\alpha \supset \tau(\alpha)^-f$, do nothing and let $\delta_k(u) = f$. Otherwise if this is the first visit to $\alpha$, set $i_\alpha \downarrow$ to a fresh value, and let $\delta_k(u) = f$. Otherwise check if there is some $k$-subnode $\alpha_0$ of $\tau(\alpha)$, such that $\alpha_0$ has a marked token. There are two possibilities:

i. $\alpha_0$ exists, and $\alpha_0 \leq_L \alpha$. By the construction $\alpha_0$ is unique and must be currently holding its token. Do nothing, and let $\delta_k(u) = f$.

ii. Not (i). Pass the $\alpha$-token to $\tau(\alpha)$ and let $\delta_k(u) = \infty$.

Initialize every node to the right of $\alpha$ and go to the next substage.

4.4. **Verification.** Define the true path $TP$ to be the left most path visited infinitely often. Each $a_x$ is clearly moved finitely often.

**Lemma 4.7.** Every node on $TP$ is initialized finitely often.
\textbf{Proof.} Suppose inductively that every $\beta \subseteq \alpha \subseteq TP$ is initialized finitely often, and suppose for a contradiction, that $\alpha$ is initialized infinitely often. Since every $P$-node extended by $\alpha$ is going to settle on a final follower $x$, it cannot initialize $\alpha$ infinitely often. The only nontrivial case to consider is the action of some top node $\tau$ under step 2(ii) of the construction. If $\alpha \supseteq \tau^{-f}$, then $\tau$ can only initialize $\alpha$ finitely often: Suppose $t$ is the final $\tau^{-\infty}$-stage. Then after stage $t$, $\tau$ can only initialize $\alpha$ under step 2(ii) due to a $k$ module for some $k < t$. Each of these $k$-modules will never require attention again once it has been attended to under step 2(ii) (since all the $k$-tokens are returned and never transferred back to $\tau$). Hence we may assume that $\alpha \supseteq \tau^{-\infty}$. At each of these stages, there is a corresponding $\tau$-subnode $\beta_0$ (whose token gets marked and returned). Since no node to the left of $\alpha$ can be visited eventually, it follows that there is a single $\beta_0$ such that $\alpha \supseteq \beta_0 f$ for which $\tau$ initializes $\alpha$ infinitely often and marks $\beta_0$. This means that there are infinitely many stages where $\beta_0$ sends its token back to $\tau$ - each of these must be a $\beta_0^{-\infty}$-stage which is a contradiction.

The argument above also shows that every node to the left of $TP$ is initialized finitely often. By Lemma 4.7, every $P_e$ is satisfied. Fix an $\mathcal{N}_e$ such that $W_e$ is low via $g_e$, and let $\tau$ be the version of $\mathcal{N}_e$ along the $TP$. We first claim that there are infinitely many $\tau^{-\infty}$-stages. Suppose not - then there is a final $\tau^{-\infty}$-stage $t > 0$ (note that there is at least one $\tau^{-\infty}$-stage). Since each subnode extending $\tau^{-\infty}$ is initialized finitely often, let $t' > t$ be a stage after which $\tau$ and all of its $k$-subnodes extending $\tau^{-\infty}$ for $k < t$ are never initialized. It follows that after stage $t'$, each $k$-module for $k < t$ is attended to under 2(ii) and 2(ii) finitely many times. Fix $\beta$ which is a $k$-subnode ($k < t$) and $i_\beta[t'] \downarrow$. Then it is easy to check that only finitely many axioms are enumerated for $J^{W_e(i_\beta)}$. It follows that every $k$-subnode for $k < t$ is predicted correctly at almost every visit to $\tau$. This gives a contradiction.

Now we show that for each $k$, $\lim t_e(k, +)$ exists. Suppose not, then there are infinitely many stages for which the $k$-module is attended to under 2(ii). Let $\beta_0$ be the leftmost $k$-subnode marked infinitely often. It is easy to see that $\beta_0 \subseteq TP$. Hence $i_{\beta_0}$ eventually settles. Let $s_1 < s_2$ be two $\tau$-stages where the $k$-module is attended to under 2(ii), and $\beta_0$ is marked. We have $g_e(i_{\beta_0}, s_1) = g_e(i_{\beta_0}, s_2) = 1$. Between $s_1$ and $s_2$, the mark on the $\beta_0$-token must be removed, and this can only be due to the action of $\tau$ during a $\tau$-visit at some $s_3$ between $s_1$ and $s_2$, in which we found $J^{W_e(i_{\beta_0})}$. After $s_3$, $\tau$ will do nothing until it finds $g_e(i_{\beta_0}) = 0$. Hence between $s_1$ and $s_2$, $g_e(i_{\beta_0})$ will have a mind-change. This contradiction shows that $\lim t_e(k, -)$ exists.

\textbf{Lemma 4.8.} Let $\beta$ be a $k$-subnode of $\tau$, where at stage $s$ we enumerate $J^{W_e(i_\beta)} \downarrow$. Then $J^{W_e \oplus A(k)[s]} \downarrow$ with the same $W_e$-use, and the latter computation persists until either $\beta$ is initialized, or $W_e$ changes below the common use.

\textbf{Proof.} Suppose that $\alpha$ is a $P$-node which changed $A$ below the $A$-use of $J^{W_e \oplus A(k)[s]}$, say at stage $s_1 > s$. Also assume that $W_e$ did not change below the use between $s$ and $s_1$, and $\beta$ was not initialized. The nontrivial case to consider is when $\alpha \supseteq \tau$. If $\alpha \supseteq \tau^{-f}$ then $\alpha$ was initialized at the previous $\tau^{-\infty}$ stage $t < s$, and $J^{W_e \oplus A(k)[s]}$ has persisted since $t$. Hence $x_s > t > A$-use. Suppose $\alpha \supseteq \tau^{-\infty}$. $\alpha$ can only be visited at the next $\tau^{-\infty}$-stage. Before the next $\tau^{-\infty}$-stage, some the $k$-module must be attended to under step 2(ii) resulting in some node $\beta_0 \leq_L \beta$ to be marked. Furthermore at this point we must have $J^{W_e(i_{\beta_0})} \downarrow$ and by assuming that $s$ is minimal, we have the use on $J^{W_e(i_\beta)}$ is the same as the use on $J^{W_e(i_\beta)[s]}$. 

22 DAVID DIAMONDSTONE, KENG MENG NG
If $\alpha$ is to the right of $\beta \preceq \infty$, then $\alpha$ is initialized when $\beta_0$ is marked between $s$ and $s_1$. If $\alpha < L \beta$ then $\beta$ will be initialized when $\alpha$ is visited. Finally suppose $\alpha \supseteq \beta \preceq \infty$. Hence the mark on $\beta_0 \leq_L \beta$ has to be removed before $s_1$. Since $\beta$ is not initialized before $s_1$, this means that the mark on $\beta$ has to be removed at some $s_2$, $s < s_2 < s_1$ when $\tau$ is visited and found $J^{W_{\tau}(i/\beta_0)}[s_2] \uparrow$. Since $\beta$ and hence $\beta_0$ is not initialized, this means that $W_{\tau}$ has changed below the common use, a contradiction. 

Finally we verify that $\lim t_\tau(k, -)$ predicts $J^{W_e \oplus A}(k)$ correctly. Suppose that $\lim t_\tau(k, -) \equiv 1$, and at some stage $s_0$ we set $t_\tau(k, s_0) = 1$ for the last time. Let $\beta_0$ be the subnode at $s_0$ whose token is marked. This mark cannot be removed, because otherwise (by the fact that there are infinitely many $\tau \preceq \infty$-stages), $t_\tau(k)$ will be flipped to 0 at some later stage. Since this mark is never removed, it follows by Lemma 4.8 that $J^{W_e \oplus A}(k)[s_0]$ applies forever. Suppose now that $J^{W_e \oplus A}(k) \downarrow$. Hence at almost every $\tau \preceq \infty$-stage, $\tau$ is holding no $k$-token (otherwise the $k$-module will be attended to). This means that there must be a marked $k$-subnode $\beta_1$ whose mark is never removed. In fact, $\beta_1 \leq_L TP \uparrow |\beta_1|$. Correspondingly there is some stage $s_1$ where $t_\tau(k, s_1)$ is set to 1. Since the mark on $\beta_1$ is never removed after $s_1$, it follows that $\tau$ will never get to hold any $k$-token after $s_1$, which means that $\lim t_\tau(k, -) \equiv 1$.

**Corollary 4.9.** Not every almost deep degree is array computable. Hence neither the almost deep degrees nor the almost superdeep degrees are contained in the other.

**References**


University of Chicago, 5734 S. University Avenue, Chicago, Illinois 60637

E-mail address: ded@math.uchicago.edu

University of Wisconsin, 480 Lincoln Drive, Madison, Wisconsin 53706

E-mail address: selwynng@math.wisc.edu