Closure and Nonclosure Properties of the Classes of Compressible and Rankable Sets[*]

Jackson Abascal[†‡]
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Lane A. Hemaspaandra[§¶]
Department of Computer Science
University of Rochester
Rochester, NY 14627, USA

Shir Maimon[† ‡]
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA

Daniel Rubery[‡]
Google, Inc.
1600 Amphitheatre Pkwy.
Mountain View, CA 94043, USA

Abstract

The rankable and the compressible sets have been studied for more than a quarter of a century. We ask whether these classes are closed under the most important boolean and other operations. We study this question for both polynomial-time and recursion-theoretic compression and ranking, and for almost every case arrive at a Closed, a Not-Closed, or a Closed-Iff-Well-Known-Complexity-Classes-Collapse result. Although compression and ranking classes are capturing something quite natural about the structure of sets, it turns out that these classes are quite fragile with respect to closure properties, and many fail to possess even the most basic of closure properties. For example, we show that with respect to the join (aka disjoint union) operation: the P-rankable sets are not closed, whether the semistrongly P-rankable sets are closed is closely linked to whether $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$, and the strongly P-rankable sets are closed.

Key words: complexity theory, closure properties, compression, ranking, computability.

1 Introduction and Related Work

Loosely put, a compression function f for a set A is a function over the domain Σ^* such that (a) $f(A) = \Sigma^*$ and (b) $(\forall a, b \in A : a \neq b)[f(a) \neq f(b)]$. That is, f puts A in 1-to-1 correspondence with Σ^*. This is sometimes described as providing a minimal perfect hash function for A: It is perfect since there are no collisions (among elements of A), and it is minimal since not a single element of the codomain is missed. Note that the above does not put any constraints on what strings the elements of $\overline{A}$ are mapped to, or even about whether the compression function needs to be defined on such strings. A ranking function is similar, yet stronger, in that a ranking function sends the ith string in A to the integer i; it respects the ordering of the members of A.

[*]A preliminary version of this paper appeared in the Proceedings of the Thirteenth International Conference on Language and Automata Theory and Applications (LATA 2019) [AHMR19].

[†]Supported in part by a CRA-W CREU grant.

[‡]Work done in part while at the University of Rochester's Department of Computer Science.

[§]Supported in part by NSF Grant CCF-2006496.

[¶]This work was done in part while on a sabbatical stay at ETH Zürich and the University of Düsseldorf, with support from ETH Zürich and a Renewed Research Stay award from the Alexander von Humboldt Foundation.

The study of ranking was started by Allender [All85] and Goldberg and Sipser [GS91], and has (along with, in some cases, the study of so-called unranking) been pursued in many papers since, e.g., [HR90, Huy90, BGS91, ÁJ93, MM01, MR01, WCCL13, MT14, Tar15, CWCC19, HS19] (see also the related study of census functions by Goldsmith, Ogihara, and Rothe [GOR00]). The study of ranking led to the study of compression, which was started—in its current form, though already foreshadowed in a notion of Goldberg and Sipser [GS91]—by Goldsmith, Hemachandra, and Kunen [GHK92] (see also Goldsmith and Homer [GH96]). Proof techniques from that compression study [GHK92] have been interestingly applied by Buhrman, Fortnow, and Laplante [BFL01] and Allender and Spakowski [AS12] in the study of resource-bounded Kolmogorov complexity. Also, many papers in complexity theory look at aspects of functions mapping, as compression functions do, "onto" Σ^*, see, e.g., [BFH78, FFNR03, HRW97, Rot99].

The abovementioned ranking and compression research focused on polynomial-time or logarithmic-space ranking or compression functions. More recently, both compression and ranking have also been studied in the recursion-theoretic context by Hemaspaandra and Rubery [HR19]; see the discussion therein about related—though not identical—precursor notions in computability theory such as retraceable sets [DM58], regressive sets [Dek62], and isolic reductions [Rog67, p. 124]. In particular, that paper [HR19] studies compression and ranking for both the case of (total) recursive compression/ranking functions (which of course must be defined on all inputs in Σ^*) and the case of partial-recursive compression/ranking functions (i.e., functions that on some or all elements of the complement of the set being compressed/ranked are allowed to be undefined). The prior work most closely related to the present paper is that paper and the papers cited in the previous paragraph, which defined and explored complexity-theoretic compression and ranking.

In the present paper, we continue the study of both complexity-theoretic and recursion-theoretic compression and ranking functions. In particular, the earlier papers often viewed the compressible sets or the rankable sets as a *class*. We take that to heart, and seek to learn whether these classes do, or do not, possess key closure properties. Doing so gives insight into the structure and behavior of these classes. What we learn is more nuanced than a blanket Yes or No as to the possession of the key closure properties; we will show that even sibling classes can sharply differ in their closure properties. Our main contributions can be seen in Table 1, where we obtain closure and nonclosure results for many previously studied variations of compressible and rankable sets under boolean operations (Section 4). We also (as Section 5) study the issue of the closure/nonclosure of these classes under additional operations, such as the join, aka disjoint union; Table 2 summarizes our results on the closure/nonclosure under the join of each of the complexity-theoretic rankability/nonrankability classes. The prior work on closure properties of complexity classes is too extensive to list in full. It ranges for example from the familiar course exercise of showing that NP and coNP are closed under union and intersection yet are closed under complementation if and only if NP = coNP to work studying closure properties of other language/complexity classes such as automata-based classes (e.g., [OS18, DFF19, MR19]), biology-operator-/counting-/formal-language-/selection-/space-based classes (e.g., [Imm88, Sze88, HJ95, HT05, Iba16, LMN16, DRS17]), and much more. We mention that the authors have obtained results [AHMR16, Appendix B.2], not covered in this paper, on compressibility in the context of selectivity (see, e.g., [Sel79, HT03]) and honesty [GS88].

We also introduce the notion of compression *onto a set* and characterize the robustness of compression under this notion. In particular, by a finite-injury priority argument with some interesting features we show that there exist RE sets that each compress to the other, yet that nonetheless are not recursively isomorphic (Section 3).

Class	$\cap$	$\cup$	Complement
strong-P-rankable	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 4.2)	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 4.2)	Yes (Prop. 4.3)
semistrong-P-rankable	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 4.2)	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 4.2)	$\approx \mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$ (Th. 4.8, Th. 4.10)
P-rankable, P-compressible$'$, $\mathrm{F}_{\mathrm{REC}}$-rankable, $\mathrm{F}_{\mathrm{REC}}$-compressible, F_{PR}-rankable, and F_{PR}-compressible	No (Th. 4.11)	No (Th. 4.12)	No (Th. 4.13)
strong-P-rankable$^{\mathsf{C}}$	No (Th. 4.14)	No (Th. 4.14)	Yes (Prop. 4.3)
semistrong-P-rankable$^{\mathsf{C}}$	No (Th. 4.14)	No (Th. 4.14)	$\approx \mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$ (Th. 4.8, Th. 4.10)
P-rankable$^{\mathsf{C}}$, P-compressible$^{\mathsf{C}}$, $\mathrm{F}_{\mathrm{REC}}$-rankable$^{\mathsf{C}}$, $\mathrm{F}_{\mathrm{REC}}$-compressible$^{\mathsf{C}}$, F_{PR}-rankable$^{\mathsf{C}}$, and F_{PR}-compressible$^{\mathsf{C}}$	No (Th. 4.14)	No (Th. 4.14)	No (Th. 4.13)

Table 1: Overview of our results for closure of these classes under boolean operations. If an entry does not include "No" or "Yes," then the class is closed under the operation if and only if the entry holds. A special case is semistrong-P-rankable and semistrong-P-rankable$^{\mathsf{C}}$, in which we deliberately use the $\approx$ symbol to indicate that the implication is true in one direction and in the other direction currently is known to be true only for a broad subclass of these sets. Specifically, if $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$ then the complements of all "nongappy" semistrong-P-rankable sets are themselves semistrong-P-rankable.

2 Definitions

Throughout this paper, "P" when used in a function context (e.g., the P-rankable sets) will denote the class of total, polynomial-time computable functions from Σ^* to Σ^*. Additionally, throughout this paper, $\Sigma = \{0, 1\}$. $\mathrm{F}_{\mathrm{REC}}$ will denote the class of total, recursive functions from Σ^* to Σ^*. F_{PR} will denote the class of partial recursive functions from Σ^* to Σ^*.

The symbol ϵ will denote the empty string. We use the standard notion of lexicographical order over Σ^*, i.e., from lexicographically least string onward we have ϵ, 0, 1, 00, 01, 10, 11, 000, etc. When we use binary comparison operators on strings—as for example is done in Definition 2.2 and many other places—the comparison is with respect to lexicographical order. We define the function $\mathrm{shift}(x, n)$ for $n \in \mathbb{Z}$. If $n \geq 0$, then $\mathrm{shift}(x, n)$ is the string n spots after x in lexicographical order, e.g., $\mathrm{shift}(\epsilon, 4) = 01$. For $n > 0$, define $\mathrm{shift}(x, -n)$ as the string n spots before x in lexicographical order, or ϵ if no such string exists. We define the symmetric difference $A \triangle B = (A - B) \cup (B - A)$. The symbol $\mathbb{N}$ will denote the natural numbers $\{0, 1, 2, 3, \ldots\}$.

We now define the notions of compressible and rankable sets.

Definition 2.1 (Compressible sets [HR19])**.**

1. *Given a set $A \subseteq \Sigma^*$, a (possibly partial) function f is a* compression function for A *exactly if*
   - *(a)* $\mathrm{domain}(f) \supseteq A$*,*
   - *(b)* $f(A) = \Sigma^*$*, and*

Class	Join ($\oplus$)
strong-P-rankable	Yes (Th. 5.5)
semistrong-P-rankable	semistrong-P-rankable is closed under complementation (Th. 5.6)
P-rankable	No (Th. 5.4)
strong-P-rankable$^{\mathsf{C}}$	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 5.2)
semistrong-P-rankable$^{\mathsf{C}}$	$\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ (Th. 5.2)
P-rankable$^{\mathsf{C}}$	No (Th. 5.3)

Table 2: Overview of our results for closure of the complexity-theoretic rankability classes, and their complements, under the join operation. If an entry does not include "No" or "Yes," then the class is closed under the operation if and only if the entry holds. For example, the class semistrong-P-rankable is closed under join if and only if it is closed under complementation; whether that class is closed under complementation is itself partially characterized in this paper as per Table 1.

(c) for all a and b in A, if $a \neq b$ then $f(a) \neq f(b)$.

2. *Let $\mathcal{F}$ be any class of (possibly partial) functions mapping from Σ^* to Σ^*. A set A is $\mathcal{F}$-*compressible *if some $f \in \mathcal{F}$ is a compression function for A.*

3. *For each $\mathcal{F}$ as above, $\mathcal{F}$-compressible $= \{A \mid A$ is $\mathcal{F}$-compressible$\}$ and $\mathcal{F}$-compressible$'$ $=$ $\mathcal{F}$-compressible $\cup \{A \subseteq \Sigma^* \mid A$ is a finite set$\}$.*

4. *For each $\mathcal{F}$ as above and each $\mathcal{C} \subseteq 2^{\Sigma^*}$, we say that $\mathcal{C}$ is $\mathcal{F}$-*compressible *if all infinite sets in $\mathcal{C}$ are $\mathcal{F}$-compressible.*

Note that a compression function f for A can have any behavior on elements of $\overline{A}$ and need not even be defined. Finite sets cannot have compression functions as they do not have enough elements to be mapped onto Σ^*. Thus part 4 of Definition 2 defines a class to be $\mathcal{F}$-compressible if and only if its *infinite* sets are $\mathcal{F}$-compressible.

Ranking can be informally thought of as a sibling of compression that preserves lexicographical order within the set. We consider three classes of rankable functions that differ in how they are allowed to behave on the complement of the set they rank. Although ever since the paper of Hemachandra and Rudich [HR90], which introduced two of the three types, there have been those three types of ranking classes, different papers have used different (and sometimes conflicting) terminology for these types. Here, we use the (without modifying adjective) terms "ranking function" and "rankable" in the same way as Hemaspaandra and Rubery [HR19] do, for the least restrictive form of ranking (the one that can even "lie" on the complement). That is the form of ranking that is most naturally analogous with compression, and so it is natural that both terms should lack a modifying adjective. For the most restrictive form of ranking, which even for strings x in the complement of the set A being ranked must determine the number of strings up to x that are in A, like Hemachandra and Rudich [HR90] we use the terms "strong ranking function" and "strong(ly) rankable." And for the version of ranking that falls between those two, since for strings in the complement it need only detect that they are in the complement, we use the terms "semistrong ranking function" and "semistrong(ly) rankable."

Definition 2.2 ([All85, GS91]). $\mathrm{rank}_A(y) = \|\{z \mid z \leq y \wedge z \in A\}\|$.

Definition 2.3 (Rankable sets ([All85, GS91], see also [HR19])).

1. *Given a set $A \subseteq \Sigma^*$, a (possibly partial) function f is a* ranking function for A *exactly if*
   (a) domain$(f) \supseteq A$ *and*
   (b) *if $x \in A$, then $f(x) = \mathrm{rank}_A(x)$.*
2. *Let $\mathcal{F}$ be any class of (possibly partial) functions mapping from Σ^* to Σ^*. A set A is* $\mathcal{F}$-rankable *if some $f \in \mathcal{F}$ is a ranking function for A.*
3. *For each $\mathcal{F}$ as above, $\mathcal{F}$-rankable$= \{A \mid A$ is $\mathcal{F}$-rankable$\}$.*
4. *For each $\mathcal{F}$ as above and each $\mathcal{C} \subseteq 2^{\Sigma^*}$, $\mathcal{C}$ is* $\mathcal{F}$-rankable *if all sets in $\mathcal{C}$ are $\mathcal{F}$-rankable.*

Definition 2.4 (Semistrongly rankable sets ([HR90], see also [HR19])).

1. *Given a set $A \subseteq \Sigma^*$, a function f is a* semistrong ranking function for A *exactly if*
   (a) domain$(f) = \Sigma^*$,
   (b) *if $x \in A$, then $f(x) = \mathrm{rank}_A(x)$, and*
   (c) *if $x \notin A$, $f(x)$ indicates "not in set" (e.g., via the machine computing f halting in a special state; we still view this as a case where x belongs to* domain(f)*).*
2. *Let $\mathcal{F}$ be any class of functions mapping from Σ^* to Σ^*. A set A is* semistrong-$\mathcal{F}$-rankable *if some $f \in \mathcal{F}$ is a semistrong ranking function for A.*
3. *For each $\mathcal{F}$ as above, semistrong-$\mathcal{F}$-rankable $= \{A \mid A$ is semistrong-$\mathcal{F}$-rankable$\}$.*
4. *For each $\mathcal{F}$ as above and each $\mathcal{C} \subseteq 2^{\Sigma^*}$, we say that $\mathcal{C}$ is* semistrong-$\mathcal{F}$-rankable *if all sets in $\mathcal{C}$ are semistrong-$\mathcal{F}$-rankable.*

Definition 2.5 (Strongly rankable sets ([HR90], see also [HR19])).

1. *Given a set $A \subseteq \Sigma^*$, a function f is a* strong ranking function for A *exactly if*
   (a) domain$(f) = \Sigma^*$ *and*
   (b) *$f(x) = \mathrm{rank}_A(x)$ for all $x \in \Sigma^*$.*
2. *Let $\mathcal{F}$ be any class of functions mapping from Σ^* to Σ^*. A set A is* strong-$\mathcal{F}$-rankable *exactly if $(\exists f \in \mathcal{F})[f$ is a strong ranking function for $A]$.*
3. *For each $\mathcal{F}$ as above, strong-$\mathcal{F}$-rankable $= \{A \mid A$ is strong-$\mathcal{F}$-rankable$\}$.*
4. *For each $\mathcal{F}$ as above and each $\mathcal{C} \subseteq 2^{\Sigma^*}$, we say that $\mathcal{C}$ is* strong-$\mathcal{F}$-rankable *if all sets in $\mathcal{C}$ are strong-$\mathcal{F}$-rankable.*

Although the above definitions let us use semistrong-$\mathcal{F}$-rankable and strong-$\mathcal{F}$-rankable as adjectives (and also as nouns for the class of sets having those properties), to help the flow of the reading we sometimes will when using the adjectival form add a "ly," i.e., we will at times for the adjectival form write semistrongly-$\mathcal{F}$-rankable and strongly-$\mathcal{F}$-rankable.

For almost any natural class of functions, $\mathcal{F}$, we will have that $\mathcal{F}$-rankable$\subseteq$ $\mathcal{F}$-compressible$'$. In particular, P, F_{PR}, and $\mathrm{F}_{\mathrm{REC}}$ each have this property. If f is a ranking function for A (in the sense of part 1 of Definition 2.3), for our same-class compression function for A we can map $x \in \Sigma^*$ to the $f(x)$-th string in Σ^* (where we consider ϵ to be the first string in Σ^*) if $f(x) > 0$, and if $f(x) = 0$ what we map to is irrelevant so map to any particular fixed string (for concreteness, ϵ).

For each class $\mathcal{C} \subseteq 2^{\Sigma^*}$, $\mathcal{C}^{\complement}$ will denote the complement of $\mathcal{C}$, i.e., $2^{\Sigma^*} - \mathcal{C}$. For example, P-rankable$^{\complement}$ is the class of non-P-rankable sets.

The class semistrong-P-rankable is a subset of P (indeed, a strict subset unless $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ [HR90]), but there exist undecidable sets that are P-rankable. Clearly, the class of semistrong-$\mathrm{F}_{\mathrm{REC}}$-rankable sets equals the class of strong-$\mathrm{F}_{\mathrm{REC}}$-rankable sets.

3 Compression onto B: Robustness with Respect to Target Set

A compression function for a set A is 1-to-1 and onto Σ^* when the function's domain is restricted to A. It is natural to wonder what changes when we switch target sets from Σ^* to some other set $B \subseteq \Sigma^*$. We now define this notion. In our definition, we do allow strings in $\overline{A}$ to be mapped to B or to $\overline{B}$, or even, for the case of F_{PR} maps, to be undefined. In particular, this definition does not require that $f(\Sigma^*) = B$. Recall from Section 2 that, throughout this paper, $\Sigma = \{0, 1\}$.

Definition 3.1 (Compressible to B)**.**

1. *Given sets $A \subseteq \Sigma^*$ and $B \subseteq \Sigma^*$, a (possibly partial) function f is a* compression function for A to B *exactly if*
   (a) domain$(f) \supseteq A$*,*
   (b) $f(A) = B$, and
   (c) for all a and b in A, if $a \neq b$ then $f(a) \neq f(b)$.
2. *Let $\mathcal{F}$ be any class of (possibly partial) functions mapping from Σ^* to Σ^*. A set A is* $\mathcal{F}$-compressible to B *if some $f \in \mathcal{F}$ is a compression function for A to B.*

The classes $\mathcal{F}$ of interest to us in this section will be $\mathrm{F}_{\mathrm{REC}}$ and F_{PR}. Clearly, compression is simply the $B = \Sigma^*$ case of this definition, e.g., a function f is a compression function for A if and only if f is a compression function for A to Σ^*, and set A is $\mathcal{F}$-compressible if and only if A is $\mathcal{F}$-compressible to Σ^*. (Throughout this section, for clarity we will generally write out explicitly the "to Σ^*." Doing so avoids theorems where one side of an "if and only if" has a "to" and the other does not. Of course, in such cases the "to Σ^*" could be omitted, e.g., Theorem 3.2's part 1 could equivalently be phrased as "Let A and B be infinite sets; if $B \in \mathrm{REC}$, then A is $\mathrm{F}_{\mathrm{REC}}$-compressible to B if and only if A is $\mathrm{F}_{\mathrm{REC}}$-compressible.")

A natural first question to ask is whether compression to B is a new notion, or whether it coincides with our existing notion of compression to Σ^*, at least for sets B from common classes such as REC and RE. The following result shows that for REC and RE this new notion does coincide with our existing one.

Theorem 3.2. *Let A and B be infinite sets.*

1. *If $B \in \mathrm{REC}$, then A is $\mathrm{F}_{\mathrm{REC}}$-compressible to B if and only if A is $\mathrm{F}_{\mathrm{REC}}$-compressible to Σ^*.*
2. *If $B \in \mathrm{RE}$, then A is F_{PR}-compressible to B if and only if A is F_{PR}-compressible to Σ^*.*

Proof. We first prove part 1, beginning with the "if" direction.

Suppose A is $\mathrm{F}_{\mathrm{REC}}$-compressible to Σ^* by a recursive function f, and suppose B is recursive and infinite. Let $f'(x)$ output the element $y \in B$ such that $\mathrm{rank}_B(y) = f(x)$. Then f' is recursive, and A is $\mathrm{F}_{\mathrm{REC}}$-compressible to B by f'.

For the "only if" direction, let B be an infinite recursive set. Suppose that A is $\mathrm{F_{REC}}$-compressible to B by a recursive function f. Let $f'(x) = \epsilon$ if $f(x)$ is not in B. Otherwise, let $f'(x) = \mathrm{rank}_B(f(x))$. Then f' is recursive, and A is $\mathrm{F_{REC}}$-compressible to Σ^* by f'.

Let us turn to part 2 of the theorem. Again, we begin with the "if" direction. Let B be an infinite RE set, and let E enumerate the elements of B without repetitions. Suppose A is $\mathrm{F_{PR}}$-compressible to Σ^* by a partial recursive function f. Then f' does the following on input x.

1. Simulate $f(x)$. This may run forever if $x \notin \mathrm{domain}(f)$.
2. If $f(x)$ outputs a value, simulate E until it enumerates $f(x)$ strings.
3. Output the $f(x)$-th string enumerated by E.

The function f' is partial recursive, and A is $\mathrm{F_{PR}}$-compressible to B via f'.

For the "only if" direction, let B be infinite and RE and let E be an enumerator for B. Suppose A is $\mathrm{F_{PR}}$-compressible to B via a partial recursive function f. On input x, our f' will work as follows.

1. Simulate $f(x)$.
2. If $f(x)$ outputs a value, run E until it enumerates $f(x)$. This step may run forever if $f(x) \notin B$.
3. Suppose $f(x)$ is the lth string output by E. Then output the lth string in Σ^*.

f' is partial recursive, and A is $\mathrm{F_{PR}}$-compressible to Σ^* by f'. □

Theorem 3.2 covers the two most natural pairings of set classes with function classes: recursive sets B with $\mathrm{F_{REC}}$ compression, and RE sets B with $\mathrm{F_{PR}}$ compression. What about pairing recursive sets under $\mathrm{F_{PR}}$ compression, or RE sets under recursive compression? We note as the following theorem that one and a half of the analogous statements hold, but the remaining direction fails.

Theorem 3.3. *1. Let A and B be infinite sets and suppose that $B \in \mathrm{REC}$. Then A is $\mathrm{F_{PR}}$-compressible to B if and only if A is $\mathrm{F_{PR}}$-compressible to Σ^*.*

2. Let A and B be infinite sets with $B \in \mathrm{RE}$. If A is $\mathrm{F_{REC}}$-compressible to Σ^*, then A is $\mathrm{F_{REC}}$-compressible to B. In fact, we may even require that the compression function for A to B satisfies $f(\Sigma^*) = B$.

3. There are infinite sets A and B with $B \in \mathrm{RE}$ such that A is $\mathrm{F_{REC}}$-compressible to B but A is not $\mathrm{F_{REC}}$-compressible to Σ^*.

Proof. The first part follows immediately from Theorem 3.2, part 2. The second part follows as a corollary to the proof of Theorem 3.2, part 2. In particular, the proof of the "if" direction proves the second part, since it is clear that if f is a recursive function the f' defined there is also recursive.

The third part follows from [HR19] in which it is shown that any set in $\mathrm{RE} - \mathrm{REC}$ is not $\mathrm{F_{REC}}$-compressible to Σ^*. Thus if we let $A = B$ be any set in $\mathrm{RE} - \mathrm{REC}$, then A is $\mathrm{F_{REC}}$-compressible to B by the function $f(x) = x$ but A is not $\mathrm{F_{REC}}$-compressible to Σ^*. □

Another interesting question is how recursive compressibility to B is, or is not, linked to recursive isomorphism. Recall that two sets A and B are recursively isomorphic (notated $A \equiv_{iso} B$) if there exists a recursive bijection $f : \Sigma^* \to \Sigma^*$ with $f(A) = B$. Although recursive isomorphism of sets implies mutual compressibility to each other (Theorem 3.4), we prove (as Theorem 3.5) via a finite-injury priority argument that the converse does not hold (even when restricted to the RE sets). The argument has an interesting graph-theoretic flavor, and involves queuing infinitely many strings to be added to a set at once.

Theorem 3.4. *If $A \equiv_{iso} B$, then A is* $\mathrm{F_{REC}}$*-compressible to B and B is* $\mathrm{F_{REC}}$*-compressible to A.*

Proof. A is $\mathrm{F_{REC}}$-compressible to B by simply letting our $\mathrm{F_{REC}}$-compression function be the recursive isomorphism function f. Since each recursive isomorphism has a recursive inverse, B is $\mathrm{F_{REC}}$-compressible to A by letting our $\mathrm{F_{REC}}$-compression function be the inverse of f. □

Theorem 3.5. *There exist* RE *sets A and B such that A is* $\mathrm{F_{REC}}$*-compressible to B and B is* $\mathrm{F_{REC}}$*-compressible to A, yet $A \not\equiv_{iso} B$.*

Proof. We will prove this result via a finite-injury priority argument.

Before defining A and B, we will define a function f which will serve as both a compression function from A to B and a compression function from B to A. First, fix a recursive isomorphism between Σ^* and $\{\langle t, j, k\rangle \mid t \in \{0,1,2,3\} \wedge j, k \in \mathbb{N}\}$. Now we will define f as follows. For each $j, k \in \mathbb{N}$, let $f(\langle 3, j, k\rangle) = \langle 3, j+1, k\rangle$. For each $j, k \in \mathbb{N}$, $j > 0$, and $t \in \{0,1,2\}$, let $f(\langle t, j, k\rangle) = \langle t, j-1, k\rangle$. Finally, for each $k \in \mathbb{N}$, let $f(\langle 0,0,k\rangle) = \langle 3,0,k\rangle$, $f(\langle 1,0,k\rangle) = \langle 0,0,k\rangle$, and $f(\langle 2,0,k\rangle) = \langle 3,0,k\rangle$. Then there is a unique function $\ell : \Sigma^* \to \{0,1\}$ such that $\ell(\langle 0,0,k\rangle) = 0$ for all $k \in \mathbb{N}$ and $\ell(f(x)) = 1 - \ell(x)$. Let D_f be the directed graph with edges $(x, f(x))$. Note that ℓ is a 2-coloring of D_f if we treat the edges as being undirected. See Figure 1.

$$\begin{tikzcd}
 & & \vdots & \\
 & & \langle 3,2,k\rangle \arrow[u] & \\
 & & \langle 3,1,k\rangle \arrow[u] & \\
 & & \langle 3,0,k\rangle \arrow[u] & \\
 & \langle 0,0,k\rangle \arrow[ur] & & \langle 2,0,k\rangle \arrow[ul] \\
\langle 0,1,k\rangle \arrow[ur] & & \langle 1,0,k\rangle \arrow[ul] & \langle 2,1,k\rangle \arrow[u] \\
\langle 0,2,k\rangle \arrow[u] & & \langle 1,1,k\rangle \arrow[u] & \langle 2,2,k\rangle \arrow[u] \\
\vdots \arrow[u] & & \vdots \arrow[u] & \vdots \arrow[u]
\end{tikzcd}$$

Figure 1: D_f is the (disjoint) union, over all $k \in \mathbb{N}$, of the above graph

Call a set C a *path set* if for all $x \in C$, $f(x) \in C$ and there is exactly one $y \in C$ such that $f(y) = x$. Suppose C is a path set. Let $C_i = \{x \in C \mid \ell(x) = i\}$ for $i \in \{0,1\}$. By the path-set property of C, we have C_0 and C_1 are $\mathrm{F_{REC}}$-compressible to each other by f. Furthermore, if C is RE then so are C_0 and C_1 since $C_i = C \cap \{x \mid \ell(x) = i\}$ is the intersection of an RE set with a recursive set. Thus if we provide an enumerator for a path set C such that $C_0 \not\equiv_{iso} C_1$, we may let $A = C_0$ and $B = C_1$ and be done.

Our enumerator for C proceeds in two interleaved types of stages: printing stages P_i and evaluation stages E_i. More formally, we proceed in stages labeled E_i and P_i for $i \geq 1$, interleaved as E_1,P_1,E_2,P_2,…,E_n,P_n,… when running. We also build a set Q of elements of the form $\langle t, k\rangle$, where $t \in \{0,1,2\}$ and $k \in \mathbb{N}$. This set Q will only ever be added to as the procedure runs. In addition to Q, we maintain an integer b (initialized at 1) and a set R of elements $\langle n,k\rangle$ where $n,k \in \mathbb{N}$. If $\langle n,k\rangle \in R$ after stage E_i, it signifies that we have not yet satisfied the condition that φ_n, the nth partial recursive function, is not an isomorphism function between C_0 and C_1.

In a printing stage P_i, we do the following for every $\langle t,k\rangle$ in Q. Enumerate $\langle 3,j,k\rangle$ and $\langle t,j,k\rangle$ for all $j \leq i$. If $t=1$, additionally enumerate $\langle 0,0,k\rangle$. Looking at this procedure, we see that adding an element $\langle t,k\rangle$ to Q in some evaluation stage E_i is essentially adding an infinite path of nodes in D_f to C.

We say that a node $\langle t,j,k\rangle$ is currently *queued* at some point in the procedure if it would be enumerated if we had only printing stages from that point on. We can check if a node is queued by simulating stage P_j given the current state of Q, since no elements are ever removed from Q.

In an evaluation stage E_i, we perform the following. Add $\langle i,b\rangle$ to R. Increment b by one. For each $\langle n,k\rangle \in R$, run φ_n on $\langle 0,0,k\rangle$ for i steps. If none of these machines halt in their allotted time, end the stage. Otherwise, let n_i be the smallest number such that φ_{n_i} produced an output $w_i = \langle x_i,y_i,z_i\rangle$ on its respective input $\langle 0,0,k_i\rangle$. We now break into cases:

1. If $\ell(w_i)=0$ add $\langle 0,k_i\rangle$ to Q.
2. If $z_i \neq k_i$ and $\ell(w_i)=1$ and w_i is not currently queued, add $\langle 0,k_i\rangle$ to Q.
3. If $z_i \neq k_i$ and $\ell(w_i)=1$ and w_i is currently queued, do nothing.
4. If $z_i = k_i$ and $\ell(w_i)=1$ and $x_i=0$, add $\langle 1,k_i\rangle$ to Q.
5. If $z_i = k_i$ and $\ell(w_i)=1$ and either $x_i=1$ or $x_i=2$, add $\langle 0,k_i\rangle$ to Q.
6. If $z_i = k_i$ and $\ell(w_i)=1$ and $x_i=3$, add $\langle 2,k_i\rangle$ to Q.

Set $b=\max(k_i,z_i)+1$. Remove all pairs $\langle n,k\rangle$ with $n \geq n_i$ from R. Then for each n from n_i+1 to i, perform the following: first add $\langle n,b\rangle$ to R, then increment b by 1.

We will first prove that C is a path set. If $x \in C$, then it is printed in some printing stage P_i. By tracing the definition of f and the procedure for printing stages, one can verify that both $f(x)$ and exactly one y such that $f(y)=x$ will be printed in stage P_j for $j \geq i$. This string y will be the only one ever printed, since no two elements with the same second coordinate will ever be added to Q, as every element added to Q has the current state of b as its second coordinate, and b only ever strictly increases between additions to Q.

Let F_n be the condition that φ_n fails to be a recursive isomorphism of C_0 onto C_1. Fix n. Say during E_i we have $n_i=n$. In cases 1, 2, 4, and 5, we force φ_n to map $\langle 0,0,k_i\rangle \in C_0$ to something out of C_1. In cases 3 and 6, we force φ_n to map $\langle 0,0,k_i\rangle \notin C_0$ to something in C_1. Thus whenever in stage E_i we have $n_i=n$, condition F_n becomes satisfied, though perhaps not permanently. Specifically, in case 2, w could be printed later to satisfy some other F_m and in doing so "injure" F_n. However, note that during E_i the variable b is set to $\max(k_i,z_i)$, thus F_n can only be injured when satisfying conditions F_m for $m<n$. Pairs with first coordinate n will only ever be added to R when after satisfying some such F_m, in addition to once initially, so in total only a finite number of times. If φ_n always halts, F_n will eventually be satisfied and never injured again.

This proves that C is a path set such that $C_0 \not\equiv_{iso} C_1$. Thus C_0 and C_1 are RE sets that are $\mathrm{F_{REC}}$-compressible to each other by f, but are not recursively isomorphic. $\square$

For those interested in the issue of isomorphism in the context of complexity-theoretic functions, which was not the focus above, we mention that: Hemaspaandra, Zaki, and Zimand [HZZ96] prove that the P-rankable sets are not closed under $\equiv_{iso}^{p}$; Goldsmith and Homer [GH96] prove that the strong-P-rankable sets are closed under $\equiv_{iso}^{p}$ if and only if $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$; and [HZZ96] notes that the semistrong-P-rankable sets similarly are closed under $\equiv_{iso}^{p}$ if and only if $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$.

4 Closures and Nonclosures under Boolean Operations

We now move on to a main focus of this paper, the closure properties of the compressible and the rankable sets. We explore these properties both in the complexity-theoretic and the recursion-theoretic domains. Table 1 on page 3 summarizes our findings.

Lemma 4.1. *Let A and B be strong-P-rankable. Then $A \cup B$ is strong-P-rankable if and only if $A \cap B$ is.*

Proof. The identity $\mathrm{rank}_{A\cap B}(x)+\mathrm{rank}_{A\cup B} = \mathrm{rank}_A(x)+\mathrm{rank}_B(x)$ allows us to compute either of $\mathrm{rank}_{A\cap B}(x)$ or $\mathrm{rank}_{A\cup B}(x)$ from the other. □

Theorem 4.2. *The following conditions are equivalent:*

1. *the classes strong-P-rankable and semistrong-P-rankable are closed under intersection,*
2. *the classes strong-P-rankable and semistrong-P-rankable are closed under union, and*
3. $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$.

Proof. It was proven in [HR90] by Hemachandra and Rudich that $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$ implies P = strong-P-rankable = semistrong-P-rankable. Since P is closed under intersection and union, this shows that 3 implies 1 and 2. To show, in light of Lemma 4.1, that either 1 or 2 would imply 3, we will construct two strong-P-rankable sets whose intersection is not P-rankable unless $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$.

Let A_1 be the set of $x1y1$ such that $|x| = |y|$, x encodes a boolean formula, and y (padded with 0s so that it has length $|x|$) encodes a satisfying assignment for the formula x. Let A_0 be the set of $x1y0$ such that $|x| = |y|$, and $x1y1 \notin A_1$. Let A_2 be the set of strings $x0^{|x|+1}1$. Let $A = A_0 \cup A_1 \cup A_2$. For every x, and every y such that $|x| = |y|$, exactly one of $x1y0$ and $x1y1$ is in $A_0 \cup A_1$. Thus, for any z, we can find $\mathrm{rank}_{A_0\cup A_1}(z)$ in polynomial time. Clearly A_2 is strong-P-rankable. Since $A_0 \cup A_1$ and A_2 are disjoint, $\mathrm{rank}_{A_0\cup A_1\cup A_2}(z) = \mathrm{rank}_{A_0\cup A_1}(z) + \mathrm{rank}_{A_2}(z)$, so A is strong-P-rankable.

Let $B = \Sigma^*1$. Then $A \cap B = A_1 \cup A_2$ is the set of $x1y1$ such that y encodes a satisfying assignment for x, along with all strings $x0^{|x|+1}1$. If $A_1 \cup A_2$ were P-rankable, then we could count satisfying assignments of a formula x in polynomial time by computing $\mathrm{rank}_{A\cap B}(\mathrm{shift}(x, 1)0^{|\,\mathrm{shift}(x,1)|+1}1) - \mathrm{rank}_{A\cap B}(x0^{|x|+1}1) - 1$. Thus #SAT is polynomial-time computable and so $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$. □

Proposition 4.3. *The class strong-P-rankable is closed under complementation.*

Proof. The identity $\mathrm{rank}_A(x) + \mathrm{rank}_{\overline{A}}(x) = \mathrm{rank}_{\Sigma^*}(x)$ allows us to compute either of $\mathrm{rank}_A(x)$ or $\mathrm{rank}_{\overline{A}}(x)$ from the other. □

Corollary 4.4. *The class strong-P-rankable$^{\complement}$ is also closed under complementation.*

Lemma 4.5. *The class semistrong-P-rankable is closed under complementation if and only if semistrong-P-rankable = strong-P-rankable.*

Proof. The "if" direction follows directly from Proposition 4.3. For the "only if" direction, let A be a semistrong-P-rankable set with ranking function r_A, and suppose $\overline{A}$ is semistrong-P-rankable with semistrong ranking function $r_{\overline{A}}$. Then $\mathrm{rank}_A(x) = r_A(x)$ if $x \in A$, and equals $\mathrm{rank}_{\Sigma^*}(x) - r_{\overline{A}}(x)$ otherwise. The function r_A decides membership in A, so we can compute $\mathrm{rank}_A(x)$ in polynomial time. □

The proof of the "only if" direction of the above proof is in fact showing the following, which we will soon draw on.

Corollary (to the proof) 4.6. *If A and $\overline{A}$ are semistrong-P-rankable, then A is strong-P-rankable.*

Definition 4.7. *A set is* nongappy *if there exists a polynomial p such that, for each $n \in \mathbb{N}$, there is some element $y \in A$ such that $n \leq |y| \leq p(n)$.*

Theorem 4.8. *If the complement of each nongappy set in the class semistrong-P-rankable is itself in semistrong-P-rankable, then* $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$*. (Thus if semistrong-P-rankable is closed under complementation, then* $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$*.)*

Proof. Suppose that the complement of each nongappy set in the class semistrong-P-rankable is itself in semistrong-P-rankable.

Let A be in $\mathrm{UP} \cap \mathrm{coUP}$. Then there exists a UP machine U recognizing A, and a UP machine V recognizing $\overline{A}$. If $x \in A$, let $f(x)$ be the unique accepting path of U on input x. Otherwise, let $f(x)$ be the unique accepting path of V on input x. Choose a polynomial p such that, without loss of generality, $p(x)$ is monotonically increasing and $|f(x)| = p(|x|)$ (we may pad accepting paths with 0s to make this true).

The language $B = \{xf(x)1 \mid x \in \Sigma^*\} \cup \{x0^{p(|x|)+1} \mid x \in \Sigma^*\}$ is semistrong-P-rankable since $\mathrm{rank}_B(x0^{p(|x|)+1}) = 2\mathrm{rank}_{\Sigma^*}(x) - 1$ and $\mathrm{rank}_B(xf(x)1) = 2\mathrm{rank}_{\Sigma^*}(x)$. Also, note that B is nongappy. Since B is semistrong-P-rankable and nongappy, by our supposition we have that $\overline{B}$ is semistrong-P-rankable. By Corollary 4.6, we thus have that B is strong-P-rankable. Let x be a string, and let $y = \mathrm{shift}(x, 1)$. We can binary search on the value of rank_B in the range from $x0^{p(|x|)+1}$ to $y0^{p(|y|)+1}$ to find the first value xz where $|z| = p(|x|) + 1$ and $\mathrm{rank}_B(xz) = 2\mathrm{rank}_{\Sigma^*}(x)$. See that $f(x)$ must equal z. We then simulate U on the path z and V on the path z. Now z must be an accepting path for one of these machines, so either U accepts and $x \in A$, or V accepts and $x \notin A$. □

Theorem 4.9. *If* $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$ *then each nongappy, semistrong-P-rankable set is strong-P-rankable.*

Proof. Let A be a nongappy semistrong-P-rankable set, and let p be a polynomial such that, for each $n \in \mathbb{N}$, there is y in A such that $n \leq |y| \leq p(y)$. Let r be a polynomial-time semistrong ranking function for A. The coming string comparisons of course will be lexicographical. Let L be the set of $\langle x, b\rangle$ such that there exists at least one string in A that is less than or equal to x and b a prefix of the greatest string in A that is lexicographically less than or equal to x. L is in $\mathrm{UP} \cap \mathrm{coUP}$ by the following procedure. Let x_0 be the lexicographically first string in A. If $x < x_0$ output 0. Otherwise, guess a string $z > x$ such that $|z| \leq p(|x| + 1)$. Then guess a $y \leq x$. If y and z are in A and $r(y) + 1 = r(z)$, then we know that y and z are the (unique) strings in A that most tightly bracket x in the $\leq$ and the $>$ directions. Since $L \in \mathrm{UP} \cap \mathrm{coUP}$, it follows from our $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$ hypothesis that $L \in \mathrm{P}$. So we can in our current case build the greatest string less than or equal to x that is in A bit by bit, querying potential prefixes, in polynomial time. Since $\mathrm{rank}_A(x) = \mathrm{rank}_A(y)$, we can compute $\mathrm{rank}_A(x)$ in polynomial time for arbitrary x. □

Theorem 4.8 (by which we mean the "if-then" claim of that theorem; we are not speaking of the parenthetical remark that is the final sentence of that theorem) and Theorem 4.9 in fact are each other's converses—although this might not be immediately apparently simply from looking at the results' statements. The fact that they are each other's converses, though, follows from the equivalence of parts 2 and 3 of the following result, which makes clear that we already have implicitly established the equivalence of four statements.

Theorem 4.10. *The following are equivalent:*

1. $\mathrm{P} = \mathrm{UP} \cap \mathrm{coUP}$.
2. *The complement of each nongappy, semistrong-P-rankable set is semistrong-P-rankable.*
3. *Each nongappy, semistrong-P-rankable set is strong-P-rankable.*
4. *The complement of each nongappy, semistrong-P-rankable set is strong-P-rankable.*

Proof. Part 3 implies part 4 by Proposition 4.3. Part 4 implies part 2 since each strong-P-rankable set is semistrong-P-rankable. Part 2 implies part 1 by Theorem 4.8. Part 1 implies part 3 by Theorem 4.9. $\square$

Though we thus have established the converse of the main claim of Theorem 4.8, what we unfortunately have not proven the converse of is the parenthetical remark that is the final sentence of Theorem 4.8. Theorem 4.9, however, establishes in some sense a weakened (due to the "nongappy" restriction; see also the comments in the caption of Table 1) version of a converse.

The next three theorems are about intersections, unions, and complements of P-rankable sets, with regard to compressibility.

Theorem 4.11. *There exist P-rankable sets A and B such that $A \cap B$ is infinite but not* $\mathrm{F_{PR}}$*-compressible.*

Proof. This paragraph gives an informal sense of the approach used in this proof. This proof will, by an inductive stage construction, define sets A and B in such a way as to ensure that A and B are both P-rankable and $A \cap B$ is infinite. However, the ith stage of our inductive construction will ensure that the partial recursive function computed by the ith Turing machine is not a compression function for $A \cap B$; and so, since each Turing machine will have been eliminated from being able to provide a compression function for $A \cap B$, the construction will ensure that $A \cap B$ is not $\mathrm{F_{PR}}$-compressible. Let us now turn to executing this approach.

In particular, in this proof we will define a set A not containing the empty string and satisfying the condition that for all $x \in \Sigma^*$, exactly one of $x0$ and $x1$ is in A. Then clearly A is P-rankable by a compression function sending $x1$ and $x0$ to $\mathrm{rank}_{\Sigma^*}(x)$. Let A_0 and B_0 be empty, and let $m_0 = \epsilon$. We will define A_i, B_i, and m_i inductively for $i > 0$. Let φ_i be the ith Turing machine in some enumeration of all Turing machines.

1. Suppose that φ_i is defined on $m_{i-1}0$, and that for all $x \in (A_{i-1} \cap B_{i-1}) \cup \{y \mid y > m_{i-1}0\}$ we have $\varphi_i(x) \neq \varphi_i(m_{i-1}0)$. In this case, we set $A_i = A_{i-1} \cup \{m_{i-1}0, \mathrm{shift}(m_{i-1}, 1)0\}$ and $B_i = B_{i-1} \cup \{m_{i-1}1, \mathrm{shift}(m_{i-1}, 1)0\}$ and set $m_i = \mathrm{shift}(m_{i-1}, 2)$, so that neither $m_{i-1}0$ nor $m_{i-1}1$ is in $A_i \cap B_i$. Note that $\mathrm{shift}(m_{i-1}, 1)0 \in A_i \cap B_i$ but $\mathrm{shift}(m_{i-1}, 1)0 \notin A_{i-1} \cap B_{i-1}$.

2. Suppose φ_i is either undefined on $m_{i-1}0$, or that for some $x \in A_{i-1} \cap B_{i-1}$ we have $\varphi_i(x) = \varphi_i(m_{i-1}0)$. In this case, set $A_i = A_{i-1} \cup \{m_{i-1}0\}$, $B_i = B_{i-1} \cup \{m_{i-1}0\}$, and $m_i = \mathrm{shift}(m_{i-1}, 1)$. Note in particular that x and $m_{i-1}0$ are both in $A_i \cap B_i$ and lexicographically less than m_i0, and take the same value under φ_i.

3. Suppose that the above cases do not hold and there is some $x > m_{i-1}1$ such that $\varphi_i(x) = \varphi_i(m_{i-1}0)$. Let y be the lexicographically largest string such that $y0 \leq x$, and let $m_i = \mathrm{shift}(y, 1)$. Set $A_i = A_{i-1} \cup \{z0 \mid m_{i-1} \leq z < y\} \cup \{x\}$ and $B_i = B_{i-1} \cup \{z0 \mid m_{i-1} \leq z < y\} \cup \{x\}$. Note in particular that x and $m_{i-1}0$ are both in $A_i \cap B_i$ and lexicographically less than m_i0, and take the same value under φ_i.

Finally, let $A = \bigcup_{i \geq 0} A_i$ and $B = \bigcup_{i \geq 0} B_i$. Notice that stage i only adds elements to A_i or B_i that are lexicographically greater than or equal to $m_{i-1}0$, so if $x < m_i0$ and $x \notin A_i \cap B_i$, then $x \notin A \cap B$. In case 1, we see that φ_i fails to be surjective (i.e, onto Σ^*) when restricted to $A \cap B$, since there is no $x < m_i0$ in $A \cap B$ mapping to $\varphi_i(m_{i-1}0)$, and also no $x > m_{i-1}1$ mapping to $\varphi_i(m_{i-1}0)$, and neither $m_{i-1}0$ nor $m_{i-1}1$ is in $A \cap B$. In case 2, we see either that φ_i is undefined on an element of $A \cap B$ or that two elements of $A \cap B$ map to the same element. In case 3, we see that two elements in $A \cap B$ map to the same element under φ_i. Thus φ_i fails to compress $A \cap B$, and no partial recursive function can compress $A \cap B$. The set $A \cap B$ is infinite since at least one new element is added to $A_i \cap B_i$ during stage i. We also maintain the condition that, for all $x < m_i$, exactly one of $x0$ and $x1$ is in A_i (resp., B_i). Each A_i (resp. B_i) consists of exactly all strings in A (resp. B) lexicographically less than m_i0, and so clearly since this statement holds for each A_i (resp. B_i) it holds for all of A (resp. B) as well. Thus A and B are P-rankable, but their intersection is not $\mathrm{F_{PR}}$-compressible. $\square$

Theorem 4.12. *There exist infinite P-rankable sets A and B such that $A \cup B$ is not $\mathrm{F_{PR}}$-compressible.*

Proof. Let a set A not containing the empty string satisfy the condition that for all $x \in \Sigma^*$, exactly one of $x0$ and $x1$ is in A. Then clearly A is P-rankable by a function sending $x1$ and $x0$ to $\mathrm{rank}_{\Sigma^*}(x)$.

Let A_0 and B_0 be empty, and let $m_0 = \epsilon$. We will construct A_i, B_i, and m_i inductively for $i > 0$. Let φ_i be the ith Turing machine in some enumeration of all Turing machines.

1. Suppose that φ_i is defined on $m_{i-1}0$, and that for all x in $A_{i-1} \cup B_{i-1} \cup \{y \mid y > m_{i-1}0\}$ we have $\varphi_i(x) \neq \varphi_i(m_{i-1}0)$. In this case, we set $A_i = A_{i-1} \cup \{m_{i-1}1\}$ and $B_i = B_{i-1} \cup \{m_{i-1}1\}$, and we set $m_i = \mathrm{shift}(m_{i-1}, 1)$.

2. Suppose φ_i is either undefined on $m_{i-1}0$, or that for some $x \in A_{i-1} \cup B_{i-1}$ we have $\varphi_i(x) = \varphi_i(m_{i-1}0)$. In this case, set $A_i = A_{i-1} \cup \{m_{i-1}0\}$, $B_i = B_{i-1} \cup \{m_{i-1}0\}$, and $m_i = \mathrm{shift}(m_i, 1)$.

3. Suppose that the above cases do not hold and there is some $x \geq m_{i-1}0$ such that $\varphi_i(x) = \varphi_i(m_{i-1}0)$. Let m_i be the lexicographically smallest string such that $m_i0 > x$. Set $A_i = A_{i-1} \cup \{y0 \mid m_{i-1} \leq y < m_i\}$ and $B_i = B_{i-1} \cup \{y1 \mid m_{i-1} \leq y < m_i\}$. Note that both $m_{i-1}0$ and x are in $A_i \cup B_i$.

Finally, let $A = \bigcup_{i \geq 0} A_i$ and $B = \bigcup_{i \geq 0} B_i$. Stage i only adds elements to A_i or B_i that are lexicographically greater than or equal to $m_{i-1}0$, so if $x < m_i0$ and $x \notin A_i \cup B_i$, then $x \notin A \cup B$. In case 1, we see that φ_i fails to be surjective (i.e., onto Σ^*) when restricted to $A \cup B$, since there is no $x < m_{i-1}0$ in $A \cup B$ mapping to $\varphi_i(m_{i-1}0)$, and also no $x > m_i0$ mapping to $\varphi_i(m_{i-1}0)$, so no element in $A \cup B$ compresses to $\varphi_i(m_{i-1}0)$. In case 2, we see either that φ_i is undefined on $m_{i-1}0 \in A \cup B$ or that $m_{i-1}0$ and some other element in $A \cup B$ map to the same element, so injectivity when restricted to $A \cap B$ fails. Similarly, in case 3, we see that $m_{i-1}0$ and some other element in $A \cup B$ will map to the same value under φ_i. Thus for all i we see that φ_i fails to compress $A \cup B$, and so no partial recursive function compresses $A \cup B$. Note that we maintain the condition that for all $x < m_i$, exactly one of $x0$ and $x1$ is in A_i (resp., B_i). This condition holds in A (resp., B), and this property carries over to A and B as well. Each A_i (resp. B_i) consists of exactly all strings in A (resp. B) lexicographically less than m_i0, and so clearly since this statement holds for each A_i (resp. B_i) it holds for all of A (resp. B) as well. Thus A and B are P-rankable, but $A \cup B$ is not $\mathrm{F_{PR}}$-compressible. $\square$

Theorem 4.13. *There exists an infinite P-rankable set whose complement is infinite but not* $\mathrm{F_{PR}}$-*compressible.*

Proof. We will construct a set A consisting of strings with length at least 2, with the property that for every $x \in \Sigma^*$, exactly one of $x00$, $x01$, $x10$, and $x11$ is in A. Clearly A will be infinite, and its complement is infinite as well. Also, A will be P-rankable by sending $x00$, $x01$, $x10$ and $x11$ to $\mathrm{rank}_{\Sigma^*}(x)$. Let $A_0 = 0$ and $m_0 = \epsilon$. We will construct A_i and m_i inductively for $i > 0$. Let φ_i be the ith Turing machine in some enumeration of all Turing machines.

1. Suppose φ_i halts on $m_{i-1}00$, and there is no $x \in \overline{A_{i-1}}$ where $x < m_{i-1}00$ such that $\varphi_i(x) = \varphi(m_{i-1}00)$, and that there is no $x > m_{i-1}00$ such that $\varphi_i(x) = \varphi(m_{i-1}00)$. Then set $A_i = A_{i-1} \cup \{m_i00\}$ and set $m_i = \mathrm{shift}(m_{i-1}, 1)$.

2. Suppose φ_i is undefined on $m_{i-1}00$, or that there is some $x < m_{i-1}00$ where $x \in \overline{A_{i-1}}$ and $\varphi(x) = \varphi(m_{i-1}00)$. Then set $A_i = A_{i-1} \cup \{m_{i-1}01\}$ and set $m_i = \mathrm{shift}(m_{i-1}, 1)$.

3. Suppose that the above cases do not hold, φ_i is defined on $m_{i-1}00$, and $\varphi_i(m_{i-1}00) = \varphi(x)$ for some $x \in \{m_{i-1}01,\ m_{i-1}10,\ m_{i-1}11\}$. Then set $A_i = A_{i-1} \cup \{z\}$, where z is a fixed arbitrary element in $\{m_{i-1}01,\ m_{i-1}10,\ m_{i-1}11\} - \{x\}$, and set $m_i = \text{shift}(m_{i-1},\ 1)$. Note that $m_{i-1}00$ and x are both in $\overline{A}$ and below m_i00, and take the same value under φ_i.

4. Suppose the above cases do not hold and φ_i is defined on $m_{i-1}00$ and $\varphi_i(m_{i-1}00) = \varphi(x)$ for some $x > m_i11$. Let y be equal to x without its last two characters, and set $m_i = \text{shift}(y, 1)$. Let $A_i = A_{i-1} \cup \{m_{i-1}01\} \cup \{z11 \mid m_i < z < y\} \cup \{w\}$, where w is some element in $\{y00, y01, y10, y11\} - \{x\}$. Note that $m_{i-1}00$ and x are both in $\overline{A_i}$ and lexicographically less than m_i00, and take the same value under φ_i.

Finally, let $A = \bigcup_{i \geq 0} A_i$. Notice that stage i adds to A only elements that are lexicographically greater than or equal to $m_{i-1}00$, so if $x < m_i00$ and $x \in \overline{A_i}$, then $x \in \overline{A}$. In case 1, we see that φ_i fails to be surjective (i.e., onto Σ^*) when restricted to $\overline{A}$, since there is no $x < m_i00$ in $\overline{A}$ mapping to $\varphi_i(m_{i-1}00)$, and also no $x \geq m_i0$ mapping to $\varphi_i(m_{i-1}00)$. In case 2, either φ_i does not halt on $m_i00 \in \overline{A}$ or there are two elements in $\overline{A}$ that take the same value under φ_i. In cases 3 and 4, we see that there are two elements in $\overline{A}$ that take the same value under φ_i. Thus in all cases φ_i fails to compress $\overline{A}$ and so $\overline{A}$ is not $\mathrm{F_{PR}}$-compressible. ☐

Theorem 4.14. *There exist sets A and B that are not* $\mathrm{F_{PR}}$*-compressible, yet $A \cup B$ is strong-P-rankable. In addition, there exist sets A and B that are not* $\mathrm{F_{PR}}$*-compressible, yet $A \cap B$ is strong-P-rankable.*

Proof. Let C be a set such that $A = C0 \cup \Sigma^*1$ is not $\mathrm{F_{PR}}$-compressible. Such a set can be constructed using a similar method to those of Theorems 4.12, 4.13, and 4.14. Then $B = C1 \cup \Sigma^*0$, $A' = C00 \cup \Sigma^*1$, and $B' = C10 \cup \Sigma^*1$ are all also not $\mathrm{F_{PR}}$-compressible, since clearly they are all recursively isomorphic. Note that $A \cup B = \Sigma^*$ and $A' \cap B' = \Sigma^*1$, both of which are strong-P-rankable. ☐

5 Additional Closure and Nonclosure Properties

How robust are the polynomial-time and recursion-theoretically compressible and the rankable sets? Do sets lose these properties under join, or subtraction, addition, or (better yet) symmetric difference with finite sets? Or even with sufficiently nice infinite sets? The following section addresses these questions.

5.1 Complexity-Theoretic Results

We focus on the join (aka disjoint union), giving a full classification of the closure properties (or lack thereof) of the P-rankable, semistrong-P-rankable, and strong-P-rankable sets, as well as their complements, under this operation. The literature is not consistent as to whether the low-order or high-order bit is the "marking" bit for the join. Here, we follow the classic computability texts of Rogers [Rog67] and Soare [Soa87] and the classic structural-complexity text of Balcázar, Díaz, Gabarró [BDG95], and define the join using low-order-bit marking: The join of A and B, denoted $A \oplus B$, is $A0 \cup B1$, i.e., $\{x0 \mid x \in A\} \cup \{x1 \mid x \in B\}$. For classes invariant under reversal, which end is used for the marking bit is not important (in the sense that the class itself is closed under upper-bit-marked join if and only if it is closed under lower-bit-marked join). However, the placement of the marking bit potentially matters for ranking-based classes, since those classes are based on lexicographical order.

The join is such a basic operation that it seems very surprising that any class would not be closed under it, and it would be even more surprising if the join of two sets that lack some nice organizational property can have that property. After all, the join of two sets is the least upper bound for them with respect to $\leq_m^p$ [Sch86] and, informally put, the join in some sense captures the power-as-a-target-of-reductions of both sets.

Nonetheless, we will now prove, as Theorem 5.1, that from a weak complexity-theoretic hypothesis it follows that the join of two sets in fact can, regarding rankability, be "simpler" than either of the sets. In particular, our Theorem 5.1 shows that if $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$, then there are two sets that are not P-rankable yet their join is P-rankable (and indeed is even strong-P-rankable). (We mention in passing that there is in the literature already an example of behavior showing that the join can be "simpler" than either of the joined sets. In particular, with EL_2 denoting the second level of the so-called extended low hierarchy [BBS86], Hemaspaandra et al. [HJRW98] proved that there are two sets neither of which belongs to EL_2 yet the sets' join does belong to EL_2, i.e., $(\mathrm{EL}_2)^{\complement}$ is not closed under the join.) We now state and prove Theorem 5.1, which is slightly broader than just indicated, since it addresses not only the join but also intersection and union.

Theorem 5.1. *If $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$ then there exist sets $A \in \mathrm{P}$ and $B \in \mathrm{P}$ that are not P-rankable yet $A \cap B$, $A \cup B$, and $A \oplus B$ are strong-P-rankable.*

Proof. This paragraph gives an informal sense of this proof's approach. In this proof we construct a set A_1 whose members, in a way that we will make formal in the next paragraph, are encodings of a satisfiable boolean formula accompanied by a satisfying assignment of the formula. We will build our set A (of the theorem's statement) by unioning A_1 with a collection of strings that we will call beacons, and that are set up in such a way that subtracting the ranks of certain beacons would let us determine the number of satisfying assignments of formulas. In particular, we will ensure that A will thus be such that if we were able to efficiently rank A, then we could efficiently compute the number of satisfying assignments of boolean formulas. The set B (which is not formally the complement of A, but is in some sense a quasi-complement of A) is constructed similarly, but in a way that helps us ensure that $A \cup B$, $A \cap B$, and $A \oplus B$ are easily seen to be strong-P-rankable.

As in the rest of the paper, $\Sigma = \{0, 1\}$. Let $A_1 = \{\alpha 01 \beta \mid \alpha, \beta \in \Sigma^* \wedge |\alpha| = |\beta| \wedge \alpha$ is a valid encoding of boolean formula F that has (without loss of generality) $k \leq |\alpha|$ variables, the first k bits of β encode a satisfying assignment of F, and the rest of the $|\beta| - k$ bits of β are $0\}$. Note that given a string $x = \alpha 01 \beta \in A_1$, we can unambiguously extract α and β because they must have length $(|x| - 2)/2$. Let $B_1 = \{\alpha 01 \beta \mid \alpha, \beta \in \Sigma^* \wedge |\alpha| = |\beta| \wedge \alpha 01 \beta \notin A_1\}$. Let $\mathit{Beacons} = \{\alpha 000^{|\alpha|} \mid \alpha \in \Sigma^*\} \cup \{\alpha 110^{|\alpha|} \mid \alpha \in \Sigma^*\}$. Similarly to A_1, strings in B_1 and *Beacons* can be parsed unambiguously. Let $A = A_1 \cup \mathit{Beacons}$. Let $B = B_1 \cup \mathit{Beacons}$. Note that A and B are both in P because checking if an assignment satisfies a boolean formula is in P and *Beacons* is clearly in P.

We will now demonstrate that if either A or B were P-rankable, then #SAT would be in polynomial-time computable. Suppose that A is P-rankable and let f be a polynomial-time ranking function for A. Let α be a string encoding a boolean formula F. Then we can compute $j = f(\alpha 110^{|\alpha|}) - f(\alpha 000^{|\alpha|})$ in polynomial time. Both $\alpha 110^{|\alpha|}$ and $\alpha 000^{|\alpha|}$ are in *Beacons* and thus in A, so f gives a true ranking for these values. Every string in A between (and not including) these *Beacons* strings is from A_1 and thus represents a satisfying assignment for F, and every satisfying assignment for F is represented by a string between these *Beacons* strings. Because the last $|\beta| - k$ bits of β are 0, where k is the number of variables in F, each satisfying assignment for F is represented exactly once between the two *Beacons* strings. Thus $j - 1$ is the number of satisfying assignments of F. We can compute j in polynomial time, so #SAT is polynomial-time computable and thus $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$, contrary to our $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$ hypothesis.

Now suppose that B is P-rankable and similarly to before we will let f be the P-time ranking function for it. Again we will let α be the encoding for some boolean formula F and $j = f(\alpha 110^{|\alpha|}) - f(\alpha 000^{|\alpha|})$. In this case the strings in B between $\alpha 110^{|\alpha|}$ and $\alpha 000^{|\alpha|}$ are the strings of the form $\alpha 01 \Sigma^{|\alpha|}$ except for those that are in A_1 (and recall that those that are in A_1 are precisely the padded-with-0s satisfying assignments for F). Because we know the number of strings of the form $\alpha 01 \Sigma^{|\alpha|}$, we can again find the number of satisfying assignments for F. Namely, we have that $j = 1 + 2^{|\alpha|} - s$, where s is the number of satisfying assignments of F. Thus if B is P-rankable, then we can find s in polynomial time and thus $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$, contrary to our $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$ hypothesis.

Finally, we show that $A \cup B$, $A \cap B$, and $A \oplus B$ are strong-P-rankable. The set $A \cap B$ is simply *Beacons*, which is strong-P-rankable as follows. Any string lexicographically below 00 has rank 0. For any $\alpha \in \Sigma^*$, the rank of $\alpha 000^{|\alpha|}$ is $2\mathrm{rank}_{\Sigma^*}(\alpha) - 1$ and the rank of $\alpha 110^{|\alpha|}$ is $2\mathrm{rank}_{\Sigma^*}(\alpha)$. For every other string, it is easy to find the lexicographically greatest string in *Beacons* that is lexicographically less than the given string in polynomial time, and so it is possible to rank the string in polynomial time.

The set $A \cup B = \{\alpha 01 \beta \mid \alpha \in \Sigma^* \wedge \beta \in \Sigma^* \wedge |\alpha| = |\beta|\} \cup \{\alpha 000^{|\alpha|} \mid \alpha \in \Sigma^*\} \cup \{\alpha 110^{|\alpha|} \mid \alpha \in \Sigma^*\}$, and is also strong-P-rankable, as follows. Any string lexicographically below 00 has rank 0. For any $\alpha \in \Sigma^*$, the rank of $\alpha 000^{|\alpha|}$ is $1 + \sum_{x <_{lex} \alpha}(2^{|x|} + 2)$, where $x <_{lex} \alpha$ denotes that x is lexicographically less than α. Note that although the sum is over an exponentially sized set, it still can be computed in polynomial time because the summands depend only on the length of the element in the set. Let $b(x)$ be the number of strings lexicographically less than α but with the same length as α. Then we have that $1 + \sum_{x <_{lex} \alpha}(2^{|x|} + 2) = 1 + b(\alpha)(2^{|\alpha|} + 2) + \sum_{i=0}^{|\alpha|-1}(2^i(2^i + 2))$.

The rank of $\alpha 110^{|\alpha|}$ is $\sum_{x \leq_{lex} \alpha}(2^{|x|} + 2)$, where $x \leq_{lex} \alpha$ denotes that x is lexicographically less than or equal to α. For any $\alpha, \beta \in \Sigma^*$ where $|\alpha| = |\beta|$, the rank of $\alpha 01 \beta$ is $b(\beta) + 2 + \sum_{x <_{lex} \alpha}(2^{|x|} + 2)$, where n is the integer such that β is the nth string of its length. As above, each term is only dependent on the length of x, and is computable in polynomial time. For any other not string in $A \cup B$, it is easy to find the greatest string in $A \cup B$ lexicographically less than the given string in polynomial time, and thus it is easy to rank that string.

We can show that $A \oplus B = \{a0 \mid a \in A\} \cup \{b1 \mid b \in B\}$ is strong-P-rankable simply by using the already-established facts that both $A \cup B$ and $A \cap B$ are strong-P-rankable and that both A and B are in P. The rank of ϵ is 0. The rank of 0 is 1 if $\epsilon \in A$ and otherwise is 0. For $x \in \Sigma^*$, we have $\mathrm{rank}_{A \oplus B}(x1) = \mathrm{rank}_{A \cup B}(x) + \mathrm{rank}_{A \cap B}(x)$. For $x \neq \epsilon$, we have $\mathrm{rank}_{A \oplus B}(x0) = \mathrm{rank}_{A \oplus B}(x1) - \delta_B(x)$, where $\delta_B(x) = 1$ if and only if $x \in B$. □

Theorem 5.2. *The following are equivalent:*

1. *strong-P-rankable*$^{\complement}$ *is closed under join,*

2. *semistrong-P-rankable*$^{\complement}$ *is closed under join, and*

3. $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$.

Proof. Theorem 5.1 shows that either of 1 or 2 would imply 3. Now we show that 3 implies 1 and 2, or equivalently the negation of either 1 or 2 would imply the negation of 3. Suppose that strong-P-rankable$^{\complement}$ (resp., semistrong-P-rankable$^{\complement}$) is not closed under join. Then there are two sets A and B that are in strong-P-rankable$^{\complement}$ (resp., semistrong-P-rankable$^{\complement}$) but $A \oplus B$ is strong-P-rankable (resp., semistrong-P-rankable$^{\complement}$). Then both A and B are in P. This is because $A \oplus B \in \mathrm{P}$ and to test x for membership in A, for example, we can just test $x0$ for membership in $A \oplus B$. It was shown by Hemachandra and Rudich [HR90] that $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}}$, P = semistrong-P-rankable, and P = strong-P-rankable are equivalent. Since A and B are in P but not strong-P-rankable (resp., semistrong-P-rankable), P $\neq$ strong-P-rankable (respectively P $\neq$ semistrong-P-rankable) and thus $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$. □

Theorem 5.3. *The class P-rankable*$^{\complement}$ *is not closed under join.*

Proof. Let A by any language that is not P-rankable and whose complement is not P-rankable. An example of such a set is $A = \{x000 \mid x \in \Sigma^*\} \cup \{x001 \mid x \in B\} \cup \{x010 \mid x \in \Sigma^*\} \cup \{x100 \mid x \in B\}$, where B is any undecidable set. This set is not even $\mathrm{F_{REC}}$-rankable. Note $x \in B$ if and only if

$\mathrm{rank}_A(x010) - \mathrm{rank}_A(x000) > 1$, so if A were $\mathrm{F_{REC}}$-rankable, we could decide B. Similarly, $x \notin B$ if and only if $\mathrm{rank}_{\overline{A}}(x101) - \mathrm{rank}_{\overline{A}}(x011) > 1$, so if $\overline{A}$ were $\mathrm{F_{REC}}$-rankable, B would be decidable, but this is a contradiction.

Then $A \oplus \overline{A} = A0 \cup \overline{A}1$ is P-rankable. It can be ranked by any function mapping $x0$ and $x1$ to $\mathrm{rank}_{\Sigma^*}(x)$. □

Theorem 5.4. *The class P-rankable is not closed under join.*

Proof. Let A be some undecidable set. Let $A' = A \oplus \overline{A}$. Then A' is P-rankable by any function mapping $x0$ and $x1$ to $\mathrm{rank}_{\Sigma^*}(x)$. Now let $B = \Sigma^* \oplus A'$. Then B is the join of two P-rankable sets. Suppose B were P-rankable, then we can query $\mathrm{rank}_B(x0)$ for all strings x. If $\mathrm{rank}_B(x0) + 2 = \mathrm{rank}_B(\mathrm{shift}(x, 1)0)$, we know that $x1 \in B$, and thus $x \in A'$. Otherwise, $x1 \notin B$ so $x1 \notin A'$. Since we can test membership in A', we can test membership of x in A by asking whether $x0 \in A'$. This is a contradiction as A was assumed undecidable; thus B cannot be P-rankable. □

Theorem 5.5. *The class strong-P-rankable is closed under join.*

Proof. Let A and B be strong-P-rankable. The rank of $x0$ in $A \oplus B$ is $\mathrm{rank}_A(x) + \mathrm{rank}_B(\mathrm{shift}(x, -1))$, and the rank of $x1$ is $\mathrm{rank}_A(x) + \mathrm{rank}_B(x)$. The rank of ϵ is 0. All of these values can clearly be computed in polynomial time so $A \oplus B$ is strong-P-rankable. □

Theorem 5.6. *The class semistrong-P-rankable is closed under complementation if and only if it is closed under join.*

Proof. Suppose semistrong-P-rankable is closed under complementation. Then semistrong-P-rankable is equal to strong-P-rankable, so semistrong-P-rankable is closed under join.

Now suppose semistrong-P-rankable is closed under join. Let set A be semistrong-P-rankable by ranking function h. Let $X = \Sigma^* \oplus A$. Then X is the join of two semistrong-P-rankable sets and thus is semistrong-P-rankable by some ranking function f. The ranking function for $\overline{A}$ does the following. Given x, if $h(x)$ returns a rank (rather than an indication that $x \notin A$) then return an indication that $x \notin \overline{A}$. Otherwise let $y = \mathrm{shift}(x, 1)$ and return $2\mathrm{rank}_{\Sigma^*}(x) + 1 - f(y0)$. There are a total of $2\mathrm{rank}_{\Sigma^*}(x) + 2$ strings lexicographically less than or equal to $y0$ in Σ^*. All those missing in X correspond to either ϵ or strings not in A that are strictly less than y. Since $y0 \in X$, we know that $f(y0)$ of these are in X. The rest are in $\Sigma^* - X = \{x1 \mid x \in \overline{A}\} \cup \{\epsilon\}$. Thus the number of strings in $\overline{A}$ below x is $2\mathrm{rank}_{\Sigma^*}(x) + 1 - f(y0)$. Thus $\overline{A}$ is semistrong-P-rankable, so semistrong-P-rankable is closed under complementation. □

Theorem 5.7. *The class P-compressible′ is closed under join.*

Proof. Let A and B be two P-compressible′ sets. Let f and g be the compression functions for A and B respectively. Let $h(x0) = f(x)0$ and $h(x1) = \mathrm{shift}(f(x)0, -1)$. Now h is a compression function for $A \oplus B$ since the image of h restricted to $A0$ is Σ^*0, and the image of h restricted to $B1$ is $\Sigma^*1 \cup \{\epsilon\}$. Each of $A0$ and $B0$ maps injectively because f and g are compression functions, and together they map injectively on all of $A \oplus B$ to all of Σ^*. The function h is clearly polynomial time, and so $A \oplus B$ is P-compressible′. □

5.2 Recursion-Theoretic Results

Theorem 5.8.

1. *If A is an $\mathrm{F_{REC}}$-rankable set, $B_1 \subseteq A$ is a recursive set, and $B_2 \subseteq \overline{A}$ is a recursive set, then $A \triangle (B_1 \cup B_2)$ (equivalently, $(A - B_1) \cup B_2$) is $\mathrm{F_{REC}}$-rankable.*

2. *If A is $\mathrm{F_{REC}}$-compressible, $B_1 \subseteq A$ is recursive, and $A - B_1$ has an infinite* RE *subset, then $A - B_1$ is $\mathrm{F_{REC}}$-compressible.*

3. *If A is an* $\mathrm{F_{REC}}$*-compressible set and $B_2 \subseteq \overline{A}$ is a recursive set, then $A \cup B_2$ is an* $\mathrm{F_{REC}}$*-compressible set.*

Proof. For the first part of this theorem, let f be an $\mathrm{F_{REC}}$-ranking function for A. Since B_1 and B_2 are recursive, their ranking functions rank_{B_1} and rank_{B_2} are in $\mathrm{F_{REC}}$. Our $\mathrm{F_{REC}}$ ranking function for $A \bigtriangleup (B_1 \cup B_2)$ is $f'(x) = f(x) + \mathrm{rank}_{B_2}(x) - \mathrm{rank}_{B_1}(x)$. This directly accounts for the additions and deletions done by B_1 and B_2.

We now prove the second part of the theorem. The statement is clearly true if B_1 is finite, even in the case that $A - B_1$ does not have an infinite RE subset (as long as $A - B_1$ is still infinite). This is because the image of $A - B_1$ under a compression function for A is cofinite, and cofinite sets are compressible. Thus composing a compression function for the cofinite image of $A - B_1$ with a compression function for A, we obtain a compression function for $A - B_1$.

So from this point on we assume that B_1 is infinite. Let h be an $\mathrm{F_{REC}}$-compression function for A. By the hypothesis of the theorem, there is an infinite RE subset of $A - B_1$, call it C. Since every infinite RE set has an infinite recursive subset, let $B_2 \subseteq C$ be infinite and recursive. Let $b_1 < b_2 < b_3 < \cdots$ be the elements in B_1, and let $c_1 < c_2 < c_3 < \cdots$ be the elements in B_2. Consider the following function.

$$g(x) = \begin{cases} x & \text{if } x \notin B_1 \cup B_2, \\ \epsilon & \text{if } x \in B_1, \\ b_{\lceil i/2 \rceil} & \text{if } x = c_i \text{ and } i \text{ is odd, and} \\ c_{i/2} & \text{if } x = c_i \text{ and } i \text{ is even.} \end{cases}$$

Let $f(x) = h(g(x))$. We claim that f is a compression function for $A - B_1$. We do this by showing g is a compression function for $A - B_1$ onto A, since we already know that h compresses A to Σ^*. See that g is the identity on $A - (B_1 \cup B_2)$. See also that $g(B_2) = B_1 \cup B_2$ injectively and surjectively. Since $(A - B_1) - B_2$ and B_2 are disjoint and have disjoint images, and since g is injective and surjective on both these domains onto their respective images, it follows that g is injective and surjective on $A - B_1$ to the image $g((A - B_1) - B_2) \cup g(B_2) = A$. Thus g is a compression function for $A - B_1$ to A, and h is a compression function for A to Σ^*, so f is a compression function for $A - B_1$ to Σ^*. In other words, $A - B_1$ is $\mathrm{F_{REC}}$-compressible.

We now prove the third part of the theorem. Let f be an $\mathrm{F_{REC}}$-compression function for A. If B_2 is finite, our $\mathrm{F_{REC}}$ compression function for $A \cup B_2$ is $f'(x) = \mathrm{shift}(f(x), \|B_2\|)$ for $x \notin B_2$ and $f'(x) = \mathrm{shift}(\epsilon, \mathrm{rank}_{B_2}(x))$ for $x \in B_2$.

On other hand, if B_2 is infinite, let g be an $\mathrm{F_{REC}}$ compression function for B_2, e.g., g can be taken to be (recall that B_2 is recursive) defined by $g(x)$ being the $\max(\mathrm{rank}_{B_2}(x), 1)$-st string in Σ^*. We define $f'(x)$ as follows. (Recall that for us Σ is always fixed as being $\{0, 1\}$.) If $x \notin B_2$ then $f'(x) = 1f(x)$ (i.e., $f(x)$ prefixed with a one). If $x \in B_2$ and $g(x) = \epsilon$ then $f'(x) = \epsilon$. And, finally, if $x \in B_2$ and $g(x) \neq \epsilon$ then $f'(x) = 0\,\mathrm{shift}(g(x), -1)$. (The shift-by-one treatment of the $x \notin B$ case is because we must ensure that ϵ is mapped to by some string in $A \cup B_2$.) Now, f' maps $A \cup B$ bijectively onto Σ^*, so f' is an $\mathrm{F_{REC}}$ compression function for $A \cup B$, so $A \cup B$ is $\mathrm{F_{REC}}$-compressible. □

Corollary 5.9. 1. *The class of* $\mathrm{F_{REC}}$*-rankable sets is closed under symmetric difference with finite sets (and thus also under removal of finite sets and under addition of finite sets).*

2. *The class of* $\mathrm{F_{REC}}$*-compressible sets is closed under removal of finite sets and under addition of finite sets.*

6 Conclusions

Taking to heart the work in earlier papers that views as classes the collections of sets that have (or lack) rankability/compressibility properties, we have studied whether those classes are closed under the most important boolean and other operations. For the studied classes, we in almost every case were able to prove that they are closed under the operation, or to prove that they are not closed under the operation, or to prove that whether they are closed depends on well-known questions about the equality of standard complexity classes. Additionally, we have introduced the notion of compression onto a set and have showed the robustness of compression under this notion, as well as the limits of that robustness.

As a final comment, we mention that the results of Hemaspaandra and Rubery [HR19] and the results of the present paper all relativize in a straightforward manner.[1] We include a few examples. This justifies our limitation to $\mathrm{F}_{\mathrm{REC}}$ and F_{PR}: By relativization, we get analogous results about more powerful function classes, such as F_{Δ_2}.[2]

Theorem 6.1. *For each $i \geq 1$, $\Delta_i = \Sigma_i \cap \mathrm{F}_{\Delta_i}$-compressible$'$.*

Proof. Relativization of [HR19, Theorem 5.3] (see also [GHK92]). □

Since, for $i \geq 1$ $\mathrm{F}_{\Delta_i} \supseteq \mathrm{F}_{\mathrm{REC}}$, we get the following easy corollary.

Corollary 6.2. *For each $i \geq 1$, $\Sigma_i \cap \mathrm{F}_{\mathrm{REC}}$-compressible$' \subseteq \Delta_i$.*

Theorem 6.3. *For each $i \geq 1$, $\Pi_i \cap \mathrm{F}_{\Delta_i}$-rankable $= \Pi_i \cap \mathrm{F}_{\mathrm{PR}}^{\Sigma_{i-1}}$-rankable.*

Proof. Relativization of [HR19, Theorem 4.6]. □

Theorem 6.4. *For each $i \geq 1$, there exist Σ_i sets A and B such that A is F_{Δ_i}-compressible to B and B is F_{Δ_i}-compressible to A, yet $A \not\equiv^i_{iso} B$.*

Proof. Relativization of Theorem 3.5. □

Acknowledgments. We are deeply grateful to the anonymous LATA 2019 [AHMR19] and JCSS referees for helpful comments, suggestions, and corrections.

References

[Aar20] S. Aaronson. MIP* = RE. Shtetl-Optimized (the blog of Scott Aaronson), www.scottaaronson.com/blog/?p=4512, 2020.

[AHMR16] J. Abascal, L. Hemaspaandra, S. Maimon, and D. Rubery. Closure and nonclosure properties of the compressible and rankable sets. Technical Report arXiv:1611.01696 [cs.LO], Computing Research Repository, arXiv.org/corr/, November 2016. Revised, October 2018.

[AHMR19] J. Abascal, L. Hemaspaandra, S. Maimon, and D. Rubery. Closure and nonclosure properties of the compressible and rankable sets. In *Proceedings of the 13th International Conference on Language and Automata Theory and Applications*, pages 177–189. Springer-Verlag Lecture Notes in Computer Science #11417, March 2019.

[1] Note that due to recent work of Ji et al. [JNV$^+$20], it is known that there do exist nonrelativizing results involving computability concepts. Quoting Scott Aaronson [Aar20] regarding that recent work, "[The result of Ji et al. [JNV$^+$20] that MIP* = RE is] the first-ever fully convincing example of a nonrelativizing computability result."

[2] $\mathrm{F}_{\Delta_{i+1}}$ will denote the class of total functions computed by Turing machines given access to a Σ_i oracle. Equivalently, $\mathrm{F}_{\Delta_{i+1}} = (\mathrm{F}_{\mathrm{REC}})^{\Sigma_i}$. Note that $\mathrm{F}_{\mathrm{REC}} = \mathrm{F}_{\Delta_1}$. The class of partial functions computed by Turing machines given access to a Σ_i oracle will be denoted $(\mathrm{F}_{\mathrm{PR}})^{\Sigma_i}$ or simply as $\mathrm{F}_{\mathrm{PR}}^{\Sigma_i}$. For each $i \geq 1$, $A \equiv^i_{iso} B$ will denote that there is a bijection $f : \Sigma^* \to \Sigma^*$ such that $f(A) = B$ and $F \in \mathrm{F}_{\Delta_i}$. Note that $\equiv^1_{iso}$ is identical to $\equiv_{iso}$.

[ÁJ93] C. Álvarez and B. Jenner. A very hard log-space counting class. *Theoretical Computer Science*, 107:3–30, 1993.

[All85] E. Allender. Invertible functions, 1985. PhD thesis, Georgia Institute of Technology.

[AS12] E. Allender and H. Spakowski. Avoiding simplicity is complex. *Theory of Computing Systems*, 51(3):282–296, 2012.

[BBS86] J. Balcázar, R. Book, and U. Schöning. Sparse sets, lowness and highness. *SIAM Journal on Computing*, 15(3):739–746, 1986.

[BDG95] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* EATCS Texts in Theoretical Computer Science. Springer-Verlag, 2nd edition, 1995.

[BFH78] G. Brassard, S. Fortune, and J. Hopcroft. A note on cryptography and $\mathrm{NP} \cap \mathrm{coNP} - \mathrm{P}$. Technical Report TR-338, Department of Computer Science, Cornell University, Ithaca, NY, April 1978.

[BFL01] H. Buhrman, L. Fortnow, and S. Laplante. Resource-bounded Kolmogorov complexity revisited. *SIAM Journal on Computing*, 31(3):887–905, 2001.

[BGS91] A. Bertoni, M. Goldwurm, and N. Sabadini. The complexity of computing the number of strings of given length in context-free languages. *Theoretical Computer Science*, 86(2):325–342, 1991.

[CWCC19] Y.-H. Chang, R.-Y. Wu, R.-S. Chang, and J.-M. Chang. Improved algorithms for ranking and unranking (k,m)-ary trees. In *Proceedings of the 13th International Conference on Algorithmic Aspects in Information and Management*, pages 16–28. Springer-Verlag Lecture Notes in Computer Science #11640, August 2019.

[Dek62] J. Dekker. Infinite series of isols. In *Proceedings of the 5th Symposium in Pure Mathematics*, pages 77–96. American Mathematical Society, 1962.

[DFF19] M. Descotte, D. Figueira, and S. Figueira. Closure properties of synchronized relations. In *Proceedings of the 36st Annual Symposium on Theoretical Aspects of Computer Science*, pages 22:1–22:17. Leibniz International Proceedings in Informatics (LIPIcs) #126, March 2019.

[DM58] J. Dekker and J. Myhill. Retraceable sets. *Canadian Journal of Mathematics*, 10:357–373, 1958.

[DRS17] J. Day, D. Reidenbach, and M. Schmid. Closure properties of pattern languages. *Journal of Computer and System Sciences*, 84:11–31, 2017.

[FFNR03] S. Fenner, L. Fortnow, A. Naik, and J. Rogers. Inverting onto functions. *Information and Computation*, 186(1):90–103, 2003.

[GH96] J. Goldsmith and S. Homer. Scalability and the isomorphism problem. *Information Processing Letters*, 57(3):137–143, 1996.

[GHK92] J. Goldsmith, L. Hemachandra, and K. Kunen. Polynomial-time compression. *Computational Complexity*, 2(1):18–39, 1992.

[GOR00] J. Goldsmith, M. Ogihara, and J. Rothe. Tally NP sets and easy census functions. *Information and Computation*, 158(1):29–52, 2000.

[GS88] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.

[GS91] A. Goldberg and M. Sipser. Compression and ranking. *SIAM Journal on Computing*, 20(3):524–536, 1991.

[HJ95] L. Hemaspaandra and Z. Jiang. P-selectivity: Intersections and indices. *Theoretical Computer Science*, 145(1–2):371–380, 1995.

[HJRW98] L. Hemaspaandra, Z. Jiang, J. Rothe, and O. Watanabe. Boolean operations, joins, and the extended low hierarchy. *Theoretical Computer Science*, 205(1–2):317–327, 1998.

[HR90] L. Hemachandra and S. Rudich. On the complexity of ranking. *Journal of Computer and System Sciences*, 41(2):251–271, 1990.

[HR19] L. Hemaspaandra and D. Rubery. Recursion-theoretic ranking and compression. *Journal of Computer and System Sciences*, 101:31–41, 2019.

[HRW97] L. Hemaspaandra, J. Rothe, and G. Wechsung. Easy sets and hard certificate schemes. *Acta Informatica*, 34(11):859–879, 1997.

[HS19] P. Hartman and J. Sawada. Ranking and unranking fixed-density necklaces and Lyndon words. *Theoretical Computer Science*, 791:36–47, 2019.

[HT03] L. Hemaspaandra and L. Torenvliet. *Theory of Semi-Feasible Algorithms*. Springer-Verlag, 2003.

[HT05] T. Hoang and T. Thierauf. The complexity of the inertia and some closure properties of GapL. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, pages 28–37. IEEE Computer Society Press, June 2005.

[Huy90] D. Huynh. The complexity of ranking simple languages. *Mathematical Systems Theory*, 23(1):1–20, 1990.

[HZZ96] L. Hemaspaandra, M. Zaki, and M. Zimand. Polynomial-time semi-rankable sets. In *Journal of Computing and Information*, 2(1), Special Issue: *Proceedings of the 8th International Conference on Computing and Information*, pages 50–67, 1996. CD-ROM ISSN 1201-8511/V2/#1.

[Iba16] O. Ibarra. On decidability and closure properties of language classes with respect to bio-operations. *Natural Computing*, 15(2):225–234, 2016.

[Imm88] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

[JNV$^+$20] Z. Ji, A. Natarajan, T. Vidick, J. Wright, and H. Yuen. MIP* = RE. Technical Report arXiv:2001.04383 [quant-ph], Computing Research Repository, arXiv.org/corr/, January 2020.

[LMN16] K. Losemann, W. Martens, and M. Niewerth. Closure properties and descriptional complexity of deterministic regular expressions. *Theoretical Computer Science*, 627:54–70, 2016.

[MM01] C. Martínez and X. Molinero. A generic approach for the unranking of labeled combinatorial classes. *Random Structures and Algorithms*, 19(3–4):472–497, 2001.

[MR01] W. Myrvold and F. Ruskey. Ranking and unranking permutations in linear time. *Information Processing Letters*, 79(6):281–284, 2001.

[MR19] F. Mazowiecki and C. Riveros. Copyless cost-register automata: Structure, expressiveness, and closure properties. *Journal of Computer and System Sciences*, 100:1–29, 2019.

[MT14] K. Mikawa and K. Tanaka. Lexicographic ranking and unranking of derangements in cycle notation. *Discrete Applied Mathematics*, 166:164–169, 2014.

[OS18] A. Okhotin and K. Salomaa. Further closure properties of input-driven pushdown automata. In *Proceedings of the 20th International Conference on Descriptional Complexity of Formal Systems*, pages 224–236. Springer-Verlag Lecture Notes in Computer Science #10952, July 2018.

[Rog67] H. Rogers, Jr. *The Theory of Recursive Functions and Effective Computability*. McGraw-Hill, 1967.

[Rot99] J. Rothe. Complexity of certificates, heuristics, and counting types, with applications to cryptography and circuit theory. Habilitation thesis, Friedrich-Schiller-Universität Jena, Institut für Informatik, Jena, Germany, June 1999.

[Sch86] U. Schöning. *Complexity and Structure*. Springer-Verlag Lecture Notes in Computer Science #211, 1986.

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Mathematical Systems Theory*, 13(1):55–65, 1979.

[Soa87] R. Soare. *Recursively Enumerable Sets and Degrees: A Study of Computable Functions and Computably Generated Sets*. Perspectives in Mathematical Logic. Springer-Verlag, 1987.

[Sze88] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26(3):279–284, 1988.

[Tar15] P. Tarau. Ranking/unranking of lambda terms with compressed de Bruijn indices. In *Proceedings of the 8th International Conference on Intelligent Computer Mathematics*, pages 118–133. Springer-Verlag Lecture Notes in Computer Science #9150, July 2015.

[WCCL13] R.-Y. Wu, J.-M. Chang, A.-H. Chen, and C.-L. Liu. Ranking and unranking t-ary trees in a Gray-code order. *Computer Journal*, 56(11):1388–1395, 2013.