

Combining Self-Reducibility and Partial Information Algorithms

André Hernich¹ and Arfst Nickelsen²

¹ Humboldt-Universität zu Berlin, Germany

`hernich@informatik.hu-berlin.de`

² Technische Universität Berlin, Germany

`nicke@cs.tu-berlin.de`

Abstract. A partial information algorithm for a language A computes, for some fixed m , for input words x_1, \dots, x_m a set of bitstrings containing $\chi_A(x_1, \dots, x_m)$. E.g., p-selective, approximable, and easily countable languages are defined by the existence of polynomial-time partial information algorithms of specific type. Self-reducible languages, for different types of self-reductions, form subclasses of PSPACE.

For a self-reducible language A , the existence of a partial information algorithm sometimes helps to place A into some subclass of PSPACE. The most prominent known result in this respect is: P-selective languages which are self-reducible are in P [9].

Closely related is the fact that the existence of a partial information algorithm for A simplifies the type of reductions or self-reductions to A . The most prominent known result in this respect is: Turing reductions to easily countable languages simplify to truth-table reductions [8].

We prove new results of this type. We show:

1. Self-reducible languages which are easily 2-countable are in P. This partially confirms a conjecture of [8].
2. Self-reducible languages which are $(2m - 1, m)$ -verbose are truth-table self-reducible. This generalizes the result of [9] for p-selective languages, which are $(m + 1, m)$ -verbose.
3. Self-reducible languages, where the language and its complement are strongly 2-membership comparable, are in P. This generalizes the corresponding result for p-selective languages of [9].
4. Disjunctively truth-table self-reducible languages which are 2-membership comparable are in UP.

Topic: Structural complexity

1 Introduction

A *partial information algorithm* for a language A computes on input of m words x_1, \dots, x_m some information on membership of these words in A . It excludes some of the 2^m bitstrings a priori possible for $\chi_A(x_1, \dots, x_m)$, where χ_A is the characteristic function for A , by computing a set $D \subset \{0, 1\}^m$ with $\chi_A(x_1, \dots, x_m) \in D$. We call such sets D m -pools. Sets \mathcal{D} of m -pools that may occur as outputs of a specific partial information algorithm are called m -families.

In the following we only treat polynomially time bounded partial information algorithms. This line of research started in the seventies with the introduction of p-selective languages (due to [28], see [16]). Many other types of partial information have been studied since then, most prominently cheatability (due to [5]), and membership comparability (due to [2, 12], also known as approximability or non-superterseness, see [8, 27]); as well as verbosity, strong membership comparability, frequency computations, easily countable languages, multi-selectivity and sortability (for detailed definitions see e.g. [2, 3, 7, 8, 15, 17, 18]). A general theory for polynomial-time partial information classes was developed in [22, 23]. For a recent survey on partial information see [26].

From the start the interplay of partial information with reducibility and self-reducibility was investigated. Languages positively Turing reducible to a p-selective language are in fact many-one reducible to that language [10], and therefore are p-selective, as well. But for non-positive reductions the reduction closures of p-selective sets form a strict hierarchy [30, 14]. For recent results on reductions and polynomial-time partial information classes see [25, 6, 4].

A language A is self-reducible if membership for a word x in A can be determined by computing membership for smaller words in A . This property is quite typical for many computational problems. E.g., natural NP-complete problems like SAT are self-reducible. The question whether SAT has partial information algorithms has been studied extensively in the literature, see e.g., [1, 8, 27, 29].

Self-reducible languages are in PSPACE. In some cases, if a self-reducible language additionally has a certain type of polynomial-time partial information algorithm, one can show a better complexity bound than PSPACE. Self-reducible p-selective languages are in P [9]. Similar results hold for p-cheatable languages [13] and frequency computations [8]. We look for new and more general results of that type. A general solution for all types of partial information seems out of reach: An unconditional negative answer would imply $P \neq PSPACE$; on the other hand there are relativized worlds where the answer is negative for some types of partial information [8].

For two types of partial information we show that in combination with self-reducibility one gets membership in P. Moreover, we give an interesting example where the combination of partial information with a restricted type of self-reduction places languages not into P, but into the class UP.

In particular, we prove the following:

- Easily 2-countable self-reducible languages are in P. In [8] it was conjectured that this even holds for easily m -countable languages for every $m \geq 2$. We think that similar ideas that we use to prove the conjecture for $m = 2$ may turn out helpful to attack the general conjecture.
- If a self-reducible language and its complement are strongly 2-membership comparable, then the language is in P. This is a strict improvement on the result of [9] for self-reducible p-selective languages.
- If a language is 2-membership comparable and disjunctively truth-table self-reducible, then it is in UP. This is especially interesting since there is a relativized world where this cannot be improved to membership in P [8].

In proofs that place self-reducible languages with partial information algorithms into P one often starts as follows: One shows that for languages in the partial information class a certain reducibility simplifies to a more restricted type of reducibility. This is e.g. the case in our proof for easily 2-countable languages. In [9] the essential proof step is to simplify self-reductions to truth-table self-reductions for p-selective languages. We address the question to which types of partial information this simplification result can be extended. We show:

- Self-reducible languages where the m -fold characteristic function is $(2m - 1)$ -enumerable are truth-table self-reducible.

This generalizes the result for p-selective languages of [9]. Note that for p-selective languages the m -fold characteristic function is $(m + 1)$ -enumerable. We will apply this result in our proof for languages where the language and its complement are strongly 2-membership comparable.

2 Basics on Partial Information and Self-Reducibility

Languages, Bitstrings, Complexity Classes. Languages are subsets of $\Sigma^* = \{0, 1\}^*$. The *characteristic function* $\chi_A: \Sigma^* \rightarrow \{0, 1\}$ for a language A is defined by $\chi_A(x) = 1 \iff x \in A$. We extend χ_A to tuples by $\chi_A(x_1, \dots, x_m) := \chi_A(x_1) \cdots \chi_A(x_m)$. \overline{A} denotes the complement of A . $\#_1(b)$ denotes the number of 1's in a bitstring b , $b[i]$ is the i -th bit of b , and $b[i_1, \dots, i_k] := b[i_1] \cdots b[i_k]$. For background on Turing machines and complexity classes see e.g. [11]. FP denotes the class of polynomial-time computable functions. A nondeterministic Turing machine is called unambiguous if for every input x there is at most one accepting computation. A language A is in UP if there is an unambiguous polynomial time bounded nondeterministic Turing machine that accepts A .

Reducibility, Self-reducibility. We define several types of reductions and self-reductions:

1. A language A is polynomial-time Turing reducible to a language B if there is a deterministic polynomial-time oracle Turing machine M with $A = L(M, B)$.
2. If the machine is non-adaptive, we say that A is polynomial-time truth-table (or tt, for short) reducible to B . Equivalently, tt-reducibility can be defined in terms of generator and evaluator: A is polynomial-time tt-reducible to B if there exists a generator $g \in \text{FP}$ and an evaluator $\alpha \in \text{FP}$ such that on input x , $g(x)$ is a tuple $\langle q_1, \dots, q_s \rangle$ and $\chi_A(x) = \alpha(x, \chi_B(q_1, \dots, q_s))$. If we define $\alpha_x(b_1, \dots, b_s) := \alpha(x, b_1, \dots, b_s)$, α_x is the boolean function that evaluates the oracle answers on input x .
3. For $k \in \mathbb{N}$, A is k -tt-reducible if on every input x there are at most k words in $g(x)$. We say that A is *dt-reducible* if for every x the boolean function α_x is a disjunction. A is *nor-nand-reducible* to B if for every x the boolean function α_x is either a nor- or a nand-function. (Note that one can decide which of the two cases for α_x holds without knowing the oracle answers.)

4. A language A is self-reducible if A is polynomial-time reducible to A by some machine M such that all words queried by M on input x are shorter than x . Corresponding to each restricted type of reducibility we also have a restricted type of self-reducibility.

We use here self-reducibility in the narrower sense where queries have to be smaller than the input word *with respect to the word length*. All our results also hold for the more liberal definition of self-reducibility in [21]. The following facts are due to Ko [19].

Fact 1.

1. Every self-reducible language is in PSPACE.
2. Every dtt-self-reducible language is in NP.
3. Every 1-tt-self-reducible language is in P.

Partial information classes. We now introduce the concept of partial information classes. We state facts from [24] (see also [26]).

Definition 1 (Pool, Family, Partial Information Class). Let $m \geq 1$.

1. A subset $D \subseteq \{0, 1\}^m$ is called an m -pool.
2. A set $\mathcal{D} = \{D_1, \dots, D_r\}$ of m -pools is called an m -family if
 - (a) \mathcal{D} covers $\{0, 1\}^m$, that is $\bigcup_{i=1}^r D_i = \{0, 1\}^m$, and
 - (b) \mathcal{D} is closed under subsets, that is $D_1 \in \mathcal{D}$ and $D_2 \subseteq D_1$ implies $D_2 \in \mathcal{D}$.
3. For an m -family \mathcal{D} , a language A is in $P[\mathcal{D}]$ if and only if there is an $f \in \text{FP}$ such that $f(x_1, \dots, x_m) \in \mathcal{D}$ and $\chi_A(x_1, \dots, x_m) \in f(x_1, \dots, x_m)$ for all words x_1, \dots, x_m .

In general, different m -families may yield the same partial information class. However, to produce all partial information classes we need only consider families in so called *normal form*. Such families are closed under permuting positions in bitstrings of pools in the family, replacing bits of some bitstring position by constant 0 (or 1), and copying bits from one bitstring position to another. For every m -family there is a unique m -family in normal form that produces the same partial information class. Given m -pools D_1, \dots, D_s , $\langle D_1, \dots, D_s \rangle$ denotes the minimal m -family in normal form that contains these pools. Inclusion of partial information classes corresponds to inclusion of families in normal form.

We next give names and informal descriptions to several pools which we will use to describe types of partial information. We define several families as well, mostly by listing a set of pools generating them.

Definition 2 (Some Pools).

1. $\text{equ}_2 := \{00, 11\}$ *The words are equivalent wrt. membership in A .*
2. $\text{xor}_2 := \{01, 10\}$ *Exactly one of the words is in A .*
3. $\text{sel}_2 := \{00, 01, 11\}$ *If the first word is in A , then also the second.*
4. $\text{sel}_m := \{0^i 1^{m-i} \mid i = 0, \dots, m\}$ *If x_i is in A , then also x_{i+1} .*
5. $\text{bottom}_m := \{b \mid |b| = m, \#_1(b) \leq 1\}$ *At most one word is in A .*
6. $\text{top}_m := \{b \mid |b| = m, \#_1(b) \geq m - 1\}$ *At least $(m - 1)$ words are in A .*

Definition 3 (Some Families).

1. $k\text{-SIZE}_m := \{D \subseteq \{0, 1\}^m \mid |D| \leq k\}$
2. $\text{SEL}_2 := \langle \text{sel}_2 \rangle$
3. $m\text{-CARD}_m := \{D \subseteq \{0, 1\}^m \mid \exists i \in \{0, \dots, m\} \forall b \in D : \#_1(b) \neq i\}$
4. $\text{SMC}_m := \langle \{0, 1\}^m \setminus \{1^m\}, \{0, 1\}^m \setminus \{01^{m-1}\} \rangle$
5. $\text{CoSMC}_m := \langle \{0, 1\}^m \setminus \{0^m\}, \{0, 1\}^m \setminus \{10^{m-1}\} \rangle$

Languages in $\text{P}[m\text{-SIZE}_m]$ are called m -cheatable, as defined in [8]. Languages in $\text{P}[\text{SEL}_2]$ are called p-selective, as defined in [28]. Languages in $\text{P}[m\text{-CARD}_m]$ are called easily m -countable, as defined in [18]. Languages in $\text{P}[(2^m - 1)\text{-SIZE}_m]$ are called m -approximable or m -membership comparable, as defined in [5]. Languages in $\text{P}[k\text{-SIZE}_m]$ are called (k, m) -verbose. Languages in $\text{P}[\text{SMC}_m]$ are called strongly m -membership comparable, as defined in [20].

We close this section with Figure 1 that shows the inclusion structure of all 2-families in normal form, and hence the inclusion structure of all partial information classes produced by 2-families.

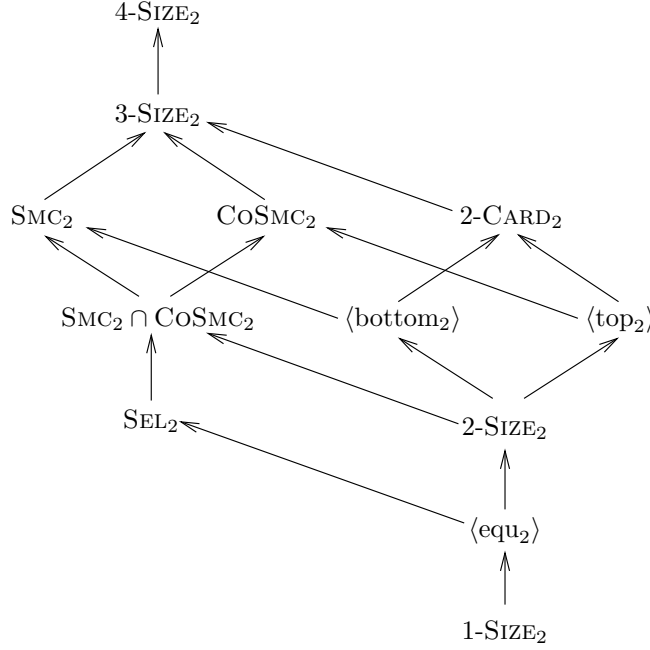


Fig. 1. Inclusion structure of all 2-families in normal form. Arrows stand for strict inclusion.

3 Self-Reducible Easily 2-Countable Languages are in P

Beigel, Kummer, and Stephan ask in [8] whether every self-reducible easily m -countable language is in P. In this section, we show that this holds at least for the case of easily 2-countable languages:

Theorem 1. *Every self-reducible language in $P[2\text{-CARD}_2]$ is in P.*

For the proof of Theorem 1 we use a result from [8] on simplification of reductions:

Fact 2. *Every language A reducible to an easily m -countable language B is tt-reducible to B .*

Because in the proof of Fact 2 the truth-table queries are a subset of the queries in the Turing reduction tree, that result also tells us that self-reducible languages in $P[2\text{-CARD}_2]$ are tt-self-reducible. Furthermore, we apply the following two lemmas. The first one breaks tt-self-reducibility down to nor-nand-self-reducibility, whereas the second one shows membership in P.

Lemma 1. *Every tt-self-reducible language in $P[2\text{-CARD}_2]$ is nor-nand-self-reducible.*

Proof. Let A be tt-self-reducible via M , and let $A \in P[2\text{-CARD}_2]$ via $f \in \text{FP}$. Without loss of generality, $f(x, y) \in \{\text{bottom}_2, \text{equ}_2, \text{top}_2\}$ for all words x, y . The following algorithm decides A in a nor-nand-self-reducing fashion.

On input x , let Q be the set of queries of $M(x)$. If there is a word $q \in Q$ with $f(x, q) = \text{equ}_2$, then $\chi_A(x) = \chi_A(q)$. Replace Q with the set of queries of $M(q)$. Iterate the above process until there is no word $q \in Q$ with $f(x, q) = \text{equ}_2$, or until M accepts or rejects (in which case we accept or reject x).

Assume we end up with a set Q with $f(x, q) \in \{\text{bottom}_2, \text{top}_2\}$ for all $q \in Q$. Then the following procedure computes sets IN, OUT and X such that $\text{IN} \cup \text{OUT} \cup X = Q$, $\text{IN} \subseteq A$, $\text{OUT} \subseteq \bar{A}$ and either for all $q \in X$, $f(x, q) = \text{bottom}_2$ or for all $q \in X$, $f(x, q) = \text{top}_2$. At the beginning, set $\text{IN} = \text{OUT} = \emptyset$ and $X = Q$. While there are two words $q_1, q_2 \in X$ such that $f(x, q_1) = \text{bottom}_2$ and $f(x, q_2) = \text{top}_2$, compute $D = f(q_1, q_2)$. If $D = \text{bottom}_2$, we have $q_1 \notin A$ (otherwise $x \notin A$ and $q_2 \notin A$ which contradicts $f(x, q_2) = \text{top}_2$) and we move q_1 from X into OUT. If $D = \text{top}_2$, we have $q_2 \in A$ (otherwise $x \in A$ and $q_1 \in A$ which contradicts $f(x, q_1) = \text{bottom}_2$) and we move q_2 from X into IN. If $D = \text{equ}_2$, then $x \in A$ if and only if $q_1 \notin A$. In this case, the algorithm stops and $\chi_A(x)$ is determined by querying q_1 .

Now we have to deal with the two cases

- $f(x, q) = \text{bottom}_2$ for all $q \in X$, or
- $f(x, q) = \text{top}_2$ for all $q \in X$.

We only treat Case 1. Case 2 is analogous, but with the nand-function instead of the nor-function. Suppose $f(x, q) = \text{bottom}_2$ for all $q \in X$. If at least one query

from X is in A , then $x \notin A$. Compute whether M accepts x in case all queries in X are not in A . If *no*, then M rejects x for all oracle answers. We can reject x without using oracle queries at all. If *yes*, then x is in A iff $\neg \bigvee_{q \in X} q \in A$. This means we can query the words in X and evaluate the answers with the nor-function. All computations can be done in polynomial time. \square

For membership in P , it suffices to show, by Fact 1, that every nor-nand-self-reducible language in $P[2\text{-CARD}_2]$ is 1-tt-self-reducible:

Lemma 2. *Every nor-nand-self-reducible language in $P[2\text{-CARD}_2]$ is 1-tt-self-reducible.*

Proof. Let A be nor-nand-self-reducible via generator $g \in \text{FP}$ and evaluator $\alpha \in \text{FP}$. Let $A \in P[2\text{-CARD}_2]$ via $f \in \text{FP}$. Without loss of generality, $f(x, y) \in \{\text{bottom}_2, \text{equ}_2, \text{top}_2\}$ for all words x, y . The following algorithm decides $x \in A$ in polynomial time with at most one query to A which is shorter than x .

On input x , let Q be the set of queries of $g(x)$, and let α_x be the boolean evaluation function for x . We consider the case that α_x is a nor-function, i.e. $x \in A$ if and only if for all $q \in Q$: $q \notin A$; the case that α_x is a nand-function is analogous. For each $q \in Q$, do the following: compute $g(q) = \langle q_1, \dots, q_k \rangle$ and determine, whether α_q is a nand- or a nor-function. Consider the following two cases:

Case 1. α_q is a nand-function, i.e. $q \in A$ if and only if $\exists i : q_i \notin A$. Then, for each $i \in \{1, \dots, k\}$ we can decide either $x \in A$ or $q_i \in A$ (clearly, if we know $\chi_A(q_i)$ for all i , we know $\chi_A(q)$, too). For all $i = 1, \dots, k$, compute $D = f(x, q_i)$ and consider the following cases. If $D = \text{bottom}_2$, then $x \notin A$, since otherwise $q_i \notin A \Rightarrow q \in A \Rightarrow x \notin A$ (because α_x is a nor- and α_q is a nand-function) which contradicts $x \in A$. If $D = \text{equ}_2$, then $x \in A \Leftrightarrow q_i \in A$ and we query q_i in order to compute $\chi_A(x)$. If $D = \text{top}_2$, then $q_i \in A$, since otherwise $q \in A \Rightarrow x \notin A \Rightarrow q_i \in A$ which contradicts $q_i \notin A$.

Case 2. α_q is a nor-function, i.e. $q \in A$ if and only if for all i : $q_i \notin A$. For all $i = 1, \dots, k$, compute $D = f(x, q_i)$ and consider the following cases until we can compute $\chi_A(x)$ or $\chi_A(q)$. If $D = \text{equ}_2$, then $x \in A \Leftrightarrow q_i \in A$ and we query q_i in order to compute $\chi_A(x)$. If $D = \text{top}_2$, then $q \notin A$ since otherwise $x \notin A \Rightarrow q_i \in A \Rightarrow q \notin A$ which contradicts $q \in A$. In the worst case we have $D = \text{bottom}_2$ for all i . But then $x \notin A$, since otherwise $q_i \notin A$ for all $i \in \{1, \dots, k\}$ (because $f(x, q_i) = \text{bottom}_2$ for all such i) and therefore $q \in A$ which implies $x \notin A$; a contradiction to $x \in A$.

We have seen that in all cases either we can decide membership in A for all words $q \in Q$, and hence for x , or we find a query q_i such that $x \in A \Leftrightarrow q_i \in A$. This means that A is 1-tt-self-reducible. Moreover, it is easy to see that the whole procedure runs in time polynomial in $|x|$. \square

We thus have shown that every self-reducible easily 2-countable language is contained in P .

4 Simplifying Turing to Truth-Table Self-Reductions for $(2m - 1, m)$ -Verbose Languages

In order to show that self-reducible languages in a certain partial information class are contained in P, it is convenient to show first that they are tt-self-reducible. We have already used such a result from [8] in the case of easily countable languages, see Fact 2. We now show a similar result for $P[(2m - 1)\text{-SIZE}_m]$ for all $m \in \mathbb{N}$. This is a generalization of Theorem 4 in [9] on p-selective languages.

Theorem 2. *For all $m \in \mathbb{N}$, if $A \in P[(2m - 1)\text{-SIZE}_m]$ is self-reducible, then A is tt-self-reducible.*

Proof. Let A be self-reducible via M and $A \in P[(2m - 1)\text{-SIZE}_m]$ via $f \in \text{FP}$. On input $\langle x_1, \dots, x_m \rangle$, f computes a set $D \subseteq \{0, 1\}^m$ with $|D| \leq 2m - 1$ and $\chi_A(x_1, \dots, x_m) \in D$.

The query tree $T_M(x)$ reflects the computations of M on input x for all possible answers to oracle queries. Each inner node of $T_M(x)$ is labeled with a query q and has two children, one for the oracle answer ‘yes’ and one for ‘no’. Leaf nodes are labeled with ‘ $\chi_A(x) = 1$ ’ if the path of queries and answers from the root to that leaf node describes an accepting computation of M on input x , otherwise they are labeled ‘ $\chi_A(x) = 0$ ’. We process $T_M(x)$ with a breadth-first extend-and-prune algorithm. Pruning means removing a subtree and replacing it by a leaf node labeled ‘ $\chi_A(x) = 1$ ’ or ‘ $\chi_A(x) = 0$ ’. Extending means taking in children of nodes labeled with queries. We thus always keep a (pruned) subtree of $T_M(x)$ satisfying the following invariants:

1. On every path in T there are at most $m - 2$ nodes having two inner nodes as children.
2. Leaf nodes are labeled ‘ $\chi_A(x) = 1$ ’ if and only if M accepts x for all oracles consistent with the query-answer path leading to that node.

Initially, let T be the full query tree up to depth $m - 1$. Suppose T has been extended such that there is a path on which $m - 1$ nodes, labeled with q_1, \dots, q_{m-1} , have two inner nodes as children. Let $a = a_1 \dots a_{m-2}$ be the bitstring encoding the answers to queries along the path: $a_i = 1$ if and only if the answer to query q_i is ‘yes’. Compute $D := f(x, q_1, \dots, q_{m-1})$. Split D into two pools by defining $D_c := \{b \in D \mid b[1] = c\}$ for $c \in \{0, 1\}$. We have $|D_0| + |D_1| = |D| \leq 2m - 1$. Hence, for at least one c we have $|D_c| \leq m - 1$. Fix such a value for c . Now there must be an $i \in \{1, \dots, m - 1\}$ such that $ca[1, \dots, i - 1]0$ is not a prefix of a string in D_c or $ca[1, \dots, i - 1]1$ is not a prefix of a string in D_c . Say, $ca[1, \dots, i - 1]0$ is not a prefix of a string in D_c . This means that if $\chi_A(x) = c$, then ‘no’ is not the answer to query q_i . This means: If ‘no’ is the answer to query q_i , then $\chi_A(x) = 1 - c$. Prune the tree at the ‘no’-edge leaving the node labeled q_i . Replace the subtree at this edge by a leaf node labeled ‘ $\chi_A(x) = 1 - c$ ’.

When $T_m(x)$ has been processed up to some depth, only a constant number of nodes of this depth with two inner nodes as children remain in T . Therefore

the whole processing can be completed in polynomial time and in the end only polynomially many queries are left in T . These are the queries that now are given in parallel to the oracle. If the path in T determined by the oracle answers leads to a node labeled ' $\chi_A(x) = 1$ ', then accept x , else reject. We conclude that A is tt-self-reducible. \square

5 On Strongly 2-Membership Comparable Languages

In this section, we consider 2-families that strictly include SEL_2 , in particular the families $\text{SMC}_2 \cap \text{CoSMC}_2$ and 3-SIZE_2 . It turns out that self-reducible languages in the partial information class produced by the first family are in P . It is unlikely that this result can be extended to larger 2-families. However, we can at least show that dtt-self-reducible languages in $\text{P}[\text{3-SIZE}_2]$ are in UP .

5.1 Self-Reducibility and P

By Theorem 2, every self-reducible language in $\text{P}[\text{SMC}_2 \cap \text{CoSMC}_2]$ is tt-self-reducible. The proof that every tt-self-reducible language in $\text{P}[\text{SMC}_2 \cap \text{CoSMC}_2]$ is 1-tt-self-reducible, and hence in P , is an easy generalization of Theorem 3 in [9], together with the following fact (Lemma 3.24 from [24]):

Fact 3. *For every language $A \in \text{P}[\text{SMC}_2 \cap \text{CoSMC}_2]$ we can compute on input $\langle x_1, \dots, x_m \rangle$ in polynomial time a partition of $X = \{x_1, \dots, x_m\}$ into disjoint sets of one of the following two types:*

1. $X = \text{IN} \cup \text{OUT} \cup S_1 \cup \dots \cup S_r$ with $r \leq m$ such that:
 - $\text{IN} \subseteq A$ and $\text{OUT} \subseteq \overline{A}$,
 - For $1 \leq i \leq r$, $S_i \neq \emptyset$ and $S_i \subseteq A$ or $S_i \subseteq \overline{A}$.
 - For $1 \leq i < j \leq r$, $x \in S_i$ and $y \in S_j$: $\chi_A(x) \leq \chi_A(y)$.
2. $X = \text{IN} \cup \text{OUT} \cup X_1 \cup X_2$ such that:
 - $\text{IN} \subseteq A$ and $\text{OUT} \subseteq \overline{A}$,
 - For $i \in \{1, 2\}$: $X_i \neq \emptyset$ and $X_i \subseteq A$ or $X_i \subseteq \overline{A}$.
 - For $x \in X_1$ and $y \in X_2$: $\chi_A(x) \neq \chi_A(y)$.

Given a tt-self-reducible language in $\text{P}[\text{SMC}_2 \cap \text{CoSMC}_2]$, we can compute a partition of the words generated in the tt-self-reduction according to Fact 3. The case that the partition is of the first type is handled in the proof of Theorem 3 in [9]. If it is of the second type and the input word x is contained in X_i , then membership of x can be determined by querying some word in X_j , $j \neq i$. We have thus established the following theorem.

Theorem 3. *Every self-reducible language in $\text{P}[\text{SMC}_2 \cap \text{CoSMC}_2]$ is in P . \square*

5.2 Disjunctive Truth-Table Self-Reducibility and UP

We do not know whether every self-reducible language in $P[\text{SMC}_2]$ is in P . Indeed, it seems that results in this respect are hard to obtain: In the proof of Theorem 7.1 in [8], an oracle is constructed such that relative to it there exists a dtt-self-reducible language in $P[\text{SMC}_2]$ that is not in P .

To get some results for families above $\text{SMC}_2 \cap \text{COSMC}_2$ we have to be satisfied to show inclusion in a class larger than P . The following result is of that kind:

Theorem 4. *Every dtt-self-reducible language in $P[3\text{-SIZE}_2]$ is in UP.*

Proof. Let A be dtt-self-reducible via M , and let $A \in P[3\text{-SIZE}_2]$ via $f \in \text{FP}$. We construct a polynomial-time unambiguous nondeterministic Turing machine N which accepts A .

The dtt-self-reducing tree of M on input x is created by iteratively computing for each query q the queries that M would ask on input q . This tree has polynomial height, but in general contains exponentially many nodes. On an input x we walk a path in the dtt self-reducing tree of M on input x . We use f to decide whether the actual query is in A (the path then ends at this point), or to choose the next node to visit. We accept x if and only if the last node on the path is in A . Note that $x \in A$ if and only if there exists such a path. Assume we already visited x_1, \dots, x_l . If $M(x_l)$ queries q_1, \dots, q_s , we compute $f(q_i, q_j)$ for all $1 \leq i, j \leq s$, $i \neq j$. If for such a pair $f(q_i, q_j) = \text{sel}_2$, we can remove q_i from the list of queries. Let q_{i_1}, \dots, q_{i_r} be the queries remaining after these removals. If $r = 1$, we set $x_{l+1} := q_{i_1}$. If $r > 1$, and $f(q_{i_j}, q_{i_k}) = \text{top}_2$ for some pair, then at least one query is in A , and hence x_l is in A . It remains the case that $f(q_{i_j}, q_{i_k}) = \text{bottom}_2$ for all $1 \leq j, k \leq r$, $j \neq k$. This means: At most one of these queries is in A . We choose a j nondeterministically and set $x_{l+1} := q_{i_j}$. Clearly, N has on input x at most one accepting computation. \square

6 Conclusion

We have improved on previous results on combining partial information and self-reducibility. The improvement from SEL_2 to $\text{SMC}_2 \cap \text{COSMC}_2$, Theorem 3, may be not so surprising since these two families share many properties and are even linked by a reducibility (which needs access to an NP-oracle, see [4]). But the result on easily 2-countable languages, Theorem 1, is, in our view, an important step forward. It partially solves the conjecture of Beigel, Kummer, and Stephan [8]. Maybe, in the future similar proof ideas may help to answer their question “Are all easily countable self-reducible languages in P ?” positively.

An important tool in proofs, but also interesting in their own right, are theorems on the simplification of reductions or self-reductions like Theorem 2. Can it be extended? Our proof is by an iterative query tree pruning procedure which leaves only constantly many branches in the tree. We think that our proof exploits this type of local pruning technique optimally. On the other hand, the simplification result from [8] for easily countable languages is also proved by a

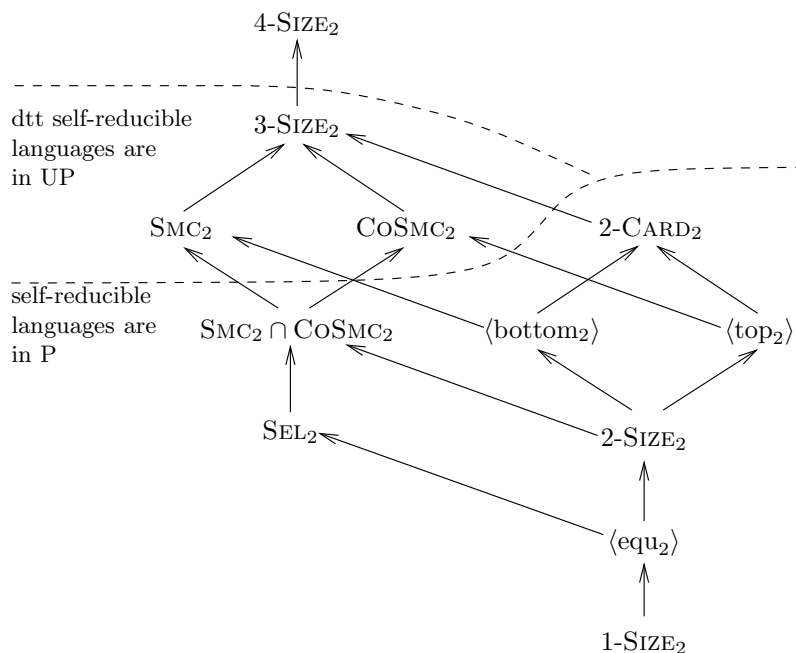


Fig. 2. This figure shows all 2-families in normal form as in Figure 1. Dashed lines separate families \mathcal{D} for which (dtt-)self-reducible languages in $P[\mathcal{D}]$ are in P (UP) from those ones where this is not known.

pruning technique involving trees of bounded rank. In their case the number of branches is not bounded by a constant. But their technique does not seem to apply to $(2m - 1, m)$ -verbose languages. Is there a proof technique powerful enough to subsume both results?

The last result on membership in UP, Theorem 4, suggests that one should not only look for conditions on the type of partial information and type of self-reducibility that yield membership in P . It would be nice if someone came up with characterizations of several classes between P and PSPACE by such combinations. Even for UP such a characterization still has to be found.

References

1. M. Agrawal and V. Arvind. Polynomial time truth-table reductions to P-selective sets. In *Proc. 9th Structure in Complexity Theory*, 1994.
2. A. Amir, R. Beigel, and W. Gasarch. Some connections between bounded query classes and non-uniform complexity. In *Proc. 5th Struct. in Complexity Th.*, 1990.
3. A. Amir and W. Gasarch. Polynomial terse sets. *Inf. and Computation*, 77, 1988.
4. S. Bab and A. Nickelsen. One query reducibilities between partial information classes. In *Proc. MFCS*, volume 31543 of *LNCS*, pages 404–415. Springer, 2004.

5. R. Beigel. *Query-Limited Reducibilities*. PhD thesis, Stanford University, 1987.
6. R. Beigel, L. Fortnow, and A. Pavan. Membership comparable and p-selective sets. Technical Report 2002-006N, NEC Research Institute, 2002.
7. R. Beigel, M. Kummer, and F. Stephan. Quantifying the amount of verboseness. In *Proc. Logical Found. of Comput. Sci.*, volume 620 of *LNCS*. Springer, 1992.
8. R. Beigel, M. Kummer, and F. Stephan. Approximable sets. *Inf. and Computation*, 120(2), 1995.
9. Buhrman, van Helden, and Torenvliet. P-selective self-reducible sets: A new characterization of P. In *Conference on Computational Complexity*, volume 8, 1993.
10. H. Buhrman, L. Torenvliet, and P. van Emde Boas. Twenty questions to a P-selector. *Information Processing Letters*, 4(4):201–204, 1993.
11. D.-Z. Du and K.I.Ko. *Theory of Computational Complexity*. Wiley & Sons, 2000.
12. W. Gasarch. Bounded queries in recursion theory: A survey. In *Proc. 6th Structure in Complexity Theory*, 1991.
13. J. Goldsmith, D. Joseph, and P. Young. Self-reducible, P-selective, near-testable, and P-cheatable sets: The effect of internal structure on the complexity of a set. In *2nd Structure in Complexity Theory*. IEEE, 1987.
14. L. Hemaspaandra, A. Hoene, and M. Ogihara. Reducibility classes of p-selective sets. *Theoretical Computer Science*, 155:447–457, 1996.
15. L. Hemaspaandra, Z. Jiang, J. Rothe, and O. Watanabe. Polynomial-time multi-selectivity. *J. of Universal Comput. Sci.*, 3(3), 1997.
16. L. Hemaspaandra and T. Torenvliet. *Theory of semi-feasible Alg.* Springer, 2002.
17. M. Hinrichs and G. Wechsung. Time bounded frequency computations. In *Proc. 12th Conf. on Computational Complexity*, 1997.
18. A. Hoene and A. Nickelsen. Counting, selecting, and sorting by query-bounded machines. In *Proc. STACS 93*, volume 665 of *LNCS*. Springer, 1993.
19. K. Ko. On self-reducibility and weak p-selectivity. *Journal of Computer and System Sciences*, 26:209–221, 1983.
20. J. Köbler. On the structure of low sets. In *Proc. 10th Structure in Complexity Theory*, pages 246–261. IEEE Computer Society Press, 1995.
21. A. R. Meyer and M. S. Paterson. With what frequency are apparently intractable problems difficult? Technical Report MIT/LCS/TM-126, MIT, Cambridge, 1979.
22. A. Nickelsen. On polynomially \mathcal{D} -verbose sets. In *Proc. STACS 97*, volume 1200 of *LNCS*, pages 307–318. Springer, 1997.
23. A. Nickelsen. Partial information and special case algorithms. In *Proc. MFCS 01*, pages 573–584. Springer LNCS 2136, 2001.
24. A. Nickelsen. *Polynomial Time Partial Information Classes*. W&T Verlag, 2001. Dissertation, TU Berlin, 1999, also available at www.tal.cs.tu-berlin.de/nickelsen/.
25. A. Nickelsen and T. Tantau. Closure of polynomial time partial information classes under polynomial time reductions. In *Proc. FCT 01*, volume 2138 of *LNCS*, pages 299–310. Springer, 2001.
26. A. Nickelsen and T. Tantau. Partial information classes. Complexity Theory Column, *SIGACT News*, 34, 2003.
27. M. Ogihara. Polynomial-time membership comparable sets. *SIAM Journal on Computing*, 24(5):1068–1081, 1995.
28. A. Selman. P-selective sets, tally languages and the behaviour of polynomial time reducibilities on NP. *Math. Systems Theory*, 13:55–65, 1979.
29. D. Sivakumar. On membership comparable sets. *Journal of Computer and System Sciences*, 59(2):270–280, 1999.
30. Seinosuke Toda. On polynomial-time truth-table reducibility of intractable sets to p-selective sets. *Mathematical Systems Theory*, 24:69–82, 1991.