Infinitely-Often Autoreducible Sets

Richard Beigel[*], Temple University
Lance Fortnow[†], University of Chicago
Frank Stephan[‡], National University of Singapore

Abstract

A set A is autoreducible if one can compute, for all x, the value $A(x)$ by querying A only at places $y \neq x$. Furthermore, A is infinitely-often autoreducible if, for infinitely many x, the value $A(x)$ can be computed by querying A only at places $y \neq x$. For all other x, the computation outputs a special symbol to signal that the reduction is undefined. It is shown that for polynomial time Turing and truth-table autoreducibility there are sets A, B, C in EXP such that A is not infinitely-often Turing autoreducible, B is Turing autoreducible but not infinitely-often truth-table autoreducible, C is truth-table autoreducible with $g(n)+1$ queries but not infinitely-often Turing autoreducible with $g(n)$ queries. Here n is the length of the input, g is nondecreasing and there exists a polynomial p such that $p(n)$ bounds both, the computation time and the value, of g at input of length n. Furthermore, connections between notions of infinitely-often autoreducibility and notions of approximability are investigated. The Hausdorff-dimension of the class of sets which are not infinitely-often autoreducible is shown to be 1.

1 Introduction

Consider a set where every element is chosen independently at random. Intuitively the membership of x should not depend on the membership of the other elements. Indeed, random

[*]Department of Computer and Information Sciences, Temple University, Wachman Hall (038-24), 1805 North Broad Street, Philadelphia PA 19122-6094, USA, Email: beigel@cis.temple.edu.

[†]Department of Computer Science, University of Chicago, 1100 E 58th Street, Chicago, IL 60637, USA, Email: fortnow@cs.uchicago.edu. Research was done while the author was at the NEC Research Institute.

[‡]School of Computing and Department of Mathematics, National University of Singapore, 3 Science Drive 2, Singapore 117543, Republic of Singapore, Email: fstephan@comp.nus.edu.sg. Research was supported by the Deutsche Forschungsgemeinschaft (DFG), Heisenberg grant Ste 967/1-1 while F. Stephan was working at the Universität Heidelberg until August 2003. From August 2003 until June 2004, F. Stephan was working at the National ICT Australia LTD which is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence Program. Since July 2004, F. Stephan is working at the National University of Singapore and partially supported by NUS grant number R252–000–212–112.

sets are not *autoreducible*, that is, it is impossible to compute for every x the membership of x from the membership of the $y \neq x$. Ebert [12, 13] gives the surprising result that one can nevertheless compute correctly the membership of an infinite number of elements of a random set from the membership of the other ones. The present work extends the study of this notion, called *infinitely-often autoreducible*.

Trakhtenbrot [24] introduced in the recursion theoretic context the notion of autoreducibility. There are many natural examples of autoreducible sets, for example any index set B of partial recursive functions is autoreducible: by the Padding Lemma there is a recursive strictly increasing function p such that, for all x, $\varphi_x = \varphi_{p(x)}$. It follows that one can for given x compute the value $p(x)$ which is different from x and query whether $p(x) \in B$. As $\varphi_x = \varphi_{p(x)}$ one has that either $x, p(x)$ are both in B or both outside B, so one knows whether $x \in B$. Other natural examples of autoreducible sets are retraceable sets, cylinders, creative sets like the halting problem, semirecursive sets and recursive sets; see the book of Odifreddi [20] for the definitions of these types of sets. On one hand, autoreducible sets are quite common and there are even nonrecursive Turing degrees containing only autoreducible sets [15], on the other hand Trakhtenbrot [24] constructed recursively enumerable sets which are not autoreducible.

The notion of autoreducibility can easily be carried over to resource bounded reducibilities r like polynomial time Turing reducibility.

Definition 1.1. A set is r-autoreducible iff there is an r-reduction that computes for every x the value $A(x)$ from the oracle A without querying A at x.

For example, a many-one EXP-complete set A satisfies that A is many-one reducible to $\overline{A}$ via some function f; that is, $A(x) = \overline{A}(f(x)) = 1 - A(f(x))$. This guarantees that $f(x) \neq x$ for all x. So one has that A is autoreducible by a polynomial time Turing reduction which asks exactly one query: what is A at $f(x)$ and knowing this, let $A(x) = 1 - A(f(x))$.

In the present work, polynomial time truth-table and Turing reducibility are considered where the number of questions might also be bounded. Truth-table reducibility is different from Turing reducibility in the sense that the place of the n-th query does not depend on the oracle answers to the queries asked before and one can compute an explicit polynomially sized list of places queried. So the oracle is queried at many places in parallel and afterwards its answers are used by the program of the truth-table reduction without any further interaction with the oracle.

Also in complexity theory there are many natural examples of autoreducible sets. The set SAT is truth-table autoreducible with two queries: If ϕ is any formula with the variable u build in, then the derived formulas $\phi[u \to 0]$ and $\phi[u \to 1]$ where u has been replaced by the logical constants 0 and 1, respectively, are different from ϕ and one can compute with two queries to SAT whether these formulas are satisfiable. Then ϕ is satisfiable iff at least one of the formulas $\phi[u \to 0]$ and $\phi[u \to 1]$ is. By the way, Schnorr [23] studied this special case where the autoreduction goes to instances shorter than the original input and called a set self-reducible if it has such an autoreduction. Buhrman, Fortnow, Melkebeek and Torenvliet [11] showed that the Turing complete sets for EXP are Turing autoreducible while some of the Turing complete sets for the class EEXPSPACE of all sets which are, for some polynomial p, computable in space $2^{2^{p(n)}}$, fail to have this property. It is unknown whether all

sets Turing complete for EEXP are autoreducible; settling this open question would separate some complexity classes which are not yet known to be different.

Random sets are not autoreducible, but Ebert [12, 13] showed that they surprisingly satisfy the following variant which is called infinitely-often autoreducible which is defined as follows.

Definition 1.2. A set A is infinitely-often r-autoreducible iff there is an r-reduction M which at input x queries A only at places $y \neq x$. For infinitely many x, M computes $A(x)$ correctly, for all other x, M is undefined and signals this by outputting a special symbol.

The result that random sets are infinitely-often autoreducible received a lot of attention. Not only because one would not expect that it is possible to make predictions about the membership of x in a random set by looking which other y are in, but also because Ebert [12] introduced for his proof a mathematical puzzle which was easy to understand, became famous and has some interest on its own right: the hat problem. It was the topic of several newspaper articles, for example in the German weekly newspaper Die Zeit [10] and in the New York Times [22].

It is already well-known that there are sets which are not infinitely-often autoreducible [7], but these examples are outside EXP, the class of all exponential time computable sets. EXP is the first deterministic time class known to contain nondeterministic polynomial time NP and polynomial space PSPACE. Therefore, it is natural to study the structure of the sets inside EXP and the main result of the present work, given in Section 2, says that there is a set A in EXP which is not infinitely-often Turing autoreducible. In Sections 3, the major autoreducibility notions are separated by showing that there are sets in EXP which are autoreducible for the first reduction but not infinitely-often autoreducible by the second reduction; this is done in particular for Turing versus truth-table and for truth-table with $g(n)+1$ queries versus Turing with $g(n)$ queries. This second result implies the separation of truth-table versus bounded truth-table. In Section 4, the relations between notions of approximability and infinitely-often autoreducibility are investigated. Finally, in Section 5, it is shown that there are quite many sets in EXP which are not autoreducible: the class of these sets has Hausdorff-dimension 1 in exponential time.

Notation 1.3. The notation follows standard textbooks like the one of Odifreddi [20] with some exceptions: The function $x \mapsto \log(x)$ denotes the logarithm of basis 2 with the exception that $\log(q) = 0$ for $q < 1$ in order to avoid to deal with too many exceptions in logarithmic expressions. The term $\log^*(x)$ denotes the number of iterations of log necessary to reach a number strictly below 1. So, $\log^*(0) = 0$, $\log^*(1) = 1$ and $\log^*(2^x) = \log^*(x) + 1$. A set D is called *supersparse* iff, for all x, $D \cap \{x, x+1, \ldots, 2^x\}$ contains at most $\log^*(x)$ many elements. $A \Delta D$ denotes the symmetric difference of A and D; the set $A \Delta D$ is called a *supersparse variant of* A if D is a supersparse set. Furthermore, the notation $O(f)$ is generalized to $Poly(f)$ which is the set of all g such that there is a polynomial p with $(\forall n)\,[g(n) \leq p(f(n))]$.

2 Some Set in EXP is not infinitely-often autoreducible

Note that a computation is exponential in the length (= logarithm) of x iff it is quasipolynomial in x itself. Therefore one considers in the case of functionals quasipolynomial time bounds. More precisely, a partial functional f which assigns to inputs of the form $A(0)A(1)\dots A(x)$ values $a_{x+1}a_{x+2}\dots a_y$ is called a quasipolynomial time extension functional iff there is a constant c permitting to compute the extensions in time $x^{\log^c(x)}$. Following Lutz [17], one can introduce the following notion of resource-bounded genericity; see the survey of Ambos-Spies and Mayordomo [3] for further details.

Definition 2.1. A set A is general generic iff, for every quasipolynomial time computable functional f,

- either $f(A(0)A(1)\dots A(x))$ is undefined for almost all x
- or there are x, y such that $x < y$, $f(A(0)A(1)\dots A(x)) = a_{x+1}a_{x+2}\dots a_y$ and $A(z) = a_z$ for $z = x+1, x+2, \dots, y$.

So, either "A almost always avoids f" or "A meets f".

General generic sets have to be distinguished from the weaker variant of generic sets as introduced by Ambos-Spies, Fleischhack and Huwig [2] which either meet or avoid every quasipolynomial time extension functionals which predicts only one bit whenever defined. Balcázar and Mayordomo [7] observed that these sets are not infinitely-often autoreducible.

Fact 2.2 [7]**.** *There are sets which are not infinitely-often autoreducible. In particular general generic sets have this property.*

On the other hand, Ambos-Spies, Neis and Terwijn [5] showed that the notions of generic sets and resource bounded randomness are compatible. As random sets are infinitely-often autoreducible (even infinitely-often truth-table autoreducible) [12, 13], there are some generic sets which are infinitely-often autoreducible and Fact 2.2 really needs the stronger version of general generic sets.

General generic sets cannot be in EXP due to the quasipolynomial time bound. The following result shows that sets which are not infinitely-often autoreducible can be found in EXP. Note that many-one EXP-complete sets are (everywhere) autoreducible since they are many-one equivalent to their complement. Buhrman, Fortnow, van Melkebeek and Torenvliet [11] show that every Turing EXP-complete set is autoreducible.

Theorem 2.3. *There is a set in EXP which is not infinitely-often autoreducible with respect to Turing reducibility.*

Proof. Let $M_0, M_1, \dots$ be an enumeration of all polynomial time autoreductions such that each M_e needs at most time $x + e$ at input x and queries the set A only at places $y \leq 2^x$ with $y \neq x$. Note that x is superpolynomial in $\log(x)$ and therefore, all polynomial time

autoreductions are covered. The set A is constructed by a priority construction and satisfies at the end for every e one of the following two possibilities.

- There is a number x such that $M_e^A(x)$ outputs $b \in \{0,1\}$ with $b \neq A(x)$.
- For almost every x, $M_e^A(x)$ is undefined.

The e-th requirement is then the first of these two conditions. The construction will be such that it either satisfies the e-th requirement explicitly or enforces the second condition implicitly.

The construction uses approximations A_x of A where $A_x(y) = A(y)$ for all x, y with $y < x$. So one has to simulate the construction only $x+1$ stages to know the value of A at x. Together with a proof that every stage needs time exponential in the length of x it follows that $A \in \mathrm{EXP}$.

Construction of A. Let A_x and $r_{e,x}$ be the values of A and r_e before stage x, in particular, $A_0 = \emptyset$ and $r_{e,0} = 0$ for all x.

In stage x one searches the least e such that there is a finite set D satisfying the following requirements where D is the set of the positions for which it is intended that A_{x+1} and A_x will differ.

Bound on e: $e \leq \log^*(x)$.

Respecting Restraints: $r_{e',x} < x$ for all $e' \leq e$.

Requirement e not yet done: There is no number $x' < x$ such that $M_e^{A_x}(x')$ queries A_x only at places below x and computes a wrong prediction for $A(x')$.

Requirement e needs attention: $M_e^{A_x \Delta D}(x)$ computes a prediction different from $(A_x \Delta D)(x)$; where $A_x \Delta D$ denotes the symmetric difference of A_x and D.

Size-constraint on D met: $D \subseteq \{x, x+1, x+2, \ldots, 2^x\}$ and D has at most $2^{-e-2} \cdot \log^*(x) - 2e$ many elements.

If the above procedure finds an e (which is the minimal one possible) and D is the corresponding set mentioned above then

$$\begin{aligned} A_{x+1} &= A_x \Delta D; \\ r_{e,x+1} &= 2^x + 2; \\ r_{e',x+1} &= r_{e',x} \text{ for all } e' \neq e; \end{aligned}$$

else nothing changes, that is, $A_{x+1} = A_x$ and $r_{e',x+1} = r_{e',x}$ for all e'.

Verification. One first notes that in stage x the search considers only $\log^*(x)$ many values of e and for each of these, the search runs over computations which make at most $x+e$ queries where at most $2^{-e-2} \cdot \log^*(x) - 2e$ of the answers can differ from the current values

of A_x as these queries hit elements in D. So, for each e, there are $O(x^{\log^*(x)})$ many possible computation paths. As $\log^*(x) \leq \log(x+2)$, the running time of step x of the algorithm is quasipolynomial in x, that is, exponential in $\log(x)$. As $A(x) = A_{x+1}(x)$, it follows that one can compute $A(x)$ by running the algorithm for the stages $0, 1, \ldots, x$ and so the overall running time is quasipolynomial in x, that is, $A \in \mathrm{EXP}$.

Note that for sufficiently large x the bound $2^{-e-2} \cdot \log^*(x) - 2e$ becomes positive as e is a constant and $\log^*(x)$ is unbounded. Therefore, cardinality requirements do not hinder to satisfy Requirement e for sufficiently large x.

By usual priority arguments, one can show that every e is found only finitely often by the search algorithm and that $r_{e,x}$ converges from below to a final value $r_{e,\infty}$. More precisely, every $r_{e,x}$ changes only if the parameter e is selected in the search at stage x. One can show by induction that this happens only finitely often for all e. Assume that all $r_{e',\infty}$ with $e' < e$ exist and have maximum $\tilde{r}_e$. If e is not selected in the search at any $x > \tilde{r}_e$ then $r_{e,x}$ changes only finitely often and converges to some value $r_{e,\infty}$. If e is selected at some $x > \tilde{r}_e$ then the following happens. A_{x+1} is updated such that the autoreduction $M_e^{A_{x+1}}(x)$ returns a wrong value and queries A_{x+1} only at places smaller than $2^x + 1$. Furthermore, as at no $x' > x$ the search selects an $e' < e$, the restraint $r_{e,x+1}$ will be respected and the diagonalization of the autoreduction $M_e^{A_{x+1}}$ will not be undone by changing A at queried places, that is, $A_x(y) = A(y)$ for all values y queried by $M_e^{A_{x+1}}(x)$. Thus Requirement e will be counted as "done" at all stages $x' > x$ and so e will not be selected again by the search condition at those stages. Thus $r_{e,\infty} = 2^x + 2$.

Now it is shown that A is not infinitely-often autoreducible. Consider any autoreduction M_e which does not make any false prediction for A but might be undefined for some inputs. Let x be so large that $2^{-e-2} \cdot \log^*(x) - 2e > 2$ and $x > r_{e',\infty}$ for all $e' \leq e$. So the search does not return any $e' \leq e$ as otherwise the corresponding restraint would be increased again. It follows that $M_e^B(x)$ does not predict any value for input x on any set B of the form $A_x \Delta D$ with $|D| \leq 2^{-e-2} \cdot \log^*(x) - 2e - 1$ and $D \subseteq \{x, x+1, \ldots, 2^x\}$ – note that the search in the algorithm actually covers sets D with up to $2^{-e-2} \cdot \log^*(x) - 2e$ many elements, but this one additional element might be needed to make the prediction to be different from $(A_x \Delta D)(x)$. On the other hand, it might happen that up to $\log^*(x)$ many requirements $e' > e$ act between stages x and 2^x. Due to the update rule of the restraints, each of them acts only once between these stages. Furthermore, each e' changes A at up to $2^{-e'-2} \cdot \log^*(2^x) - 2e' \leq 2^{-e'-2} \cdot \log^*(x) - 2e - 1$ many positions between x and 2^x. The symmetric difference of A and A_x contains below x no element and between x and 2^x at most $2^{-e-2} \cdot \log^*(x) - 2e - 1$ many elements. So, $M_e^A(x)$ does also not predict any value for $A(x)$ and M_e^A is undefined for almost all inputs. ∎

3 On Truth-Table Autoreducibility

The topic of this section is the interrelation of autoreducibility notions with respect to truth-table reducibility, Turing-reducibility and the variants of these reducibilities where the number of queries is bounded. It is shown that only the trivial implications hold and that all differences between the notions can be witnessed by sets in EXP. Theorem 3.2 in particular shows that

there is a set in EXP which is tt$(n+1)$-autoreducible but not infinitely-often Turing(n)-autoreducible, where n is the logarithm (= length) of the input, tt$(n+1)$ means that the truth-table reduction is permitted to make up to $n+1$ queries and Turing(n) means that the Turing reduction is permitted to make up to n queries. This implies that this set is tt-autoreducible but not btt-autoreducible.

The proof of the following theorem uses Kolmogorov complexity arguments to construct a starting set A_0 which is Turing but not tt-complete for EXP. Watanabe [25] first constructed such a set; his ideas are also based on Kolmogorov complexity arguments.

Theorem 3.1. *There is a set A in EXP which is Turing autoreducible but not infinitely-often tt-autoreducible.*

Proof. The construction is a modification of the one from Theorem 2.3 with the following two differences:

- The set A_0 is not empty but a set which is Turing complete (but not tt-complete) for EXP.
- Furthermore, $M_0, M_1, \ldots$ is a list of all polynomial time truth-table autoreductions which satisfy the same side conditions as in Theorem 2.3; that is, for any oracle X, $M_e^X(x)$ is computed in time $e+x$ and X is queried only at places $y \leq 2^x$ which are different from x.

The choice of A_0 is sensitive to the success of the construction because it must be guaranteed that one can compute all elements queried by M_e. This is done by placing the more difficult elements at positions which can be accessed by an adapted search but not by a nonadaptive truth-table reduction.

First one takes a sequence $b_0 b_1 \ldots$ of bits which is random for computations using computation time 2^{n^3} but can be computed in time 2^{n^6} and let B be the set of all numbers of the form $2^n + \sum_{m<n} 2^m \cdot b_m$. B is in EXP. Furthermore let C be a set which is many-one EXP-complete and contains 0. The set A_0 is then given by

$$A_0 = \{\langle b, c, d\rangle : b \in B, c \in C, c \leq b, d \in \mathbb{N}\}.$$

Now one uses that the e-th truth-table autoreduction M_e queries at most x many places and so any query of it has at most Kolmogorov complexity $\log(e) + \log(x) + k$ for a constant k where the Kolmogorov complexity is measured with respect to time bound x^3. It follows that every $\langle b, c, d\rangle$ queried satisfies that b has also this bound (plus perhaps a constant) and that, whenever $\langle b, c, d\rangle$ is in A_0, then $b = 2^n + \sum_{m<n} 2^m \cdot b_m$ for some $n \leq 3(\log(x) + \log(e) + k')$ and $c \leq b$ where k' is a suitable constant independent of e, x. So the algorithm to compute all the values of A_0 at places queried by M_e at x with $e \leq \log^*(x)$ is exponential in x and this gives that the set A is also in EXP.

The construction gives that A is not infinitely-often tt-autoreducible in the same way as the construction in Theorem 2.3 that the set constructed there is not infinitely-often Turing autoreducible. It remains to show that in this theorem the set A is Turing autoreducible.

Note that for fixed b, c and the two third majority of the numbers $y = \langle b, c, d\rangle$ with

$0 \leq d \leq 3\log^*(b+c)+6$ satisfies that $A(y) = A_0(y)$. Thus one can every query whether $\langle b,c,0\rangle \in A_0$ reduce to polynomially many queries to A (omitting the query to x if it is among these queries) and so it is sufficient to give in the construction below the queries to A_0.

- For input x, compute the c such that $A(x) = C(c)$. This can be done without querying an oracle as C is many-one EXP-complete.
- For every $b \in B$, there is an n such that $b = 2^n + \sum_{m<n} 2^m \cdot b_m$. Then one of the following numbers is the next element of B: $b+2^n, b+2^{n+1}$ — the first one in case $b_n = 0$, the second one in case $b_n = 1$. So one can find for each member of B the next member of B by two queries to B and starting with 1, which is the minimum of B, one can find a $b \geq c$ such that $b \in B$ with $2\log(c)+2$ queries to B. Since $b \in B$ iff $\langle b,0,0\rangle \in A_0$, this computation can also be done with the same number of queries to A_0.
- Having this $b \geq c$ such that $b \in B$, one can determine $A(x)$ with one query to A_0 as $x \in A$ iff $\langle b,c,0\rangle \in A_0$.

The so constructed algorithm is a Turing autoreduction as the size $\log(c)$ of c is polynomially bounded in the size $\log(x)$ of x. Furthermore, as $3\log^*(b+c)+6 \leq 3\log(b+c)+12$, the number of queries to A is only by a factor polynomial in x greater than the number of queries to A_0 and so the whole algorithm queries A only polynomially often. One can easily verify, that the running time of the algorithm is also polynomial. ∎

Theorem 3.2. *For every polynomial-time computable and nondecreasing function $g \in Poly(n)$ there is a set A in EXP such that A is tt$(g(n)+1)$-autoreducible but not infinitely-often Turing$(g(n))$-autoreducible.*

Proof. Let $M_0, M_1, \ldots$ be a list of all polynomial time Turing$(g(n))$ autoreductions such that every value $M_e(x)$ can be computed in time $e+x$ and M_e queries only places below 2^x which are different from x. Furthermore, one can produce a polynomial time-computable partition Π of almost all natural numbers with the following properties:

- There is an integer m such that $x \in I$ for some $I \in \Pi$ iff $x \geq 2^m$, that is, $\cup_{I\in\Pi} I = \{2^m, 2^m+1, \ldots\}$;
- For every $I \in \Pi$ there is an integer $n(I)$ such that all numbers $x \in I$ satisfy $n(I) \leq \log(x) < n(I)+1$;
- The cardinality of every $I \in \Pi$ is either $g(n(I))+2$ or $2g(n(I))+3$.

As $g \in Poly(n)$, one can find an m such that $2(g(n)+2)^2 < 2^n$ holds for all $n \geq m$. For each $n \geq m$, one can find a, b such that $2^n = a(g(n)+2) - b$ and $0 \leq b \leq g(n)+1$. Note that $a > 2b$ and so one can partition the set $\{2^n, 2^n+1, \ldots, 2^{n+1}-1\}$ into b intervals of length $2g(n)+3$ followed by $a-2b$ intervals of length $g(n)+2$ and put these intervals into Π. The number $n(I)$ is for these intervals exactly the n with $I \subseteq \{2^n, 2^n+1, \ldots, 2^{n+1}-1\}$ and the

numbers $0, 1, \ldots, 2^m - 1$ do not belong to any interval.

Now given $I \in \Pi$ let $L(I)$ denote the $g(n(I))+2$ smallest and $H(I)$ the $g(n(I))+2$ largest elements in I. The overall construction of the set A will be such that for every interval $I \in \Pi$ the cardinalities of $L(I) \cap A$ and $H(I) \cap A$ are both even. Therefore one has the following $\mathrm{tt}(g(n)+1)$-autoreduction for A.

- If $x < 2^m$ then output "$x \notin A$" and halt.
- Otherwise find the $I \in \Pi$ with $x \in I$.
- If $x \in L(I)$ then query the $g(n)+1$ elements in $L(I) - \{x\}$ and let a be the cardinality of $A \cap (L(I) - \{x\})$ else query the $g(n)+1$ elements in $H(I) - \{x\}$ and let a be the cardinality of $A \cap (H(I) - \{x\})$.
- If a is odd then output "$x \in A$" else output "$x \notin A$".

This reducibility is definitely a $\mathrm{tt}(g(n)+1)$-autoreduction, can be done in time polynomial in $\log(x)$ and is correct since the cardinalities of $L(I) \cap A$ and $H(I) \cap A$ are even.

It remains to show that one can choose in exponential time the set A such that the resulting set is not infinitely-often Turing$(g(n))$-autoreducible. The construction is done in stages.

Construction. If $x < 2^m$ then let $A_{x+1} = \emptyset$ and $r_{e,x} = 0$ for all e. If $x \geq 2^m$ then determine the interval I with $x \in I$. Search for the least e such that there is a set D satisfying the following conditions.

Bound on e: $e \leq \log(x+1)$.

Respecting Restraints: $r_{e',x} < \min(I)$ for all $e' \leq e$.

Requirement e not yet done: There is no $x' < x$ such that $M_e^{A_x}(x')$ is defined, queries only places below $\min(I)$ and outputs something different from $A_x(x')$.

Requirement e needs attention: $M_e^{A_x \Delta D}(x)$ is defined and differs from $(A_x \Delta D)(x)$.

D not too large: D intersects only intervals J which either contain x or an $y > x$ queried by the $M_e^{A_x \Delta D}(x)$. Furthermore, $\min(I) \leq \min(D \cup \{x\})$, $\max(D \cup \{x\}) \leq 2^x$ and the sets $L(J) \cap D$ and $H(J) \cap D$ have an even number of elements for every interval $J \in \Pi$.

If the above procedure finds an e (which is the minimal one possible) and D is the corresponding set mentioned above then

$$\begin{aligned} A_{x+1} &= A_x \Delta D; \\ r_{e,x+1} &= 2^x + 2; \\ r_{e',x+1} &= r_{e',x} \text{ for all } e' \neq e; \end{aligned}$$

else nothing changes, that is, $A_{x+1} = A_x$ and $r_{e',x+1} = r_{e',x}$ for all e'.

Verification. The set A is in EXP as the search in every stage goes only through exponentially many possibilities ($e \leq \log(x+1)$ and at most $g(\log(x))+2$ many queries by M_e where g is polynomially bounded). Furthermore, every set D has on every interval of the form $L(I)$ and $H(I)$ an even number of elements so that the resulting set A has the same property: $A \cap L(I)$ and $A \cap H(I)$ have an even number of elements.

It follows from standard priority arguments that every e is found only finitely often by the above search conditions and that the restraints $r_{e,x}$ all take eventually a final value $r_{e,\infty}$.

Now assume that M_e^A never outputs an incorrect value. Consider any x in an interval $I \in \Pi$ such that $\min(I) > \max(\{r_{0,\infty}, r_{1,\infty}, \ldots, r_{e,\infty}\})$. Assume by way of contradiction that $M_e^A(x)$ is defined. Let E be the union of all intersections $J \cap (A_x \Delta A)$ where J contains either x or an $y > x$ which is queried by $M_e^A(x)$. Note that for all intervals J, $L(J) \cap E$ and $H(J) \cap E$ have an even number of elements. Furthermore, $A_x \Delta E$ and A coincide on all relevant elements, thus $M_e^{A_x \Delta E}(x) = A(x)$.

Let l and h be elements of I different from x and not queried by $M_e^{A_x \Delta E}(x)$ such that $l \in L(I)$, $h \in H(I)$ and both sets $\{l, h, x\} \cap L(I)$ and $\{l, h, x\} \cap H(I)$ have even cardinality; note that $L(I) = H(I) = I$ and $l = h$ in the case that I has $g(n)+2$ elements. It follows from easy cardinality arguments that l, h exist. Now let $D = E \Delta \{l, h, x\}$. The autoreduction M_e behaves at x for the sets $A_x \Delta E$ and $A_x \Delta D$ exactly in the same way, but the output is wrong in the case that one considers $A_x \Delta D$. So, this e witnessed by this D or some $e' < e$ would qualify for the search in stage x which contradicts to the restraints $r_{0,\infty}, r_{1,\infty}, \ldots, r_{e,\infty}$ being below x in the limit. From this contradiction it follows that $M_e^A(x)$ is undefined for almost all x.

So, A is in EXP, A is $\mathrm{tt}(g(n)+1)$-autoreducible but A is not infinitely-often $\mathrm{Turing}(g(n))$-autoreducible. ∎

An autoreduction is a bounded truth-table reduction iff there is a constant k such that the reduction makes for every input at most k queries. If a set A is $\mathrm{tt}(n+1)$-autoreducible but not infinitely-often $\mathrm{Turing}(n)$-autoreducible, then A is clearly tt-autoreducible but not infinitely-often btt-autoreducible. This gives the following corollary.

Corollary 3.3. *There is a set in EXP which is truth-table autoreducible but not infinitely-often bounded truth-table autoreducible.*

4 Notions of Approximability

A set is called (a, b)-recursive iff there is a function f such that for all distinct $x_1, x_2, \ldots, x_b$ the function f computes a tuple $(y_1, y_2, \ldots, y_b)$ of bits such that at least a of the equations $A(x_k) = y_k$ are true. If there are a, b such that $1 \leq a \leq b$ and if there is a polynomial time computable function f such that A is (a, b)-recursive via f then A is called approximable [9] and if in addition $2a > b$ then A is called easily approximable.

In the recursion theoretic setting, every set which is $(1, b)$-recursive for some b is also autoreducible [16]. This does not carry over to complexity theory: Returning to the world

of polynomial time computations, supersparse sets are $(1,2)$-recursive but not Turing autoreducible. Supersparse sets are related to k-cheatable sets [8]. Nevertheless, approximable sets are still infinitely-often autoreducible.

Proposition 4.1. *Every approximable set is infinitely-often btt-autoreducible.*

Proof. Assume that A is approximable via f and let l be the largest constant such that for infinitely many inputs of the form $x+1, x+2, \ldots, x+k$ at least l of the bits $y_1, y_2, \ldots, y_k$ are wrong. Let u be so large that it never happens for an $x > u$ that more than l of these bits are wrong. Now A is btt-autoreducible as follows.

Algorithm. Ignore inputs $x < u+k$. For $x \geq u+k$ query A at all numbers x' with $x-k < x' < x+k$ and $x' \neq x$. If there is a x' with $x-k \leq x' < x$ such that the answers $y_1, y_2, \ldots, y_k$ computed by f from input $x'+1, x'+2, \ldots, x'+k$ satisfy that $y_{k'} \neq A(x'+k')$ for l numbers $k' \in \{1,2,\ldots,k\} - \{x-x'\}$ then predict $y_{x-x'}$ for $A(x)$ else do not make any prediction.

Verification. The correctness follows from the fact that there are at most l differences between $A(x'+k')$ and $y_{k'}$ and these errors have been localized at places different from x. That this reduction works for infinitely many x is a consequence of the choice of l. ∎

Ogihara [21] also considered sets which are $(1, b(n))$-recursive where b is a function in $Poly(n)$ and the parameter n is $\log(\max\{1, x_1, x_2, \ldots, x_b\})$ where $x_1, x_2, \ldots, x_b$ is the input to the function to compute the approximation. This generalized notion does no longer enforce that A is infinitely-often autoreducible.

Example 4.2. *The set from Theorem 2.3 satisfies Ogihara's generalized approximability condition but is not infinitely-often Turing autoreducible.*

Proof. It is sufficient to show that the set A constructed in Theorem 2.3 is $(1, b(n))$-recursive with $b(n) = 6\log^*(n) + 3$: On input $x_1, x_2, \ldots, x_{b(n)}$, let

$$y_k = \begin{cases} 0 & \text{if } x_k > \log\log(n); \\ A(x_k) & \text{if } x_k \leq \log\log(n). \end{cases}$$

and predict $(y_1, y_2, \ldots, y_{b(n)})$. If $x_k \leq \log\log(n)$ then $A(x_k) = y_k$ and this prediction is correct. Otherwise one knows that between $\log\log(n)$ and n the set A has at most $6\log^*(n) + 2$ many elements. It follows that at least one of the predicted 0s is correct. Furthermore, the computations involved can all be done in polynomial time since $A(x_k)$ can be computed in time $2^{Poly(\log\log(n))}$ whenever $x_k \leq \log\log(n)$. ∎

An easily approximable set A has the property that every set B Turing reducible to A is also tt-reducible to A. This property is called *T-easy*. The next theorem shows that every T-easy set is either infinitely-often autoreducible or satisfies Ogihara's generalized approximability notion with $a = 1$ and $b = \log^2(n)$.

Theorem 4.3. *Every set A satisfies at least one of the following properties.*

(a) *Not T-easy: There is a set B which is polynomial time Turing reducible but not polynomial time truth-table reducible to A;*

(b) *Infinitely-often truth-table autoreducible: For infinitely many z, $A(z)$ can be computed by queries to places different from z;*

(c) *Generalized approximable: A is $(1, 1+\log^2(n))$-recursive via some function computable in polynomial time.*

Proof. Let A be any set. Furthermore, define a tuple-function which at input $x_1, x_2, \ldots, x_m$ computes the binary representations $u_0, u_1, \ldots, u_m$ of these numbers and outputs a number, denoted as $\langle x_1, x_2, \ldots, x_m \rangle$, which has the ternary representation $u_1 2 u_2 2 \ldots 2 u_m 2$. Note that $\langle x_1, x_2, \ldots, x_m \rangle > x_k$ for $k = 1, 2, \ldots, m$. Furthermore, $n = \log(\max\{1, x_1, x_2, \ldots, x_m\})$ satisfies $\langle x_1, x_2, \ldots, x_m \rangle < 3^{(n+3)m}$. Let $y(x_1, x_2, \ldots, x_m : A)$ be the number represented by the binary string $A(x_1)A(x_2)\ldots A(x_n)$. Depending on the question what type of reducibilities from

$$B = \{\langle x_1, x_2, \ldots, x_m \rangle : \langle x_1, x_2, \ldots, x_m, y(x_1, x_2, \ldots, x_m : A)\rangle \in A\}$$

to A exist, one of three properties hold.

(a) B is not truth-table reducible to A. Then A *is not T-easy.* This is verified by showing that B is Turing reducible to A as follows. One first queries whether $x_1, x_2, \ldots, x_m \in A$, then computes $y(x_1, x_2, \ldots, x_m : A)$ and at last queries whether $\langle x_1, x_2, \ldots, x_m, y(x_1, x_2, \ldots, x_m : A)\rangle$ is in A.

(b) B is truth-table reducible to A by a reduction which for infinitely many tuples $\langle x_1, x_2, \ldots, x_m \rangle$ computes $B(\langle x_1, x_2, \ldots, x_m \rangle)$ without querying A whether $\langle x_1, x_2, \ldots, x_m, y(x_1, x_2, \ldots, x_m : A)\rangle$ is in A. Then A *is infinitely-often truth-table autoreducible* by f defined as follows. Let $F(x)$ denote the set of queries which the truth-table reduction from B to A makes at input x. On input z, f checks whether there is a set E and numbers $m, x_1, x_2, \ldots, x_m$ such that

- $z = \langle x_1, x_2, \ldots, x_m, y(x_1, x_2, \ldots, x_m : E)\rangle$ and
- $z \notin F(E(x_1)E(x_2)\ldots E(x_m)) \cup \{x_1, x_2, \ldots, x_m\}$.

If so, then f queries A at the members of the set $F(\langle x_1, x_2, \ldots, x_m \rangle) \cup \{x_1, x_2, \ldots, x_m\}$ and outputs the result of the truth-table reduction from B to A computed for the value $B(\langle x_1, x_2, \ldots, x_m \rangle)$ in the case that $A(x_1) = E(x_1), \ldots, A(x_m) = E(x_m)$. If z is not of the above form or if the supposed values of E do not coincide with A or $z \in F(\langle x_1, x_2, \ldots, x_m \rangle)$ then $f(z)$ is undefined. Note that whenever $f(z)$ is defined, the value computed coincides with $A(z)$ and f does not query $A(z)$ as $x_1, x_2, \ldots, x_m < \langle x_1, x_2, \ldots, x_m, y(x_1, x_2, \ldots, x_m : E)\rangle = z$ and $z \notin F(E(x_1)E(x_2)\ldots E(x_m))$.

(c) The two previous cases do not hold. Then A *is* $(1, b(n))$*-recursive with* $b(n) = 1 + \lfloor \log^2(n) \rfloor$. As (a) does not hold, there is a tt-reduction from B to A. Let $F(x)$ be the set of queries made at input x. There is a constant c such that the cardinality of $F(x)$ is at most $\log^c(x)$ for almost all x. If n is sufficient large and $x = \langle x_1, x_2, \ldots, x_m \rangle$

with $x_1 < x_2 < \ldots < x_m$ and $n = \lfloor \log(x_m) \rfloor$ and $m = b(n)$, then the following facts hold: $y(x_1, x_2, \ldots, x_m : A) \in F(\langle x_1, x_2, \ldots, x_m \rangle)$ as (b) does not hold, $n > m$ and $(\log(3) \cdot m \cdot (n+3))^c < n^{4c} < n^{\log(n)}$. As $2^m > n^{\log(n)}$, it follows that there is a number y with binary representation $d_1 d_2 \ldots d_m$ such that $\langle x_1, x_2, \ldots, x_m, y \rangle \notin F(\langle x_1, x_2, \ldots, x_m \rangle)$. Thus at least one of the conditions $A(x_1) \neq d_1, A(x_2) \neq d_2, \ldots, A(x_m) \neq d_m$ holds and so A is $(1, b(n))$-recursive by outputting the vector $(1 - d(x_1), 1 - d(x_2), \ldots, 1 - d(x_m))$. The remaining case is where n is too small. But there are only finitely many $x_1, x_2, \ldots, x_m$ where this can happen and one can output the characteristic vector $(A(x_1), A(x_2), \ldots, A(x_m))$ in these finitely many cases. One can easily verify that the proposed algorithm runs in polynomial time and so A is $(1, b(n))$-recursive. ∎

5 Hausdorff Dimension

Hausdorff [14] introduced a generalized notion of dimension for metric spaces; it also enables to measure the size of fractal objects. Lutz [19] adapted the notion for complexity theory in order to measure the size of subclasses of the natural numbers. The following definition – one among several equivalent ones – defines the Hausdorff-dimension in terms of the growth rate of the capital accumulated by a gambler who bets inductively on the bits of the characteristic functions of any set in the given class. Such growth rate functionals are called martingales.

Definition 5.1. A quasipolynomial time computable functional f is called a *martingale*, iff it maps binary strings to positive numbers, the empty string to 1 and satisfies

$$2f(a_0 a_1 \ldots a_x) = f(a_0 a_1 \ldots a_x 0) + f(a_0 a_1 \ldots a_x 1)$$

for all binary strings $a_0 a_1 \ldots a_x$. The *Hausdorff-dimension* (with respect to EXP) of a class S is the infimum of all s such that there is a quasipolynomial time computable martingale f succeeding on every set in $A \in S$ with growth rate 2^{1-s}, that is, f satisfies

$$(\forall A \in S)\,(\exists^\infty x)\,[f(A(0)A(1)\ldots A(x)) \geq 2^{(1-s)x}].$$

Remark 5.2. The class of all sets as well as the class EXP have Hausdorff-dimension 1. The upper bound is witnessed by the martingale mapping all strings to 1, the lower bound is obtained by constructing for any given quasipolynomial martingale f the set A in EXP which satisfies for $x = 0, 1, \ldots$ that

$$f(A(0)A(1)\ldots A(x)A(x+1)) \leq f(A(0)A(1)\ldots A(x)),$$

that is, $A(x+1)$ takes just the value adversary to $A(x)$. Note that in EXP one cannot have one set doing this for all martingales due to the fact that every single set in EXP is predictable by a suitable quasipolynomial time computable martingale. There are also sets A random for all quasipolynomial time computable martingales and thus satisfy that the Hausdorff-dimension of $\{A\}$ is 1, but no such A is in EXP.

Ambos-Spies, Merkle, Reimann and Stephan [4, Corollary 16] considered Hausdorff-dimension adapted to the class $E = \{A : (\exists c)\,(\exists M)\,(\forall x)\,[M \text{ computes } A(x) \text{ in time } c + x^c]\}$. They showed that the class of many-one autoreducible sets in E has Hausdorff-dimension 1. This fact directly carries over to EXP.

Fact 5.3. *The class $\{A$ in EXP: $(\forall x)\,[A(x) = A(x^2)]\}$ consists only of many-one autoreducible sets and has Hausdorff-dimension* 1*. In particular, the classes of the many-one autoreducible, truth-table autoreducible and Turing autoreducible sets in EXP have Hausdorff-dimension* 1*.*

An interesting obvious question is to determine the Hausdorff-dimension of the class of those sets in EXP which are not infinitely-often r-autoreducible. The next theorem shows, that the Hausdorff-dimension is already 1 for the case of the polynomial time Turing reducibility; it then follows also for the other polynomial time reducibilities r. As Ebert [12, 13] showed that every set which is x^3-random is already infinitely-often autoreducible, the class S is one of the natural witnesses that there are classes which have Hausdorff-dimension 1 and measure 0 with respect to quasipolynomial time martingales.

Theorem 5.4. *The class of all sets in EXP which are not infinitely-often autoreducible has Hausdorff-dimension* 1*.*

Proof. The basic idea of the proof is to use the same construction as in Theorem 2.3. But one has to do the following changes. One has to construct for every quasipolynomial time martingale f such a set A. Furthermore, for given f, A_0 has to be chosen such that f does not have a fast growth rate neither on A_0 nor on any supersparse variant of A_0. In particular the supersparse variant A of A_0 will satisfy that there is no $s < 1$ with $(\exists^\infty x)\,[f(A(0)A(1)\ldots A(x)) \geq 2^{(1-s)\cdot x}]$. Furthermore, A_0 has to be chosen such that not only A_0 itself but also the A constructed from A_0 is in EXP.

To meet these constraints, one chooses the polynomial time Turing reductions $M_0, M_1, \ldots$ a bit more restrictive than in Theorem 2.3, namely the M_e have to satisfy the following properties:

- $M_e^X(x)$ queries any given oracle X at up to $(\log(x))^{\log\log(x)}$ many places; these places are below 2^x and different from x;
- $M_e^X(x)$ needs at most running time $e + (\log(x))^{\log\log(x)}$.

Note that the bound $(\log(x))^{\log\log(x)}$ is superpolynomial and thus one has, starting with a given reduction, only to modify it on finitely many inputs, where one can put the correct result into a table so that no query is necessary at all. Thus whenever A is infinitely-often autoreducible then A is infinitely-often autoreducible by some M_e.

As f might behave on A very differently as on A_0, one has to define A_0 such that the growth rate of f does not only on A_0 but also on all supersparse variants of A_0 fail to reach 2^{1-s} for any $s < 1$.

So one considers the following functional f': Let r_D be the product of all $1 + 1/(1 +$

$d \cdot d$) where $d \in D$, $r_\emptyset = 1$. Now $f'(B(0)B(1)\ldots B(x))$ is the sum over all terms $r_D \cdot f(C(0)C(1)\ldots C(x))$ where $C \subseteq \{0,1,\ldots,x\}$, $D = \{y \leq x : B(y) \neq C(y)\}$ and D is a supersparse set. The functional f' is no longer a martingale, but it satisfies the following condition:

$$f'(a_0a_1\ldots a_x0) + f'(a_0a_1\ldots a_x1) \leq 2 \cdot \tfrac{2+(x+1)\cdot(x+1)}{1+(x+1)\cdot(x+1)} \cdot f'(a_0a_1\ldots a_x).$$

Now one defines the set A_0 inductively as follows. $A_0(0) = 0$. $A_0(x+1) = 0$ iff one of the following two conditions holds and $A_0(x+1) = 1$ otherwise:

(a) there is an $y < x^{1/5}$, an $e \leq \log^*(x)$ and a supersparse set $D \subseteq \{0,1,\ldots,2^x\}$ such that $M_e^{(A_0 \cap \{0,1,\ldots,x\})\Delta D}(y)$ queries x;

(b) $f'(a_0a_1\ldots a_x0) \leq \tfrac{2+(x+1)\cdot(x+1)}{1+(x+1)\cdot(x+1)} \cdot f'(a_0a_1\ldots a_x)$.

Note that the search over the queried elements can be done in exponential time: every M_e queries at most $(\log(x))^{\log\log(x)}$ many elements and at most $(\log^*(x)+1)^2$ of them can differ from the corresponding value of $A_0 \cap \{0,1,\ldots,x\}$ as one searches over answers given by $A_0\Delta D$ and not just A_0. So one has to consider $((\log(x))^{\log\log(x)})^{(\log^*(x)+2)^2}$ many computation paths for every $y \leq x^{1/5}$ and $e \leq \log^*(x)$, each path might need up to $\log^*(x) \cdot (\log(x))^{\log\log(x)}$ many steps. As except $x^{1/5}$ all these factors grow slower than every function x^q, $q > 0$, there is a constant c such that, for every x, the number of steps to be simulated is at most $x^{1/4} + c$. As every 0 caused by condition (a) in the construction of A_0 is due to a query of one of these simulated paths, there are at most $x^{1/4} + c$ many of places $y \leq x$ where $A_0(y+1) = 0$ due to condition (a).

If $A_0(x+1) = 0$ is caused by condition (a), then the value of f' can go up by the factor $2 \cdot \frac{2+(x+1)\cdot(x+1)}{1+(x+1)\cdot(x+1)}$ else $A(x+1)$ is chosen such that the value of f' goes only up by half of that, that is, by the factor $\frac{2+(x+1)\cdot(x+1)}{1+(x+1)\cdot(x+1)}$. The product over $f'(A(0))$ and all numbers $\frac{2+(x+1)\cdot(x+1)}{1+(x+1)\cdot(x+1)}$ is bounded by a constant d. So it follows that, for every x, $f'(A_0(x)A_1(x)\ldots A_x(x)) \leq 2^{x^{1/4}+c} \cdot d$. The construction of Theorem 2.3 gives that A is a supersparse variant of A_0. Let $B = A\Delta A_0$. For almost all x, $D = B \cap \{0,1,\ldots,x\}$ has at most $(\log^*(x))^2$ many elements, $1/r_D \leq x^{|D|} \leq 2^{x^{1/4}}$ and $f(A(0)A(1)\ldots A(x)) \leq f'(A(0)A(1)\ldots A(x))/r_D \leq 2^{x^{1/4}+c+1} \cdot d$. It follows that there is no $s < 1$ such that $f(A(0)A(1)\ldots A(x)) \geq 2^{(1-s)x}$ for infinitely many x.

It remains to show that A is in EXP. Note that it follows from the construction of Theorem 2.3, that at every stage x and every D considered in the search condition, the set $A_x\Delta D$ is a supersparse variant of A_0. It follows from the definition of A_0 that whenever $M_e^{A_x\Delta D}(x)$ queries an element $y > x^5$, then $A_0(y) = 0$. As A_0 is in EXP, there is a polynomial p such that one can compute in time $2^{p(\log(x))}$ the value of $A_0(y)$ for any given $y \leq x^5$. It follows that one can compute $A_0(y)$ in all queried places in time exponential in $\log(x)$. Every stage in the construction changes the approximation of A at less than x places, so one can bookkeep these changes and compute A in exponential time. This completes the proof. ∎

Acknowledgments. The authors would like to thank Klaus Ambos-Spies, Jack H. Lutz and Wolfgang Merkle for helpful discussions; furthermore, Jack H. Lutz proposed to investigate the Hausdorff-dimension of the class of sets in EXP which are not infinitely-often autoreducible.

References

[1] Klaus Ambos-Spies. Resource-bounded genericity. In S. B. Cooper et al., editor, *Computability, Enumerability, Unsolvability*, volume 224 of *London Mathematical Society Lecture Notes Series*, pages 1–59. Cambridge University Press, 1996.

[2] Klaus Ambos-Spies, Hans Fleischhack and Hagen Huwig. Diagonalizations over polynomial time computable sets. *Theoretical Computer Science* 51:177-204, 1997.

[3] Klaus Ambos-Spies and Elvira Mayordomo. Resource-bounded measure and randomness. In A. Sorbi, editor, *Complexity, Logic, and Recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Mathematics*, pages 1–47, 1997.

[4] Klaus Ambos-Spies, Wolfgang Merkle, Jan Reimann and Frank Stephan. Hausdorff dimension in exponential time. *Proceedings Sixteenth Annual IEEE Conference on Computational Complexity*, IEEE Computer Society, 210–217, 2001.

[5] Klaus Ambos-Spies, Hans-Christian Neis and Sebastiaan A. Terwijn. Genericity and measure for exponential time. *Theoretical Computer Science*, 168:3–19, 1996.

[6] Klaus Ambos-Spies, Sebastiaan A. Terwijn and Xizhong Zheng. Resource bounded randomness and weakly complete problems. *Theoretical Computer Science*, 172:195–207, 1997.

[7] José L. Balcázar and Elvira Mayordomo. A note on genericity and bi-immunity. *Proceedings of the Tenths Annual Structure in Complexity Theory Conference*, IEEE Computer Society Press, pages 193–196, 1995.

[8] Richard Beigel. Bi-immunity results for cheatable sets. *Theoretical Computer Science*, 73:249–263, 1990.

[9] Richard Beigel, Martin Kummer and Frank Stephan. Approximable sets. *Information and Computation*, 120:304–314, 1995.

[10] W. Blum. Denksport für Hutträger. *Die Zeit*, Hamburg, 03 May 2001.

[11] Harry Buhrman, Lance Fortnow, Dieter van Melkebeek and Leen Torenvliet. Using autoreducibility to separate complexity classes. *Siam Journal on Computing*, 29(5):1497–1520, 2000.

[12] Todd Ebert. *Applications of Recursive Operators to Randomness and Complexity.* PhD Thesis, University of California at Santa Barbara, 1998.

[13] Todd Ebert, Wolfgang Merkle and Heribert Vollmer. On the autoreducibility of random sequences. *SIAM Journal on Computing*, 32(6):1542–1569, 2003.

[14] Felix Hausdorff. Dimension und äusseres Maß. *Mathematische Annalen*, 79:157–189, 1919.

[15] Carl G. Jockusch and Micheal S. Paterson. Completely autoreducible degrees. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 22:571–575, 1976.

[16] Martin Kummer and Frank Stephan. Recursion theoretic properties of frequency computation and bounded queries. *Information and Computation*, 120:59–77, 1995.

[17] Jack H. Lutz. Category and measure in complexity classes. *SIAM Journal on Computing*, 19:1100–1131, 1990.

[18] Jack H. Lutz. Almost everywhere high non-uniform complexity. *Journal of Computer and System Sciences*, 44:220–258, 1992.

[19] Jack H. Lutz. Dimension in complexity classes. *Proceedings Fifteenth Annual IEEE Conference on Computational Complexity* (*formerly Structure in Complexity Theory*), IEEE Computer Society, 158–169, 2000.

[20] Piergiorgio Odifreddi. *Classical recursion theory.* North-Holland, Amsterdam, 1989.

[21] Mitsunori Ogihara. Polynomial-time membership comparable sets. In *Proceedings of the 9th Conference on Structure in Complexity Theory*, IEEE Computer Society Press, 2–11, 1994.

[22] Sarah Robinson. Why mathematicians now care about their hat color. The New York Times, New York, 10 April 2001.

[23] Claus-Peter Schnorr. Optimal algorithms for self-reducible problems. In *Third International Colloquium on Automata, Languages and Programming*, ICALP 1976, University of Edinburgh Press, pages 322–337, 1976.

[24] Boris A. Trakhtenbrot. On autoreducibility. *Soviet Mathematics Doklady* 11:814–817, 1970.

[25] Osamu Watanabe. A comparison of polynomial time completeness notions. *Theoretical Computer Science* 54:249–265, 1987.