Effective Search Problems

Martin Kummer and Frank Stephan *

Institut für Logik, Komplexität und Deduktionssysteme,

Universität Karlsruhe, D-76128 Karlsruhe, Germany.

E-mail: {kummer; fstephan}@ira.uka.de

Abstract

The task of computing a function F with the help of an oracle X can be viewed as a search problem where the cost measure is the number of queries to X. We ask for the minimal number that can be achieved by a suitable choice of X and call this quantity the *query complexity* of F. This concept is suggested by earlier work of Beigel, Gasarch, Gill, and Owings on "Bounded query classes". We introduce a fault tolerant version and relate it with Ulam's game. For many natural classes of functions F we obtain tight upper and lower bounds on the query complexity of F. Previous results like the Nonspeedup Theorem and the Cardinality Theorem appear in a wider perspective.

1991 Mathematics Subject Classification: Primary 03D20;
Secondary 68Q15, 68R05
Keywords: Search problems, bounded queries, query complexity, recursive functions

1 Introduction

The task of computing a function F with the help of an oracle $X \subseteq \omega$ (ω is the set of all natural numbers) can be viewed as a search problem where the cost measure is the number of queries to X. We ask for the minimal number that can be achieved by a suitable choice of X and call

*Supported by the Deutsche Forschungsgemeinschaft (DFG) grant Me 672/4-1.

this quantity the *query complexity* of F. A more expressive notion is the *enumeration complexity* of F. Here one considers algorithms that enumerate on input x a set of finitely many possibilities with $F(x)$ among them. The enumeration complexity of F is the least number k such that there is an enumeration algorithm for F which enumerates at most k possibilities for each x. It is known that the query complexity of F is uniquely determined by its enumeration complexity.

Query complexity and enumeration complexity were introduced in [2, 6]. In the first part of the paper we consider a fault tolerant version of query complexity based on *l-robust* algorithms: these are required to output the correct value even if up to l queries to the oracle are answered incorrectly. We characterize the l-robust query complexity in terms of the enumeration complexity and information theoretic bounds for the corresponding finite search problem: *Ulam's searching game with l lies.*

In the second part of the paper we determine the extremal values of the enumeration complexity (and hence of the various query complexities) for large classes of natural functions. Historically, the first function whose enumeration complexity was studied is the n-fold characteristic function $C_n^A(x_1, \ldots, x_n) = (\chi_A(x_1), \ldots, \chi_A(x_n))$, $A \subseteq \omega$. (C_n^A has traditionally been called F_n^A in previous papers.) It is proved in [2, 6] that the enumeration complexity of C_n^A is at least $n+1$ if A is nonrecursive ("Nonspeedup Theorem"). In [14] the analogous result ("Cardinality Theorem") is obtained for the n-fold cardinality function $\#_n^A(x_1, \ldots, x_n) = |\{i : x_i \in A\}|$. Utilizing the proof of the latter result we determine for any finite function $f : \{0,1\}^n \to S$ the minimal query complexity of the function $F^A := f \circ C_n^A$ for nonrecursive A. Intuitively, C_n^A gathers information from A, but f says what to do with that information. It might be possible to get what f wants without first having to compute C_n^A. The above mentioned results are the special cases $f = id_n$ and $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$, respectively. We present some further examples and applications. For instance we show that for *every* nonrecursive set A there is a function with query complexity k that can be computed with k queries *to* A. This shows that k (sequential) queries to any nonrecursive set cannot be simulated by $k-1$ queries to any other set.

We then transfer the results from characteristic functions to functions with finite range, i.e. we consider $g(F(x_1), \ldots, F(x_n))$ where $g : \{0, \ldots, p\}^n \to S$ and $F : \omega \to \{0, \ldots, p\}$. Furthermore, for arbitrary nonrecursive functions F we show that the sum $\lambda x_1, \ldots, x_n.\ F(x_1) + \cdots + F(x_n)$ has at least the enumeration complexity $n+1$.

Finally, we discuss the relation with a strong form of enumeration complexity where the algorithm is required to output all possibilities at once. This leads to an alternative proof of the Cardinality Theorem which is of methodological interest.

Definition 1.1 Let $k \in \omega$ and $X \subseteq \omega$. A function $F : \omega \to \omega$ is in $FQ(k, X)$ if f is recursive in X via an algorithm that makes at most k (sequential) queries to X.

The *query complexity* $q(F)$ of F is the least number k such that $F \in FQ(k, X)$ for some $X \subseteq \omega$. If k does not exist then $q(F) = \infty$.

The *enumeration complexity* $e(F)$ of F is the least number m such that F is computable by m partial recursive functions, in the sense that there are $\varphi_{e_1}, \ldots, \varphi_{e_m}$ such that for every x there is some j with $F(x) = \varphi_{e_j}(x)$. Equivalently, $e(F)$ is the least number m such that there is an algorithm that enumerates for each x a set of at most m numbers including $F(x)$. If m does not exist then $e(F) = \infty$.

All of these notions are defined for n-place functions F by identifying $F(x_1, \ldots, x_n)$ with $F(\langle x_1, \ldots, x_n \rangle)$ via a standard pairing function $\langle \cdots \rangle$.

Fact 1.2 *If A is nonrecursive then*

- $e(C_n^A) \geq n + 1$ *("Nonspeedup Theorem"* [6]*).*
- $e(\#_n^A) \geq n + 1$ *("Cardinality Theorem"* [14]*).*

The following basic fact shows that the query complexity is determined by the enumeration complexity.

Fact 1.3 [2, 6] $q(F) = \lceil \log_2 e(F) \rceil$.

Note that the query complexity of F equals the number of queries that are needed in the binary search process on a domain of $e(F)$ elements with all tests admitted [1]. This observation will be generalized in Section 2.

The above notions can be adapted to *promise problems* as explained in [11, Section 13]. In particular Fact 1.3 generalizes. Beigel and Gasarch [4] consider the following promise problem CN_c: "Given the index of a recursive graph G of chromatic number at most c. Determine the chromatic number of G." Using an easy reduction to the Nonspeedup Theorem they show that the enumeration complexity of CN_c is $c + 1$.

2 Ulam's Game and Robust Computations

In Ulam's searching game on $S = \{1, \ldots, n\}$ with l lies the first player thinks of an integer $i \in S$. The second player must determine i with questions of the form "Is $i \in D$?", where $D \subseteq S$. He may use previous answers before deciding his next question. The first player is permitted to lie but she may lie at most l times through the course of the game. Let $u(n, l)$ denote the least number of questions that are needed for the second player to determine the unknown integer. It is well-known that $u(n, 0) = \lceil \log_2 n \rceil$, but no explicit formula is known for $u(n, l)$ in general. For some interesting partial results see [16, 19, 21] and the literature cited therein. For example it is shown in [19] that $u(n, l) = \log_2 n + l \cdot \log_2 \log_2 n + O(l \cdot \log_2 l)$.

Definition 2.1 Let $k \in \omega$ and $X \subseteq \omega$. A function $F : \omega \to \omega$ is in $FQR(k, l, X)$ if there is an algorithm that computes F with at most k queries to X and outputs the correct value even if up to l queries are answered incorrectly. We call such an algorithm *l-robust*.

The *l-robust query complexity* $r(F, l)$ of F is the least number k such that $F \in FQR(k, l, X)$ for some $X \subseteq \omega$.

The following result shows that the l-robust query complexity of F is completely determined by $e(F)$.

Theorem 2.2 $r(F, l) = u(e(F), l)$.

Proof: $r(F, l) \leq u(e(F), l)$: Assume that F is computable by the $e(F)$ partial recursive functions $\psi_1, \ldots, \psi_{e(F)}$. Let $g(x)$ denote the least i with $F(x) = \psi_i(x)$, $S = \{1, \ldots, e(F)\}$, and define $X = \{cod(x, D) : D \subseteq S \ \wedge\ g(x) \in D\}$ where *cod* is an effective coding of pairs of natural numbers and finite sets. The l-robust algorithm with input x simulates an optimal strategy for the second player in Ulam's game on S with l lies. If the strategy asks "Is $i \in D$?" then the algorithm queries "$cod(x, D) \in X$?" and uses the answer from the oracle. In this way $i = g(x)$ is determined robustly with at most $u(e(F), l)$ queries. Finally, $F(x)$ is obtained by computing $\psi_i(x)$.

$u(e(F), l) \leq r(F, l)$: Assume that there is an l-robust algorithm M^X that computes F with at most $k = r(F, l)$ queries. We may assume that the algorithm makes at most k queries for every oracle. We call y *potentially correct* for x iff there is a finite set B such that every computation path of $M^B(x)$ that is obtained by answering at most l queries incorrectly terminates with output y.

It is easy to see that $F(x)$ is potentially correct for x, and the set of all numbers that are potentially correct can be enumerated uniformly in x. Therefore, by the definition of $e(F)$, there exists x_0 such that at least $e(F)$ values are enumerated for x_0, i.e. there are $e(F)$ pairwisely different $y_1, \ldots, y_{e(F)}$ which are potentially correct for x_0. Choose finite sets B_j such that y_j is potentially correct for x_0 via $B = B_j$.

We claim that the following strategy for the second player succeeds with at most k questions in Ulam's game on $\{1, \ldots, e(F)\}$ with l lies.

Strategy to find out the unknown integer i:
Simulate the computation of $M^{\emptyset}$ to find out which number a is queried first. Ask

$$\text{“Is } i \in \{m : a \in B_m\}\text{?”}$$

If the answer is positive follow the YES-path (i.e., simulate $M^{\{a\}}$), otherwise follow the NO-path (i.e., simulate $M^{\emptyset}$). Find out the next query and continue recursively until the computation path terminates. If it outputs y_j then output j. $\square$

Note that the question “Is $i \in \{m : a \in B_m\}$?” is equivalent to “Is $a \in B_i$?”. Since at most l queries are answered incorrectly it follows by the definition of y_i and B_i that the computation path terminates with output y_i, i.e. the strategy outputs the correct value. By construction there are at most k questions. Hence we conclude that $u(e(F), l) \leq r(F, l)$. ∎

Remarks: a) If we consider only algorithms that are total for every oracle we obtain the class $FQC(k, X)$ (see [11, Definition 22]). Let $FQRC(k, l, X)$ be the corresponding robust version. Define $q'(F)$ and $r'(F, l)$ as $q(F)$ and $r(F, l)$ by using FQC and $FQRC$ instead of FQ and FQR. Let $e'(F)$ denote the least number m such that F is computable by a set of m *total* recursive functions. Then we get: $r'(F, l) = u(e'(F), l)$. An analogous statement holds if we replace “total” by “polynomial time bounded”. See section 6 for a connection between $e'(F)$ and $e(F)$.

b) We can also obtain a version for parallel queries. Let $r_p(F, l)$ denote the *parallel* l-robust query complexity of F (i.e. the queries are asked all at once). Let $u_p(n, l)$ be the least number of questions that are needed in a *predetermined* strategy (where the questions of the second player are independent from the answers of the first player) for the second player in Ulam's game. Then $r_p(F, l) = u_p(e(F), l)$. It is well-known that $u_p(n, 0) = u(n, 0)$ (see [1, p. 19]), hence it makes no difference whether we define $q(F)$ with respect to sequential or parallel queries. However, this no longer true for $l > 0$; for instance, $u_p(2^{78}, 2) > u(2^{78}, 2) = 90$ (see [16, p. 301]).

c) Theorem 2.2 can be directly adapted to promise problems. In particular, any l-robust algorithm that solves CN_c (as defined in the note after Fact 1.3) needs at least $u(c+1,l)$ queries. As in [4] this number of queries suffices if we choose K^{tt} (the truth-table cylinder of K) as our oracle.

Finally there is a slightly refined version as in [5] (see also [3]). Call a strategy for the second player in Ulam's game on $S = \{0, \ldots, n\}$ with l lies an *f-strategy* ($f : S \to \omega$) iff for all $i \in S$ it asks at most $f(i)$ questions if the first player chooses i. Generalizing [5, Theorem 9] one can show that there is an f-strategy iff there is an l-robust algorithm that solves the promise problem CN_c and for any index of a recursive graph with chromatic number $c \leq n$ the algorithm makes at most $f(c)$ queries.

3 Query Complexity of Operations on Sets

In this section $f : \{0,1\}^n \to S$ denotes a fixed finite function. As explained in the introduction for any $A \subseteq \omega$ we define $F^A : \omega^n \to S$ by $F^A := f \circ C_n^A$. Let $e(f, A) = e(F^A)$. We identify S with a finite subset of ω via some fixed implicit coding. There are two extremal values associated with f:

$$\begin{aligned} e_{\min}(f) &:= \min\{e(f,A) : A \text{ is nonrecursive}\} \text{ and} \\ e_{\max}(f) &:= \max\{e(f,A) : A \subseteq \omega\}. \end{aligned}$$

It is easy to determine $e_{\max}(f)$:

Proposition 3.1 $e_{\max}(f) = |\mathrm{range}(f)|$.

Proof: By a straightforward diagonalization one can construct A with $e(f,A) = |\mathrm{range}(f)|$. Alternatively, if $e(f,A) < |\mathrm{range}(f)|$ then $e(C_n^A) < 2^n$. But in [6, Theorem 14, Lemma 19] it is shown that there exist so-called "superterse" sets A with $e(C_n^A) = 2^n$ for all n. Hence $e(f,A) = |\mathrm{range}(f)|$ if A is superterse. ∎

The value $e_{\min}(f)$ is more interesting since it determines the borderline of decidability. To determine $e_{\min}(f)$ is a nontrivial task however, though most of the work is already done in [14]. We start with a particular combinatorial quantity of f:

Definition 3.2 Let L_n be the set of all sequences $(v_0, \ldots, v_n)$ of vectors from $\{0,1\}^n$ such that $v_0 = 0^n$ and v_{i+1} is obtained from v_i by flipping one component from 0 to 1 ($0 \leq i < n$). In particular, v_i contains exactly i 1s and $v_n = 1^n$. Further let

$$s(f) := \max\{|\{f(v_i) : 0 \leq i \leq n\}| : (v_0, \ldots, v_n) \in L_n\}.$$

Note that in any case $s(f) \leq n+1$. Recall that a set A is called *semirecursive* iff A is an initial segment of a recursive linear ordering of ω (cf. [17, III.5.4]). There are nonrecursive semirecursive sets.

Proposition 3.3 *If A is semirecursive then $e(f, A) \leq s(f)$.*

Proof: Let $<_A$ be a recursive linear ordering of ω having A as an initial segment. We first find the permutation π such that $x_{\pi(1)} <_A x_{\pi(2)} <_A \ldots <_A x_{\pi(n)}$. The only possible values for $C_n^A(x_1, \ldots, x_n)$ are the v_i, $0 \leq i \leq n$, where the coordinates $\pi(1), \ldots, \pi(i)$ of v_i are one and the others $(\pi(i+1), \ldots, \pi(n))$ are zero. Hence there are at most $s(f)$ possible values of f. In fact, we even have $e'(F^A) \leq s(f)$. ∎

Theorem 3.4 $e_{\min}(f) = s(f)$.

Proof: It remains to prove: $e(f, A) < s(f) \Rightarrow A$ is recursive. We proceed as in [14] using its definitions and notations. Assume that $e(f, A) < s(f)$, i.e. there is a recursive function $g(x_1, \ldots, x_n)$ such that, for any n-tuple of natural numbers, (1) $|W_{g(x_1,\ldots,x_n)}| < s(f)$, and (2) $F^A(x_1, \ldots, x_n) \in W_{g(x_1,\ldots,x_n)}$. We define an r.e. tree T of possibilities as follows:

$$\begin{aligned} T &= \{t \in \{0,1\}^{<\omega} : \\ &\qquad (\forall x_1, \ldots, x_n < |t|)[f(t(x_1), \ldots, t(x_n)) \in W_{g(x_1,\ldots,x_n)}]\}. \end{aligned}$$

Note that, by (2), χ_A is a branch of T. If the binary tree of depth $4^n - 2$ is not embeddable into T then, by [14, Lemma 1], it follows that A is recursive. Suppose for a contradiction that this tree is embeddable. Then, by [14, Lemma 3], there exist nodes $t_0, \ldots, t_n$ of T and numbers $x_1 < \cdots < x_n$, and $b \in \{0, 1\}$ such that for $j = 0, \ldots, n$ and $i = 1, \ldots, n$:

$$t_j(x_i) = 1 - b \text{ for } i \leq j \text{ and } t_j(x_i) = b \text{ for } i > j.$$

Assume that $b = 0$, the case $b = 1$ is similar. Then, for every sequence $(v_0, \ldots, v_n) \in L_n$ there is a permutation π of $\{1, \ldots, n\}$ such that: $v_i = (t_i(x_{\pi(1)}), \ldots, t_i(x_{\pi(n)}))$, for $i = 0, \ldots, n$. In particular, if we choose a sequence that realizes $s(f)$, it follows that $|W_{g(x_{\pi(1)},\ldots,x_{\pi(n)})}| \geq s(f)$, which contradicts (1). ∎

For example, note that $s(f) = n+1$ for $f = id_n$ and $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$. So Fact 1.2 follows from Theorem 3.4.
From Theorems 2.2, 3.4 we obtain at once:

Corollary 3.5 *If $r(F^A, l) < u(s(f), l)$ then A is recursive.*

Remarks: a) Note that the lower bound even holds for the restriction of F^A to pairwise different $x_1, \ldots, x_n$.
b) The results of this section can be proved in a slightly more general context. Consider any finite relation $r \subseteq \{0,1\}^n \times S$ such that for every $u \in \{0,1\}^n$ there is $a \in S$ with $(u,a) \in r$. $F : \omega^n \to S$ is an *r-selector for A* iff $(\chi_A(x_1), \ldots, \chi_A(x_n), F(x_1, \ldots, x_n)) \in r$ for all $x_1, \ldots, x_n$. Let $\overline{e}(r, A) = \min\{e(F) : F \text{ is an } r\text{-selector for } A\}$.

Here we have the following extremal values:

$$\begin{aligned} \overline{e}_{\min}(r) &:= \min\{\overline{e}(r,A) : A \text{ is nonrecursive}\} \text{ and} \\ \overline{e}_{\max}(r) &:= \max\{\overline{e}(r,A) : A \subseteq \omega\}. \end{aligned}$$

It is easy to check that $\overline{e}_{\max}(r) = \min\{|S'| : S' \subseteq S \ \wedge \ (\forall u \in \{0,1\}^n) (\exists a \in S')[(u,a) \in r]\}$ and this value is attained for any superterse set. For a semirecursive set A we obtain $\overline{e}(r,A) \leq \overline{s}(r)$ where

$$\overline{s}(r) := \max_{(v_0, \ldots, v_n) \in L_n} \min\{|S'| : S' \subseteq S \ \wedge \ (\forall i \leq n)(\exists a \in S')[(v_i, a) \in r]\}.$$

Using a similar argument as in the proof of Theorem 3.4 it follows that $\overline{e}_{\min}(r) = \overline{s}(r)$. Again the lower bound holds even if r-selectors are restricted to pairwise different inputs.

4 Examples

4.1 Graph Problems

Let $n = \frac{m(m-1)}{2}$. Identify the edges of the complete graph on m vertices in a fixed way with the numbers $1, \ldots, n$. For any $(b_1, \ldots, b_n) \in \{0,1\}^n$ let $G(b_1, \ldots, b_n)$ denote the subgraph in which edge i is present iff $b_i = 1$.
In the following cases we have $e_{\min}(f) = s(f) = m = e_{\max}(f)$ and every sequence in L_n realizes $s(f)$.
a) $f(v) := \omega(G(v)) = \max\{k : G(v) \text{ contains a } k\text{-clique}\}$.
b) $f(v) := \chi(G(v))$, the chromatic number of $G(v)$.
c) $f(v) :=$ the number of connected components in $G(v)$.

These results are loosely related to the *recognition complexity* of graph properties (see [1, Section 5.6]) where one has to determine whether an unknown graph G on m vertices has a certain property by asking questions of the form "Is edge e in G?". There is a version of query complexity which matches recognition complexity more closely: Consider the following *local* form of query complexity. To compute $(f \circ C_n^A)(x_1, \ldots, x_n)$ only queries of the form "Is x_i in A?" are allowed. If A is semirecursive then $\lceil \log_2(s(f)) \rceil$ local queries suffice (using binary search). Let $p(f)$ denote the minimal number of probes it takes to evaluate f on an unknown input $(b_1, \ldots, b_n) \in \{0,1\}^n$. Clearly $p(f)$ local queries suffice to compute $(f \circ C_n^A)(x_1, \ldots, x_n)$. A straightforward diagonalization argument shows that there exists A for which less than $p(f)$ local queries do not suffice. In the case of graph properties $p(f)$ is just the recognition complexity. For example, let $f(v) := 1$ if $G(v)$ is connected, and $f(v) := 0$ otherwise; then $p(f) = \frac{m(m-1)}{2} = n$, see [1, p. 180].

4.2 k Versus $k-1$ Queries

Let $n = 2^k - 1$. Let $b_1, \ldots, b_n$ be the inner nodes of the (ordered) full binary tree of depth k. Identify the leaves with the numbers $1, \ldots, 2^k$. For any $v \in \{0,1\}^k$ we define a path through the tree as follows: Start at the root. When at the inner node b_i go to the left successor if $v[i] = 0$ and go to the right successor if $v[i] = 1$. Let $f(v)$ denote the index of the leaf at which the path ends. For later reference let $TREE_k^A$ be $f \circ C_n^A$.

It is easy to see that $s(f) = 2^k$: Traverse the tree in depth first search and flip the i-th bit from 0 to 1 when node b_i is visited for the second time ("inorder traversal"). Hence $e(TREE_k^A) = 2^k$ and $q(TREE_k^A) = k$ for A nonrecursive. Note that $TREE_k^A$ can be computed with k queries to A. Thus, for any nonrecursive set A there is a function that can be computed with k queries to A but it cannot be computed with less than k queries regardless of the oracle. There is no corresponding result for parallel queries: If A is not superterse then there is an oracle X such that for some k, k parallel queries to A can be computed by $k-1$ parallel queries to X (see [6, Theorem 20]).

In some applications of the Nonspeedup Theorem one obtains a stronger result if one uses $TREE_k^A$ instead of C_k^A. See for instance [10, Section 4.1, Note 6.34].

4.3 Frequency Computation

A set A is called (m,n)-recursive iff there is a recursive function $f : \omega^n \rightarrow \{0,1\}^n$ such that for all pairwise different $x_1, \ldots, x_n$ at least m of the n equalities $\chi_A(x_i) = f(x_1, \ldots, x_n)[i]$ hold $(1 \leq i \leq n)$. Trakhtenbrot [22] (see also [12]) proved that every (m,n)-recursive set is recursive if $m > \frac{n}{2}$. This can be represented and generalized in our framework. Define $r_{m,n} \subseteq \{0,1\}^n \times \{0,1\}^n$ such that $(v,w) \in r_{m,n}$ iff $v, w \in \{0,1\}^n$ and they agree in at least m components, i.e. $|\{i : v[i] = w[i]\}| \geq m$. For $\overline{s}$ (defined at the end of section 3) we can show that $\overline{s}(r_{m,n}) = \lceil \frac{n+1}{2(n-m)+1} \rceil$, $1 \leq m \leq n$. (This can be also proved by a reduction from the Cardinality Theorem, and has been independently obtained in [7].) As we remarked above the lower bound already holds for pairwise different inputs. Hence we obtain Trakhtenbrot's result as a special case, since $\lceil \frac{n+1}{2(n-m)+1} \rceil > 1$ for $m > \frac{n}{2}$. The Nonspeedup Theorem follows if we let $m = n$.

4.4 Operations on K

For the halting problem K we can characterize the various enumeration complexities by finite combinatorial properties. In the following we just state the results and omit the proofs. The upper bounds are straightforward using only that K is r.e., the lower bounds are obtained by using the recursion theorem (e.g., see [7] or [11, Theorem 23]). For functions $f : \{0,1\}^n \rightarrow S$ we get $e(f,K) = s(f)$ as in the semirecursive case, on the other hand, $e'(F^K) = e_{\max}(f)$ in contrast to the semirecursive case. For total relations $r \subseteq \{0,1\}^n \times S$ we get $\overline{e}(r,K) = g(r)$ where $g(r)$ is the least number k such that player I has a winning strategy in the following game:

At the beginning $v_0 = 0^n$. In round i $(0 \leq i \leq n)$ player I chooses $a_i \in S$ such that $(v_i, a_i) \in r$. Then, if $i < n$, player II flips one component of v_i from 0 to 1, which defines v_{i+1}. After round n player I wins iff $|\{a_i : 0 \leq i \leq n\}| \leq k$.

Note that the result $\overline{e}_{\min}(r) = \overline{s}(r)$ corresponds to the variant of the game where player II makes all of his moves in advance, and on the basis of this information, player I chooses the a_i's. The easiest case where $\overline{s}(r) < g(r)$ is the "diamond": $r = \{(00,a),(00,b),(01,a),(10,b),(11,c)\}$ with $\overline{s}(r) = 2$ and $g(r) = 3$. For $r_{m,n}$ from the previous subsection one obtains $g(r_{m,n}) = \lfloor \frac{n}{n-m+1} \rfloor + 1 = \lceil \frac{n+1}{(n-m)+1} \rceil$ which is approximately a factor 2 above $\overline{s}(r_{m,n})$. (In [7] $\overline{e}(r_{m,n}, K)$ was determined by a direct argument and a reduction from the Cardinality Theorem.)

5 Query Complexity of Operations on Functions

5.1 Functions with Finite Range

Let $g : \{0, \dots, p\}^n \to S$ denote a fixed finite function with $p \geq 1$. For any function $F : \omega \to \{0, \dots, p\}$ we define the operation $G^F(x_1, \dots, x_n) := g(F(x_1), \dots, F(x_n))$. In this context we get the following extremal values for g:

$$
\begin{aligned}
e_{\min}(g) &:= \min\{e(G^F) \mid F : \omega \to \{0, \dots, p\} \ \wedge \ F \text{ is nonrecursive}\} \text{ and} \\
e_{\max}(g) &:= \max\{e(G^F) \mid F : \omega \to \{0, \dots, p\}\}.
\end{aligned}
$$

The case $p = 1$ has been treated in Section 3. By a similar diagonalization argument we get $e_{\max}(g) = |\text{range}(g)|$. It turns out that $e_{\min}(g)$ can be characterized in terms of certain subfunctions as we now explain.
We call $f : \{0, 1\}^n \to S$ a 2-*subfunction* of g iff there exist two different elements $a, b \in \{0, \dots, p\}$ such that for all $u_1, \dots, u_n \in \{0, 1\}$: $f(u_1, \dots, u_n) = g(h(u_1), \dots, h(u_n))$ where $h(0) = a$ and $h(1) = b$. Let $s_{\min}(g) = \min\{s(f) : f \text{ is a 2-subfunction of } g\}$. Note that for $p = 1$ we get $s_{\min}(g) = s(g)$.

Theorem 5.1 $e_{\min}(g) = s_{\min}(g)$.

Proof: The upper bound follows directly from Theorem 3.4. For the lower bound assume that $F : \omega \to \{0, \dots, p\}$ satisfies $e(G^F) < s_{\min}(g)$. Then there is a recursive function $h(x_1, \dots, x_n)$ such that for any n-tuple of natural numbers, (1) $|W_{h(x_1,\dots,x_n)}| < s_{\min}(g)$ and (2) $G^F(x_1, \dots, x_n) \in W_{h(x_1,\dots,x_n)}$. We need to show that F is recursive and proceed as in [14] with the difference that here we consider $(p+1)$-ary trees instead of binary trees. Define an r.e. tree T of possibilities as follows:

$$
\begin{aligned}
T \ = \ \{&t \in \{0, \dots, p\}^{<\omega} : \\
&(\forall x_1, \dots, x_n < |t|)[g(t(x_1), \dots, t(x_n)) \in W_{h(x_1,\dots,x_n)}]\}.
\end{aligned}
$$

Note that F is a branch of T. By [14, Lemma 1], F is recursive if we can show that some finite binary tree is not embeddable into T. We need the following straightforward generalizations of Lemmas 2, 3 in [14].

Claim 1: For any $(p + 1)$-coloring of $B_{(p+1)k}$ there is a monochromatic embedding of B_k into $B_{(p+1)k}$. (B_k denotes the full binary tree of depth k.)

Claim 2: For each $n \geq 1$, any tree $T' \subseteq \{0, \ldots, p\}^{<\omega}$, and any sufficiently large k such that B_k is embeddable into T there exist nodes $t_0, \ldots, t_n$ of T and numbers $x_1 < \cdots < x_n$, and $a, b \in \{0, \ldots, p\}$, $a \neq b$ such that for $j = 0, \ldots, n$ and $i = 1, \ldots, n$:

$$(+) \quad t_j(x_i) = a \text{ for } i \leq j \text{ and } t_j(x_i) = b \text{ for } i > j.$$

For the proof of Claim 2 we iterate the process from the proof of [14, Lemma 3] for $r = p(p+1)(n-1)+1$ steps. In each step we use Claim 1 instead of [14, Lemma 2]. We obtain nodes $s_0, \ldots, s_r$ and numbers $z_1 < \cdots < z_r$ such that $s_i(z_j) = s_{i-1}(z_j)$ for $1 \leq j < i$, $s_i(z_i) \neq s_{i-1}(z_i)$, and $s_i(z_j) = s_i(z_{i+1})$ for $i < j \leq r$. By the pigeonhole principle there exist different numbers $a, b \in \{0, \ldots, p\}$ and n indices $1 \leq u_1 < \cdots < u_n \leq r$ such that $s_{u_i}(z_{u_i}) = a \wedge s_{u_i - 1}(z_{u_i}) = b$ for $i = 1, \ldots, n$. By the properties of s_i, z_i it follows that $(+)$ holds if we define $x_i = z_{u_i}, t_{i-1} = s_{u_i - 1}$ for $i = 1, \ldots, n$ and $t_n = s_{u_n}$. This completes the proof sketch of Claim 2.

Suppose for a contradiction that every finite binary tree is embeddable into T. Then we find nodes and numbers as in $(+)$. Let f be the 2-subfunction of g induced by a, b. As in the proof of Theorem 3.4 there is a permutation π of $\{1, \ldots, n\}$ such that $|W_{g(x_{\pi(1)}, \ldots, x_{\pi(n)})}| \geq s(f)$ contradicting (1). Hence some finite binary tree is not embeddable into T, which implies that F is recursive. ▮

5.2 Functions with Unbounded Range

In this section we consider functions $g : \omega^n \to \omega$. For any $F : \omega \to \omega$ we define G^F as in the previous subsection. However, now we cannot hope to get a general explicit characterization of the extremal values. The following result is an interesting special case which leads to a generalization of the Cardinality Theorem to functions.

Theorem 5.2 *Let $g(x_1, \ldots, x_n)$ be a total recursive function which is strictly increasing in each argument. Then $e_{\min}(g) = n + 1$.*

Proof: The upper bound holds if we let F be the characteristic function of any nonrecursive semirecursive set. For the lower bound we use a modification of the proof of Theorem 5.1. Assume that $e(G^F) \leq n$ for some $F : \omega \to \omega$. Then there is a recursive function $h(x_1, \ldots, x_n)$

such that for any n-tuple of natural numbers, (1) $|W_{h(x_1,\dots,x_n)}| \leq n$ and (2) $G^F(x_1,\dots,x_n) \in W_{h(x_1,\dots,x_n)}$. This time we define an r.e. tree $T \subseteq \omega^{<\omega}$:

$$T = \{t \in \omega^{<\omega} : (\forall x_1,\dots,x_n < |t|)[g(t(x_1),\dots,t(x_n)) \in W_{h(x_1,\dots,x_n)}]\}.$$

Note that F is a branch of T. By (1) it follows that for any x there are at most n different values of the form $t(x)$ with $t \in T$. Let $r_0(x) < r_1(x) < \cdots < r_{l(n)}(x)$ be a listing of these values in order of magnitude ($l(n) < n$). If we map each node t to t' with $t'(x) = m \Leftrightarrow t(x) = r_m(x)$ then we obtain a tree $T' \subseteq \{0,\dots,n-1\}$. Again it suffices to show that some finite binary tree is not embeddable into T to conclude that F is recursive. Suppose for a contradiction that every finite binary tree is embeddable into T. Then the same holds for T' and we obtain from Claim 2 of the previous proof (with $p = n-1$) nodes $t'_0,\dots,t'_n$ of T' and numbers $x_1 < \cdots < x_n$ and $a,b \in \{0,\dots,n-1\}$, $a \neq b$ such that for $j = 0,\dots,n$ and $i = 1,\dots,n$:

$$(+)\quad t'_j(x_i) = a \text{ for } i \leq j \text{ and } t'_j(x_i) = b \text{ for } i > j.$$

Let $t_0,\dots,t_n$ be preimages of $t'_0,\dots,t'_n$, respectively. Assume that $a > b$, the other case is symmetric. Then we get for $j = 0,\dots,n-1$:

$$g(t_j(x_1),\dots,t_j(x_n)) < g(t_{j+1}(x_1),\dots,t_{j+1}(x_n))$$

because $t_j(x_i) = t_{j+1}(x_i)$ for $i \neq j+1$ and $t_j(x_{j+1}) = r_b(x_{j+1}) < r_a(x_{j+1}) = t_{j+1}(x_{j+1})$ and g is strictly increasing in each argument. Thus, $|W_{h(x_1,\dots,x_n)}| \geq n+1$, which contradicts (1). Hence some finite binary tree is not embeddable into T, which implies that F is recursive. ∎

By a slight modification of the proof (namely by hard-wiring in the values $F(0), F(1),\dots,F(n-2)$) the lower bound holds even if G^F is restricted to arguments $x_1 < \cdots < x_n$. Hence we obtain the following result which was announced in [12, p. 687].

Corollary 5.3 *("Sum Theorem") If $F : \omega \to \omega$ and $h(x_1,\dots,x_n)$ is a recursive function such that, for all $x_1 < \cdots < x_n$,*

$$F(x_1) + \cdots + F(x_n) \in W_{h(x_1,\dots,x_n)} \ \wedge\ |W_{h(x_1,\dots,x_n)}| \leq n,$$

then F is recursive.

Remark: There is another special case where we have an explict result, though it is rather trivial. If range(g) is finite it follows that $e_{\min}(g) = 1$: By Ramsey's Theorem there is an infinite subset E of ω such that for $x_1,\dots,x_n \in E$ the value of $g(x_1,\dots,x_n)$ depends only on the relative order of $x_1,\dots,x_n$. Take any nonrecursive subset of E and let $F : \omega \to \omega$ enumerate E in order of magnitude. Then $e(G^F) = 1$ and F is nonrecursive.

6 A Methodological Remark

Let $e'_A(F)$ denote the least number m such that $F : \omega \to \omega$ is computable by a set of m total A-recursive functions. Hence $e'(F) = e'_\emptyset(F)$. Suppose we want to show a lower bound n for $e(F)$ where F is nonrecursive. Of course, in general it is not sufficient to prove this bound for $e'(F)$. (For instance, $e'(C_n^K) = 2^n$ but $e(C_n^K) = n+1$, see Subsection 4.4 or [11, Theorem 23].) However, the following result gives a condition under which a lower bound transfers from $e'(F)$ to $e(F)$. Informally, if we have a relativizable proof that $e'(F) > n$ and F is known to be recursively bounded then $e(F) > n$.

Theorem 6.1 *If $(\forall A)[e'_A(F) \leq n \Rightarrow\ F \leq_T A]$ and F is recursively bounded and nonrecursive then $e(F) > n$.*

Proof: Suppose for a contradiction, that the hypothesis of the theorem holds and $e(F) \leq n$. Choose a recursive function h with $F(x) \leq h(x)$ for all x. Let $\psi_1, \ldots, \psi_n$ be n partial recursive function such that F is computable by $\{\psi_1, \ldots, \psi_n\}$. We may assume that $\psi_i(x)\downarrow\ \Rightarrow\ \psi_i(x) \leq h(x)$ for all i, x. By a well-known result of Jockusch and Soare ([13, Theorem 2.4], [20, VI.5.15]) there is a set A of hyperimmune-free degree such that there exist total A-recursive extensions $\overline{\psi_1}, \ldots, \overline{\psi_n}$ of $\psi_1, \ldots, \psi_n$, respectively. In other words, $e'_A(F) \leq n$. Thus, by the hypothesis, we get $F \leq_T A$. Let $g(x)$ be the least s such that $F(x) = \psi_{i,s}(x)$ for some i where as usual $\psi_{i,s}(x)$ is the value of $\psi_i(x)$ after s steps of computation. Since F is recursive in A, g is recursive in A, too. As A is hyperimmune-free there exists a total recursive function $\overline{g}$ that majorizes g. Define $\psi'_i(x) = \psi_i(x)$ if $\psi_{i,\overline{g}(x)}(x)$ is defined, otherwise let $\psi'_i(x) = 0$. Then F is computed by the set of n total recursive functions $\{\psi'_1, \ldots, \psi'_n\}$, i.e. $e'_\emptyset(F) \leq n$. By the hypothesis, this implies that F is recursive, a contradiction. ▮

Theorem 6.1 together with Theorem 1 in [18] gives an alternative proof of the Cardinality Theorem (CT). So the original approach of Owings towards a proof of CT can in fact be salvaged. This can be seen as a positive answer to the question of whether CT can be proven from the co-r.e. tree lemma (cf. [12, p. 686]). Interestingly, both proofs of CT are completely different. However, the proof on the lines of Owings looks quite fortuitous and is less powerful than the methods used in [14] and in the present paper. Nevertheless Theorem 6.1 may have other interesting applications.

7 Conclusion and Open Problems

We have shown that the notion of enumeration complexity allows to characterize the query complexity, even in the fault tolerant case. Then we went on to determine for any finite function $f : \{0,1\}^n \to S$ the extremal values of $e(f, A)$ which can be obtained for some nonrecursive set A. It is open to determine the possible intermediate values. More formally, there is the following open problem. For any $k \in \omega$ let $V(f, k) = \{A : e(f, A) \leq k\}$. The "general inclusion problem" (GIP) is the question for which $f : \{0,1\}^n \to S_1$, $g : \{0,1\}^m \to S_2$ and k_1, k_2 we have $V(f, k_1) \subseteq V(g, k_2)$. In particular, this would answer the question for which k_1, k_2, the classes $V(f, k_1)$ and $V(f, k_2)$ coincide. By now we know that for $s(f) > 1$: $V(f, 1) = V(f, k)$ iff $1 \leq k < s(f)$. For the interesting special case $f = id_n$, $g = id_m$ the inclusion problem has been completely solved in [8]. However, it is an open question whether GIP is decidable, since the solution of that special case does not generalize. This stands in contrast to the analogous inclusion problem in the context of polynomial time bounded computation (i.e. the version indicated in Remark (a) following Theorem 2.2) which has a positive answer: a combinatorial condition can be formulated similarly as for the special case treated in [8, Section 7].

If we consider instead of $V(f, k)$ the classes $R(r, k) = \{A : e(r, A) \leq k\}$ for total relations $r \subseteq \{0,1\}^n \times S$ an even more general inclusion problem arises. Here already the special cases when r is of the form $r_{m,n}$ is open (partial results can be found in [9, 15]).

Acknowledgement: We would like to thank Bill Gasarch for proof reading, comments, and suggestions which led to the present form of Sections 4.1 and 4.4. Thanks also to Richard Beigel and Jim Owings for comments.

References

[1] M. Aigner. *Combinatorial search.* Wiley-Teubner Series in Computer Science. J. Wiley & Sons Ltd, and B. G. Teubner, Stuttgart, 1988.

[2] R. Beigel. Query-limited reducibilities. Ph.D. thesis, Stanford University, Stanford, USA, 1987.

[3] R. Beigel. Unbounded searching algorithms. *SIAM J. Comput.*, 19:522–537, 1990.

[4] R. Beigel, W. I. Gasarch. On the complexity of finding the chromatic number of a recursive graph I: The bounded case. *Annals of Pure and Applied Logic*, 45:1–38, 1989.

[5] R. Beigel, W. I. Gasarch. On the complexity of finding the chromatic number of a recursive graph II: The unbounded case. *Annals of Pure and Applied Logic*, 45:227–246, 1989.

[6] R. Beigel, W. I. Gasarch, J. Gill, J. C. Owings, Jr. Terse, superterse, and verbose sets. *Information and Computation*, 103:68–85, 1993.

[7] R. Beigel, W. I. Gasarch, E. Kinber. Frequency computation and bounded queries. Manuscript, 1993.

[8] R. Beigel, M. Kummer, F. Stephan. Quantifying the amount of verboseness. To appear in: Information and Computation. (A preliminary version appeared in: *Logical Foundations of Computer Science–Tver'92*, (Eds. A. Nerode, M. Taitslin), Lecture Notes in Computer Science Vol.620, pp. 21–32, Springer-Verlag, Berlin, 1992.)

[9] A. N. Dëgtev. On (m, n)-computable sets. In *Algebraic Systems (Edited by D.I. Moldavanskij).* Ivanova Gos. Univ. 88–99, 1981. (Russian) (MR 86b:03049)

[10] L. Fortnow, W. Gasarch, S. Jain, E. Kinber, M. Kummer, S. Kurtz, M. Pleszkoch, T. Slaman, R. Solovay, F. Stephan. Extremes in the degrees of inferability. To appear in: Annals of Pure and Applied Logic.

[11] W. I. Gasarch. Bounded queries in recursion theory: a survey. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference.* IEEE Computer Society Press, pp. 62–78, 1991.

[12] V. Harizanov, M. Kummer, J. C. Owings, Jr. Frequency computation and the cardinality theorem. *J. Symb. Log.*, 57:682–687, 1992.

[13] C. G. Jockusch, Jr., R. I. Soare. Π_1^0 classes and degrees of theories. *Trans. Amer. Math. Soc.*, 173:33-56, 1972.

[14] M. Kummer. A proof of Beigel's cardinality conjecture. *J. Symb. Log.*, 57:677–681, 1992.

[15] M. Kummer, F. Stephan. Some aspects of frequency computation. Technical Report Nr. 21/91, Fakultät für Informatik, Universität Karlsruhe, 1991.

[16] D. Mundici. The complexity of adaptive error-correcting codes. In *Computer Science Logic. Proceedings 1990.* E. Börger, et.al. (Eds.). Lecture Notes in Computer Science, Vol. 533, pp. 300-307, Springer-Verlag, Berlin, 1991.

[17] P. Odifreddi. *Classical recursion theory.* North-Holland, Amsterdam, 1989.

[18] J. C. Owings, Jr. A cardinality version of Beigel's nonspeedup theorem. *J. Symb. Log.*, 54:761–767, 1989.

[19] R. L. Rivest, A. R. Meyer, D. J. Kleitman, and K. Winkelmann. Coping with errors in binary search procedures. *Journal of Computer and System Sciences*, 20:396–404, 1980.

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

[21] J. Spencer. Ulam's searching game with a fixed number of lies. *Theoretical Computer Science*, 95:307–321, 1992.

[22] B. A. Trakhtenbrot. On frequency computation of functions. *Algebra i Logika*, 2:25–32, 1963. (Russian)