An Exposition of the Main Theorem in
Enumerations of the Kolmogorov Function

Authors of paper:
Beigel, Buhrman, Fejer, Fortnow Grabowski, Longpre, Muchnik, Stephan, Torenvliet
Author of this writeup: Gasarch

1 Introduction and Definitions

The following definition is basic to Kolmogorov complexity (see [**?**]).

Def 1.1 Let x be a string of length n.

1. $C(x)$ is the size of the smallest program that outputs x. This is the Kolmogorov complexity of x. (Note- to formalize this we would need so specify what a program is; however, the Kolmogorov complexity of a string changes by only a constant when you change programming systems.)

2. We define $C_s(x)$ to be an approximation to C after s steps. Formally we define $C_0(x) = n + O(1)$ since without any work you know there is a program that stores x and prints it. (The $O(1)$ depends on the particular programming system.) $C_s(x)$ is obtained by running the first s Turing machines for s steps on 0; if any of them prints x and has size $\le C_{s-1}(x)$ then output the size of the smallest such machine.

Intuitively a function f is m-enumerable if there is a process that, on input x, enumerates $\le m$ candidates for $f(x)$ one of which really is $f(x)$. We formalize this.

Notation 1.2 W_e is the domain of the eth Turing machine, so $W_0, W_1, \ldots$ is a list of all c.e. sets. W_e^A is the domain of the eth oracle Turing machine using oracle A, so $W_0^A, W_1^A, \ldots$ is a list of all c.e.-in-A sets.

Def 1.3 [1, 2] Let $m \ge 1$ and let $A \subseteq \mathsf{N}$.

1. f is *m-enumerable* if there is a computable function h such that
$$(\forall x)[|W_{h(x)}| \le m \wedge f(x) \in W_{h(x)}].$$

2. f is *m-enumerable-in-A* if there is a computable function h such that $(\forall x)[|W_{h(x)}^A| \le m \wedge f(x) \in W_{h(x)}^A]$.

3. $\mathrm{EN}^A(m)$ is the class of all m-enumerable-in-A functions.

We need the following definition and theorem from computability theory.

Def 1.4 Let f be a partial function and F be a total function. f is *dominated by* F if, for every x such that $f(x)$ exists, $f(x) < F(x)$. f is *computably dominated* if there is a computable function F such that f is dominated by F.

Def 1.5 [3] A set X is *extensive* if, for every computably dominated partial computable function f, there is a total function $g \le_{\mathrm{T}} X$ such that g extends f.

Lemma 1.6 *[3] Let A be a set. There exists a set X such that the following hold.*

1. $A \le_{\mathrm{T}} X$.
2. $K \le_{\mathrm{T}} X \to K \le_{\mathrm{T}} A$.
3. *X is extensive.*

We need the following definition and theorem from bounded queries.

Def 1.7 Let $k \in \mathsf{N}$ and $D \subseteq \mathsf{N}$. Then $\#_k^D(x_1, \ldots, x_k) = |D \cap \{x_1, \ldots, x_k\}|$.

Lemma 1.8 *[1, 2] Let $k \in \mathsf{N}$. If $\#_k^K \in \mathrm{EN}^A(k)$ then $K \le_{\mathrm{T}} A$.*

Note 1.9 Kummer showed [4] that, for all D, $\#_k^D \in \mathrm{EN}^A(k)$ then $D \le_{\mathrm{T}} A$.

We need the following easy lemma and corollary from kolgmorov theory. They are both folklore; we include their proofs for completeness.

Lemma 1.10 *Let $a, b \in \mathsf{N}$ such that $a + 1 \le b$. Let G be a set of at least 2^b strings. Then there exists at least 2^a strings $w \in G$ such that $C(w) \ge a$.*

Proof: Assume, by way of contradiction, that

$|\{w \in G : C(w) \geq a\}| < 2^a$.

Note that

$|\{w \in G : C(w) < a\}| \leq |\{w : C(w) < a\}| \leq 2^0 + 2^1 + \cdots + 2^{a-1} = 2^a - 1$.

Hence

$$2^b \leq |G| = |\{w \in G : C(w) < a\}| + |\{w \in G : C(w) \geq a\}| \leq 2^a - 1 + 2^a < 2^{a+1}.$$

This implies $b < a+1$ which contradicts the hypothesis that $a+1 \leq b$.

■

Corollary 1.11 *Let $i, m \in \mathsf{N}$. If G is a set of $2^{m-(i-1)\lceil\sqrt{m}\rceil}$ strings then there exists at least $2^{m-i\lceil\sqrt{m}\rceil+\lceil m^{1/3}\rceil}$ strings $w \in G$ such that $C(w) \geq m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$.*

Proof: Apply Lemma 1.10 with $a = m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$ and $b = m - (i-1)\lceil\sqrt{m}\rceil$. ■

2 An Easy Theorem about C

Theorem 2.1 *$C \leq_{\mathrm{tt}} K$ and $K \leq_{\mathrm{T}} C$.*

Proof:

1) $C \leq_{\mathrm{tt}} K$. Given x we can compute $C(x)$ as follows. For all machines M of length $\leq |x| + O(1)$ ask K "does $M(0)$ halt and output x?" Once you get the answers, output the length of the shortest such M for which the answer was YES.

2) $K \leq_{\mathrm{T}} C$. We need to look at the partial computable function f below:
f: On input x find s such that $x \in K_s - K_{s-1}$ (this might not happen). Let $|x| = n$ and $m = 2^n$. Find $C_s(z)$ for every z of length m. Output the z with the largest C_s-value (break ties lexicographically). Note the following:

If $x \in K$, $z = f(x)$, and s is such that $z \in K_s - K_{s-1}$ then the following hold.

$C_s(z) \geq |z| = m + O(1)$ (since $(\exists z', |z'| = m)[C(z') \geq m + O(1)]$).

$C(z) \leq \log m + O(1)$ (since z can be computed from the code for f and the input x, $|x| = n = \log m$).

Here is the key: If $x \in K_s - K_{s-1}$ then there exists a string $z = f(x)$ of length m such that $C_s(z) > C(z)$. Hence, if s is such that $(\forall z)[|z| = m \rightarrow C_s(z) = C(z)]$ then $x \in K$ iff $x \in K_s$. Using this we have the following algorithm for $K \leq_{\mathrm{T}} C$.

$K \leq_{\mathrm{T}} C$: on input x, let $|x| = n$ and $m = 2^n$. Find $C(z)$ for all $z \in \{0,1\}^m$. Find s such that, for all $z \in \{0,1\}^m$, $C_s(z) = C(z)$. If $x \in K_s$ then output YES, otherwise output NO. ∎

Note 2.2 Kummer has shown that $K \leq_{\mathrm{tt}} C$ [5].

3 Main Theorem

Theorem 3.1 *Let $k \in \mathsf{N}$. If $C \in \mathrm{EN}^A(k)$ then $K \leq_{\mathrm{T}} A$.*

Proof:

Let $C \in \mathrm{EN}^A(k)$ via h. Note that h is computable. We will not use h until later.

By Lemma 1.6 there exists a set X such that $A \leq_{\mathrm{T}} X$, $K \leq_{\mathrm{T}} X \rightarrow K \leq_{\mathrm{T}} A$, and X is extensive (Definition 1.5). We show that $\#_k^K \in \mathrm{EN}^X(()k)$, hence by Lemma 1.8, $K \leq_{\mathrm{T}} X$; so $K \leq_{\mathrm{T}} A$.

We need to define $k+1$ partial computable functions on ordered k-tuple $(x_1, \ldots, x_k)$. We assume throughout that $\sum_{i=1}^k |x_i| = n$ and that $m = 2^n$.

$f_0(x_1, \ldots, x_k) = \{0,1\}^m$.

For $1 \leq i \leq k$, $f_i(x_1, \ldots, x_k)$ is defined as follows: find the least s such that $\#_k^{K_s}(x_1, \ldots, x_k) = i$ (this might not ever happen). Compute $C_s(z)$ for every $z \in f_{i-1}(x_1, \ldots, x_k)$. Order the strings by largest to smallest value of C_s (break ties via lexicographic ordering). Output the highest ranked $2^{m-i\lceil\sqrt{m}\rceil}$ strings.

Clearly $f_0, \ldots, f_k$ are partial computable functions that are computably dominated. Hence, for each i, $0 \leq i \leq k$, there exists total $g_i \leq_{\mathrm{T}} X$ such that g_i extends f_i. We may assume that, for all $(x_1, \ldots, x_k)$, for all i, $g_i(x_1, \ldots, x_k)$ is a set of size $2^{m-i\lceil\sqrt{m}\rceil}$. In particular, it is not empty.

Claim 0: Let $(x_1, \ldots, x_k) \in \mathsf{N}$. If there exists i, $1 \leq i \leq k$, such that $g_i(x_1, \ldots, x_k) \not\subseteq g_{i-1}(x_1, \ldots, x_k)$ then $\#_k^K(x_1, \ldots, x_k) \neq k$.

Proof: We prove the contrapositive. If $\#_k^K(x_1, \ldots, x_k) = k$ then, for i, $0 \leq i \leq k$, $f_i(x_1, \ldots, x_k) = g_i(x_1, \ldots, x_k)$. Hence, for all i, $1 \leq i \leq k$, $g_i(x_1, \ldots, x_k) \subseteq g_{i-1}(x_1, \ldots, x_k)$. ∎

Claim 1: Let $n \in \mathsf{N}$. Let $x_1, \ldots x_k \in \mathsf{N}$ be such that $\sum_{i=1}^{k} |x_i| = n$. Let $m = 2^n$. We assume that for all i, $1 \le i \le k$, $g_i(x_1, \ldots, x_k) \subseteq g_{i-1}(x_1, \ldots, x_k)$. For $1 \le i \le k$ define

$$s_i = \begin{cases} \text{the least } s \text{ such that } \#_k^{K_s}(x_1, \ldots, x_k) = i & \text{if } \#_k^{K}(x_1, \ldots, x_k) \ge i; \\ \infty & \text{otherwise.} \end{cases}$$

For all i, $1 \le i \le k$, if $s_i < \infty$ then

1. $(\forall z \in g_i(x_1, \ldots, x_k))[C_{s_i}(z) \ge m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil]$, and
2. $(\forall z \in g_i(x_1, \ldots, x_k))[C(z) \le m - i\lceil\sqrt{m}\rceil + 2\log m + O(1)]$.

Proof: Let i be such that $s_i < \infty$. Note that, for all $1 \le j \le i$, $f_j(x_1, \ldots, x_k)$ exists, so $g_j(x_1, \ldots, x_k) = f_j(x_1, \ldots, x_k)$. Let $z \in g_i(x_1, \ldots, x_k)$.

(1) We show that $C_{s_i}(z) \ge m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$. Since $|g_{i-1}(x_1, \ldots, x_k)| = 2^{m-(i-1)\lceil\sqrt{m}\rceil}$, by Corollary 1.11, there are at least $2^{m-i\lceil\sqrt{m}\rceil+\lceil m^{1/3}\rceil}$ strings $w \in g_{i-1}(x_1, \ldots, x_k)$ such that $C(w) \ge m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$; hence, $C_{s_i}(w) \ge C(w) \ge m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$. Since $z \in g_i(x_1, \ldots, x_k)$, $C_{s_i}(z)$ is in the top $2^{m-i\lceil\sqrt{m}\rceil}$ of $g_{i-1}(x_1, \ldots, x_k)$ in terms of C_{s_i}-complexity. Hence $C_{s_i}(z) \ge m - i\lceil\sqrt{m}\rceil + \lceil m^{1/3}\rceil$.

(2) We show that $C(z) \le m - i\sqrt{m} + 2\log m + O(1)$.

Given $(x_1, \ldots, x_k)$ one can produce $f_i(x_1, \ldots, x_k)$ as follows: Let $f_0(x_1, \ldots, x_m) = \{0,1\}^k$. For $1 \le j \le i$ do the following: find the least s such that $\#_k^{K_s}(x_1, \ldots, x_k) = j$, rank all the strings in $\{0,1\}^m$ via their C_s complexity (break ties via lexicographic ordering), and let $f_j(x_1, \ldots, x_k)$ be the top $2^{m-j\sqrt{m}}$ strings in $f_{j-1}(x_1, \ldots, x_k)$.

Given the lexicographic rank of z in $f_i(x_1, \ldots, x_k)$ one can easily produce z from $f_i(x_1, \ldots, x_k)$.

Hence, to describe z, you need $(x_1, \ldots, x_k)$ and the lexicographic rank r of z in $f_i(x_1, \ldots, x_k)$. The space needed for $(x_1, \ldots, x_k)$ is $2n$ (use the standard trick of encoding 0 by 00, 1 by 11, and commas by 01). Note that $2n = 2\log m$. The space needed for r is $\log|f_i(x_1, \ldots, x_k)| = \log(2^{m-i\sqrt{m}}) = m - i\sqrt{m}$. Hence the total description is size $m - i\sqrt{m} + 2\log m + O(1)$. ∎

Claim 2: For almost all k-tuples $(x_1, \ldots, x_k) \in \mathsf{N}$, if $z \in g_k(x_1, \ldots, x_k)$, and s is the least stage such that $C_s(z) = C(z)$, then $\#_k^{K}(x_1, \ldots, x_k) = \#_k^{K_s}(x_1, \ldots, x_k)$.

Proof: If $\#_k^K(x_1,\ldots,x_k)=0$ then the claim is obvious. Let $s_1,\ldots,s_k$ be as in Claim 1. By Claim 1, if $\#_k^K(x_1,\ldots,x_k)=i$, and $\sum_{i=1}^k |x_i|$ is large enough, then $C_{s_i}(z) > C(z) = C_s(z)$, hence $s > s_i$. Therefore $\#_k^K(x_1,\ldots,x_k)=\#_k^{K_s}(x_1,\ldots,x_k)$. ▮

We now give an algorithm for $\#_k^K(x_1,\ldots,x_k) \in \mathrm{EN}^X(k)$. The algorithm uses h (recall that $C \in \mathrm{EN}^A(k)$ via h and h is computable), and $g_1,\ldots,g_k \leq_{\mathrm{T}} X$. The algorithm works for almost all k-tuples; however, one can easily code the finite information needed to make it always work.

1. Input$(x_1,\ldots,x_k)$.
2. For $0 \leq i \leq k$ compute $g_i(x_1,\ldots,x_k)$.
3. If there exists i, $1 \leq i \leq k$, such that $g_i(x_1,\ldots,x_k) \not\subseteq g_{i-1}(x_1,\ldots,x_k)$ then output $\{0,1,\ldots,k-1\}$ and stop. (This is correct by Claim 0.)
4. (Assume $g_k(x_1,\ldots,x_k) \subseteq \cdots \subseteq g_0(x_1,\ldots,x_k)$.) Let z be the lexicographic least element of $g_k(x_1,\ldots,x_k)$ (such a z must exist since $g_k(x_1,\ldots,x_k)$ is not empty). Enumerate $W_{h(z)}^A$. For each number enumerated we might output a candidate for $\#_k^K(x_1,\ldots,x_k)$. Assume $W_{h(z)}^A$ enumerates c. Find the least s such that $C_s(z)=c$ (this will happen if $c=C(z)$ but might not happen otherwise). Output $\#_k^{K_s}(x_1,\ldots,x_k)$. If $c=C(z)$ then, by Claim 2, $\#_k^K(x_1,\ldots,x_k)=\#_k^{K_s}(x_1,\ldots,x_k)$.

Note that (1) for every number enumerated by $W_{h(z)}^A$ our algorithm may output a candidate for $\#_k^K(x_1,\ldots,x_k)$, and (2) when the correct value of $C(z)$ is enumerated by $W_{h(z)}^A$ our algorithm outputs the correct value for $\#_k^K(x_1,\ldots,x_k)$. Hence $\#_k^K \in \mathrm{EN}^X(k)$. ▮

References

[1] R. Beigel, W. Gasarch, J. Gill, and J. Owings. Terse, Superterse, and Verbose sets. *Information and Computation*, 103(1):68–85, Mar. 1993. Earlier version is TR 1806, Univ of MD, 1987.

[2] W. Gasarch and G. Martin. *Bounded Queries in Recursion Theory.* Progress in Computer Science and Applied Logic. Birkhäuser, Boston, 1999.

[3] C. Jockusch and R. Soare. Π^0_1 classes and degrees of theories. *Transactions of the American Math Society*, 173:33–56, 1972.

[4] M. Kummer. A proof of Beigel's cardinality conjecture. *Journal of Symbolic Logic*, 57(2):677–681, June 1992. `http://www.jstor.org/action/showPublication?journalCode=jsymboliclogic`.

[5] M. Kummer. On the complexity of random strings. In *Thirteenth International Symposium on Theoretical Aspects of Computer Science: Proceedings of STACS 1996,* Grenoble, France, Lecture Notes in Computer Science, New York, Heidelberg, Berlin, 1996. Springer-Verlag. `http://www.springerlink.com`.