What Percentage of Programs Halt?

Laurent Bienvenu\textsuperscript{1,(S)}, Damien Desfontaines\textsuperscript{2}, and Alexander Shen\textsuperscript{3,4}

\textsuperscript{1} LIAFA - CNRS and Université Paris 7, Paris, France  
laurent.bienvenu@liafa.univ-paris-diderot.fr  
\textsuperscript{2} Google Inc., Zurich, Switzerland  
damien@desfontain.es  
\textsuperscript{3} LIRMM - CNRS and Université Montpellier, Montpellier, France  
alexander.shen@lirmm.fr

Abstract. Fix an optimal Turing machine $U$ and for each $n$ consider the ratio $\rho_n^U$ of the number of halting programs of length at most $n$ by the total number of such programs. Does this quantity have a limit value? In this paper, we show that it is not the case, and further characterise the reals which can be the limsup of such a sequence $\rho_n^U$. We also study, for a given optimal machine $U$, how hard it is to approximate the domain of $U$ from the point of view of coarse and generic computability.

1 Introduction

1.1 Motivation

The title of this paper, ‘What percentage of programs halt?’ is intentionally provocative; obviously, the answer depends on the programming language. To make this question reasonable, we need to put some restrictions on the programming language (=interpreter). Following the theory of algorithmic information, we consider “optimal programming languages”. That is, we consider an optimal Turing machine $U$ (see below for the exact definition) and look, for each $n$, at the fraction $\rho_n^U$ of inputs of length at most $n$ on which $U$ halts (among all inputs of those lengths). It is well known that the sequence $\rho_n^U$ is not computable (knowing the exact values of $\rho_n^U$, one can solve the halting problem). What else can be said about it? For example, can $\rho_n^U$ converge to some limit? As we will see, this cannot happen (Theorem 4). What can then be said about the limit points of $\rho_n^U$? They are Martin-Löf random numbers, even relative to $0'$ (Theorem 5). What are the possible values of $\lim \sup \rho_n^U$? All $0'$-lower semicomputable $0'$-random numbers (Theorem 6; for $\lim \inf$ similar question remains open).

In the second part of the paper we build on these results to study a related question: can we somehow approximate the domain of $U$? That is, can we find an algorithm that tells us whether $U(p)$ terminates or not, giving the correct answer for most inputs $p$? This question may be formalized in different ways. For most of them, the answer will not depend on the particular choice of optimal machine, with the notable exception of Theorem 15.

\textcopyright{} Springer-Verlag Berlin Heidelberg 2015  
M.M. Halldórsson et al. (Eds.): ICALP 2015, Part I, LNCS 9134, pp. 219–230, 2015.  
DOI: 10.1007/978-3-662-47672-7_18
All these questions are quite natural and similar results appeared in different settings. In 1974 Nancy Lynch [11] considered similar questions for a more restricted class of machines that are optimal in some effective sense, as defined by Schnorr [15]. Later the question for some specific universal machine was studied by Hamkins and Miasnikov who showed [7] that the halting problem can be approximated in this case. They considered Turing machines with one-sided tape. (Their result implies that corresponding universal machine is not optimal and thus not effectively optimal.) The criterion for the domains of optimal machines (a set is a domain for some optimal machine if and only if it is a computably enumerable set such that the complexity of the number of strings of length at most $n$ in this set is $n - O(1)$) was obtained by Calude, Nies, Staiger, and Stephan [4]. The recursion-theoretic properties of different versions of approximate computability have been studied by Downey, Jockusch, and Schupp [6]. See also Antti Valmari [16] who provides a survey of some other results, including the ones from [14] and [8]. Our goal in this paper is to provide a unified approach that allows us to give simple proofs of known results (sometimes in a more general form) and establish some new ones.

1.2 Definitions and Notation

For a set $E$, by $\chi_E$ we denote the characteristic function of this set. If $E$ is finite, $|E|$ denotes its cardinality. We write ‘log’ for base 2 logarithms.

We denote by $\{0, 1\}^*$ the set of all (finite) binary strings, by $\{0, 1\}^n$ the set of strings of length $n$ and by $\{0, 1\}^{\leq n}$ the set of strings of length at most $n$. The length of a string $x$ is denoted by $|x|$. We denote by $\{0, 1\}^\omega$ the set of infinite binary sequences. They are also identified with real numbers in $[0, 1]$ in binary notation; we mention the cases when the non-uniqueness (the same number has two representations) creates problems.

For a partial computable function $f$, the domain of $f$, denoted by $\text{dom}(f)$, is the set of inputs on which $f$ halts. A machine is a partial computable function from $\{0, 1\}^*$ to $\{0, 1\}^*$. An input $p$ of a machine $M$ is sometimes referred to as a program, and if $M(p) = x$, we say that $p$ is a description of $x$ (relative to $M$), or that $x$ is the output of program $p$.

By $C$ and $K$ we respectively denote the plain and prefix-free versions of Kolmogorov complexity. We assume that the reader has some background in computability theory, Kolmogorov complexity and algorithmic randomness (see, e.g., [5, 10, 13, 17]).

**Definition 1.** A machine $U$ is said to be optimal if for every machine $M$ there is a constant $c_M$ such that whenever $M(p) = x$, there is a $q$ such that $|q| \leq |p| + c_M$ and $U(q) = x$.

This definition is used to define plain Kolmogorov complexity: if $U$ is optimal, then $C_U(x) = \min\{|q|: U(q) = x\}$ is the plain Kolmogorov complexity function (defined up to $O(1)$ additive term).

In the rest of this paper, we assume that $U$ is a fixed optimal Turing machine. Let $H_n$ be the number of programs of length at most $n$ on which $U$ halts, and