Anytime Algorithms for Non-Ending Computations

C. S. Calude[1], D. Desfontaines[2]

[1],University of Auckland, NZ
[2]École Normale Supérieure, Paris, France

CDMTCS-463
June 2014

Anytime Algorithms for Non-Ending Computations

Cristian S. Calude[1] and Damien Desfontaines[2]

[1]Department of Computer Science, University of Auckland
Private Bag 92019, Auckland, New Zealand
www.cs.auckland.ac.nz/~cristian
[2]École Normale Supérieure
45 rue d'Ulm, 75005 Paris, France
desfontain.es/serious.html

June 24, 2014

Abstract

A program which eventually stops but does not halt "too quickly" halts at a time which is algorithmically compressible. This result—originally proved in [4]—is shown to be true in a more general setting. Following Manin [9] we convert the result into an anytime algorithm for the halting problem and we show that the cut-off temporal bound is optimal. We conclude with a final discussion about how undecidable is the halting problem.

1 Introduction

Anytime algorithms exchange execution time for quality of results [6]. Anytime algorithms can be executed in two modes: either by being given a contract time (a set amount of time to execute), or an interruptable method. To improve the solution, anytime algorithms can be continued after they have halted. Instead of correctness, an anytime algorithm returns a result with a "quality measure" which evaluates how close the obtained result is to the result that would be returned if the algorithm ran until completion.

Standard anytime algorithms eventually stop, albeit in a prohibitively long time. Following Manin [9] we use a more general form of anytime algorithm as an approximation for a computation which may not end. The proposed anytime algorithm for the halting problem works in the following way: to test whether a program eventually stops we first compute a temporal bound—the interruptable condition—and execute the program for that specific time. If the computation stops then the program was proved to halt; if the computation does not stop, then we *declare* that the program never stops and evaluate the error probability. By running the program a longer time we can improve its performance either by getting to the halting time or by improving the probability error. The essence of the algorithm is based on the fact that programs which take a long time to halt stop at an algorithmically compressible time.

In the following we will denote by $\mathbb{Z}^+$ the set of positive integers $\{1, 2, \cdots\}$ and let $\overline{\mathbb{Z}^+} = \mathbb{Z}^+ \cup \{\infty\}$. The domain of a partial function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ is denoted $\operatorname{dom}(F)$: $\operatorname{dom}(F) = \{x \in \mathbb{Z}^+ \mid F(x) \neq \infty\}$. All logarithms (log) are implicitly binary. We denote by $\#S$ the cardinality of the set S. We assume familiarity with elementary algorithmic information theory, see [8, 1, 5].

2 A glimpse of algorithmic complexity

2.1 Algorithmic complexity

The *algorithmic complexity* relative to a partially computable function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ is the partial function $\nabla_F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ defined by $\nabla_F(x) = \inf\{y \in \mathbb{Z}^+ \mid F(y) = x\}$; if $F(y) \neq x$ for every $y \geq 1$, then $\nabla_F(x) = \infty$.

The algorithmic complexity is similar to the complexities studied in [8, 4, 9]; the plain Kolmogorov complexity is about the logarithm of the algorithmic complexity. While the Kolmogorov complexity is optimal up to an additive constant, the optimality of ∇ is up to a multiplicative constant.

Proposition 2.1. *Let F be a partially computable function. The following are true:*

(1) *The algorithmic complexity ∇_F is injective and for all $x \in \mathbb{Z}^+$, $F(\nabla_F(x)) = x$ and $\nabla_F(F(x)) \leq x$.*

(2) *For every $M \geq 1$ there exists $x > M$ such that $\nabla_F(x) \geq x$.*

(3) *For every $N \geq 1$, $\#\{i \in \mathbb{Z}^+ \mid \nabla_F(i) \leq N\} \leq N$.*

2.2 Universality

In this section we give a new characterisation of universality which will be useful for some applications.

A partially computable function U is called *universal* if for every partially computable function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ there exists a constant $k_{U,F}$ such that for every $x \in \operatorname{dom}(F)$ we have

$$\nabla_U(x) \leq k_{U,F} \cdot \nabla_F(x). \tag{1}$$

A universal partially computable function U "simulates" any other partially computable function F in the following sense: if $x \in \operatorname{dom}(F)$, then from (1), one can deduce that $\nabla_U(F(x)) \leq k_{U,F} \cdot \nabla_F(F(x)) \leq k_{U,F} \cdot x$, hence there exists $y < k_{U,F}$ in $\operatorname{dom}(U)$ such that $U(y) = F(x)$.

Theorem 2.1. *A partially computable function U is universal iff for every partially computable function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ there exists a constant $c_{U,F}$ such that for every $x \in \operatorname{dom}(F)$ we have*

$$\nabla_U(F(x)) \leq c_{U,F} \cdot x. \tag{2}$$

Proof. Assume U satisfies the condition (1). Taking F to be the identity we get a constant $k_{U,\mathrm{id}}$ such that for every $z \in \mathbb{Z}^+$

$$\nabla_U(z) \leq k_{U,\mathrm{id}} \cdot \nabla_{\mathrm{id}}(z) = k_{U,\mathrm{id}} \cdot z. \tag{3}$$

Next take F satisfying (1) and $x \in \operatorname{dom}(F)$. By definition of ∇_U and the hypothesis, $\nabla_U(x) < \infty$ and we have $U(\nabla_U(x)) = x$, hence $F(U(\nabla_U(x))) = F(x)$ and $\nabla_U(F(U(\nabla_U(x)))) = \nabla_U(F(x))$. Let $M_F = F \circ U$. Using in order (1), the inequality $\nabla_{M_F}(F(x)) \leq \nabla_U(x)$ and (3) we deduce (2):

$$\nabla_U(F(x)) \leq k_{U,M_F} \cdot \nabla_{M_F}(F(x)) \leq k_{U,M_F} \cdot \nabla_U(x) \leq k_{U,M_F} \cdot k_{U,\mathrm{id}} \cdot x,$$

hence $c_{U,F} = k_{U,M_F} \cdot k_{U,\mathrm{id}}$.

Conversely, assume F satisfies the condition (2). For every $x \in \mathbb{Z}^+$ with $\nabla_F(x) < \infty$ we deduce in order the relations $\nabla_F(x) \in \operatorname{dom}(F)$ and $F(\nabla_F(x)) = x$, hence:

$$\nabla_U(x) = \nabla_U(F(\nabla_F(x))) \leq c_{U,F} \cdot \nabla_F(x).$$

The relation (1) is satisfied for $k_{U,F} = c_{U,F}$. □

Comment. The difference between (1) and (2) is in the role played by F: in the traditional condition (1), F appears through ∇_F (which sometimes can be incomputable), while in (2) F appears as argument of ∇_U making the second member of the inequality always computable.

Comment. In [9] a partially computable function $U\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ is called *strongly universal* if for every partially computable function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ there exists a constant $k_{U,F}$ such that for every $x \in \mathbb{Z}^+$ there exists $y \leq k_{U,F} \cdot x$ with $U(y) = F(x)$. It is easy to prove that a partially computable function U is universal iff it is strongly universal and the constant $k_{U,F}$ is the same in both definitions.

Corollary 2.1. *For every universal partially computable function U, every partially computable function $F : \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ and all $x \in dom(F)$ we have:*

$$\nabla_U(F(x)) \leq k_{U,F\circ U} \cdot \nabla_U(x),$$

where $k_{U,F\circ U}$ comes from (1).

Proof. Applying (1) on $F \circ U$ and $F(x)$ and using the definition of ∇, we get:

$$\nabla_U (F(x)) \leq k_{U,F\circ U} \cdot \nabla_{F\circ U} (F(x)) \leq k_{U,F\circ U} \cdot \nabla_U (x) .$$

□

In what follows we will fix a universal partially computable function U and write ∇ instead of ∇_U.

Theorem 2.2. *The complexity ∇ is incomputable.*

Proof. Assume by contradiction that ∇ is computable. Then the partial function $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ defined by $F(x) = \inf\left\{i \in \mathbb{Z}^+ \mid \nabla(i) \geq x^2\right\}$ is partially computable, and, by Proposition 2.1, (4), total. Clearly, $\nabla(F(x)) \geq 2x$, for all $x \in \mathbb{Z}^+$.

By the universality condition (2), there exists a constant $c_F = c_{U,F}$ such that for all $x \in \mathbb{Z}^+$ we have: $\nabla(F(x)) \leq c_F \cdot x$, in contradiction with the inequality $\nabla(F(x)) \geq x^2$.

□

2.3 Algorithmic incompressibility (randomness)

Following [9], an *incompressibility (randomness) cut-off function* is a computable, increasing and divergent function $r\colon \mathbb{Z}^+ \longrightarrow \mathbb{R}^+$ such that the function $x \mapsto \frac{x}{r(x)}$ is increasing and divergent.

Example 2.1. *The following are incompressibility cut-off functions:*

- $r(x) = \log(x), x > 1,$
- $r(x) = x^{\alpha}, 0 < \alpha < 1,$
- $r(x) = \frac{x}{\log(x+1)}.$

Let r be an incompressibility cut-off function. An integer $x \in \mathbb{Z}^+$ is said to be *r-(algorithmic) incompressible (random)* if $\nabla(x) \geq \frac{x}{r(x)}$.

Theorem 2.3. [2] *The set*

$$\text{Incompress}(r) = \left\{x \in \mathbb{Z}^+ \mid \nabla(x) \geq \frac{x}{r(x)}\right\}$$

is immune, i.e. it contains no infinite computably enumerable subsets.

Proof. By the definition of r and Proposition 2.1 (2) the set $\mathrm{Incompress}\,(r)$ is infinite. Assume by absurdity that $\mathrm{Incompress}\,(r)$ contains an infinite computably enumerable subset, hence, it contains an infinite computable subset E. Define the function $F\colon \mathbb{Z}^+ \longrightarrow \mathbb{Z}^+$ by $F\,(x) = \inf\left\{i \in E \mid \frac{i}{r(i)} \geq x^2\right\}$ and observe that F is computable. By the universality condition (2) there is a constant c_F such that for all $x \in \mathbb{Z}^+$, $\nabla\,(F\,(x)) \leq c_F \cdot x$.

In view of the definition of F we have $x^2 \leq \frac{F(x)}{r(F(x))}\cdot$ Because $E \subset \mathrm{Incompress}\,(r)$, we then have $\nabla\,(F\,(x)) \geq \frac{F(x)}{r(F(x))}\cdot$ Consequently, for every $x \in \mathbb{Z}^+$ we have:

$$x^2 \leq \frac{F\,(x)}{r\,(F\,(x))} \leq \nabla\,(F\,(x)) \leq c_F \cdot x,$$

a contradiction. □

Using Proposition 2.1 we get the following two corollaries.

Corollary 2.2. *The set* $\left\{x \in \mathbb{Z}^+ \mid \nabla\,(x) \geq x\right\}$ *is immune.*

Proof. The set $\left\{x \in \mathbb{Z}^+ \mid \nabla\,(x) \geq x\right\}$ is an infinite subset of the immune set $\mathrm{Incompress}\,(r)$, for any incompressibility cut-off function r. □

Corollary 2.3. *Let r be an incompressibility cut-off function. Then, for all $N \in \mathbb{Z}^+$ we have:*

$$\frac{\#\left\{1 \leq x \leq N \mid \nabla\,(x) \geq \frac{x}{r(x)}\right\}}{N} \geq 1 - \frac{1}{r\,(N)} \underset{N\to\infty}{\longrightarrow} 1.$$

3 Incompressibility cut-off

In this section we generalise a result proved by Manin [9] which gives a sufficient condition that the value of a partially computable function F in a point x from its domain is r–compressible.

Theorem 3.1. *Let $F : \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ be a partially computable function and $x \in dom\,(F)$. Assume that*

$$\frac{F\,(x)}{r\,(F\,(x))} \geq k_F \cdot \nabla\,(x)\,, \tag{4}$$

where k_F comes from (1)*. Then, $F\,(x)$ is r-compressible.*

Proof. Using (1) we get: $\nabla\,(F\,(x)) \leq k_F \cdot \nabla\,(x) \leq \frac{F(x)}{r(F(x))}\cdot$

□

Example 3.1. [Manin's incompressibility cut-off] *Assume that F is a partially computable function satisfying the following two conditions for some $x \in \mathrm{dom}\,(F)$ and $\varepsilon > 0$:*

1) $F\,(x) \geq \nabla\,(x)^{1+\varepsilon}$,
2) $\frac{\nabla(x)^{\varepsilon}}{(1+\varepsilon) log(\nabla(x))} \geq k_F$.

Then, $F\,(x)$ is log*–compressible.*

Proof. We have:

$$\frac{F\,(x)}{\log\,(F\,(x))} \geq \frac{\nabla\,(x)^{1+\varepsilon}}{(1+\varepsilon)\log\,(\nabla\,(x))} \geq k_F \cdot \nabla\,(x)\,,$$

so by Theorem 3.1:

$$\nabla\,(F\,(x)) \leq \frac{F\,(x)}{r\,(F\,(x))}.$$

□

The bound (4) used in Theorem 3.1 depends on $\nabla\,(x)$—an incomputable quantity. This choice is due to the fact that by (3), $\nabla\,(x) = O\,(x)$, so a bound of the form $g\,(\nabla\,(x))$ is better than the bound $g\,(x)$. These bounds are asymptotically (up to a multiplicative constant) the same if x is r–incompressible, but the first one can be significantly smaller if $\nabla\,(x) \ll x$. The disadvantage of bound (4) comes from its incomputability. We can get a computable bound in the following way:

Corollary 3.1. *Let $F\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ be a partially computable function and $x \in dom\,(F)$. Assume that*

$$\frac{F\,(x)}{r\,(F\,(x))} \geq c_F \cdot x, \tag{5}$$

where c_F comes from (2). Then, $F\,(x)$ is not r–incompressible.

Proof. Using (2) we have: $\nabla\,(F\,(x)) \leq c_F \cdot x \leq \frac{F(x)}{r(F(x))}$.

□

4 Temporal bounds

Theorem 3.1 and Corollary 3.1 are general results in the sense that they apply to every partially computable function. "Computing" an r–compressible output doesn't seem so difficult (in contrast with generating r–incompressible positive integers). So, what is the use of such a computation?

In this section we will illustrate the use of Corollary 3.1 for a special partially computable function, the time complexity.

Let Steps: $\mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ be the partially computable function such that $U(x) < \infty$ iff $U(\text{Steps}(\text{x})) < \infty$, and if $U(x) < \infty$, then $U(x)$ stops in Steps (x) steps.

If we apply Theorem 3.1 and Corollary 3.1 to Steps we get a similar result to the main theorem of [4], where the bound can be expressed with or without $\nabla(x)$.

Theorem 4.1. *Assume that $U(x)$ halts in t_x steps, with t_x such that $\frac{t_x}{r(t_x)} > k_{\text{Steps}} \cdot \nabla(x)$ or $\frac{t_x}{r(t_x)} > c_{\text{Steps}} \cdot x$. Then, t_x is not r–incompressible.*

To get the entire power of Theorem 4.1 we need to use it in conjunction with the following result stating that the r–incompressible times (at which a computation can halt) is a "small" set of positive integers. To this aim we will work with the *(natural) density* on $\mathcal{P}(\mathbb{Z}^+)$. The natural density is not a probability in Kolmogorov's sense (no such probability can be defined for all subsets of positive integers). However, if a positive integer is "randomly" selected from the set $\{1, 2, \ldots, m\}$, then the probability that it belongs to a given set $A \subset \mathbb{Z}^+$ is

$$p_m(A) = \frac{\#(\{1, \ldots, m\} \cap A)}{m}.$$

If $\lim_{N \longrightarrow \infty} p_m(A)$ exists and is equal to δ, then the set $A \subset \mathbb{Z}^+$ has density $d(A) = \delta$.

In a sense, the density $d(A)$ models "the probability that a randomly chosen integer $x \in \mathbb{Z}^+$ is in A".

A set $A \subset \mathbb{Z}^+$ is said to have *constructive density zero* if there exists a computable function $b\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ such that for every $i \in \mathbb{Z}^+$ we have $p_m(A) < 2^{-i}$ provided $m \geq b(i)$.

Fact 4.1. *For every incompressibility cut-off function r, the following set $\left\{1 \leq x \leq N \mid \nabla(x) < \frac{x}{r(x)}\right\}$ has constructive density zero.*

Proof. The map $x \mapsto \frac{x}{r(x)}$ is increasing as r is an incompressibility cut-off function, so we have

$$\left\{1 \leq x \leq N \mid \nabla(x) < \frac{x}{r(x)}\right\} \subseteq \left\{1 \leq x \leq N \mid \nabla(x) < \frac{N}{r(N)}\right\}.$$

Consequently,

$$p_m\left(\{1 \leq x \leq N \mid x \notin \text{Incompress}\,(r)\}\right) \leq \frac{1}{r\,(N)} \leq 2^{-i},$$

for $N \geq 2^i + 1$, as r is computable, increasing and divergent.

□

Assume that $U\,(x)$ does not stop in time T_x satisfying the second inequality in Theorem 4.1, i.e.

$$\frac{T_x}{r\,(T_x)} > k_{\text{Steps}} \cdot x.$$

Then for every $m \geq T_x$, we have:

$$p_m\left(\{1, \ldots, T_x\} \cap \overline{\text{Incompress}\,(r)}\right) \leq \frac{1}{r\,(m)}.$$

Hence, for every $s \in \mathbb{Z}^+$, if $n \geq M_s^x = \min\left\{n \in \mathbb{Z}^+ \mid r\,(n) \geq s\right\}$, then

$$p_n\left(\{1, \ldots, T_x\} \cap \overline{\text{Incompress}\,(r)}\right) \leq \frac{1}{s}.$$

Given $x, s \in \mathbb{Z}^+$, compute M_s^x, and run $U\,(x)$ for the contracted time M_s^x. If the computation doesn't stop in time M_s^x, then either

- $U\,(x)$ eventually halts and the halting time belongs to a set of density smaller than $\frac{1}{s}$, or
- $U\,(x)$ never stops.

We have obtained the following interruptable divergence anytime algorithm:

```
If $U\,(x)$ doesn't stop in time $M_s^x$, then the probability
(according to density) that $U\,(x)$ never stops is larger than
$1 - \frac{1}{s}$.
```

We can improve the estimation of the time M_s^x in the following way. Consider an injective enumeration $(u_n)_{n \in \mathbb{Z}^+}$ of the complement of Incompress (r) (which is computably enumerable). Then, for every K, we define

$$E\,(K) = \min\left\{j \in \mathbb{Z}^+ \mid u_i \leq K, \text{ for all } 1 \leq i \leq j\right\}.$$

Hence, for every $n \geq T_x$, we have:

$$\begin{aligned}
p_n\left(\left\{z \mid 1 \leq z \leq T_x, \nabla(z) < \frac{z}{r(z)}\right\}\right) &= \frac{\#\left(\{1,...,n\} \bigcap \left\{z \mid 1 \leq z \leq T_x, \nabla(z) < \frac{z}{r(z)}\right\}\right)}{n} \\
&= \frac{\#\left(\{T_x+1,...,n\} \bigcap \left\{z \mid \nabla(z) < \frac{z}{r(z)}\right\}\right)}{n} \\
&\leq \frac{\#\left(\{1,...,n\} \bigcap \left\{z \mid \nabla(z) < \frac{z}{r(z)}\right\}\right) - E(T_x)}{n} \\
&\leq \frac{\frac{n}{r(n)} - E(T_x)}{n} \\
&\leq \frac{1}{r(n)} - \frac{E(T_x)}{n}.
\end{aligned}$$

For $s \geq 1$ we define:

$$m_x^s = \min\left\{n \mid \frac{1}{r(n)} - \frac{E(T_x)}{n} \leq \frac{1}{k}\right\},$$

hence, for every $n \geq m_x^s$, $p_n\left(\left\{z \mid 1 \leq z \leq T_x, \nabla(z) < \frac{z}{r(z)}\right\}\right) \leq \frac{1}{s}$.

Comment. Theorem 4.1 was formulated for the time complexity. In fact it works for every abstract Blum complexity measure for U, i.e. for every partially computable function $B\colon \mathbb{Z}^+ \longrightarrow \overline{\mathbb{Z}^+}$ with the following two properties: a) $B(x) < \infty$ iff $U(x) < \infty$; and b) the predicate "$B(x) = n$" is computable.

5 Optimality of temporal bounds

Let us assume that we have some control over the number of computational steps taken by U. More precisely, we consider the following

> **Assumption.** There exist two integers a and b and a (computable) family of programs $(x_R)_{R\in\mathbb{Z}^+}$ such that $U(x_R)$ halts in exactly $a + R \cdot b$ steps and $x_R \leq b \cdot R$.

This condition may seem artificial, but it is actually verified by all "reasonable" models of computation. Indeed, one can write a program x_R executing the following instructions:

1. compute a large number b from a constant c hard-coded in the source code (for example, $b = c^3$);
2. read the input tape, on which we have placed R, and executes a dummy loop b times;

3. and halt.

The number corresponding to the program is bounded by $K \cdot c^2 \cdot R$, where K is constant: c needs $2\log(c)$ bits to be stored in the source code, while R needs $\log(R)$ bits to be written on the input tape. So, we have

$$\frac{x_R}{b \cdot R} \leq \frac{K \cdot c^2 \cdot R}{c^3 \cdot R} = \frac{K}{c}.$$

If we make c large enough then we have $x_R \leq b \cdot R$.

One can easily verify that this method allows to effectively write a 2–tape Turing machine, or a C/C++ program, having the desired property.

The Assumption above allows us to show that the bound in Theorem 4.1 is optimal. First we need the following independent result.

Lemma 5.1. *Let f be a computable, strictly increasing function such that $\frac{f(x)}{r(f(x))} = o(x)$. Then $g(R) = a + b \cdot f(R)$ is r-incompressible for infinitely many R.*

Proof. Firstly, let us remark that $\frac{g(x)}{r(g(x))} = o(x)$. Indeed, we have:

$$\frac{g(x)}{r(g(x))} = \frac{a + b \cdot f(x)}{r(a + b \cdot f(x))} \leq \frac{a + b \cdot f(x)}{r(f(x))} \leq a + b \cdot \frac{f(x)}{r(f(x))} = o(x).$$

Secondly, as g is injective, there exists a computable function g^{-1} such as for all $x \in \mathbb{Z}^+$, $g^{-1}(g(x)) = x$. Using the universality of U and Corollary 2.1, there exists a constant $k_{g^{-1}}$ such that $\nabla(g^{-1}(x)) \leq k_{g^{-1}} \cdot \nabla(x)$. Using Proposition 2.1 (2) and the fact that $\frac{g(x)}{r(g(x))} = o(x)$, we can choose R so that it satisfies the inequalities:

$$\nabla(R) > R \text{ and } R > k_{g^{-1}} \cdot \frac{g(R)}{r(g(R))}. \qquad (6)$$

We then have:

$$k_{g^{-1}} \cdot \frac{g(R)}{r(g(R))} < R < \nabla(R) = \nabla\left(g^{-1}(g(R))\right) \leq k_{g^{-1}} \cdot \nabla(g(R)),$$

hence $\nabla(g(R)) \geq \frac{g(R)}{r(g(R))}$, that is, $g(R)$ is r-incompressible. As we can choose arbitrarily large integers R verifying (6), the proof is concluded.

□

Theorem 5.1. *In Theorem 4.1, the condition "$\frac{t_x}{r(t_x)} > c_{Steps} \cdot x$" cannot be replaced with the condition "$t_x > f(x)$", where f is a computable, strictly increasing function that verifies the relation $\frac{f(x)}{r(f(x))} = o(x)$.*

Proof. By contradiction, let us assume that Theorem 4.1 is true with the new condition. Let (x_R) be the family of programs previously defined: there exist two integers a and b such that $x_{f(R)}$ halts in $a + b \cdot f(R)$ steps for any R, and $x_{f(R)} \leq b \cdot f(R)$. Let us prove that this quantity can be made r-incompressible.

Choose R be such that $g(R)$ is r-incompressible. Using Lemma 5.1 and the fact that f is strictly increasing we have:

$$f\left(x_{f(R)}\right) \leq b \cdot f(R) < g(R) = t_{x_{f(R)}}.$$

Applying Theorem 4.1 with the new condition implies that $g(R)$ is not r-incompressible, a contradiction. □

6 How incomputable is halting set?

It is well known that the halting set $\mathrm{Halt}(U) = \{x \in \mathbb{Z}^+ \mid U(x) < \infty\}$ is computably enumerable but not computable. In [7] it was proved that if we take U to be the single semi-infinite tape, single halt state, binary alphabet universal Turing machine, then $\mathrm{Halt}(U)$ is decidable on a set of asymptotic probability one, i.e. there exists a decidable, density one set $R \subset \mathbb{Z}^+$ such that $\mathrm{Halt}(U) \cap S$ is decidable.

The result in [7] depends on the chosen universal U as it was proved in [3]. Can we prove a similar result for every universal U? A weak positive answer based on Theorem 4.1 is provided below.

To this aim we extend the notion of density for pairs of positive integers. Let $E \subset \mathbb{Z}^+ \times \mathbb{Z}^+$ and denote by $E(N,M)$ the set $\#(\{(x,t) \in E \mid 1 \leq x \leq N, 1 \leq t \leq M\})$. If $E(N,M)/(MN)$ tends to a limit δ as N and M tend to infinity independently, then we say that the set E has density δ and we write $d(E) = \delta$.

The halting set $\mathrm{Halt}(U) = \{x \in \mathbb{Z}^+ \mid U(x) < \infty\}$ can be rewritten as $\mathrm{Halt}(U) = \{x \in \mathbb{Z}^+ \mid \text{ there exists } t \text{ such that } U(x) \text{ halts in time } t\}$. This form suggests to rewrite $\mathrm{Halt}(U)$ as a set of pairs (x,t) such that U halts in time t.

Theorem 6.1. *The set $\{(x,t) \in \mathbb{Z}^+ \times \mathbb{Z}^+ \mid U(x) \text{ halts in time } t\}$ can be written as a disjoint union of a decidable set and a constructive density zero set.*

Proof. By Theorem 4.1 and Fact 4.1 the set $\{(x,t) \in \mathbb{Z}^+ \times \mathbb{Z}^+ \mid U(x) \text{ halts in time } t\}$ can be written as a disjoint union of the following two sets:

- $\{(x,t) \in \mathbb{Z}^+ \times \mathbb{Z}^+ \mid t \leq T_x \text{ and } U(x) \text{ halts in time } t\}$, and
- $\{(x,t) \in \mathbb{Z}^+ \times \mathbb{Z}^+ \mid t > T_x \text{ and } U(x) \text{ halts in time } t\}$.

The first set is decidable. We show that the second set has constructive density zero. Indeed, if we define

$$H\left(N,M\right) = \#\left(\left\{(x,t) \in \{1,2,\ldots,N\} \times \{1,2,\ldots,M\} \mid t > T_x, U\left(x\right) \text{ halts in time } t\right\}\right),$$

then

$$\begin{aligned}
&\lim_{\substack{N\to\infty \\ M\to\infty}} \frac{H\left(N,M\right)}{NM} \\
&\qquad \leq \lim_{\substack{N\to\infty \\ M\to\infty}} \frac{\#\left(\left\{(x,t) \in \{1,2,\ldots,N\} \times \{1,2,\ldots,M\} \mid t \notin \text{Incompress}\left(r\right)\right\}\right)}{NM} \\
&\qquad \leq \lim_{M\longrightarrow\infty} \frac{\#\left(\left\{1 \leq t \leq M \mid t \notin \text{Incompress}\left(r\right)\right\}\right)}{M} = 0,
\end{aligned}$$

constructively by Fact 4.1. □

Acknowledgement

The authors have been supported in part by the Quantum Computing Research Initiatives at Lockheed Martin.

References

[1] C. S. Calude. *Information and Randomness: An Algorithmic Perspective*, Springer-Verlag, Berlin, 2002 (2nd Edition).

[2] C. Calude, I. Chiţescu. Strong noncomputability of random strings, *Internat. J. Comput. Math.* 11 (1982), 43–45.

[3] C. S. Calude, D. Desfontaines. Universality and Almost Decidability, *CDMTCS Research Report* 462, 2014

[4] C. S. Calude, M. A. Stay. Most programs stop quickly or never halt, *Advances in Applied Mathematics*, 40 (2008), 295–308.

[5] R. Downey, D. Hirschfeldt. *Algorithmic Randomness and Complexity*, Springer, Heidelberg, 2010.

[6] J. Grass. Reasoning about computational resource allocation. An introduction to anytime algorithms, *Magazine Crossroads* 3, 1 (1996), 16–20.

[7] J. D. Hamkins, A. Miasnikov. The halting problem is decidable on a set of asymptotic probability one, *Notre Dame J. Formal Logic* 47 (4) (2006), 515–524.

[8] Yu. I. Manin. *A Course in Mathematical Logic for Mathematicians*, Springer, Berlin, 1977; second edition, 2010.

[9] Yu. I. Manin. Renormalisation and computation II: time cut-off and the Halting Problem, *Math. Struct. in Comp. Science* 22 (2012), 729–751.