International Journal of Foundations of Computer Science
Vol. 26, No. 1 (2015) 153–157

DOI: 10.1142/S0129054115500082

World Scientific
www.worldscientific.com

Complexity of Approximating Functions on Real-Life Computers

Boris Ryabko

Siberian State University of Telecommunications and Information Sciences
Institute of Computational Technology of Siberian Branch of Russian
Academy of Science, Novosibirsk, Russia
boris@ryabko.net

Received 10 May 2014
Accepted 12 August 2014
Communicated by Cristian S. Calude

We address the problem of estimating the computation time necessary to approximate a function on a real computer. Our approach gives a possibility to estimate the minimal time needed to compute a function up to the specified level of error. This can be explained by the following example. Consider the space of functions defined on [0,1] whose absolute value and the first derivative are bounded by C. In a certain sense, for almost every such function, approximating it on its domain using an `Intel x86` computer, with an error not greater than ε, takes at least $k(C, \varepsilon)$ seconds. Here we show how to find $k(C, \varepsilon)$.

Keywords: Computational complexity; performance evaluation; ε-entropy; information theory.

1. Introduction

The problem of estimating the difficulty of computation of functions has attracted attention of mathematicians for a long time; see, for example, [2, 7]. However, despite many efforts, there are many important unsolved problems. One practically important problem can be described as follows. A function and a computer are given. What is the time necessary for approximating the function on its domain with an error of at most ε on this computer? A more specific example: what kind of computer is required in order to calculate the change in temperature of the atmosphere over the next three days, with error of at most 1 degree, if the computation time should not be grater than one hour?

In this paper we obtain some results which can be seen as a step towards solving such problems. Namely, we show that if one wants to approximate functions over a compact set F with an error not greater than ε on a computer *comp*, the computation time t for almost all functions from F satisfies the following inequality:

$$t \geq H_\varepsilon(F)/C(comp),$$

where $H_\varepsilon(F)$ is the ε-entropy of F and $C(comp)$ is the capacity of the computer. Definitions of these values can be found in [3,6] and [4,5], correspondingly, and are briefly introduced below.

2. Preliminaries

2.1. *ε-entropy*

First we recall the definition of ε-entropy [3,6]. Let (X,ρ) be a metric space, let $F \subset X$ and $\varepsilon > 0$. A set Γ is called an ε-net (of F) if for any $x \in F$ there exists a $y \in \Gamma$ such that $\rho(x,y) \le \varepsilon$. The ε-entropy $H_\varepsilon(F)$ is defined as

$$H_\varepsilon(F) = \log N_\varepsilon(F),\ N_\varepsilon(F) = \inf\{|\Gamma| : \Gamma \in \hat{\Gamma}_\varepsilon\}, \tag{1}$$

where $\hat{\Gamma}_\varepsilon$ is the set of all ε-nets and $\log x \equiv \log_2 x$.

$H_\varepsilon(F)$ is also called the relative ε-entropy because it depends on the space (X,ρ) in which the set F is located, that is, on the metric extension of F. The infimum of $H_\varepsilon(F)$ over all metric extensions of F is called the absolute ε-entropy of F. In order to avoid technical complications we consider only the ε-entropy (1).

Informally, if one wants to approximate functions from the set F with an error not greater than some fixed ε, one cannot use less than $2^{H_\varepsilon(F)}$ approximating functions.

Let us consider an example. Let θ, c and t be positive. We denote by $L_t(\theta)$ the set of real functions f defined on $[0,t]$, $t > 0$, which satisfy $|f(x)| \le c$ and $|f(x)-f(y)| \le \theta|x-y|$ for $f \in L_t(\theta), x, y \in [0,t]$ and let $\rho(f,g) = \sup_{x\in[0,t]} |f(x) - g(x)|$. Then

$$H_\varepsilon(L_t(\theta)) = t\,\theta/\varepsilon + O(\log(1/\varepsilon)),$$

where $O()$ is with respect to ε going to 0, see [3].

For the similar set of functions defined on a cube in a multidimensional space, the entropy increases as *const* $(1/\varepsilon)^n$, where n is the dimension. It is worth noting that estimates of the ε-entropy are known for many sets of functions [3].

2.1.1. *Computer capacity*

Let us now briefly introduce the definition of the computer capacity recently suggested in [4,5]. By definition, a computer consists of a set of instructions I and an accessible memory M. Any instruction $x \in I$ contains not only its name (say, JUMP), but memory addresses and indexes of registers. For example, all instructions JUMP which deal with different memory addresses are considered different elements of the set of instructions I. We also suppose that there are instructions for reading data from external devices. By definition, a computer task P is a sequence of instructions $P = x_1\ x_2\ x_3...$, where $x_i \in I$. It is not assumed that all possible sequences of instructions are allowed. In principle, there can be sequences of instructions which are prohibited. In other words, it is possible that sequences of instructions should satisfy some rules or limitations. We denote the set of all

allowable sequences of instructions by S_C, and consider any two different sequences of computer instructions from S_C as two different computer tasks. Thus, any computer task can be represented as a sequence of instructions from S_C. Moreover, the opposite is also true: any sequence of instructions from S_C can be considered as a computer task.

Let us denote the execution time of an instruction x by $\tau(x)$. Then the execution time $\tau(X)$ of a sequence of instructions $X = x_1x_2x_3...x_t$ is given by $\tau(X) = \sum_{i=1}^{t} \tau(x_i)$.

The key observation is as follows: the total number of computer tasks that can be executed in time T is equal to

$$N(T) = |\{X : \tau(X) = T\}|. \tag{2}$$

We define the computer capacity $C_T(I)$ and the limit computer capacity $C(I)$ as follows:

$$C_T(I) = \frac{\log N(T)}{T}, \tag{3}$$

$$C(I) = \limsup_{T\to\infty} \frac{\log N(T)}{T}, \tag{4}$$

where $N(T)$ is defined in (2). In [4, 5] this notation is extended to the case of multicore computers with different kinds of memory.

If we assume that all execution times $\tau(x), x \in I$, are integers and the greatest common divisor of $\tau(x), x \in I$, equals 1, lim sup is equal to lim in the definition of the computer capacity (4):

$$C(I) = \lim_{T\to\infty} \frac{\log N(T)}{T}, \tag{5}$$

see [4]. (This assumption is valid for real computers if the time unit is determined by the so-called clock rate, [4, 5].)

It is worth noting that a simple method of estimation of the computer capacity was suggested in [4, 5] and the capacities of many real computers were estimated, see [1].

3. The Main Result

Let there be a computer (I) which can calculate any function from a set F in time T with an error not greater than $\varepsilon > 0$. The main observation is as follows: the total number of computer tasks executed in time T $(N_T(I))$ cannot be less than the number of elements in the minimal ε-net of the set F, i.e.

$$N_T(I) \geq 2^{H_\varepsilon(F)}. \tag{6}$$

From this observation we obtain the following statement:

Theorem 1. *Suppose that a computer (I) can calculate every function from a set F in time T with an error not greater than $\varepsilon > 0$. Then*

$$T \geq H_\varepsilon(F) \,/\, C_T(I)\,, \tag{7}$$

where $H_\varepsilon(F)$ is the ε-entropy of the set F and $C_T(I)$ is the computer capacity (3).

This statement immediately follows from (6) and the definitions (1) and (3).

In a certain sense, Theorem 1 estimates the computation time C(I) in the worst case. Theorem 1 below shows that the estimate (7) is valid for almost all functions. First we give some definitions. Let $\tau_\varepsilon(f)$ be the time of computation of a function f with an error not greater than $\varepsilon > 0$ (on a computer I). From Theorem 1 we can see that there exists at least one function for which the time of calculation is not less than $H_\varepsilon(F) \,/\, C_T(I)$. Having taken into account this observation we define

$$t^* = H_\varepsilon(F) \,/\, C(I) \tag{8}$$

and consider the set of functions for which the computation time is significantly less than t^*. This set is defined as follows:

$$F_\Delta = \{f : f \in F \ ,\ \tau_\varepsilon(f) \ \leq \ t^*\,\Delta\}, \tag{9}$$

where $\Delta \in (0,1)$ is a constant. For example, if $\Delta = 1/2$, the time of calculation of functions from F_Δ is not greater than half of t^*. In a certain sense, the size of this set is quite small. More precisely, the following statement is true:

Theorem 2. *Let F be a set of functions and let I be a computer that can calculate any function from F in time T with an error not greater than $\varepsilon > 0$. Suppose that Eq.* (5) *is valid for this computer. If $H_\varepsilon(F)$ is large ($H_\varepsilon(F)$ goes to infinity), then the proportion of functions for which the computation time is less than $t^*\,\Delta$, $\Delta \in (0,1)$, does not exceed $2^{-(1-\Delta)\,H_\varepsilon(F)}$:*

$$\frac{2^{H_\varepsilon(F_\Delta)}}{2^{H_\varepsilon(F)}} \ \leq \ 2^{-(1-\Delta)\,H_\varepsilon(F)}\,. \tag{10}$$

Proof. From (5) we can see that for any $\delta > 0$ there exists such T_δ that

$$1 - \delta < C(I)/C_T(I) < 1 + \delta\,, \tag{11}$$

if $T > T_\delta\Delta$. Let us consider a set of functions F such that $H_\varepsilon(F)$ is large enough to have $t^* > T_\delta\Delta$. The following chain of inequalities and equalities concludes the proof:

$$\frac{2^{H_\varepsilon(F_\Delta)}}{2^{H_\varepsilon(F)}} \ \leq \ \frac{2^{C_{t^*\Delta}\ t^*\,\Delta}}{2^{H_\varepsilon(F)}} \ = \ \frac{2^{C_{t^*\Delta}(H_\varepsilon(F)/C(I))\Delta}}{2^{H_\varepsilon(F)}} \leq 2^{-(1-\Delta(1+\delta))\,H_\varepsilon(F)}, \tag{12}$$

where the first inequality follows from the definition of the set F_Δ (9) and (2),(3), the equation from the definition (8), and the second inequality from (11). Having taken into account that these inequalities are true for every δ we obtain (10). □

References

[1] A. Fionov, Y. Polyakov and B. Ryabko, Application of computer capacity to evaluation of intel x86 processors, *Proc. of the* 2011 2*nd Int. Congress on Computer Applictions and Computational Science. Advances in Intellegent and Soft Computation* (*CACS'*2011), **145** (2011), pp. 99–104.

[2] A. Kolmogorov, Various approaches to estimating the difficulty of approximate definition and computation of functions, *Proc. Int. Congress of Mathematicians*, **14** (1963), pp. 369–376.

[3] A. Kolmogorov and V. M. Tikhomirov, ε-entropy and ε-capacity of sets in function spaces, *Amer. Math. Soc. Transl. Ser.* **2 17** (1961) 277–364.

[4] B. Ryabko, An information-theoretic approach to estimate the capacity of processing units, *Performance Evaluation* **69** (2012) 267–273.

[5] B. Ryabko, On the efficiency and capacity of computers, *Applied Mathematics Letters* **25** (2012) 398–400.

[6] V. M. Tikhomirov, Epsilon-entropy, *Encyclopedia of Mathematics* http://www.encyclopediaofmath.org/index.php/Epsilon-entropy.

[7] J. F. Traub and H. Wozniakowski, *A general theory of optimal algorithms* (Academic Press, 1980).