Information Processing Letters 42 (1992) 145–149
North-Holland

25 May 1992

Average case complexity under the universal distribution equals worst-case complexity *

Ming Li

Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

Paul M.B. Vitányi

Centrum voor Wiskunde en Informatica, Kruislaan 413, 1098 SJ Amsterdam, Netherlands
and Faculteit Wiskunde en Informatica, Universiteit van Amsterdam, Amsterdam, Netherlands

Communicated by M.J. Atallah
Received 13 March 1991
Revised 24 February 1992

Abstract

Li, M. and P.M.B. Vitányi, Average case complexity under the universal distribution equals worst-case complexity, Information Processing Letters 42 (1992) 145–149.

The average complexity of any algorithm whatsoever under the universal distribution is of the same order of magnitude as the worst-case complexity. This holds both for time complexity and for space complexity. To focus our discussion, we use as illustrations the particular case of sorting algorithms, and the general case of the average case complexity of NP-complete problems.

Keywords: Analysis of algorithms, computational complexity, average case complexity, universal distribution, Kolmogorov complexity, uniform distribution, worst case complexity, Quicksort, average NP completeness

1. Introduction

For many algorithms the average case running time under some distributions on the inputs is less than the worst-case running time. For instance, using Quicksort on a list of n items to be sorted gives under the Uniform Distribution on the inputs an average running time of $O(n \log n)$ while the worst-case running time is $\Omega(n^2)$. The worst-case running time of Quicksort is typically reached if the list is already sorted or almost sorted, that is, exactly in cases where we actually should not have to do much work at all. Since in practice the lists to be sorted occurring in computer computations are very likely to be sorted or almost sorted, programmers implementing systems involving sorting algorithms tend to resort to fast sorting algorithms of which the provable average run-time is of equal order of magnitude as

Correspondence to: P.M.B. Vitányi, Centrum voor Wiskunde en Informatica, Kruislaan 413, 1098 SJ Amsterdam, Netherlands.

* The work of the first author was supported in part by NSERC Operating Grant OGP0036747. Part of the work was performed while he was with the Department of Computer Science, York University, North York, Ontario, Canada. The work of the second author was supported in part by NSERC International Scientific Exchange Award ISE0046203. A preliminary version of this work was presented at the *30th Ann. IEEE Symp. on Foundations of Computer Science*, 1989, pp. 34–39.

the worst-case run-time, even though this average running time can only be proved to be $O(n \log^2 n)$ under the Uniform Distribution as in the case of Shellsort, or to some randomized version of Quicksort.

In the case of NP-complete problems the question arises whether there are algorithms that solve them in polynomial time "on the average". Whether this phenomenon occurs must depend on the combination of the particular NP-complete problem to be solved and the distribution of the instances. Obviously, some combinations are easy on the average, and some combinations are hard on the average, by tailoring the distribution to the ease or hardness of the individual instances of the problem. This raises the question of a meaningful definition of a "hard on the average" problem.

Levin [4] has shown that for the Tiling problem with uniform distribution of instances there is no polynomial on the average algorithm, unless there exists such an algorithm for each combination of an NP-complete problem and polynomial time computable probability distribution.

Here we show that under the "Universal Distribution" on the inputs *all* algorithms run in average case time (space) which is of the order of magnitude of the worst-case time (space). It is shown that if a probability distribution P is computable or even enumerable, then the P-probability of an object is close to its universal probability with high P-expectation. Hence, the average complexity with respect to the universal distribution seems much more meaningful an indicator for practical considerations than the average with respect to the uniform distribution. Since the former corresponds to the worst-case complexity, the worst-case complexity of an algorithm would appear to be of considerable practical importance after all.

The consequence for the previously mentioned examples is as follows. If the inputs are distributed according to the Universal Distribution, then

1. Quicksort runs in average case time $\Omega(n^2)$, and
2. *all* NP-complete problems are hard to compute on the average unless P = NP.

2. The universal distribution

Let N, Q, and R denote the set of nonnegative integers, nonnegative rational numbers, and nonnegative real numbers, respectively. A superscript "+" excludes zero. We consider countably infinite sample spaces, say $S = N \cup \{u\}$, where u is an "undefined" element not in N. A function P from S into R, such that $\Sigma_{x \in S} P(x) = 1$ defines a *probability distribution* on S. (This allows us to consider defective probability distributions on the natural numbers, which sum to less than one, by concentrating the surplus probability on u.) A probability distribution P is called *enumerable*, if the set of points

$$\{(x, y): x \in N, y \in Q, P(x) > y\},$$

is recursively enumerable. That is, $P(x)$ can be approximated from below by a Turing machine, for all $x \in N$. ($P(u)$ can be approximated from above. A probability distribution P is recursive if $P(x)$ can be approximated both from below and above by a Turing machine, for all x.)

Levin has shown that we can effectively enumerate all enumerable probability distributions, $P_1, P_2, \ldots$. In particular, there exists a *universal enumerable probability distribution*, denoted by, say, $\mathbf{m}$, such that

$$\forall k \in N^+ \exists c > 0 \forall x \in N \quad [c\mathbf{m}(x) \geq P_k(x)]. \quad (1)$$

That is, $\mathbf{m}$ dominates each P_k multiplicatively. It is convenient to define

$$\mathbf{m}(x) = 2^{-K(x)}, \quad (2)$$

where $K(x)$ is the prefix variant of Kolmogorov complexity [2]. In equation (1), the constant c can be set to

$$c = 2^{K(P_k)+O(1)} = 2^{K(k)+O(1)} = O(k \log^2 k). \quad (3)$$

This means that we can take c to be exponential in the length of the shortest self-delimiting binary program to compute P_k.

The universal distribution (rather, its continuous version) was originally discovered by R.J. Solomonoff in 1964, with the aim of predicting continuations of finite prefixes of infinite binary sequences. We can view the discrete probability

density $\mathbf{m}$ as the *a priori* probability [1] of finite objects in absence of any knowledge about them [8]. Levin has shown that Solomonoff's definition, and the two definitions (1) and (2) given above, are equivalent up to a multiplicative constant. Thus, three very different formalizations turn out to define the same notion of universal probability. Such a circumstance is often taken as evidence that we are dealing with a fundamental concept. See [9] for the analogous notions in continuous sample spaces, [3], and [5] or [6] for elaboration of the cited facts and proofs.

This universal distribution has many important properties. Under $\mathbf{m}$, easily describable objects have high probability, and complex or random objects have lo9w probability. Other things being equal, it embodies Occam's Razor, which says we should prefer simple explanations over complicated ones. To give an example, with $x = 2^n$ we have $K(x) \leqslant \log n + 2 \log \log n + O(1)$ and $\mathbf{m}(x) = \Omega(1/n \log^2 n)$. If we generate the binary representation of y by n tosses of a fair coin, apart from the leading "1", then for the overwhelming majority of outcomes we shall have $K(y) > n$ and $\mathbf{m}(y) = O(2^{-n})$.

By Markov's inequality, for any two probability distributions P and Q, for all k, we have $Q(x) < P(x)/k$ with P-probability at least $1 - 1/k$. By equations (1) and (3) therefore, for each enumerable probability distribution $P(x)$ we have

$$\Sigma\{P(x): 2^{K(P)}\mathbf{m}(x) \geqslant P(x) \geqslant \mathbf{m}(x)/k\} \geqslant 1 - 1/k, \tag{4}$$

for all $k > 0$. In this sense, with high P-probability, $P(x)$ is close to $\mathbf{m}(x)$, for each enumerable P. The distribution $\mathbf{m}$ is the unique (up to multiplicative constant) distribution which has that property. If the problem instances are generated algorithmically, then the distribution is enumerable. In absence of any a priori knowledge of the actual distribution therefore, apart from that it is enumerable, studying the average behavior under $\mathbf{m}$ is considerably more meaningful than studying the average behavior under any other particular enumerable distribution (like the uniform one).

3. Average case complexity

Let $x \in N$. Let $l(x)$ denote the *length* of the binary representation of x. Let $t(x)$ be the running time of algorithm A on problem instance x. Define the *worst-case time complexity* of A as $T(n) = \max\{t(x): l(x) = n\}$. Define the *average time complexity* of A with respect to a probability distribution P on the sample space S by

$$T^P_{average}(n) = \frac{\sum_{l(x)=n} P(x)t(x)}{\sum_{l(x)=n} P(x)}.$$

Example (Quicksort). Let us compare the average time complexity for Quicksort under the Uniform Distribution $L(x)$ and the one under the Universal Distribution $\mathbf{m}(x)$. Define $L(x) = 2^{-2l(x)}$, such that the conditional probability $L(x \mid l(x) = n) = 2^{-n}$. We encode the list of elements to be sorted as nonnegative integers in some standard way.

For Quicksort, $T^L_{average}(n) = \Theta(n \log n)$. We may expect that $T^{\mathbf{m}}_{average}(n) = \Omega(n \log n)$. But the Theorem will tell us much more, namely, $T^{\mathbf{m}}_{average}(n) = \Omega(n^2)$! Let us give some intuition why this is the case. With the low average time-complexity under the Uniform Distribution, there can only be $o((\log n)2^n/n)$ strings x of length n with $t(x) = \Omega(n^2)$. Therefore, given n, each such string can be described by its sequence number in this small set, and hence for each such x we find

[1] Consider an enumeration $T_1, T_2, \ldots$ of Turing machines with a separate binary one-way input tape. Let T be such a machine. If T halts with output x, then T has scanned a finite initial segment of the input, say p, and we define $T(p) = x$. The set of such p for which T halts is a prefix code: no such input is a proper prefix of another one. Assume the input is provided by tosses of a fair coin. The probability that T halts with output x is $P_T(x) = \Sigma_{T(p)=x} 2^{-l(p)}$, where $l(p)$ denotes the length of p. Then $\Sigma_{x \in N} P_T(x) \leqslant 1$, the deficit from one being the probability that T does not halt. Concentrate this surplus probability on $P_T(u)$, such that $\Sigma_{x \in S} P_T(x) = 1$. It can be shown that P is an enumerable probability distribution iff $P = \Theta(P_T)$ for some T. In particular, $P_U(x) = \Theta(\mathbf{m}(x))$ for a universal machine U. From this, properties (1), (2), and (3) can be derived.

$K(x \mid n) \leqslant n - \log n + 3 \log \log n$. (Since n is known, we can find each $n-k$ by coding k self-delimiting in 2 log k bits. The inequality follows by setting $k = \log n - \log \log n$.) Therefore, no really random x's, with $K(x \mid n) \geqslant n$, can achieve the worst-case run-time $\Omega(n^2)$. Only strings x which are non-random, with $K(x \mid n) < n$, among which are the sorted or almost sorted lists, and lists exhibiting other regularities, can have $\Omega(n^2)$ running time. Such lists x have relatively low Kolmogorov complexity $K(x)$ since they are regular (can be shortly described), and therefore $\mathbf{m}(x) = 2^{-K(x)}$ is very high. Therefore, the contribution of these strings to the average running time is weighted very heavily. This intuition can be made precise in a much more general form. We assume that all inputs to an algorithm are coded as integers according to some standard encoding.

Theorem. *Let A be any algorithm, provided it terminates for all inputs in N. Let the inputs to A be distributed according to* $\mathbf{m}$. *Then the average case time complexity is of the same order of magnitude as the corresponding worst-case time complexity.*

Proof. We define a probability distribution $P(x)$ on the inputs that assigns high probability to the inputs for which the worst-case complexity is reached, and zero probability for other cases.

Let A be the algorithm involved. Let $T(n)$ be the worst-case time complexity of A. Clearly, $T(n)$ is recursive (for instance by running A on all x's of length n). Define the probability distribution $P(x)$ by:

1. For each $n = 1, 2, \ldots$, define $a_n := \sum_{l(x)=n} \mathbf{m}(x)$;
2. if $l(x) = n$ and x is lexicographically least with $t(x) = T(n)$, then $P(x) := a_n$, else $P(x) := 0$.

It is easy to see that a_n is enumerable since $\mathbf{m}(x)$ is enumerable. Therefore, $P(x)$ is enumerable. Setting $P(u) = \mathbf{m}(u)$, we have defined $P(x)$ such that $\sum_{x \in S} P(x) = \sum_{x \in S} \mathbf{m}(x)$, and $P(x)$ is an enumerable probability distribution. The average case time complexity $T^{\mathbf{m}}_{average}(n)$ with respect to the $\mathbf{m}(x)$ distribution on the inputs, using $c_P \mathbf{m}(x) \geqslant P(x)$ by (1), is obtained by:

$$T^{\mathbf{m}}_{average}(n) = \frac{\sum_{l(x)=n} \mathbf{m}(x) t(x)}{\sum_{l(x)=n} \mathbf{m}(x)}$$

$$\geqslant \frac{1}{c_P} \sum_{l(x)=n} \frac{P(x)}{\sum_{l(x)=n} \mathbf{m}(x)} T(n)$$

$$= \frac{1}{c_P} \sum_{l(x)=n} \alpha \frac{P(x)}{\sum_{l(x)=n} P(x)} T(n)$$

$$= \frac{\alpha}{c_P} T(n),$$

where

$$\alpha = \frac{\sum_{l(x)=n} P(x)}{\sum_{l(x)=n} \mathbf{m}(x)} = 1.$$

The proof of the theorem is finished by the observation that

$$T(n) \geqslant T^{\mathbf{m}}_{average}(n)$$

holds vacuously. □

If P in the proof is P_k in the standard effective enumeration $P_1, P_2, \ldots$ of enumerable semimeasures, then we can set $c_P \leqslant k \log^2 k$ by equation (3). Namely, considering the binary representations of positive integers, $c(k) = \overline{l(k)}k$ is a prefix code with $l(c(k)) = \log k + 2 \log \log k$. Since there is a Turing machine halting with output k iff the input is $c(k)$, the length $K(k)$ of the shortest prefix free program for k does not exceed $l(c(k))$. This gives an interpretation to the constant of proportionality between the $\mathbf{m}$-average complexity and the worst-case complexity: if the algorithm to approximate $P(x)$ from below is the kth algorithm in the standard effective enumeration of all algorithms, then:

$$T^{\mathbf{m}}_{average}(n) \geqslant \frac{T(n)}{k \log^2 k}.$$

Hence we must code the algorithm to compute P as compact as possible to get the most significant lower bound. That is, the ease with which we can describe (algorithmically) the strings which produce a worst-case running time determines the closeness of the average time complexity to the worst-case time complexity.

4. Conclusion

It would seem that the result has implications for algorithm design. Average case analysis obtaining a lower average complexity (like by using the uniform distribution) may be misleading because in reality one deals often with computable or enumerable distributions where the inputs are distributed close to the universal distribution, not according to the uniform distribution. Namely, if x is a random sample from simple computable distribution P, then $\mathbf{m}(x)$ is a good estimate for $P(x)$. In such case the simple inputs have high probability, and the complex (random) inputs have low probability. This may be taken as a formal form of the intuitive statement known as 'Occam's Razor': if we have a choice between two hypotheses we ought to prefer the simplest.

We finish with some immediate corollaries of the analysis in the previous section.

Corollary. *The analogue of the Theorem holds for other complexity measures (like* space *complexity), by about the same proof.*

Corollary. *The* **m***-average time complexity of Quicksort is* $\Omega(n^2)$.

Corollary. *For each* NP*-complete problem, if the problem instances are distributed according to* **m**, *then the average running time of any algorithm that solves it is superpolynomial unless* P = NP. (*A result related to this corollary is suggested in* [1], *apparently using different arguments.*)

Suppose we draw the questions considered in this note in the feasible domain. Does there exist a polynomial time computable probability distribution such that all algorithms in a restricted class (like polynomial time) have an average case running time of the same order of magnitude as the worst-case running time? Following the work reported here, these and related problems are studied by Milterson [7].

Acknowledgment

Richard Beigel, Benny Chor, Gloria Kissin, John Tromp, and Vladimir Uspenskii commented on the manuscript. Mike O'Donnell raised the question we address here during a lecture given by the second author.

References

[1] S. Ben-David, B. Chor, O. Goldreich and M. Luby, On the theory of average case complexity, in: *Proc. 21th STOC* (1989) 204–216.

[2] P. Gács, On the symmetry of algorithmic information, *Soviet Math. Dokl.* **15** (1974) 1477–1481; Correction, *ibid.* **15** (1974) 1481.

[3] P. Gács, Lecture notes on descriptional complexity and randomness, Unpublished manuscript, Boston University, Boston, MA, 1987.

[4] L.A. Levin, Average case complete problems, *SIAM J. Comput.* **15** (1986) 285–286.

[5] M. Li and P. Vitányi, Inductive reasoning and Kolmogorov complexity, in: *Proc. 4th IEEE Structure in Complexity Theory Conf.* (1989) 165–185.

[6] M. Li and P. Vitányi, Kolmogorov complexity and its applications, in: J. van Leeuwen, ed., *Handbook for Theoretical Computer Science, Vol. A* (Elsevier, Amsterdam and MIT Press, Cambridge, MA, 1990), Chapter 4, pp. 187–254.

[7] P.B. Milterson, The complexity of malign ensembles, *SIAM J. Comput.*, to appear.

[8] R.J. Solomonoff, A formal theory of inductive inference, Parts 1 and 2, *Inform. and Control* 7 (1964) 1–22 and 224–254.

[9] A.K. Zvonkin and L.A. Levin,, The complexity of finite objects and development of the concepts of information and randomness by means of the theory of algorithms, *Russian Math. Surveys* **25** (1970) 83–124.