Kolmogorov Complexity Revisited

By Arve Meisingset

Draft 05.10.2017 Norway

This paper discusses addition of context to the Kolmogorov complexity. We argue that addition of name and parameters of programs to generate a finite string may result in assigning the same complexity to all strings of the same length, independently of the program that generated the string.

Finite strings

The **Kolmogorov complexity** of a string is defined to be the length of the shortest computer program (in a predetermined programming language) that produces the string as output.

Let us assume that a string x has the length n. We assume that the alphabet used comprises the digits 0 … 9. The number of possible strings of length n is then 10**n. These numbers have n digits. We want to identify each string. We chose to use the same alphabet as the one being used in the contents of the string. Then the number contained in the string may be used as the identifier of the string.

The n first digits of PI will be one of these strings. We often use the letters PI as the identifier of the string. Here the alphabet is different from that of the string, and the most interesting strings are assigned a short identifier, but has disregarded identification of all the other possible strings of the same length.

We may reduce the lengths of our digital strings by replacing leading zeros by blanks, but this renaming changes nothing. We will use the decimal expansion of PI as the identifier of PI.

Identifying the program

Kolmogorov is searching the shortest program to produce each string, and use the length of this program as the definition of the complexity of the string. The program has no external parameter. Hence, there must be a different program for each of the strings of length n. These programs will need an identifier. Due to the one-to-one correspondence between the original string and its program, we may use the identifier of the string, ie. its contents, as the identifier of the program. When we call the program, we give this identifier, ie. the string, as a parameter. This essential string, that identifies the program, is disregarded in the original formulations by Kolmogorov.

There are Conditioned versions of the Kolmogorov complexity K(x) of the string x. The Conditioned Kolmogorov complexity K(xIy) takes a string y as input, and produces x as output. We have reasoned that we minimum need x as input. Hence, the Conditioned Kolmogorov complexity becomes IxI+IpI, ie. the sum of the length of the input plus the length of the program. There is no Conditioned Kolmogorov complexity that is shorter than the original string IxI+IpI>IxI.

Since the input identifier contains the entire output string, we need no complex program to create each output. We may just copy the input. The copy program may be identical for all string, and it will be the shortest possible program.

We conclude: The Kolmogorov complexity is the same for all strings of the same length n. We need not develop particular series for any of these strings. Each particular series will only add to the complexity.

We observe that the Kolmogorov complexity of decimal expansion of n digits of PI=3,14159265 … and of 11111 … are exactly the same. This observation is contradictory to the very purpose of introducing the Kolmogorov complexity.

We may add other parameters to the input string y. We may add a number z, such that we calculate PI+z, etc. However, this will just map to another of the 10**n strings. Even if we start the decimal expansion of PI at position one million, this will map to the same set of strings.

Final remark: The Kolmogorov complexity may be used as an encryption mechanism. You keep the basic series, eg. about the decimal expansion of PI from position one million, secret, and then add a message coded in z. I have difficulties with seeing other practical engineering applications.

Total expansion

We may create a program that generates the 10**n strings automatically. If we consider the leading zeros to be blanks, then this total set will additionally contain all strings shorter than n. The contents of each string will be the identifier of the string.

In this scenario, the identifier is not given as an input to the program. The program will generate all possible strings with their identifiers. This program will be very short, and its length will be independent of the length of the string.

Rather than using numerals, we may use the English alphabet as base. If so, all possible messages of less than n characters will be among the generated strings. If we set the length to one million, all existing and future publications in the world literature may be included in the set of generated strings. The length of the program will be the same for all publications, independently of their length and contents. Hence, all publications of the same length have the same Kolmogorov complexity.

Note that most of the automatically generated strings will contain plain rubbish. Very few of the texts will be meaningful in some language.

Kolmogorov does not deal with total expansions. He studies the relation between one particular string and each of the particular programs that may produce the string.

Series

There exist 20-30 different series for calculating PI. They all use some kind of recursion of similar order as of the length of the string. Some are converging slowly, while others are faster. One of the series calculates one decimal without knowing the previous ones, but the order of recursions is similar.

Even if the formulas may be written as a short program with recursion or with symbols for repeating SUMs or PRODucts, we may question if the expanded series should be written out for calculating n digits. If so, this consideration makes us conclude that no program without parameters is shorter than the string to be calculated.

Probabilities

“The digits of π have no apparent pattern and have passed tests for statistical randomness, including tests for normality; a number of infinite length is called normal when all possible sequences of digits (of any given length) appear equally often.[21] The conjecture that π is normal has not been proven or disproven.[21]

Since the advent of computers, a large number of digits of π have been available on which to perform statistical analysis. Yasumasa Kanada has performed detailed statistical analyses on the decimal digits of π and found them consistent with normality; for example, the frequencies of the ten digits 0 to 9 were subjected to statistical significance tests, and no evidence of a pattern was found.[22] Any random sequence of digits contains arbitrarily long subsequences that appear non-random, by the infinite monkey theorem. Thus, because the sequence of π's digits passes statistical tests for randomness, it contains some sequences of digits that may appear non-random, such as a sequence of six consecutive 9s that begins at the 762nd decimal place of the decimal representation of π.[23] This is also called the "Feynman point" in mathematical folklore, after Richard Feynman, although no connection to Feynman is known.”

This means that if the universe is finite, a detailed description of the entire universe may exist inside the decimal expansion of PI. When this description is finished, the remaining decimal expansion is still infinite, may contain another description of the same universe, etc. PI is a monster. You may have to do decimal expansion of PI across the gaps between many multiverses until you find this description of our universe.

The one who do not feel unease by the notion of infinity by now should start to wonder about his reasoning. I am a happy finitist. Hence, I do not have this monster in my mind.

Note that string of English texts are not normal, as different letters have different probability, and different conditioned probabilities.

The notion of statistical independence may not be meaningful at any resolution. Each previous digit has a resolution that is ten times larger than that of the current digit. Does this have any impact on the statistical independence? Is statistical independence defined in some context that we yet have not identified?

Shannon

Shannon’s information theory gives a black box statistical description of the information source. This approach has many engineering applications within IT and telecommunication.

Kolmogorov studies the complexity of a fixed string and tries to find a white box description of the information source. This approach has been the subject for much theoretical investigations.

Conclusion

I have become rather sceptical to use of Kolmogorov complexity.

Some ten years ago, I studied Chaitin on system complexity, but I found nothing that I could use.

Note that I am a finitist, and study size of finite systems only.

Addition

I prefer to design a database application to become like an editor of data. The editor software is generic. Hence, the database schema only, including the business rules, distinguishes one application from another.

I count the number of data elements in the schema, and multiply with a constant, to estimate the work effort to implement the application this way. This approach has been used on very large and complex applications, and is my preferred approach.

Most database applications are not designed this way. They design work processes outside the database. They tailor the software to the users way of work. I will say that they programme the user to work in one predefined way of work. Hence, I discourage this approach.

In this approach, you will have to add the complexity of the work processes to the complexity of the database schema. Hence, you will have to use Function point estimation to come up with the planned work effort.

The editor approach to system design, focusing on the database schema, is a kind of focus on a minimal program, a la Kolmogorov. However, the two approaches are not equivalent, even when the data and the business rules are the same. The process oriented approach adds the complexity of these processes.

In this Addition, I have shown that I do not use Kolmogorov complexity according to his definition. I change the definition in order to find an approach that I can use.