12-2014

Among Several Successful Algorithms, Simpler Ones Usually Work Better: A Possible Explanation of an Empirical Observation

Vladik Kreinovich
University of Texas at El Paso, vladik@utep.edu

Olga Kosheleva
University of Texas at El Paso, olgak@utep.edu

Comments:
Technical Report: UTEP-CS-14-73

Among Several Successful Algorithms, Simpler Ones Usually Work Better: A Possible Explanation of an Empirical Observation

Vladik Kreinovich and Olga Kosheleva
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
vladik@utep.edu, olgak@utep.edu

Abstract

Often, several different algorithms can solve a certain practical problem. Sometimes, algorithms which are successful in solving one problem can solve other problems as well. How can we decide which of the original algorithms is the most promising – i.e., which is more probable to be able to solve other problem? In many cases, the simplest algorithms turns out to be the most successful. In this paper, we provide a possible explanation for this empirical observation.

1 Empirical Fact

Search for efficient algorithms. Many practical problems appear all the time. Often, several different algorithms are all successful in solving a certain specific practical problem. Once an algorithm is successful in solving a specific problem, it is reasonable to check if this algorithm – or its modification – can also be used to solve other similar problems.

An empirical observation. In her plenary talk at the IEEE Series of Symposia on Computational Intelligence SSCI'2014 (Orlando, Florida, December 9–12, 2014), Dr. Alice Smith mentioned the following interesting empirical observation [2]: among several successful algorithms for solving a specific problem, usually, simpler ones are the most promising – in the sense that these algorithms and/or their modifications are most successful in solving other problems.

How can we explain this empirical observation?

Comment. This observation is similar to the well-known *Occam razor*, according to which, among several possible hypotheses explaining empirical data, it is beneficial to select the simplest one.

What we plan to do. In this paper, we provide a possible theoretical explanation for this empirical observation.

A known formalization of Occam's razor is based on Kolmogorov complexity (see, e.g., [1]); similarly, our explanation of the above similar empirical fact will also use a similar (but more general) notion of complexity.

2 Analysis of the Problem

Problem: reminder. We have several algorithms x, with different *complexity* $c(x)$. Complexity can be described in different ways: as a number of bits of words in the description of an algorithm, as a weighted number, as Kolmogorov complexity $K(x)$ (i.e., the length of the shortest program that can print the description of x; see [1]), etc.

Based on these complexity values, we want to predict how far each of these algorithms is from the ideal. The corresponding "*distance*" $d(x)$ of an algorithm x can be also measured differently: as average computation time on a certain set of practical problems, as the worst-case computation time, as a more complex characteristic that takes into account average or worst-case accuracy of the result, etc. Once we know the distances $d(x)$, we can select the algorithms which are most promising in the sense that they are the closest to the ideal – i.e., the corresponding distances $d(x)$ are the smallest.

Idea. Of course, the distance $d(x)$ is not a function of complexity: we can have more complex algorithms which are more efficient and thus closer to the ideal, and we can have added complexity that only decreases the algorithm's efficiency. So, if we have two algorithms x and y with different complexities $c(x) < c(y)$, then we cannot definitely conclude whether $d(x) < d(y)$ or $d(x) > d(y)$. However, what we can try to do is see what happens *on average*, over different pairs of algorithms:

- if, over all pairs with $c(x) < c(y)$, the average value of the difference $d(x) - d(y)$ is negative, then, in the absence of any other information, it is reasonable to conclude that

  $$\text{when } c(x) < c(y), \text{ then } d(x) < d(y);$$

  in this case, the simpler algorithms are the most promising;

- on the other hand, if, over all pairs with $c(x) < c(y)$, the average value of the difference $d(x) - d(y)$ is positive, then, in the absence of any other information, it is reasonable to conclude that

  $$\text{when } c(x) < c(y), \text{ then } d(x) > d(y);$$

  in this case, the more complex algorithms are more promising.

From this viewpoint, we need to analyze whether the average value of the difference $d(x) - d(y)$ is positive or negative.

Main assumption. In principle, we can have different measures of complexity. However, all possible measures have one common property: that for each level c, there are only finitely many algorithms c for which $c(x) \leq c$.

Indeed, no matter whether we count number of bits, number of words, number of lines, or some weighted number, once this number is fixed, there are only finitely many possible places for different symbols, and thus, only finitely many possible combinations of symbols.

Similarly, no matter how we measure the distance $d(x)$, for each level d, there are only finitely many algorithms x for which $d(x) \leq d$.

Indeed, whether $d(x)$ describes the average number of elementary computational steps on a given finite set of practical examples, or the largest number of steps, a limitation on $d(x)$ implies a limitation on the number of steps on each of these examples. Since we have a bound on the number of computational steps, and there are only finitely many possible choices for each step, we end up with finitely many possible algorithms.

Let us show that these two properties are sufficient to determine the sign of the average value of the difference $d(x) - d(y)$.

3 Main Result

Definition 1.

- *Let X be a countable set of words in a given language. Elements of this set will be called* algorithms*.*
- *Let us assume that two functions c and d are defined, both functions transform elements $x \in X$ into positive real numbers $c(x) > 0$ and $d(x) > 0$.*
- *The value $c(x)$ will be called* complexity *of an algorithm x, while the value $d(x)$ will be called the* distance *of an algorithm x from the ideal case.*
- *We assume that for every positive number c, there are only finitely many algorithms x for which $c(x) \leq c$.*
- *We also assume that for every positive number d, there are only finitely many algorithms x for which $d(x) \leq d$.*
- *Let us assume that for every $c > 0$, there exists a function that assigns to each algorithm x with $c(x) = c$, a number $w(x) > 0$ (called its* weight*) in such a way that $\sum\limits_{x:c(x)=c} w(x) = 1$.*
- *For each $c > 0$, the* average distance *$d_{\text{av}}(c)$ is defined as $\sum\limits_{x:c(x)=c} w(x) \cdot d(x)$.*

- *For each $k > 0$, n_0, and $n > n_0$, the* average value $A(k, n_0, n)$ *is defined as*
$$A(k, n_0, n) = \frac{1}{n - n_0 + 1} \cdot \sum_{c=n_0}^{n} (d_{\rm av}(c) - d_{\rm av}(c - k)).$$

Comment. The value $A(k, n_0, n)$ is the average difference between the non-idealness of algorithms of larger complexity c and algorithms of smaller complexity $c - k$:

- If this difference is positive, this means that more complex algorithms are further from ideal than simpler algorithms, i.e., that simpler algorithms are more efficient.
- If this difference is negative, this means that more complex algorithms are closer to the ideal than simpler algorithms, i.e., that more complex algorithms are more efficient.

We prove the following result:

Proposition. *For every $k > 0$ and n_0, there exists an integer N such that for all $n \geq N$, we have $A(k, n_0, n) > 0$.*

Discission. In other words, we prove that, on average, *simpler algorithm are* closer to the ideal and thus, *more efficient.* This is exactly what we wanted to explain.

Proof.

1°. Let us first notice that the difference $d_{\rm av}(c) - d_{\rm av}(c - k)$ in which the algorithms' complexity differ by k can be represented as the sum of k differences in which this difference is 1:

$$d_{\rm av}(c) - d_{\rm av}(c - k) = (d_{\rm av}(c) - d_{\rm av}(c - 1)) + (d_{\rm av}(c - 1) - d_{\rm av}(c - 2)) + \ldots +$$
$$(d_{\rm av}(c - (k - 1)) - d_{\rm av}(c - k)).$$

Thus, it is sufficient to prove that the average value of this difference is positive for $k = 1$; once this is proven, the average value of the larger difference will also be positive, as the sum of k positive terms.

Because of this fact, in the following proof, we will only consider the case $k = 1$.

2°. For $k = 1$, we can simplify the expression $A(1, n, n_0)$, since

$$\sum_{c=n_0}^{n} (d_{\rm av}(c) - d_{\rm av}(c - 1)) =$$

$$(d_{\rm av}(n_0) - d_{\rm av}(n_0 - 1)) + ((d_{\rm av}(n_0 + 1) - d_{\rm av}(n_0)) + \ldots + (d_{\rm av}(n) - d_{\rm av}(n - 1)).$$

Here, the term $d_{\rm av}(n_0)$ appears both with a plus sign and with a minus sign, and there two occurrences cancel each other. Similarly, all other terms disappear, and the only remaining terms are $-d_{\rm av}(n_0-1)$ and $d_{\rm av}(n)$. Thus,

$$\sum_{c=n_0}^{n}(d_{\rm av}(c)-d_{\rm av}(c-1))=d_{\rm av}(n)-d_{\rm av}(n_0-1).$$

For $n_0<n$, the denominator $n-n_0+1$ is always positive, so the value $A(1,n_0,n)$ is positive if and only if

$$\sum_{c=n_0}^{n}(d_{\rm av}(c)-d_{\rm av}(c-1))=d_{\rm av}(n)-d_{\rm av}(n_0-1)>0,$$

i.e., if and only if $d_{\rm av}(n)>d_{\rm av}(n_0-1)$. So, to prove the Proposition, it is sufficient to prove that $d_{\rm av}(n)>d_{\rm av}(n_0-1)$ for all sufficiently large n.

We will prove an even stronger statement: that $d_{\rm av}(n)\to\infty$ when $n\to\infty$. Moreover, we will prove that $d_{\rm min}(n)\to\infty$, where

$$d_{\rm min}(c)\stackrel{\rm def}{=}\min_{x:c(x)=c}d(x).$$

Since $d_{\rm av}(c)$ is a weighted average (with positive weights adding up to 1) of the values $d(x)$ with $c(x)=c$, and each of these values $d(x)$ is greater than or equal to $d_{\rm min}(c)$, we thus conclude that $d_{\rm av}(c)\ge d_{\rm min}(c)$ and hence, $d_{\rm min}(n)\to\infty$ implies $d_{\rm av}(n)\to\infty$.

3°. We will prove that $d_{\rm min}(n)\to\infty$ by contradiction. The desired convergence means that

$$\forall M\,\exists N\,\forall n\,(n\ge N\to d_{\rm min}(n)\ge M).$$

Let us assume that this convergence statement is not true. This means that

$$\exists M\,\forall N\,\exists n_N\,(n_N\ge N\,\&\,d_{\rm min}(n_N)<M).$$

By definition of the function $d_{\rm min}(c)$, the value $d_{\rm min}(n_N)$ is the smallest of all the distances $d(x)$ for algorithms x of complexity $c(x)=n_N$. Let x_N be the algorithm of complexity $c(x_N)=n_N$ for which this distance is the smallest, i.e., for which $d(x_N)=d_{\rm min}(n_N)$. Then, for every N, we have $d(x_N)<M$ and $c(x_N)=n_N\ge N$ – hence $c(x_N)\ge N$.

On the other hand, by definition of a distance function, there are only finitely many algorithms with distance $<M$. Let c_0 denote the largest of the complexities for all these algorithms. Then, $d(x_N)<M$ implies that $c(x_N)\le c_0$. On the other hand, for $N=c_0+1$, we should have $c(x_N)\ge N>c_0$, i.e., $c(x_N)>c_0$ – a contradiction.

This contradiction proves that our assumption that $d_{\rm min}(n)\not\to\infty$ is wrong, and thus, indeed, $d_{\rm min}(n)\to\infty$. Therefore, $d_{\rm av}(n)\to\infty$. As we have already shown, this convergence implies the proposition. The statement is proven.

Acknowledgments. This work was supported in part by the National Science Foundation grants HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence), and DUE-0926721.

The authors are thankful to all the participants of the IEEE Series of Symposia on Computational Intelligence SSCI'2014 (Orlando, Florida, December 9–12, 2014) for valuable discussions.

References

[1] M. Li and P. Vitanyi, *Introduction to Kolmogorov Complexity and Its Applications*, Springer, New York, 2008.

[2] A. Smith, "Blast from the past – revisiting evolutionary strategies for the design of engineering systems", *Abstracts of the IEEE Series of Symposia on Computational Intelligence for Engineering Solutions SSCI'2014*, Orlando, Florida, December 9–12, 2014.