6-2010

Minimum Description Length (MDL) Principle as a Possible Approach to Arc Detection

Jan Beck

David Nemir

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

Comments:
Technical Report: UTEP-CS-10-10
Published in *Applied Mathematical Sciences*, 2010, Vol. 4, No. 63, pp. 3143-3152.

Applied Mathematical Sciences, Vol. x, 200x, no. xx, xxx - xxx

Minimum Description Length (MDL) Principle as a Possible Approach to Arc Detection

Jan Beck and David Nemir

X-L Synergy
P.O. Box 23189
El Paso, TX 79923-3189, USA
email appliedmath@janbeck.com

Vladik Kreinovich

Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
email vladik@utep.edu

Abstract

Detecting arcing faults is an important but difficult-to-solve practical problem. In this paper, we show how the Minimum Description Length (MDL) Principle can help in solving this problem.

Mathematics Subject Classification: 68Q30, 93AXX

Keywords: Minimum Description Length (MDL), arcing faults

1 Introduction: The Problem of Arc Detection and How It Is Currently Solved

Electrical systems sometimes start an unplanned "arcing", i.e., producing an electric connection in the normally nonconducting media (e.g., in the air). Arcing can be a result of an unplanned connection between two wires, or the result of an electric wire break down or a connector becoming loose.

Unplanned arcing not only disrupts the normal functioning of an electric system, it can also produce damage. Arcing damage is especially dangerous

in aerospace systems where an arc fault can cause the damage to the wiring system causing an aircraft to crash. Because of this danger, it is extremely important to be able to detect the arcing based on the observed electric current or its rate of change; see, e.g., [10]. The corresponding signal will be denoted by $x(t)$.

Arcing is very difficult to detect because every individual arc is different. The material makeup of the wires and of the insulation, air pressure, and arc severity all affect the arc behavior. As a result, arcs are difficult to model, difficult to predict, and difficult to detect.

The problem of detecting unplanned arcs is made even more complex by the fact that in some practical systems such as brush motors, arc welders, and arc discharge lamps, there are permissible arcs. Electric switches and relays are also sources of permissible arcs. For such systems, we must be able to distinguish between the effect of permissible and unplanned arcs.

There exist many different methods of arc detection; see, e.g., [2, 8, 9, 11]. Most of these methods are based on the analysis of the signal's power spectrum.

A different idea was proposed and implemented in [1]. In this paper, arc detection is based on the following idea:

- Without an arc, the behavior of the system is predictable – in the sense that the future values of the signal $x(t)$ can be reasonably well predicted based on the past value $x(t_1), \ldots, x(t_n)$, $t_1 < t_2 < \ldots < t_n < t$, of this signal.
- In contrast, an arc is unpredictable; so, in the presence of an arc, our attempts to predict the value $x(t)$ based on the past values $x(t_i)$, $t_i < t$, will be much less accurate.

In [1], this idea is implemented based on the usual assumption that the electrical system is linear. Indeed, most components of an electrical system are described by linear differential equations, for which a linear combination of solutions is also a solution. For example, a resistor is described by the equation $V_r = R \cdot I$, a capacitor is described by the equation $q = C \cdot V_c$ $\left(\text{hence } V_c = \dfrac{q}{C}\right)$, and an inductance is described by an equation $\dfrac{dI}{dt} = \dfrac{V_c}{L}$, hence $V_c = L \cdot \dfrac{dI}{dt}$. Thus, e.g., in a closed circuit with a battery of voltage V, we have $V + V_r + V_c + V_i = 0$ hence

$$R \cdot I + \frac{q}{C} + L \cdot \frac{dI}{dt} = -V.$$

Taking into account that $I = \dfrac{dq}{dt}$, we get the usual second-order linear differential equation

$$R \cdot \frac{dq}{dt} + \frac{q}{C} + L \cdot \frac{d^2 I}{dt^2} = -V.$$

To avoid possible confusion between several different meaning of the word "linearity", it should be mentioned that

- while the *equation* is linear – in the sense that a linear combination of solutions is also a solution;
- the *solutions* to this (and other) equations are usually *not* linear functions of time.

For example:

- the above circuit usually leads to a sinusoidal signal,
- while, e.g., the discharge of a capacitor leads to a signal that exponentially decreases with time.

For predictable linear systems, the predicted value $x(t)$ is a linear combination of the previous values $x(t_i)$: $x(t) \approx \sum\limits_{i=1}^{n} a_i \cdot x(t_i)$. In accordance with this idea, the method described in [1] tried to find the values a_i which provide the best prediction, and then makes a decision on whether the arc is present by the amount of the discrepancy between the actual signal and its predictions.

The existing approaches lead to a reasonable arc detection, but sometimes, they miss an arc or lead to a fault arc detection in a situation where there is actually no arcing. It is therefore desirable to develop better arc detection techniques.

2 Non-Linearity of Real-Life Electrical Circuits

The fact that a method from [1] based on the linearity assumption does not always work well is a good indication that non-linearity needs to be taken into account.

While standard components of an electrical system such as resistors, capacitor, and inductors are indeed described by linear equations, devices like solid-state relays have a non-linear Volt-Ampere characteristic and therefore, lead to non-linearity of the system's reaction.

Also, while resistors are linear at low currents typical for electronic devices, at high currents, resistors exhibit non-linear behavior: indeed, the temperature of the resistor increase with the current, and the characteristic of most resistor materials are temperature-dependent.

To some extent, the same effect holds for capacitors as well. Indeed,

- first, the capacitor is often heated up by a nearby resistor;
- second, since capacitors also have resistance, they also change their temperature with the current.

Since the electrical characteristics of most capacitors are temperature-dependent, these capacitors thus also exhibit non-linearity.

This non-linearity explains why the linearity-based method [1] may miss an arc or lead to a false arc detection.

For example, when a linearity-based method from [1] pick up a non-linearity, this non-linearity may not be necessarily an indication of an arc, it may be caused by non-linearity of solid-state switches and other electrical components. In this case, we have a false arc detection.

On the other hand, when the linearity-based method does not detect any change in the level of prediction inaccuracy, it does not necessarily may simply mean that there is no arc, it may simply mean that the increase in non-linearity cased by the presence of the arc is masked by the larger nonlinearity solid-state switches. In this case, the detection method misses the arc.

It should be mentioned that in arc detection, it is important to detect an arc as early as possible, in the earliest moment of time when the signal start exhibiting unpredictable behavior typical of an arc. From this viewpoint, the non-linearity exhibited by resistors and capacitors is a relatively "long-term" effect, since the temperature takes time to rise; thus, it is of a lesser effect on the arc detection. In contrast, the non-linear Volt-Ampere characteristics of semiconductor devices such as solid-state relays are intrinsic, so they produce non-linearity from the very beginning. Thus, the non-linearity of solid-state devices is the main reason why linearized arc detection methods are not always working.

3 Extending the Idea of Detecting Arcs as Unpredictable Behavior to Non-Linear Systems: Main Idea

In view of the non-linear character of the actual electrical systems, we need to extend the above idea of detecting arcs as unpredictable behavior to non-linear systems.

If there is no arc, then the electrical system is deterministic, in the sense that once we know the parameters of the electrical system and the initial values of the currents and charges, we can use the corresponding differential equations to predict the electrical signal $x(t)$ at all moments of time t. In [1], the fact was used for linear differential equations, but the same prediction possibility holds also for differential equations that take the non-linear character of electrical components into account.

For example, to take into account the dependence of the resistance on the temperature, we replace the term $I \cdot R$ with a constant R by a terms that describes this dependence, such as $I \cdot (R_0 + R_1 \cdot (T - T_0))$, and we add extra

differential equations that describe how the resistor's temperature T changes with time.

So, in the absence of the arcs, once we know a reasonably small number of parameters and the equations, we can predict all the measured values $x(t_1), \ldots, x(t_n)$ of the observed electrical signal. In the computer, storing a real number takes a few bytes (usually, 4 or 8 bytes), so storing all the parameters and the equations takes a very small number of bytes.

In contrast, to detect an arc, we perform measurements at tens to hundreds of KHz and above, and keep it for at least a few milliseconds. Usually, thousands of numbers are stored, so storing all the measured values $x(t_1), \ldots, x(t_n)$ requires a few Kilobytes.

The arc bring un-predictability to the system; in the presence of the arc, it is no longer possible to predict the future values of the signal: we can predict the "non-arc" component, but the unpredictable arcs contributes unpredictable additions $\Delta x(t_i)$ to the corresponding signals. Thus, in the presence of the arc, to be able to predict all the values $x(1_1), \ldots, x(t_n)$, it is no longer sufficient to only know a few parameters of the electrical system, we also need to know the values $\Delta x(t_i)$ for all the moments of time at which the arcing occurred.

Let us summarize this situation:

- in the absence of an arc, to predict the observed sequence of measurement results $x(t_1), \ldots, x(t_n)$, it is sufficient to use a few values and a reasonably simple program;
- in the presence of the arc, to predict this sequence, we need a large number of values.

When described in these terms, such a difference becomes not specific to arcing, it is a phenomenon well-studied in computational sciences by Kolmogorov and others, and it is known as *Algorithmic Information Theory*, or *Kolmogorov Complexity*; see, e.g., [7].

Kolmogorov complexity does not deal specifically with real numbers ot differential equations, it deals with general information as it is represented in the computer. Inside the computer, everything is represented as a sequence of 0s and 1s, so every object – be it a real number or a sequence of real numbers (or an image) – can be described as a *binary sequence* s (i.e., as a sequence of 0s and 1s).

Kolmogorov complexity $K(s)$ of a binary sequence s is defined as the smallest size of a the code (program + data) that enables to computer to generate this sequence s. The size of a program is defined, e.g., as the number of bytes that are needed to store this program in the computer. (Strictly speaking, the above definition depends on the choice of a programming language, but it is know that asymptotically, this does not matter much [7].)

From this viewpoint, a sequence s_0 consisting of 1,000,000 zeros is easy to generate: it is sufficient to write a simple loop

```
for(i = 0; i < 1000000; i++)
  printf("0");
```

This program is very short (even if we add all the needed headers etc.). Thus, the Kolmogorov complexity of the regular sequence s_0 is, by defintion, very small, a few dozen bytes.

On the other hand, if we take a sequence s_1 containing 1,000,000 bits generated by a truly random process, then the only way to reproduce this sequence is to store this sequence in the program – which would require 1,000,000 bits, i.e., 125,000 bytes. There may be a few places where accidentally, there is some order, and which can, therefore, be compressed, but asymptotically, this will not significantly decrease the size of the needed program. Thus, the Kolmogorov complexity of the truly random sequence s_1 is approximately 125,000 bytes. (It is important to make sure that we have a truly random physical process, because random number generators in many computer systems often produce deterministic results, uniquely determined by the seed; the Kolmogorov complexity of such a pseudo-random sequence is thus equal to the size of the seed + the size of the program, and is very small.)

The notion of Kolmogorov complexity allows us to reformulate the above observation about the arcs in the following terms:

- When there is no arc, the sequence $s = x(t_1) \ldots x(t_n)$ containing all the observed signals can be reproduced by using a short program with a few parameters. So, the Kolmogorov complexity of this sequence is small.
- In contrast, when there is an arc, we have unpredictable additions to each value $x(t_i)$ and thus, the shortest way to describe this sequence is to store all these additions. Thus, in the presence of the arc, the Kolmogorov complexity of the signal s is large.

This leads to the following natural idea of detecting an arc:

- we measure the Kolmogorov complexity $K(s)$ of the signal

$$s = x(t_1) \ldots x(t_n);$$

- if this value is small – e.g., smaller than a certain threshold K_0 – then we conclude that there is no arc; if $K(s) > K_0$, then we conclude that there is an arc.

This idea, unfortunately, cannot be implemented "as is", because it has been proven that, in general, Kolmogorov complexity of a given sequence cannot be algorithmically computed; see, e.g., [7]. Since we cannot algorithmically

compute the Kolmogorov complexity *exactly*, we therefore have to use algorithms for computing *approximate* values of Kolmogorov complexity.

These algorithm exist – although they were not originally designed to estimate Kolmogorov complexity, they were originally designed for *data and signal compression*, and only later, the relation between compression and Kolmogorov complexity because known – and efficiently utilized (see, e.g., [4, 5]; see also [6], Slides 65–74).

The relation between compression and Kolmogorov complexity is rather straighforward. Indeed, suppose that we have a large image, or a large video, or a large data file. As we have mentioned, in the computer, the image (video, data file) is represented simply as a sequence s of 0s and 1s. Compression means that instead of storing the image bit-by-bit, we store some shorter-size information (compressed file) and a program (decompressor) that enables us to reconstruct the original image. The smaller the size of the compressed file, the better. Thus, an ideal compression of a binary string s would mean that we have found the shortest possible program + data that would enable us to reconstruct s. This is exactly the definition of Kolmogorov complexity.

The fact that Kolmogorov complexity is not algorithmically computable means that we are not able to find the *shortest* (*best*) compression. However, nowadays, there are many reasonable compression algorithms that lead to very good quality compression. Therefore,

- since we cannot compute the *exact* values of the Kolmogorov complexity $K(s)$ – i.e., the size of the compressed string s under ideal (*best*) compression,
- a natural idea is to use the size $\widetilde{K}(s)$ of the compressed string under a *good* compression algorithm.

This idea has been efficiently use in many applications [4, 5, 6].

Our proposal is to use it for arc detection. Specifically, to detect an arc:

- we apply a good compression algorithm to the binary sequence $s = x(t_1) \ldots x(t_n)$ representing the signal $x(t)$, and find the size $\widetilde{K}(s)$ of the compression result;
- if this size is small – e.g., smaller than a certain threshold K_0 – then we conclude that there is no arc; if $\widetilde{K}(s) > K_0$, then we conclude that there is an arc.

Our preliminary experiments have shown that this idea indeed leads to a good arc detection.

4 Important Technical Details

In the current definition of Kolmogorov complexity, we want to reproduce the string s *exactly*. In other words, the notion of Kolmogorov complexity corresponds to *lossless compression*.

In many practical situations, we do not really need the exact reproduction. For example, in arc detection, the values $x(t_i)$ are obtained by measurement. Measurement is never 100% accurate; there are always measurement inaccuracies (= measurement errors), as a result of which the measured values $\widetilde{x}(t_i)$ are, in general, slightly (up to a few percents) different from the actual (unknown) values of the corresponding current. An important part of the measurement error is a random error component, unpredictable (white-noise-type) additions to the measured values.

When we say that we are able to reconstruct the measured values

$$x(1_1), \ldots, x(t_n),$$

what we mean is that we can predict the *actual* values of the current – and thus, we can predict the measurement result with the accuracy of the corresponding measurements.

From this viewpoint, instead of the original Kolmogorov complexity, it is reasonable to use a modification that takes the accuracy into account. Since we only know the string s only approximately (with some acquisition error ε), there is no need to reproduce the string exactly – it is sufficient to reproduce the string s with the accuracy with which it was acquired.

For example, if the know the standard deviation σ of the measurement error, then for the sequence $\widetilde{s} = \widetilde{x}(t_1) \ldots \widetilde{x}(t_n)$ of the actual measurement results and the sequence $s = x(t_1) \ldots x(t_n)$ of actual (unknown) values, we have a relation

$$d(\widetilde{s}, s) \stackrel{\text{def}}{=} \frac{1}{n} \cdot \sum_{i=1}^{n} (\widetilde{x}(t_i) - x(t_i))^2 \approx \sigma^2.$$

In general, we may have different descriptions of the measurement error, but one thing remains the same: there is a function $d(s, \widetilde{\ })$ that describes how close two strings are, and there is a value ε that describes the accuracy of the original string acquisition.

In there terms, the modified definition of Kolmogorov complexity takes the following form: under the distance function d, an ε-Kolmogorov complexity $K_\varepsilon(s)$ of a string s can be described as the smallest size of a program + data that generates a string $\widetilde{s}$ which is ε-close to s – i.e., for which $d(\widetilde{s}, s) \le \varepsilon$.

We have already mentioned that since we cannot compute the exact values of the Kolmogorov complexity, we have to use compression. The above modification means that instead of the *lossless* compression (that reproduced the signal exactly), we can use a *lossy* compression, i.e., a compression that

reproduces the signal with a given accuracy. So, to detect an arc based on the signal $s = x(t_1) \ldots x(t_n)$, we do the following:

- we apply a good lossy compression algorithm to the binary sequence $s = x(t_1) \ldots x(t_n)$ representing the signal $x(t)$, with the reconstruction accuracy equal to the acquisition accuracy ε, and find the size $\widetilde{K}_\varepsilon(s)$ of the compression result;
- if this size is small – e.g., smaller than a certain threshold K_0 – then we conclude that there is no arc; if $\widetilde{K}_\varepsilon(s) > K_0$, then we conclude that there is an arc.

Lossy compressions are usually faster and easier to implement in hardware; thus, the possibility to use a lossy compression enables us to save time on arc detection.

There is an additional potential positive effect of using a lossy compression: when we set this accuracy threshold right, the lossy modification will enable us to filter out (at least some) effects of the measurement noise and thus, lead to a (somewhat) more accurate arc detection.

Acknowledgments

This work was supported in part by NSF grant HRD-0734825 and by Grant 1 T36 GM078000-01 from the National Institutes of Health.

The authors are thankful to Marcus Hutter and Paul Vitanyi for valuable discussions and links.

References

[1] J. Beck and D.C. Nemir, Arc Fault Detection through Model Reference Estimation, Proc. 2006 Aerospace Congress, *Proceedings of the Society Of Automotive Engineers (SAE) Power Systems Conference*, New Orleans, Louisiana, SAE paper #2006-01-3090.

[2] S.J. Brooks, J.W. Dickens, and W.H. Strader, *Arcing Fault Detection System*, U.S. Patent 5,682,101, October 28, 1997.

[3] S. de Rooij and P.M.B. Vitányi, Approximating Rate-Distortion Graphs of Individual Data: Experiments in Lossy Compression and Denoising, *Computing Research Repository (CoRR)*, 2006, paper abs/cs/0609121.

[4] D.L. Donoho, *The Kolmogorov sampler*, Stanford University, Department of Statistics, Technical Report, January 2002.

[5] P.D. Grünwald, *The Minimum Description Length Principle*, MIT Press, Cmbridge, Massachusetts, 2007.

[6] M. Hutter, *Sequential Decisions Based on Algorithmic Probability*, Australia Information and Computation Technology (ICT) Centre of Excellence NICTA, Statistical Machine Learning (SML) group, materials for the Course COMP4670/COMP6467 "Reinforcement Learning and Planning under Uncertainty", http://sml.nicta.com.au/rlp08/Marcus_Slides.pdf

[7] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*, Springer Verlag, New York, 1997.

[8] P. Meckler, K.J. Eichhorn, and W. Ho, Detecting and extinguishing of arcs in aircraft electrical systems, *Proc. of the 2001 Aerospace Congress*, Seattle, Washington, September 10–14, 2001, paper # SAE-2001-01-2657.

[9] B.D. Russell and B.M. Aucoin, *Arc Spectral Analysis System*, U.S. Patent 5,578,931, November 26, 1996.

[10] Underwriters Laboratories, *UL1699 Standard for Arc Fault Circuit Interrupters*, April 2006.

[11] J.J. Zuercher and C.J. Tennies, *Arc Detection Using Current Variation*, U.S. Patent 5,452,223, September 19, 1995.

Received: Month xx, 200x