Ex-Post algorithmic probability

Probabilité algorithmique ex-post

Jean-Louis Dessalles

2011D009

Novembre 2011

Département Informatique et Réseaux
Groupe IC2 : Interaction, Cognition et Complexité

Ex-Post Algorithmic Probability

Jean-Louis Dessalles

Institut Telecom - Telecom ParisTech – LTCI (CNRS UMR 5141)
46 rue Barrault – F-75013 Paris, France
Dessalles@Telecom-ParisTech.fr

Abstract: Classical Probability theory is badly equipped to compute the probability of unique events, as it lacks the means to determine the relevant class to which the event belongs. Algorithmic Probability is designed to deal with individual events. It computes the chances that a given event will be output by a universal computing machine fed with random programs. This definition, however, fails to detect what makes an event noticeable. All non-trivial events are therefore assigned probabilities close to zero. The definition of probability proposed in this paper determines remarkable aspects of a situation *ex post* (after the fact). Contrary to Algorithmic Probability, it computes probability by considering a *difference* in complexity. Improbable situations are situations that are *simpler to describe than to generate*. This definition of probability leads to a new notion of information.

Keywords: probability, unexpectedness, information.

Probabilité algorithmique *ex-post*

Résumé: La théorie classique des probabilités est inadaptée lorsqu'il s'agit de calculer la probabilité d'événements uniques, car elle ne comporte pas de moyens permettant de déterminer la classe à laquelle l'événement appartient. La notion de probabilité algorithmique a été conçue pour s'appliquer à des événements isolés. Elle calcule les chances qu'un événement donné soit produit par une machine de Turing universelle alimentée par des programmes aléatoires. Cette définition, toutefois, ne permet pas de détecter ce qui fait qu'un événement est remarquable. Tous les événements, en pratique, se voient affecter une probabilité proche de zéro. La définition de la probabilité proposée dans ce papier détermine les aspects remarquables d'une situation *ex post* (après coup). Contrairement à la probabilité algorithmique, elle calcule la probabilité en considérant une *différence* de complexité. Les situations improbables sont les situations qui sont *plus simples à décrire qu'à produire*. Cette définition de la probabilité conduit à une nouvelle notion d'information.

Mots clés: probabilité, inattendu, information.

1. Introduction

Most events, in Science or in everyday life, are unique: they will never occur again as such, in exactly the same conditions. Classical Probability calculus does not say much about the occurrence of unique events. A given occurring event or situation *s* must first be generalized by ignoring 'irrelevant' features, before any probability computation can take place. Unfortunately, what counts as relevant or irrelevant is not known. Let's illustrate the problem with an intuitive example.

Suppose that five coins are thrown on the table and that we are asked what the probability of the actual outcome was. Giving a definite answer proves difficult, as one does not know which features are relevant: the face of each coin, their position on the table, the face in relation to the year of issue of the coin, their position in relation to their color or to their monetary value, etc. Now suppose that the five falling coins happen to end up aligned. The event seems amazingly improbable. One 'intuitively' knows that relative positions are relevant to compute probability, whereas face and value remain irrelevant. Once a relevant feature *f*, such as 'being aligned (within a certain precision)', is available, the actual outcome *s* is generalized while maintaining *f* true, to give the set of all situations that share *f*. The set is then numerically compared with the set of all situations, and the ratio gives an estimate of the probability. This example illustrates the fact that computing the probability of a unique event requires two induction operations: one that gives the set $R(s)$ of all situations that are similar to *s* in a relevant way, and another that provides the set $A(s)$ of all alternatives to *s*. $R(s)$ is characterized by the fact that some relevant feature *f* holds, whereas *f* ignored when determining $A(s)$. This induction phase requires that *f* be available *ex ante* (before the fact). When playing dice, one knows in advance that numbers showing on the upper surface are the only relevant features and that positions or orientations should be ignored. In most situations of practical interest, however, relevant features must be discovered *ex post, i.e.* after the event has occurred. Traditional Probability Calculus is not appropriate to compute *ex-post* probability.

Algorithmic Probability looks like a better candidate to deal with unique events. It has been introduced [1] to measure the probability, not of classes of situations, but of individual situations. The algorithmic probability of *s* is the probability that randomly chosen programs generate *s*. It can be assessed using the length of the shortest program that can generate *s* as result. Contrary to Classical Probability, Algorithmic Probability avoids the burden of putting *s* into a class $R(s)$ of presumably similar events. However, as we will see, it still lacks the ability to determine which features are relevant in *s*. As a consequence, any complex situation receives a probability close to zero, since producing it from scratch by mere chance is highly unlikely.

The central claim of this paper is that Algorithmic Probability should be based on the difference between *generation* complexity and *description* complexity, rather than on absolute complexity. In the Algorithmic Probability framework, there is hardly any distinction between *describing* an object or situation *s* and *generating* it. A *description* of *s* is any program *pr* that outputs *s* when run on a Turing machine *T*. We note: $T(pr) = s$. The description complexity $K(s)$ of *s*, also called Kolmogorov complexity, is the length of the shortest program that outputs *s* when run on a *universal* Turing machine *U*.

$$K(s) = \min \{\text{length}(pr) \mid U(pr) = s\} \quad (1)$$

To *generate s*, one considers all programs by increasing length that may run on *U* and that produce *s* as output. The probability of generating *s* through such a process is called *algorithmic probability* $p_a(s)$ [1]. It amount to:

$$p_a(s) = \sum_{U(pr)=s} 2^{-\text{length}(pr)} \quad (2)$$

For the sum to converge, one must make clear that any program *pr* such that $U(pr) = s$ has stopped after having output *s*, so that no program producing *s* is the prefix of another program producing *s*. Since the shortest program provides the major contribution to the sum, one may choose to define p_a by:

$$p_a(s) = 2^{-K(s)} \quad (3)$$

Formulas (1) and (3) illustrate the conceptual proximity between description and generation in the standard algorithmic framework. The conflation between description and generation is however a major problem. It can only encompass situations in which objects result from an unconstrained constructive process, in which case complexity and probability tend to be aligned [2]. In most situations of practical interest, however, observed situations are not produced by unconstrained computing devices.

Physical phenomena result from devices that are not universal Turing machines. Even if we regard these devices as computing machines, they are particular Turing machines, bound to perform certain classes of computations. This is the case for random events, which are considered to result from memoryless devices. A disintegration event in a radioactive sample occurs independently from the duration between the two preceding disintegration events. If it were not the case, periodic disintegration would be the most probable and the most often observed situation, as it would be easiest for the system to merely copy the preceding interval. More generally, one cannot prevent a universal Turing machine from using one of its previous partial outputs *s* when computing new ones. The complexity $K((s, s))$ of the couple (s, s) is close to $K(s)$: $K((s, s)) \approx K(s) + K(c)$, where *c* is a 'copy' operator. Though *c* may not always be trivially simple, we have $K((s, s)) < 2\,K(s)$ for any sufficiently complex situation *s*. The repetition of *s* is more probable than the generation of any new structure *s′* of comparable complexity. Since *c* can be implemented on any universal Turing machine, such a machine cannot be memoryless. Only a particular Turing machine *T*, or equivalently a universal Turing machine *U* bound to execute a particular program p_0 *first*, such that $U(<p_0,s>) = T(s)$ for any *s*, can behave in a memoryless way (<,> denotes an appropriate concatenation).

In what follows, the divergence between description and generation complexity will be used to define the notion of simplicity. From there, a new definition of algorithmic probability that subsumes definition (3) will be derived. To make things more concrete, two simple models of description and generation machines will be provided. They will be useful to show that the new definition of probability produces results which are more in conformity with the intuition of what probability should measure.

2. Probability and abnormal simplicity

2.1 Randomness deficiency

The idea that description and generation might be performed by two different machines has been implicitly used to measure the *randomness deficiency* of a situation [3]. The idea was already present in Kolmogorov initial writings on complexity [4]. Randomness deficiency (RD) is assessed by contrasting a given (computable) probability distribution P with the universal probability distribution p_a, given by (3). The point of RD is that the universal distribution, since it is based on shortest computations, assigns greater probabilities to most considered situations. As the size of computations may vary depending on the machine, we have $\alpha\, p_a(s) \geq P(s)$ for any considered situation s, where the constant α depends on P but not on s [5].

In the definition of RD, p_a is computed from optimal descriptions (simple situations are considered more probable) whereas P (the actual distribution of observed situations) can be seen as the distribution of outputs of a generation machine. Suppose that P is a uniform lottery that outputs series of 10 binary digits. Such a device is expected to produce maximally complex objects most of the time. Maximally complex binary sequences x are incompressible, which means here that $K(x) = 10$. For maximally complex objects, $P(x) = 2^{-10} = p_a(x)$. If the device happens to produce a simple object s, we get $P(s) = 2^{-10} << p_a(s)$, which is the signature of a lack of randomness. If the first observed series is s = 0000000000, $P(s) = 2^{-10}$ as for any series of length 10. On the other hand, $p_a(s)$ is significantly larger, as s may be described using a very short program. The output is much simpler than expected from the generating device and RD has a high value.

Ex-post algorithmic probability $p(s)$ generalizes the notion of randomness deficiency. However, contrary to the standard RD notion, the definition of $p(s)$ avoids any reference to some preexisting probability distribution P. In the spirit of the algorithmic approach to probability, we should only consider computations performed by machines. Instead of contrasting two probability distributions p_a and P, we will contrast the length of the computation leading to the shortest description of s with the length of the computation leading to its generation.

One crucial departure from the definition of randomness deficiency comes from the fact that the objective character of probability is no longer a desired property. *Ex-post* algorithmic probability is intrinsically observer-dependent. The fact that a national Lottery draw matches my phone number is highly improbable to me, but not to the crowd. To represent this property, we suppose that the description of an observed situation s is performed by a particular Turing machine called "observation-machine" or O, which represents the computing powers of the observer (see section 4). *Observation complexity* $C(s)$ is the minimum information needed for O to unambiguously designate s.

$$C(s) = \min \{\text{length}(pr) \mid O(pr) = s\} \qquad (4)$$

This definition is identical to Kolmogorov complexity (1), except that the machine is imposed. Observation complexity will also be called *description complexity.*

The observer not only describes s, but also makes assumptions about the way s has been produced. To represent this ability, we suppose that s is produced by a generation device called "world-machine" or W. Since W is defined by the observer, it appears as a constrained version of O (see section 3).

Generation complexity $C_w(s)$ is defined as the minimum information needed for the "world" W to generate s.

$$C_w(s) = \min \{\text{length}(pr) \mid W(pr) = s\} \quad (5)$$

This definition is again identical to Kolmogorov complexity, except that the machine is a particular one. A situation s is unexpected when $C(s)$ is significantly smaller than $C_w(s)$. *Unexpectedness* is defined as:

$$U(s) = C_w(s) - C(s) \quad (6)$$

Since the observation machine is less constrained than the world machine, $U(s) \geq 0$. *Ex-post* algorithmic probability is defined by converting unexpectedness into binary choices.

$$\boldsymbol{p = 2^{-U}} \quad (7)$$

Note that (7) is an inverted version of Shannon's definition of information, if we equate information with unexpectedness. This is a first reason why *ex-post* algorithmic probability is rightfully called 'probability'. The next section offers further justification.

2.2 Link with standard probability

Suppose that an urn contains N objects with various complicated shapes. One object s is blindly drawn from it. The generation complexity of this event is $C_w(s) = \log(N)$, as the "world" requires $\log(N)$ bits to opt for any given object (all logarithms are supposed to be in base 2). Suppose that we use some feature f to describe s. If K objects in the urn are known to share f, we need $\log(K)$ bits to designate s once f is available. We may write $C(s) \leq C(f) + \log(K)$ (as characterizing s through f may be sub-optimal). We get from (7):

$$p \leq 2^{C(f)} \times K/N \quad (8)$$

When computing *ex-ante* probability, f is given in advance: $C(f) = 0$, and expression (8) coincides with set-theoretic probability K/N. In situations in which probability must be evaluated *ex-post*, however, expression (8) departs from set-theoretic probability by a corrective term, $2^{C(f)}$. This is another justification why *ex-post* algorithmic probability is a genuine probability notion, which subsumes the standard one.

The corrective term $2^{C(f)}$ explains why, in the example of the urn, the event appears more improbable (*ex post*) if the drawn object is the only spherical object rather than if it is the only object that has an odd number of spots on it. Though the drawn object is still unique in the latter case, the reason why it is unique is more complex than in the former case, what makes the event less improbable, according to (8). More generally, every object, if described with enough precision, can be said to be unique; therefore, in a set-theoretic approach to probability, every object should be regarded as improbable. Relation (8) gets us out of this conundrum. Rare objects must be rare for a *simple* reason.

Equation (7) is the central equation of *simplicity theory* [6]. The point is that any situation that appears abnormally simple, *i.e.* that is simpler to describe than to generate, is regarded as improbable. Conversely, situations cannot be improbable if they are not complex to generate or simple to describe.

Some readers may be troubled about the fact that $p(s)$, when cumulated over all possible situations s, adds up to more than 1. The problem seems obvious, since for situations that are complex to describe, $U(s) \approx 0$ and $p(s) \approx 1$. The problem is only apparent, however, as it results from a conflation between *ex-ante* and *ex-post* probability. From a set-theoretic perspective, any situation s, when considered *ex-post*, should be replaced by the set $R(s)$ of all situations that share the relevant features with s, as we saw in the five-coin example. These sets are not mutually exclusive for different situations. No wonder that the sum of the corresponding probabilities is larger than 1.

From an algorithmic probability perspective, the distinction between generation and description demands that we apply formula (3) twice. On the generation side, $2^{-C_w(s)}$ is the algorithmic probability that the world produces s, what we can note $\text{Prob}_a(h(s))$ (where the predicate $h()$ means "happen"). On the description side, $2^{-C(s)}$ is the algorithmic probability $\text{Prob}_a(s)$ that s comes out when the observer considers random descriptions. We can see that $p(s) = \text{Prob}_a(h(s)) / \text{Prob}_a(s)$. Since $h(s)$ entails s (generating $h(s)$ is a way of generating s), we can write:

$$p(s) = \text{Prob}_a(h(s) \mid s) \tag{9}$$

(here, '|' means probabilistic conditional). This probabilistic writing explains why $p(s)$ adds up to more than one when summed over different situations s. Relation (9) is congruent with the name "*ex-post* algorithmic probability", as the probability of the occurrence of s is assessed, considering that s is available. It is also another reason why $p(s)$ is rightfully named 'probability'.

Definition (7) relies on the postulated existence of two machines, W and O. For the sake of concreteness, we propose now two illustrations showing what these machines may consist of.

3. Generation complexity

Considering particular machines instead of a universal Turing machine has several advantages. One is that $C_w(s)$ or $C(s)$ may be computable, whereas Kolmogorov complexity $K(s)$ is not [3]. Another advantage is that W and O may implement models of actual observers (humans or machines) characterized by particular knowledge or biases. The purpose of this section is to show that it is easy to sketch concrete instance of a generation machine. The following section will do the same for the observation machine.

Rational observers have some ideas about the way situations are generated by the world. The world is more constrained than the observer's mind (represented by O). In particular, the world is supposed to function causally: it is not sufficient to think of a property f to have it implemented in reality. We can sketch a simple model for W.

The simplest device representing the world's action is an unbiased lottery. If s is chosen among N alternatives, $C_w(s) = \log(N)$. The world's causal behavior can be elaborated by considering a cascade of random choices. The successive potential states of the world form a tree, *i.e.* a graph $W = (S, T)$ where S is a set of states and $T \subset N \times N$ represents the set of admissible transitions. One particular state s_0 has no incoming transition, whereas all other s_i have exactly one incoming transition: $\exists!\ (x, s_i) \in T$. The degree $d(s_i)$ of a state s_i is the number of outgoing transitions: $d(s_i) = |\{x \mid (s_i, x) \in T\}|$. States s such that $d(s) = 0$ are resulting states, or leaves. In such a graph, there is always one single path $\pi(s_0, s)$ leading from s_0 to a leave s. This path can be seen as a causal explanation for s. The generation complexity of a resulting state is:

$$C_w(s) = \sum_{s_i \in \pi(s_0, s)} \log(d(s_i)) \tag{10}$$

Importantly, the computation of $C_w(s)$ does not require the knowledge of the set of leaves, not even their number. This is a fundamental difference with the computations underlying set-theoretic probability.

The previous definition of the World-machine can be extended to acyclic graphs. In that case, we must add a minimization among all possible paths π_m leading from s_0 to s.

$$C_w(s) = \min_m \sum_{s_i \in \pi_m} \log(d(s_i)) \tag{11}$$

Note that thanks to the restriction to acyclic graphs, W is a prefix-free machine for all states that are mutually exclusive, *i.e.* that belong to different paths (and, in particular, for leaves).

Another extension of the generation process consists in splitting situation s into several independent aspects s_j such that s = & s_j (where & designates co-occurrence). By definition, independence holds if the generation complexity of s is the sum of the partial generation complexities.

$$C_w(s) = \sum_j C_w(s_j) \tag{12}$$

A predicate f may be considered true for a subset of S. We define conditional generation complexity $C_w(s|f(s))$ as the complexity of the shortest path leading to s from a state in which f holds.

$$C_w(s|f(s)) = \min\left\{ \sum_{s_i \in \pi(s_k, s)} \log(d(s_i)) \,\middle|\, \pi(s_k, s) \neq \varnothing \ \& \ f(s_k) = \text{True} \right\} \tag{13}$$

If f is an observed property of s (which means $f(s)$ = True), then $f(s)$ can be generated first.

$$C_w(f(s)) = \min\left\{ \sum_{s_i \in \pi(s_0, s_k)} \log(d(s_i)) \,\middle|\, \forall s_j \in \pi(s_k, s);\ f(s_j) = \text{True} \right\} \tag{14}$$

Then we get:

$$C_w(s) \leq C_w(f(s)) + C_w(s \mid f(s)) \tag{15}$$

The inequality comes from the fact that the best path leading from s_0 to s is no longer guaranteed to be globally minimal in (11) if it is bound to be minimal between s_0 and a state where $f(s)$ is true.

4. Description complexity

The simplest observation machine consists in a short-term memory, which can be implemented as a list (figure 1). This list functions as a stack, with the most recently observed objects being located first. Figure 1 shows a positional code (with no left completion by zeros) that can be used to address elements in the list. The complexity $C(s)$ of a known object s is obtained using the size of its address.

Figure 1: Positional code

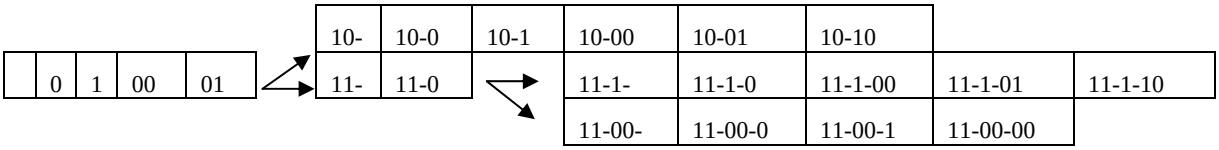
Rank	1	2	3	4	5	6	7	8	9	10	11	12	…
Address		0	1	00	01	10	11	000	001	010	011	100	…

The purpose of *O* is to provide the shortest representation of a given situation *s*. The code shown in figure 1 is designed to be maximally compact. Unsurprisingly, the complexity of an object at rank *r* is approximately log(*r*). The point of designing such a code is to spare bits when addressing the top of the list. Note that the first item requires a null code, which means that it is meant by default when the list itself is addressed.

Note that the code is not prefix-free. To keep the probabilistic interpretation of equation (9), we must turn it into a prefix-free code, for instance by prefixing the address by the indication of its length (this can be done using less than 2×log(1+*L*) bits, where *L* is the size of the address).

The memory structure of the observation machine can be elaborated by distinguishing long-term memory (*LTM*) from short-term memory (*STM*). Ideally, *LTM* should be an associative memory device that can be addressed by content. Figure 2 shows a rough approximation where *LTM* has a tree structure and where each branch has its own prefix. After a branching node, the first node in each branch is used as prefix for the whole subtree (figure 2). Note that the code in figure 2 is not optimally compact, as there is no word starting with 000.

Figure 1: Branching positional code



LTM can be organized by a learning mechanism. If the observer is adapted to its environment, frequent objects are stored higher in the *LTM* tree. Addresses in *LTM* will thus approximate a Shannon-Fano code.

When a situation is new to the observer, it must be constructed, using a set of available operators, instead of being merely retrieved from memory. The observer may rely on preferred operations, such as ‘copy’ or ‘symmetry’. If we wish the observation machine to reflect human cognitive capabilities, it may be given a specialized device to detect group invariants [7]. These preferred operators are stored in an operator list, *OP*, and thus have their own complexity. The complexity of a constructed object is obtained by summing the complexity of the operations and of the operands that are used to construct it.

Our observation machine is thus defined as a triple (*STM*, *LTM*, *OP*). The complexity *C*(*s*) of a situation *s* is either the size of its shortest address in *STM* or in *LTM*, or the size of the cheapest computation that generates *s* (where cost is the cumulated complexity of operators and operands). Due to the minimization in (4), the computation of *C*(*s*) may not be always tractable. For practical purposes, however, it may be performed under limited time resources, as in [8].

The previous description aims at providing a concrete picture of what a basic observation machine may look like. The main point is that the observation machine is only constrained by its own biases: its

operator list and the organization of its memory. It can reuse any previously stored element (even addresses) if it helps finding the shortest path to a given object.

If the long-term memory is associative, a situation s may be described using one of its property $f(s)$. This means that to describe s, the observer may first retrieve the predicate f itself, and then use $f(s)$ to disambiguate s.

$$C(s) \leq C(f) + C(s \mid f(s)) \tag{16}$$

The symbol | in $C(s \mid f(s))$ is the standard conditional for complexity computation. It means that $f(s)$ is available when constructing a description of s. Note that contrary to the generation machine, the observer just counts the complexity of retrieving the feature f, not $f(s)$. For instance, the Pisa Tower, C_w includes the complexity of some causal explanation of the fact that the tower is leaning. On the description size, however, the tower can be individualized among all towers by merely retrieving the predicate 'leaning' from memory, and then by discriminating among all towers that have the 'leaning' property. If this fails to reach uniqueness, an upper bound is obtained by writing $C(s \mid f(s)) \leq \log(K)$, where $K = |\{x \mid f(x) = \text{True}\}|$ is the size of the class characterized by predicate f (note that the extension of predicate f has no objective character here, as it is assessed by the observer). Several characteristics f_i may be successively used to diminish the eventual discrimination among the K undistinguished situations.

5. Relevant features

The main drawback we found in set-theoretic probability and in algorithmic probability is their inability to discard irrelevant properties *ex post*. Definition (7) solves the problem. Let's call *optimal* any feature f used in the optimal description. When only one feature f is used, f is optimal if:

$$C(s) = C(f) + C(s \mid f(s)) \tag{17}$$

A feature f is *relevant* if it can be used to generate compression between $C_w(s)$ and $C(s)$. If s is unexpected: $U(s) > 0$, then any optimal feature is relevant. Even if a large number of features can be found in a situation, only those that contribute to compression are worth paying attention to. In the five-coin example, relative positions are relevant because their description is more compressed than their generation. When computing C_w, we need two numbers per coin to guide the world's decision process. If numbers are represented with 10 bits each (what provides roughly one-millimeter precision), $C_w(s) = 100$. If coins happen to be aligned, their positions can be described with less information. We need two numbers to determine the position of an extreme coin, then one number to designate the angle, and then one additional number for each of the four remaining coins. This makes 7 numbers, $U(s) = 30$ bits and $p(s) = 2^{-30}$. Relevant "observables" are all the characteristics that generate compression, as do positions when coins are aligned. If all five coins show a head, then the common side of the coins becomes relevant, since it contributes to unexpectedness.

We can say that a feature f is *directly relevant* if f is sufficient to make s improbable:

$$C_w(f(s)) - C(f) > 0 \tag{18}$$

Most situations that we encounter are such that $C_w(s) \approx C(s)$, which means that they are not unexpected. This is a non-trivial fact, which is due to our capacity to adapt to the surrounding world. In the previous example about the urn where $C_w(s) = \log(N)$, we are able to find optimal features to

describe *s* so that (in the case of one single feature *f*): $C(f) + C(s|f(s)) = \log(N)$. Experience allows observers to promote frequent properties higher in the memory tree. Memory organization then approximates a Shannon-Fano code, which means that $C(f) \approx \log(N/k)$, where k/N is the frequency of occurrence of *f* in the "world". Once we know that $f(s)$ is true, *s* must be distinguished among *k* objects on average, which means that $C(s|f(s)) \approx \log(k)$. This makes the similarity $C_w(s) \approx C(s)$ most of the time valid. But there are exceptions. Those are the unexpected ones.

We now analyze a few cases in which (7) provides non-trivial results.

6. Examples of unexpected situations

6.1 Improbable structures

A remarkable lottery draw, such as 1–2–3–4–5–6, is perceived as highly improbable, and players are highly reluctant to bet on them [9,10]. In a Lottery game, $C_w(s)$ is supposed to be identical for any draw. $C(s)$, however, is very low for the consecutive combination, compared to a "standard", *i.e.* complex, one. With the kind of coding described in section 4, a consecutive sequence has minimal complexity, since the increment operator is, at least in a generic lottery context, the simplest among all available operators. If we neglect this complexity, then according to formula (7), *ex-post* probability is the same as the probability of winning. *Ex-post* probability grows with the complexity of the draw and is close to 1 for any combination devoid of apparent structure.

Contrary to improbability feelings in the coin throwing experiment, such judgment about lottery draws can be regarded as erroneous. They are traditionally attributed to a representativeness bias [11]. Note, however, that unexpected lottery draws are a reliable clue of cheating whenever fraud can be suspected.

6.2 Improbable closeness

An observer would consider a fire occurring in the vicinity as more improbable than if it occurred in a distant location. Formula (7) explains this effect. The most concise method to code for locations in an isotropic 2-D space consists in numbering them following concentric circles centered on the observer's position. Performed this way, the description complexity of a location *x* of size *a* at distance *d* requires no more than $\log(\pi d^2/a^2)$ bits. Generation complexity depends on the spatial density of similar events. It can be estimated by the distance *D* to the last remembered event of the same class [12]. As far as spatial position is concerned, generation is equivalent to choosing among $\pi D^2/a^2$ locations. Unexpectedness due to location amounts to:

$$U(x) = 2\times\log(D/d) \tag{19}$$

The formula correctly predicts the effect of closeness on probability. The minimal description of location *x* may go through landmarks, when available. The best landmark L_0 satisfies:

$$L_0 = \text{argmin}\,(C(L) + \log(\pi d_L^2/a^2)) \tag{20}$$

Here d_L represents the distance from *L* to *x*. The contribution of position to unexpectedness is thus:

$$U(x) = 2\times\log(D/d_{L_0}) - C(L_0) \tag{21}$$

This explains why a fire occurring on a famous landmark such as the Eiffel Tower is regarded as news.

6.3 Improbable coincidences

Coincidences are universally perceived as improbable [13]. Consider two situations s_1 and s_2. The description complexity of the joint situation s_1 & s_2 obeys the following relation:

$$C(s_1 \,\&\, s_2) \leq C(s_1) + C(s_2|\, s_1) \tag{22}$$

By definition, s_1 and s_2 are *independent* iff $C_w(s_1 \,\&\, s_2) = C_w(s_1) + C_w(s_2)$. We get:

$$U(s_1 \,\&\, s_2) \geq U(s_1) + C_w(s_2) - C(s_2|\, s_1) \tag{23}$$

If we suppose that s_1 and s_2 are not unexpected separately, we get:

$$U(s_1 \,\&\, s_2) \geq C(s_2) - C(s_2|\, s_1) \tag{24}$$

Relation (24) shows that the joint situation (s_1 & s_2) is all the more unexpected as both situations are analogue, since analogy minimizes conditional complexity [14]. It makes correct predictions about the various parameters that influence coincidence intensity [15].

7. Discussion

The definition of probability given by (7) shares with definition (3) the crucial property of computing probability of unique events, in contrast with set-theoretic probability which assigns probability only to sets of events. Contrary to algorithmic probability, which assigns virtually zero probability to all events of significant complexity, definition $p = 2^{-U}$ is able to ignore irrelevant features in the situation. The definition of *ex-post* algorithmic probability is based, not just on complexity, but on a contrast between generation complexity and description complexity. It can be successfully applied in many situations in which other definitions of probability are either silent or give erroneous results.

By turning away from extensional reasoning, e*x-post* algorithmic probability can no longer represent the effect of disjunction or even negation. Kolmogorov's axioms are relevant to *ex-ante* probability, not to *ex-post* probability. It is a necessary price to pay for being able to detect unexpected situations. However, although definition (7) cannot represent p(*not s*) for an unexpected situation *s*, it is possible to look for the most probable (*i.e.* less unexpected) way of undoing *s* [16].

Definition (7) has many advantages that compensate for these limitations and make it useful to perform *ex post* judgments of probability. Section 6 lists some of them. In addition, we may mention the fact that it offers a new way of resolving the information/entropy divorce. Maximally complex states are given maximum probability by definition (7). This is in congruence with the principle of maximal entropy for closed systems in statistical Physics, where the most probable macro-states correspond to complex (unordered) micro-states. This makes sense if we consider that rather than a measure of disorder, entropy is a measure of the typicality of disorder [17].

Definition (7) also distinguishes information from randomness. If we equate information with unexpectedness, a random DNA sequence offers no information whatsoever. According to (6), information requires an observer and two viewpoints: generation and description. A DNA molecule would bring different information values to a forensic investigator who instantiates the murderer's

identity and to a molecular biologist who instantiates a gene. In both cases, formula (6) creates a complexity drop. The amplitude of the drop defines the amount of information retrieved from the DNA sample by the observer.

Ex-post algorithmic probability implements the idea that improbable situations are abnormally simple. The notion of *simplicity* has been introduced to demonstrate that Kolmogorov complexity plays a role in cognition [18, 19]. Simplicity, as defined by (6), contributes to the definition of *interestingness* [16]. It differs from Schmidhuber's notion of interestingness, which is $\partial C(s)/\partial t$ [8]. Unexpected situations elicit a subjective feeling of improbability and of interest, and people feel obliged to talk about them [16].

Simplicity theory [6] builds on definition (7) to explore situations in which human judgments are involved, including bets, decision and emotional intensity [20]. A possible extrapolation would be to consider non-human observers, *e.g.* in data mining or in monitoring, where automata are supposed to spot any unusual pattern.

Acknowledgment

The author is thankful to Paul Vitányi for his constructive comments on a previous version of this paper. This research has been supported in part by the "Chaire Modélisation des Imaginaires, Innovation et Création" (http://imaginaires.telecom-paristech.fr).

References

1. Solomonoff, R. J.. The discovery of algorithmic probability. *J. Comput. Syst. Sci. Int.* **1997**, *55* (1), 73-88.
2. Delahaye, J-P.; Zenil, H. On the Kolmogorov-Chaitin Complexity for short sequences. In *Randomness and Complexity: From Leibniz to Chaitin*. Calude, C.S., Ed.; Singapore: World Scientific; 2007, 1-21**.** Available online: http://arxiv.org/abs/0704.1043 (accessed on 26 May 2011).
3. Li, M.; Vitányi, P. *An introduction to Kolmogorov complexity and its applications*, 3rd edition; New York: Springer Verlag, 2008; Chapter 4.
4. Vovk, V.; Shafer, G. Kolmogorov's contributions to the foundations of probability. *Prob. Pered. Inform.* **2003**, *39* (1), 24-35. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.6531&rep=rep1&type=pdf
5. Vitányi, P.; Li, M., Simplicity, information, Kolmogorov complexity and prediction. In *Simplicity, inference and modelling: Keeping it sophisticatedly simple*, Zellner, A.; Keuzenkampf H. A.; McAleer M., Ed.; Cambridge, UK: Cambridge University Press; 2001, 135-155. Available online: http://www.cwi.nl/~paulv/papers/cup02.ps (accessed on 26 May 2011).
6. http://www.simplicitytheory.org (accessed on 4 January 2009).
7. Leyton, M. *A generative theory of shape*. Springer Verlag, New York, NY, USA; 2001.
8. Schmidhuber, J. Simple algorithmic theory of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. *J. SICE* **2009**, *48* (1), 21-32. Available online: http://www.idsia.ch/~juergen/sice2009.pdf (accessed on 02.06.2011).
9. Savoie, D.; Ladouceur, R. Évaluation et modification de conceptions erronées au sujet des loteries. *Can. J. Behav. Sci.* **1995**, *27* (2), 199-213.

10. Dessalles, J-L. A structural model of intuitive probability. *Proc. Int. Conf. Cogn. Mod.*; Trieste, IT: Edizioni Goliardiche; 2006, 86-91. Available online: http://www.dessalles.fr/papiers/pap.cogni/Dessalles_06020601.pdf (accessed on 31 May 2011)
11. Kahneman, D.; Tversky, A. Subjective probability: A judgement of representativeness. *Cognitive Psychol.* **1972**, *3*, 430-454.
12. http://www.dessalles.fr/Data/Closest-occurrence.pdf (accessed on 3 July 2007).
13. Kern, K.; Brown, K. Using the list of creepy coincidences as an educational opportunity. *Hist. Teach.* **2001**, *34* (4), 531-536.
14. Cornuéjols, A. Analogie, principe d'économie et complexité algorithmique. *Act. J. Fr. Apprent.* ; 1996. Available online: http://www.lri.fr/~antoine/Papers/JFA96-final-osX.pdf (accessed on 26 May 2011)
15. Dessalles, J-L. Coincidences and the encounter problem: A formal account. *Proc. Annu. Conf. Cogn. Sci. Soc.*; Austin, TX: Cognitive Science Society; 2008, 2134-2139. Available online: http://www.dessalles.fr/papiers/pap.conv/Dessalles_08020201.pdf (accessed on 26 May 2011)
16. Dessalles, J-L. *La pertinence et ses origines cognitives - Nouvelles théories.* Paris: Hermes-Science Publications ; 2008. Description available online: http://pertinence.dessalles.fr
17. Lesne, A. *Shannon entropy: a rigorous mathematical notion at the crossroads between probability, information theory, dynamical systems and statistical physics.* IHES prepr. **2011**. Available at: http://preprints.ihes.fr/2011/M/M-11-04.pdf (accessed on 02.06.2011).
18. Chater, N. The search for simplicity: A fundamental cognitive principle? *Q. J. Exp. Psychol.* **1999**, *52* (A), 273-302.
19. Chater, N.; Vitányi, P. Simplicity: a unifying principle in cognitive science? *Trends Cogn. Sci.* **2003**, *7* (1), 19-22.
20. Dessalles, J-L. Simplicity Effects in the Experience of Near-Miss. *Proc. Annu. Conf. Cogn. Sci. Soc.*; Austin, TX: Cognitive Science Society; 2011, 408-413.

Dépôt légal : 2011 – 4ème trimestre
Imprimé à Télécom ParisTech – Paris

ISSN 0751-1345 ENST D (Paris) (France 1983-9999)

Télécom ParisTech
Institut TELECOM - membre de ParisTech
46, rue Barrault - 75634 Paris Cedex 13 - Tél. + 33 (0)1 45 81 77 77 - www.telecom-paristech.frfr
Département INFRES