On-line probability, complexity and randomness

Alexey Chernov[1], Alexander Shen[2], Nikolai Vereshchagin[3], and Vladimir Vovk[1]

[1] Royal Holloway, University of London, Egham, Surrey,
TW20 0EX, UK, {chernov,vovk}@cs.rhul.ac.uk
[2] LIF (Université Aix-Marseille & CNRS), Marseille and Institute
of Information Transmission Problems, Moscow, alexander.shen@lif.univ-mrs.fr
[3] Moscow State University, nikolay.vereshchagin@gmail.com

Abstract. Classical probability theory considers probability distributions that assign probabilities to all events (at least in the finite case). However, there are natural situations where only part of the process is controlled by some probability distribution while for the other part we know only the set of possibilities without any probabilities assigned.
We adapt the notions of algorithmic information theory (complexity, algorithmic randomness, martingales, a priori probability) to this framework and show that many classical results are still valid.

1 On-line probability distributions

Consider the following "real-life" situation. There is a tournament (say, chess or football); before each game the referee tosses a coin to decide which player will start the next game. Assuming the referee is honest, we would be surprised to learn that, say, all 100 coin tosses have produced a tail. We would be surprised also if the result of the coin tossing always turned out to be equal to some (simple) function of the results of previous games. However, it is quite possible that the results of coin tossing can be easily computed from the results of *subsequent* games. Indeed, it may well happen that the coin bit influences the results of the subsequent games and therefore can be reconstructed if these results are known.

Another similar example: if there were a rule that predicts the lucky numbers in a lottery using the previous day newspaper, we would not trust the lottery organizers. However, for the next day newspaper the situation is different (e.g., the newspaper may publish the results of the lottery).

Let X_i be the information string available before the start of ith game (say, the text of the newspaper printed just before the game) and let the bit b_i be the result of coin tossing at the start of ith game. We would like to say that for every function f and for every i the probability of the event $b_i = f(X_i)$ is $1/2$, assuming the referee is honest. And for N games the probability of the event $\forall i\,(b_i = f(X_i))$ equals 2^{-N}.

However, we cannot directly refer to classical probability theory framework in this example. Indeed, when speaking about probability of some event, one usually assumes that some probability distribution is fixed, and this distribution assigns probabilities to all possible events (at least in the finite case). In

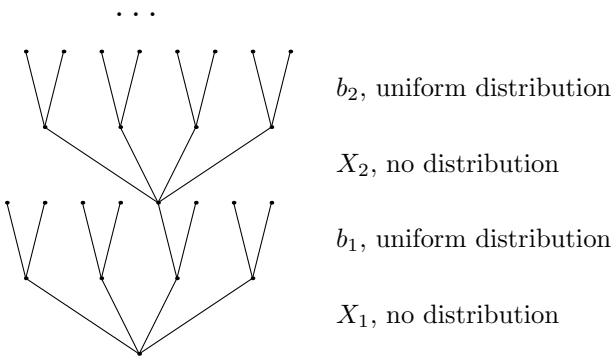

Fig. 1. The tree of possibilities

our example we do not have a probability distribution for X_i; the only thing we have is the "conditional probability" of the event $b_i = 1$ for any condition $X_1, b_1, \ldots, X_{i-1}, b_{i-1}, X_i$; this conditional probability equals $1/2$.

Formally speaking, we get a "tree of possibilities". The sons of the root are possible values of X_1. Each of them has two sons that correspond to two possible outcomes of the first coin tossing ($b_1 = 0$ or 1). Next level branching corresponds to the values of X_2, then each vertex has two sons ($b_2 = 0$ or 1), etc.

In other words, tree vertices are finite sequences $(X_1, b_1, \ldots, X_k, b_k)$ for even layers and $(X_1, b_1, \ldots, X_k, b_k, X_{k+1})$ for odd layers; X_i are binary strings and b_i are bits. We may consider a finite tree with $2N$ layers; its leaves are sequences $(X_1, b_1, \ldots, X_N, b_N)$. Or we may consider an infinite tree whose vertices are sequences of any length.

What we have is not a probability distribution but something that can be called an *on-line probability distribution* on this tree. By definition, to specify an on-line probability distribution one must fix, for each i and for all values of $X_1, b_1, \ldots, X_i$, two non-negative reals with sum 1. They are called *conditional probabilities* of 0 and 1 after $X_1, b_1, \ldots, X_i$ and denoted by

$$\Pr[b_i = 0 \text{ or } 1 | X_1, b_1, \ldots, X_{i-1}, b_{i-1}, X_i].$$

For the case of a fair coin all these conditional probabilities are equal to $1/2$.

As usual, we can switch to unconditional probabilities (i.e., can multiply conditional probabilities on the path from the tree root). Then we arrive to the following version of the definition: an on-line probability distribution is a function P defined on tree vertices such that $P(\Lambda) = 1$ (Λ is the tree root),

$$P(X_1, b_1, \ldots, X_i, b_i, X_{i+1}) = P(X_1, b_1, \ldots, X_i, b_i)$$

(on vertices where no random choice is made, the function propagates without change), and

$$\begin{aligned} P(X_1, b_1, \ldots, X_i, b_i, X_{i+1}) = \\ = P(X_1, b_1, \ldots, X_i, b_i, X_{i+1}, 0) + P(X_1, b_1, \ldots, X_i, b_i, X_{i+1}, 1). \end{aligned}$$

The intuitive meaning of $P(v)$ is the probability to arrive at v if the environment (that chooses $X_1, X_2, \ldots$) wants this and makes suitable moves in its turn.

This definition makes sense both for finite and infinite trees.

Remark. A technical problem arises when some values of an on-line probability distribution are zeros: in this case conditional probabilities cannot be reconstructed from the products. However, in this case they are usually not important, so we can mostly ignore this problem.

Similar on-line probability distributions can be considered for more general trees where on the odd levels, instead of 0 and 1, we have a (countable) list of possible values of b_i.

Now let us assume that the tree is finite (has finite height and finite number of vertices on every level). Consider an event E, i.e., some set of tree leaves. We cannot define the probability of an event under a given on-line probability distribution P. However, we can define an *upper probability* of E. (It may be called a "worst case probability" if the event E is considered undesirable.) This notion can be defined in several (equivalent) ways.

Definition.

(1) Consider all probability distributions on the leaves of the tree. Some of them are consistent with the given on-line probability distribution (i.e., give the same conditional probabilities for b_i when $X_1, b_1, \ldots, X_i$ are given). Upper probability of E is a maximum of $\Pr[E]$ under all these distributions.

(2) Consider the following probabilistic game: a player ("adversary", if the event is undesirable) chooses some X_1, then b_1 is chosen at random with prescribed probabilities (condition X_1), then player chooses X_2, then b_2 is chosen at random (according to the conditional probabilities with condition X_1, b_1, X_2), etc. The player wins if the resulting leaf belongs to the event E. The upper probability of E is the maximal probability that player wins (maximum is taken over all deterministic strategies).

(3) Let us define the *cost* of a vertex in the tree inductively starting from the leaves. For a leaf in E the cost is 1, for a leaf outside E the cost is zero. For a non-leaf vertex v where the choice of X_i is performed, the cost of v is the maximal cost of its sons; for a vertex that corresponds to the choice of b_i, the cost is the weighted sum of the sons' costs where weights are conditional probabilities. Upper probability of E is the cost of the tree root.

(4) Let us consider *on-line martingales* with respect to P, i.e., non-negative functions V defined on tree vertices such that

$$V(X_1, b_1, \ldots, X_i, b_i) = V(X_1, b_1, \ldots, X_i, b_i, X_{i+1}); \tag{1}$$

$$V(X_1, b_1, \ldots, X_i) = V(X_1, b_1, \ldots, X_i, 0) \cdot \Pr[b_i = 0 | X_1, b_1, \ldots, X_i] + \\ + V(X_1, b_1, \ldots, X_i, 1) \cdot \Pr[b_i = 1 | X_1, b_1, \ldots, X_i]; \tag{2}$$

these functions correspond to the player's capital in a fair game (when player observes X_i, the capital does not change; when player splits the capital between bets on $b_i = 0$ and $b_i = 1$, the winning bet is rewarded according to the conditional probabilities determined by the on-line distribution). The upper

probability of E is the minimal value of $V(\Lambda)$ over all V such that $V \geq 1$ for all leaves that belong to E. In other terms, the upper probability of E is 1 divided by the fair price of the option to play such a game with initial capital 1 knowing in advance that the sequence of outcomes belongs to E.

Remark. As we have mentioned, we need some precautions for the case when some values of P are zeros, since in this case conditional probabilities are not uniquely defined. However, it is easy to see that all choices of conditional probabilities compatible with P will lead to the same value of upper probability.

Theorem 1. *All four definitions are equivalent.*

Proof. Note that player's strategy in the second definition determines a distribution on the leaves (X_i is chosen deterministically according to the strategy while b_i is chosen according to the prescribed conditional probabilities). This distribution is consistent with the given on-line distribution. So the upper probability as defined in (2) does not exceed the upper probability as defined in (1). On the other hand, any probability distribution can be considered as a mixed strategy in the game (player chooses her moves randomly using independent random bits), and the winning probability of a mixed strategy is the weighted average of the winning probabilities for pure strategies, so we get the reverse inequality. The inductive definition (3) computes the winning probability for the optimal strategy (induction on tree height).

The equivalence with the martingale definition can be proved in the same way as for the classical off-line setting (this argument goes back to Ville, see, e.g., [7]). If a martingale V starts with capital p and achieves 1 on every leaf in E, then for every probability distribution compatible with P and for every tree vertex the current value of V is an upper bound for the expectation of V if the game starts at this vertex. Therefore, $V(\Lambda)$ is an upper bound for the probability to end the game in E for every probability distribution compatible with P. The reverse inequality: the vertex cost (defined inductively) satisfies the conditions in the definition of a martingale if we replace $=$ by $\geq$ in the condition (1). Increasing this function, we can get a martingale. $\square$

Remarks. **1**. Note that upper probability is not additive: e.g., both the event and its negation can have upper probabilities 1, just the strategies to achieve them are different. However, it is sub-additive: the upper probability of $A \cup B$ does not exceed the sum of upper probabilities of A and B.

2. We can define *supermartingales* in the same way as martingales replacing $=$ by $\geq$ in (2). We relax the requirement (2) and not (1) since it is more natural from the game viewpoint: getting information about X_i does not change the player's capital. It is easy to see that supermartingales may be used instead of martingales in the the definition of upper probability.

3. Proving Theorem 1, we assumed that the tree is finite. However, the same argument shows that it is valid for infinite trees of finite height (and even for the trees having no infinite branches), if we use supremum instead of maximum.

Classical probability theory says that events with very small probability can be safely ignored (and when they happen, we have to reconsider our assumptions about the probability distribution). In the on-line setting we can say the

same about events that have very small upper probability: believing in the probabilistic assumption, we may safely ignore the possibilities that have negligible upper probabilities, and if such an event happens, we have to reconsider the assumption.

Remarks. 1. In fact upper probability (though not with this name) is used in the definition of the Arthur – Merlin class in computational complexity theory where a tree of polynomial height and a polynomially decidable event are considered and we distinguish between events of low and high upper probability.

2. It is easy to see that on-line martingales with respect to on-line probability distribution P are just the ratios Q/P where Q is some other on-line probability distribution. (Some evident precautions are needed if P can be zero somewhere.)

2 On-line Kolmogorov complexity *KR*

We can adapt the standard definition of Kolmogorov complexity (see, e.g., [3, 5] for the definition and discussion of different versions of Kolmogorov complexity) for the on-line setting. Consider a sequence $X_1, b_1, X_2, b_2, \ldots, X_n, b_n$ where X_i are binary strings and b_i are bits. Look for a shortest interactive program that after getting input X_1 produces b_1, then after getting X_2 (in addition to X_1) produces b_2, then after getting X_3 produces b_3 etc. We call its length the *on-line decision complexity* with respect to the programming language π used, and denote it by $KR_\pi(X_1 \to b_1; X_2 \to b_2; \ldots; X_n \to b_n)$. The reason for the name "decision complexity": if all X_i are empty, we get the standard notion of decision complexity of a bit string $b_1 \ldots b_n$ (the length of the shortest program that generates b_i given i). It is easy to see that a natural version of optimality theorem holds (there exists an optimal "programming language"), so the on-line decision complexity (for an optimal programming language) $KR(X_1 \to b_1; X_2 \to b_2; \ldots; X_n \to b_n)$ is well defined (up to an additive $O(1)$-term).

Theorem 2. *The on-line complexity* $KR(X_1 \to b_1; \ldots; X_n \to b_n)$ *does not exceed the decision complexity* $KR(b_1 b_2 \ldots b_n)$ *and is greater than the conditional complexity* $KS(b_1 b_2 \ldots b_n | X_1, X_2, \ldots, X_n)$ *up to* $O(1)$ *terms.*

In other terms, knowing X_i in an on-line setting may help to describe $b_1, \ldots, b_n$, but knowing all X_i in advance is even better. (The proof is straightforward.)

3 On-line a priori probability and *KA*

It is well known that Kolmogorov complexity is related to the a priori probability (maximal lower semicomputable semimeasure). The latter can be naturally defined in the on-line setting. Let us give two equivalent definitions.

Consider an interactive probabilistic machine T that has internal random bit generator. This machine gets some binary string X_1 (say, on the tape where the end of X_1 is marked by a special separator), performs a computation that uses X_1 and random bits and may produce bit b_1 (or hang). After b_1 is produced, T gets

the second input string X_2, continues its work (using fresh random bits) and may produce second output bit b_2, etc. In other words, we write $X_1\#X_2\#\ldots\#X_n$ on the input tape, but T cannot get access to X_i before it produces $i-1$ output bits $b_1,\ldots,b_{i-1}$.

For a given T consider a function M_T: Let $M_T(X_1,b_1,\ldots,X_n,b_n)$ be the probability that T outputs $b_1,\ldots,b_n$ getting $X_1,X_2,\ldots,X_n$ as input (with restrictions described above). We extend the function M_T to the sequences of odd length: $M_T(X_1,b_1,\ldots,X_n,b_n,X_{n+1})$ is equal to $M_T(X_1,b_1,\ldots,X_n,b_n)$. We let $M_T(\Lambda)=1$. It is easy to see that if T never hangs (or hangs with zero probability), then M_T is an on-line probability distribution. In general, M_T is an on-line semimeasure in the sense of the following

Definition. An *on-line semimeasure* is a function M that maps tree vertices to non-negative reals such that $M(\Lambda)=1$, $M(X_1,b_1,\ldots,X_i,b_i,X_{i+1}) = M(X_1,b_1,\ldots,X_i,b_i)$ (on vertices where no random choice is made, the function propagates without change), and the inequality $M(X_1,b_1,\ldots,X_i,b_i,X_{i+1}) \geq M(X_1,b_1,\ldots,X_i,b_i,X_{i+1},0)+M(X_1,b_1,\ldots,X_i,b_i,X_{i+1},1)$ holds. (We have replaced "=" by "$\geq$" in the definition of an on-line probability distribution.)

It is easy to see that semimeasure M_T that corresponds to a probabilistic machine T of described type is a *lower semicomputable* function, i.e., there is an algorithm that gets its input and produces an increasing sequence of rational numbers that converges to the value of the function.

Theorem 3. *Every lower semicomputable on-line semimeasure corresponds to some probabilistic machine.*

Proof is similar to the off-line case. For an on-line semimeasure M we perform a "memory allocation", so that for each finite sequence $X_1,b_1,\ldots$ an open subset of $[0,1]$ that has measure $M(X_1,b_1,\ldots)$ is allocated. Adding X_i to the end of the sequence does not change the set; adding bits 0 and 1 replaces the corresponding set by two its disjoint subsets. (Note that these subsets may depend not only on b_i, but also on X_i.) If M is lower semicomputable, these sets can be made uniformly effectively open. Then we consider a machine T that generates a uniformly distributed random real number $\alpha\in[0,1]$ bit by bit and generates $T(X_1,b_1,\ldots,X_i)=b_i$ if the effectively open set that corresponds to $X_1,b_1,\ldots,X_i,b_i$ contains α. $\square$

Theorem 4. *There exists the largest (up to $O(1)$-factor) lower semicomputable on-line semimeasure.*

Proof. Again we can use standard trick: a universal machine first generates randomly a machine of described type in such a way that every machine appears with a positive probability, and then simulates this machine. $\square$

We call this maximal semimeasure an *on-line a priori probability* and denote it by $A(X_1,b_1,\ldots,X_n,b_n)$. (If all X_i are empty strings, we get a standard a priori probability on a binary tree.) Minus logarithm of this semimeasure is called *on-line a priori complexity* and denoted by $KA(X_1\to b_1;X_2\to b_2;\ldots;X_n\to b_n)$.

4 Relations between *KR* and *KA*

Now, when two complexities KA and KR are defined in the on-line framework, one may ask how they are connected. Their off-line versions are close to each other: it is known that $KR(x) \le KA(x) \le KR(x) + 2\log KR(x)$ (up to $O(1)$-terms) for all binary strings x.

These inequalities remain true for the on-line setting (with the same $O(1)$-precision):

Theorem 5. $KR(\ldots) \le KA(\ldots) \le KR(\ldots) + 2\log KR(\ldots)$; *here* "$\ldots$" *stands for* $X_1 \to b_1; X_2 \to b_2; \ldots; X_n \to b_n$.

Proof of the second inequality goes in the same way as usual; we consider a randomized algorithm that chooses machine number i with probability $1/i^2$.

The first inequality needs more care, since in the on-line case we are more restricted and need to ensure that programs are indeed on-line and do not refer to the inputs that are not yet available.

We need to allocate 2^n strings of length n to objects that have KA-complexity less than n (=have a priori probability greater than 2^{-n}). We do it inductively (first for $X_1 \to b_1$, then for $X_2 \to b_2$, etc.) and ensure a stronger requirement: if a priori probability of some object exceeds $k2^{-n}$ for some k, then there are at least k different programs of length n allocated to this object.

So we start looking at the approximations (from below) to the (a priori) probabilities of $X_1 \to 0$ and $X_1 \to 1$ (independently for each n and each X_1); when probability of $X_1 \to b_1$ exceeds $k2^{-n}$, we allocate a new (kth) program of length n that transforms X_1 to b_1. On top of this process we look at the approximations to a priori probabilities of $X_1 \to b_1; X_2 \to b_2$ and add new programs that map X_2 to b_2 among the programs that mapped X_1 to b_1; we have enough programs for that since $M(X_1, b_1, X_2, 1) + M(X_1, b_1, X_2, 0) \le M(X_1, b_1)$, so if k_1 programs are needed for the first term and k_0 are needed for the second, then there are already $k_0 + k_1$ programs allocated to $X_1 \to b_1$ to choose from. On top of that, we allocate programs for $X_1 \to b_1; X_2 \to b_2; X_3 \to b_3$ etc. □

5 On-line randomness

Let us return to the "real-life" example and make it less real: imagine that we observe an *infinite* sequence of games and (for every i) know the bit b_i produced by the referee when ith game starts and the string X_i that is known before ith game. There are cases when we intuitively reject the fair coin assumption. Can we make the intuition more formal and define a notion "in the sequence $X_1, b_1, X_2, b_2, X_3, b_3, \ldots$ the bits $b_1, b_2, \ldots$ are random"? For the off-line case the most popular notion is called *Martin-Löf randomness* (ML-randomness; see [3, 6] for details). Now we want to extend it to the on-line setting.

Assume that a computable on-line probability distribution P (on the infinite tree) is fixed. Martin-Löf definition starts with a notion of an "effectively null" set. Adapting this definition to on-line setting, we need to remember that probability of events is now undefined; moreover, the notion of upper probability (that replaces it) has been defined for finite case only.

Consider the space Π of all (infinite) sequences $X_1, b_1, X_2, b_2, \ldots$. A *cone* in this set is a set of all sequences with given finite prefix.

Definition. Let U be a finite union of cones. Then the *upper probability* of U with respect to P is defined as the upper probability of the corresponding event in the finite part of the tree (large enough to contain all the roots of the cones).

(It is easy to see that this probability does not change if we increase the size of the finite part of the tree. The upper probability is monotone with respect to set inclusion.)

Then we can define an on-line version of null sets.

Definition. A set $Z \subset \Pi$ is an *on-line null set* if for any $\varepsilon > 0$ there exists a sequence of cones such that: (1) the union of cones covers Z; (2) the union of any finite number of these cones has upper probability less than ε.

Martin-Löf definition of randomness deals with effectively null sets, so our next step is to define them in an on-line setting.

Definition. A set Z is an *on-line effectively null set* if there exists an algorithm that for any given rational $\varepsilon > 0$ generates a sequence of vertices such that the corresponding cones cover Z and the union of any finite number of these cones has upper probability less than ε. (Note that we require the upper probability of the union of the cones to be small, not the sum of upper probabilities of the cones. This difference matters since upper probability, unlike classical probability, is not additive.)

Theorem 6. *There exists an on-line effectively null set that contains every other on-line effectively null set.*

Proof is similar to the off-line case. Having any algorithm that given rational $\varepsilon > 0$ generates sequences of vertices, we can "trim" it so that the union of any finite number of generated cones has upper probability less than ε. (Indeed, for a computable on-line measure the upper probability of the finite union of cones is computable, and we may quarantine new strings until they are cleared.) So we can enumerate all the algorithms that satisfy these restrictions and then take the union of corresponding on-line effectively null sets (combining covers of size $\varepsilon/2$, $\varepsilon/4$ etc. to get the cover of size ε; here we use the subadditivity of upper probability). $\square$

Now we can give a

Definition. Bits $b_1, b_2, \ldots$ are *on-line ML-random* in a sequence $\omega = X_1, b_1, X_2, b_2, \ldots$ if ω does not belong to the maximal on-line effectively null set.

In other words, $b_1, b_2, \ldots$ are not random in ω if and only if $\{\omega\}$ is an on-line effectively null set (if and only if some on-line effectively null set contains ω).

6 On-line randomness criterion

A classical Levin – Schnorr theorem gives a criterion of randomness in terms of complexity (in particular, a priori complexity KA) or supermartingales. Similar criterion exists for the on-line version.

Theorem 7. (Levin – Schnorr theorem, on-line version). *Assume that a computable on-line probability distribution P is fixed. Bits $b_1, b_2, \ldots$ are on-line*

ML-random (with respect to P) in a sequence $\omega = X_1, b_1, X_2, b_2, \ldots$ if and only if $KA(X_1 \to b_1; \ldots, X_n \to b_n) \geq -\log_2 P(X_1, b_1, \ldots, X_n, b_n) - c$ for some c and all n.

Recalling that KA is the minus logarithm of a priori probability A, we can reformulate the criterion: bits b_i are random in $(X_1, b_1, X_2, b_2, \ldots)$ if and only if the ratio $A(X_1, b_1, \ldots, X_n, b_n)/P(X_1, b_1, \ldots, X_n, b_n)$ has a constant upper bound. (Note that A is the maximal semimeasure and P is a measure (and therefore a semimeasure), so this ratio always has a positive lower bound.)

One more reformulation of the same result uses on-line supermartingales. As we have noted, on-line (super)martingales with respect to P are ratios Q/P where Q is an on-line (semi)measure. It allows us to reformulate the criterion as follows: bits b_i are random in a sequence $\omega = X_1, b_1, X_2, b_2, \ldots$ if and only if any lower semicomputable supermartingale is bounded on prefixes of ω.

For a more advanced (and more difficult to prove) version of Theorem 7, see [8].

Proof of the on-line version of Levin – Schnorr randomness criterion follows the off-line argument with some changes: we have to be more careful since we have to deal with upper probability instead of an (additive) measure.

First, we have to show that if a sequence is not random, then the ratio A/P is unbounded on its prefixes. Since A is maximal, it is enough to construct some lower semicomputable semimeasure Q such that Q/P is unbounded.

Lemma. *Assume that some algorithm enumerates a sequence of cones C_1, $C_2, \ldots$ and the upper probability of the union $C_1 \cup \ldots \cup C_N$ is less than ε for some rational $\varepsilon > 0$ and for all N. Knowing this algorithm and ε, we can construct a lower semicomputable semimeasure S that exceeds P/ε at any finite sequence that belongs to one of the cones.*

Proof of the Lemma. For any vertex v let us consider the cone $C(v)$ with root v and for any N let us compute the upper probability of the intersection $C(v) \cap (C_1 \cup C_2 \cup \ldots \cup C_N)$. Since P is computable, doing this for $N = 1, 2, \ldots$, we get an increasing computable sequence of computable reals, and its limit is lower semicomputable. Let $S(v)$ be this limit divided by ε. This limit is not technically a semimeasure since $S(X_1, b_1, \ldots, X_i, b_i)$ can be bigger than $S(X_1, b_1, \ldots, X_i, b_i, X_{i+1})$. But if we increase the latter by letting $S(X_1, b_1, \ldots, X_i, b_i, X_{i+1}) := S(X_1, b_1, \ldots, X_i, b_i)$, and also let $S(\Lambda) = 1$, we do get a lower semicomputable semimeasure that satisfies the requirements of the Lemma. □

Now we can finish the proof of the first part of Levin – Schnorr on-line randomness criterion. Let $\varepsilon_n = 2^{-2n}$. Since ω belongs to an on-line effectively null set, we can get a sequence of cones with upper probability bounded by ε_n; applying the Lemma to it, we get a lower semicomputable semimeasure S_n that exceeds $2^{2n}P$ on any vertex that belongs to one of the cones. Then the sum $S = \sum_n 2^{-n} S_n$ exceeds $2^n P$ on any vertex that belongs to some of the cones generated for ε_n. By assumption ω has a prefix of this type for every n, so S/P is unbounded on prefixes of ω.

In remains to prove the second part of the theorem. For any lower semicomputable semimeasure S we have to show that the set of all sequences ω such that S/P is unbounded on the prefixes of ω is an on-line effectively null set.

For a given $\varepsilon > 0$ let us consider all the vertices where S/P exceeds $1/\varepsilon$. They can be enumerated if ε is given since S is lower semicomputable and P is computable. We need to check that the upper probability of the union of any finite number of corresponding cones is less than ε. Indeed, while computing the costs inductively in a top-down fashion, the cost is upper-bounded by ε times the value of S in the vertex. (Induction base is guaranteed by the assumption: we start with vertices where S/P is greater that $1/\varepsilon$; the induction step works since S is a semimeasure and P is a measure.) $\square$

Remarks. **1**. In the off-line setting the similar construction almost gives a lower semicomputable measure (with one exception: the measure of the entire space may be less than 1) or, in other terms, a martingale whose initial amount is lower semicomputable. In the on-line setting it is no more true (at least for this construction), and we get a semimeasure (or supermartingale).

2. On the other hand, the proof gives more than we claimed: if a sequence is not random, then some lower semicomputable on-line supermartingale is not only unbounded but also tends to infinity. It implies that if some lower semicomputable on-line supermartingale is unbounded on some ω, then some *other* semicomputable on-line supermartingale tends to infinity on ω.

The notion of randomness of b_i in a sequence $X_1, b_1, X_2, b_2, \ldots$ lies in-between Martin-Löf randomness and Martin-Löf randomness with respect to an oracle. Indeed, it implies ML-randomness since we can consider semimeasures (supermartingales) that do not depend on X_i at all. On the other hand, each on-line supermartingale can be transformed into a supermartingale that uses the entire sequence $X_1, X_2, \ldots$ as an oracle (getting access not only to the past X_i, but also to the future ones). Both inclusions are strict for evident reasons: a ML-random sequence b_i is not on-line random if $X_i = b_i$; it is on-line random if $X_i = b_{i-1}$ but not random with oracle $X_1, X_2, \ldots$.

Other observations (the proof is straightforward):

Theorem 8.

(a) *If the sequence X_i is computable (or if X_i is a computable function of $X_1, b_1, \ldots, X_{i-1}, b_{i-1}$) then on-line randomness is equivalent to (standard) ML-randomness with respect to induced measure where X_i are fixed.*

(b) *Changing finitely many terms among b_i or X_i does not make a random sequence non-random or vice versa, assuming that all conditional probabilities are not zeros.*

(c) *The on-line random sequence remains on-line random if we replace X_i by some Y_i that is a computable function of $b_1, X_1, \ldots, b_{i-1}, X_i$.*

7 Muchnik's paradox

In this section we consider the case of fair coin (all conditional probabilities are equal to 1/2). Let $b_1, b_2, \ldots$ be a ML-random sequence. It is easy to see that

then the sequence $b_2, b_4, b_6, \ldots$ is on-line ML-random if $b_1, b_3, b_5 \ldots$ are used as external information. Indeed, any lower semicomputable on-line supermartingale can be transformed into a (lower semicomputable) off-line supermartingale that makes no bets on b_1, b_3, b_5 etc. For the same reason the sequence $b_1, b_3, b_5, \ldots$ is on-line random inside the sequence $\Lambda, b_1, b_2, b_3, \ldots$ (bits $b_2, b_4, \ldots$ are treated as external information).

One may naturally expect that the reverse is also true: if both odd and even bits are unpredictable (with all previous bits used as the external information), then the entire sequence should be random. Indeed, our intuition says that if the coin tossing is performed by two referees that alternate (each of them works every second day), and both referees do their job perfectly, the resulting sequence of bits should be also perfectly random.

This would make a nice on-line version of van Lambalgen theorem that says that if a sequence $b_1, b_3, b_5, \ldots$ is ML-random and at the same time $b_2, b_4, b_6, \ldots$ is ML-random with oracle $b_1, b_3, b_5, \ldots$, then the entire sequence $b_1, b_2, b_3, b_4, \ldots$ is ML-random.

We may note also that if we replace *semicomputable* supermartingales by *computable* supermartingales, the corresponding statement becomes true. Indeed, assume that a computable supermartingale S is unbounded on some sequence. We may assume without loss of generality that it is at least 1 on every sequence (just by adding 1). At each vertex it splits the current capital in computable proportions (since the ratio of two computable numbers separated from zero is uniformly computable). So we can consider two computable supermartingales S_1 and S_2; one does not make any bet on odd steps and follows the proportions of S at the even steps, the other does the opposite. Then the capital of S is the product of S_1 and S_2, so if S is unbounded on some sequence, then at least one of S_1 and S_2 is unbounded on it.

However, all these arguments do not make the desired statement true, as An. Muchnik [4] has shown. He showed that there is a sequence which is not ML-random but still both odd and even terms are on-line random. This construction is rather delicate and we do not explain it here.

8 Selection rules and on-line randomness

The classical definition of randomness (for the case of independent fair coin tossing) suggested by R. von Mises is based on selection rules: each subsequence that is selected by an "admissible selection rule" should have limit frequency 1/2. It can also be naturally transferred into the on-line framework.

In Mises – Church definition of randomness an admissible selection rule is a total computable function that can be applied to any binary string and produces one of two answers: S ("selected") or O ("observed"). An application of this rule to a sequence ω goes as follows: the value of selection rule on n-bit prefix of ω determines if the next bit should be selected or just observed. Another version that goes back to R.P. Daley considers partial functions as selection rules; if such

a function is undefined at some prefix of ω, then the selection process hangs and the selected subsequence is finite.

Both definitions (with total and partial selection rules) can be easily extended to the on-line framework. We just allow the selection rule to use the external information that is available at the moment (i.e., all previous values of X_i). It is easy to see that we get a weaker notion of randomness (compared to on-line Martin-Löf randomness with respect to the uniform Bernoulli on-line measure, i.e., the measure where all conditional probabilities are equal to 1/2). Moreover, the following is true:

Theorem 9. *Assume that bits b_i are on-line ML-random (with respect to the uniform Bernoulli on-line measure) in a sequence $X_1, b_1, X_2, b_2, \ldots$ and a (partial) selection rule S is given that is defined on all prefixes of this sequence and selects infinitely many bits $b_{i_1}, b_{i_2}, \ldots$ for increasing sequence of indices $i_1 < i_2 < \ldots$. Then the selected bits are on-line ML-random in a sequence $Y_1, b_{i_1}, Y_2, b_{i_2}, \ldots$ where Y_k is the prefix of the original sequence (both X_i and b_i) that precedes b_{i_k}.*

(This implies, as we have said, that b_{i_k} are ML-random and therefore satisfy the strong law of large numbers.)

Proof. Indeed, a semicomputable on-line supermartingale U that plays with b_{i_k} using information that precedes them can be transformed into a supermartingale that deals with the original sequence. While selection rule is not yet defined, the supermartingale does not bet anything; if selection rule says "observe", the supermartingale keeps the same capital not making any bets; if the selection rule says "select", the supermartingale follows U. □

9 Randomness with respect to classes of measures

On-line randomness is connected with the notion of randomness with respect to effectively closed classes of measures; this notion was introduced by Levin [2] (see [1] for the detailed exposition).

Consider some class $\mathcal{S}$ of measures (probability distributions) on the Cantor space Ω of infinite sequences of zeros and ones. A set $Z \subset \Omega$ is called a *$\mathcal{S}$-null* set if $P(Z) = 0$ for every $P \in \mathcal{S}$. The effective version of this definition: $Z \subset \Omega$ is an *effectively $\mathcal{S}$-null set* if there is an algorithm that given rational $\varepsilon > 0$ produces a sequence of intervals $I_1, I_2, \ldots \subset \Omega$ that covers Z such that $P(I_1 \cup I_2 \cup \ldots) \leq \varepsilon$ for every $P \in \mathcal{S}$.

Levin noted that for an effectively closed class $\mathcal{S}$ the union of all effectively $\mathcal{S}$-null set is an effectively $\mathcal{S}$-null set. (An effectively closed class $\mathcal{S}$ is defined in a natural way: we consider a topology on the set of all measures where basic open set are the sets $\mathcal{U}_{x,p,q} = \{P | P(\Omega_x) \in (p,q)\}$ for all binary strings x and all intervals (p,q) with rational endpoints, as well as their finite intersections. A class $\mathcal{S}$ is effectively closed if its complement is a union of a computable sequence of basic open sets.) Then we say that a sequence ω is *random with respect to $\mathcal{S}$* if it does not belong to largest effectively $\mathcal{S}$-null set.

Now we can relate the definition of the on-line randomness to Levin's definition. Consider the class $\mathcal{S}$ of measures P that are consistent with the given online distribution, i.e., have conditional probabilities $1/2$ at odd layers: $P(b_1b_2\dots b_{2n+1}0) = P(b_1b_2\dots b_{2n+1}1)$ for every bit string $b_1\dots b_{2n+1}$ of odd length. It is easy to see that randomness with respect to $\mathcal{S}$ is equivalent to the on-line randomness. (The strings X_i in the definition of the on-line randomness are replaced by bits for simplicity, but this is not essential.)

10 On-line and prequential randomness

The classical definition of ML-randomness with respect to a computable measure P does not really use the ordering of the bits in the sequence: if we perform a computable permutation of a sequence and change the measure accordingly, then the sequence remains random. However, for the case when the bits are generated sequentially, the standard Martin-Löf definition does not look natural. Indeed, according to the definition, we need to know the measure of any interval in the Cantor space, including the intervals that can not contain the sequence in question. If our sequence starts with, say, 01001, it seems that the value $P(\Gamma_{100})$, where Γ_z is a set of all continuations of a finite string z, should be totally irrelevant. The only thing that seems to be relevant in this sequential setting (when a sequence ω is generated bit by bit from left to right) is the values of conditional probabilities of 0 and 1 after $\omega_1\omega_2\dots\omega_{n-1}$, i.e., the ratios $P(\Gamma_{x0})/P(\Gamma_x)$ and $P(\Gamma_{x1})/P(\Gamma_x)$ for all x that are prefixes of ω.

This intuition is supported by the following observation: let P and P' be two computable measures that have the same conditional probabilities along some sequence ω (as defined above). If ω is ML-random with respect to P, then it is ML-random with respect to P'. (This is an immediate corollary of Levin – Schnorr randomness criterion.)

So we come to a natural question: can we give a natural definition of randomness in such a way that only conditional probabilities along ω are used in the definition?

Imagine an adjustable random bit generator, i.e., a device for generating random bits with prescribed probabilities. Such a device gets some real number $p \in [0,1]$ as an input and generates a bit $b \in \{0,1\}$ claiming that this bit is a result of a "fresh" random experiment (independent of all the past information) with probability of success p. Then we can send the next probability value to the device, it generates the next random bit, and so on.

Assume that we observe the behavior of the device and have its work recorded. The record (protocol) is a sequence $p_1, b_1, p_2, b_2, \dots$ where $p_i \in [0,1]$ and $b_i \in \{0,1\}$. Can we say whether this random bit generator works properly or not looking at this protocol? Our intuition says that there are at least some cases when we don't trust such a generator. For example, if all p_i are greater than $1/2$ but vast majority of b_i are zeros, it is clear that something is wrong with the device. Similarly, if all p_i are, say, between $1/3$ and $2/3$ while

$b_1b_2b_3\ldots = 010101\ldots$ (alternating zeros and ones), again we would not trust the generator.

Let us restrict ourselves to the case when all p_i are rational numbers in $(0,1)$. Then we can define the randomness of the sequence $p_1, b_1, p_2, b_2, \ldots$ in the following way. Consider an on-line probability distribution on sequences $X_1, b_1, X_2, b_2, \ldots$ where strings X_i are identified with rational numbers p_i in $(0,1)$ using some computable one-to-one correspondence, and the conditional probability $\Pr[b_i = 1|X_1, b_1, \ldots, X_i]$ equals p_i (where p_i is the rational number that corresponds to X_i). This on-line probability reflects our intuition: the probabilities p_i (strings X_i) are chosen in an arbitrary way, and the following bit b_i should follow the declared distribution.

Theorem 10. *Let p be a computable function on binary strings with positive rational values that determines a measure, i.e., $p(b_1\ldots b_n) = p(b_1\ldots b_n0) + p(b_1\ldots b_n1)$. Then a sequence $b_1b_2\ldots$ is Martin-Löf random with respect to this measure if and only if the bits b_i are on-line random in a sequence $p_1, b_1, p_2, b_2, \ldots$ where p_i is a conditional probability of $b_i = 1$ after the prefix $b_1\ldots b_{i-1}$, i.e., the ratio $p(b_1\ldots b_{i-1}1)/p(b_1\ldots b_{i-1})$.*

Proof. This is a direct consequence of the supermartingale criterion. Assume that the sequence $b_1b_2\ldots$ is not random with respect to measure p. Then there exists a lower semicomputable supermartingale with respect to p that is unbounded on the prefixes of this sequence. This supermartingale can be extended to an on-line lower semicomputable supermartingale: we let it to be zero on finite sequence $p_1, b_1, \ldots, p_i, b_i$ where one of p_j differs from conditional probability of 1 after $b_1\ldots b_{j-1}$ according to p. Therefore the sequence $p_1, b_1, p_2, b_2, \ldots$ is not on-line random.

On the other hand, if the sequence $p_1, b_1, \ldots$ is not on-line random, there exists a lower semicomputable on-line supermartingale that is unbounded on this sequence. The restriction of this supermartingale on the subtree that contains only the vertices compatible with p, is a lower semicomputable on the binary tree with respect to measure p. This martingale is unbounded on $b_1b_2\ldots$, therefore this sequence is not Martin-Löf random with respect to measure p. □

Remarks. 1. We restrict our attention to the simplest case where the measure has rational values and is computable as a rational-valued function. More general definition is analyzed in [8].

2. Note that probabilities p_i play a double role in this definition. First, they determine the coefficients in the definition of on-line supermartingale; this is their "primary" role. However, they are also source of information that can be used in the computation of this supermartingale. This was not important, since if p is a computable rational-valued measure, then the conditional probabilities can be computed from other available information.

In fact, our randomness intuition is quite contradictory here. Imagine that, for example, p_i are rational numbers that converge very fast to $1/2$, e.g., $p_i = 1/2 + c_i/2^{2^i}$, where c_i is equal either to 0 or to 1. Since the convergence is very fast, we would naturally expect that randomness would be the same as for the uniform Bernoulli measure (independently of c_i). On the other hand, if we watch

the random number generator of the type described and observe that generated bit b_i is always equal to c_i, this compromises the fairness of the generator. Our definition follows the second direction: b_i that is equal to c_i is *not* random.

Acknowledgements

We thank the participants of the Kolmogorov seminar (Moscow), Workshop on effective randomness (Chicago University, 2007) and Workshop on game-theoretic probability and related topics (Tokyo University, 2008), where some of these results were presented.

This work was partly supported by EPSRC grant EP/F002998/1, Sycomore ANR grant and RFBR grants 05-01-02803-CNRS-a, 06-01-00122-a. Alexey Chernov is grateful for support to J. Schmidhuber and IDSIA (Lugano, Switzerland) where part of the work was done under J. Schmidhuber's SNF grant 200021-113364.

References

1. Gacs, P., Lecture notes on descriptional complexity and randomness, see `http://www.cs.bu.edu/faculty/gacs/papers/ait-notes.pdf`
2. Levin, L.A., Uniform tests of randomness, *Soviet Math. Dokl.*, **17**(2), p. 337–340 (1976).
3. Li, M., and Vitányi, P., *An Introduction to Kolmogorov Complexity and Its Applications*, Second edition, New York: Springer, 1997.
4. Muchnik, An. A. (recorded by A. Chernov, A. Shen), *Algorithmic randomness and splitting of supermartingales*, `arxiv.org:0807.3156`
5. Shen, A., *Algorithmic information theory and Kolmogorov complexity*, Technical Report 2000-034. Uppsala Universitet publication, available online at: `http://www.it.uu.se/research/publications/reports/2000-034`.
6. Uspensky, V. A., Semenov, A. L., and Shen, A., Can an individual sequence of zeros and ones be random? *Russian Mathematics Surveys*, **45**, 121–189 (1990).
7. Shafer, G., and Vovk, V., *Probability and Finance: It's only a Game*, New York, Wiley, 2001.
8. Vovk, V., and Shen, A., Prequential randomness. Submitted to ALT 2008.