Algorithm that constructs two sequence-set betting strategies that predict all compressible sequences

Tomislav Petrović
01tomislav.petrovic@gmail.com

Abstract. A new type of betting game that characterizes Martin-Löf randomness is introduced, the sequence-set betting game. This betting game can be compared to martingale processes of Hitchcock and Lutz as well as the nonmonotonic betting game. In a nonmonotonic betting game, a player successively bets against an infinite binary sequence by choosing the position of the bit in the sequence and placing a bet on the value of the bit in that position. The choice of position divides the set of sequences into two disjoint, clopen sets of equal measure, where sequences in one set have bit 1 in the chosen position and sequences in the other set have bit 0 in that position. The sequence-set betting is a generalization of nonmonotonic betting. In a sequence-set betting game a player is allowed to divide the set of sequences into two disjoint, clopen sets in any way as long as the sets have equal measure, and then places a bet on one of those sets. A further generalization of sequence-set betting, where a player is allowed to divide the sequences into two disjoint, clopen sets of unequal measure and bet on one of them is shown to be equivalent to martingale processes. The player is said to succeed on a sequence if her gain of capital through successive betting is unbounded. Whether the set of sequences on which no computable nonmonotonic betting strategy succeeds is the same as the set of Martin-Löf random sequences is considered to be a major open question in the field of algorithmic randomness. In contrast to martingale processes, and similarly to computable nonmonotonic betting strategies, no single computable sequence-set betting strategy succeeds on all non-Martin-Löf random sequences. The main result of this paper shows that there are two total computable sequence-set betting strategies such that on each non-Martin-Löf random sequence at least one of them succeeds.

1 Introduction

In 1998, Muchnik, Semenov, and Uspensky [12] introduced the concept of a nonmonotonic betting game (in their paper it’s simply called a game). Informally, in a nonmonotonic betting game the player starts with a finite amount of capital and bets successively on bits in an infinite binary sequence by choosing the position of the bit in the sequence and placing some amount of capital on one of the two possible values of that bit. The game is fair in the sense that if
the player’s guess about the value of the bit is correct, the wagered amount is doubled, otherwise the wagered amount is lost. A nonmonotonic betting strategy describes which positions and bets the player chooses when betting against infinite binary sequences. A betting strategy is said to succeed on a sequence if it earns a player an infinite amount of capital when betting against that sequence. A sequence on which no computable nonmonotonic betting strategy succeeds is called Kolmogorov-Loveland random. The question whether the set of Kolmogorov-Loveland random sequences is the same as the set of Martin-Löf random sequences is considered a major open problem in the field of algorithmic randomness [9, 5, 3, 4, 8, 7, 6, 12]. In Section 2 a sequence-set betting game is introduced and it is compared to a nonmonotonic betting game and the martingale processes of Hitchcock and Lutz [1]. The sequence-set betting is a generalized form of nonmonotonic betting. Informally, in a sequence-set betting game the player also starts with a finite amount of capital. However, instead of choosing the position of the bit in the sequence, the player can divide the set of sequences any way she likes into two disjoint, clopen sets of equal measure and wager some amount of capital on one of those sets. If the player guessed correctly and the sequence is in the set on which the player bets, the wagered amount is doubled, otherwise it is lost. The sequence-set betting can be further generalized by allowing the player to divide sequences into sets of unequal measure. This generalized form of betting game is shown to be equivalent to martingale processes. By the result of Merkle, Mihailovic, and Slaman [2] there is a computable martingale process such that it succeeds on every non-Martin-Löf random sequence. By contrast, there is no computable sequence-set betting strategy that succeeds on all non-Martin-Löf random sequences. In Section 3 it is shown that there are two total computable sequence-set betting strategies such that on each non-Martin-Löf random sequence at least one of them succeeds and that is the main result of this paper.

1.1 Notation

The notation used is fairly standard, for unexplained terms refer to the textbooks [11, 10, 13]. We consider finite binary sequences or words, the empty word is denoted by \( \varepsilon \). The set of all words is denoted by \( \{0, 1\}^* \). Sequence is an infinite binary sequence, we denote it with \( \alpha \). The set of all sequences is referred to as Cantor space and is denoted by \( \{0, 1\}^\infty \). The set of all sequences that have a word \( v \) as a prefix is called a cylinder generated by \( v \) and is denoted \( v\{0, 1\}^\infty \). A union of a finite number of disjoint cylinders is called a clopen set. The Lebesgue measure of a set of sequences is denoted by \( \lambda \).

2 Comparison of betting games

A betting game is defined by a set of rules the player must adhere to when betting against an infinite binary sequence. For all of the betting games considered in this paper the player iteratively places bets on sets of sequences. If the sequence
the player is betting against, \( \alpha \), is in the set on which the player has placed a bet, the total amount of capital of the player is increased, whereas if the \( \alpha \) is not in the set the wagered amount is lost. The player succeeds on a sequence (predicts the sequence) if the gain of capital is unlimited while betting successively against the sequence. In each iteration of betting the player starts with some amount of capital and some set of sequences. The first iteration is started with the set of all sequences, \( \{0, 1\}^\infty \), and some initial amount of capital. The player divides the set of sequences into two sets and wagers some part of capital on one of them. In the next iteration the player starts with the set that contains \( \alpha \), and the amount of capital that is available to betting depends on the player’s bet in the previous iteration.

**Definition 1.** A sequence-partition function \( \text{sp} \) is a possibly partial function from words to clopen sets of sequences such that \( \text{sp}(\epsilon) = \{0, 1\}^\infty \) and for each word \( v \) either \( \text{sp}(v0) \) and \( \text{sp}(v1) \) are both undefined or \( \text{sp}(v0) \) and \( \text{sp}(v1) \) are both defined and

\[
\text{sp}(v) = \text{sp}(v0) \cup \text{sp}(v1) \\
\text{sp}(v0) \cap \text{sp}(v1) = \{}
\]

A sequence-partition function defines the way the player divides the sets of sequences. The games we are considering differ in definitions of admissible sequence-partition functions. This is the most general definition, other games are obtained by adding restrictions to the sequence-partition function.

**Definition 2.** A mass function \( m \) is a nonnegative, possibly partial, function from words (finite binary sequences) to rationals such that for each word \( v \) either \( m(v0) \) and \( m(v1) \) are both undefined or \( m(v0) \) and \( m(v1) \) are both defined and \( m(v) = m(v0) + m(v1) \).

The mass function corresponds to the capital of player. If the player starts the iteration with the set \( \text{sp}(v) \), the capital \( c \) that the player has available for betting is \( c = m(v) / \lambda(\text{sp}(v)) \). The equation \( m(v) = m(v0) + m(v1) \) ensures the fairness of the betting game in the sense that the return of capital for the wagered amount is proportional to the measure of the set \( \text{sp}(v) \) divided by the measure of the set the player bets on. Let’s say the player wagers some amount \( wg \) on \( \text{sp}(v0) \). If the sequence is not in \( \text{sp}(v0) \) then the next iteration is started with the set \( \text{sp}(v1) \) and capital \( c - wg \), on the other hand if the sequence is in \( \text{sp}(v0) \), then the next iteration is started with \( \text{sp}(v0) \) and capital \( c - wg + wg \lambda(\text{sp}(v)) / \lambda(\text{sp}(v0)) \).

**Definition 3.** A betting strategy \( R \) is a pair of functions \( R = (m, \text{sp}) \) where \( m \) is a mass function and \( \text{sp} \) is a sequence-partition function. A strategy \( R \) is total if \( m \) and \( \text{sp} \) are defined for every word and it is computable if both functions are computable. Strategy \( R \) succeeds on sequence \( \alpha \) if there is an infinite sequence of words \( v_1, v_2, \ldots \) such that \( \alpha \in \text{sp}(v_i) \) and \( m(v_i) / \lambda(\text{sp}(v_i)) > i \).

A betting strategy formalizes the concept of player, it describes the bets the player places when betting against sequences. Betting strategies can be compared to martingale processes of Hitchcock and Lutz.
Definition 4. A **martingale process** $p$ is a nonnegative, total function from words to reals. Two words have the same capital history, $v \approx_p w$, for short, if both words have the same length, and we have $p(v') = p(w')$ for any two prefixes $v'$ of $v$ and $w'$ of $w$ that have the same length. The relation $\approx_p$ is an equivalence relation on words and is called $p$-equivalence. Function $p$ is such that it satisfies the fairness condition 2 \[ \sum_{v : v \approx_p w} p(v) = \sum_{v : v \approx_p w} [p(v_0) + p(v_1)] \]
The martingale process $p$ succeeds on a sequence $\alpha$ if \( \limsup_{k \to +\infty} (p(v_k)) = +\infty \) where $v_k$ is the prefix of $\alpha$ of length $k$.

Remark 1. For a real-valued martingale process $p$ there is a rational-valued martingale process $q$ such that if two words are $p$-equivalent then they are $q$-equivalent and \( \frac{1}{2}q(v) < q(w) < q(v) \) for each word $v$. Martingale process $q$ succeeds on exactly the same sequences as $p$.

**Proposition 1.** A computable martingale process $p$ succeeds on a set of sequences $E$ if and only if there is a computable betting strategy $R$ such that $R$ succeeds on the sequences in $E$.

**Proof.** (IF) We define a martingale process $p$ from a betting strategy $R$. Set $p(\epsilon) = m(\epsilon)$. Assume that for a word $w$ the set of $p$-equivalent words $V = \{ v : v \approx_p w \}$ has a subset $V' \subseteq V$ that coincides with a sequence-partition and mass for word $r$ in strategy $R$, in the sense that $sp(r) = \bigcup_{v \in V'} v\{0,1\}^\infty$ and that $m(v) = m(r)/\lambda(sp(r)) = p(w)$. Until the computation of $sp(r0)$, $sp(r1)$, $m(r0)$, $m(r1)$ is finished, iteratively increase length $l$ and for words $v$ that have a prefix in $V$, and whose length is $l$ set $p(v) = p(w)$. If the strategy never finishes the computation then all words $v$ that have a prefix in $V$ will have $p(v) = p(w)$. On the other hand if the strategy at some point does finish we have set the value $p(v) = p(w)$ for all words $v$ that have a prefix in $V$ and whose length is less than or equal to some $l$. Let’s say that the measure of the smallest cylinders in $sp(r0)$, $sp(r1)$ is $2^{-l'}$. Assume without loss of generality that $l \geq l'$. Denote with $L$ the set of words that have a prefix in $V$ and whose length is $l + 1$. For words $v \in L$ such that $v$ doesn’t have a prefix in $V'$ set $p(v) = p(w)$. For words $v \in L$ such that $v\{0,1\}^\infty \subseteq sp(r0)$ set $p(v) = m(r0)/\lambda(sp(r0))$. For words $v \in L$ such that $v\{0,1\}^\infty \subseteq sp(r1)$ set $p(v) = m(r1)/\lambda(sp(r1))$. We have that $p$ is a martingale process and that $p$ succeeds on exactly the same sequences as strategy $R$.

(ONLY IF): We define strategy $R$ from a martingale process $p$. From Remark 1 we can assume that $p$ is a rational-valued function. Set $m(\epsilon) = p(\epsilon)$. Assume that for some word $r$ the sequence-partition and mass of strategy $R$ coincide with the set of words $p$-equivalent to some word $w$, $V = \{ v : v \approx_p w \}$, in a sense that $sp(r) = \bigcup_{v \in V} v\{0,1\}^\infty$ and that $m(r)/\lambda(sp(r)) = p(w)$. Compute $p$ for words of increasing length that have a prefix in $V$ until word $v$ is found such that $p(v) \neq p(w)$. If there is no word $v$ that has a prefix in $V$ and has value $p(v) \neq p(w)$ then the functions $m$ and $sp$ are undefined for words $r0$, $r1$. On the other hand if there is such word $v$ then for words of length $l(v)$ which have a prefix in $V$ there
is a finite number of equivalence classes $V_1, \ldots, V_n$ with respect to martingale process $p$. We set the $sp(r1), m(r1)$ so that the word $r1$ coincides with $V_1$, then set $sp(r01), m(r01)$ so that $r01$ coincides with $V_2$, and so on. We have that for $i < n$ the word $r0^{i-1}1$ coincides with $V_i$. We have that $sp(r0^{n-1})$ contains the sequences that have a prefix in $V_n$ and due to the fairness condition of martingale processes we have that $m(r0^{n-1})/\lambda(sp(r0^{n-1})) = p(w'), \ w' \in V_n$. Therefore the strategy $R$ succeeds on exactly the sequences on which the martingale process succeeds.

Merkle, Mihailovic, and Slaman have shown in [2] that there is a computable martingale process such that it succeeds on exactly the sequences which are not Martin-Löf random, therefore there is a computable betting strategy that succeeds on non-Martin-Löf random sequences.

**Definition 5.** A sequence-set function $S$ is a sequence-partition function with an additional requirement that for each word $v$ for which the $sp(v0)$ and $sp(v1)$ are defined we also have $\lambda(sp(v0)) = \lambda(sp(v1))$.

A sequence-set betting strategy is a betting strategy that uses a sequence-set function for the sequence-partition function and a sequence-set betting game is a game for which the admissible strategies are sequence-set betting strategies.

**Proposition 2.** For each computable sequence-set betting strategy there is a non-Martin-Löf random sequence on which the strategy does not succeed.

**Proof.** If the sequence-set betting strategy $R = (m, S)$ is partial then there is some word $v$ for which the set of sequences $S(v)$ is defined and $S$ is not defined for words $v0, v1$. The $S(v)$ contains some sequence that ends in an infinite number of zeroes, and this sequence is clearly not Martin-Löf random. On the other hand if the strategy is total then by choosing the words $v$ for which the capital $m(v)/\lambda(S(v))$ doesn’t increase, we can find an infinite series of words $v_1, v_2, \ldots$ such that $l(v_i) = i$ and $v_i$ is a prefix of $v_{i+1}$. Since the series of words and function $S$ are computable and since $\lambda(S(v_i)) = 2^{-l(v_i)}$ the $S(v_i)$ comprise a Martin-Löf test, the sequences in $\bigcap_{i=1}^{\infty} S(v_i)$ are not Martin-Löf random and the betting strategy doesn’t succeed on them.

In contrast to the general betting strategies there is no single computable sequence-set betting strategy that succeeds on all non-Martin-Löf random sequences, but as the main result of this paper shows, there are two total computable sequence-set betting strategies such that at least one of them succeeds on each non-Martin-Löf random sequence.

Next we give a standard definition of non-monotonic betting strategies by using a martingale instead of a mass function.

**Definition 6.** A martingale $d$ is a nonnegative, possibly partial, function from words to rationals such that for each word $v$ either $d(v0)$ and $d(v1)$ are both undefined or $d(v0)$ and $d(v1)$ are both defined and $2d(v) = d(v0) + d(v1)$. A martingale succeeds on a sequence $\alpha$ if $d$ is defined for all words that are prefixes of $\alpha$ and $\limsup_{k \to +\infty} d(v_k) = +\infty$ where $v_k$ is the prefix of $\alpha$ of length $k$. 
For a sequence-set betting strategy a martingale and the mass function are essentially the same concepts. From the condition \( \lambda(sp(v_0)) = \lambda(sp(v_1)) \) we have that \( \lambda(sp(v)) = 2^{-l(v)} \) and \( m(v)/\lambda(sp(v)) = m(v)2^{l(v)} \). Given a mass function \( m \) the function \( v \to m(v)2^{l(v)} \) is a martingale and given a martingale \( d \) the function \( v \to d(v)2^{-l(v)} \) is a mass function.

**Definition 7.** A scan rule \( \sigma \) is a possibly partial function from words to natural numbers such that for each word \( v \) if \( \sigma(v) \) is defined then for every prefix \( v' \) of \( v \) \( \sigma(v') \) is defined and \( \sigma(v') \neq \sigma(v) \).

A nonmonotonic betting strategy is a pair \( b = (d, \sigma) \) where \( d \) is a martingale and \( \sigma \) is a scan rule. Given an infinite binary sequence \( \alpha \) we define by induction a sequence of positions \( n_0, n_1, \ldots \) by

\[
\begin{align*}
    n_0 &= \sigma(\epsilon) \\
    n_{k+1} &= \sigma(\alpha(n_0)\alpha(n_1)\ldots\alpha(n_k)) \quad \text{for all } k \geq 0
\end{align*}
\]

and we say that \( b = (d, \sigma) \) succeeds on \( \alpha \) if the \( n_i \) are all defined and

\[
\limsup_{k \to +\infty} d(\alpha(n_0)\alpha(n_1)\ldots\alpha(n_k)) = +\infty
\]

We can define a sequence-set function that partitions the sequences in the same way as the scan rule of nonmonotonic betting strategies.

**Definition 8.** A nonmonotonic-partition function \( nm \) is a sequence-set function with an additional requirement that for each word \( v \) for which the \( S(v_0) \) and \( S(v_1) \) are defined there is some position \( i \) such that the sequences in \( S(v_0) \) have at position \( i \) bit 0 and the sequences in \( S(v_1) \) have at position \( i \) bit 1.

We can define a nonmonotonic betting strategy \( b = (d, \sigma) \) from a betting strategy \( R = (m, nm) \) where \( nm \) is a nonmonotonic-partition function. Set \( d(v) = m(v)2^{l(v)} \) and set \( \sigma(v) \) to be the position \( i \) for which the sequences in \( nm(v_0) \) have bit 0 and the sequences in \( nm(v_1) \) have bit 1. Conversely, we can define \( R = (m, nm) \) from \( b = (d, \sigma) \). Set \( m(v) = d(v)2^{-l(v)} \) and set \( nm(v_0) \) to be the set of sequences in \( nm(v) \) which have 0 at position \( \sigma(v) \) and \( nm(v_1) \) the sequences that have 1 at that position.

### 3 Algorithm that constructs two sequence-set betting strategies

**Theorem 1.** There are two total and computable sequence-set betting strategies such that on every non-Martin-Löf random sequence at least one of them succeeds.

#### 3.1 Algorithm overview

A sequence-set betting strategy \( R = (m, S) \) can be viewed as a set of triplets \((v, S(v), m(v))\) where \( v \) is a word, and the triplet for word \( v \) is in the set that
defines strategy \( R \) if both functions \( m \) and \( S \) are defined for word \( v \). We’ll call the triplet nodes and we identify them with words. For a node \( t \) we denote the sequence-set with \( S(t) \) and the mass with \( m(t) \). For nodes \( t, t' \) we say that \( t' \) is a child node of \( t \) and conversely \( t \) is a parent of \( t' \), if \( t \) is a prefix of \( t' \).

The algorithm constructs two sets of nodes, \( A \) and \( B \), that define two total and computable sequence-set betting strategies. A node in the set of nodes is a leaf node if there are no child nodes for that node in the set.

The algorithm iteratively adds nodes to both sets \( A \) and \( B \). After each iteration the sets \( A \) and \( B \) are such that the node \( t \) in one of the sets is either a leaf node or the set contains nodes \( t_0 \) and \( t_1 \).

The input for the algorithm instance, \( \text{alin} \), is a pair of nodes \((a, b)\), \( a \in A, b \in B \) such that their masses are greater than zero, \( m(a) > 0 \), \( m(b) > 0 \) and their sequence-sets are the same and consist of a single cylinder \( w \), \( S(a) = S(b) = w \). Initially the set \( A \) contains only one node \((\epsilon, \{0, 1\}^\infty, 1)\) and the set \( B \) contains only \((\epsilon, \{0, 1\}^\infty, 1)\). That is, the strategies initially haven’t placed any bets yet and they have the initial capital of 1. We start the algorithm by running an instance of the algorithm, \( \text{alin}(\epsilon, \epsilon) \). The algorithm instance calculates value \( k \) from the inputs and enumerates a disjoint set of cylinders \( P = \{p_1, p_2, \ldots\} \), \( p_i \subset w \), such that for each infinite sequence \( \alpha \) that is in the \( k \)-th component of the universal Martin-Löf test and \( \alpha \in w \) there is some \( p_i, \alpha \in p_i \). In each iteration \( i \) of the algorithm instance, \( p_i \) from \( P \) is enumerated and child nodes are added to the leaf nodes of both sets \( A \) and \( B \). At the end of iteration \( i \) we have a new set of leaf nodes in \( A \) and \( B \). There will be some set of cylinders \( W_i = \{w_1, \ldots, w_{z_i}\} \) such that \( p_i = \bigcup_{j=1}^{z_i} w_j \) and for each cylinder \( w_j \) there are two leaf nodes \( a' \) and \( b' \), one in each of the sets \( A \) and \( B \) such that their sequence-sets consist of the cylinder \( w_j \), \( S(a') = S(b') = w_j \), their masses are greater than zero \( m(a') > 0 \), \( m(b') > 0 \) and at least one of the nodes has at least doubled the initial capital of the algorithm instance,

\[
\frac{m(a')}{\lambda(w_j)} \geq \frac{2(m(a) + m(b))}{\lambda(w)} \quad \text{or} \quad \frac{m(b')}{\lambda(w_j)} \geq \frac{2(m(a) + m(b))}{\lambda(w)}
\]

The algorithm instance whose input was \((a, b)\) doesn’t add any more child nodes for these pairs of leaf nodes in further iterations. For these pairs of leaf nodes the algorithm instance starts other algorithm instances, \( \text{alin}(a', b') \). The remaining added leaf nodes in \( A \) and \( B \), the ones that don’t contain sequences from the so far enumerated cylinders from \( P \), in general have sequence-sets that consist of more then one cylinder and the sequence-sets of leaf nodes in one strategy have intersections with many of the leaf nodes sequence-sets in the other strategy.

We have that for the input node pair of the algorithm instance and their sequence-set, the cylinder \( w \), there is a set of cylinders \( W = \bigcup_{i=1}^{\infty} W_i \) such that for each infinite sequence \( \alpha \) which is non-Martin-Löf random and \( \alpha \in w \) there is a cylinder \( w' \in W \) and \( \alpha \in w' \). For each cylinder \( w' \) the algorithm instance in some iteration adds nodes to both strategies whose sequence-set is the cylinder
and at least one of them doubles the sum of capital that both strategies have on the initial cylinder $w$. In turn the node pairs whose sequence-set is the cylinder $w'$ are inputs for other algorithm instances. By induction we have that on each non-Martin-Löf random sequence at least one of the strategies has an unbounded gain of capital.

In the remainder of the paper the algorithm is described and it is shown that an algorithm instance can calculate $k$ from the inputs and that on each sequence in $P$ at least one of the strategies $A$, $B$ doubles the initial capital of algorithm instance.

### 3.2 Initialization step

In addition to the node’s sequence-set and mass the algorithm uses two other values, the reserved part of mass $me$ that ensures that the mass of a node is always greater than zero and the rational value $L$ which is the part of the mass used by the algorithm instance to increase the capital on infinite sequences in $P$. The input for an algorithm instance is a pair of nodes $(a_0, b_0)$ such that their sequence-sets are the cylinder $w_0$, $S(a_0) = S(b_0) = w_0$ and their masses $m(a_0), m(b_0)$ are greater than zero. Denote with $m_0$ the half of the smaller of the input masses, $m_0 = min(m(a_0), m(b_0))/2$. We set the $ms$ of the initial nodes to be the value $m_0$, and we set the value $L$ to be zero since we haven’t yet enumerated any of the cylinders in $P$. The $me$ part of mass is distributed evenly among the child nodes. For all nodes $a \in A$, $b \in B$ added by the algorithm instance, except the nodes for which we start other algorithm instances, we have $me(a) = \frac{\lambda(S(a))}{\lambda(w_0)}me(a_0)$ and $me(b) = \frac{\lambda(S(b))}{\lambda(w_0)}me(b_0)$. The capital gained on the infinite sequences in $P$ will be at least $c = \frac{2(m(a_0)+m(b_0))}{\lambda(w_0)}$ for at least one of the strategies $A$, $B$. The set $P$ is a disjoint set of cylinders that contain all infinite sequences in the $k$-th component of the universal Martin-Löf test. Denote $\text{const} = \frac{m_0^2}{\lambda(w_0)4c^2(1+cs)}$ where $cs$ is some predetermined constant $0 < cs < 1$. We choose the value $k$ such that the measure of the enumerated set $P$ is

$$\lambda(P) < \text{const}$$

We set the number of iteration $n := 0$ and proceed to the first iteration of the algorithm instance.

### 3.3 Iteration of the algorithm instance

**Definition 9.** For a set of nodes $T$ define the set of leaf nodes as $LN(T)$

$$LN(T) = \{ t : t \in T, \forall t' \in T, t \neq t' \}$$

At the beginning of the iteration we increment the number of the iteration $n := n + 1$ and enumerate the next cylinder $p_n \in P$. Denote with $P_n$ the union
of so far enumerated cylinders, \( P_n = \bigcup_{i=1}^{n} p_i \) where \( n \) is the number of the iteration and \( P_0 = \{ \} \). Denote with \( LT S_n^A \) the set of leaf nodes in \( A \) at the beginning of the iteration \( n \) such that \( a \) is a child node of the initial node \( a_0 \), \( a_0 \preceq a \) and the sequence-set doesn’t contain sequences from the cylinders enumerated in the previous iterations, \( S(a) \cap P_{n-1} = \{ \} \)

\[
LT S_n^A = \{ a : a \in LN(A), \ a_0 \preceq a, \ S(a) \cap P_{n-1} = \{ \} \}
\]
and analogously denote \( LT S_n^B \) as the set of leaf nodes \( b \) in \( B \) at the beginning of the iteration \( n \) such that \( b_0 \preceq b \) and \( S(b) \cap P_{n-1} = \{ \} \)

\[
LT S_n^B = \{ b : b \in LN(B), \ b_0 \preceq b, \ S(b) \cap P_{n-1} = \{ \} \}
\]
The algorithm instance during iteration \( n \) adds child nodes to nodes in sets \( LT S_n^A \) and \( LT S_n^B \). The remaining leaf nodes which are child nodes of the initial nodes \( a_0 \) and \( b_0 \) have sequence-sets which contain only sequences from the cylinders enumerated in the previous iterations and child nodes are added to them by other algorithm instances.

In the next step we add child nodes to all nodes \( a \in LT S_n^A \) for which there is some node \( b \in LT S_n^B \) such that the intersection of their sequence-sets contains both the sequences in the cylinder \( p_n \) and the sequences not in \( p_n \). Define \( LT N_n^A \) as the set of leaf nodes in \( A \) such that \( a_0 \preceq a \) and \( S(a) \cap P_{n-1} = \{ \} \) after these child nodes are added.

\[
LT N_n^A = \{ a : a \in LN(A), \ a_0 \preceq a, \ S(a) \cap P_{n-1} = \{ \} \}
\]
The added child nodes are such that for all the leaf nodes \( a \in LT N_n^A, \ b \in LT S_n^B \), the intersection of their sequence-sets contains only sequences from \( p_n \) or doesn’t contain sequences from \( p_n \).

\[
\forall a \in LT N_n^A \forall b \in LT S_n^B, \ S(a) \cap S(b) \subseteq p_n \lor S(a) \cap S(b) \cap p_n = \{ \} \quad (2)
\]

Next we add child nodes to the leaf nodes in \( LT N_n^A \) and \( LT S_n^B \). The added child nodes are such that the sequence-sets of the added leaf nodes contain either only sequences in \( p_n \) or only sequences not in \( p_n \). Denote the set of leaf nodes added to \( A \) that have only sequences in \( p_n \) with \( LT P_n^A \) and the set of leaf nodes added to \( B \) that have only sequences in \( p_n \) with \( LT P_n^B \). The nodes in \( LT P_n^A \) and \( LT P_n^B \) have sequence-sets that consist of a single cylinder and for each node \( a \in LT P_n^A \) there is a node in \( b \in LT P_n^B \) such that their sequence-sets are the same, \( S(a) = S(b) \). These node pairs are initial nodes of other algorithm instances. Denote the set of leaf nodes added to \( A \) that contain only sequences not in \( p_n \) with \( LT S_{n+1}^A \) and the set of leaf nodes added to \( B \) that contain only sequences not in \( p_n \) with \( LT S_{n+1}^B \) and start the next iteration of the algorithm instance.

We have that in all iterations \( n \) the sequence-sets of the nodes in sets \( LT P_n^A \) and \( LT P_n^B \) contain all the sequences in \( p_n \),

\[
\bigcup_{a \in LT P_n^A} S(a) = \bigcup_{b \in LT P_n^B} S(b) = p_n.
\]
It follows that the nodes in \( LT S_n^A \) and \( LT S_n^B \) contain the sequences from the
initial cylinder $w_0$ which are not contained in the cylinders from $P$ that were enumerated in previous iterations, $\bigcup_{a \in LTS_n^A} S(a) = \bigcup_{b \in LTS_n^B} S(b) = w_0 \setminus P_{n-1}$.

Also note that for nodes $a \in LTN_n^A$, $b \in LTS_n^B$ such that the intersection of their sequence-sets contains sequences in $p_n$, their child nodes $a' \in LTS_{n+1}^A$, $a \prec a'$ and $b' \in LTS_{n+1}^B$, $b \prec b'$ have sequence-sets with an empty intersection.

$$a \in LTN_n^A, \ b \in LTS_n^B, \ a' \in LTS_{n+1}^A, \ b' \in LTS_{n+1}^B, \ a \prec a', \ b \prec b', \ S(a) \cap S(b) \subseteq p_n \implies S(a') \cap S(b') = \emptyset$$  \hspace{1cm} (3)

The node $t \in LTS_n^B$ that contains both the sequences in $p_n$ and the sequences not in $p_n$ at the end of iteration $n$ has child nodes in both the set $LT_P^{n+1}$ and the set $LT_B^{n+1}$. We add the measure of the sequences in $p_n$ contained in the sequence-set of the node $t$ to the value $L(t)$ and the sum $L(t) + \lambda(S(t) \cap p_n)$ is distributed evenly among child nodes of $t$ in $LT_{B}^{n+1}$. For $t' \in LT_{B}^{n+1}$, $t \prec t'$ we have

$$L(t') = \frac{(L(t) + \lambda(S(t) \cap p_n))}{\lambda(S(t) \setminus p_n)} \lambda(S(t'))$$  \hspace{1cm} (4)

The value $L$ of a node $t \in LT_{A}^{n}$ is distributed evenly among the child nodes $t' \in LT_{A}^{n}$, $t \prec t'$

$$L(t') = \frac{\lambda(S(t'))}{\lambda(S(t))} L(t)$$  \hspace{1cm} (5)

For a node $t \in LT_{A}^{n}$ the sum $L(t) + \lambda(S(t) \cap p_n)$ is distributed evenly among child nodes of $t$ in $LT_{A}^{n+1}$, that is for $t' \in LT_{A}^{n+1}$, $t \prec t'$ we have $\mathcal{L}$. On the other hand if a node $t \in LT_{A}^{n}$ has sequence-set that contains only sequences from the cylinder $p_n$ then it has no child nodes in $LT_{A}^{n+1}$, also a node $t \in LT_{B}^{n}$ has no child nodes in $LT_{B}^{n+1}$ if it’s sequence-set contains only sequences from $p_n$. Therefore for all iterations $n$ and $T$ either $A$ or $B$ we have

$$\sum_{t \in LTS_n^T} L(t) \leq P_{n-1}$$

For a node $t$ denote the sum of measure of its sequence-set and the part of the measure of the so far enumerated cylinders from the set $P$ which is assigned to the node with $SL(t)$, $SL(t) = \lambda(S(t)) + L(t)$. For nodes $a \in LT_{A}^{n}$, $b \in LT_{B}^{n}$ we say that they satisfy the orthogonality condition if:

$$\lambda(w_0)\lambda(S(a) \cap S(b)) < (1 + cs)SL(a)SL(b)$$  \hspace{1cm} (6)

We will have that in all iterations $n$ for all nodes $a \in LT_{A}^{n}$, $b \in LT_{B}^{n}$ the orthogonality condition is valid. In particular, in the first iteration, $n = 1$, we have that $LT_{A}^{1}$ contains only the initial node $a_0$ and $LT_{B}^{1}$ contains only the initial node $b_0$ and we have that $\lambda(w_0)\lambda(S(a_0) \cap S(b_0)) = SL(a_0)SL(b_0)$. In Section 3.5 the construction of sequence-sets for nodes in $LT_{A}^{n}$ is described. It is shown that if $\mathcal{L}$ is valid for all nodes $a \in LT_{A}^{n}$, $b \in LT_{B}^{n}$ it is possible to construct $LT_{A}^{n}$ such that $\mathcal{L}$ and $\mathcal{L}$ are valid for all nodes $a \in LT_{A}^{n}$, $b \in LT_{B}^{n}$.
Define the remaining relative mass $f$ of a node $t$ is $f(t) = \frac{\lambda(w_0) \cdot ms(t)}{SL(t)}$ where $w_0$ is the cylinder of the input nodes and $m_0$ is the mass of the input nodes.

The mass $ms$ of a node $a \in LTS_n^A$ is evenly distributed among the child nodes $a' \in LTN_n^A$, $a < a'$

$$ms(a') = \frac{\lambda(S(a'))}{\lambda(S(a))}ms(a)$$  

(7)

From (6) and (7) we have that the relative mass of a node in $LTS_n^A$ is the same as the relative masses of its child nodes in $LTN_n^A$.

Next we iterate in any order through pairs of nodes $(a, b)$, $a \in LTN_n^A$, $b \in LTS_n^B$ whose intersection of sequence-sets contains sequences in $p_n$ and determine which of the strategies will gain capital of at least $c$ on infinite sequences in the intersection $S(a) \cap S(b)$. Define a set of node pairs $ABP_n$ whose intersection contains sequences from $p_n$

$$ABP_n = \{(a, b) : a \in LTN_n^A, b \in LTS_n^B, S(a) \cap S(b) \subseteq p_n\}$$

Index the node pairs in $ABP_n$ in any order, $ABP_n = \{(a, b)_1, \ldots, (a', b')_{z_n}\}$. Denote with $d^A(a, b)$ the part of mass $ms(a)$ assigned to the intersection $S(a) \cap S(b)$ by strategy $A$ and with $d^B(a, b)$ the part of mass $ms(b)$ assigned to the intersection by strategy $B$. Define the remaining mass, $ms_i^A$, of a node $a \in LTN_n^A$ after the mass has been assigned to the $i$th node pair in $ABP_n$:

$$ms_i^A(a) = ms(a) - \sum_{i = 0}^{i = z_n, u = a} d^A(a, v)$$

Analogously define the remaining mass $ms_i^B$ of a node $b \in LTS_n^B$ as

$$ms_i^B(b) = ms(b) - \sum_{i = 0}^{i = z_n, v = b} d^B(u, b)$$

Define the remaining relative mass $f_i^A$ of a node $a \in LTN_n^A$ as $f_i^A(a) = \frac{\lambda(w_0) \cdot ms_i^A(a)}{SL(a)}$ and the remaining relative mass $f_i^B$ of a node $b \in LTS_n^B$ as $f_i^B(b) = \frac{\lambda(w_0) \cdot ms_i^B(b)}{SL(b)}$. For all node pairs $(a, b)_i \in ABP_n$, we have that the sum of mass assigned by the nodes $a, b$ to the intersection of their sequence-sets is such that the capital is at least doubled on the sequences in the intersection by one and possibly both strategies, $d^A(a, b) + d^B(a, b) = 2c\lambda(S(a) \cap S(b))$. We choose the values $d^A(a, b)$, $d^B(a, b)$ so that the $b$ node always assigns the mass to the intersection unless it doesn’t have enough remaining mass. In that case $b$ assigns all
that it has and the rest is assigned from the remaining mass of node \(a\):

\[
\text{if } ms^{B}_{i-1}(b) \geq 2c\lambda(S(a) \cap S(b)) \\
\text{then } d^B(a, b) = 2c\lambda(S(a) \cap S(b)) \\
\text{else } d^A(a, b) + d^B(a, b) = 2c\lambda(S(a) \cap S(b)), d^B(a, b) = ms^{B}_{i-1}(b)
\]  

(8)

Denote with \(ms'\) the remaining mass of a node after the mass has been assigned to the sequence-set intersections of all node pairs in \(ABP_i\). For the node \(a \in LTN^A_n\) we have that \(ms'(a) = ms^A_n(a)\) and for the node \(b \in LTS^B_n\), \(ms'(b) = ms^B_n(b)\). For the node \(a \in LTN^A_n\) we distribute the remaining mass \(ms'(a)\) evenly among the child nodes \(a' \in LTS^A_{n+1}\). Analogously for the node \(b \in LTS^B_n\) we distribute the remaining mass \(ms'(b)\) evenly among the child nodes \(b' \in LTS^B_{n+1}\). We have for \(t \in LTN^A_n, t' \in LTS^A_{n+1}, t < t'\)

\[
ms(t') = \frac{ms'(t)}{\lambda(S(t) \setminus p_n)}\lambda(S(t'))
\]

(9)

and analogously for the nodes \(t \in LTS^B_n, t' \in LTS^B_{n+1}, t < t'\).

The nodes in \(LTP^A_n\) and \(LTP^B_n\) are the initial nodes of other algorithm instances. We have

\[
\bigcup_{a' \in LTP^A_n} a' = \bigcup_{b' \in LTP^B_n} b' = p_n \land \forall a' \in LTP^A_n \exists b' \in LTP^B_n \exists v \in \{0, 1\}^*, S(a') = S(b') = v\{0, 1\}^\infty
\]

(10)

The masses assigned to the intersection \(S(a) \cap S(b)\) by both strategies, \(d^A(a, b), d^B(a, b)\) are evenly distributed among the cylinders \(u'\) contained in the intersection. The sequence-set of a node \(t\) in \(LTN^A_n\) or \(LTS^B_n\) could be a subset of the enumerated cylinder \(p_n\) and some mass of the node \(t\) might be left unassigned. Denote the unassigned mass of a node \(t\) with \(um(t)\),

\[
um(t) = \begin{cases} 
ms'(t) & \text{if } S(t) \subseteq p_n \\
0 & \text{otherwise}
\end{cases}
\]

The unassigned mass is distributed evenly among the cylinders \(u' \in W_n\) which are contained in the sequence-set of a node \(t\). Denote the mass assigned to the cylinder \(u' \in W_n\) by strategy \(A\) with \(m_{a'}\) and the mass assigned by \(B\) with \(m_{b'}\). We have that

\[
m_{a'} = me(a_0)\frac{\lambda(u')}{\lambda(w)} + d^A(u, v)\frac{\lambda(u')}{\lambda(S(u) \cap S(v))} + um(u)\frac{\lambda(u')}{\lambda(S(u))} \\
m_{b'} = me(b_0)\frac{\lambda(u')}{\lambda(w)} + d^B(u, v)\frac{\lambda(u')}{\lambda(S(u) \cap S(v))} + um(v)\frac{\lambda(u')}{\lambda(S(v))}
\]

(11)

The masses \(m_{a'}, m_{b'}\) assigned to the cylinder \(u'\) are such that at least one of the strategies doubles the initial capital on sequences in \(u'\) and both masses are greater than zero. For nodes \(a', b'\) such that \(S(a') = S(b') = u'\) and \(m(a') = m_{a'}, m(b') = m_{b'}\) we start another algorithm instance \(alin(a', b')\).
Informally the algorithm instance is as follows

\textit{Initialization:}
1. Set \( L(a_0) = L(b_0) = 0, m_0 = \min(m(a_0), m(b_0))/2, \)
2. define the set to be enumerated, \( P, \) s.t. (1)
3. set iteration \( n := 0 \)

\textit{Iteration:}
1. Increment iteration \( n := n + 1, \) enumerate \( p_n \in P. \)
2. Add child nodes to leaf nodes in \( LTN_A^n, \) a new set of leaf nodes is \( LTN_A^n. \)
3. Assign mass from nodes in \( LTN_A^n \) and \( LTS^B_n \) to sequences in \( p_n \) by mass assignment rule (5).
4. Add child nodes to \( LTN_A^n \) and \( LTS^B_n \) to get sets of leaf nodes \( LTS^A_{n+1}, \)
   \( LTS^B_{n+1}, LTP^A_n, LTP^B_n. \) The node pairs in \( LTS^A_{n+1}, LTS^B_{n+1} \) satisfy the orthogonality condition (6) and (7). The construction of sequence-sets is described in Section 3.6. The \( L, ms \) for the nodes in \( LTN_A^n \) are defined by (5), (7).
5. Start other algorithm instances for the node pairs \((a', b') \) s.t. \( a' \in LTP^A_n, b' \in LTP^B_n, S(a') = S(b'). \)
6. goto Iteration.

We show that if the strategies don’t have enough mass to double the capital on sequences in \( P \) than the measure of the set \( P \) has to be greater than \( \text{const} \) which is a contradiction to (1). Assume that in some iteration \( r \) for some node in \( LTN_A^n \) there isn’t enough mass to be assigned to some intersection containing sequences in \( p_r. \) Denote this node with \( a_r. \) Denote with \( a_i \) the parent node of \( a_r \) in iteration \( i, a_i \in LTN_A^n. \) For iterations \( n \) prior to and including iteration \( r, \) denote with \( F_n \) the indices of node pairs \( ABP_n \) such that for the node pair \((a_n, b_i) \in ABP_n \) some mass of the the node \( a_n \) is assigned to the intersection \( S(a_n) \cap S(b) \).
\[
F_n = \{i : (a_n, b_i) \in ABP_n, d^A(a_n, b) > 0\}
\]
Denote with \( F \) the set of nodes in strategy \( B \) such that \( a_r \) or its parent node assigned some mass to the intersection, \( F = \bigcup_{n=[1, r]} \bigcup \{b : (a_n, b) \in ABP_n\}. \)

Note that \( F \) is a prefix-free set of nodes. The nodes from the same iteration are prefix-free since they are leaf nodes in that iteration. From (8) we have that if a pair of nodes in some iteration \( n \) had an intersection that contained sequences
in \( p_n \), then their child nodes in \( LTS \) sets of subsequent iterations don't have an intersection.

Also note that the relative mass of a node \( a_{n+1} \) before assignment of mass to the sequences in \( p_{n+1} \) is equal to the relative mass of node \( a_n \) after the mass has been assigned to sequences in \( p_n \), \( f^A(a_{n+1}) = f^A_z(a_n) \). From (8), (9) we have that nodes in \( LTN_i^A \) have the same relative mass as their parent nodes in \( LT_{i+1}^A \). From (8), (9) we have that the relative mass of a node in \( LT_{i+1}^A \) is the same as the relative mass of its parent node in \( LT_{i+1}^A \) after the mass has been assigned to sequences in \( p_i \).

**Proof.** (of Theorem 1) For some node pair \((a, b) \in ABP_0 \) we have that if the strategy \( A \) assigned some mass \( d^A(a, b) \) to the intersection then the relative mass of node \( a \) has decreased from \( f^A_{k-1}(a) \) to \( f^A_k(a) \). That is, \( d^A(a, b) = m_{\lambda w_0}^{-A} (f^A_{k-1}(a) - f^A_k(a)) SL(a) \). From the mass assignment rule (8) we have that \( f^B_k(b) = 0 \) and \( d^A(a, b) \leq 2c \lambda (S(a) \cap S(b)) \). From the orthogonality condition (10) we get \( SL(b) > m_{\lambda w_0}^{-A} (f^A_{k-1}(a) - f^A_k(a)) \). Summing over nodes in \( F \) we get \( \sum_{b \in F} SL(b) > \frac{m_{\lambda w_0}^{-A}}{2c (1 + cs)} \). From Lemma 1 we have

\[
\lambda(P_r) \geq \sum_{i \in [1, r]} \sum_{b \in F : b \in V \cap LTS^B_i} \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} SL(b) > \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0) 4c^2 (1 + cs)}
\]

which is a contradiction to \( (1) \).

**Lemma 1.** Given a prefix-free set of nodes \( V \) in strategy \( B \) we have

\[
\sum_{i \in [1, r]} \sum_{b \in V \cap LTS^B_i} \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} (1 - f^B_{z_i}(b)) SL(b) \leq \lambda(P_r)
\]

**Proof.** We prove by induction. The set \( I_1 \) contains the only node in \( LTS^B_1 \), the initial node of the algorithm instance of strategy \( B \), that is \( b_0 \). For \( b_0 \) we have \( \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} (1 - f^B_{z_1}(b_0)) SL(b_0) = \sum_{(a, b) \in ABP_1} d^B(a, b) \). We get the set of nodes \( I_{i+1} \) from set \( I_i \) by replacing the nodes in \( I_i \) which have child nodes in \( V \) and in \( LTS^B_{i+1} \), with their child nodes in \( LTS^B_{i+1} \). Assume for \( I_k \)

\[
\sum_{i \in [1, k]} \sum_{b \in LTS^B_i \cap I_i} \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} (1 - f^B_{z_i}(b)) SL(b) \leq \sum_{i \in [1, k]} \sum_{(a, b) \in ABP_i} d^B(a, b) \tag{12}
\]

For a node \( b \in LTS^B_k \) and its child nodes in \( LTS^B_{k+1} \) we have from (8), (9), (10):

\[
\sum_{b' \in LTS^B_{k+1}, b < b'} \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} (1 - f^B_{z_k+1}(b')) SL(b') =
\]

\[
\frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)} (1 - f^B_{z_k}(b)) SL(b) + \sum_{(a, b') \in ABP_{k+1}, b < b'} d^B(a, b')
\]

Therefore for \( I_{k+1} \) we have (12). Since \( \bigcup_{i \in [1, r]} V \cap LTS^B_i \subseteq I_r \) and from mass assignment rule (8) we have

\[
\sum_{i \in [1, r]} \sum_{b \in V \cap LTS^B_i} \frac{m_{\lambda w_0}^{-A}}{\lambda(w_0)^2 c} (1 - f^B_{z_i}(b)) SL(b) \leq \lambda(P_r)
\]
3.5 Constructing $LTN_n^A$

For a leaf node $a$ in strategy $A$ that has an intersection with a leaf node $b \in LT S_n^B$ such that the intersection contains both sequences in $p_n$ and not in $p_n$ we add child nodes so that we get $2^m$ leaf nodes with sequence sets of measure $2^{-m} \lambda(S(a))$. There is some integer $d$ such that the cylinders of measure $2^{-d}$ that are in $S(a)$ are contained in exactly one intersection with a node from $LT S_n^B$ and are either a subset of $p_n$ or are disjoint with $p_n$. Divide these cylinders into $2^m$ smaller cylinders and initialy put one of them in each sequence-set of the leaf nodes whose parent is $a$. Assign $2^{-m}L(a)$ to each of those leaf nodes. Since the orthogonality condition was satisfied for $a$ it is also satisfied for these leaf nodes. Next, rearrange the cylinders so that the measure of intersection between the newly added leaf nodes and node $b$ is unchanged but there is at most one leaf node $a'$ whose intersection with $b$ contains both the sequences in $p_n$ and not in $p_n$. The remaining $2^m - 1$ leaf nodes either contain only sequences in $p_n$ or only sequences not in $p_n$. The intersection $S(a') \cap S(b)$ consists of $r$ cylinders of measure $2^{-(d+m)}$ that are subsets of $p_n$ and $g$ cylinders of the same measure that are disjoint with $p_n$. Divide the cylinders in $S(a')$ that are subsets of $p_n$ into $r2^h$ smaller cylinders of measure $2^{-(d+m+h)}$ and distribute them over the leaf nodes whose intersection with $b$ contains only sequences in $p_n$ so that each node gets at most one smaller cylinder. From the leaf nodes whose intersection with $b$ contains only sequences not in $p_n$, choose $r2^h$ nodes and take a cylinder in the interaction with $b$ of measure $2^{-(d+m+h)}$ and put it in the sequence-set of the node $a'$. From the leaf nodes that received a cylinder from $a'$ take a cylinder of measure $2^{-(d+m+h)}$ from the interaction with some other node $b' \in LT S_n^B$ and put it in the sequence set of a node that gave a cylinder to node $a'$. For $m$ large enough and an appropriate value for $h$ we have that leaf nodes whose parent is $a$ satisfy the orthogonality condition \( \Box \) and their intersections with $b$ contains either sequences in $p_n$ or the sequences not in $p_n$. We repeat this procedure until there are no more leaf nodes in $A$ whose intersection with some leaf node $b \in LT S_n^B$ contains both sequences in $p_n$ and not in $p_n$. Now the set of leaf nodes in $A$ whose sequence-sets are in $w_0 \setminus P_{n-1}$ is $LT N_n^A$.

3.6 Constructing $LT S_{n+1}^A$, $LT S_{n+1}^B$, $LTP_n^A$, $LTP_n^B$

Since the sets of sequences in nodes of the strategies and the cylinder $p_n$ are clopen sets there is some integer $d$ such that for a cylinder $e$ of measure $\lambda(e) = 2^{-d}$ and $e \in w_0 \setminus P_{n-1}$, $e$ is either a subset of $p_n$ or $e$ and $p_n$ have an empty intersection. Also for each such cylinder $e$ there is a pair of nodes $a, b$ where $a \in LT N_n^A$, $b \in LT S_n^B$ such that $e$ is contained in the intersection of the sequence-sets of those two nodes.

A sequence set of a node $a \in LT N_n^A$ consists of $2^{d} \lambda(S(a))$ cylinders of measure $2^{-d}$, of which $ra$ are a subset of $p_n$. Similarly, in a sequence-set of a node $b \in LT S_n^B$ there are $rb$ cylinders of measure $2^{-d}$ which are a subset of $p_n$. To each node in $LT N_n^A$ and $LT S_n^B$ child nodes are added. The child nodes are such that the sequence-sets of the new leaf nodes have measure $2^{-d}$. Of the $2^{d} \lambda(S(a))$
leaf nodes in strategy $A$ whose parent is node $a$ there are $ra$ leaf nodes such that their sequence-sets consists of a single cylinder that is a subset of $p_n$. These leaf nodes are in $LT^A_n$. The cylinders $e$ in $S(a)$ of measure $2^{-d}$ which are not in $p_n$ are divided into $2^h$ cylinders and these smaller cylinders are distributed evenly among the remaining $2^d \lambda(S(a)) - ra$ leaf nodes. These leaf nodes are in $LT^A_{n+1}$. Similarly, for node $b$ there are $rb$ leaf nodes whose parent is $b$ and whose sequence-set consists of a single cylinder of measure $2^{-d}$ that is a subset of $p_n$. These leaf nodes are in $LT^B_n$. The remaining cylinders $e$ of measure $2^{-d}$ in $S(b)$ are divided into $2^h$ cylinders $es$, these are further divided into $2^h$ cylinders and these are distributed evenly among the remaining $2^d \lambda(S(b)) - rb$ leaf nodes. These leaf nodes are in $LT^B_{n+1}$. By choosing $h$ large enough, from (8) for nodes in $LT^A_n$, $LT^B_n$ and from (6) we have (6) for nodes in $LT^A_{n+1}$, $LT^B_{n+1}$.

References

1. John M. Hitchcock, Jack H. Lutz. Why Computational Complexity Requires Stricter Martingales. Theory of Computing Systems, 39(2):277-296, 2006.
2. Wolfgang Merkle, Nenad Mihailovic, and Theodore A. Slaman. Some results on effective randomness. Theory of Computing Systems 39(5):707-721, 2006.
3. Laurent Bienvenu. Kolmogorov-Loveland stochasticity and Kolmogorov complexity. Theory of Computing Systems 46(3):598-617, 2010.
4. Wolfgang Merkle, Joseph S. Miller, André Nies, Jan Reimann, Frank Stephan. Kolmogorov-Loveland Randomness and Stochasticity. Annals of Pure and Applied Logic 138:183-210, 2006.
5. Bart Kastermans and Steffen Lempp. Comparing notions of randomness. Theoretical Computer Science 411(3): 602-616, 2010
6. Laurent Bienvenu, Rupert Holzl, Thorsten Kraling, and Wolfgang Merkle. Separations of non-monotonic randomness notions. Journal of Logic and Computation 22(4):701-715, 2012.
7. Joseph S. Miller and André Nies. Randomness and computability: Open questions. Bulletin of Symbolic Logic 12(3):390-410, 2006.
8. K. Ambos-Spies and A. Kucera. Randomness in computability theory. In Peter Cholak, Steffen Lempp, Manny Lerman, and Richard Shore, editors, Computability Theory and Its Applications: Current Trends and Open Problems. Contemporary Mathematics 257:1-14, American Mathematical Society, 2000.
9. Rodney G. Downey, Denis R. Hirschfeldt, André Nies, Sebastiaan A. Terwijn. Calibrating randomness, Bulletin of Symbolic Logic 12(3): 411-491, 2006.
10. André Nies. Computability and randomness, Calrendon Press, Oxford, 2009.
11. Rodney G. Downey and Denis R. Hirschfeldt. Algorithmic Randomness and Complexity, Springer, 2010.
12. An. A. Muchnik, A. L. Semenov, and V. A. Uspensky. Mathematical metaphysics of randomness. Theoretical Computer Science 207:263-317, 1990.
13. M. Li and P. Vitanyi. An Introduction to Kolmogorov Complexity and Its Applications, 2nd Edition. Springer, 1997.