On Small Space Complexity Classes of Stochastic Turing Machines and Arthur-Merlin Games*

Maciej Liśkiewicz

Instytut Informatyki
Uniwersytet Wrocławski
Przesmyckiego 20
51-151 Wrocław, Poland
liskiewi@ii.uni.wroc.pl

Rüdiger Reischuk†

Institut für Theoretische Informatik
Med. Universität zu Lübeck
Wallstraße 40
D-23560 Lübeck, Germany
reischuk@informatik.mu-luebeck.de

October 1997

Abstract

A *Stochastic Turing machine* (STM) is a Turing machine that can perform nondeterministic and probabilistic moves and alternate between both types. Such devices are also called games against nature, Arthur-Merlin games, or interactive proof systems with public coins. We investigate stochastic machines with space resources between constant and logarithmic size, and, in addition, constant or sublinear bounds on the number of alternations between nondeterministic and probabilistic moves. Lower space bounds are shown for a specific family of languages by exploiting combinatorial properties. These results imply an infinite hierarchy with respect to the number of alternations of STMs, and nonclosure properties of certain classes.

1 The Computational Model

A stochastic Turing machine (STM) is a nondeterministic machine with the additional ability to perform random moves, also called games against nature by Papadimitriou [Pa85]. Alternative characterizations can be given by so called Arthur-Merlin-games [Ba85, Co89] and interactive proof systems with public coins [GoSi86]. A stochastic Turing machine M models a 2-person game where one of the players determines his moves randomly. In a probabilistic move M chooses among the successor configurations with equal probability. A computation of M on an input X can be described by a tree. To define acceptance of X, for each nondeterministic – also called existential – configuration one chooses a successor that maximizes the probability of reaching an accepting leaf. The acceptance probability of X is then given by the acceptance probability of the starting configuration in this truncated tree.

The error probability of a STM M has to be bounded away from $1/2$. We make the following requirements for M in order to accept a language L in space S: for every X it never uses more than $S(|X|)$ space and strings in L are accepted with probability more than $3/4$, while for strings not in L this probability is less than $1/4$. Mainly sublogarithmic space bounds S above the twice iterated logarithm will be considered, but for certain languages we will also investigate smaller space bounds. In that case, however, the space bounds can only be maintained with high probability since such functions are no longer space-constructible. W.l.o.g all space bounds S are assumed to be nondecreasing.

*this is an extended version of an invited lecture given at 22. MFCS, Bratislava, August 1997 [LiRe97b]

†the paper has been written while visiting the Department of Informatics at Kyushu University, Fukuoka, supported by the Japan Society for the Promotion of Science under Grant JSPS 29716102.

Definition 1 *Let* $\boldsymbol{MA_k Space(S)}$ *(resp.* $\boldsymbol{AM_k Space(S)}$ *) denote the set of languages that can be accepted by an* S *-space-bounded STM that makes at most* $k-1$ *alternations between nondeterministic and probabilistic configurations and starts with a nondeterministic (resp. probabilistic) move.* k *will be a fixed constant in most cases, but later we will also consider functions* k *that grow with the input size. For such a machine we also say that it works in space* S *and* k *rounds. The corresponding time classes will be denoted by* $MA_k Time(T)$ *and* $AM_k Time(T)$ *. If there is no bound on the number of alternations we simply write* $\boldsymbol{AMSpace(S)}$ *and* $\boldsymbol{AMTime(T)}$ *.* □

The power of polynomial time bounded stochastic machines has been investigated quite extensively using the characterization by Arthur-Merlin-games and interactive proof systems [Pa85, GoSi86, Sh92]. The complexity classes defined by stochastic machines can equivalently be described with the help of existential and probabilistic quantifiers. For these classes strong connections to standard classes have been obtained. Machines that arbitrarily often may alternate between existential and probabilistic configurations have exactly the same power as alternating Turing machines. Alternating machines work in existential and universal configurations, and one can directly interpret them as games, in which both players try to optimize their moves (existential and universal quantifiers) [CKS81]. The corresponding complexity classes are denoted by $\boldsymbol{ATime(T)}$. Thus it holds

$$AM\,Time(\mathrm{POL}) \;=\; A\,Time(\mathrm{POL}) \;=\; \mathcal{PSPACE}\ ,$$

where **POL** denotes the set of polynomial time bounds.

One could therefore say that given polynomial time and an unbounded number of quantifiers, universal quantifiers can be "interchanged" with probabilistic ones. If one however bounds the number of quantifiers the situation is different. For the combination of existential/universal quantifiers it is still unknown whether an interchange of quantifiers or the addition of a quantifier changes the complexity classes. Since it is generally believed that the Polynomial Hierarchy, which can be obtained this way, is strict it has been surprising that on the contrary for existential/probabilistic quantifier expressions any number of quantifiers can be replaced by just two [Ba85], more precisely using **CON** for the union of all constant bounds

$$MA_2 Time(\mathrm{POL}) \;\subseteq\; AM_2 Time(\mathrm{POL}) \;=\; AM_{\mathrm{CON}} Time(\mathrm{POL})\ .$$

2 Sublogarithmic Space Classes: What is Known?

Switching from time bounds to space bounds changes the situation completely. Now, due to Immerman's and Szelépcsenyi's simulation result alternating machines with a constant number of alternations have no more power than nondeterministic machines – at least for space bounds above the logarithm [Im88, Sz88]. In other words, existential and universal quantifiers can be interchanged and then merged to a single one:

$$\Sigma_k Space(S) \;=\; \Pi_k Space(S) \;=\; \Sigma_1 Space(S) \;=\; NSpace(S)$$

for every $k \geq 1$ and $S \geq \log$, where $\Sigma_k Space(S)$ (resp. $\Pi_k Space(S)$) denotes the alternating S -space bounded complexity class, where the machine starts in an existential (resp. universal) state and alternates $k-1$. However, for space bounds below the logarithm this is no longer true.

Definition 2 *Let* $\log^{(i)}$ *denote the* i *-times iterated logarithmic function* $n \mapsto \log_2 \ldots \log_2 n$ *. For the twice iterated logarithm instead of* $\log^{(2)}$ *we will also use the shorter notation* **llog** *.*

$$\mathbf{SUBLOG} \;:=\; \Omega(\mathrm{llog}\,) \cap o(\log\,)$$

denotes the set of all nontrivial (with resp. to DTM and NTM) sublogarithmic space bounds.

□

Several authors, Geffert, von Braunmühl/Gengler/Rettinger, and we have shown that the space hierarchy defined by such alternating machines is infinite [BGR93, Ge94, LiRe93, LiRe96a]:

$$\Sigma_1 Space(\text{SUBLOG}) \subset \Sigma_2 Space(\text{SUBLOG}) \subset \Sigma_3 Space(\text{SUBLOG}) \subset \dots$$

Furthermore the situation becomes trivial for space bounds below the twice iterated logarithm, since then even alternating machines accept only regular languages [Iw93]. On the other hand, it is easy to see that all these classes lie inside $\mathcal{P}$ since, by definition, they are contained in $\mathcal{AL} := ASpace(\log)$, which has shown to be equal to $\mathcal{P}$ [CKS81]. For a recent overview on alternating sublogarithmic space classes see [LiRe97a].

Space-bounded stochastic machines seem to be quite different. This even holds for the subclass of probabilistic machines without any nondeterminism. Freivalds has shown that probabilistic machines with constant space (probabilistic finite automata) are quite powerful already. They can accept nonregular languages [Fr81]. An example is the language

$$\texttt{COUNT} := \{1^n 0 1^m | n = m\}$$

which can be accepted by a probabilistic TM in constant space with an arbitrarily small error probability, that means

$$\texttt{COUNT} \in AM_1 Space(\text{CON}) = BPSpace(\text{CON}) ,$$

where BP stands for bounded error probabilistic machines (Monte Carlo).

Furthermore, probabilistic space classes cannot be separated by standard diagonalization techniques. Only recently, the first nontrivial separation results for such classes have been established, and these apply only to sublogarithmic space classes [DwSt92], [FrKa94].

Dwork and Stockmeyer have investigated stochastic Turing machines and interactive proof systems with small space bounds first. Extending an impossibility result for 2-way probabilistic finite automata they have shown that the language

$$\texttt{CENTER} := \{w0x \mid w, x \in \{0,1\}^* \quad \text{and} \quad |w| = |x|\}$$

cannot be recognized by a probabilistic TM even when sublogarithmic space is available [DwSt92]. However, a stochastic automata alternating between probabilistic and nondeterministic states can recognize this language easily, that is

$$\texttt{CENTER} \in AMSpace(\text{CON}) \setminus AM_1 Space(\text{SUBLOG}) .$$

It has been conjectured to be crucial that constant space-bounded STMs use (expected) exponential time for nonregular languages. This has been known for probabilistic machines, and has been shown for stochastic machines with a 1-way input restriction in [CHPW94] only recently, but in general is still open.

For sublogarithmic space bounds it makes a dramatic difference whether the random moves are known, as it is the case for stochastic machines and Arthur-Merlin games, or whether they are hidden as in standard interactive proof systems. Let use denote the corresponding complexity classes for the secret randomness by $\boldsymbol{IPSpace(S)}$. The language PALINDROME of all strings that are palindromes, is an example that can easily be recognized by an interactive proof system with a constant space-bounded verifier and hidden random moves, but requires logarithmic space in case of public coins [DwSt92]:

$$\texttt{PALINDROME} \in IPSpace(\text{CON}) \setminus AMSpace(\text{SUBLOG}) .$$

Note that for the corresponding time classes public or private coins do not make an essential difference [GoSi86].

By results of Condon/Lipton [CoLi89] and [DwSt92] (for an overview see [Co93]) we know that interactive proof systems even with only constant space – that means the verifier is a probabilistic automata – are already extremely powerful. A precise characterization by standard complexity classes, however, is still open. Let EXL , resp. EEXL denote the class of linear exponential ($2^{O(n)}$), resp. linear double exponential bounds. The best known upper and lower bounds seem to be

$$DTime(\mathrm{EXL}) \subseteq IPSpace(\mathrm{CON}) \subseteq ATime(\mathrm{EEXL}) .$$

Fortunately, in case of stochastic machines with public random moves the gap is not that huge. Condon has shown that such machines with logarithmic space still stay inside $\mathcal{P}$, more precisely [Co89]

$$AM\ Space(\log) = \mathcal{P} = A\ Space(\log) .$$

This means that for logarithmic space – as we have seen above for polynomial time – unbounded sequences of alternating quantifiers, either existential/probabilistic or existential/universal, define the same complexity class. For quantifier sequences of fixed length the situation is not clear.

We will consider this question for sublogarithmic space bounds, where nothing has been known so far. Motivated by the separation results for alternating Turing machines, we will investigate stochastic space complexity classes without any restrictions on the time complexity, but with bounds on the number of alternations, or equivalently interactions between the prover and the verifier.

A sublogarithmic STM may run for exponential expected time. It has been shown in [DwSt92] that some languages cannot be recognized faster – `CENTER` is one of such example. On the other hand, the power of sublogarithmic space-bounded STMs restricted to polynomial expected time is still an open problem.

Our approach will be to define a sequence of simple languages parameterized by an index k , and to show that in order for a stochastic machine to accept them a certain number of alternations are necessary and sufficient, where the bound depends on k . As a consequence, almost tight separations for the corresponding complexity classes and an infinite hierarchy are obtained.

3 Lower Space Bounds, Separating Results, and Nonclosure under Complement

The results for the languages `CENTER` and `PALINDROME` cited above, for any space bound $S \in o(\log)$, yield the separations

$$BPSpace(S) = AM_1Space(S) \subset AMSpace(S) \subset AMSpace(\log) = \mathcal{P} .$$

By limiting the number of alternations this chain will be refined.

Definition 3 *Let Σ be an alphabet, and $\mathrm{bin}(i)$ denote the binary representation of integer i of length $\max\{1, \lceil \log_2(i+1) \rceil\}$ and*

$$\mathtt{BIN}(m) := \mathrm{bin}(0)\#\,\mathrm{bin}(1)\#\,\mathrm{bin}(2)\#\ldots\#\,\mathrm{bin}(m) .$$

For a natural number k and strings $U, u \in \Sigma^+$ we say that u occurs k times in U *if there exist $W_0, W_1, \ldots, W_k \in \Sigma^*$ such that $U = W_0uW_1uW_2\ldots W_{k-1}uW_k$, and define the following languages*

$$\mathtt{PATTERN}_k := \{\ U\#u\#\mathtt{BIN}(2^d) \mid U, u \in \{0,1,2\}^+,\ d \in \mathbb{N},\ |u| = d, \quad u \text{ occurs } k \text{ times in } U\ \}.$$

□

As a first step towards separating the round hierarchy for space-bounded stochastic machines we show

Theorem 1 *For $S \in o(\log)$, no S –space-bounded STM starting with a probabilistic move can recognize* $\mathtt{PATTERN}_1$ *in 2 rounds , that is*

$$\mathtt{PATTERN}_1 \notin AM_2Space(o(\log)) .$$

This result has been extended in [Li96] by showing that for every $k > 1$ the language $\texttt{PATTERN}_{k^{k-1}}$ requires more than k rounds on sublogarithmic STMs. Developing a simpler and more general proof technique we are now able to improve this separation substantially

Theorem 2 *For $S \in o(\log)$ and for any integer $k \geq 2$, no S-space-bounded STM can recognize $\texttt{PATTERN}_k$ in $\lceil 2k/3 \rceil - 1$ rounds, that is*

$$\texttt{PATTERN}_k \notin AM_{\lceil 2k/3 \rceil -1} Space(o(\log)) \cup MA_{\lceil 2k/3 \rceil -1} Space(o(\log)) .$$

With a larger number of alternations the languages $\texttt{PATTERN}_k$ can be accepted with little space.

Theorem 3 *For any integer $k \geq 1$, and for arbitrary small $\epsilon > 0$, there exists a llog-space-bounded STM for $\texttt{PATTERN}_k$ with error probability ϵ that works in $2k$ rounds starting with a nondeterministic move, that is*

$$\texttt{PATTERN}_k \in MA_{2k} Space(\operatorname{llog}) .$$

Moreover, the machine works in polynomial time (actually almost linear time).

Hence, if one restricts the space bound to the twice iterated logarithm, the above upper bound for the number of rounds is optimal for $k = 1$, but there is still a gap from $\lceil 2k/3 \rceil - 1$ to $2k$ for $k > 1$. We conjecture that no sublogarithmic machine can recognize $\texttt{PATTERN}_k$ in less than $2k$ rounds. The theorem below shows that the space bound llog is critical for STMs to recognize the languages $\texttt{PATTERN}$.

Theorem 4 *For $S \in o(\operatorname{llog})$ and for any integer $k \geq 1$, no S-space-bounded STM can recognize $\texttt{PATTERN}_k$ that is*

$$\texttt{PATTERN}_k \notin AMSpace(o(\operatorname{llog})) .$$

Therefore, the following separations are obtained:

Theorem 5 *For any integer $k \geq 1$ it holds*

$$MA_{3k+2} SpaceTime(\operatorname{llog}, \mathrm{POL}) \not\subseteq AM_k Space(o(\log)) \cup MA_k Space(o(\log)) .$$

In addition, it holds $MA_2 SpaceTime(\operatorname{llog}, \mathrm{POL}) \not\subseteq AM_2 Space(o(\log))$.

These results imply that the round/alternation hierarchy for sublogarithmic space-bounded AM_k machines is infinite – similar to alternating TMs.

Corollary 1 *For any function $S \in \mathrm{SUBLOG}$ and any integer $k \geq 1$, it holds*

$$AM_k Space(S) \cup MA_k Space(S) \subset MA_{3k+2} Space(S) .$$

Thus we get an infinite hierarchy with a factor 3 distance between different levels. One can also obtain strong lower bounds on the space and the number of rounds for recognizing the complement of the language $\texttt{PATTERN}_1$. Since $\texttt{PATTERN}_1$ itself lies in $MA_2 Space(\operatorname{llog})$, this shows that for some languages that can be accepted in few rounds and little space even a drastic increase of the number of alternations does not help to recognize their complement with small space.

Theorem 6 *For any function $K \in O(\sqrt{n})$ it holds*

$$\overline{\texttt{PATTERN}_1} \notin AM_K Space(o(\log)) .$$

Obviously, this result implies the following

Theorem 7 *For any $S \in \mathrm{SUBLOG}$, and for any function K with $2 \leq K \leq O(\sqrt{n})$ the complexity classes $AM_K Space(S)$ and $MA_K Space(S)$ are not closed under complement.*

These impossibility results for space bounds in SUBLOG can be translated to arbitrary small bounds (remember functions in SUBLOG grow at least as the twice iterated logarithm). Such a technique will not work for deterministic or nondeterministic machines. Similarly to [FrKa94] for Monte Carlo machines, this can be done by a translation technique based on a suitable encoding.

Definition 4 *For a string $W = w_0 w_1 \ldots w_n$ over Σ and a symbol $a \notin \Sigma$ define*

$$E(W,a) \;:=\; w_0 a^{2^0} w_1 a^{2^1} w_2 a^{2^2} \ldots w_i a^{2^i} \ldots w_n a^{2^n} \; .$$

Let $\#_1, \#_2, \ldots$ be symbols not in Σ. Then

$$E_0(W) \;:=\; W \;, \qquad \text{and for } i \geq 1 \qquad E_i(W) \;:=\; E(E_{i-1}(W), \#_i) \; .$$

For integers k and i a padded version of `PATTERN` *is constructed by*

$$\texttt{PATTERN}_{k,i} \;:=\; \{\, U\#u\#E_i(\texttt{BIN}(2^d)) \mid U,u \in \{0,1,2\}^+,\ d \in \mathbb{N},\ |u| = d,\ u \text{ occurs at least } k \text{ times in } U \,\}.$$

□

Note that $\texttt{PATTERN}_{k,0}$ is just the language $\texttt{PATTERN}_k$. Combining the lower bound argument and this translation technique one can show

Theorem 8 *For $k \geq 1$ and $i \geq 0$, it holds*

$$\texttt{PATTERN}_{2k,i} \;\notin\; AM_k Space(o(\log^{(i+1)})) \;\cup\; MA_k Space(o(\log^{(i+1)})) \; .$$

On the other hand one can show a suitable upper bound by combining the recognition algorithm for the unpadded languages with Freivalds' probabilistic finite automaton for the language

$$\{01^{2^0} 01^{2^1} 01^{2^2} \ldots 01^{2^i} 01^{2^{i+1}} \ldots 01^{2^n} \mid n \in \mathbb{N}\} \; .$$

This yields

Theorem 9 *For any integer $k \geq 1$ and any $i \geq 1$*

$$\texttt{PATTERN}_{k,i} \;\in\; AM_{2k+1} Space(\log^{(i+2)}) \; .$$

Thus, not only for space bounds in SUBLOG, stochastic Turing machines possess an infinite round-quantifier hierarchy, but also for arbitrary small nonconstant bounds $\log \ldots \log$ with an exponential gap from $\log^{(i+1)}$ to $o(\log^{(i)})$.

Theorem 10 *For any nonconstant space bound S growing suitably smoothly, that means for some i it holds $\Omega(\log^{i+1}) \leq S \leq o(\log^i)$, the stochastic complexity classes form an infinite hierarchy by*

$$AM_k Space(S) \;\cup\; MA_k Space(S) \;\subset\; MA_{4k+1} Space(S) \; .$$

In the remainder of this paper we will present our techniques to obtain these separation results. In Section 4 notions for analyzing stochastic Turing machines are introduced. Section 5 contains the main technical details and proofs that some specific combinatorial properties of a language are necessary to accept it within a bounded number of rounds. These imply the lower bounds for the `PATTERN` languages (Theorem 1, 2, 8, and 6). In Section 5 the upper bounds for $\texttt{PATTERN}_k$ are explained (Theorem 3 and 9). Finally, we will discuss some open questions concerning space-bounded stochastic Turing machines.

4 Technical Formalism to Analyse STMs

Let us start with some basic notion concerning stochastic Turing machines. A stochastic machine looks like an ordinary Turing machine except that its states are divided into the 4 types probabilistic (or random), nondeterministic (or existential), accepting, and rejecting.

A **memory state** of a STM is an ordered triple $\alpha = (q, u, i)$, where q is a state of the machine, u a string over the work tape alphabet, and i a position in u, the location of the work tape head. By $|\alpha|$ we denote the length of the work tape contents u in α, that is the space complexity of α.

Given an input X, a **configuration** of a machine M is a pair $C = (\alpha, j)$ consisting of a memory state α and a position $j \in [0..|X|+1]$ of the input head. $j = 0$ (resp. $j = |X| + 1$) means that the input head scans the left (resp. the right) end-marker of the input tape. Let $\texttt{head}(C) := j$ denote the input head position in this configuration.

To each configuration $((q, u, i), j)$ we associate a type, which is the type of its state q. Let us assume that a machine starts with its input head on the left end-marker of the input tape and that every accepting or rejecting configuration is terminal.

The computation of a stochastic machine M on an input X can be described by a tree $\mathcal{T}_M(X)$ similar to what is done for alternating machines. The root is the initial configuration C_0 of M on X, the sons of a node are its successor configurations, and leaves are the terminal configurations. Nodes are labeled as being probabilistic, nondeterministic, etc. according to the type of the configuration. In the following we will consider a fixed STM M.

A **phase** of a computation of M is a linear sequence of consecutive steps. We call a phase Φ a **probabilistic (resp. nondeterministic) round** if M starts the phase in a probabilistic (resp. nondeterministic) configuration and makes only probabilistic (resp. nondeterministic) moves during Φ (note that the last configuration in Φ may be of a different type). For configurations C, D, where C is probabilistic, let $\mathbf{Rmove}[\boldsymbol{C}, \boldsymbol{D}, \boldsymbol{X}]$ denote the probability that with X on its input tape and starting in C the machine reaches the configuration D in a probabilistic round. We are particular interested in configurations D that are maximal with respect to reachability from C in probabilistic rounds. This means that D is not a probabilistic configuration (hence nondeterministic, accepting or rejecting), but the predecessors of D are. For such a D let $\mathbf{Rmove}^{\Lambda}[\boldsymbol{C}, \boldsymbol{D}, \boldsymbol{X}] = \mathrm{Rmove}[C, D, X]$, whereas for non-maximal D $\mathrm{Rmove}^*[C, D, X] := 0$.

For a nondeterministic configuration C let $\mathbf{Nmove}(\boldsymbol{C}, \boldsymbol{D}, \boldsymbol{X})$ denote the predicate that is true iff starting in C with X the machine can reach D in a nondeterministic round. For the predicate $\mathbf{Nmove}^{\Lambda}(\boldsymbol{C}, \boldsymbol{D}, \boldsymbol{X})$ to be true we require in addition that D is maximal, that means D is a probabilistic or terminal configuration.

In order to simplify the analysis we will compact a computation tree to its essential parts, which are the steps where the machine alternates between nondeterministic and probabilistic mode.

A "subtree" $\mathcal{S}$ of $\mathcal{T}_M(X)$ is called a **strategy** of M on X if the following holds:

- the root of $\mathcal{S}$ equals the root of $\mathcal{T}_M(X)$,
- for a probabilistic node C every son D fulfills $\mathrm{Rmove}^*[C, D, X] > 0$,
- a nondeterministic node C has at most one son D and for it the predicate $\mathrm{Nmove}^*(C, D, X)$ is true.

A strategy $\mathcal{S}$ is a **full strategy** if it is a maximal subtree of $\mathcal{T}_M(X)$ with respect to these conditions.

Note that $\mathcal{S}$ is not a subtree in the strict graph-theoretic sense. A path of the computation tree may be collapsed to an edge, thus a son of a node C in $\mathcal{S}$ in general is a descendent of C in the computation tree. Obviously, the depth of $\mathcal{S}$ is bounded by k if M works in k rounds.

Let $\mathbf{Reach}[\boldsymbol{C}, \boldsymbol{\mathcal{S}}, \boldsymbol{X}]$ denote the probability that the node C is reached by a path starting at the root. Formally, $\mathrm{Reach}[C_0, \mathcal{S}, X] := 1$. For any other node D with father C,

$$
\begin{array}{lcll}
\mathrm{Reach}[D, \mathcal{S}, X] & := & \mathrm{Reach}[C, \mathcal{S}, X] \cdot \mathrm{Rmove}^*[C, D, X] & \text{if } C \text{ is probabilistic,} \\
\mathrm{Reach}[D, \mathcal{S}, X] & := & \mathrm{Reach}_X[C, \mathcal{S}, X] & \text{if } C \text{ is nondeterministic.}
\end{array}
$$

Finally, define **Accept[$\boldsymbol{\mathcal{S}}, \boldsymbol{X}$]** as the probability that the strategy $\mathcal{S}$ for input X is accepting, i.e.

$$\text{Accept}[\mathcal{S}, X] \;:=\; \sum_{D \text{ accepting}} \text{Reach}[D, \mathcal{S}, X] \;.$$

We say that M accepts X with probability

$$p[X] \;:=\; \max_{\mathcal{S}} \{\text{Accept}[\mathcal{S}, X] \mid \mathcal{S} \text{ is a full strategy on } X\} \;.$$

The following terminology is basically taken from [GrWe86, DwSt92].

Let Z be a string. A **probabilistic starting event** is a pair $\langle \alpha, h \rangle$, where α is a probabilistic memory state and $h \in \{\text{Left}, \text{Right}\}$. The interpretation is that M starts in memory state α according to h on the leftmost, resp. on the rightmost symbol of Z. A **stopping event** is either

1. a pair $\langle \alpha, h \rangle$ as above meaning that the input head crosses the left (resp. right) boundary of Z with M being in memory state α, or
2. "Accept" meaning that M halts in the accepting state, or
3. "Reject" meaning that M halts in the rejecting state, or
4. "Alter" meaning that within Z the machine alternates from a probabilistic to a nondeterministic round, or
5. "R-Loop" meaning that a probabilistic round loops forever within Z.

For each starting event σ and each stopping event τ, let $\boldsymbol{p[Z, \sigma, \tau]}$ be the probability that M started on Z in σ reaches the event τ within a probabilistic round, in which the input head does not leave Z (except possibly in τ itself). These numbers are called **word probabilities** of Z.

Let s be a positive integer and consider all pairs (σ, τ) of starting and stopping events that are generated by memory states α of space complexity at most s in some fixed arbitrary ordering. Let $\mathtt{WP}_{\boldsymbol{s}}\boldsymbol{(Z)}$ be the vector of all word probabilities of these pairs.

An s-space-bounded computation within a probabilistic round of M can be modeled by a Markov chain with a finite state space, say $\sigma_1, \ldots, \sigma_m$ for some number $m \leq \exp O(s)$. The Markov chain is completely defined by the transition matrix $R = \{r_{ij}\}_{1 \leq i,j \leq m}$, where r_{ij} denotes the probability that a single step moves from state σ_i to state σ_j. Let σ_1 be the designated starting state. There is also a set T_R of **trapping states** σ_t with the property $r_{tt} = 1$. For $\sigma_t \in T_R$, let $\boldsymbol{p^{\Lambda}[t, R]}$ denote the probability that the Markov chain starting in σ_1 is trapped in state σ_t.

With respect to nondeterministic rounds a word Z is characterized by **word transitions**. A nondeterministic starting event is a pair $\langle \alpha, h \rangle$ similar to above, but now α has to be a nondeterministic memory state. A stopping event is an event from above of type 1., 2. or 3. For a starting event σ and a stopping event τ, define the predicate $\boldsymbol{t(Z, \sigma, \tau)}$ to be true iff M starting on Z in σ can reach τ in a nondeterministic round without leaving Z.

Let $\mathtt{WT}_{\boldsymbol{s}}\boldsymbol{(Z)}$ denote the vector of all word transitions involving memory states of space complexity at most s, similar to the vector of word probabilities.

5 Exploiting the Limitations of Stochastic Machines with Small Memory

In this section we will present the main line of reasoning to prove the lower bounds. For this purpose, methods developed in [GrWe86, DwSt92] and [LiRe96a] will be applied and extended to sublogarithmic space-bounded stochastic machines. For a real number $\beta \geq 1$ two numbers r and r' are called **$\boldsymbol{\beta}$-close** if either

(i) $r = r' = 0$, or
(ii) $r > 0$, $r' > 0$, and $\beta^{-1} \le r/r' \le \beta$.

Two word probabilities $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(\overline{w})$ are β-close if the vectors are componentwise β-close. Similarly, two Markov chains $R = \{r_{ij}\}_{1\le i,j\le m}$ and $R' = \{r'_{ij}\}_{1\le i,j\le m}$ with identical state spaces are β-close if r_{ij} and r'_{ij} are β-close for all pairs i, j. It has been shown that close transition probabilities guarantee that the trapping probabilities are also somehow close.

Lemma 1 ([GrWe86, DwSt92]) *Let R and R' be two m-state Markov chains which are β-close, and let σ_t be a trapping state of both R and R'. Then $p^*[t,R]$ and $p^*[t,R']$ are β^{2m}-close.*

Suppose M is a STM of space complexity S, where for simplicity we may assume that S is nondecreasing. Let $\mathrm{Vol}(s)$ denote the number of possible s-space bounded memory states of M. It holds $\mathrm{Vol}(s) \le \exp O(s)$. We restrict all word probabilities, word transitions, and strategies considered throughout this section to this fixed machine.

Below we will consider strings w and $\overline{w}$ of some bounded length that have the same word transitions and almost the same word probabilities, and investigate k-round strategies of M on inputs X of the following periodic structure

$$X \;:=\; w\ u_1\ w\ u_2\ w \ldots u_{b-1}\ w\ u_b\ ,$$

where k and b are positive integers and $u_1, \ldots, u_b$ are arbitrary strings. The key argument will show that for some position i in X one can replace the i-th substring w by $\overline{w}$ such that the probability for a specific k-round strategy to accept the distorted input does not decrease drastically. We can show that the loss is small enough if b exceeds $\frac{3}{2}k$, but further improvements of this bound seem possible. In other words, if the verifier asks too few questions to the prover there is a good chance that the prover can cheat successfully.

Proposition 1 *Let $\ell, k,$ and b be positive integers with $k \ge b$, and define $s := S(\ell)$. Assume that w and $\overline{w}$ are words with identical word transitions, that is $\mathtt{WT}_s(w) = \mathtt{WT}_s(\overline{w})$, and β-close word probabilities $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(\overline{w})$ for some $\beta \ge 1$. Furthermore, let $X := w\ u_1\ w\ u_2 \ldots w\ u_b$ be of length at most ℓ, and assume that this bound on the length also holds if in X the occurrence of w is replaced by $\overline{w}$.*

Then for any k-round strategy $\mathcal{S}$ on X there exists an integer $i \in [1..b]$ and a k-round strategy $\mathcal{S}'$ such that for the input $X_i := w\ u_1\ w \ldots u_{i-1}\ \overline{w}\ u_i \ldots w\ u_b$ and for $z := 10 \cdot \mathrm{Vol}(s) + 20$ it holds

$$\mathrm{Accept}[\,\mathcal{S}', X_i\,] \quad \ge \quad \frac{b-k}{b} \cdot \beta^{-2z\lceil k/2\rceil} \cdot \mathrm{Accept}[\,\mathcal{S}, X\,]\ .$$

Proof. We will consider configurations of the full strategy $\mathcal{S}$ on X with regard to input head positions of the configurations. The crucial point is that there exists an index $i \in [1..b]$ such that if in $\mathcal{S}$ we prune away subtrees rooted to configurations with the input head reading the i-th string w of X then

- the probability that the pruned strategy $\mathcal{S}'$ is accepting on X is greater than $\frac{b-k}{b} \cdot \mathrm{Accept}[\mathcal{S}, X]$, and
- $\mathcal{S}'$ is a strategy for X_i, too, i.e. for any probabilistic configuration C_1 and its son C_2 it holds that $\mathrm{Rmove}^*[C_1, C_2, X_i] > 0$ and for C_3, a son of C_2 it is true that $\mathrm{Nmove}^*(C_2, C_3, X_i)$. For C_1 and C_2 we will say even more, namely the probabilities for the inputs X and X_i that M starting in C_1 reaches C_2 in a probabilistic round are β^{2z}-close (see Fig. 1).

These properties imply the claim of the proposition.

Before we give the proof in detail let us make the following convention concerning the input head positions. Let $C = (\alpha, h)$ describe a configuration for input X such that $h > |wu_1 \ldots wu_{i-1}w|$, i.e. let C be a configuration with the input head right of the i-th substring w in X. Then if we consider C for input X_i we will understand C as (α, h'), with $h' := h + |\overline{w}| - |w|$, i.e. with the input head at position h' reading the same symbol of suffix $u_i w u_{i+1} \ldots w u_b$ as in h. In the other case $h \le |wu_1 \ldots wu_{i-1}|$, we consider configuration C for both input words without any change of the value h.

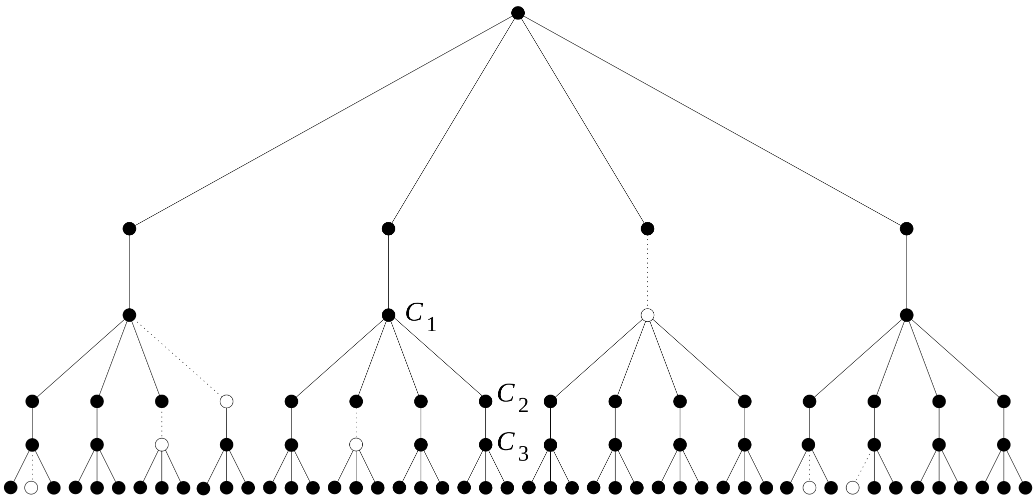

Figure 1: *An example of a full 5-round strategy* $\mathcal{S}$ *on the input* $X = wu_1 \dots wu_b$ *that starts in a probabilistic round. White nodes describe configurations in which* M *reads the* i *-th string* w *of* X *, where* i *is chosen in a particular way. Then in the strategy we prune away subtrees rooted at these nodes. The probability that the pruned strategy* $\mathcal{S}'$ *is accepting on* X *is greater than* $\frac{b-5}{b} \cdot \mathrm{Accept}[\mathcal{S}, X]$ *. The probability that* $\mathcal{S}'$ *is accepting on* X_i *is at least* $\frac{b-5}{b} \cdot \beta^{-2z \cdot 3} \cdot \mathrm{Accept}[\mathcal{S}, X]$ *.*

Now, let V be the set of all configurations in $\mathcal{S}$. Partition V into subsets V_j for $j \in [0..b]$ according to the input head position as follows. Let V_0 contain all configurations in which the machine reads a substring u_j or one of the end markers. For $j > 0$ define

$$V_j \ := \ \{C \in V \mid |w\ u_1 \dots u_{j-1}| \ < \ \mathtt{head}(C) \ \leq \ |w\ u_1 \dots u_{j-1}\ w| \ \} \ .$$

Since M starts reading the left end-marker the root of $\mathcal{S}$ belongs to V_0 .

Let V_{acc} be the set of all accepting configurations in $\mathcal{S}$. Then for any $C \in V_{acc}$ let $P(C)$ be the set of nodes that lie on the path from the root to C , including C , but excluding the root. Define $J(C)$ as the subset of $[0..b]$ containing all indices j such that an element from $P(C)$ is in V_j , i.e.

$$J(C) \ := \ \{j \in [0..b] \mid P(C) \cap V_j \neq \emptyset\} \ .$$

Since the length of each path in $\mathcal{S}$ is bounded by k , we get $1 \leq |J(C)| \leq k$. For $J \subseteq [0..b]$ of size at most k , and for $d \in [1..b]$ define

$$A_J \ := \sum_{C \in V_{acc},\ J(C)=J} \mathrm{Reach}[C, \mathcal{S}, X] \qquad \text{and} \qquad B_d \ := \sum_{\substack{J \subseteq [0..b] \\ 1 \leq |J| \leq k \text{ and } d \in J}} A_J \ .$$

Obviously, it holds

$$\mathrm{Accept}[\mathcal{S}, X] \ = \sum_{\substack{J \subseteq [0..b] \\ 1 \leq |J| \leq k}} A_J \ .$$

We claim that there is an index $i \in [1..b]$ such that

$$B_i \leq \frac{k}{b} \cdot \mathrm{Accept}[\mathcal{S}, X] \ . \tag{1}$$

To see this consider the sum $B_1 + B_2 + \ldots + B_b$. According to the definition of B_d, and because each subset J has at most k numbers, we conclude that each element A_J occurs in the sum above at most k times. Therefore

$$B_1 + B_2 + \ldots + B_b \leq \sum_{\substack{J \subseteq [0..b] \\ 1 \leq |J| \leq k}} k \cdot A_J = k \cdot \text{Accept}[\mathcal{S}, X] .$$

Hence, by the Pigeon-Hole-Principle there exists an index $i \in [1..b]$, such that (1) holds. Remove from $\mathcal{S}$ all nodes $v \in N_i$ together with all their descendants. Denote the strategy obtained this way by $\mathcal{S}'$. In particular, since only nodes C with $i \in J(C)$ have been removed from V_{acc}

$$\text{Accept}[\mathcal{S}', X] = \text{Accept}[\mathcal{S}, X] - B_i \geq \frac{b-k}{b} \cdot \text{Accept}[\mathcal{S}, X] .$$

Now consider the input $X_i = w\ u_1\ w \ldots u_{i-1}\ \bar{w}\ u_i \ldots w\ u_b$. Remember that w and $\bar{w}$ have been assumed to possess the same word transitions and that their word probabilities are β-close. Using these properties one can prove the following

Lemma 2 *Let C, D be two configurations in $\mathcal{S}'$ such that D is a son of C. Then it holds:*

1. *if C is nondeterministic then* $\text{Nmove}^*(C, D, X) \iff \text{Nmove}^*(C, D, X_i)$,
2. *if C is probabilistic then* $\text{Rmove}^*[C, D, X]$ *and* $\text{Rmove}^*[C, D, X_i]$ *are* β^{2z}*-close.*

Proof. Claim (1) follows easily from the property that $\mathtt{WT}_s(w) = \mathtt{WT}_s(\bar{w})$.
To prove (2), we construct two z-state Markov chains R and $\bar{R}$, with $z = 10 \cdot \text{Vol}(s) + 20$, which model a probabilistic round of M on inputs X, resp. X_i, when the machine starts in C. R and $\bar{R}$ are constructed in such a way that they are β-close and that there is a trapping state σ_t of both R and $\bar{R}$ which corresponds to the configuration D. Then using Lemma 1 we obtain that $p^*[t, R]$ and $p^*[t, \bar{R}]$ are β^{2z}-close which proves (2).

The full details of the proof of (2) are as follows. Denote the prefix $wu_1w \ldots wu_{i-1}$ of X by Y and the suffix $u_iwu_{i+1} \ldots wu_b$ by Z. Hence $X = YwZ$ and $X_i = Y\bar{w}Z$. Note that the strategy $\mathcal{S}'$ is constructed in such a way that the machine M being in configurations C and D does not scan the string w in YwZ. Below we will consider only the case that M in configuration C reads Y and in D reads the suffix Z. The remaining cases can be showed in a very similar way.

Denote by Y_1 the prefix of Y of the length $\mathtt{head}(C)$ and let the remaining part of Y will be denoted by Y_2. Similarly, let Z_1 be the prefix of Z such that M in configuration D reads the rightmost symbol of Z_1 and let Z_2 be a suffix such that $Z = Z_1 Z_2$. Each Markov chain we describe has z states. The first $10 \cdot \text{Vol}(s)$ states have the form $\langle \alpha, h \rangle$, where α is a probabilistic memory state of the length bounded by s and h, with

$$h \in \{\text{Left}_U, \text{Right}_U \mid U \in \{\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$\}\} ,$$

denotes the leftmost, resp. the rightmost symbol of words $\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$$. An intuitive meaning of a state $\langle \alpha, h \rangle$ of the chain R is: start M on the input $Y_1 Y_2 w Z_1 Z_2$ in memory state α and with the input head position according to h. E.g. if $h = \text{Left}_{Y_2}$ then start with the input head reading the leftmost symbol of the substring Y_2 and for $h = \text{Right}_{Y_2}$, the rightmost symbol of Y_2. The meaning of state $\langle \alpha, h \rangle$ for the chain $\bar{R}$ is analogous. The only difference is that Left_w (Right_w) denotes the leftmost (the rightmost, resp.) symbol of $\bar{w}$. The next twenty states are the following: $\text{Accept}_U, \text{Reject}_U, \text{Alter}_U$ and R-Loop_U, for $U \in \{\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$\}$. For chain R they mean that the probabilistic computation of M accepts, rejects, alternates or loops forever within $\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$$, respectively. For $\bar{R}$ the meaning is analogous.

The transition probabilities r_{ij} of R for non-trapping states σ_i are obtained from word probabilities of M on the substrings: $\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$$. More precisely, the transitions r_{ij} such that states

σ_i, σ_j are applied both to the same substring are equal to an appropriate word probabilities of M on this substring. E.g. if $\sigma_i = \langle \alpha_i, \mathrm{Right}_w \rangle$ and $\sigma_j = \langle \alpha_j, \mathrm{Left}_w \rangle$ then $r_{ij} = p[w, \langle \alpha_i, \mathrm{Right} \rangle, \langle \alpha_j, \mathrm{Left} \rangle]$. Remaining values r_{ij}, i.e. transitions for states σ_i, σ_j connecting with two different substrings are defined as follows. If $\sigma_i = \langle \alpha_i, h_i \rangle$ and $\sigma_j = \langle \alpha_j, h_j \rangle$ then the transition equals to 0 if h_i and h_j are not connected with two consecutive symbols of the input. Otherwise, e.g. for $h_i = \mathrm{Left}_w$ and $h_j = \mathrm{Right}_{Y_2}$, r_{ij} equals to the the probability that M reaches in one step the memory state α_j starting in α_i and moving the input head according to h_i and h_j. The transition probabilities of $\bar{R}$ are obtained analogously.

The states $\mathrm{Accept}_U, \mathrm{Reject}_U, \mathrm{Alter}_U, \text{R-Loop}_U$, for $U \in \{\$Y_1,\ Y_2,\ w,\ Z_1,\ Z_2\$\}$ as well as all states $\langle \alpha, h \rangle$, with non-probabilistic α, are defined to be trapping states for both R and $\bar{R}$. For any trapping state σ_t the transitions $r_{t,t}$ are defined to be 1.

Let the memory state of C (resp. D) be α (α', resp.). Then the initial state of each chain is $\langle \alpha, \mathrm{Right}_{Y_1} \rangle$. Note that according to the definition of the trapping states, $\langle \alpha', \mathrm{Right}_{Z_1} \rangle$ is a trapping state of both Markov chains.

Now from the fact that $\mathtt{WP}(w)$ and $\mathtt{WP}(\bar{w})$ are componentwise β-close we have that chains R and $\bar{R}$ are β-close. Hence using Lemma 1 we obtain that for the configuration D the probabilities $p^*[\langle \alpha', \mathrm{Right}_{Y_1} \rangle, R]$ and $p^*[\langle \alpha', \mathrm{Right}_{Y_1} \rangle, \bar{R}]$ are β^{2z}-close, what proves (2). ∎

As an immediate corollary of this lemma we obtain that $\mathcal{S}'$ is a strategy of M on the input X_i, too. To estimate $\mathrm{Accept}[\mathcal{S}', X_i]$ first note that for any node C in $\mathcal{S}'$ it holds

$$\mathrm{Reach}[C, \mathcal{S}', X_i] \ \geq\ \beta^{-2z\lceil k/2 \rceil} \mathrm{Reach}[C, \mathcal{S}', X] \ .$$

This follows from Lemma 2 since for the subsequence $C_1, \ldots, C_{r-1}$ of probabilistic configurations on the path from the root C_0 to configuration $C = C_r$, where $r \leq \lceil k/2 \rceil$, it holds:

$$\begin{aligned} \mathrm{Reach}[C, \mathcal{S}', X_i] &= \prod_{0 \leq t < r} \mathrm{Rmove}^*[C_t, C_{t+1}, X_i] \ \geq \prod_{0 \leq t < r} \beta^{-2z} \cdot \mathrm{Rmove}^*[C_t, C_{t+1}, X] \\ &\geq \beta^{-2z\lceil k/2 \rceil} \prod_{0 \leq t < r} \mathrm{Rmove}^*[C_t, C_{t+1}, X] \ = \ \beta^{-2z\lceil k/2 \rceil} \cdot \mathrm{Reach}[C, \mathcal{S}', X] \ . \end{aligned}$$

Adding up we get:

$$\begin{aligned} \mathrm{Accept}[\mathcal{S}', X_i] &= \sum_{C \in \mathcal{S}' \text{ accepting}} \mathrm{Reach}[C, \mathcal{S}', X_i] \ \geq \sum_{C \in \mathcal{S}' \text{ accepting}} \beta^{-2z\lceil k/2 \rceil} \mathrm{Reach}_X[C, \mathcal{S}'] \\ &= \beta^{-2z\lceil k/2 \rceil} \cdot \mathrm{Accept}[\mathcal{S}', X] \ \geq\ \frac{b-k}{b} \cdot \beta^{-2z\lceil k/2 \rceil} \cdot \mathrm{Accept}[\mathcal{S}, X] \ , \end{aligned}$$

which proves the proposition. ∎

This proposition provides the main technical lower bound tool to obtain a sufficient condition for a language not to be recognizable by STMs of low space and round complexity.

Theorem 11 *Let $L \subseteq \Sigma^*$ be a language, and $S \leq o(\log)$. Suppose there exists a constant c_L, a nondecreasing integer functions K, and a sequence of integers $\ell = \ell_1 < \ell_2 < \ell_3 < \ldots$ with $S(\ell_m) \leq o(\log m)$ and $\log K(\ell_m) \leq m^{1/3}$ such that the following holds for every $m \geq 1$:*

> *One can find a set $W_m = \{w_1, w_2, \ldots, w_{2^m}\}$ of 2^m strings each of length at most m^{c_L} with the property:*
> *for every pair $w \neq \bar{w} \in W_m$ there exists $b > \frac{3}{2} K(\ell_m)$ words $u_1, \ldots, u_b \in \Sigma^*$ such that $X := w\ u_1\ w \ldots w\ u_b \in L$ and for all $i \in [1..b]$ $X_i := w\ u_1 \ldots u_{i-1}\ \bar{w}\ u_i \ldots w\ u_b \notin L$.*
> *Furthermore, all words X, X_i have to be of length at most ℓ_m.*

Then $L \notin AM_K Space(S) \cup MA_K Space(S)$.

Proof. Let L fulfill the requirements of the theorem, and choose a set W_m with the properties stated above. Suppose M is an S-space-bounded STM accepting L in with the number of alternations bounded by K. Using a counting argument, similar to the one in [DwSt92], we prove first that for $s := S(\ell_m)$

Lemma 3 *There exist two words w and $\bar{w}$ in W_m such that $\mathtt{WT}_s(w) = \mathtt{WT}_s(\bar{w})$ and that the word probabilities $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(\bar{w})$ are β-close for $\beta := 2^{2^{-\sqrt{m}}}$.*

Proof. Obviously, the number of word probabilities and the number of word transitions of starting and stopping events that are generated by memory states of space bounds at most s are

$$d_p = 4 \cdot (\mathrm{Vol}(s))^2 + 8 \cdot \mathrm{Vol}(s) \qquad \text{and} \qquad d_t = 4 \cdot (\mathrm{Vol}(s))^2 + 4 \cdot \mathrm{Vol}(s) \ , \quad \text{respectively.}$$

Let w be a string in W_m. Observe first that if p is a nonzero element of $\mathtt{WP}_s(w)$, then $p \geq 2^{-\mathrm{Vol}(s) \cdot m^{c_L}}$, thus the interval in which each nonzero coordinate of $\mathtt{WP}_s(w)$ lies is $[2^{-\mathrm{Vol}(s) \cdot m^{c_L}}, 1]$. This claim follows easily from the fact that the shortest computation path of M on w that starts and ends in specific configurations is not longer than $\mathrm{Vol}(s) \cdot |w|$.

Divide W_m into equivalence classes where two words w and w' are equivalent iff $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(w')$ are zero in exactly the same coordinates. Let E_m be the largest equivalence class. Hence $|E_m| \geq |W_m|/2^{d_p}$. Choose $w \in E_m$.

Let $\log \mathtt{WP}_s(w)$ be the vector with the i-th coordinate equals to $\log \mathtt{WP}_s(w)(i)$ if $\mathtt{WP}_s(w)(i) > 0$ and zero otherwise. Hence each coordinate of $\log \mathtt{WP}_s(w)$ lies in the interval $[-\mathrm{Vol}(s) \cdot m^{c_L}, 0]$. We divide each such interval into subintervals of length μ. In this way we divide the space $[-\mathrm{Vol}(s) \cdot m^{c_L}, 0]^{d_p}$ into $(\frac{\mathrm{Vol}(s) \cdot m_{c_L}}{\mu})^{d_p}$ cubes, each of volume μ^{d_p}.

We want to choose μ very small, but large enough such that the number of cubes is smaller than $|E_m|/2^{d_t}$. To guarantee the last condition we require

$$\left(\frac{\mathrm{Vol}(s) \cdot m^{c_L}}{\mu} \right)^{d_p} < \frac{|W_m|}{2^{d_p} 2^{d_t}} \, . \tag{2}$$

Define

$$\mu := 2^{-\sqrt{m}} \, .$$

For this value inequality (2) holds, for sufficiently large m, since

$$2^{8 \cdot (\mathrm{Vol}(s))^2 + 12 \cdot \mathrm{Vol}(s)} \cdot \left(\frac{\mathrm{Vol}(s) \cdot m^{c_L}}{\mu} \right)^{4 \cdot (\mathrm{Vol}(s))^2 + 8 \cdot \mathrm{Vol}(s)} < 2^m \, .$$

Assuming (2) there must be more than 2^{d_t} words w in E_m such that $\log \mathtt{WP}_s(w)$ belongs to the same cube. Let us denote such a subset of E_m by F_m. Therefore for any pair of different words w and w' from F_m, if p and p' are two nonzero probabilities in the same coordinate of $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(w')$, respectively, then

$$|\log p - \log p'| \leq \mu \, .$$

It follows that p and p' are 2^{μ}-close. Therefore $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(w')$ are componentwise 2^{μ}-close. Since there are at most 2^{d_t} different transition vectors, hence there must be two different words $w, \bar{w} \in F_m$ with the same transition vector which proves the fact. ∎

Now, choose $b > \frac{3}{2} K(\ell_m)$ and select strings $u_1, \ldots, u_b$ such that $X = w\ u_1\ w \ldots w\ u_b \in L$ and $X_i = w\ u_1 \ldots u_{i-1}\ \bar{w}\ u_i \ldots w\ u_b \notin L$ for every i, and apply Proposition 1 with $\ell := \ell_m$, $k := K(\ell_m)$. Since $X \in L$ the machine M accepts X with probability at least $3/4$, in other words $\mathrm{Accept}[\mathcal{S}, X] \geq 3/4$ for some full k-round strategy $\mathcal{S}$ on X. From Proposition 1, however, we conclude that there exists an integer $i \in [1..b]$ and a k-round strategy $\mathcal{S}'$ such that

$$\mathrm{Accept}[\ \mathcal{S}', X_i\ ] \geq \frac{b-k}{b} \cdot \beta^{-2z\lceil k/2 \rceil} \cdot \frac{3}{4} \geq \left(\frac{1}{3} + \frac{1}{6K(\ell_m)} \right) \cdot 2^{-2^{-\sqrt{m}} \cdot O(m \cdot K(\ell_m))} \cdot \frac{3}{4} > \frac{1}{4}$$

for m large enough. But, by assumption, the input X_i does not belong to L – a contradiction. ∎

It is easy to check that for any nondecreasing function $S \in o(\log)$ and every constant $k \geq 2$, the languages $\mathtt{PATTERN}_k$ fulfill the assumptions of Theorem 11 if we choose the sequence $\ell = 1, 2, 3, \ldots$, the function $K \equiv \lceil 2k/3 \rceil - 1$, and the set of words

$$W_m \ := \ \{x_1 2 x_2 2 \ldots 2 x_m \mid x_i \in \{0, \mathrm{bin}(i)\}\} \ .$$

Hence Theorem 2 holds.

It is also easy to verify the assumptions of Theorem 11 for the complement of the language $\mathtt{PATTERN}_1$ for any $K \in O(\sqrt{n})$. Now, with W_m defined above we choose $\ell_m = m^6$ and $b := m^4$. Therefore, Theorem 6 holds, too.

By Theorem 2 we have, in particular, that languages $\mathtt{PATTERN}_2$ and $\mathtt{PATTERN}_3$ do not belong to $AM_1Space(o(\log))$ and that $\mathtt{PATTERN}_4 \notin AM_2Space(o(\log))$. Using a different technique than this general one from the proof of Theorem 11, below we show a stronger result for the AM_2Space class. Namely, we prove a lower bound which implies Theorem 1 that $\mathtt{PATTERN}_1$ does not belong to $AM_2Space(o(\log))$.

Theorem 12 *Let $L \subseteq \Sigma^*$ be a language, and $S \leq o(\log)$.*
Suppose there exists a constant c_L and an integer sequence $\ell = \ell_1 < \ell_2 < \ldots$ with $S(\ell_m) \leq o(\log m)$ such that for every m one can find a set $W_m = \{w_1, w_2, \ldots, w_{2^m}\}$ of 2^m strings each of length at most m^{c_L} with the property:
for every pair $w \neq \bar{w} \in W_m$ there exist words u_1 and u_2 in Σ^* such that $\bar{w}u_1wu_2$ and $wu_1\bar{w}u_2$ belong to L, but $\bar{w}u_1\bar{w}u_2 \notin L$, and all these words are of length at most ℓ_m.
Then $L \notin AM_2Space(S)$.

Proof. Let L and m fulfill the requirements of the theorem and suppose M is an S-space-bounded STM accepting L in 2 rounds starting in probabilistic one. Let $\ell := \ell_m$ and define $s := S(\ell)$. Using the same counting argument as to show Fact 5 in the proof of Theorem 11 one can deduce that there exist two words w and $\bar{w}$ in W_m such that $\mathtt{WT}_s(w) = \mathtt{WT}_s(\bar{w})$ and that the word probabilities $\mathtt{WP}_s(w)$ and $\mathtt{WP}_s(\bar{w})$ are β-close, where

$$\beta \ := \ 2^{2^{-\sqrt{m}}} \ .$$

We will show that if M accepts L then

$$\Pr[\ M \text{ accepts } \bar{w}u_1\bar{w}u_2] \ > \ 1/3 \ ,$$

which implies the theorem since $\bar{w}u_1\bar{w}u_2 \notin L$. To prove the above inequality we will show that using strategies $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$ for inputs $\bar{w}u_1wu_2$ and $wu_1\bar{w}u_2$, respectively, one can construct a strategy $\mathcal{S}_{\bar{w}\bar{w}}$ for input $\bar{w}u_1\bar{w}u_2$ such that

$$\mathrm{Accept}[\mathcal{S}_{\bar{w}\bar{w}}, \bar{w}u_1\bar{w}u_2] > 1/3 \ . \tag{3}$$

Let $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$ be strategies such that

$$\mathrm{Accept}[\mathcal{S}_{\bar{w}w}, \bar{w}u_1wu_2] \geq 3/4 \quad \text{and} \quad \mathrm{Accept}[\mathcal{S}_{w\bar{w}}, wu_1\bar{w}u_2] \geq 3/4 \ .$$

Such strategies exist since both inputs belong to L. We will consider nondeterministic configurations of $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$ which terminate in probabilistic rounds. Let us say that such a configuration C is a **1-type** configuration for $\mathcal{S}_{\bar{w}w}$ if machine M being in C with $\bar{w}u_1wu_2$ on the input tape and reads substring $\bar{w}$. Otherwise we will say that C is a **2-type** configuration for $\mathcal{S}_{\bar{w}w}$. Similarly, we call C a 1-type configuration for $\mathcal{S}_{w\bar{w}}$ if M being in C with $wu_1\bar{w}u_2$ on the input tape reading substring w. Otherwise, C is a 2-type configuration for $\mathcal{S}_{w\bar{w}}$.

Let C_0 denote the initial configuration for M. To construct strategy $\mathcal{S}_{\bar{w}\bar{w}}$ we start with the root C_0. Next we use all 1-type configurations for $\mathcal{S}_{\bar{w}w}$ and all 2-type configurations for $\mathcal{S}_{w\bar{w}}$ and define them as sons of C_0. Finally, we describe a successor for any son C of C_0 as follows: if C is 1-type configuration for $\mathcal{S}_{\bar{w}w}$ then the successor of C is a configuration D' such that $\mathrm{Nmove}(C, D', \bar{w}u_1\bar{w}u_2)$ holds and D' is accepting if and only if the successor D of C in $\mathcal{S}_{\bar{w}w}$ is accepting. We define successors of 2-type configurations for $\mathcal{S}_{w\bar{w}}$ analogously (see Fig. 2). We claim that $\mathcal{S}_{\bar{w}\bar{w}}$ is a strategy for input $\bar{w}u_1\bar{w}u_2$ and that the probability that $\mathcal{S}_{\bar{w}\bar{w}}$ is accepting is greater that $1/3$.

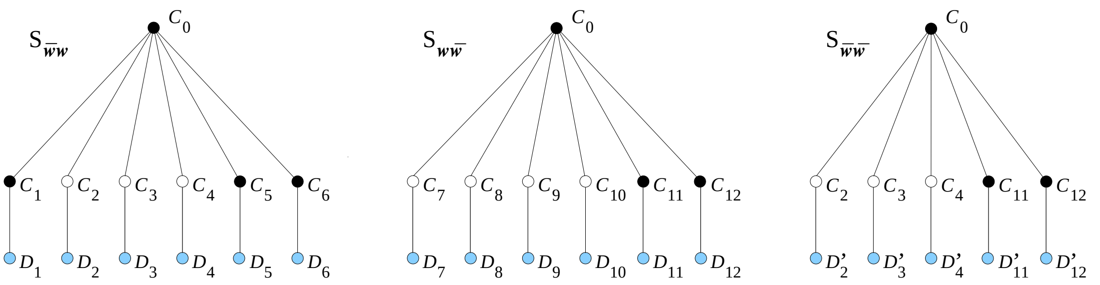

Figure 2: *Construction of strategy* $\mathcal{S}_{\bar{w}\bar{w}}$. *The white nodes of the first level of the above trees describe 1-type configurations and the black ones – 2-type configurations. The input head positions of the terminal configurations (gray nodes) are inessential for our construction. For* $\mathcal{S}_{\bar{w}\bar{w}}$ *we choose configurations* D'_i *in such a way that* D'_i *is accepting if and only if* D_i *is accepting.*

The fact that for any configuration C one can find a configuration D' as described above, follows easily from the property that w and $\bar{w}$ are words with identical word transitions. Hence to prove that $\mathcal{S}_{\bar{w}\bar{w}}$ is a strategy for $\bar{w}u_1\bar{w}u_2$ it remains to show that for any son C of C_0 in $\mathcal{S}_{\bar{w}\bar{w}}$ it holds $\mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2] > 0$. But this property follows from the following lemma that will be used below to prove inequality (3).

Lemma 4 *Let* $z = 10 \cdot \mathrm{Vol}(s) + 20$. *Then*

1. *probabilities* $\mathrm{Rmove}^*[C_0, C, \bar{w}u_1wu_2]$ *and* $\mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2]$ *are* β^{2z}*-close for any 1-type configuration* C *for* $\mathcal{S}_{\bar{w}w}$ *and*
2. *probabilities* $\mathrm{Rmove}^*[C_0, C, wu_1\bar{w}u_2]$ *and* $\mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2]$ *are* β^{2z}*-close for any 2-type configuration* C *for* $\mathcal{S}_{w\bar{w}}$.

It is easy to check that the proof of Lemma 2(2) works for the lemma, too, hence we omit it here.

Let us say that a son C of C_0 in $\mathcal{S}_{\bar{w}\bar{w}}, \mathcal{S}_{\bar{w}w}$, or $\mathcal{S}_{w\bar{w}}$ is accepting if the successor of C is accepting. Note that for any C in $\mathcal{S}_{\bar{w}\bar{w}}$ C is accepting if and only if it is accepting in $\mathcal{S}_{\bar{w}w}$ or $\mathcal{S}_{w\bar{w}}$. We have

$$\mathrm{Accept}[\mathcal{S}_{\bar{w}\bar{w}}, \bar{w}u_1\bar{w}u_2] = \sum_{\substack{C \text{ son of } C_0 \text{ in } \mathcal{S}_{\bar{w}\bar{w}} \\ \text{and accepting}}} \mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2] .$$

Analogous equations hold for strategies $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$. We partition now the sums for considered strategies according to the type of configurations C. Let

$$P_1[\mathcal{S}_{\bar{w}\bar{w}}] := \sum_{\substack{C \text{ son of } C_0 \text{ in } \mathcal{S}_{\bar{w}\bar{w}}, \\ 1\text{-type and accepting}}} \mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2]$$

$$P_2[\mathcal{S}_{\bar{w}\bar{w}}] := \sum_{\substack{C \text{ son of } C_0 \text{ in } \mathcal{S}_{\bar{w}\bar{w}}, \\ 2\text{-type and accepting}}} \mathrm{Rmove}^*[C_0, C, \bar{w}u_1\bar{w}u_2] .$$

Analogously we define $P_1[\mathcal{S}_{\bar{w}w}], P_2[\mathcal{S}_{\bar{w}w}], P_1[\mathcal{S}_{w\bar{w}}]$, and $P_2[\mathcal{S}_{\bar{w}w}]$. By Lemma 4 and by the fact that $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$ are accepting with probabilities greater than $3/4$ we have

$$\begin{aligned}
\mathrm{Accept}[\mathcal{S}_{\bar{w}\bar{w}}, \bar{w}u_1\bar{w}u_2] &= P_1[\mathcal{S}_{\bar{w}\bar{w}}] + P_2[\mathcal{S}_{\bar{w}\bar{w}}] \\
&\geq \beta^{-2z} \cdot (P_1[\mathcal{S}_{\bar{w}w}] + P_2[\mathcal{S}_{w\bar{w}}]) \\
&\geq \beta^{-2z} \cdot (6/4 - P_2[\mathcal{S}_{\bar{w}w}] - P_1[\mathcal{S}_{w\bar{w}}])
\end{aligned}$$

To estimate $P_2[\mathcal{S}_{\bar{w}w}] + P_1[\mathcal{S}_{w\bar{w}}]$ we consider a tree $\mathcal{S}_{ww}$ which we construct in a very similar way to $\mathcal{S}_{\bar{w}\bar{w}}$ using the strategies $\mathcal{S}_{\bar{w}w}$ and $\mathcal{S}_{w\bar{w}}$. The root of $\mathcal{S}_{ww}$ is C_0. The sons of C_0 are all 2-type configurations for $\mathcal{S}_{\bar{w}w}$ and all 1-type configurations for $\mathcal{S}_{w\bar{w}}$. A successor for any son C of C_0 is defined as follows: if C is 2-type configuration for $\mathcal{S}_{\bar{w}w}$ then the successor of C is a configuration D' such that $\mathrm{Nmove}(C, D', wu_1wu_2)$ holds and D' is accepting if and only if the successor D of C in $\mathcal{S}_{\bar{w}w}$ is accepting. Successors of 1-type configurations for $\mathcal{S}_{w\bar{w}}$ are defined analogously (see Fig. 3). We claim that $\mathcal{S}_{ww}$ is a strategy for input wu_1wu_2.

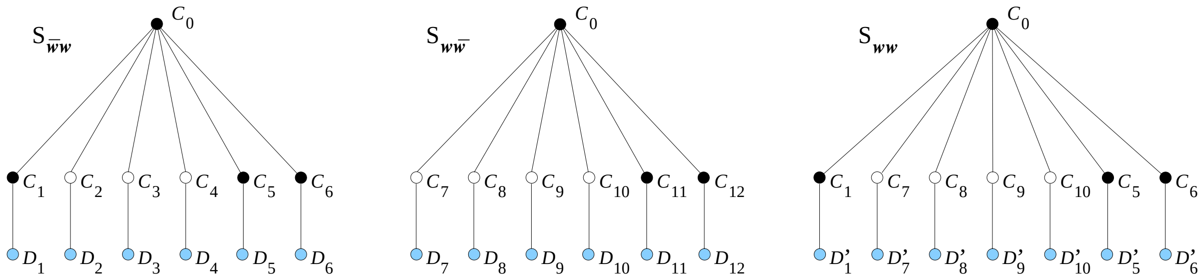

Figure 3: *Construction of strategy $\mathcal{S}_{ww}$. As previously, the white nodes of the first level of the trees describe 1-type configurations and the black ones – 2-type configurations. For $\mathcal{S}_{ww}$ we choose configurations D'_i such that D'_i is accepting if and only if D_i is accepting.*

The existence of configurations D' follows, as in the case of $\mathcal{S}_{\bar{w}\bar{w}}$, from the fact that w and $\bar{w}$ are words with identical word transitions. Similarly, we have that

- probabilities $\mathrm{Rmove}^*[C_0, C, \bar{w}u_1wu_2]$ and $\mathrm{Rmove}^*[C_0, C, wu_1wu_2]$ are β^{2z}-close for any 2-type configuration C for $\mathcal{S}_{\bar{w}w}$, and
- probabilities $\mathrm{Rmove}^*[C_0, C, wu_1\bar{w}u_2]$ and $\mathrm{Rmove}^*[C_0, C, wu_1wu_2]$ are β^{2z}-close for any 1-type configuration C for $\mathcal{S}_{w\bar{w}}$,

what can be proved in the same way as Lemma 2(2). Next define

$$P_1[\mathcal{S}_{ww}] := \sum_{\substack{C \text{ son of } C_0 \text{ in } \mathcal{S}_{ww}, \\ 1-\text{type and accepting}}} \mathrm{Rmove}^*[C_0, C, wu_1wu_2]$$

$$P_2[\mathcal{S}_{ww}] := \sum_{\substack{C \text{ son of } C_0 \text{ in } \mathcal{S}_{ww}, \\ 2-\text{type and accepting}}} \mathrm{Rmove}^*[C_0, C, wu_1wu_2] .$$

Then we conclude

$$\begin{aligned}
\mathrm{Accept}[\mathcal{S}_{\bar{w}\bar{w}}, \bar{w}u_1\bar{w}u_2] &\geq \beta^{-2z} \cdot (6/4 - \beta^{2z} \cdot P_2[\mathcal{S}_{ww}] - \beta^{2z} \cdot P_1[\mathcal{S}_{ww}]) \\
&= \beta^{-2z} \cdot (6/4 - \beta^{2z}(\mathrm{Accept}[\mathcal{S}_{ww}, wu_1wu_2])) \geq \beta^{-2z} \cdot 6/4 - 1 .
\end{aligned}$$

But for the chosen value β and sufficiently large m we have

$$\beta^{-2z} \cdot 6/4 - 1 \;=\; 2^{-\mu\cdot(28Vol(s)+8)} \cdot 6/4 - 1 \;>\; 1/3\ .$$

what proves the inequality(3). ∎

It is easy to verify that the assumptions of Theorem 12 are fulfiled for $\mathtt{PATTERN}_1$ and any nondecreasing function $S \in o(\log)$ and the set of words

$$W_m := \{x_1 2 x_2 2 \ldots 2 x_m \mid x_i \in \{0, \mathrm{bin}(i)\}\}\ .$$

This proves Theorem 1.

6 The Upper Bound for the Pattern Languages

In this section we show that to accept $\mathtt{PATTERN}_k$ according to Theorem 3, one can apply the following stochastic algorithm:

First, check deterministically whether the input is of the form $U\#u\#\mathtt{BIN}(2^d)$, for some words $U, u \in \{0,1,2\}^+$, and whether $|u| = d$.
Reject and stop if this condition does not hold.
Otherwise, move the input head to the left end-marker, set $i := 1$ and execute the following loop.

1. Moving the input head to the right, nondeterministically choose a starting position for a substring u_i of length d in U and interpret u_i and u as natural numbers between 0 and $2^d - 1$.
2. Randomly choose a prime $q_i \in [2..d^2]$, compute $r_i := u_i \bmod q_i$ and store both q_i and r_i.
3. If $i = k$ or if the whole string U has been scanned, go to step 4. Otherwise go to the right end of u_i, increase i by 1 and go to step 1.
4. If $i = k$ and for all $j \in [1..k]$ the equality $r_j = u \bmod q_j$ holds then accept, otherwise reject.

Obviously, a stochastic Turing machine which performs the algorithm uses $O(\mathrm{llog})$ space and works in $2k$ rounds.

For the padded languages $\mathtt{PATTERN}_k(i)$, with $i \geq 1$, one can show an upper bound described in Theorem 9 by combining the above algorithm for the unpadded languages with the following

Lemma 5 ([Fr81]) *Let*

$$L := \{b_0 a^{2^0} b_1 a^{2^1} b_2 a^{2^2} \ldots b_i a^{2^i} \ldots b_n a^{2^n} \mid b_i \in \Sigma \ \text{ for } \ i = 1, \ldots, n \ \text{ and } \ a \notin \Sigma\}\ .$$

Then for arbitrary $\epsilon > 0$ there is a probabilistic finite automaton recognizing the language L with probability at least $1 - \epsilon$.

7 Conclusions and Open Problems

Considering the infinite hierarchy for the sublogarithmic $AM_k Space$ classes, an interesting open problem is whether the distance given by the factor 3 can be reduced. One is tempted to conjecture that a maximal separation

$$AM_k Space(S) \overset{?}{\subset} AM_{k+1} Space(S)$$

holds for any integer k and any sublogarithmic space bound S. Unfortunately, standard translation techniques do not work for these classes.

However, the lower and upper bounds for the languages $\mathtt{PATTERN}_k$ can be improved. If we insist on a quite small error probability then the lower bound for the number of rounds can be sharpened to $k - O(1)$. On the other hand, for a simple structured sublanguage of $\mathtt{PATTERN}_k$, for which the lower bound still holds, an upper bound of $5/3\,k$ rounds can be obtained, and relaxing the error probability to $1/2-\epsilon$ this can further be reduced to something like $(1+\delta)k$ for some small δ that depends on ϵ. These methods, however, do not seem to lead to the ultimate goal k versus $k+1$.

How looks the round/alternation hierarchy for at least logarithmic space bounds? Using a simple simulation of space-bounded NTMs by one-sided-error probabilistic TMs (see e.g. [Gil77] or the survey paper [Ma95]) one can easily show that

$$\mathcal{NL} = MA_1 Space(\log) \subseteq AM_2 Space(\log) = AM_1 Space(\log) = \mathcal{BPL} ,$$

where $\mathcal{BPL}$ denotes the probabilistic bounded error logarithmic space class. This means that the AM_2-class is quite weak in case of space bounds – just opposite to the time bounded case. Is it also true that

$$AM_2 Space(\text{SUBLOG}) \stackrel{?}{=} AM_1 Space(\text{SUBLOG}) ,$$

or at least $\quad MA_1 Space(\text{SUBLOG}) \stackrel{?}{\subseteq} AM_1 Space(\text{SUBLOG})$,
which in the standard notation reads as

$$NSpace(\text{SUBLOG}) \stackrel{?}{\subseteq} BPSpace(\text{SUBLOG}) .$$

What is the situation for constant space bounds? It has been well known that $MA_1 Space(\text{CON}) = NSpace(\text{CON})$ coincides with the class of regular languages. As we have already discussed at the beginning this result does not extend to the probabilistic classes because of Freivalds' algorithm for the language `COUNT`. However, it requires exponential expected time and this cannot be avoided. Is there a language that for some $k > 1$ separates probabilistic from k-round stochastic automata, that means

$$AM_1 Space(\text{CON}) \stackrel{?}{\subset} AM_k Space(\text{CON}) .$$

From [DwSt92] we know that `CENTER` does not belong to $AM_1 Space(\text{CON})$ and that there exists a constant space Arthur-Merlin game for this language. It, however, requires exponential expected time and a similar number of alternations. Since it is also shown that the time complexity of an stochastic automata recognizing `CENTER` cannot be polynomial it seems unlikely that the number of rounds required for this language can be improved to a bounded number.

References

[Ba85] L. Babai, *Trading Group Theory for Randomness,* Proc. 17. ACM Symp. on Theory of Computing, 1985, 421-429.

[BGR93] B. von Braunmühl, R. Gengler, R. Rettinger *The Alternation Hierarchy for Machines with Sublogarithmic Space is Infinite,* Comp. Compl. 3, 1993, 207-230.

[CKS81] A. Chandra, D. Kozen, L. Stockmeyer, *Alternation,* J. ACM 28, 1981, 114-133.

[Co89] A. Condon, *Computational Model of Games,* MIT Press, 1989.

[Co93] A. Condon, *The Complexity of Space Bounded Interactive Proof Systems,* in *Complexity Theory: Current Research,* S. Homer, U. Schöning, K. Ambos-Spies (Eds.), Cambridge Univ. Press, 1993, 147-190.

[CoLi89] A. Condon, R. Lipton, *On the Complexity of Space Bounded Interactive Proofs,* Proc. 30. IEEE Symp. on Found. of Comp. Science, 1989, 462-467.

[CHPW94] A. Condon, L. Hellerstein, S. Pottle, A. Wigderson, *On the Power of Finite Automata with Both Nondeterministic and Probabilistic States,* Proc. 26. ACM Symp. on Theory of Computing, 1994, 676-685.

[DwSt92] S. Dwork, L. Stockmeyer, *Finite State Verifiers I: the Power of Interaction,* J. ACM 39, 1992, 800-828.

[Fr79] R. Freivalds, *Fast Probabilistic Algorithms,* Proc. 8. Int. Symp. on Math. Found. of Comp. Science, 1979, LNCS, 57-69.

[Fr81] R. Freivalds, *Probabilistic Two-way Machines,* Proc. 10. Int. Symp. on Math. Found. of Comp. Science, 1981, LNCS, 33-45.

[FrKa94] R. Freivalds, M. Karpinski, *Lower Space Bounds for Randomized Computation,* Proc. 21. EATCS Int. Colloq. on Automata, Languages, and Programming, 1994, LNCS, 580-592.

[Ge94] V. Geffert, *A Hierarchy that Does not Collaps: Alternation in Low Level Space,* Theo. Information and Applications 28, 1994, 465-512.

[GrWe86] A. Greenberg, A. Weiss, *A Lower Bound for Probabilistic Algorithms for Finite State Machines,* J. Comput. Syst. Sci. 33, 1986, 88-105.

[Gil77] J. Gill, *Computational Complexity of Probabilistic Turing Machines,* SIAM J. Computing 7, 1977, 675-695.

[GMR89] S. Goldwasser, S. Micali, C. Rackoff, *The Knowledge Complexity of Interactive Proof Systems,* SIAM J. Computing 18, 1989, 186-208.

[GoSi86] S. Goldwasser, M. Sipser, *Private Coins versus Public Coins in Interactive Proof Systems,* Proc. 18. ACM Symp. on Theory of Computing, 1986, 59-68.

[Im88] N. Immerman, *Nondeterministic Space Is Closed under Complementation,* SIAM J. Computing 17, 1988, 935-938.

[Iw93] K. Iwama, *ASpace* $(o(\log\log n))$ *is Regular,* SIAM J. Computing 22, 1993, 136-146.

[Li96] M. Liśkiewicz, *Interactive Proof Systems with Public Coins: Lower Space Bounds and Hierarchies of Complexity Classes,* ICSI Technical Report 1996, also Proc. 14. GI-AFCET Symp. on Theo. Aspects of Comp. Science, 1997, LNCS 1200,129-140.

[LiRe93] M. Liśkiewicz, R. Reischuk, *Separating the Lower Levels of the Sublogarithmic Space Hierarchy,* Proc. 10. GI-AFCET Symp. on Theo. Aspects of Comp. Science, 1993, LNCS, 16-27.

[LiRe96a] M. Liśkiewicz, R. Reischuk, *The Sublogarithmic Alternating Space World,* SIAM J. Computing 24, 1996, 828-861.

[LiRe97a] M. Liśkiewicz and R. Reischuk, *Computing with Sublogarithmic Space,* in *Complexity Theory Retrospective II,* A. Selman, L. Hemaspaandra (Eds), Springer Verlag, 1997.

[LiRe97b] M. Liśkiewicz and R. Reischuk, *Computational Limitations of Stochastic Turing Machines and Arthur-Merlin GAmes with Small Space Bounds,* Proc. 22. MFCS, 1997, LNCS 1295, 91-107.

[Ma95] I. Macarie, *Space-bounded Probabilistic Computation: Old and New Stories,* SIGACT News 26, 1995, 2-12.

[Pa85] C. Papadimitriou, *Games against Nature,* J. CSS 31, 1985, 288-301.

[Sh92] A. Shamir, *IP=PSPACE,* J. ACM 39, 1992, 869-877.

[Sz88] R. Szelépcsenyi, *The Method of Forced Enumeration for Nondeterministic Automata,* Acta Informatica 26, 1988, 279-284.