On the Monte Carlo Space Constructible Functions and Separation Results for Probabilistic Complexity Classes (Revised Version)

MAREK KARPINSKI AND RUTGER VERBEEK
UNIVERSITY OF BONN

Abstract It is proven that contrary to the deterministic and nondeterministic cases there is no recursive lower bound for Monte Carlo space constructible functions. The existence of small constructible bounds enables the separation of Monte Carlo space $f(n)$ from probabilistic space $g(n)$ ($f(n) = o(g(n))$) and – together with a new halting lemma for probabilistic machines with small space bounds – a hierarchy of "provable" Monte Carlo space classes with small bounds. We are also able to separate $O(\log\log n)$ terminating Monte Carlo space in the sense of [AKLLR 79], [We 83] from $NSPACE(\log\log n)$.

0. Introduction

It is known from [LSH 65] that any meaningful padding-control requires at least $\log\log n$ space. Therefore there is 'no life' for deterministic machines between $O(1)$ and $O(\log\log n)$.

In contrast to this Freivalds [Fr 81a] displayed exponential (and therefore arbitrary elementary recursive) padding doable in $O(1)$ Monte Carlo space. We prove that even arbitrary recursive paddings are achievable within $O(1)$ Monte Carlo space (Theorem 1). This enables the proofs of separation results for complexity classes with arbitrary small bounds.

For the definition of *probabilistic Turing machines (PTMs)* see [Gi 77]. If $\mathcal{M}$ is a PTM, then $\Phi_{\mathcal{M}}$ is the function computed by $\mathcal{M}$. f is in *probabilistic* $SPACE(s(n))$ ($PrSPACE(s(n))$) if there is a PTM $\mathcal{M}$ such that $\Phi_{\mathcal{M}} = f$ and for all x

$$Pr\{\mathcal{M} \text{ uses on input } x \text{ at most } s(|x|) \text{ space and outputs } \Phi_{\mathcal{M}}(x)\} > \frac{1}{2}.$$

If in addition $\mathcal{M}$ always stops (stronger condition than terminating with probability 1!), then $f \in Pr^T SPACE(s(n))$ [AKLLR 79], [We 83]. If in the definition above $\frac{1}{2}$ is replaced by $\frac{3}{4}$, we call the corresponding machines *Monte Carlo Turing machines (MTMs)*. The Monte Carlo space complexity classes are denoted by $MSPACE(s(n))$ and $M^T SPACE(s(n))$, respectively. Sets are recognized by PTMs or MTMs computing their characteristic functions.

$RSPACE(s(n))$ and $R^T SPACE(s(n))$ (for sets only) are defined in the same way as the Monte Carlo classes, with the restriction that the error probability is 0 on the complement of the recognized set.

A function $f : I\!N \to I\!N$ will be called *Monte Carlo (MC-)constructible* if there is a MTM $\mathcal{M}$ with space bound $f(n)$ for which for all n there is some $x, |x| = n$, such that $\Phi_{\mathcal{M}} = f(n)$. If $\mathcal{M}$ satisfies the above for $x = o^n$, then f is called *fully MC-constructible.*

The notions of MC^T-constructibility and full MC^T-constructibility will correspond to the class of (always) terminating MTMs.

1. Monte Carlo $O(1)$ Space Simulation of Deterministic Counters and Small MC-Constructible Functions

We use two machine models

(i) off line two counter Turing machines (2CTs) [HU 79, p.171] and
(ii) classical (unary input) three counter (Minsky) machines (3CMs) [Mi 61].

A configuration of a 3CM $\mathcal{M}$ is to be encoded in the form $0^{q+1}1^{z_1}2^{z_2}3^{z_3}$, where q is the state of $\mathcal{M}$, z_i is the content of the i-th counter for $i = 1, 2, 3$. The code of a computation is a sequence of encoded $\mathcal{M}$-configurations according to its transition table. 3CMs are able to compute all partial recursive functions (with unary input/output) [Mi

61], whereas 2CMs are not [Ba 62]. In the case of 2CTs, configurations are encoded by $0^{q+1}1^{z_1}2^{z_2}$ (note that we do not mind the input). 2CTs are able to compute all p.r. functions (input/output binary) [HU 79].

Given 3CM M and an accepted input n, $comp_M(n)$ will denote the code of the accepting computation on n. If n is not accepted, $comp_M(n)$ is undefined. In the same way $comp_T(x)$ is defined for 2CT T.

Lemma 1 For every 3CM M, $\{comp_M(n) \mid n \in I\!N\} \in MSPACE(O(1))$.

PROOF The recognition of the set $\{comp_M(n)\}$ is based on the idea of Freivalds' example $\left\{0^{2^0}10^{2^1}1\cdots0^{2^k}1 \mid k \in I\!N\right\}$ [Fr 81a] (cf. also Lemmas 1 and 2 of [Fr 81a]) used for the exponential padding. A deterministic finite automaton can check whether the sequence of states is correct (the next state depends only on the zero-tests and the current state). What remains is to compare the (non-zero) contents of the counters in succeeding configuration by a sequence of tests of roughly the form "Is $n = m$ in $1^n0^*1^m$?" (the differences +1 or -1 can be handled in the finite control). These tests are performed by tossing $8n$ coins on 1^n and $8m$ coins on 1^m. This procedure is repeated until two times the outcomes of all the $8n$ or $8m$ tosses were 'heads'. If this happens both times on the same substring, decide '$n \neq m$'; otherwise decide '$n = m$'. If $n = m$, $Pr\{\text{outcome is '}n = m\text{'}\} = \frac{1}{2}$; if $n \neq m$, $Pr\{\text{outcome is '}n = m\text{'}\} \leq 1 - \left(\frac{1}{1+2^{-8}}\right)^2 \leq 2^{-7}$.

Thus, in a sequence of ℓ tests the probability that all tests give the result

$$\text{'}n_i = m_i\text{' is } \begin{cases} 2^{-\ell} & \text{if } n_i = m_i \text{ for all } i, 1 \leq i \leq \ell \\ \leq 2^{-\ell-6} & \text{otherwise.} \end{cases}$$

Thus we must "compare" $Pr\{\text{all } \ell \text{ tests have outcome '}n = m\text{'}\}$ with $2^{-\ell}$ or, better, with $2^{-\ell-3}$. This is done in a way similar to the single comparisons:

```
begin repeat
            t1 := true; t2 := true;
            for i := 1 to ℓ do
            begin  compare n_i with m_i;
                   if 'n_i ≠ m_i' then  t1 := false;
                   toss a coin;
                   if  the outcome is 'tail' then t2 := false;
            end;
            toss 3 coins;
            if  one of them is 'tail' then  t2 := false;
      until  t1 or  t2;
      if  t1 then  write ('(∀i)n_i = m_i')
           else   write ('(∃i)n_i ≠ m_i')
end
```

Observe that this algorithm requires finite storage only (using a two way input tape, i and ℓ need not be stored).

The probability analysis is quite simple:

If $(\forall i) n_i = m_i$, then $Pr\{\text{answer is } \neq\} \leq \frac{2^{-(\ell+3)}}{2^{-\ell}} < \frac{1}{4}$,

if $(\exists i) n_i \neq m_i$, then $Pr\{\text{answer is } =\} \leq \frac{2^{-\ell-6}}{2^{-(\ell+3)}} < \frac{1}{4}$. □

Lemma 1 entails the following corollaries:

Corollary 1 Every unary r.e. set is homomorphic image of some set from $MSPACE(O(1))$. □

Corollary 2 ([Fr 81b]) For every r.e. set X, there is a set $Y \leq \Sigma^* \times \Sigma^*$ recognizable by an $O(1)$ space bounded MTM with two input tapes, and X is its first projection.

PROOF Suppose $\mathcal{T}$ is a 2CT recognizing X, $Y := \{(x, y) \mid x \in X, y = comp_{\mathcal{T}}(x)\}$. A MTM with x on one input tape and $comp_{\mathcal{T}}(x)$ on the other can clearly verify if $y = comp_{\mathcal{T}}(x)$. □

Corollary 3 ([Fr 81a], Algorithmic Problems 1 and 2) The emptiness (and everywhere acceptance) problem for Monte Carlo two-way finite automata are both Π_1^0-complete. □

Lemma 2 For every recursive function $f : \mathbb{N} \to \mathbb{N}$ there is a 3CM $\mathcal{M}$ such that for all n , $f(n) \leq |comp_{\mathcal{M}}(n)| \leq |comp_{\mathcal{M}}(n+1)|$.

PROOF Given recursive f, there is a 3CM $\mathcal{M}_1$ computing f. We construct a 3CM $\mathcal{M}$, computing $f' : n \mapsto f(0)\, f(1) \cdots f(n)$ in a canonical way. Then for all n, $f(n) \leq f'(n) \leq |comp_{\mathcal{M}}(n)| \leq |comp_{\mathcal{M}}(n+1)|$. □

Theorem 1 For every unbounded nondecreasing (*u.nd.*) recursive function f there is an u.nd. MC-constructible minorant g with $g(n) \leq f(n)$ for all n.

PROOF Given $f : \mathbb{N} \to \mathbb{N}$, $F(n) := \max\{m \mid f(m) \leq n\}$. Take 3CM $\mathcal{M}$ of Lemma 2 for the function F, i.e. satisfying $F(n) \leq |comp_{\mathcal{M}}(n)| \leq |comp_{\mathcal{M}}(n+1)|$.

Define $g(n) = \min\{m \mid |comp_{\mathcal{M}}(m)| \geq n\}$. Construct by Lemma 1 a MTM with space $O(1)$ that recognizes all strings having a prefix of the form $comp_{\mathcal{M}}(m)$ and outputs m for such a word and 0 otherwise. Obviously on input w the output is at most $g(|w|)$. For the input of the form $comp_{\mathcal{M}}(g(n))0^{n-|comp_{\mathcal{M}}(g(n))|}$ $\mathcal{T}$ computes exactly $g(n)$. □

The function g constructed above has an important predictability property: for all n there is an x, such that the MTM constructing g either outputs $g(n)$ (with probability $> \frac{3}{4}$) or outputs 0. We call such a function *predictably MC-constructible.*

2. Probabilistic Machines Halting with Probability 1

For space bounds $s(n)$ greater than $\log n$ there is no problem with halting: a probabilistic clock can switch off every computation longer than $2^{2^{c \cdot s(n)}}$ with high probability.

Then the modified machine (with clock) computes the same function with almost the same probability and halts with probability 1. In the case of smaller space bounds this method is not available since the number of configurations is not bounded by $2^{O(s(n))}$ but by $n \cdot 2^{O(s(n))}$, and thus computations of length $2^{n \cdot 2^{O(s(n))}}$ may be relevant.

On the other hand, for Monte Carlo machines which diverge with probability > 0, we cannot apply the "majority vote" technique for decreasing the probability of error. Thus we have to develop a new method which switches off cycles at the cost of double exponential increase of the space bound (which makes sense for constant space and very small non-constant bounds).

Theorem 2 For every PTM $\mathcal{M}$ there is a PTM $\mathcal{M}'$ and $c \in \mathbb{N}$ such that for all x, y, z

$Pr\{\mathcal{M} \text{ with input } x \text{ outputs } y \text{ within space } z\} \leq$

$Pr\{\mathcal{M}' \text{ with input } \langle x, z\rangle \text{ outputs } y \text{ within space } 2^{2^{c \cdot z}}\}$

and $Pr\{\mathcal{M}' \text{ with input } \langle x, z\rangle \text{ stops }\} = 1$.

PROOF Given a PTM $\mathcal{M}$ with working alphabet Σ and set of states Q, and given an input x, the set of storage configurations is $SC(\mathcal{M}, x) = Q \times \Sigma^z$. The set of all IDs of $\mathcal{M}$ on input x is given by $ID(\mathcal{M}, x) = SC(\mathcal{M}, x) \times \{1, \cdots, |x|\} = SC(\mathcal{M}, x) \times$ {positions of the input head}. For $\alpha, \beta \in ID(\mathcal{M}, x)$ we write $\alpha \vdash \beta$, meaning that $Pr\{ID\,\alpha \text{ goes to } ID\,\beta \text{ in one step}\} > 0$. In the same way we define $\alpha \overset{*}{\vdash} \beta$ if at least one step is involved. β is said to be *halting* if $\beta \vdash \beta$ and for all $\gamma \neq \beta$, $\beta \not\vdash \gamma$. We say β is a *trap* if for all halting IDs γ, $\beta \overset{*}{\not\vdash} \gamma$.

The decision '$x \in X$' is uniquely determined by the halting (accepting) IDs (which produce the output 'yes'), whereas the decision '$x \notin X$' can implicitly be made by going into a trap. In what follows we construct a symmetrical machine identifying the set X by halting with output 'yes' or 'no'. (It is known by [Sim 81] that this is possible above $\log n$ without changing the space bound. In the deterministic case, this is possible even below $\log n$ [Sip 80]. We do not know whether the $2^{2^{O(f(n))}}$ space bound of our construction can be improved.)

Given a position i on the input x of length n, the left table of $\mathcal{M}$ for position i ($1 \leq i \leq n$) of $\mathcal{M}$ on x $LT(i) \subseteq SC(\mathcal{M}, x) \times (SC(\mathcal{M}, x) \cup \{*\})$ is defined by

$(a, b) \in LT(i) \iff Pr\{(a, i) \text{ goes to } (b, i) \text{ without visiting } (i+1)\text{st square}\} > 0.$

$(a, *) \in LT(i) \iff Pr\{(a, i) \text{ goes to halting ID without visiting } (i+1)\text{st square}\} > 0.$

$RT(i)$ is defined in the same way by replacing '$(i+1)$st square' by '$(i-1)$st square'. It is easily seen that the following is doable deterministically in space $2^{O(z)}$:

- store a fixed number of tables,
- compute $LT(1)$ from x_1 or $RT(n)$ from x_n,
- compute $LT(i+1)$ from $LT(i)$, x_i, and x_{i+1} $(1 \leq i < n)$
  or $RT(i-1)$ from $RT(i)$, x_i, and x_{i-1} $(1 < i \leq n)$,
- decide from $LT(i)$ and $RT(i)$ whether (a, i) is a trap.

Suppose e.g. that $LT(i)$ is stored, the machine scans the ith square. Store a pair (a, b) in $LT(i+1)$ if one of the following is true (start with $LT(i+1) = \emptyset$):
- $a = b$,
- $(a, i+1) \vdash (b, i+1)$,
- $(a, i+1) \vdash (a', i)$, $(b', i) \vdash (b, i+1)$, $(a', b') \in LT(i)$,
- $(a, a') \in LT(i+1)$ and $(a', b) \in LT(i+1)$,
- $b = *$ and a is halting,
- $b = *$, $(a, i+1) \vdash (a', i)$, $(a', *) \in LT(i)$.

Thus it is possible (starting at the right end of the input) to compute successively $RT(n)$, RT(n-1), $\cdots$, $RT(1)$, $LT(1)$, using $2^{O(s)}$ space.

What remains is to show that
- $LT(i-1)$ can be computed from $LT(i)$ and the input (without losing the position),
- $RT(i+1)$ can be computed from $RT(i)$.

This is a bit difficult, since the position cannot be stored. The following procedure depends on a trick of Hopcroft and Ullman [HU 67] (we describe it only for the left tables LT):

(0) **If** there is (depending on the input x_{i-1} and x_i) only one possible predecessor for $LT(i)$ (i.e. only one possible table $LT(i-1)$)
then stop
else let $T_1, \cdots, T_k$ $(k > 1)$ be the possible predecessors of $LT(i)$; $\mathcal{T}_j := \{T_j\}$ for $j = 1, \cdots, k$; **goto** (1)

(1) Move one step left (on the input) and determine $\mathcal{T}'_j$ = set of all the possible predecessors of tables in $\mathcal{T}_j$ for $j = 1, \cdots, k$; **goto** (2)

(2) **If** the left boundary (square 1) is reached
then determine the j_0 such that $LT(1) \in \mathcal{T}'_{j_0}$; **goto** (3)
else if only $\mathcal{T}'_{j_0}$ is nonempty
then goto (3)
else $\mathcal{T}_j := \mathcal{T}'_j$ for $j = 1, \cdots, k$; **goto** (1)

(3) Choose two tables T, T' from different (nonempty) subsets $\mathcal{T}$; move right and compute the successors until these become equal. Then the desired table is T_{j_0} and the machine scans square i.

Note that the (right) successors of left tables are unique and hence the sets $\mathcal{T}_j$ are pairwise disjoint. Thus j_0 is uniquely determined and i is in fact the first position such that the successors of T, T' converge. Since sets of tables must be stored, the space is $2^{2^{O(s)}}$.

Now we are ready to start our terminating (with probability 1) simulation. Denote the simulating PTM by $\bar{M}$. $\bar{M}$ is to simulate M step by step keeping the exact record of left and right tables at the current position. At each step M recomputes both tables at the new position. Whenever $\bar{M}$ detects that the current configuration is a trap, it halts and outputs 'no'. Obviously $\bar{M}$ halts with probability 1.

To go ahead we need to exclude the case that the probabilities for the correct and wrong anwers are equal. The standard technique of [Gi 77] and [Sim 81] guarantees

that for every PTM M there is effectively M' with probability of all answers $\neq \frac{1}{2}$ and working within the same space boundaries. If moreover M terminates with probability 1 and computes a (partial) 0–1 valued function, then M' computes the characteristic function of the recognized set of M. □

The main application of Theorem 2 in this paper is Theorem 5 in section 3. – Below we give another application for probabilistic finite two-way automata:

Theorem 3 For every probabilistic finite two-way automaton (PFA) M with s states recognizing a set X with error probability $e(x) (< \frac{1}{2}$, if $x \in X)$ there is a PFA M' with error probability $\leq e(x)$ computing the characteristic function of X and stopping with probability 1. M' has $2^{2^{s^k}}$ states for an appropriate k.

PROOF Take the machine M' constructed in the proof of Theorem 2 and modify it so that it outputs '$x \notin X$' whenever a trap is detected. □

Corollary 4 The sets recognized by PFAs are closed under the complement. □

Corollary 5 The sets recognized by Monte Carlo finite two-way automata form a Boolean algebra. □

Nothing is known about the lower bounds for complementation for both the probabilistic and Monte Carlo automata – as it is known for non-deterministic automata (one-exponential function).

3. Separation Results for Small Space Classes

Unlike the deterministic case the existence of chains of constructible space bounds does not guarantee the existence of a Monte Carlo small space hierarchy. The reason for this anomaly is that we do not know whether there is a universal *Monte Carlo* simulator for all Monte Carlo machines working in the smaller bound, that itself works in the greater bound.

It is not sure whether for every function in $MSPACE(f)$ there exists a machine that is explicitly Monte Carlo and f-space bounded. On the other hand, diagonalization over f-bounded probabilistic machines seems not to be possible when the bound is merely MC-constructible. What we can do, however, is to diagonalize over all f-bounded Monte Carlo machines using a g-bounded probabilistic machine, provided g is sufficiently greater than f.

Theorem 4 Suppose g is predictably MC-constructible and $f = o(g)$.
Then $MSPACE(f) \subsetneq PrSPACE(g)$.

PROOF We construct a PTM M working in space $O(g)$ such that $\varphi_M \notin MSPACE(f)$: Suppose the input of M is divided in two tracks containing x_1 and x_2, $x = \langle x_1, x_2 \rangle$. Suppose M' is a MTM constructing g, z is the output of M' on input x_1. M simulates

$M_y(x)$ in space z, where y is a prefix of x_2 of length z. If $z \neq 0$ and the simulation is successful, then M outputs contrary to the output of $M_y(x)$, else it outputs 0.

If $Pr\{M' \text{ outputs } z > 0\} > \frac{1}{2}$, then $Pr\{M \text{ outputs } z > 0\} > \frac{3}{4}$ and $z \leq g(|x|)$ and M is strictly $g(|x|)$ space bounded. Else $Pr\{M' \text{ outputs } z > 0\} < \frac{1}{4}$, $\varphi_M(x) = 0$ and $s_M(x) = 0$. Thus, $\varphi_M \in PrSPACE(g)$.

Suppose $\varphi_M \in PrSPACE(f)$, M_y computes φ_M with probability $\frac{3}{4}$ in space f. Choose x_1 so that $M'(x_1)$ outputs $g(|x_1|) = g(|x|)$ with probability $\frac{3}{4}$, $|y| \leq g(|x_1|)$ and $M_y(x)$ can be simulated in space $g(|x|)$, if it works in space $f(|x|)$. Choose x_2 so that it describes a padded version of the machine M. Then, if M outputs $g(|x_1|)$ with probability $\frac{3}{4}$, M on input x outputs $\neq \varphi_{M_y}(x)$. Since $(\frac{3}{4})^2 > \frac{1}{2}$, $\varphi_M \neq \varphi_{M_y}$. □

As outlined above, a diagonalization over Monte Carlo space classes seems not to be possible. For subclasses of "provable" Monte Carlo machines, however, the standard diagonalization method in connection with the halting lemma (Theorem 2) can be applied. Of course, for practical purposes, the only interesting class of algorithms is this for which a Monte Carlo property, which is Π_1^1-complete, is provable in some reasonable theory.

Definition Suppose T is an enumerable theory (e.g., Peano arithmetic or Zermelo-Fraenkel set theory).
Then $MSPACE^T(f) =$
$\{\varphi_M \mid$ "$\forall x Pr\{M(x) \text{ outputs } \varphi_M(x) \text{ in space } f(|x|)\} > \frac{3}{4}$" is a theorem of $T\}$.

Theorem 5 If T is an enumerable theory, g is MC-constructible and $f = o(g)$, then $MSPACE^T(f) \subsetneq MSPACE^T(2^{2^g})$.

PROOF By diagonalization over all machines which are provably (in T) Monte Carlo and working in $f(n)$ space we get a machine M with the (provable) properties:
(1) $Pr\{M \text{ works in space } g(n)\} > \frac{3}{4}$
(2) $Pr\{M \text{ computes } \varphi_M \text{ in space } g(n)\} > \frac{3}{4} \cdot \frac{3}{4} = \frac{9}{16} > \frac{1}{2}$.

Application of Theorem 2 yields a machine M' halting with probability 1 and the same error probability ($\frac{7}{16}$) working in space $2^{2^{g(n)}}$. Application of majority vote reduces (provably!) the error probability to $\frac{1}{4}$. Thus, $\varphi_M = \varphi_{M'} \in MSPACE^T(2^{2^g}) \setminus MSPACE(f)$. □

4. log log n is Fully (Terminating) Monte Carlo Constructible

Unlike the deterministic *fully constructible* functions we are able to give a *Monte Carlo terminating* algorithm constructing the space function log log n. Log log n is *not* a deterministic fully constructible function ([AMe 76], see for details proofs of Lemma 6 and Theorem 3).

Lemma 3 $\log n \in M^T(\log \log n)$.

PROOF We design a log log n space Monte Carlo algorithm (terminating!) counting in binary a maximal number of consecutive 'heads' in n-coin tosses for the input of length n. Given a n in unary, we toss a coin n times. The probabilities $\alpha_K = Pr\{$ maximal number of 'heads' $= K\}$ have their peak at the value $K = \log n$. The expected value of α_K is *approximately* (up to a constant) log n (see, e.g., [Fe 57], pp. 190-197). Thus we are able to construct log n in log log n Monte Carlo space up to a small constant. □

5. Separation of Random and Non-Deterministic log log n Space

The problem whether random polynomial (terminating) log n space ($RSPACE^{POLY}(\log n)$) is equal to $DSPACE(\log n)$ was formulated in [AKLLR 79]. Any separation result in

$$DSPACE(\log n) \subseteq RSPACE^{POLY}(\log n) = R^T SPACE(\log n) \subseteq NSPACE(\log n)$$

clearly solves the long-standing problem: $DSPACE(\log n) = NSPACE(\log n)$? [BS 81] suggested the likelihood that the space-bounded fast probabilistic algorithms are less powerful than non-deterministic ones.

In this section we separate the *terminating random log log n space* from the *deterministic log log n space*, and the *terminating Monte Carlo log log n space* from the *non-deterministic $o(\log n)$ space*. The random separation result is the first known space separation for an $RSPACE^{POLY}$-class, proving that the class $R^T SPACE(\log \log n)$ *is* more powerful than $DSPACE(\log \log n)$. The Monte Carlo separation result says that the fast ($n^2 \log n$-time) terminating Monte Carlo algorithms are provably more powerful than those in $NSPACE(O(\log \log n))$.

In [Fr 83] Freivalds was able to prove that the one-way (on-line) Monte Carlo probabilistic Turing machines do have a one-logarithmic space advantage over corresponding one-way deterministic Turing machines. He proves also that the sets recognized by one-way Monte Carlo probabilistic Turing machines in $o(\log \log n)$ are all regular. Our results entail that the one-logarithmic space advantage holds for arbitrary multi-tape terminating Monte Carlo PTMs (over arbitrary non-deterministic TMs).

Denote $\{0^n 1^n\} = \{0^n 1^n \mid n \in I\!N\}$, and $C\{0^n 1^n\} = \{0^n 1^m \mid n \neq m,\ n, m \in I\!N\}$.

Lemma 4 $C\{0^n 1^n\} \in R^T SPACE(\log \log n)$.

PROOF By Lemma 3, log log n is fully MC^T-constructible. We apply the Gauss prime number formula (see [AMe 76]) to obtain the $O(\log \log(n+m))$-space-deterministic algorithm for $n \neq m$: if $n \neq m \Rightarrow \exists k \leq 4 \cdot \log(n+m)\,(n \not\equiv m (\mathrm{mod}\ k))$. Then, apply the coin tosses to create 'many' (more than half of all) computations for the unique deterministic accepting computation for $0^n 1^m$, $n \neq m$. The switch-off of the algorithm for the case $0^n 1^n$ is made by the Monte Carlo algorithm of Lemma 3. □

Theorem 6 (Separation of random and deterministic log log n space)
$R^TSPACE(\log\log n) \neq DSPACE(\log\log n)$

PROOF $C\{0^n1^n\} \in R^TSPACE(\log\log n)$ by Lemma 6, and $C\{0^n1^n\} \notin DSPACE(\log\log n)$ by [AMe 76],[Al 79]. □

Corollary
$R^TSPACE(\log\log n) = NSPACE(\log\log n) \Rightarrow DSPACE(\log\log n) \neq NSPACE(\log\log n)$ □

Theorem 7 $M^TSPACE(\log\log n) \subsetneq NSPACE(o(\log n))$

PROOF $\{0^n1^n\} \in M^TSPACE(\log\log n)$ by Lemma 6, since the terminating Monte Carlo space classes are closed under the complement. It follows from [Al 79], Theorem 5, that $\{0^n1^n\} \notin NSPACE(o(\log n))$. This completes the proof. □

We were not able to disprove that "$\{0^n1^n\} \in R^TSPACE(\log\log n) \cap$ *co*-$R^TSPACE(\log\log n)$", i.e., (by Lemma 4) that $\{0^n1^n\}$ is *not* computable by any Las Vegas algorithm ($\in R^TSPACE(\log\log n) \cap$ *co*-$R^TSPACE(\log\log n)$), [BGM 82], [AMa 77]. If indeed there is a Las Vegas algorithm for $\{0^n1^n\}$, then we get the full separation of $DSPACE$, Las Vegas-$SPACE$ and $NSPACE$ for log log n:

$$DSPACE(\log\log n) \subsetneq R^TSPACE(\log\log n) \cap \text{co-}R^TSPACE(\log\log n) \subsetneq NSPACE(\log\log n).$$

We conclude with an open problem on the Monte Carlo full constructibility:

Problem Is (contrary to the case of $MSPACE(o(\log\log n))$ class) $M^TSPACE(o(\log\log n)) \equiv REG$? In particular, is log log log n fully (terminating) Monte Carlo constructible?

Acknowledgement

The initial ideas of this paper evolved during the course CS 858 given by the first author at the Carnegie-Mellon University in the spring of 1984. Our thanks go to Rick Statman, Bud Mishra, Robert Wilber, and Merrick Furst for a number of very interesting conversations. Burchard von Braunmühl shared with us some very clear insights and interesting ideas. We are thankful to Sylvio Micali for many illuminating discussions during the early stages of this research, and his final encouragement to write down the paper.

References

[AKLLR 79] Aleliunas, R., Karp, R.M., Lipton, R.J., Lovász, L., and Rackoff, C.,
Random Walks, Traversal Sequences and the Complexity of Maze Problems
Proc. 20^{th} IEEE FOCS (1979), pp.218-223

[Al 79] Alt, H.,
Lower Bounds on Space Complexity for Context-Free Recognition
Acta Informatica **12** (1979), pp.33-61

[AMa 77] Adleman, L., and Manders, K.,
Reducibility, Randomness and Intractability
Proc. 9^{th} ACM STOC (1977), pp. 151-163

[AMe 76] Alt, H., and Mehlhorn, K.,
Lower Bounds for the Space Complexity of Context-Free Recognition
in: S. Michaelson and R. Milner, eds.,
Proc. ICALP '76, Edinburgh University Press 1976, pp. 338-351

[Ba 62] Bardzin, Ya.M.,
On One Class of Turing Machines (Minsky Machines)
Algebra and Logic Seminar, Novosibirsk **6**, (1962), pp. 42-51 (Russian)

[BCP 83] Borodin, A., Cook, S., and Pippenger, N.,
Parallel Computation for Well-Endowed Rings and Space-Bounded Probabilistic Machines
Information and Control **58** (1983), pp. 113-136

[BG 81] Bennet, C., and Gill, J.,
Relative to a Random Oracle A, $P^A \neq NP^A \neq co\text{-}NP^A$ with Probability 1
SIAM J. Comput. **10** (1981), pp. 96-114

[BGM 82] Babai, L., Grigoryev, D.Yu., and Mound, D.M.,
Isomorphism of Graphs with Bounded Eigenvalue Multiplicity
Proc. 14^{th} ACM STOC (1982), pp. 310-324

[BS 83] Berman, P., and Simon, J.,
Lower Bounds on Graph Threading by Probabilistic Machines
Proc. 24^{th} IEEE FOCS (1983), pp. 304-311

[Co 83] Cook, S.,
The Classification of Problems which have Fast Parallel Algorithms
in: M. Karpinski, ed., Foundations of Computation Theory,
Springer Lecture Notes in Computer Science **158** (1983), pp. 78-93

[Fe 57] Feller, W.,
An Introduction to Probability Theory and its Applications
John Wiley, New York 1957

[Fr 81a] Freivalds, R.,
Probabilistic Two-Way Machines, MFCS '81
Springer Lecture Notes in Computer Science **118** (1981), pp. 33-45

[Fr 81b] Freivalds, R.,
Projections of Languages Recognizable by Probabilistic and Alternating Finite Multitape Automata
Information Processing Letters **13** (1981), pp. 195-198

[Fr 83] Freivalds, R.,
Space and Reversal Complexity of Probabilistic One-Way Turing Machines
in: M. Karpinski, ed., Foundations of Computation Theory,
Springer Lecture Notes in Computer Science **158** (1983), pp. 159-170

[Gi 77] Gill, J.,
Computational Complexity of Probabilistic Turing Machines
SIAM J. Comput. **6** (1977), pp. 675-694

[GJ 79] Garey, M.R., and Johnson, D.S.,
Computers and Intractability: A Guide to the Theory of NP-Completeness
W.H. Freeman, San Francisco (1979)

[HU 67] Hopcroft, J.E., and Ullman, J.D.,
An Approach to a Unified Theory of Automata
The Bell System Technical J., vol. 46, no. 8, (1967), pp. 1793-1829

[HU 79] Hopcroft, J.E., and Ullman, J.D.,
Introduction to Automata Theory, Languages, and Computation
Addison-Wesley, Reading, Ma., (1979)

[Ju 84] Jung, H.,
On Probabilistic Tape Complexity and Fast Circuits for Matrix Inversion Problems
Springer Lecture Notes in Computer Science **172** (1984), Proc. ICALP '84, pp. 281-291

[LSH 65] Lewis, P.M., Stearns, R.E., and Hartmanis, J.,
Memory Bounds for Recognition of Context-Free and Context-Sensitive Languages
Proc. 6^{th} IEEE Symp. on Switching Circuit Theory and Logical Design (1965), pp. 191-202

[Mi 61] Minsky, M.L.,
Recursive Unsolvability of Post's Problem of 'Tag' and Other Topics in the Theory of Turing Machines
Annals of Math. **74** (1961), pp. 437-455

[MS 82] Monien, B., and Sudborough, I.H.,
On Eliminating Non-Determinism from Turing Machines which Use Less than Logarithmic Work Tape Space
Theoretical Computer Science **21** (1982), pp. 237-253

[Sim 81] Simon, J.,
Space-Bounded Probabilistic Turing Machine Complexity Classes Are Closed Under Complement
Proc. 13^{th} ACM STOC (1981), pp. 158-167

[Sip 80] Sipser, M.,
Halting Space-Bounded Computations
Theoretical Computer Science **10** (1980), pp. 335-338

[SS 77] Solovay, R., and Strassen, V.,
A Fast Monte Carlo Test for Primality
SIAM J. Comput. **6** (1977), pp. 84-85

[We 83] Welsh, D.J.A.,
Randomised Algorithms
Discrete Applied Mathematics **5** (1983), pp. 133-145