The Complexity World below Logarithmic Space

Maciej Liśkiewicz[*]
University of Wrocław

Rüdiger Reischuk[†]
Med. Universität zu Lübeck

June 1994

Abstract

We investigate space complexity classes defined by Turing machines that use less than logarithmic space. Because of the limited counting ability of such machines most of the standard simulation techniques do not work for sublogarithmic space classes. However, machines with such little space may still be quite powerful. Therefore, it was not obvious how to obtain analogs for inclusion and separation results known for classes above logspace.

We review known facts about the sublogarithmic space world and present several new results, which show that these classes really behave differently. For example, certain closure properties do not hold. The restricted power of these machines makes it possible to prove explicit separations – even for alternating complexity classes – by combinatorial arguments and to obtain a hierarchy of non-relativized complexity classes without any unproven assumption. We will also discuss upward and downward translation issues.

Finally, these complexity classes are related to other classes within $\mathcal{P}$, in particular to context-free languages.

[*]Instytut Informatyki, Przesmyckiego 20, 51-151 Wrocław, Poland e-mail: liskiewi@ii.uni.wroc.pl
The research of this author was supported by the Alexander-von-Humboldt-Stiftung
[†]Institut für Theoretische Informatik, Seefahrtsschule, Wallstraße 40, 23560 Lübeck, Germany
e-mail: reischuk @ informatik.mu-luebeck.de

1 Why Should One Care about Sublogarithmic Space?

The logarithm seems to be the most crucial bound for space complexity. Since a machine with less space cannot implement a counter for a polynomial size range neither the position of the input head nor the number of steps of a computation can be recorded. Therefore, most of the standard techniques used for machine simulations do not work in this case, and for many standard properties it was not clear whether they also hold for sublogarithmic space bounds. For example, a novel technique was necessary to prove for small bounds S that $DSpace(S)$ is closed under complement [Si80], where $DSpace(S)$ denotes the set of all languages that can be recognized by S-space bounded deterministic Turing machines (DTM). Savitch's technique to simulate nondeterministic machines by deterministic machines requires space $(\max\{\log, S\})^2$ because of the recursion depth and the necessity to specify machine configurations. This gives a quadratic increase of the space for logarithmic or larger bounds. By some work the space can be reduced to $S \cdot \max\{\log, S\}$ [MoSu82], but still one gets an exponential blowup for very small space bounds.

Another important problem of this type is whether the closure under complement for NTM

$$NSpace(S) \;=\; co\text{-}NSpace(S)$$

shown by Immerman and Szelépcsenyi [Im88, Sz88] remains valid for space bounds $S < \log$. If this equality were not valid then obviously $DSpace(S)$ is properly contained in $NSpace(S)$.

Do relations between sublogarithmic classes have impact for larger complexity classes? In general,the principles of upward collapse and downward separation which can easily be proved by standard translation techniques for at least logarithmic space bounds do not seem to hold in the sublogarithmic space world. However, by a more complicated argument Szepietowski could at least show the implication:

Theorem 1 [Sz90]
If $DSpace(\log\log) \;=\; NSpace(\log\log)$ *then* $DSpace(\log) \;=\; NSpace(\log)$.

Hence equality between deterministic and nondeterministic space at a very low level still gives the identity for large space classes.

Although because of the missing counters sublogarithmic space complexity classes are technically quite difficult to handle, at first glance they may look very weak. For example, it has been known for quite a while that the deterministic context-free language

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

cannot be recognized with sublogarithmic space [LSH65], even by nondeterministic machines. This has been generalized in [AGM92] to the result that $NSpace(o(\log))$ does not contain *any* deterministic context-free nonregular language. These results, however, do not extend to probabilistic classes. Freivalds has shown the suprising result [Fr81] that `COUNT` can be accepted by a probabilistic TM in constant space with an arbitrarily small constant for the error probability.

Thus, why should one be interested in sublogarithmic space complexity classes? Let us use the shorthand llog for the twice iterated logarithmic function $n \mapsto \log_2 \log_2 n$. A deterministic or nondeterministic Turing machine (TM) that uses less than llog space can recognize regular languages only [SHL65, HoUl69]. Since there exist nonregular languages in $DSpace(\mathrm{llog})$ the function llog has turned out to be the smallest nontrivial space bound, at least for deterministic and nondeterministic TM. Later, this lower space bound has been extended to alternating Turing machines (ATM) [Su80, Iw93], thus

Theorem 2

$$DSpace(o(\mathrm{llog})) \;=\; NSpace(o(\mathrm{llog})) \;=\; ASpace(o(\mathrm{llog})) \;=\; \mathcal{REG} \;.$$

The machine model we will refer to is the standard one for space bounds, which is equipped with a separate 2-way read-only input tape and one (or more) worktape(s). Restricted machines that can read their input only 1-way, also called *on line machines*, will not be considered here. For this model space complexity investigations are considerably easier. For example, the recognition of nonregular languages with 1-way input access requires at least logarithmic workspace, and simple tasks like the recoginition of palindromes needs linear space [SHL65]. However, Sudborough has shown the surprising result that the closure of the class $ASpace(\mathrm{llog})$ with 1-way restriction under logspace reducibility is quite strong [Su80]. It equals

$$\mathcal{SC} \;:=\; DTimeSpace(\mathrm{POL}, \mathrm{PLOG}) \;,$$

where POL denotes the set of all polynomial and PLOG the set of all polylogarithmic complexity bounds, i.e. $O(\log^k)$ for arbitrary k. Furthermore, $DTimeSpace(T, S)$ is the class of all languages that can be recognized with simultaneous time bound T' and space bound S' for some $T' \in T$ and $S' \in S$.

Since in [CKS81] it has been shown that $ASpace(\log) = \mathcal{P}$ sublogarithmic space classes even defined by alternating machines are contained in $\mathcal{P}$. However, without the 1-way restriction the best upper bound known for alternating sublogarithmic space classes seems to be:

Theorem 3 [Su80] *For* $S \in$ `SUBLOG` *holds*

$$ASpace(S) \;\subseteq\; DSpace(\mathrm{ExL}(S)) \;.$$

Here, $\mathrm{ExL}(S)$ denotes the set of all complexity bounds that are bounded exponentially with respect to S, i.e. by a function of the form $\exp(k \cdot S)$ for some $k \in \mathbb{N}$, and S itself has to fulfill certain weak constructibility conditions. In particular, for the smallest nontrivial space bound we get

$$ASpace(\mathrm{llog}) \;\subseteq\; DSpace(\mathrm{PLOG}) \;.$$

However, this simulation of ATM within polylogarithmic space takes more than polynomial time, and it is an open problem whether $ASpace(\text{llog})$ is contained in $\mathcal{SC}$. Thus, sublogarithmic alternating space classes may actually be quite strong. Even, if we bound the number of alternations by some function A and let $AAlterSpace(A, S)$ denote the corresponding complexity class the best known upper bound drops down to:

Theorem 4 [MoSu82, To81, Su80]

$$AAlterSpace(A, S) \subseteq DSpace(S \cdot (S + A + \log)) .$$

This means for a constant number of alternations – denoted by the set CON of all bounded functions –

$$AAlterSpace(\text{CON}, \text{llog}) \subseteq DSpace(\log \cdot \text{llog}) ,$$

and also this class may not be included in $\mathcal{SC}$. Therefore, sublogarithmic space bounded machines with the ability of alternations may be quite powerful devices. In the following we will investigate such machines in more detail and present combinatorial techniques to prove lower bounds. For this purpose, let us denote by

$$\texttt{SUBLOG} := \Omega(\text{llog}) \cap o(\log)$$

the set of all nontrivial sublogarithmic space bounds.

2 How Does the Sublogarithmic Space World Look like?

Complexity classes are typically described by nicely growing functions the values of which can be computed easily. This includes powers and roots of the logarithmic function or iterated logarithms. However, logarithmic space seems to be necessary for this task.

Definition 1 A function $A : \mathbb{N} \to \mathbb{N}$ is *computable in space* S if there exists a DTM that for all inputs of the form 1^n writes down the binary representation of $A(n)$ on an extra 1-way output tape using no more than $S(n)$ work space. A is *approximable from below in space* S if there exists a function A' that is computable in space S with $A'(n) \leq A(n)$ for all $n \in \mathbb{N}$ and $A'(n) = A(n)$ for infinitely many $n \in \mathbb{N}$.

If S itself is computable in space S it is called *fully space-constructible*. An equivalent requirement would be the existence of a TM that on every input of length n uses exactly $S(n)$ memory cells.

There exists fully space-constructible functions in $o(\log)$. The best known example is

$$F(n) := \text{smallest integer that does not divide } n ,$$

which due to the distribution of primes even belongs to $O(\text{llog})$. However, any such function has to equal some constant value infinitely often [FrLa75]. This means

Fact 1 *No fully space-constructible function belongs to* `SUBLOG`.

This property even holds for monotone increasing space bounds if one generalizes to nondeterministic constructors [Ge91], where it is required that for each input X at least one computation path uses space $S(|X|)$ and no path uses more than that.

If a complexity bound S is fully space-constructible for each input one could mark in advance the number of available memory cells to prevent that this space bound is exceeded. Thus assuming that for each string X in a language L an acceptor M for L has at least one accepting computation that uses space at most $S(|X|)$ we can easily achieve that all computation paths stay within this bound. For sublogarithmic bounds, however, the following distinction becomes important.

Definition 2 An ATM M is (*strongly*) S space-bounded if on every input X it only enters configurations that use at most $S(|X|)$ space. M is *weakly* S space-bounded if, for every input X that is accepted, it has an accepting computation tree all of which configurations use at most $S(|X|)$ space. $DSpace(S)$ denotes the class of languages accepted by S space-bounded DTMs and $weakDSpace(S)$ denotes the languages accepted by weakly S space-bounded DTMs. A corresponding notation is used for NTMs and ATMs.

We will concentrate on the more natural strong requirement for space complexity, which is also the one used in the fundamental work of Hartmanis, Lewis and Stearns. When studying closure properties and alternating hierarchies the weak measure is not appropriate. To recognize a language L in the weak sense an acceptor for L may use arbitrary space for strings in $\overline{L}$, while a machine for $\overline{L}$ has to be bounded on this set.

Another important distinction is that a machine with space bound $S \geq \log$ can make at most exponentielly in S many steps without getting into a loop. Thus, on a separate storage track one could also count the number of steps to detect infinte computation paths. This implies immediately that for such S the class $DSpace(S)$ is closed under complement. This method cannot be applied in the sublogarithmic space world. Sipser, however, found another way to check whether the starting configuration can reach an accepting configuration without the danger of being trapped in a loop. He simulates "backwards".

Theorem 5 [Si80] *$DSpace(S)$ is closed under complement for arbitrary S.*

As the next result will show this property absolutely fails for weakly space-bounded complexity classes.

A special situation holds for bounded languages containing only strings of a certain block structure.

Definition 3 Let $Z : \mathbb{N} \to \mathbb{N}$ be a function. A language $L \subseteq \{0,1\}^*$ is Z*–bounded* if each $X \in L$ contains at most $Z(|X|)$ zeros. L is *bounded* if it is Z–bounded for some constant function Z.

The language COUNT defined above is an example for 1-boundedness. Sublogarithmic space bounded machines already face problems within a large block of identical symbols since they cannot record the exact length of such a block. The formal argument to show this property is known as the $n \mapsto n + n!$–technique developed by Stearns, Hartmanis, and Lewis in [SHL65]. For n large enough the machine will not notice any difference between the blocks 1^n and $1^{n+n!}$. In particular, if the TM obeys the space bound $s = S(n)$ in the smaller block it will use no more than s space in the large block, too.

Theorem 6 [LSH65, AlMe76, CIRB87, LiRe93a]

$$\begin{aligned}
\texttt{COUNT} &\notin DSpace(o(\log)) \ , \\
\overline{\texttt{COUNT}} &\in \text{-}weakDSpace(\text{llog}\,) \ , \\
\texttt{COUNT} &\notin weakNSpace(o(\log)) \ , \\
\overline{\texttt{COUNT}} &\notin ASpace(o(\log)) \ .
\end{aligned}$$

These bounds also show that with respect to the weak measure there can occur an exponential difference between the space complexity of a language and its complement. Even an alternating machine cannot simulate a weakly bounded deterministic machine with less than an exponential increase of the space bound.

The inability to count, however, has a positive effect, at least in case of bounded languages. As exploited in the proof of the theorem above within a block of identical input symbols the machine has to behave in a periodic fashion implying that important things only happen at the right and left boundary of a block. This observation can be used to prove

Theorem 7 [HaBe76, CIRB87]
Every infinte bounded language in $NSpace(\texttt{SUBLOG})$ contains an infinite regular subset.

This restricted behaviour of machines with small space complexity for bounded languages also allows one to implement the inductive counting method within the same space bound and yields:

Theorem 8 [Ge93b, AGM92, Sz]
For the class of Z–bounded languages, where Z is a constant or a small growing function, holds

$$NSpace(S) \ = \ co\text{-}NSpace(S)$$

also for sublogarithmic bounds S.

Definition 4 *For $k \geq 1$, the class $\Sigma_k Space(S)$ and $\Pi_k Space(S)$ are defined as all languages accepted by alternating S space-bounded TMs that make at most $k-1$ alternations and start in an existential (resp. universal) state. We will also consider ATMs with a non-constant bound A for the number of alternations. In this case, the notation $\Sigma_A Space(S)$ and $\Pi_A Space(S)$ is used. Finally, let $AAlterSpace(A, S)$ denote $\Sigma_A Space(S) \cup \Pi_A Space(S)$.*

Hence, $\Sigma_1 Space(S) = NSpace(S)$. By standard techniques it follows from Immerman-Szelépcsenyi's result:

Theorem 9 *For* $S \geq \log$ *and for all* $k \geq 1$

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

Note that these techniques do not work for sublogarithmic space bounds. Chang et al. have shown in [CIRB87] that in the sublogarithmic space world the situation is really different.

Theorem 10 *There is a language in* $\Pi_2 Space(\mathrm{llog}\,)$ *that does not belong to* $\Sigma_1 Space(o(\log\,))$.

Clearly, this proves that for space bounds S in `SUBLOG` the alternating S space hierarchy does not collapse to the first level and that

$$\Sigma_1 Space(S) \;\subset\; \Pi_2 Space(S)\ .$$

Does this relation also hold for higher levels? This question has received a lot of attention recently. Independently in a series of papers the answer was found [Br93, Ge93a, LiRe93a, BGR93, Ge93c, LiRe93b]. These classes indeed form a strict infinite hierarchy. A chronology of these results can be found in [Wa93].

Theorem 11 For all $k > 1$ holds

$$\begin{array}{lcl} \Sigma_k Space(\mathrm{llog}\,) \setminus \Pi_k Space(o(\log\,)) & \neq & \emptyset \qquad \text{and} \\ \Pi_k Space(\mathrm{llog}\,) \setminus \Sigma_k Space(o(\log\,)) & \neq & \emptyset. \end{array}$$

These results give a complete and best possible separation for the sublogarithmic space world, except for the first level $k = 1$. It is open whether also $\Sigma_1 Space(S) \neq \Pi_1 Space(S)$ for $S \in$ `SUBLOG` . Although different methods have been used to separate Σ_k and Π_k none of them seems to be applicable to the first level. The theorem implies that the alternating sublogarithmic space hierarchy is an infinite one, contrary to the case for logarithmic or larger space bounds.

Corollary 1 For any $S \in$ `SUBLOG` and all $k \geq 1$ holds

$$\begin{array}{lcl} \Sigma_k Space(S) & \subset & \Sigma_{k+1} Space(S)\ , \\ \Pi_k Space(S) & \subset & \Pi_{k+1} Space(S)\ . \end{array}$$

Thus, the only open question is how $\Sigma_1 Space(S)$ and $\Pi_1 Space(S)$ relate. It is somewhat annoying that the new techniques do not give any help for the basic case $k = 1$. The possibility that both classes are equal is not completely unrealistic, which would give the novel result that a hierarchy is infinite, although its first level collapses. We conjecture that for $k = 1$ the classes are different, but a proof seems to be significantly more difficult than for larger k . The proofs of the separation results above use in an essential way that counting to a certain extent can be replaced by alternating quantifiers. In the base case, however we have only a nondeterministic machine, that is a single existential quantifier.

That the sublogarithmic space world can be even more complex can be observed if one restricts to bounded languages. In this case we have seen that $\Sigma_1 \mathit{Space}(S)$ is closed under complementation and that the Σ_1- and Π_1-classes are identical. The second and third level, however, can be separated by bounded languages [LiRe93a]. Nothing seems to be known for level 4 and higher. Thus, the sublogartihmic space hierarchy for bounded languages may be quite strange. We have made some observations leading to the conjecture that for bounded languages this hierarchy might indeed consist of only a finite number of distinct levels.

Let us come back to the full class of languages without any boundedness restrictions. The separation of the alternating levels can be generalized to an unbounded number of alternations as long as the product of alternation and space bounds remains sublogarithmic.

Theorem 12 [LiRe93b] For any pair of functions $S \in \texttt{SUBLOG}$ and $A > 1$ with $A \cdot S \in o(\log)$, where A is approximable from below in space S, holds:

$$\begin{aligned} \Sigma_A \mathit{Space}(S) \setminus \Pi_A \mathit{Space}(S) &\neq \emptyset \, , \\ \Pi_A \mathit{Space}(S) \setminus \Sigma_A \mathit{Space}(S) &\neq \emptyset \, . \end{aligned}$$

Corollary 2 For any S and A as in the theorem above holds:

$$\begin{aligned} \Sigma_A \mathit{Space}(S) &\subset \Sigma_{A+1} \mathit{Space}(S) \, , \\ \Pi_A \mathit{Space}(S) &\subset \Pi_{A+1} \mathit{Space}(S) \, . \end{aligned}$$

Note that the class of functions that are approximable from below in space $S \in \texttt{SUBLOG}$ is quite large. For example, it contains the bounds $\lfloor \log \rfloor$, $\lfloor \log^{1/2} \rfloor$, $\lfloor \mathrm{llog} \rfloor$ and $\lfloor \log^* \rfloor$. Thus one obtains:

1. $\Sigma_{\mathrm{llog}} \mathit{Space}(\mathrm{llog}) \subset \Sigma_{(\mathrm{llog})+1} \mathit{Space}(\mathrm{llog})$.
2. $\Sigma_S \mathit{Space}(S) \subset \Sigma_{S+1} \mathit{Space}(S)$
   for $S = \log^{1/2-\epsilon}$ with $0 < \epsilon < 1/2$.
3. $\bigcup_{k \in \mathbb{N}} \Sigma_k \mathit{Space}(S) \subset \Sigma_{\log^*} \mathit{Space}(S)$
   for $S \in \Omega(\mathrm{llog}) \cap o(\log \, / \, \log^*)$.
4. For $k \in \mathbb{N}$ let
   $$\mathcal{ALSL}^k := \mathit{AAlterSpace}(\mathrm{llog}^k, \mathrm{llog}) \, .$$
   Then for any k holds $\mathcal{ALSL}^k \subset \mathcal{ALSL}^{k+1}$.

Note that for logarithmic bounds the corresponding question is still open, i.e. for any k it is unknown whether

$$\mathit{AAlterSpace}(\log^k, \log) \subset \mathit{AAlterSpace}(\log^{k+1}, \log) \, ?$$

For any function S the class $\Sigma_1 \mathit{Space}(S)$ is closed under union and intersection. However, it is still an open problem whether for $S \in \texttt{SUBLOG}$ the class $\Sigma_1 \mathit{Space}(S)$ is closed under complementation. More general, for arbitrary k the classes $\Sigma_k Space(S)$ are closed under union, and symmetrically the $\Pi_k Space(S)$ are closed under intersection. In [LiRe93a] we have developed a technique showing that for $S \in \texttt{SUBLOG}$ and for $k = 2, 3$, $\Sigma_k Space(S)$ and $\Pi_k Space(S)$ are not closed under complementation. Furthermore, $\Sigma_k Space(S)$ is not closed under intersection, and $\Pi_k Space(S)$ not under union. Combining these ideas with the separation results above we get the same closure properties for all levels.

Theorem 13 *For any $S \in \texttt{SUBLOG}$ and all $k > 1$ holds: $\Sigma_k Space(S)$ and $\Pi_k Space(S)$ are not closed under complementation and concatenation.*
Moreover, $\Sigma_k Space(S)$ is not closed under intersection and $\Pi_k Space(S)$ is not closed under union.

Note that non-closure under complementation for Σ_k and Π_k classes is not trivially equivalent to Theorem 11, which says that sublogarithmic $\Sigma_k Space$ and $\Pi_k Space$ are distinct, since sublogarithmic space-bounded machines cannot that easily detect an infinite computation path. And Sipser's method only works for deterministic machines. Thus, it is an interesting open problem whether

$$\Pi_k Space(S) \;=\; co\text{-}\Sigma_k Space(S)$$

for $k = 1, 2, \ldots$ (for more details see [LiRe93a]). But at least one can obtain the following partial solution generalizing Theorem 5 to ATMs in case of bounded languages. Such acceptors can always be guaranteed to halt. This implies

Theorem 14 *Let $S \in \texttt{SUBLOG}$ be a space bound and Z be a function computable in space S with $Z \leq \exp S$. Then for all $k \geq 1$ and for every Z-bounded language $L \subseteq \{0,1\}^*$ holds:*

$$L \in \Sigma_k Space(S) \iff \overline{L} \in \Pi_k Space(S) \ .$$

Observe that for $S \geq \log$ the function Z can grow linearly and then Z does not put any restriction on the structure of the strings in L. Thus, this theorem gives a smooth approximation of the fact that for at least logarithmic space bounds Σ_k and Π_k are complementary for arbitrary languages. We conjecture that the computability of Z is needed in the claim above. Furthermore, there are some indications that the theorem might not be true in general for bounds Z much larger than $\exp S$.

Let us finally investigate the relation between the sublogarithmic space world and context-free languages.

Definition 5 [St67] and [Gi72]
A language L is called strictly nonregular if one can find strings u, v, w, x and y such that $L \cap \{u\}\{v\}^*\{w\}\{x\}^*\{y\}$ is context-free, but nonregular.

We have seen that the simple deterministic context-free language `COUNT` cannot even be recognized by an ATM in sublogarithmic space. This is not a coincidence because of the following general result.

Theorem 15 [AGM92, LiRe93b]
If L is any nonregular deterministic context-free, strictly nonregular language, or non-regular context-free bounded language, then $L \notin \mathit{AAlterSpace}(\mathrm{CON}, o(\log))$.
Furthermore, without bounds on the number of alternations it is not possible that L and $\overline{L}$ both belong to $\mathit{ASpace}(o(\log))$.

There are more nontrivial languages besides `COUNT` that can be recognized in sublogarithmic space. An interesting example is

$$\mathtt{BIN} \;:=\; \{\mathrm{bin}(1)\#\,\mathrm{bin}(2)\#\ldots\#\,\mathrm{bin}(m) \mid m \in \mathbb{N}\}\ ,$$

where $\mathrm{bin}(i)$ denotes the binary representation of the natural number i. However, the theorem above implies that such languages cannot be too simple like being context-free and deterministic or bounded.

The language `BIN` has the property that it can serve as an advice to construct the function llog , which as we have seen is not fully space-constructible. A machine can simply check the correctness of the sequence by looking at the i-th bit position of all subwords sequentially for $i = 0, 1, \ldots$. For this purpose counters of length $\log m$ suffice, where m is roughly the logarithm of the input length.

3 Conclusions

This concludes our overview on sublogarithmic space classes. Proofs for some of the newer results can be found in [LiRe93b].

It seems that most phenomena about the sublogarithmic space world have been understood. Still, there are some open questions. It would be nice to exactly locate the classes $\mathit{ASpace}(S)$ and $\mathit{AAlterSpace}(\mathrm{CON}, S)$ for $S \in \mathtt{SUBLOG}$. Are they contained in $\mathcal{SC}$? Is it true that they do not belong to $\mathcal{NC}^1$ and $\mathcal{L} = \mathcal{SC}^1$?

What is the relevance of the alternating sublogarithmic space hierarchy? Let us denote it by

$$\mathcal{ASL} \;:=\; \mathit{AAlterSpace}(\mathrm{CON}, \mathrm{llog})\ .$$

Several other alternating hierarchies have been defined in the past. The alternating logarithmic time hierarchy

$$\mathcal{AC}^0 \;=\; \mathit{AAlterTime}(\mathrm{CON}, O(\log))$$

is also known to consist of an infinite number of distinct levels. It is contained in $\mathcal{NC}^1$ and hence in $\mathcal{SC}^1$. At the moment $\mathcal{ASL}$ seems to be the most powerful sequence of alternating complexity classes, for which one can prove a strict inclusion without using relativations or unproven assumptions. However, $\mathcal{AC}^0$ and $\mathcal{ASL}$ are not comparable. The languages `COUNT` and `PARITY` are somehow extremal examples showing that neither class is contained in the other.

What is the relation between the two lowest levels of $\mathcal{ASL}$? Is $\Sigma_1 Space(\text{llog})$ different from $DSpace(\text{llog})$ and from $\Pi_1 Space(\text{llog})$? The inability to record input head positions seems to be crucial if one wants to separate these classes. For a stronger notion of machine simulation that exploits this fact Kannan and Szepietowski have shown that such a kind of simulation of a nondeterministic TM is not possible within the same sublogarithmic space bound for a deterministic or co-nondeterministic machine [Ka83, Sz89].

Furthermore, characterize the alternating sublogarithmic space hierarchy for bounded languages completely and find out the exact relationship between co-$\Sigma_k Space(S)$ and $\Pi_k Space(S)$ for sublogarithmic space bounds S and arbitrary languages.

Finally, let us make a remark on small space probabilistic classes. Using completely different methods Freivalds and Karpinski have recently shown that the language PALINDROME cannot be recognized by probabilistic TMs with bounded error in sublogarithmic space [FrKa93]. Using the fact that there are arbitrarily small functions that are probabilistically space constructible in this case one can exploit padding techniques to obtain separation results for probabilistic space complexity classes with space bounds even between constant and llog .

Is it true that – as for probabilistic polynomial time – probabilistic small space machines with bounded error can be simulated nonuniformly in determinstic space? It is not obvious how to achieve exponentially small error probability in this case.

References

[Al79] H. Alt, *Lower bounds on space complexity for context-free recognition,* Acta Inform. 12, 1979, 33-61.

[AGM92] H. Alt, V. Geffert, and K. Mehlhorn, *A lower bound for the nondeterministic space complexity of context-free recognition,* Inform. Process. Lett. 42, 1992, 25-27.

[AlMe76] H. Alt, and K. Mehlhorn, *Lower bounds for the space complexity of context free recognition,* Proc. 3rd ICALP, 1976, 339-354.

[Br93] B. von Braunmühl, *Alternation for two-way machines with sublogarithmic space,* Proc. 10. STACS, Würzburg, 1993, 5-15.

[BGR93] B. von Braunmühl, R. Gengler, and R. Rettinger *The alternation hierarchy for machines with sublogarithmic space is infinite,* Research Report, Universität Bonn, January, 1993.

[CIRB87] J. Chang, O. Ibarra, B. Ravikumar, and L. Berman, *Some observations concerning alternating Turing machines using small space,* Inform. Proc. Letters 25, 1987, 1-9.

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

[FrLa75] A. Freedman, R. Ladner, *Space bounds for processing contentless inputs,* J. CSS 11, 1975, 118-128.

[Fr81] R. Freivalds, *Probabilistic 2-way machines,* Proc. 10. MFCS, 1981, 33-45.

[FrKa93] R. Freivalds, M. Karpinski, *Lower space bounds for randomized computation,* Technical Report, Universität Bonn, 1993, to appear Proc. ICALP94.

[Ge91] V. Geffert, *Nondeterministic computations in sublogarithmic space and space constructability,* SIAM J. Comput. 20, 1991, 484-498.

[Ge93a] V. Geffert, *Sublogarithmic* Σ_2 *-space is not closed under complement and other separation results,* Theoretical Informatics and Applications 27, 1993, 349-366.

[Ge93b] V. Geffert, *Tally version of the Savitch and Immerman-Szelepcsényi theorems for sublogarithmic space,* SIAM J. Comput. 22, 1993, 102-113.

[Ge93c] V. Geffert, *A hierarchy that does not collapse: alternations in low level space,* manuscript.

[Gi72] S. Ginsburg, *The mathematical theory of context-free languages,* McGraw-Hill, 1972.

[HaBe76] J. Hartmanis, L. Berman, *On tape bounds for single-letter alphabet processing,* TCS 3, 1976, 213-224.

[HoUl69] J. Hopcroft, J. Ullman, *Some results on tape-bounded Turing machines,* J. ACM 16, 1969, 168-177.

[Im88] N. Immerman, *Nondeterministic space is closed under complementation,* SIAM J. Comput. 17, 1988, 935-938.

[Iw93] K. Iwama, *ASpace(o(loglog n)) Is Regular,* SIAM J. Comput. 22, 1993, 136-146.

[Ka83] R. Kannan, *Alternation and the Power of Nondeterminism,* Proc. 15. SToC, 1983, 344-346.

[LiRe93a] M. Liśkiewicz, and R. Reischuk, *Separating the lower levels of the sublogarithmic space hierarchy,* Proc. 10. STACS, Würzburg, 1993, 16-27.

[LiRe93b] M. Liśkiewicz, and R. Reischuk, *The sublogarithmic space world,* Technical Report 048-93 ICSI Berkeley.

[Li85] B. Litow, *On efficient deterministic simulation of Turing machine computations below logspace,* Math. Systems Theory 18, 1985, 11- 18.

[LSH65] P. Lewis, R. Stearns, J Hartmanis, *Memory bounds for the recognition of context free and context sensitive languages,* Proc. IEEE Conf. Switching Circuit Theory and Logical Design, 1965,191-202.

[MoSu82] B. Monien, H. Sudborough, *On elmiminating nondeterminism from Turing machines which use less than logarithm worktape space,* TCS 21, 1982, 237-253.

[Si80] M. Sipser, *Halting space-bounded computations,* Theoret. Comput. Sci. 10, 1980, 335-338.

[SHL65] R. Stearns, J. Hartmanis, R. Lewis, *Hierarchies of memory limited computations,* Proc. IEEE Conf. Switching Circuit Theory and Logical Design, 1965,179-190.

[St67] R. E. Stearns, *A regularity test for pushdown-machines,* Information and Control 11, 1967, 323-340.

[Su80] I. Sudborough, *Efficient algorithms for path system problems and applications to alternating and time-space complexity classes,* Proc. 21. FoCS, 1980, 62-73.

[Sz88] R. Szelépcsenyi, *The method of forced enumeration for nondeterministic automata,* Acta Informatica 26, 1988, 279-284.

[Sz89] A. Szepietowski, *Some remarks on the alternating hierarchy and closure under complement for sublogarithmic space,* IPL 33, 1989, 73-78.

[Sz90] A. Szepietowski, *If deterministic and nondeterminstic space complexities are equal for loglog n then they are also equal for log n,* TCS 74, 1990, 73-78.

[Sz] A. Szepietowski, *Turing machines with sublogarithmic space,* unpublished manuscript.

[To81] M. Tompa, *An extension of Savitch's Theorem to small space bounds,* IPL 12, 1981, 106-108.

[Wa93] K. Wagner, Editorial note: *The alternation hierarchy for sublogarithmic space: an exciting race to STACS'93,* Proc. 10. STACS, Würzburg, 1993, 2-4.