The Log Space Oracle Hierarchy Collapses DRAFT

Samuel R. Buss
University of California, Berkeley

Stephen A. Cook[*]
University of Toronto

Patrick W. Dymond[†]
University of California, San Diego

Louise Hay
University of Illinois

July 20, 1987

The polynomial time hierarchy of Meyer and Stockmeyer has several equivalent characterizations — in particular it can be defined either in terms of polynomial time oracle Turing machines [Sto76], or in terms of polynomial time alternating Turing machines where the number of alternations is finitely bounded [CKS81]. One of the most important open questions in the area is whether or not this hierarchy collapses. (It collapses to P iff $P = NP$.) Similar hierarchies based on space bounded computations have been investigated by Ruzzo, Simon and Tompa [RST84]. They introduced a well-behaved model of space-bounded oracle Turing machine, and showed that the entire log space alternation hierarchy was contained in the second level of the log space oracle hierarchy. Subsequently Lange, Jenner and Kirsig [LJK87] proved that the alternation hierarchy collapsed to its second level; this suggested the possibility that the log space oracle hierarchy might not coincide with the log space alternation hierarchy. In this paper we show that the log space oracle hierarchy also collapses, that it thus does coincide with the log space alternation hierarchy, and that the resulting complexity class L^{NL} has other interesting characterizations in terms of circuits with oracle gates.

A crucial aspect of these results is the exact definition of log space and nondeterministic log space oracle machines. We follow [RST84] which contains a discussion of the issues involved. In their model the oracle tape is

[*]Research supported by Natural Sciences and Engineering Research Council
[†]Research sponsored by the National Science Foundation under grant DCR-8604031

not subject to the space bound, but is write-only and is erased after each query, and in addition the machine is in deterministic mode while writing each query (and hence any query is of polynomial size.) We will also need the notion of boolean circuits with oracles; we follow the approach taken in [Coo85] and [Wil85]. A relativized family of circuits is a sequence $\{\alpha_n\}$ where each α_n is a boolean circuit of n inputs which has in addition to the usual boolean gates of fanin two or less, *oracle gates* of arbitrary fanin. (The depth of an oracle gate of fanin m is taken to be $\log m$.) We only consider uniform families of depth $O(\log n)$ in this paper; the unrelativized version of this complexity class is called NC^1 (see [Coo85].)

We use the following notation:

$L = DSPACE(\log n)$

$NL = NSPACE(\log n)$

$A\Sigma_i^{\log} = \{A|A$ is accepted by a log space alternating Turing machine (ATM) starting in an existential state and using at most $i-1$ subsequent alternations between existential and universal states. $\}$

$L^S = \{A|A$ is accepted by a log space deterministic oracle Turing machine using a language in the class S as its oracle set.$\}$

$NL^S = \{A|A$ is accepted by a log space nondeterministic oracle Turing machine using a language in the class S as its oracle set.$\}$

$O\Sigma_1^{\log} = A\Sigma_1^{\log} = NL$
$O\Sigma_{i+1}^{\log} = NL^{O\Sigma_i^{\log}}$

$GAP = \{$ strings encoding a directed graph with a directed path from vertex 1 to vertex 2 .$\}$ (This set is complete for NLOG under log space or log depth reducibility [Sav70].)

$Odepth_1 NC_1^{NL} = \{A|A$ is accepted by a uniform log depth family of boolean circuits with oracle gates where the oracle set is in NL, with the proviso that oracles are not nested (i.e. no path from root to leaf can hit more than one oracle gate.) $\}$

$Odepth_i NC_1^{NL} = \{A|A$ is accepted by a uniform log depth family of boolean circuits with oracle gates where the oracle set is in NL, with the proviso that the output is an oracle gate, and all other oracles are nested no deeper than $i-1$ (i.e. no path from the output to a leaf can hit more than $i-1$ other oracle gates.) $\}$ (Here $i > 1$.)

Theorem: $L^{NL} = \bigcup_i A\Sigma_i^{\log} = \bigcup_i O\Sigma_i^{\log} = \bigcup_i Odepth_i NC_1^{NL} = Odepth_1 NC_1^{NL}$

The following set of inclusions furnishes the basis of the proof:

$$A\Sigma_2^{\log} \subseteq Odepth_1 NC_1^{NL} \subseteq L^{NL} \subseteq NL^{NL} \subseteq Odepth_2 NC_1^{NL} \subseteq A\Sigma_2^{\log}$$

Taking these inclusions in order, let M be a logspace ATM which starts its computation with existential states and then switches to universal states. This machine accepts its input iff there is some universal configuration C which cannot lead to any rejecting state and which is accessible from the start configuration through a sequence of existential configurations ([RST84]). These two conditions can be checked for a given C by two GAP oracle gates. Since there are only polynomially many possible such configurations C, acceptance can be checked by a log depth oracle circuit with no nested oracle gates. This shows the first inclusion.

To prove the second inclusion, a log space machine with GAP oracle gates evaluates a log depth circuit with unnested oracle gates in the usual depth first manner. When the output of a circuit oracle gate is needed, the machine evaluates each input to that gate in turn, and writes it on its oracle tape. For this to work, it is important that the oracle gates not be nested.

The third inclusion is obvious. To prove the fourth inclusion, let M be an NL machine with a GAP oracle. By our convention for oracles mentioned above, queries are made in the following manner: M (with its input) has a set of configurations $\{ C \}$ in which the oracle tape is blank, and from which M proceeds deterministically to write a string on the oracle tape and make its query, after which the oracle tape is erased. The problem: given configurations C and C' and the input to M, does C' result from C after one query as just described, is in NL, and hence can be answered by one oracle call to GAP in the simulating circuit. Thus a circuit can set up a complete transition table for the configurations of a modification M' of M such that M' makes no oracle calls (M' jumps from C to C'). To do this, the circuit need only make unnested oracle calls. The resulting transition table is fed into one final GAP oracle gate in the circuit, which simulates M' in the usual way.

The final inclusion is proved using a "counting technique" as in the proof of [LJK87] that the alternating log space hierarchy collapses. The idea that an efficient nondeterministic algorithm for a set S coupled with census information about the number of elements in S of a given length yields an efficient nondeterministic algorithm for the complement of S, is used in [HM80]. In our application, the census information can be computed within the alternation bound.

Let α be a log depth circuit of type $Odepth_2 NC_1^{NL}$. Thus α's output is a GAP oracle gate, and α has other GAP oracle gates, (call them top gates) no two of which are nested. We can assume without loss of generality that α is a tree. A Σ_2 log space ATM machine M can evaluate α as follows. M first guesses at the number i of top gates which output 1. Next M continues in existential mode and tries to solve the GAP instance specified by the inputs to the final oracle gate in α. To do this, M will need to know various inputs to this gate, from time to time. To calculate each input, M does the usual depth first evaluation of the subtree of α determining that input. Each time M reaches a top oracle gate, it guesses the output to that gate, and keeps track of the number of yes outputs (it will reach each top gate at most once, since α is a tree). If it guesses yes, it must verify the guess (by determining the inputs to the gate and solving the resulting reachability problem.) If its guess is no, it does not perform a verification immediately.

Instead, after determining a particular input to the final oracle gate in this manner, M must verify (by guessing outputs and verifying yes answers) that enough of the remaining top gates (i.e. those top gates not used for that particular input) output yes so that the total number of top gates outputting yes is at least i. This last check must be carried out again each time M calculates an input to α's final oracle gate. Note that the above computation of the final gate of α will be correct, under the assumption that M's guess at the value of i is correct. After M successfully completes the above simulation, assuming that the output to the final oracle gate is 1, it must verify its choice of i. To do this, it enters universal mode, and checks that it is not the case that $i + 1$ or more of the top gates output yes. This predicate is easily seen to be in co-NL.

This completes the chain of inclusions. To conclude that the log space oracle hierarchy collapses, we argue as follows. Since we know that $L^{NL} = NL^{NL}$ it is enough, by induction, to show that $O\Sigma_{k+2}^{\log} \subseteq O\Sigma_{k+1}^{\log}$, under the assumption that $NL^{O\Sigma_k^{\log}} \subseteq L^{O\Sigma_k^{\log}}$. Thus $NL^{NL^{O\Sigma_k^{\log}}} \subseteq NL^{L^{O\Sigma_k^{\log}}}$, and it is an easy exercise to prove this last class is included in $O\Sigma_{k+1}^{\log}$.

To conclude that the oracle gate nesting hierarchy collapses, we argue that the subcircuit rooted at an oracle gate of nesting depth 2 in the circuit can be transformed into an equivalent subcircuit with no nesting using the transformations of the above chain. Since the size of the new circuit is only polynomially larger and deeper by a constant factor, the circuit resulting from repeating this process a constant number of times along any path is still of depth $O(\log n)$.

Following preparation of the previous version of this manuscript and independently of it, Neil Immerman [Imm87] proved that $NL = co-NL$. His result provides a direct proof and improvement of our main result, by collapsing the log space oracle hierachy into NL. Thus at this point the remaining interest of the results presented here are in the techniques used in relating circuits with oracles to Turing machine complexity classes. For example, it is interesting to consider allowing a non-constant bound on the nesting level of oracle gates. The class NL^* was defined by Cook [Coo85] as functions computed by log depth uniform circuit families with unbounded GAP oracle nesting again subject to the condition that the depth of an oracle with m inputs is counted as $\log m$. (By standard notational extension, we here use NL^* to refer to the class of sets recognized by such circuits.) Although Immerman does not consider circuits with oracles, we observe that using his result it is easy to show that $NL = NL^*$.

Acknowledgement: We thank Martin Tompa for a helpful discussion about the alternation hierarchy.

References

[CKS81] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28:114–133, 1981.

[Coo85] Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64:2–22, 1985.

[HM80] J. Hartmanis and S. Mahaney. *On census complexity and sparseness of NP-complete sets.* Technical Report 80/416, Cornell University, April 1980.

[Imm87] Neil Immerman. *Nondeterministic Space is Closed Under Complement.* Technical Report DCS/TR 552, Yale University, July,1987.

[LJK87] Klaus-Jorn Lange, Birgit Jenner, and Bernd Kirsig. The logarithmic alternation hierarchy collapses: $A\Sigma_2^{\mathcal{L}} = A\Pi_2^{\mathcal{L}}$. In *Proceedings* *of Fourteenth International Colloquium on Automata, Languages and Programming*, 1987. To appear.

[RST84] Walter L. Ruzzo, Janos Simon, and Martin Tompa. Space bounded hierarchies and probabilistic computations. *Journal of Computer and System Sciences*, 28:216–230, 1984.

[Sav70] W.J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4:177–192, 1970.

[Sto76] L. J. Stockmeyer. The polynomial time hierarchy. *Theoretical Computer Science*, 3:1–22, 1976.

[Wil85] Chris Wilson. *Relativized circuit size and depth.* Technical Report 85/179, University of Toronto, 1985.