# A positive relativization of polynomial time vs. polylog space

Ricard Gavaldà

Report LSI–91–15

# Note: A positive relativization of polynomial time vs. polylog space

Ricard Gavaldà *

*Department of Software (L.S.I.)*
*Universitat Politècnica de Catalunya*
*Pau Gargallo, 5*
*08028 Barcelona, Spain*
*gavalda@lsi.upc.es*

**May 3, 1991**

**Abstract.** Can every set in P be solved in polylogarithmic space? We show that this question is equivalent to asking whether the classes PSPACE and EXPTIME are always equal under relativization. We use an oracle access mechanism that is fair, in the sense that it maintains all known relationships between unrelativized PSPACE and EXPTIME.

## 1. Motivation and statement of results

Whether all problems solvable in polynomial time can also be solved using short (polylogarithmic) space has been an open question in complexity theory for several years. We show in this note that this question is equivalent to asking whether two exponentially more powerful classes are always equal relative to an oracle. This constitutes what is called a *positive relativization* of the first question (see, for instance, Chapter 8 of [2]).

The two relativized classes that we compare are PSPACE $= \bigcup_{k>0} \mathrm{DSPACE}(n^k)$ and EXPTIME $= \bigcup_{k>0} \mathrm{DTIME}(2^{n^k})$. However, whenever considering relativized space-bounded classes such as PSPACE, one should define precisely how the oracle is to be accessed. A first possibility is that the space bound does not apply to the oracle tape: Consider an oracle machine whose worktape space is bounded by function $s(n)$. Without further restrictions, this machine can use $2^{O(s(n))}$ time, and therefore can make queries much larger than it can store.

It is known since the work of Ladner and Lynch [7] that the classes defined using this model may not have properties that are true for the corresponding unrelativized ones: Savitch and Immerman-Szelepcsényi theorems, as well as the inclusion $\mathrm{NSPACE}(s(n)) \subseteq \mathrm{DTIME}(2^{s(n)})$, may fail relative to some oracles in this model (see also [6]). For the particular case of the classes PSPACE and EXPTIME, Gavaldà, Torenvliet, Watanabe, and Balcázar present in [4] two oracles $A$ and $B$ such that, under this model, $\mathrm{PSPACE}(A) \neq \mathrm{NPSPACE}(A) \neq \mathrm{coNPSPACE}(A)$, and $\mathrm{NPSPACE}(B) \not\subset \mathrm{EXPTIME}(B)$.

* This research was partially supported by the ESPRIT II Basic Research Actions Program of the EC under contract No. 3075 (project ALCOM).

To avoid this strange behaviour, some other models of nondeterministic space-bounded oracle machines have been studied in the literature. Let us recall the restriction proposed by Ruzzo, Simon, and Tompa in [10], that allows long queries to the oracle provided that they are generated in a deterministic way. Another possibility, discussed by Hartmanis [6], is to consider that the space bound applies to the oracle tape too. This option has some inconvenients for sublinear space-bounds: Indeed, it easy to construct oracles $A$ with the strange property that $A \notin \mathrm{NSPACE}^A(s(n))$, for any $s(n) = o(n)$. Of course, this anomaly does not appear for the class PSPACE.

So, we consider from now on machines that have all their tapes, even the oracle tape, bounded by a polynomial in the length of the input. Let $\mathrm{PSPACE}_p(A)$ (respectively, $\mathrm{NPSPACE}_p(A)$) denote the class of sets accepted by deterministic machines with this restriction (resp., nondeterministic) relative to oracle $A$. We first argue that this is a fair model, making sure that all known relationships between unrelativized PSPACE, NPSPACE, and EXPTIME hold here relative to every oracle. The proof is a routine adaptation of the one for the unrelativized case (see for instance Chapter 2 of [1]).

**Proposition 1.** For every oracle $A$ $\mathrm{PSPACE}_p(A) = \mathrm{NPSPACE}_p(A) \subseteq \mathrm{EXPTIME}(A)$.

Now let us consider oracle EXPTIME machines. Without restrictions, these machines can make queries of exponential length that are inaccessible to $\mathrm{PSPACE}_p$ machines. Using this property, Orponen [9] presents an oracle that makes EXPTIME strictly greater than $\mathrm{APSPACE}_p$ (i.e., alternating PSPACE with polynomial length queries), although these two classes are equal without relativization. Consequently, if we wish to compare PSPACE to EXPTIME in a fair way, we must also impose a polynomial bound on the length of the queries made by EXPTIME machines, obtaining the class $\mathrm{EXPTIME}_p(A)$ relative to an oracle $A$. It is clear that for any $A$, $\mathrm{PSPACE}_p(A)$ is included in $\mathrm{EXPTIME}_p(A)$, and that the canonical complete set for EXPTIME will make these two classes equal.

Can we give an oracle separating them? Note that $\mathrm{PSPACE}_p$ and $\mathrm{EXPTIME}_p$ do not differ in the number or length of strings that they can query to their oracle: both classes can produce all words that are at most polynomially longer than their inputs. It is true that an $\mathrm{EXPTIME}_p$ machine seems to have a slight advantage: it can remember exponentially many oracle answers in its worktape, whereas $\mathrm{PSPACE}_p$ machines cannot in general do this. But this does not help it to produce new strings: every new query must have again polynomial length. Thus, one might think that an oracle to make these classes different cannot be based in "hiding" oracle information to $\mathrm{PSPACE}_p$ machines; rather, it should exploit intrinsic differences between unrelativized computation, certainly a hard problem.

We show next that, indeed, the existence of a set $A$ such that $\mathrm{PSPACE}_p(A)$ is properly included in $\mathrm{EXPTIME}_p(A)$ has strong unrelativized consequences. While intuition tells us that this separation could affect classes PSPACE and EXPTIME, the precise statement of the positive relativization is different: Any relativized separation of $\mathrm{PSPACE}_p$ and $\mathrm{EXPTIME}_p$ translates downward to show that not all P sets can be decided in poly-

logarithmic space. Let DPLOG stand for the class $\bigcup_{k>0} \mathrm{DSPACE}(\log^k n)$. Then we have:

**Theorem 2.** $\mathrm{P} \subseteq \mathrm{DPLOG}$ iff $\mathrm{PSPACE}_p(A) = \mathrm{EXPTIME}_p(A)$ for every oracle $A$.

Note that it is not known whether PSPACE = EXPTIME (unrelativized) implies $\mathrm{P} \subseteq$ DPLOG. The last section presents a similar result for other pairs of complexity classes.

## 2. Proof of Theorem 2

Let us introduce first the following convention: A *finitely defined oracle* is a mapping from a finite subset of $\{0,1\}^\star$ to $\{0,1\}$. We associate to each string $o \in \{0,1\}^\star$ a finitely defined oracle in the natural way: the value of the oracle for a string $y$ is the $y$th bit of $o$. When no ambiguity is possible we sometimes identify string $o$ with the oracle it encodes. We will prove separately each direction of Theorem 2.

**Theorem 3.** If $\mathrm{P} \subseteq \mathrm{DPLOG}$ then $\mathrm{PSPACE}_p(A) = \mathrm{EXPTIME}_p(A)$ for every oracle $A$.

**Proof.** This is shown by a standard downward translation argument. Let $E$ be an $\mathrm{EXPTIME}_p$ machine that runs in $2^{n^i}$ time and makes queries of length at most $n^i$. Define

$$L_E = \{\, \langle x, o\rangle \mid |o| = 2^{|x|^i+1}, \text{ and } E^o(x) \text{ accepts (in at most } |o| \text{ steps)} \,\}$$

where $E^o$ denotes the computation of $E$ relative to the oracle encoded in $o$. Note that every possible query made by $E(x)$ can be answered from $o$. Also, it is easy to see that this set can be decided in polynomial time. Then, assuming that $\mathrm{P} \subseteq \mathrm{DPLOG}$, there is a machine $M$ that decides $L_E$ using workspace $\log^j n$, for some $j$.

Consider now the following oracle machine: On an input $x$, simulate $M$ with input $\langle x, o\rangle$, where $|o| = 2^{|x|^i+1}$ and $o$ is never fully constructed. Instead, when the $y$th bit of $o$ is needed, query $y$ to the oracle and return the answer as 0 or 1.

It is clear that $M$ accepts such an input $\langle x, o\rangle$ iff $E$ accepts $x$ relative to the oracle encoded in $o$. Hence, relative to an oracle $A$, the procedure above accepts exactly $L(E, A)$. Furthermore, on an input $x$ it uses space $\log^j |\langle x, o\rangle| = \log^j O(2^{|x|^i}) = |x|^{O(1)}$. Therefore, for any oracle $A$, $L(E, A)$ is in $\mathrm{PSPACE}_p(A)$. ∎

To prove the opposite implication, we need the following lemma. It roughly states that if an $\mathrm{EXPTIME}_p$ machine always accepts $\mathrm{PSPACE}_p$ languages, then we can fix a single $\mathrm{PSPACE}_p$ machine that is almost equivalent to it.

**Lemma 4.** Let $E$ be a $\mathrm{EXPTIME}_p$ machine such that

$$\forall A \quad \exists M_A, \text{ a } \mathrm{PSPACE}_p \text{ machine, such that } L(E, A) = L(M_A, A).$$

Then

$$\exists M, \text{ a } \mathrm{PSPACE}_p \text{ machine}, \quad \exists B \text{ (finitely defined) such that}$$

$$\forall A \text{ extension of } B,\ L(E, A) = L(M, A).$$

**Proof.** Suppose that the second statement does not hold for $E$, that is, for every $\mathrm{PSPACE}_p$ machine $M$ and every finitely defined oracle $B$, we can extend $B$ such that $M$ and $E$ do not accept the same language under the extended oracle. We will show that the first statement cannot hold. Let $X$ be the oracle constructed by the following stage-by-stage process: Set $X_0 := \emptyset$; at every stage $k > 0$, find $X_k$, a minimum (finitely defined) extension of $X_{k-1}$ such that $L(E, X_k) \neq L(M_k, X_k)$, where $M_k$ denotes the $k$th $\mathrm{PSPACE}_p$ machine. (with our hypothesis, we can always find such an extension). Define $X$ to be $\bigcup_{k>0} X_k$.

Clearly, $X$ diagonalizes against all $\mathrm{PSPACE}_p$ machines, and so, the first statement is not true for $X$. ∎

**Theorem 5.** If for every oracle $A$, $\mathrm{PSPACE}_p(A) = \mathrm{EXPTIME}_p(A)$, then $\mathrm{P} \subseteq \mathrm{DPLOG}$.

**Proof.** Consider any language $L$ in P. It is easily seen that $L$ is log-space equivalent to the padded language

$$L' = \{\, x10^k \mid x \in L \text{ and } |x10^k| = m + 2^m \text{ for some } m \,\}$$

(i.e., $L'$ contains only words of lengths $m + 2^m$). To prove the theorem, we will show that, if $\mathrm{PSPACE}_p = \mathrm{EXPTIME}_p$ relative to every oracle, then $L' \in \mathrm{DPLOG}$.

Let $P_i$ be a machine that accepts $L'$ in time at most $n^i$. Define the machine $E$ that acts as follows: On an input $u$, $|u| = n$, it queries the words $u + 1$, $u + 2$,..., $u + 2^n$ to its oracle and interprets the answers as consecutive bits of a word $v$ of length $2^n$. Then, it simulates $P_i(uv)$.

It is clear that $E$ runs in time $2^{O(n^i)}$ (with constants not depending on the oracle that it is using), and that it makes queries of polynomial length; therefore, for any oracle $A$, $L(E, A) \in \mathrm{EXPTIME}_p(A)$. With our hypothesis, this means that $L(E, A) \in \mathrm{PSPACE}_p(A)$ for any $A$.

Therefore, for each oracle $A$ there is a $\mathrm{PSPACE}_p$ machine $M_A$ such that $L(E, A) = L(M_A, A)$. Now, by Lemma 4, we can find *one* $\mathrm{PSPACE}_p$ machine $M$ that accepts $L(E, A)$ relative to any oracle $A$ that extends a finitely defined oracle $B$. Assume that $M$ uses space at most $n^j$ on inputs of length $n$.

And finally, to accept $L'$ do the following: On an input $x$ of length $m + 2^m$, imagine $x$ split into two parts $u$ and $v$ with $|u| = m$, $|v| = 2^m$. Simulate $M(u)$. To solve a query about a word $y$ return *YES* if $y \in B$ or if the $y$th bit of $0^{|u|}v$ exists and is 1; return *NO* otherwise.

To see why this procedure works, take any word $uv$ such that $\log |u|$ is greater than the length of any word in $B$. Also let $C$ stand for the union of oracle $B$ and the oracle associated to $0^{|u|}v$. This procedure accepts $uv$ iff $M(u)$ accepts with oracle $C$, and, since $C$ extends $B$, this happens iff $E(u)$ accepts with $C$. But by the definition of $E$, this is true iff $uv$ is accepted by $P_i$. Therefore, this algorithm accepts $L'$ except for a finite number of words (namely, words $uv$ with $u$ small with respect to $B$). Furthermore, this algorithm uses space $|u|^j = m^j \leq O(\log |uv|)^j$. This implies that $L' \in \mathrm{DPLOG}$. ∎

## 3. An application to other classes

The proof of the main result is based on a translation of equalities from higher to lower classes. Similar techniques had been used before to relate equalities of exponential and polynomial time classes.

Let now E (NE) stand for the class of problems decided by deterministic (nondeterministic) machines running in $2^{O(n)}$ time. It is known that E = NE iff there are no tally sets in NP − P [3], and iff there are no sparse sets in NP − P [5]. As a direct consequence of these results, it is clear that P = NP implies E = NE. On the other hand, it is not known whether E = NE implies P = NP. The strongest result known so far is that this implication is false under certain oracles [11].

We can show the following stronger relation between polynomial and exponential time:

**Theorem 6.** The following are equivalent:

(a) P = NP.

(b) For every tally set $T$, $\mathrm{P}(T) = \mathrm{NP}(T)$.

(c) For every tally set $T$, $\mathrm{E}(T) = \mathrm{NE}(T)$.

That is, the P $\stackrel{?}{=}$ NP question is equivalent to its relativization to tally sets and to the relativization of E $\stackrel{?}{=}$ NE to tally sets, although it may not be directly equivalent to unrelativized E $\stackrel{?}{=}$ NE. Long proved the equivalence of (a) and (b) ([8], Lemma 3.5). Equivalence of (a) and (c) is shown by a modification of the translation technique, as used in Theorem 5. Again, it is critical for the proof that we can fix one E machine that is equivalent to a NE machine under the hypothesis of the theorem.

## References


[1] J.L. Balcázar, J. Díaz, J. Gabarró: *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science, vol. 11 (1988), Springer-Verlag.

[2] J.L. Balcázar, J. Díaz, J. Gabarró: *Structural Complexity II.* EATCS Monographs on Theoretical Computer Science, vol. 22 (1990), Springer-Verlag.

[3] R. Book: "Tally languages and complexity classes". *Information and Control* **26** (1974), 186–193.

[4] R. Gavaldà, L. Torenvliet, O. Watanabe, J.L. Balcázar: "Generalized Kolmogorov complexity in relativized separations". *15th MFCS Symposium* (1990), 269–276. Springer-Verlag LNCS 452.

[5] J. Hartmanis: "On sparse sets in NP-P". *Information Processing Letters* **16** (1983), 55–60.

[6] J. Hartmanis: "New developments in Structural Complexity Theory". *15th ICALP* (1988), 271–286. Springer-Verlag LNCS 317.

[7] R. Ladner, N. Lynch: "Relativization of questions about log space computability". *Mathematical Systems Theory* **10** (1976), 19–32.

[8] T. Long: "On restricting the size of oracles compared with restricting access to oracles". *SIAM Journal on Computing* **14** (1985), 585–597.

[9] P. Orponen: "Complexity classes of alternating machines with oracles". *10th ICALP* (1983), 573–584. Springer-Verlag LNCS 154.

[10] W. Ruzzo, J. Simon, M. Tompa: "Space-bounded hierarchies and probabilistic computations". *Journal of Computer and System Sciences* **28** (1984), 216–230.

[11] C. Wilson: "Relativization, reducibilities and the exponential hierarchy". M.S. Thesis. Technical Report 140/80, Deptartment of Computer Science, University of Toronto, 1980.