An Example of a Theorem that has Contradictory Relativizations and a Diagonalization Proof

Richard Chang *
Computer Science Department
Cornell University, Ithaca, NY 14853

Introduction

The central questions in complexity theory (e.g. the P =?NP question) can only be solved with proof techniques that do not relativize [BGS75]. There had been some debate about whether such techniques are within reach of the "current state of mathematics" [Hop84]. Recent advances in the area of interactive protocols have produced techniques that do not relativize [Sha90, LFKN90, HCRR90]. However, these advances do not resolve the debate on whether *diagonalization* can be used to solve problems that have contradictory relativizations [BGS75, Koz80]. In the following, we give an example of a theorem that has contradictory relativizations *and* a diagonalization proof.

The Example

Theorem. There exists a computable space bound $S(n)$ and an oracle A such that, $n^2 \leq S(n) < n^3$,

$$\text{DSPACE}[S(n)] = \text{DSPACE}[S(n)\log n],$$

but

$$\text{DSPACE}^A[S(n)] \neq \text{DSPACE}^A[S(n)\log n].$$

Proof: We construct $S(n)$ so that no Turing machine uses space between $S(n)$ and $S(n)\log n$. At step n of the construction, we compute $S(n)$ by diagonalizing over a slowly growing list of Turing machines which use at most n^3 space. We make this list of machines grow so slowly that it contains only the first $r(n) - 1$ machines, where

$$r(n) = \tfrac{\log n}{\log\log n}.$$

Consider the following function:

*Supported in part by NSF research grant CCR-88-23053.

$$f(i,n) = n^2 \log^i n.$$

Note that $f(i+1,n) = f(i,n)\log n$ and that $f(r(n),n) \leq n^3$. For each i, $0 \leq i < r(n)$, $f(i,n)$ is possible value for $S(n)$. The only requirement we need to satisfy is that no machine on the list uses space between $f(i,n)$ and $f(i,n)\log n$. Since there are only $r(n)-1$ machines, by the Pigeonhole Principle, there must be some i_0, $0 \leq i_0 < r(n)$ such that no machine's space bound falls between $f(i_0,n)$ and $f(i_0+1,n) = f(i_0,n)\log n$. Let $S(n) = f(i_0,n)$. To find this particular i_0, simply compute and record the maximum amount of space that each machine uses for any input of length n. (This is computable since the machines are space bounded.) Then, find the first i_0 that satisfies the requirement.

Thus, every machine which uses less than n^3 space either uses less than $S(n)$ space cofinitely often or uses greater than $S(n)\log n$ space infinitely often. So, DSPACE[$S(n)$] = DSPACE[$S(n)\log n$]. This construction does not violate the Space Hierarchy Theorem [HU79, SHL65] because while $S(n)$ is easily computable in PSPACE, it is not fully space constructible.[1] In fact, this theorem is an application of the Gap Theorem [Bor72, Tra67].

Now, consider the oracle

$$A = \{k\#n \mid k \leq S(n)\}.$$

$S(n)$ becomes space constructible with A as an oracle. So, by the Space Hierarchy Theorem,

$$\text{DSPACE}^A[S(n)] \neq \text{DSPACE}^A[S(n)\log n]. \qquad \square$$

Two observations are worth noting. First, the construction above works just as well for nondeterministic space, deterministic time and nondeterministic time. For the nondeterministic cases, the oracle A will also provide a language which diagonalizes over $\text{NSPACE}^A[S(n)]$ and $\text{NTIME}^A[S(n)]$. Second, $S(n)$ constructed above may not be monotonic. However, the same construction restricting $S(n)$ to be between 2^{n^2} and $2^{(n+1)^2}$ will give a monotonic space bound with the same properties.

References

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, December 1975.

[Bor72] A. Borodin. Computational complexity and the existence of complexity gaps. *Journal of the ACM*, 19(1):158–174, 1972.

[Cha89] R. Chang. An example of a theorem that has contradictory relativizations and a diagonalization proof. Technical Report 88-1059, Department of Computer Science, Cornell University, November 1989.

[1] A space bound $S(n)$ is fully space constructible if there exists a Turing machine that uses exactly $S(n)$ tape cells on input 1^n.

[HCRR90] J. Hartmanis, R. Chang, D. Ranjan, and P. Rohatgi. Structural complexity theory: Recent surprises. In *Proceedings of the 2nd Scandinavian Workshop on Algorithm Theory*, Springer Verlag *Lecture Notes in Computer Science #447*, pages 1–12, 1990.

[Hop84] J. E. Hopcroft. Turing machines. *Scientific American*, pages 86–98, May 1984.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[Koz80] D. Kozen. Indexings of subrecursive classes. *Theoretical Computer Science*, 11:277–301, 1980.

[LFKN90] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. To appear in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 1990.

[Sha90] A. Shamir. IP = PSPACE. To appear in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 1990.

[SHL65] R. E. Stearns, J. Hartmanis, and P. M. Lewis II. Hierarchies of memory limited computations. In *Proceedings of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 179–190, 1965.

[Tra67] B. A. Trakhtenbrot. *Complexity of Algorithms and Computation* [Russian]. Novosibrisk, 1967.