Notre Dame Journal of Formal Logic
Volume 31, Number 2, Spring 1990

Complexity for Type-2 Relations

MIKE TOWNSEND*

Abstract If Σ is an alphabet, then a type-2 functional is a (partial) function whose arguments are elements of Σ^* and functions from Σ^* into Σ^*. Type-2 relations are the domains of such functionals. We consider the natural extension, **POLY**, of the class of polynomial time functions to include type-2 functionals, and a variant, *POLY*, in which the time bounds depend only on the string arguments. Using these, we define two possible extensions of the (relativized) polynomial hierarchy to include type-2 relations. For example, $\Sigma_0^P = \Pi_0^P$ is the class of relations whose characteristic functionals are in **POLY**. Then Σ_{n+1}^P is the class of relations definable by **POLY** length bounded existential quantification of relations in Π_n^P, and dually for Π_{n+1}^P. Thus Σ_1^P is the type-2 analogue of NP. For any function g, we may relativize the definitions of Σ_n^P (Π_n^P) to obtain $\Sigma_n^{P,g}$ ($\Pi_n^{P,g}$). Let $\underline{\Sigma}_n^P$ denote $\bigcup_g \Sigma_n^{P,g}$. Some properties of the (relativized) hierarchy are studied. A similar analysis is carried out for the hierarchy based on *POLY*. In addition, we consider some topological notions that seem 'naturally' associated with time and space bounded computations of oracle Turing machines, and we give topological characterizations of several classes of type-2 relations. In particular, we give a topological characterization of $\underline{\Sigma}_1^P$. We use these characterizations to examine analogues of several well-known open questions of computational complexity theory. For example, we show that a certain type-2 analogue of the NP = PSPACE question has a negative answer. These results suggest that topological considerations are an integral part of the study of resource bounded computations of oracle Turing machines.

1 Introduction and preliminaries One (sometimes criticized) line of research concentrates on transferring, as far as possible, the concepts and techniques of recursion theory to the theory of computational complexity. Type-2 recursion theory extends ordinary recursion theory by permitting arguments that are func-

*I would like to thank Professor Peter Hinman and the referee, both of whom made many helpful comments, suggestions, and corrections.

Received April 18, 1988; revised October 18, 1988