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

tions. Historically, type-2 recursion theory is the recursion-theoretic end of an interface with descriptive set theory. Thus the subject has a 'feel' somewhere between recursion theory and topology (see Hinman [5]). In this paper, we consider extensions of the (relativized) polynomial hierarchy to include type-2 relations. In particular, we have **NP**, a type-2 analogue of NP. 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 (for example, the set of relations **NP** in some oracle). 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. We begin by briefly discussing the notation and terminology used in this paper.

If f is a function, then $\mathrm{Dom}(f)$ and $\mathrm{Im}(f)$ denote the domain and the image of f. We say that $f(x)$ is *defined* if x is in the domain of f; otherwise, $f(x)$ is undefined. If f and g are two functions, then $f(x) = g(x)$ means that either $f(x)$ and $g(x)$ are both undefined, or are both defined and have the same value. We write $f\colon X \to Y$ to mean that f is a function with $\mathrm{Dom}(f) \subseteq X$ and $\mathrm{Im}(f) \subseteq Y$. Two functions f and g from X into Y are equal iff $f(x) = g(x)$ for all x in X. We let ${}^X Y$ denote the set of all total functions from X into Y. If $n = \{0,\ldots,n-1\}$ is a natural number, then an element of ${}^n X$ is identified with the corresponding finite sequence of elements from X. For natural numbers n, m, we let ${}^{m,n}Y = {}^m Y \times {}^n({}^Y Y)$. If y_x is an element of Y whenever x is an element of X, then we use the expression $\lambda x.y_x$ to denote the function $\{(x,y_x) : x \in X\}$. For any set X, $\mathrm{Card}(X)$ denotes the cardinality of X.

We fix our alphabet $\Sigma = \{0,1\}$. The set of strings over Σ is denoted by Σ^*. We let ϵ denote the empty string, and let B be the symbol for a blank. Unless otherwise specified, the following notational conventions will be used. Natural numbers will be denoted by k, l, m, n, ..., and strings will be denoted by u, v, w, x, y, z. Functions from Σ^* into Σ^* are denoted by f, g, ... Subsets of ${}^n\Sigma^*$ will be written as A, B, R, S, etc. Functions from ${}^{m,n}\Sigma^*$ into Σ^*, also called *functionals of rank* (m,n) or *type-2 functionals*, will be written as $\mathbf{F}$, $\mathbf{G}$, ... Classes of functionals will be denoted by Γ_0, Γ_1, ... Subsets of ${}^{m,n}\Sigma^*$, also called *relations of rank* (m,n) or *type-2 relations*, will be denoted by $\mathbf{A}$, $\mathbf{B}$, $\mathbf{R}$, $\mathbf{S}$, etc. We use underscoring to represent sequences. Thus x is a string and $\underline{x}$ is a finite sequence of strings.

If $\mathbf{R}$ is a relation of rank (m,n), then $-\mathbf{R}$ denotes the *complement* of $\mathbf{R}$ with respect to ${}^{m,n}\Sigma^*$. Subsets of Σ^* and functions in ${}^{\Sigma^*}\Sigma^*$ are also referred to as *oracles*. A functional of rank $(n,0)$ is identified with the corresponding function on n-tuples of strings, and relations of rank $(n,0)$ are identified with subsets of ${}^n\Sigma^*$.

The *lexicographic ordering* of Σ^* is $\epsilon < 0 < 1 < 00 < \ldots$. We denote the nth string in this ordering by $\mathrm{St}(n)$, and let $\lambda x.\mathrm{Num}(x)$ be the inverse function. We let $|x|$ denote the length of x. The concatenation of x with y is denoted by xy, and x^n denotes x concatenated with itself n times. If $x = x_0x_1\ldots x_{n-1}$ and $m \le n$, then $x \downarrow m = x_{n-m}\ldots x_{n-1}$; if $m > n$, then $x \downarrow m = x$.

Let $\langle\ \rangle$ be any fixed coding of finite sequences of strings such that if $\underline{x} = x_0, x_1, \ldots, x_{n-1}$, then x_k may be extracted from $\langle \underline{x} \rangle$ in $O(n|x_k|)$ steps. Such a coding is described in Townsend [13]. For any u, if $u = \langle x_0, x_1, \ldots, x_{n-1}\rangle$ and $k \le n-1$, then $(u)_k = x_k$; otherwise, $(u)_k = \epsilon$. If $\underline{f} = f_0, \ldots, f_n$ is a sequence of functions, then $\langle \underline{f} \rangle = \lambda x.\langle f_0(x), \ldots, f_{n-1}(x)\rangle$. We often identify sequences with their codes.

A function f is *length-increasing* if $|x| \le |y|$ implies that $|f(x)| \le |f(y)|$. We say that $\mathbf{F}(x_0, \ldots, x_n, f_0, \ldots, f_m)$ is *length-increasing* in its kth string argument if whenever $x_0, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n$ are fixed strings, $f_0, \ldots, f_m$ are fixed functions, and $|x| \le |y|$, then $|\mathbf{F}(x_0, \ldots, x_{k-1}, x, x_{k+1}, \ldots, x_n, f_0, \ldots, f_m)| \le |\mathbf{F}(x_0, \ldots, x_{k-1}, y, x_{k+1}, \ldots, x_n, f_0, \ldots, f_m)|$. A functional $\mathbf{F}$ is *length-increasing* if it is length-increasing in each of its string arguments. We write $|\mathbf{F}| \le |\mathbf{G}|$ to mean that for all $(\underline{x}, \underline{f})$ in the domain of $\mathbf{F}$, $|\mathbf{F}(\underline{x}, \underline{f})| \le |\mathbf{G}(\underline{x}, \underline{f})|$, and in this case we say that $\mathbf{G}$ *bounds* $\mathbf{F}$. We say that Γ_0 *majorizes* Γ_1 if for every $\mathbf{G}_1$ in Γ_1, there is a $\mathbf{G}_0$ in Γ_0 bounding $\mathbf{G}_1$.

We associate with each $\mathbf{R}$ its *characteristic functional* $\mathbf{K_R}$ defined by

$$\mathbf{K_R}(\underline{x}, \underline{f}) = \begin{cases} 0, & \text{if } \mathbf{R}(\underline{x}, \underline{f}) \\ 1, & \text{otherwise.} \end{cases}$$

We also associate with each $\mathbf{R}$ its *positive characteristic functional* $\mathbf{K}^+_\mathbf{R}(\underline{x}, \underline{f})$ defined by

$$\mathbf{K}^+_\mathbf{R}(\underline{x}, \underline{f}) = \begin{cases} 0, & \text{if } \mathbf{R}(\underline{x}, \underline{f}) \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

We will also identify subsets of Σ^* with their characteristic functions, and thus extend functionals to include subsets of Σ^* as arguments.

For the following definitions, we omit the mention of rank, trusting that the ranks involved are clear from the context. The *zero string successor function*, S_0, is defined by $S_0(x) = x0$; we define S_1 similarly. The *length bounded exponential function*, E, is defined by $E(x, y) = 0^{|x||y|}$. The *application functional*, $\mathbf{APP}$, is defined by $\mathbf{APP}(x, f) = f(x)$. We say that $\mathbf{F}$ is defined from $\mathbf{G}$ by *expansion* if for all $\underline{x}$, $\underline{y}$, $\underline{f}$, and $\underline{g}$, $\mathbf{F}(\underline{x}, \underline{y}, \underline{f}, \underline{g}) = \mathbf{G}(\underline{x}, \underline{f})$. We say that $\mathbf{F}$ is defined from $\mathbf{G}$ by *explicit transformation* if for all $\underline{x}$ and $\underline{f}$, $\mathbf{F}(\underline{x}, \underline{f}) = \mathbf{G}(\underline{y}, \underline{g})$, where each y_k is either a fixed string or one of the x_j's, and each g_k is either a fixed function or one of the f_j's. We say that $\mathbf{F}$ is defined from $\mathbf{G}_0, \ldots, \mathbf{G}_k$, $\mathbf{R}_0, \ldots, \mathbf{R}_{k-1}$ by *cases* if for each $\underline{x}$ and $\underline{f}$ there is at most one $m \le k-1$ with $\mathbf{R}_m(\underline{x}, \underline{f})$, and for all $\underline{x}$ and $\underline{f}$

$$\mathbf{F}(\underline{x}, \underline{f}) = \begin{cases} \mathbf{G}_m(\underline{x}, \underline{f}), & \text{if } \mathbf{R}_m(\underline{x}, \underline{f}) \text{ for some } m \le k-1 \\ \mathbf{G}_k(\underline{x}, \underline{f}), & \text{otherwise.} \end{cases}$$

We say that $\mathbf{F}$ is defined from $\mathbf{G}_0, \ldots, \mathbf{G}_{k-1}$, $\mathbf{R}_0, \ldots, \mathbf{R}_{k-1}$ by *positive cases* if for each $\underline{x}$ and $\underline{f}$ there is at most one $m \le k-1$ with $\mathbf{R}_m(\underline{x}, \underline{f})$, and for all $\underline{x}$ and $\underline{f}$

$$\mathbf{F}(\underline{x},\underline{f}) = \begin{cases} \mathbf{G}_m(\underline{x},\underline{f}), \text{ if } \mathbf{R}_m(\underline{x},\underline{f}) \text{ for some } m \le k-1 \\ \text{undefined, otherwise.} \end{cases}$$

We say that **F** is defined from **G**, $\mathbf{H}_0,\ldots,\mathbf{H}_{k-1}$ by *functional composition* if for all $\underline{x}$ and $\underline{f}$, $\mathbf{F}(\underline{x},\underline{f}) = \mathbf{G}(\mathbf{H}_0(\underline{x},\underline{f}),\ldots,\mathbf{H}_{k-1}(\underline{x},\underline{f}),\underline{f})$. We say that **R** is defined from **S**, $\mathbf{G}_0,\ldots,\mathbf{G}_{k-1}$ by *relational composition* if for all $\underline{x}$ and $\underline{f}$, $\mathbf{R}(\underline{x},\underline{f})$ iff $\mathbf{S}(\mathbf{G}_0(\underline{x},\underline{f}),\ldots,\mathbf{G}_{k-1}(\underline{x},\underline{f}),\underline{f})$. We say that **F** is defined from **G** and **H** by *functional substitution* if for all $\underline{x}$ and $\underline{f}$, $\mathbf{F}(\underline{x},\underline{f}) = \mathbf{G}(\underline{x},\underline{f},\lambda y.\mathbf{H}(y,\underline{x},\underline{f}))$. We say that **R** is defined from **S** and **H** by *relational substitution* if for all $\underline{x}$ and $\underline{f}$, $\mathbf{R}(\underline{x},\underline{f})$ iff $\mathbf{S}(\underline{x},\underline{f},\lambda y.\mathbf{H}(y,\underline{x},\underline{f}))$. We say that **F** is defined from **G**, $\mathbf{H}_0$, $\mathbf{H}_1$, **B** by *limited recursion on strings* if for all $\underline{x}$, $\underline{y}$, and $\underline{g}$,

$$\begin{aligned} &\mathbf{F}(\underline{x},\epsilon,\underline{g}) = \mathbf{G}(\underline{x},\underline{g}) \\ &\mathbf{F}(\underline{x},y0,\underline{g}) = \mathbf{H}_0(\underline{x},y,\mathbf{F}(\underline{x},y,\underline{g}),\underline{g}) \\ &\mathbf{F}(\underline{x},y1,\underline{g}) = \mathbf{H}_1(\underline{x},y,\mathbf{F}(\underline{x},y,\underline{g}),\underline{g}) \\ &|\mathbf{F}(\underline{x},y,\underline{g})| \le |\mathbf{B}(\underline{x},y,\underline{g})|. \end{aligned}$$

Multitape Turing machines (TM's) are described fully in Hopcroft and Ullman [6]; we use their notation and conventions. Such a machine consists of a finite control, an input tape, a finite number of worktapes, and possibly an output tape. Certain states are designated as *halting states*, and no moves are possible from halting states. An *instantaneous description* (ID) of a machine is an encoding of the configuration of the machine. A *computation* is a finite sequence $I_0,\ldots,I_n$ of ID's such that I_0 is the initial ID relative to some input, and for each $1 \le k \le n$, I_k encodes an ID obtainable from I_{k-1} in one move. Such a computation has *n steps*. *Accepting* computations are those that end with an ID containing a halting state. A string is accepted by a machine if it generates an accepting computation. The language accepted by a machine is the set of strings accepted by the machine. A machine is *deterministic* if its finite control specifies at most one possible move for each ID; otherwise, it is *nondeterministic*. On input x, the machine computes *value* z if there is some computation halting with z on the output tape and no other computation halting with some $w \neq z$ on the output tape. As described, our machines compute functions from Σ^* into Σ^*, but through the use of codings a machine can compute a function from ${}^n\Sigma^*$ into Σ^*. We will often identify a machine with the function it computes. If t is a function from Σ^* into Σ^* and M is a Turing machine, then *M runs in time (space) t* if for every input x no halting computation for x requires more than $|t(x)|$ steps (tape cells). If **T** is a class of functions, then the phrase 'in **T** time (space)' means in time (space) t for some t in **T**.

We may extend Turing machines to accept functions from Σ^* into Σ^* as parameters or arguments by considering *oracle machines* as described in Mehlhorn [8]. An oracle machine (OTM) is a TM augmented by two special tapes, a write-only oracle input tape and a read-only oracle output tape. It takes as input an element (x, f) of ${}^{1,1}\Sigma^*$. An OTM operates exactly as a TM except that from time to time it can, by entering a special state, ask the oracle about the function value of a string z on the oracle input tape. When this happens, the machine replaces the contents of the oracle output tape with $f(z)$, places the read head on the rightmost symbol of $f(z)$, and erases the oracle input tape. ID's for

OTM's include encodings of the current oracle input tape, the last oracle input tape, and that portion of the current oracle output tape that has been scanned. As before, by using codings, a deterministic oracle machine can compute a function from ${}^{m,n}\Sigma^*$ into Σ^*. The oracle input and output tapes have been distinguished so that multi-fold composition cannot be computed cheaply. We write M^f for machine M using oracle f. Note that the machine is charged one step (space) for each symbol of the oracle input written and each symbol of the oracle output actually read. This convention is natural if we consider oracles to be representing arbitrary function (subroutine) inputs. For a contrary convention with respect to space see Ladner and Lynch [7]. If $\mathbf{T}$ is a functional from ${}^{m,n}\Sigma^*$ into Σ^* and M is an oracle machine, then *M runs in time (space)* $\mathbf{T}$ if for every input $(\underline{x}, \underline{f})$, no halting computation requires more than $|\mathbf{T}(\underline{x}, \underline{f})|$ steps (tape cells). If Γ is a class of functionals, then the phrase 'in Γ time (space)' means in time (space) $\mathbf{T}$ for some $\mathbf{T}$ in Γ.

Hopcroft and Ullman [6] give a *coding* of machines, and we let DTM_x, DOTM_x, and NOTM_x represent the xth deterministic, deterministic oracle, and nondeterministic oracle machines, respectively.

A functional is *partial recursive* (*in* g) if it is computable by a Turing machine (using oracle g as a parameter). If the functional is total, we often drop the word 'partial'. A set is *recursive* (*in* g) if its characteristic functional is recursive (in g). A set is *semi-recursive* (*in* g) if its positive characteristic functional is partial recursive (in g).

POLY is the smallest class of functions that contains the successor and length bounded exponentiation functions and is closed under composition, explicit transformation, and limited recursion on strings. It was first described in Cobham [4]. We assume some enumeration POLY_x of POLY. The use of the expression 'POLY' is justified by the following two results.

Lemma 1.1 (Observation 1.5 of Mehlhorn [8])
(1) *For every g in* POLY, *there is a polynomial p such that for all x, $|g(x)| \le p(|x|)$.*
(2) *For every polynomial q, there is an f in* POLY *such that for all x, $q(|x|) \le |f(x)|$.*

Lemma 1.2 (See Cobham [4] and Fact 1.3 of Mehlhorn [8]) *The following are equivalent:*
(1) *f is in* POLY.
(2) *f is deterministically computable in* POLY *time.*
(3) *f is deterministically computable in polynomial time* (*i.e., for some polynomial p, no halting computation on x requires more than $p(|x|)$ steps*).

Note that the Turing machine definition of polynomial time is not suitable for generalization to functionals because functions do not have a 'length'. As seen below, however, POLY does provide a way to define polynomial time that is suitable for generalization to functionals.

We let P^X denote the class of sets *polynomial in X*; that is, the collection of languages accepted by deterministic OTM's running in POLY time and using oracle X. Note that $\mathrm{A} \in \mathrm{P}^X$ iff K_A is computable by some deterministic OTM running in POLY time and using oracle X. We let NP^X denote the class of sets accepted by nondeterministic OTM's running in POLY time and using oracle X. If A is in NP^X, then we say that A is *NP in X*. We let CO-NP^X denote the collection of sets whose complements are in NP^X. Polynomial Turing reducibility, $\leq^{\mathrm{P}}_{\mathrm{T}}$, is the polynomial time analogue of ordinary Turing reducibility, and is defined by $\mathrm{A} \leq^{\mathrm{P}}_{\mathrm{T}} \mathrm{B}$ iff A is in P^{B}. Polynomial *many-one* reducibility, $\leq^{\mathrm{P}}_{\mathrm{m}}$, is the polynomial time analogue of ordinary many-one reducibility, and is defined by $\mathrm{A} \leq^{\mathrm{P}}_{\mathrm{m}} \mathrm{B}$ iff A is reduced to B via an f computable in POLY time; that is, in case for all x, $x \in \mathrm{A}$ iff $f(x) \in \mathrm{B}$.

Quantifications of the form $\exists |y| \leq |w_{\underline{x},f}|$ and $\forall |y| \leq |w_{\underline{x},f}|$ are called *length bounded quantifications.* If $\lambda(\underline{x}, f).w_{\underline{x},f}$ comes from some class Γ of functionals, then these are also referred to as Γ *bounded quantifications.*

The *relativized polynomial hierarchy* is the complexity-theoretic analogue of the relativized arithmetical hierarchy (see Rogers [10]) and is defined for any set A as follows. We begin with $\Sigma^{P,A}_0 = \Pi^{P,A}_0 = \Delta^{P,A}_0 = \mathrm{P}^{\mathrm{A}}$. Then $\Sigma^{P,A}_{n+1}$ is the class of sets definable by POLY bounded existential quantification over sets in $\Pi^{P,A}_n$. Similarly, we define $\Pi^{P,A}_{n+1}$ as the class of sets definable by POLY bounded universal quantification over relations in $\Sigma^{P,A}_n$. We let $\Delta^{P,A}_{n+1}$ be the class of sets $\leq^{\mathrm{P}}_{\mathrm{T}}$ reducible to some set in $\Sigma^{P,A}_n$. In particular, $\Sigma^{P,A}_1 = \mathrm{NP}^{\mathrm{A}}$. If $\mathrm{A} = \varnothing$, then we have the ordinary polynomial hierarchy. In this case, we will drop the superscript and write Δ^{P}_2, etc.

For any A and n, $R \in \Sigma^{P,A}_n$ iff $-R \in \Pi^{P,A}_n$. Moreover, $\Sigma^{P,A}_n$, $\Pi^{P,A}_n$, and $\Delta^{P,A}_n$ are closed under $\leq^{\mathrm{P}}_{\mathrm{m}}$, finite union, and finite intersection. Also, $\Delta^{P,A}_n$ is closed under $\leq^{\mathrm{P}}_{\mathrm{T}}$ and complementation. We have that $\Sigma^{P,A}_{n+1}$ is closed under POLY bounded existential quantification, and $R \in \Sigma^{P,A}_{n+1}$ iff R is NP in some $S \in \Sigma^{P,A}_n$. Finally, $\Sigma^{P,A}_n \cup \Pi^{P,A}_n \subseteq \Delta^{P,A}_{n+1} \subseteq \Sigma^{P,A}_{n+1} \cap \Pi^{P,A}_{n+1}$, and $\Sigma^{P,A}_{n+1} = \Pi^{P,A}_{n+2}$ iff $\Sigma^{P,A}_{n+1} = \Sigma^{P,A}_{n+2}$ iff $\Sigma^{P,A}_{n+1} = \Pi^{P,A}_{n+1}$. Yao [14] describes the construction of an oracle relative to which all levels of the polynomial hierarchy are distinct. PSPACE is the collection of sets whose characteristic functions are deterministically computable in POLY space. We define NPSPACE to be the collection of sets nondeterministically acceptable in POLY space. As is well-known, PSPACE = NPSPACE by Savitch's Theorem (see [11]). We let EXP denote $\{\lambda x.0^{2^{|\mathrm{POLY}_y(x)|}} : y \in \Sigma^*\}$; we also denote by EXP the corresponding class of sets.

If the discrete topology is assigned to Σ^*, then the *Baire topology* on ${}^{\Sigma^*}\Sigma^*$ is the product topology (see Munkres [9]). A basic open set is denoted by $[\langle \underline{s} \rangle]$, where $\underline{s}$ is a finite sequence of elements of Σ^*. The set $[\langle \underline{s} \rangle]$ consists exactly of those functions which extend $\underline{s}$, where $\underline{s}$ is viewed as a finite initial function from Σ^* into Σ^*. Here 'initial' refers to the lexicographic ordering. A functional is continuous with respect to the Baire topology iff it is partial recursive in some g. Hence, a set is open iff it is semi-recursive in some g, and a set is clopen iff it is recursive in some g (see Hinman [5]).

2 The extended arithmetical hierarchy In this section, we extend the polynomial hierarchy to include subsets of ${}^{m,n}\Sigma^*$. Many of the results in this section are straightforward adaptations of well-known results in recursion and complexity theory. For this reason, many proofs have been omitted. Complete details of the omitted proofs may be found in Townsend [12]. Mehlhorn [8] defines **POLY** to be the smallest class of functionals containing the successor and length bounded exponentiation functions, the application functional, and which is closed under composition, explicit transformation, and limited recursion on strings. This is a reasonable definition in light of Lemmas 1.1 and 1.2. It can be thought of as describing what it means to say that a computer program is efficient with respect to arbitrary string and function (subroutine) inputs. We have that **F** is deterministically computable in **POLY** time iff **F** is in **POLY.** We assume that we have an enumeration $\mathbf{POLY}_x$ of **POLY.** We have defined B to be in P^A in terms of an oracle machine using A as parameter. We might also have defined B to be in P^A if $K_B = \lambda x.\mathbf{G}(x, A)$ for some **G** in **POLY.** Lemma 2.3 of [8] shows that for every **F** in **POLY** there is an f in POLY such that for all characteristic functions g and all x, $|\mathbf{F}(x,g)| \leq |f(x)|$. Hence the two definitions are equivalent.

Definition 2.1 For any subset **A** of ${}^{m,n}\Sigma^*$,
(1) **A** is *polynomial* if $\mathbf{K_A}$ is deterministically computable in **POLY** time;
(2) **A** is *nondeterministic polynomial* if for some x and y, A is accepted by NOTM_x running in time $\mathbf{POLY}_y$;
(3) **A** is *co-nondeterministic polynomial* if $-\mathbf{A}$ is nondeterministic polynomial.

We denote the set of polynomial, nondeterministic polynomial, and co-nondeterministic polynomial relations by **P**, **NP**, and **CO-NP**, respectively. Clearly, $\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{CO\text{-}NP} \subseteq \mathbf{NP}$. We define **PSPACE**, **NPSPACE**, and **EXP** in the obvious way. Because of our convention about space and our encoding of OTM ID's, the proof of Savitch's Theorem given in Hopcroft and Ullman [6] carries over, and hence **PSPACE** = **NPSPACE**. Indeed, if the nondeterministic machine runs in space **T**, then the carried-over proof produces a polynomial p and a deterministic machine that uses no more than $p(|\mathbf{T}(\underline{x}, \underline{f})|)$ cells for any input $(\underline{x}, \underline{f})$.

Proposition 2.2 (For an analogous result, see Baker, Gill, and Solovay [1]) *For any subset* **A** *of* ${}^{m,n}\Sigma^*$,
(1) $\mathbf{A} \in \mathbf{NP}$ *iff for some* **F** *in* **POLY** *and* **R** *in* **P**, $\mathbf{A}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| \leq |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{R}(u, \underline{x}, \underline{f})$;
(2) $\mathbf{A} \in \mathbf{CO\text{-}NP}$ *iff for some* **G** *in* **POLY** *and* S *in* **P**, $\mathbf{A}(\underline{x}, \underline{f}) \leftrightarrow \forall |u| \leq |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{R}(u, \underline{x}, \underline{f})$.

By definition **POLY** is closed under functional composition, from which it follows that the class of functionals deterministically computable in **POLY** time and polynomial relations is closed under relational composition. We also have the following.

Lemma 2.3 **POLY** *is closed under expansion and functional substitution.*

Corollary 2.4 *The class of functionals deterministically computable in* **POLY** *time and polynomial relations is closed under relational substitution.*

Lemma 2.5
(1) *The class of (partial) functions deterministically computable in* **POLY** *time and polynomial relations is closed under definition by cases*;
(2) *The class of (partial) functionals nondeterministically computable in* **POLY** *time and nondeterministic polynomial relations is closed under definition by positive cases*;

(3) *If the class of (partial) functionals deterministically computable in* **POLY** *time and the nondeterministic polynomial relations is closed under definition by positive cases, then* **P** = **NP** ∩ **CO-NP.**

Corollary 2.6 **P** *is closed under complementation, finite union, and (therefore) finite intersection.*

With appropriate modifications, the results of Lemma 2.3, Corollary 2.4, Lemma 2.5, and Corollary 2.6 hold for POLY, P^{A}, NP^{A}, and $\mathrm{CO\text{-}NP}^{\mathrm{A}}$ for any A.

Definition 2.7 The class of *polynomially bounded* relations is the smallest class containing **P** and closed under **POLY** bounded quantification.

We next define a classification of the polynomially bounded relations based on the type of quantification needed to define the relation.

Definition 2.8 (The polynomial hierarchy) For all n,
(1) $\Sigma_0^P = \Pi_0^P = \mathbf{P}$;
(2) Σ_{n+1}^P is the class of relations definable by **POLY** bounded existential quantification over relations in Π_n^P;
(3) Π_{n+1}^P is the class of relations definable by **POLY** bounded universal quantification over relations in Σ_n^P.

We have used the same notation for this hierarchy as for the previously defined polynomial hierarchy, because when restricted to relations on ${}^n\Sigma^*$ they are equivalent (in this case, the type of quantification used to define the relations is identical). Moreover, many of the definitions and results concerning this hierarchy carry over to the previously defined polynomial hierarchy when they do not rely on the presence of function arguments in any essential way. We will comment when this is the case. Note that $\Sigma_1^P = \mathbf{NP}$ and $\Pi_1^P = \mathbf{CO\text{-}NP}$. We do not define a Δ class for this hierarchy. If we follow recursion theory, we might define $\Delta_n^P = \Sigma_n^P \cap \Pi_n^P$. Another possibility, based on complexity theory, would be to define the notion of being 'polynomial in a functional', and define a set to be in Δ_{n+1}^P if it is polynomial in the characteristic functional of some Σ_n^P set.

Example 2.9
(1) $\{(x,f) : f(x)$ 'codes' a propositional formula with no shorter equivalent formula$\}$ is in Π_2^P.
(2) $\{(x,f) : \mathrm{NDTM}_x$ accepts $f(x)$ in $|x|$ steps$\}$ is a member of Σ_1^P.

Next we give a sequence of technical results detailing some basic properties of the classes of the polynomial hierarchy.

Lemma 2.10 *All classes of the polynomial hierarchy are included in the class of polynomially bounded relations.*

Lemma 2.11 *For all n,* $\mathbf{R} \in \Sigma_n^P \leftrightarrow -\mathbf{R} \in \Pi_n^P$.

The converse of Lemma 2.10 follows from the next two results.

Theorem 2.12 *The classes of the polynomial hierarchy have the following properties:*

(1) *For all n,* Σ_n^P *and* Π_n^P *are closed under composition and substitution with functionals in* **POLY**
(2) *For all n,* Σ_n^P *and* Π_n^P *are closed under expansion*
(3) *For all n,* $\mathbf{P} \subseteq \Sigma_n^P \cap \Pi_n^P$
(4) *For all n,* Σ_n^P *and* Π_n^P *are closed under finite union and intersection.*

We next consider closure under bounded quantification. Suppose that **S** in Σ_{n+1}^P is defined by $\mathbf{S}(u, v, \underline{x}, \underline{f}) \leftrightarrow \exists |w| \leq |\mathbf{F}(u, v, \underline{x}, \underline{f})| \mathbf{Q}(w, u, v, \underline{x}, \underline{f})$ for some **F** in **POLY** and **Q** in Π_n^P. Consider the relation **R** defined by $\mathbf{R}(v, \underline{x}, \underline{f}) \leftrightarrow \exists |s| \leq |v| \mathbf{S}(s, v, \underline{x}, \underline{f}) \leftrightarrow (\exists |s| \leq |v|) \exists |w| \leq |\mathbf{F}(s, v, \underline{x}, \underline{f})| \mathbf{Q}(w, s, v, \underline{x}, \underline{f})$. We would like to have a **G** in **POLY** such that if $|s| \leq |v|$ then $|\mathbf{F}(s, v, \underline{x}, \underline{f})| \leq |\mathbf{G}(v, \underline{x}, \underline{f})|$, because in this case we would have

$$\mathbf{R}(v,\underline{x},\underline{f}) \leftrightarrow \exists |z| \leq |0^{c(|\mathbf{G}|(v,\underline{x},\underline{f})+|v|)}|(|(z)_0| \leq |v| \text{ and } |(z)_1| \leq |\mathbf{F}((z)_0,v,\underline{x},\underline{f})| \text{ and } \mathbf{Q}((z)_1,(z)_0,v,\underline{x},\underline{f}))$$

and we could conclude that Σ_{n+1}^P is closed under **POLY** bounded existential quantification. Such a **G** exists if **F** is bounded by a **POLY** functional that is length-increasing with respect to s. However, if $\mathbf{F}(s, v, \underline{x}, \underline{f}) = f_0(s)$, then there is no reason to believe that such a bounding functional exists. We will consider below another extension of the polynomial hierarchy of Section 1 for which this problem does not arise.

We do have the following.

Lemma 2.13 *For all n*:
(1) *If* **S** *is in* Σ_{n+1}^P *and* **R** *is defined by* $\mathbf{R}(v, \underline{x}, \underline{f}) \leftrightarrow \exists |s| \leq |v| \mathbf{S}(s, v, \underline{x}, \underline{f})$, *then* **R** *is in* Σ_{n+3}^P
(2) *If* **S** *is in* Π_{n+1}^P *and* **R** *is defined by* $\mathbf{R}(v, \underline{x}, \underline{f}) \leftrightarrow \exists |s| \leq |v| \mathbf{S}(s, v, \underline{x}, \underline{f})$, *then* **R** *is in* Π_{n+3}^P.

Proof: (2) follows from (1). For (1) let **F** and **Q** be as above and define **Q′** by $\mathbf{Q}'(z, w, u, v, \underline{x}, \underline{f}) \leftrightarrow \mathbf{Q}(w, u, v, \underline{x}, \underline{f})$. Then **Q′** is in Σ_{n+1}^P by Theorem 2.12 and $\mathbf{R}(v, \underline{x}, \underline{f}) \leftrightarrow (\exists |s| \leq |v|)(\forall |z| \leq 1) \exists |w| \leq |\mathbf{F}(s, v, \underline{x}, \underline{f})| \mathbf{Q}'(z, w, s, v, \underline{x}, \underline{f})$.

We remark that with the appropriate modifications the proof of Theorem 2.12 applies to the polynomial hierarchy defined in Section 1. Moreover, in that case, if **R** and **S** are as in Lemma 2.13 (1) then **R** is in Σ_{n+1}^P, since Lemma 1.1 implies that bounding functions can be chosen to be length-increasing.

Corollary 2.14 *For all n*:
(1) $\Sigma_n^P \cup \Pi_n^P \subseteq \Sigma_{n+1}^P \cap \Pi_{n+1}^P$
(2) $\cup \Sigma_n^P = \cup \Pi_n^P = \cup (\Sigma_n^P \cap \Pi_n^P) =$ *the class of polynomially bounded relations.*

We do not know that each inclusion is proper, but conjecture that this is the case.

Proposition 2.15 (For an analogous result, see Baker and Selman [2]) *For all n, the following are equivalent:*
(1) $\Sigma_n^P = \Pi_{n+1}^P$
(2) $\Pi_n^P = \Sigma_{n+1}^P$
(3) $\Sigma_n^P = \Sigma_{n+1}^P$
(4) $\Pi_n^P = \Pi_{n+1}^P$.

Moreover, each of (1)-(4) *imply*
(5) $\Sigma_n^P = \Pi_n^P$.

We turn next to the question of universal sets for **NP** and **CO-NP.**

Definition 2.16 For all w, $\underline{x}$, f, and m, $\mathbf{U}_1^P(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^m) \leftrightarrow \mathrm{NOTM}_w$ accepts $(\underline{x}, \underline{f})$ in at most m steps.

Lemma 2.17
(1) $\mathbf{U}_1^P \in \Sigma_1^P$
(2) *For every* $\mathbf{R}$ *in* Σ_1^P, *there exists a* w *and* y *such that* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \mathbf{U}_1^P(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{POLY}_y(\underline{x}, \underline{f})|})$
(3) $-\mathbf{U}_1^P \in \Pi_1^P$
(4) *For every* $\mathbf{R}$ *in* Π_1^P, *there exists a* w *and* y *such that* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow -\mathbf{U}_1^P(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{POLY}_y(\underline{x}, \underline{f})|})$.

Proof: The padding 0^m ensures that (1) holds; (2) is clear from the definitions; (3) and (4) follow from (1), respectively (2).

In this sense, $\mathbf{U}_1^P$ and $-\mathbf{U}_1^P$ are *universal* for Σ_1^P, respectively Π_1^P. We postpone the discussion of universal sets for other classes until later in the section.

Next we describe another possible extension of the polynomial hierarchy to include type-2 relations. Define *POLY* to be the class of functionals deterministically computable in POLY time; that is, the time bound for some computation on (x, f) is of the form $|t(x)|$ for some t in POLY (rather than $|\mathbf{T}(x, f)|$ for some $\mathbf{T}$ in **POLY**). This definition appears in Townsend [12], and a similar definition appears in Buss [3]. Note that $\mathbf{F}$ is in *POLY* iff $\mathbf{F}$ is computable in *POLY* time. Similarly, we define *PSPACE*, *NPSPACE*, and *EXP*. As before, *PSPACE* = *NPSPACE*. We define P to be the class of sets whose characteristic functionals are in *POLY*. Similarly, *NP* is the class of sets nondeterministically acceptable by a machine running in *POLY* time. The appropriate version of Lemma 2.2 holds with essentially the same proof. Straightforward machine simulation shows that *POLY* is closed under composition, explicit transformation, and functional substitution. Define $\mathrm{SIG}_0^P = \mathrm{PI}_0^P = P$. Then SIG_{n+1}^P is the class of relations definable by *POLY* bounded existential quantification of relations in PI_n^P, and PI_{n+1}^P is the class of relations definable by *POLY* bounded universal quantification of relations in SIG_{n+1}^P. Again, when restricted to relations on ${}^n\Sigma^*$, this hierarchy is equivalent to the polynomial hierarchy of Section 1. Note that $\mathrm{SIG}_n^P \subseteq \Sigma_n^P$ for all n. All of the previous results of this section hold for the SIG and PI classes with essentially the same proofs.

We now turn to the question of universal sets for the SIG and PI classes and remark that a similar discussion, with corresponding results, can be carried out for the polynomial hierarchy of Section 1. Before describing the universal sets, we develop some normal form representations for sets in the SIG and PI classes.

Lemma 2.18
(1) *For every* f *in* POLY, *there is an* h *in* POLY *such that for all* $\underline{x}$, $|f(\underline{x})| \leq |h(\underline{x})|$ *and* h *is length-increasing*
(2) *For every* $\mathbf{F}$ *in POLY, there is an* $\mathbf{H}$ *in POLY such that for all* $(\underline{x}, \underline{f})$, $|\mathbf{F}(\underline{x}, \underline{f})| \leq |\mathbf{H}(\underline{x}, \underline{f})|$ *and* $\mathbf{H}$ *is length-increasing.*

Proof: The proof of (1) is an easy induction on the definition of POLY. If f is a successor or the length bounded exponential function, then the statement is obvious. We next consider limited recursion on strings; the arguments for composition and explicit transformation are similar. Suppose that f is defined from g, h_0, h_1, and b by limited recursion on strings. By the induction hypothesis, there is an appropriate h for b. Since b is a bound for f, h works for f. For (2), if $\mathbf{F}$ is computed in time g, and h is a length-increasing bound for g, then for $\mathbf{H}$ take $\lambda(\underline{x}, \underline{f}).0^{|h(\underline{x})|}$.

We now derive some normal form representations for the SIG and PI classes.

Corollary 2.19
(1) *For any n, $\mathbf{R} \in \mathrm{SIG}_n^P$ iff there exist length-increasing functionals $\mathbf{H}_0, \ldots, \mathbf{H}_{n-1}$ in POLY and $\mathbf{T}$ in P such that $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u_0| \le |\mathbf{H}_0(\underline{x}, \underline{f})| \ldots Q_{n-1} |u_{n-1}| \le |\mathbf{H}_{n-1}(\underline{x}, \underline{f})| \mathbf{T}(u_0, \ldots, u_{n-1}, \underline{x}, \underline{f})$, where Q_{n-1} is $\exists$ if n is odd and $\forall$ if n is even*
(2) *For any n, $\mathbf{R} \in \mathrm{PI}_n^P$ iff there exist length-increasing functionals $\mathbf{H}_0, \ldots, \mathbf{H}_{n-1}$ in POLY and $\mathbf{T}$ in P such that $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \forall |u_0| \le |\mathbf{H}_0(\underline{x}, \underline{f})| \ldots Q_{n-1} |u_{n-1}| \le |\mathbf{H}_{n-1}(\underline{x}, \underline{f})| \mathbf{T}(u_0, \ldots, u_{n-1}, \underline{x}, \underline{f})$, where Q_{n-1} is $\forall$ if n is odd and $\exists$ if n is even*
(3) *For any $n \ge 1$, $\mathbf{R} \in \mathrm{SIG}_n^P$ iff there is a length-increasing functional $\mathbf{F}$ in POLY and $\mathbf{Q} \in \mathrm{PI}_{n-1}^P$ with $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| \le |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{Q}(u, \underline{x}, \underline{f})$*
(4) *For any $n \ge 1$, $\mathbf{R} \in \mathrm{PI}_n^P$ iff there is a length-increasing functional $\mathbf{F}$ in POLY and $\mathbf{Q} \in \mathrm{SIG}_{n-1}^P$ with $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \forall |u| \le |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{Q}(u, \underline{x}, \underline{f})$.*

Proof: It is clear that any $\mathbf{R}$ so defined in (1) and (2) is in SIG_n^P, respectively PI_n^P. We prove the converse by induction on n. There is nothing to prove if $n = 0$. Assume the result for n, and suppose that $\mathbf{R}$ in SIG_{n+1}^P is defined by $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| \le |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{Q}(u, \underline{x}, \underline{f})$ for some $\mathbf{F}$ in *POLY* and $\mathbf{Q}$ in PI_n^P. By the induction hypothesis, there exist length-increasing functionals $\mathbf{H}_1, \ldots, \mathbf{H}_{n-1}$ and $\mathbf{T}$ in P with $\mathbf{Q}(u, \underline{x}, \underline{f}) \leftrightarrow \forall |u_1| \le |\mathbf{H}_1(u, \underline{x}, \underline{f})| \ldots Q_{n-1} |u_{n-1}| \le |\mathbf{H}_{n-1}(u, \underline{x}, \underline{f})| \mathbf{T}(u_1, \ldots, u_{n-1}, u, \underline{x}, \underline{f})$. If $\mathbf{H}_0$ is a length-increasing bound for $\mathbf{F}$, then

$$\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow (\exists |u_0| \le |\mathbf{H}_0(\underline{x}, \underline{f})|)(\forall |u_1| \le |\mathbf{H}_1(\mathbf{H}_0(\underline{x}, \underline{f}), \underline{x}, \underline{f})|) \ldots$$
$$(|u_0| \le |\mathbf{F}(\underline{x}, \underline{f})| \text{ and } |u_1| \le |\mathbf{H}_1(\underline{x}, \underline{f})| \ldots \text{ and } \ldots \mathbf{T}(u_1, \ldots, u_n, u_0, \underline{x}, \underline{f}))$$

and the result follows from the properties of P and *POLY*. (3) follows from (1), and (4) follows from (2).

We remarked before the proof of Lemma 2.13 that if our bounds are increasing in length, then we can conclude that there is closure under length bounded quantification.

Corollary 2.20 *For all $n \ge 1$,*
(1) SIG_n^P *is closed under* POLY *bounded existential quantification*
(2) PI_n^P *is closed under* POLY *bounded universal quantification.*

Moreover, in the corresponding version of Lemma 2.15, we have $\mathrm{SIG}_n^P = \mathrm{PI}_n^P$ implies that $\mathrm{SIG}_{n+1}^P = \mathrm{PI}_{n+1}^P$ ($n \ge 1$) because SIG_n^P and PI_n^P are closed under length bounded existential quantification, respectively length bounded universal quantification. We prove one more normal form representation.

Lemma 2.21 *For all n:*
(1) **R** *is in* SIG^P_{n+1} *iff for some* **S** *in* PI^P_n *and length-increasing* **F** *in POLY,* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| = |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{S}(u, \underline{x}, \underline{f})$
(2) **R** *is in* PI^P_{n+1} *iff for some* **S** *in* SIG^P_n *and length-increasing* **F** *in POLY,* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \forall |u| = |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{S}(u, \underline{x}, \underline{f})$.

Proof: (1) Clearly any **R** so defined is in SIG^P_{n+1}. Conversely, suppose **R** in SIG^P_{n+1} is defined by $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| \leq |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{T}(u, \underline{x}, \underline{f})$ for some length-increasing **F** in *POLY* and **T** in PI^P_n. Then $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow (\exists |u| = |\mathbf{F}(\underline{x}, \underline{f})|) \exists |v| \leq |u| \mathbf{T}(v, \underline{x}, \underline{f})$. (2) follows from (1).

We are now ready to define universal sets for the SIG and PI classes.

Definition 2.22
(1) $\mathbf{UN}^P_1(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^m) \leftrightarrow \mathrm{NOTM}_w$ accepts $(\underline{x}, \underline{f})$ in at most m steps;
(2) $\mathbf{UN}^P_{n+1}(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{m_0}, \ldots, 0^{m_n}) \leftrightarrow \exists |u| = m_n - \mathbf{UN}^P_n(w, \langle u, \underline{x} \rangle, \langle \underline{f} \rangle, 0^{m_0}, \ldots, 0^{m_{n-1}})$.

Theorem 2.23 *For all $n \geq 1$:*
(1) $\mathbf{UN}^P_n$ *is in* SIG^P_n *and* $-\mathbf{UN}^P_n$ *is in* PI^P_n
(2) *If* **R** *is in* SIG^P_n*, then for some w and length-increasing functionals* $\mathbf{F}_0, \ldots, \mathbf{F}_{n-1}$ *in POLY,* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \mathbf{UN}^P_n(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{F}_0(\underline{x}, \underline{f})|}, \ldots, 0^{|\mathbf{F}_{n-1}(\underline{x}, \underline{f})|})$
(3) *If* **R** *is in* PI^P_n*, then for some w and length-increasing functionals* $\mathbf{F}_0, \ldots, \mathbf{F}_{n-1}$ *in POLY,* $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow -\mathbf{UN}^P_n(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{F}_0(\underline{x}, \underline{f})|}, \ldots, 0^{|\mathbf{F}_{n-1}(\underline{x}, \underline{f})|})$.

Proof: (1) is proved by an easy induction on n. We prove (2) and (3) by induction on n. If $n = 1$, then the statement is clear from the definitions. Suppose **R** in SIG^P_{n+1} is defined by $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \exists |u| = |\mathbf{F}(\underline{x}, \underline{f})| \mathbf{S}(u, \underline{x}, \underline{f})$ for some length-increasing **F** in *POLY* and **S** in PI^P_n. By the induction hypothesis

$$\mathbf{S}(u, \underline{x}, \underline{f}) \leftrightarrow -\mathbf{UN}^P_n(w, \langle u, \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{F}_0(\underline{x}, \underline{f})|}, \ldots, 0^{|\mathbf{F}_{n-1}(\underline{x}, \underline{f})|})$$

for some w and length-increasing $\mathbf{F}_0, \ldots, \mathbf{F}_{n-1}$ in *POLY.* Hence

$$\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \mathbf{UN}^P_n(w, \langle \underline{x} \rangle, \langle \underline{f} \rangle, 0^{|\mathbf{F}_0(\underline{x}, \underline{f})|}, \ldots, 0^{|\mathbf{F}_{n-1}(\underline{x}, \underline{f})|}, 0^{|\mathbf{F}(\underline{x}, \underline{f})|}).$$

The argument for PI^P_{n+1} is similar.

We cannot use the same argument for the Σ and Π classes because **POLY** does not satisfy the appropriate version of Lemma 2.18. However, Lemma 2.3 of Mehlhorn [8] shows that for every **F** in **POLY**, there is an f in POLY such that, for all characteristic functions g, $|\mathbf{F}(\underline{x}, g)| \leq |f(\underline{x})|$. It follows that if we restrict our attention to relations on strings and characteristic functions, then the SIG and Σ classes are the same.

There are two natural definitions for the notion of '**POLY** in g'.

Definition 2.24 For any g:
(1) $\mathbf{POLY}[g] = \{\mathbf{F}$: there exists a **G** in **POLY** with $\mathbf{F}(\underline{x}, \underline{f}) = \mathbf{G}(\underline{x}, \underline{f}, g)$ for all $\underline{x}$ and $\underline{f}\}$;
(2) $\mathbf{POLY}^g$ is the smallest class containing g, the successor and length bounded exponentiation functions, the application functional, and which is closed under composition, explicit transformation, and limited recursion on strings.

Lemma 2.25 *For any g*, $\mathbf{POLY}[g] = \mathbf{POLY}^g$.

We will write $\mathbf{POLY}^g_w$ for $\lambda(\underline{x}, \underline{f}).\mathbf{POLY}_w(\underline{x}, \underline{f}, g)$. There are two natural definitions for the relativized hierarchy.

Definition 2.26 For any g:
(1) $\Sigma^P_0[g] = \Pi^P_0[g] = \mathbf{P}[g] = \{\mathbf{A} : \mathbf{K}_\mathbf{A} \in \mathbf{POLY}[g]\}$;
(2) For any $n \geq 1$, $\Sigma^P_{n+1}[g]$ is the class of relations definable by $\mathbf{POLY}[g]$ bounded existential quantification of relations in $\Pi^P_n[g]$;
(3) For any $n \geq 1$, $\Pi^P_{n+1}[g]$ is the class of relations definable by $\mathbf{POLY}[g]$ bounded universal quantification of relations in $\Sigma^P_n[g]$;
(4) For all n, $\mathbf{R} \in \Sigma^{P,g}_n$ iff there exists an $\mathbf{S}$ in Σ^P_n such that for all $(\underline{x}, \underline{f})$, $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \mathbf{S}(\underline{x}, \underline{f}, g)$;
(5) For all n, $\mathbf{R} \in \Pi^{P,g}_n$ iff there exists an $\mathbf{S}$ in Π^P_n such that for all $(\underline{x}, \underline{f})$, $\mathbf{R}(\underline{x}, \underline{f}) \leftrightarrow \mathbf{S}(\underline{x}, \underline{f}, g)$.

Lemma 2.27 *For all n*, $\Sigma^P_n[g] = \Sigma^{P,g}_n$ *and* $\Pi^P_n[g] = \Pi^{P,g}_n$.

For any g, all of the previous results of this section hold for the relativized classes. For universal sets, we need g to be a characteristic function. If we restrict our attention to relations on strings and use characteristic functions for oracles, then these classes agree with the relativized hierarchy of Section 1. Hence by a result of Yao [14] there are g's relative to which all levels of these hierarchies are distinct.

Definition 2.28 For any n:
(1) $\underline{\Sigma}^P_n = \bigcup\{\Sigma^P_n[g] : g \in {}^{\Sigma^*}\Sigma^*\}$;
(2) $\underline{\Pi}^P_n = \bigcup\{\Pi^P_n[g] : g \in {}^{\Sigma^*}\Sigma^*\}$.

In a similar manner, we may define $\mathit{POLY}[g]$, $\mathrm{SIG}^P_n[g]$, $\mathrm{PI}^P_n[g]$, $\mathbf{PSPACE}[g]$, $\mathit{PSPACE}[g]$, $\mathbf{NPSPACE}[g]$, $\mathit{NPSPACE}[g]$, $\mathbf{EXP}[g]$, $\mathit{EXP}[g]$, $\underline{\mathrm{SIG}}^P_n$, $\underline{\mathrm{PI}}^P_n$, $\underline{\mathbf{PSPACE}}$, $\underline{\mathit{PSPACE}}$, $\underline{\mathbf{NPSPACE}}$, $\underline{\mathit{NPSPACE}}$, $\underline{\mathbf{EXP}}$, and $\underline{\mathit{EXP}}$. As before, the appropriate versions of Savitch's Theorem hold. In the next two sections, we discuss some topological properties of $\underline{\Sigma}^P_1$ and $\underline{\mathrm{SIG}}^P_1$.

3 Topology

In this section, we introduce some topological concepts that are 'naturally' associated with time and space bounded computations of oracle machines. For simplicity, we consider our machines to be taking inputs of the form (x, f). Our goal is to describe certain subsets of ${}^{1,1}\Sigma^*$ both complexity theoretically and topologically.

Recall the operation of an oracle machine. When making an oracle call, the machine is charged one step. In addition, the machine is charged one step (space) for each symbol of the input written and each symbol of the output read. This suggests that we take into account three things: the number of oracle calls made, the size of the oracle input, and the amount of information actually used from the oracle output.

In ordinary recursion theory, we use the Baire topology to study oracle computations. Because oracle machines have finite controls, two inputs (x, f) and (x, g) will produce the same halting computations on a machine M if f and g agree on a sufficiently long initial segment of Σ^*. In other words, if (x, f) produces a certain set of halting computations, then there is some basic open interval $[\langle \underline{s} \rangle]^{\text{Baire}}$ such that if g is in this interval then (x, g) produces the same set of halting computations. This analysis takes into account only the number of calls made and is not an appropriate model for complexity bounded computations.

We begin with some preliminary definitions. Recall that B is the symbol for a blank. We let $B\Sigma^*$ denote $\{Bx : x \in \Sigma^*\}$.

Definiton 3.1 Let p: $\Sigma^* \to (\Sigma^* \cup B\Sigma^*)$ be a finite function. We define the interval $[p]^* \subseteq {}^{\Sigma^*}\Sigma^*$ by $f \in [p]^*$ iff for every $w \in \mathrm{Dom}(p)$,

$$f(w) = \begin{cases} x, \text{ if } p(w) = Bx \\ zp(w) \text{ for some } z \in \Sigma^*, \text{ otherwise.} \end{cases}$$

In other words, if $p(w)$ is prefixed by a B, then $f(w)$ must equal the suffix; otherwise, $f(w)$ need only have $p(w)$ as a suffix. A finite function p will also be denoted by $\langle x_0,p(x_0)\rangle,\ldots,\langle x_n,p(x_n)\rangle$, where $\{x_0,\ldots,x_n\}$ is the domain of p. We will assume that there is a coding of finite functions into Σ^*, and will identify finite functions with their codes.

Example 3.2
(1) If $f = \lambda x.11$, then $f \in [\langle 0,1\rangle]^*$, $f \notin [\langle 0,B1\rangle]^*$, $f \in [\langle \epsilon,11\rangle]^*$, $f \in [\langle \epsilon,B11\rangle]^*$, and $f \in [\langle 1,\rangle, \langle 10,1\rangle]^*$.
(2) $[\langle \epsilon,01\rangle]^* \cap [\langle \epsilon,B01\rangle]^* = [\langle \epsilon,B01\rangle]^*$.
(3) $[\langle \epsilon,101\rangle]^* \cap [\langle \epsilon,01\rangle]^* = [\langle \epsilon,101\rangle]^*$.
(4) $[\langle \epsilon,101\rangle]^* \cap [\langle 1,1\rangle]^* = [\langle \epsilon,101\rangle, \langle 1,1\rangle]^*$.
(5) $[\langle \epsilon,101\rangle]^* \cap [\langle \epsilon,B01\rangle]^* = \emptyset$.
(6) $[\langle \epsilon,B01\rangle]^* = [\langle 01\rangle]^{\text{Baire}}$.
(7) $[\langle 01\rangle]^{\text{Baire}}$ is a proper subset of $[\langle \epsilon,01\rangle]^*$.
(8) $[\langle \epsilon,B01\rangle, \langle 1,\rangle]^*$ is a proper subset of $[\langle 01\rangle]^{\text{Baire}}$.
(9) $[\langle \epsilon,01\rangle, \langle 1,1\rangle]^*$ and $[\langle 01\rangle]^{\text{Baire}}$ are incomparable with respect to set inclusion.

The next result shows that these intervals generate a topology equivalent to the Baire topology.

Lemma 3.3 *$\{[p]^*$: p is a finite function from Σ^* into $\Sigma^* \cup B\Sigma^*\}$ is a basis for a topology that is equivalent to the Baire topology on ${}^{\Sigma^*}\Sigma^*$.*

Proof: For any $\underline{s} = s_0,\ldots,s_k$, $[\langle \underline{s}\rangle]^{\text{Baire}} = [\langle \epsilon,Bs_0\rangle,\ldots,\langle \mathrm{St}(k),Bs_k\rangle]^*$. On the other hand, if $x_0 < x_1 < \cdots < x_m$ and f is in $[\langle x_0, p(x_0)\rangle,\ldots,\langle x_m, p(x_m)\rangle]^* = [p]^*$, then $f \in [\langle f(\epsilon),\ldots,f(x_m)\rangle]^{\text{Baire}} \subseteq [p]^*$. Thus $[p]^*$ generates a topology equivalent to the Baire topology.

We have the following immediate corollary.

Corollary 3.4 *The product topologies on ${}^{1,1}\Sigma^*$ given by the discrete topology on Σ^* and the topology generated by $[p]^*$, respectively the Baire topology, are equivalent.*

The following key definition will be used to describe time and space complexity in a topological manner.

Definition 3.5 For any $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{U}$, $\mathbf{U}$ is called ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-*open* if for every (x, f) in $\mathbf{U}$ there is an interval $[p]^*$ containing f such that the following hold:
(i) $\{x\} \times [p]^* \subseteq \mathbf{U}$
(ii) $\mathrm{Card}(\mathrm{Dom}(p)) \leq |\mathbf{G}_0(x, f)|$
(iii) For all $w \in \mathrm{Dom}(p)$, $|w| \leq |\mathbf{G}_1(x, f)|$
(iv) For all $w \in \mathrm{Dom}(p)$, $|p(w)| \leq |\mathbf{G}_2(x, f)|$.

The use of the word 'open' is justified by (i). Intuitively, (ii) says that the number of oracle calls is bounded by $\mathbf{G}_0$; (iii) says that the size of the oracle input is bounded by $\mathbf{G}_1$; and (iv) says that the size of the part of the output actually used is bounded by $\mathbf{G}_2$.

Definition 3.6
(1) For any $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{U}$, $\mathbf{U}$ is ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-*closed* if $-\mathbf{U}$ is ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-open. $\mathbf{U}$ is ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-*clopen* if both $\mathbf{U}$ and $-\mathbf{U}$ are ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-open.
(2) For any Γ_0, Γ_1, Γ_2, and $\mathbf{U}$, $\mathbf{U}$ is (Γ_0, Γ_1, Γ_2)-*open* (*clopen*) if $\mathbf{U}$ is ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-open (clopen) for some $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$ in Γ_0, Γ_1, Γ_2, respectively.

For notational simplicity, we will identify a functional with the class consisting solely of that functional.

Example 3.7
(1) $\{(x, f) : f(w) = 1 \text{ for } w \leq x\}$ is ($POLY$, $POLY$, $\lambda(x, f).0^2$)-open.
(2) $\{(x, f) : f(w) = 1 \text{ for } |w| \leq |x|\}$ is (EXP, $POLY$, $\lambda(x, f).0^2$)-open but not ($POLY$, $POLY$, $\lambda(x, f).0^2$)-open, since membership of (x, f) depends on an exponential number of values of f.
(3) $\{(x, f) : f(10) = 11\}$ is ($\lambda(x, f).0$, $\lambda(x, f).0^2$, $\lambda(x, f).0^3$)-open but not ($\lambda(x, f).0$, $\lambda(x, f).0^2$, $\lambda(x, f).0^2$)-open, since membership of (x, f) depends on the fact that $f(10)$ is 11 rather than simply containing 11 as a suffix.
(4) $\{(x, f) : f(x) = 1\}$ is ($\lambda(x, f).0$, $\lambda(x, f).0^{|x|}$, $\lambda(x, f).0^2$)-open but not ($\lambda(x, f).0$, $\lambda(x, f).0$, $\lambda(x, f).0^2$)-open, since membership of (x, f) depends on an oracle input of length $|x|$.

Definition 3.5 requires us to consider three things: the number of oracle calls made, the size of the input for these calls, and the amount of information actually used from the results. The previous example shows that these are three different things; modification of one component can lead to a different class of open sets. We might also consider modifying the definition of the [ ]* intervals. We could, for example, require p to be a finite function whose domain is an initial segment of Σ^*, but this would be different from our definition because there are potentially 2^n oracle inputs of length n. The action of B is also critical because without it we cannot specify a function value with the resulting basic intervals.

We let $C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2}$ denote the class of ($\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$)-open sets, and let $C_{\Gamma_0,\Gamma_1,\Gamma_2}$ denote the class of (Γ_0, Γ_1, Γ_2)-open sets.

Proposition 3.8 *For any* $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$:
(1) $\varnothing \in C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2}$
(2) *If* $|\mathbf{G}_0|$, $|\mathbf{G}_2| \geq |\lambda(x, f).0|$ *and* $|\mathbf{G}_1| \geq |\lambda(x, f).\epsilon|$, *then* ${}^{1,1}\Sigma^* \in C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2}$

(3) $C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2}$ *is closed under arbitrary union*
(4) *For any* $\mathbf{H}_0$, $\mathbf{H}_1$, $\mathbf{H}_2$ *majorizing* $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, *respectively*, $C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2} \subseteq C_{\mathbf{H}_0,\mathbf{H}_1,\mathbf{H}_2}$.

Proof: (1) and (3) are easy. For (2) note that $f \in [\langle \epsilon, f(\epsilon) \downarrow 1\rangle]^*$. (4) follows from the definitions.

On the other hand, the following example shows that $C_{\mathbf{G}_0,\mathbf{G}_1,\mathbf{G}_2}$ need not be a topology.

Example 3.9 Let $\mathbf{A} = \{(x,f) : f(0) = 1\}$ and $\mathbf{B} = \{(x,f) : f(1) = 1\}$. Then $\mathbf{A} \cap \mathbf{B} = \{(x,f) : f(0) = f(1) = 1\}$. Note that $\mathbf{A}$ and $\mathbf{B}$ are $(\lambda(x,f).0, \lambda(x,f).0, \lambda(x,f).0^2)$-open but $\mathbf{A} \cap \mathbf{B}$ is not.

Definition 3.10
(1) Γ is *nontrivial* if there is an $\mathbf{F}$ in Γ such that for all (x,f), $|\mathbf{F}(x,f)| \geq 1$.
(2) Γ is *self-bounding* if for every $\mathbf{F}$ and $\mathbf{G}$ in Γ there is an $\mathbf{H}$ in Γ with $|\mathbf{F}| + |\mathbf{G}| \leq |\mathbf{H}|$.

Note that POLY, **POLY**, and *POLY* are self-bounding since they are closed under composition and length bounded exponentiation.

Proposition 3.11 *For any* Γ_0, Γ_1, Γ_2, Γ_0', Γ_1', Γ_2':
(1) If Γ_0, Γ_1, Γ_2 *are nonempty, then* $\varnothing \in C_{\Gamma_0,\Gamma_1,\Gamma_2}$
(2) *If* Γ_0, Γ_1, Γ_2 *are nontrivial, then* ${}^{1,1}\Sigma^* \in C_{\Gamma_0,\Gamma_1,\Gamma_2}$
(3) *If* Γ_0', Γ_1', Γ_2' *majorize* Γ_0, Γ_1, Γ_2, *respectively, then* $C_{\Gamma_0,\Gamma_1,\Gamma_2} \subseteq C_{\Gamma_0',\Gamma_1',\Gamma_2'}$.

If Γ_0, Γ_1, *and* Γ_2 *are self-bounding, then*
(4) $C_{\Gamma_0,\Gamma_1,\Gamma_2}$ *is closed under finite union*
(5) $C_{\Gamma_0,\Gamma_1,\Gamma_2}$ *is closed under finite intersection.*

Proof: (1), (2), and (3) follow from Proposition 3.8. For (4), let $\mathbf{U}_k$ be $(\mathbf{F}_k, \mathbf{G}_k, \mathbf{H}_k)$-open for $0 \leq k \leq n-1$. By the hypothesis, there exist $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ in Γ_0, Γ_1, and Γ_2, respectively, with $|\mathbf{F}| \geq \Sigma|\mathbf{F}_k|$, $|\mathbf{G}| \geq \max|\mathbf{G}_k|$, and $|\mathbf{H}| \geq \max|\mathbf{H}_k|$. It is easy to see that $\cup\mathbf{U}_k$ is $(\mathbf{F}, \mathbf{G}, \mathbf{H})$-open. For (5) let $\mathbf{F}_k$, $\mathbf{G}_k$, $\mathbf{H}_k$, $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ be as in (4). Suppose $(x,f) \in \cap\mathbf{U}_k$. Then for $k \leq n-1$, there exists $[p_k]^*$ with $(x,f) \in \{x\} \times [p_k]^*$. Define finite $p : \cup\mathrm{Dom}(p_k) \to \Sigma^* \cup \mathrm{B}\Sigma^*$ by

$$p(w) = \begin{cases} \mathrm{B}x, \text{ if there exists a } k \text{ with } p_k(w) = \mathrm{B}x \\ \text{longest of the } p_k(w), \text{ otherwise.} \end{cases}$$

Then $(x,f) \in \{x\} \times [p]^* \subseteq \cap\mathbf{U}_k$. Moreover, $\mathrm{Card}(\mathrm{Dom}(p)) \leq |\mathbf{F}(x,f)|$, and for every $w \in \mathrm{Dom}(p)$, $|w| \leq |\mathbf{G}(x,f)|$ and $|p(w)| \leq |\mathbf{H}(x,f)|$.

Example 3.9 shows that $C_{\Gamma_0,\Gamma_1,\Gamma_2}$ need not be closed under finite intersection, and the next two examples show that $C_{\Gamma_0,\Gamma_1,\Gamma_2}$ need not be closed under finite, union, and that it need not be closed under infinite union even if it is closed under finite union.

Example 3.12 Let $\mathbf{A} = \{(x,f) : f(x) = 0$ if $|x|$ is even, and 01 otherwise$\}$ and let $\mathbf{B} = \{(x,f) : f(x) = 0$ if $|x|$ is odd, and 01 otherwise$\}$. Let $g(x,f) = 2$ if $|x|$ is even and 3 otherwise, and let $h(x, f) = 2$ if $|x|$ is odd and 3 otherwise.

Then $\mathbf{A}$ is $(\lambda(x,f).0,\ \lambda(x,f).0^{|x|},\ g)$-open and $\mathbf{B}$ is $(\lambda(x,f).0,\ \lambda(x,f)0^{|x|},\ h)$-open, but $\mathbf{A} \cup \mathbf{B}$ is neither $(\lambda(x,f).0,\ \lambda(x,f).0^{|x|},\ g)$-open nor $(\lambda(x,f).0,\ \lambda(x,f)0^{|x|},\ h)$-open.

Example 3.13 $C_{POLY,POLY,POLY}$ is closed under finite union and intersection by Proposition 3.11. It is not closed under either infinite union or infinite intersection. Let $\mathbf{A}_k = \{(x,f) : f(w) = 1^{k+1} \text{ for } w \le \mathrm{St}(|x|^k)\}$. Each $\mathbf{A}_k$ is $(POLY,\ POLY,\ POLY)$-open and all are disjoint, but clearly $\cup\mathbf{A}_k$ is not $(POLY,\ POLY,\ POLY)$-open. Next let $\mathbf{B}_k = \{(x,f) : f(w) = 1 \text{ for } w \le \mathrm{St}(k)\}$. Then $\cap\mathbf{B}_k = \{(x,f) : f \text{ is identically } 1\}$, which is not an open set.

4 Topological properties of Σ_1^P and $\underline{\mathbf{SIG}}_1^P$ In this section, we present some results to the effect that in certain cases subsets of ${}^{1,1}\Sigma^*$ can be characterized both complexity theoretically and topologically. They suggest that topological complexity is a useful notion for relativized oracle computations.

Definition 4.1 For any $\mathbf{F}$, $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, Γ_0, Γ_1, and Γ_2:
(1) $\mathbf{F}$ is $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-*partial continuous* if $\mathbf{F}^{-1}(x)$ is $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-open for every x in $\mathrm{Im}(\mathbf{F})$;
(2) $\mathbf{F}$ is $(\Gamma_0, \Gamma_1, \Gamma_2)$-*partial continuous* if $\mathbf{F}$ is $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-partial continuous for some $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$ in Γ_0, Γ_1, Γ_2, respectively.

We drop the word 'partial' if $\mathbf{F}$ is total. (2) requires a uniform restriction on the openness of $\mathbf{F}^{-1}$, and hence the restriction in (2) is not equivalent to allowing $\mathbf{G}_k$ to depend on x.

Example 4.2
(1) $\mathbf{F}(x,f) = 1^{2^{|x|}}$ is $(POLY,\ POLY,\ POLY)$-continuous but is not deterministically computable in **POLY** time (otherwise, the bounds for characteristic functions f would be in POLY).
(2) The application functional is $(POLY,\ POLY,\ \mathbf{POLY})$-continuous and is computable in **POLY** time.

Proposition 4.3 *For any* $\mathbf{A}$, $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, Γ_0, Γ_1, *and* Γ_2:
(1) $\mathbf{A}$ *is* $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-*open iff* $\mathbf{A}$ *is the domain of a* $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-*partial continuous functional*
(2) $\mathbf{A}$ *is* $(\Gamma_0, \Gamma_1, \Gamma_2)$-*open iff* $\mathbf{A}$ *is the domain of a* $(\Gamma_0, \Gamma_1, \Gamma_2)$-*partial continuous functional.*

Proof: (1) $(\rightarrow)$: $\mathbf{K}_\mathbf{A}^+$ is $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-partial continuous and $\mathbf{A} = \mathrm{Dom}(\mathbf{K}_\mathbf{A}^+)$. $(\leftarrow)$: If $\mathbf{A}$ is the domain of some $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-partial continuous functional $\mathbf{F}$, then $\mathbf{A} = \cup\{\mathbf{F}^{-1}(x) : x \in \mathrm{Im}(\mathbf{F})\}$. (2) follows from (1).

Corollary 4.4 *For any* $\mathbf{A}$, $\mathbf{G}_0$, $\mathbf{G}_1$, $\mathbf{G}_2$, Γ_0, Γ_1, *and* Γ_2:
(1) $\mathbf{A}$ *is* $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-*clopen iff* $\mathbf{K}_\mathbf{A}$ *is* $(\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2)$-*continuous*
(2) $\mathbf{A}$ *is* $(\Gamma_0, \Gamma_1, \Gamma_2)$-*open iff* $\mathbf{K}_\mathbf{A}$ *is* $(\Gamma_0, \Gamma_1, \Gamma_2)$-*continuous.*

Proof: Immediate from the proof of the preceding proposition.

Example 4.5 The class of $(EXP,\ EXP,\ EXP)$-partial continuous functionals properly includes the class of $(EXP,\ EXP,\ POLY)$-partial continuous functionals. This follows from Proposition 4.3 by noting that $\mathbf{A} = \{(x,f) : f(w) = 1^{2^{|x|}}$ for $|w| \leq |x|\}$ is $(\mathit{EXP}, \mathit{EXP}, \mathit{EXP})$-open but not $(\mathit{EXP}, \mathit{EXP}, \mathit{POLY})$-open, because membership in $\mathbf{A}$ depends on an exponential amount of information from the oracle output.

In ordinary recursion theory, $\mathbf{A}$ is semi-recursive in some g iff $\mathbf{A}$ is open. Analogously, we have the following.

Theorem 4.6 *For any* $\mathbf{A}$,
(1) $\mathbf{A} \in \underline{\Sigma}_1^P$ *iff* $\mathbf{A}$ *is* $(\underline{\mathbf{POLY}},\underline{\mathbf{POLY}},\underline{\mathbf{POLY}})$*-open*
(2) $\mathbf{A} \in \underline{\mathrm{SIG}}_1^P$ *iff* $\mathbf{A}$ *is* $(\underline{\mathit{POLY}},\underline{\mathit{POLY}},\underline{\mathit{POLY}})$*-open.*

Proof: (1) $(\rightarrow)$: Suppose that $\mathbf{A}$ is accepted by nondeterministic oracle machine M operating in time $\mathbf{POLY}_w^h$. Suppose that $(x,f) \in \mathbf{A}$ and let u be an accepting computation. It follows that the number of oracle calls, the size of the inputs for the calls, and the amount of information used from the results of the calls in u are bounded by $|\mathbf{POLY}_w^h(x,f)|$. If $z_0, \ldots, z_{k-1}$ are the oracle inputs and if $g \in [\langle z_0, \mathrm{B}f(z_0) \downarrow |\mathbf{POLY}_w^h(x,f)|\rangle, \ldots, \langle z_{k-1}, \mathrm{B}f(z_{k-1}) \downarrow |\mathbf{POLY}_w^h(x, f)|\rangle]^*$, then u is an accepting computation for (x, g). $(\leftarrow)$: Let $\mathbf{A}$ be $(\mathbf{POLY}_u^{h_0}, \mathbf{POLY}_v^{h_1}, \mathbf{POLY}_w^{h_2})$-open. Define g by

$$g(\langle x, p\rangle) = \begin{cases} 1, \text{ if } (x,f) \in \mathbf{A} \text{ for all } f \in [p]^* \\ 0, \text{ otherwise.} \end{cases}$$

It follows that $(x,f) \in \mathbf{A}$ iff for some choice of $z_0, \ldots, z_{k-1}$ with $k \leq |\mathbf{POLY}_u^{h_0}(x,f)|$ and $|z_m| \leq |\mathbf{POLY}_v^{h_1}(x,f)|$, and some choice $Y_0, \ldots, Y_{k-1}$ where Y_m is either B or ϵ, we have $g(\langle x, \langle z_0, Y_0 f(z_0) \downarrow |\mathbf{POLY}_w^{h_2}(x,f)|\rangle, \ldots \rangle) = 1$. Therefore $\mathbf{A} \in \Sigma_1^P[\langle g, h_0, h_1, h_2\rangle]$ by the relativized version of Proposition 2.5.

The argument for (2) is nearly identical, using the fact that our coding allows us to obtain the necessary information about g, h_0, h_1, and h_2 in an efficient manner.

Note that the proof shows that if $\mathbf{A} \in \mathbf{NP}$, then $\mathbf{A}$ is $(\mathbf{POLY}, \mathbf{POLY}, \mathbf{POLY})$-open, and similarly for $\mathbf{A} \in \mathit{NP}$.

Corollary 4.7 *For any* $\mathbf{A}$:
(1) $\mathbf{A} \in \underline{\Sigma}_1^P$ *iff* $\mathbf{K}_\mathbf{A}^+$ is $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$*-partial continuous*
(2) $\mathbf{A} \in \underline{\Pi}_1^P$ *iff* $\mathbf{A}$ *is* $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$*-closed*
(3) $\mathbf{A} \in \underline{\Sigma}_1^P \cap \underline{\Pi}_1^P$ *iff* $\mathbf{K}_\mathbf{A}$ *is* $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$*-continuous iff* $\mathbf{A}$ is $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}\ \underline{\mathbf{POLY}})$*-clopen*
(4) $\mathbf{A} \in \underline{\mathrm{SIG}}_1^P$ *iff* $\mathbf{K}_\mathbf{A}^+$ *is* $(\underline{\mathit{POLY}}, \underline{\mathit{POLY}}, \underline{\mathit{POLY}})$*-partial continuous*
(5) $\mathbf{A} \in \underline{\mathrm{PI}}_1^P$ *iff* $\mathbf{A}$ *is* $(\underline{\mathit{POLY}}, \underline{\mathit{POLY}}, \underline{\mathit{POLY}})$*-closed*
(6) $\mathbf{A} \in \underline{\mathrm{SIG}}_1^P \cap \underline{\mathrm{PI}}_1^P$ *iff* $\mathbf{K}_\mathbf{A}$ *is* $(\underline{\mathit{POLY}}, \underline{\mathit{POLY}}, \underline{\mathit{POLY}})$*-continuous iff* $\mathbf{A}$ *is* $(\underline{\mathit{POLY}}, \underline{\mathit{POLY}}, \underline{\mathit{POLY}})$*-clopen.*

Proof: (1), (2), (4), and (5) are immediate. (3) and (6) follow since $\underline{\mathbf{POLY}}$ and $\underline{\mathit{POLY}}$ are self-bounding.

Clearly, any set in $\underline{\Sigma}_1^P$ is recursive in some g, and hence there are sets which are open but not $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$-open. Before presenting the next theorem, we make several comments about Theorem 4.6. We would like to say, in analogy with ordinary recursion theory, that **F** is nondeterministically computable in some g in **POLY** time iff it is $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$-(partial) continuous. The first problem is illustrated by Example 4.2, which indicates that some restriction on the size of the value of **F** is necessary. An obvious solution is to require that **F** be **POLY** bounded. Our proposed result would say that if **F** is **POLY** bounded, then **F** is nondeterministically computable in **POLY** time iff **F** is $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$-(partial) continuous. Recall that a nondeterministic machine computes value z if some halting computation has output z and no other halting computation has output $w \neq z$. It is still the case that if u is a halting computation for (x, f) with oracle calls $z_0, \ldots, z_k$ and at most $|\mathbf{POLY}^h_w(x, f)|$ steps, then u is a halting computation for any (x, g) with $g \in [\langle z_0, \mathrm{B}f(z_0) \downarrow |\mathbf{POLY}^h_w(x, f)|\rangle, \ldots]^*$. However, we do not know what happens with the other computations. We do though have the following.

Proposition 4.8 *For any* **F**:
(1) *If* **F** *is* **POLY** *bounded and* $(\underline{\mathbf{POLY}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$-*(partial) continuous, then* **F** *is nondeterministically computable in* **POLY** *time using some oracle g*
(2) If **F** *is POLY bounded and* $(\underline{POLY}, \underline{POLY}, \underline{POLY})$-*(partial) continuous, then* **F** *is nondeterministically computable in POLY time using some oracle g.*

Proof: Let **F** be $(\mathbf{POLY}^{h_0}_u, \mathbf{POLY}^{h_1}_v, \mathbf{POLY}^{h_2}_w)$-(partial) continuous. Define g by

$$g(\langle x,p\rangle) = \begin{cases} w1, \text{ if } \mathbf{F}(x,f) = w \text{ for all } f \in [p]^* \\ 0, \text{ otherwise.} \end{cases}$$

It follows that $\mathbf{F}(x, f)$ is defined iff for some choice of $z_0, \ldots, z_{k-1}$ with $k \leq |\mathbf{POLY}^{h_0}_u(x, f)|$ and $|z_m| \leq |\mathbf{POLY}^{h_1}_v(x, f)|$, and some choice $Y_0, \ldots, Y_{k-1}$ where Y_m is either B or ϵ, we have $g(\langle x, \langle z_0, Y_0 f(z_0) \downarrow |\mathbf{POLY}^{h_2}_w(x, f)|\rangle, \ldots\rangle) = w1$. This represents a nondeterministic **POLY** bounded computation using an oracle for $\langle g, h_0, h_1, h_2\rangle$. Moreover, if for each such choice of z_m and Y_m, $g(\langle x, \ldots\rangle) = w1$, then $\mathbf{F}(x, f) = w$. Hence **F** is nondeterministically computable in **POLY** time using an oracle for $\langle g, h_0, h_1, h_2\rangle$. The argument for (2) is nearly identical.

The above results reinforce the claim that our definitions are 'natural' when considering complexity-theoretic constraints. Moreover, we have the following result.

Theorem 4.9 *For any* **A**:
(1) $\mathbf{A} \in \underline{\mathbf{NPSPACE}}$ *iff* **A** *is* $(\underline{\mathbf{EXP}}, \underline{\mathbf{POLY}}, \underline{\mathbf{POLY}})$-*open*
(2) $\mathbf{A} \in \underline{NPSPACE}$ *iff* **A** *is* $(\underline{EXP}, \underline{POLY}, \underline{POLY})$-*open.*

Proof: (1) ($\rightarrow$): Let M be a nondeterministic machine accepting **A** in $\mathbf{POLY}^h_w$ space. Suppose that $(x, f) \in \mathbf{A}$ and let u be an accepting computation using at most $|\mathbf{POLY}^h_w(x, f)|$ cells. It follows that for some n, the number of oracle calls, the size of the oracle inputs for the calls, and the amount of information used from these calls is bounded by $n^{|\mathbf{POLY}^h_w(x,f)|}$, $|\mathbf{POLY}^h_w(x, f)|$, $|\mathbf{POLY}^h_w(x, f)|$, respectively. The rest of the argument proceeds as in ($\rightarrow$) of Theorem 4.6.
($\leftarrow$): Suppose that **A** is $(\lambda(x, f).0^{n^{|\mathbf{POLY}^{h_0}_u(x,f)|}}, \mathbf{POLY}^{h_1}_v, \mathbf{POLY}^{h_2}_w)$-open. Define g by

$$g(\langle x,\langle z,y\rangle,s\rangle) = \begin{cases} 01, & \text{if } \langle z,y\rangle \text{ is the Num}(s)\text{th and last member of some } p \text{ with } (x,f) \in \mathbf{A} \text{ for all } f \in [p]^* \\ 1, & \text{if } \langle y,z\rangle \text{ is the Num}(s)\text{th but not the last member of such a } p \\ 0, & \text{otherwise.} \end{cases}$$

We describe a **POLY** space bounded nondeterministic machine M using an oracle for $\langle g, h_0, h_1, h_2\rangle$ which accepts **A**. On input (x, f), M guesses a z_0 and Y_0 where Y_0 is either B or ϵ and $|z_0| \leq |\mathbf{POLY}_v^{h_1}(x,f)|$, and asks g about $(\langle x, \langle z_0, Y_0 f(z_0) \downarrow |\mathbf{POLY}_w^{h_2}(x,f)|\rangle, \epsilon\rangle)$. If the answer is 01, then M accepts. If the answer is 0, then M goes into an infinite loop. Otherwise, M guesses a z_1 and $Y_1, \ldots$, and so on. The restriction on space is that M must keep track of the last s, and be able to ask about strings of the form $\langle x, \langle z, y\rangle, s\rangle$. Note that the lengths of z and y are assumed to be bounded and $\text{Num}(s) \leq n^{|\mathbf{POLY}_u^{h_0}(x,f)|}$ if $(x,f) \in \mathbf{A}$. Hence s is also bounded in length if $(x, s) \in \mathbf{A}$, so that M runs in **POLY** space. The argument for (2) is similar.

Note that the proof shows that if **A** is in **NPSPACE** then A is (**EXP**, **POLY**, **POLY**)-open, and similarly for $\mathbf{A} \in$ *NPSPACE*.

5 Applications We close by presenting two applications of the results of the previous sections. First, we consider generalizations of the polynomial jump operator. The *polynomial jump* of A, denoted by K(A), is defined by $\text{K(A)} = \{(x, y, 0^n) : \text{NOTM}_x^A \text{ accepts } y \text{ in at most } n \text{ steps}\}$. The basic properties of the jump are that K(A) is NP in A, and A is NP in B iff $\text{A} \leq_{\text{m}}^{\text{P}} \text{K(B)}$. A more detailed discussion of the jump is contained in Townsend [13]. We can extend the jump operation to functions as follows.

Definition 5.1 For $f \in {}^{\Sigma^*}\Sigma^*$, we define $\text{K}(f)$, the *jump* of f, by

$$\text{K}(f)(\langle x,y,0^n\rangle) = \begin{cases} 0, & \text{if NOTM}_x^f \text{ accepts } y \text{ in at most } n \text{ steps} \\ 1, & \text{otherwise.} \end{cases}$$

$\text{K}(f)(z)$ is also 0 if z is not of the form $\langle x, y, 0^n\rangle$.

Corresponding to the previously described properties of the jump for sets we have only the following.

Proposition 5.2 *For any f and* A, $\text{A} \in \text{SIG}_1^P[f]$ *implies* $\text{A} \in$ *POLY*$[\text{K}(f)]$.

Proof: Suppose that $y \in \text{A}$ iff $\mathbf{R}(y,f)$ for some $\mathbf{R} \in \text{SIG}_1^P[f]$. Then $\mathbf{R}$ is accepted by a NOTM_x^f running in time POLY_w, so that $y \in \text{A}$ iff $\text{K}(f)(\langle x, y, 0^{|\text{POLY}_w(x)|}\rangle) = 0$.

There is a problem in extending the previous proposition to **POLY**. Following the proof, we would have $y \in \text{A}$ iff $\text{K}(f)(\langle x, y, 0^{|\mathbf{POLY}_w(x,f)|}\rangle) = 0$. This, however, may require values of f. We can avoid this by defining $\text{K}'(f) = \langle f, \text{K}(f)\rangle$. Then we have $\text{A} \in \Sigma_1^P$ implies $\text{A} \in \mathbf{POLY}[\text{K}'(f)]$. As the next example shows, we cannot reverse the implication in Proposition 5.2.

Example 5.3 Let A be such that $\mathrm{NP}^{\mathrm{A}} \neq \text{CO-NP}^{\mathrm{A}}$ (see [1]), and take $\mathrm{R} = -\mathrm{K}(\mathrm{A})$. Then using an oracle for the jump of K_{A}, R is deterministically computable in POLY time. $\mathrm{R} \notin \Sigma_1^P[\mathrm{K}_{\mathrm{A}}]$ because otherwise we would have $-\mathrm{K}(\mathrm{A}) \leq_{\mathrm{m}}^{\mathrm{P}} \mathrm{K}(\mathrm{A})$, which contradicts the assumption about A.

In any case, the restriction of these results to relations on strings is necessary if $\Sigma_1^P - \underline{\Sigma}_0^P \neq \emptyset$ ($\mathrm{SIG}_1^P - \underline{\mathrm{SIG}}_0^P \neq \emptyset$), because if **R** is in **NP**(*NP*) but not in $\underline{\Sigma}_o^P$ ($\underline{\mathrm{SIG}}_0^P$), then **R** is **NP** (*NP*) but not in **POLY**[g] (*POLY*[g]) for any g.

Finally, recall the following open questions.

(1) NP = PSPACE?
(2) NPSPACE = EXP?

Compare these with the following.

Proposition 5.4
(1) **NP ≠ PSPACE**
(2) *NPSPACE ≠ EXP.*

Proof: (1) Let $\mathbf{A} = \{(x,f) : f(w) = 1 \text{ for } |w| \leq |x|\}$. Clearly, **A** is in *PSPACE.* We show that **A** is not (**POLY, POLY, POLY**)-open. If $g = \lambda x.1$, then for all x, $(x, g) \in \mathbf{A}$. We have already mentioned that for every $\mathbf{POLY}_y$ there is a POLY_z such that for all characteristic functions f and all x, $|\mathbf{POLY}_y(x,f)| \leq |\mathrm{POLY}_z(x)|$. Since membership in **A** requires information about exponentially many values of f, **A** is not (**POLY, POLY, POLY**)-open.

(2) Define $\mathbf{B} = \{(x,f) : f(w) = 1^{2^{|x|}} \text{ for } |w| \leq |x|\}$. Then **B** is in *EXP* but is not (*EXP, POLY, POLY*)-open.

6 *Conclusion*

Type-2 recursion theory extends ordinary recursion theory by permitting arguments that are functions. Historically, type-2 recursion theory is the recursion-theoretic end of an interface with descriptive set theory. Thus, the subject has a 'feel' somewhere between recursion theory and topology. We have developed a polynomialization of type-2 recursion theory. This development is parallel to the polynomialization of ordinary recursion theory that is the basis for structural complexity theory. In addition, we have considered some topological notions associated with time and space bounded computations of oracle Turing machines. The results presented suggest that topological considerations are an integral part of the study of resource bounded computations.

REFERENCES

[1] Baker, T. P., J. Gill, and R. Solovay, "Relativizations of the P = ? NP question," *SIAM Journal*, vol. 4 (1975), pp. 431-442.

[2] Baker, T. P. and A. L. Selman, "A second step towards the polynomial hierarchy," *Theoretical Computer Science*, vol. 8 (1979), pp. 177-187.

[3] Buss, S., *Bounded Arithmetic*, dissertation, Princeton University, 1985.

[4] Cobham, A., "The intrinsic computational difficulty of functions," pp. 24-30 in *Proceedings of the 1964 Congress for Logic, Mathematics, and Philosophy of Science*, North Holland, Amsterdam, 1964.

[5] Hinman, P. G., *Recursion-Theoretic Hierarchies*, Springer-Verlag, New York, 1978.

[6] Hopcroft, J. E. and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Massachusetts, 1979.

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

[8] Mehlhorn, K., "Polynomial and abstract subrecursive classes," *Journal of Computer and System Sciences*, vol. 12 (1976), pp. 147–178.

[9] Munkres, J. R., *Topology: A First Course*, Prentice-Hall, New York, 1975.

[10] Rogers, H., *The Theory of Recursive Functions and Effective Computability*, McGraw-Hill, New York, 1967.

[11] Savitch, W. J., "Relationships between nondeterministic and deterministic tape complexities," *Journal of Computer and System Sciences*, vol. 4 (1970), pp. 177–192.

[12] Townsend, M., *A Polynomial Jump Operator and Complexity for Type Two Relations*, dissertation, University of Michigan, 1982.

[13] Townsend, M., "A polynomial jump operator," *Information and Control*, vol. 68 (1986), pp. 146–170.

[14] Yao, A., "Separating the polynomial-time hierarchy by oracles, I," pp. 1–11 in *Proceedings of the 26th Annual Symposium on the Foundation of Computer Science*, 1975.

Department of Mathematics
Harvey Mudd College
Claremont, California 91711