GENERALIZED BOUNDED QUERY HIERARCHIES

by

Meera Sitharam

Computer Sciences Technical Report #961

August 1990

GENERALIZED BOUNDED QUERY HIERARCHIES

by

MEERA SITHARAM

A thesis submitted in the partial fulfillment of the
requirements for the degree of

Doctor of Philosophy
(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN-MADISON

1990

GENERALIZED BOUNDED QUERY HIERARCHIES

Meera Sitharam

Under the supervision of Professor Deborah A. Joseph
at the Department of Computer Sciences
University of Wisconsin-Madison

Abstract.

The power of an oracle computation is determined by three basic factors: the complexity of the computation, the complexity of the oracle and oracle access restrictions. Although it is natural to expect that the influence of these factors are correlated, this thesis establishes that the influence of the third factor is largely independent of the first two. In other words, we establish that the influence of natural oracle access restrictions on the power of an oracle computation remains the same for all reasonable and natural choices of oracle and computation complexities. We consider the following access restrictions in oracle computations: 1. restrictions on the number of queries to the oracle, and 2. restrictions on how the oracle's answers are used by the computation. The influence of these restrictions on polynomial-time computations that make a constant number of queries to *NP* oracles has been much investigated in recent years, since many interesting problems can be solved by such restricted computations. We give uniform generalizations of these studies to general oracle classes that include *RNC, RP, NEXP, RE,* etc; general computation complexities that, for instance, include any

deterministic complexity class stronger than log-space; and general bounds on the number of queries.

Acknowledgements

My advisor, Deborah Joseph, has taken the time to instruct me on how to start formalizing an idea and how to present a result. She introduced me to Paul Young, Danilo Bruschi, and Tim Long. Working with the four of them, I learnt much. Paul Young provided the main push to work on the problems that this thesis is based on, and a big portion of Chapter 2 relates to work done jointly with him. Anne Condon, and Eric Bach patiently sat through my talks, read my thesis with care and have given helpful comments on both. Kenneth Kunen has given me much time, inspiration and ideas on just about any research problem.

Giri Narasimhan's enthusiasm about Theory and his help during my screening exam preparations contributed much to my initial involvement in the area. Judy Goldsmith has given me much useful advice from her experiences as a starting researcher, and has been a source of encouragement throughout; Kirk Pruhs gave me his thesis format, and practical suggestions that have mitigated the aggravations of finishing up; and Jörg Peters and Gautam Das have gone through the finishing up process with me and have shared these aggravations. All of these people and Divesh Srivastava have discussed work with me at some level, making it more fun.

Narendran, Victor Shoup, Jon Sorenson and Marty Wolf have all been fun colleagues, sat through many a practice talk and given useful suggestions.

I thank all of these people.

Table of Contents

Chapter 1

Introduction and unified review.

The power of an oracle computation is determined by three basic factors: the complexity of the computation, the complexity of the oracle and the oracle access restrictions. Although it is natural to expect that the influence of these factors are correlated, this thesis establishes that the influence of the third factor is largely independent of the first two. In other words, we establish that the influence of natural oracle access restrictions on the power of an oracle computation remains the same for all reasonable and natural choices of oracle and computation complexities.

We consider the following access restrictions in oracle computations:

1. restrictions on the number of queries to the oracle, and
2. restrictions on how the oracle's answers are used by the computation.

Since many interesting problems can be solved by computations with such restrictions, the influence of these restrictions on polynomial-time (resp. *NP)* computations with *NP* oracles has been much investigated in recent years in [PaYa 82], [WeWa 85], [Kre 86], [CaHe 86], [KöScWa 87], [Be 87], [Ka 88], (resp. [BoWr 81], [BoLoSe 84,85], [Lo 85]) et cetera (see surveys in [Wa 88] and [CGHHSWW 88,89]). It was shown in [PaYa 82] that while many of these problems require computations that make a constant number of queries to an *NP* oracle, they depend on both positive and negative answers from the oracle, so that the membership of these problems in *NP* would imply

that $NP = coNP$. The following is a standard example of an optimization problem whose solution requires one positive and one negative answer from an NP oracle.

EXACT OPTIMUM TSP ROUTE.

Instance: An $m \times m$ distance matrix denoting the (integer) weights on the edges of a complete graph, and an integer M.

Question: Is it true that the optimum TSP route has cost M? (Is it true that there is TSP route of cost M, but there is no TSP route of cost $< M$?)

The transparent definition of this problem makes clear that while both SAT and $\overline{SAT}$ can be $\leq_m^P$-reduced to it, neither of the reverse reductions is possible unless $NP = coNP$.

Besides requiring a constant number of queries to the NP oracle, determining membership in such languages as *EXACT OPTIMUM TSP* requires little processing of the oracle's answers: a simple fixed Boolean combination of the oracles answers determines membership.

This leads us to the second oracle access restriction under consideration: how does the oracle computation utilize the oracle's answers? Many possibilities arise: Does the computation use the oracle's answers to previous queries to decide what future queries will be, or does the computation make all the queries together (in parallel)? Is there an arbitrary computation after the oracle's answers are received, or are the oracle's answers processed in a highly restricted manner? For computations with an NP oracle, the above questions have been investigated quite thoroughly in recent years. (See surveys in [Wa 88] and [CGHHSWW 88,89]).

All of the papers mentioned so far are mainly geared towards the study of oracle access restrictions on polynomial-time (or sometimes log-space) computations with NP oracles. Naturally, therefore, the proofs that appear in these papers utilize properties that are specific to these complexity classes. In addition, most of these papers severely narrow the range of possibilities for the first oracle access restriction under consideration: the results in most of these papers deal only with computations that make a constant number of queries to an oracle, although a restriction on the number of queries to the oracle does not prevent this number from being a slowly growing function of the input.

This thesis establishes that many of these earlier results - predominantly about polynomial-time computations that make a constant number of queries to an NP oracle and use the oracle's answers in a restricted manner - generalize naturally in the following sense: they extend to analogous results about fairly general computations that make a bounded but not necessarily constant number of queries to oracles from fairly general complexity classes, but still use the oracle's answers in a restricted manner. In other words, the influence of natural oracle access restrictions on the power of oracle computations remains the same whether the oracle class is NP or any other natural complexity class, and whether the computation (or reduction) has polynomial-time complexity or any other natural complexity.

Before providing a formal statement of the main result, and some comments on how we proceed with its proof, we first subject the reader to a description of our basic notation.

Preliminary definitions and notation.

We assume the reader's familiarity with well-known classes such as $L =$

$DSPACE[O(\log\ n)]$, NC, P, NP, $coNP$, RP, BPP, EXP, $NEXP$, UP, $FewP$, *sparse sets, tally sets,* and with the standard notation used in complexity theory (see [Sc] and [BaDiGa]).

Unless otherwise mentioned, all languages consist of strings over $\{0,1\}$. The letters, X, Y, Z, usually denote languages, and the letters, B, C, D, E, F, Q, R denote specific complexities, their corresponding complexity classes, or function classes depending on the context. For instance, one could refer to an algorithm of complexity C, a complexity class C, or a function class (of complexity) C. The letters, f, g, h, r, s, t, u, v usually denote functions, a, b vectors, k, l, m, constants, and y, z, strings. The letter, L, is reserved for the complexity class $DSPACE[O(\log\ n)]$, N, for the set of natural numbers, x, for the input string, and n, for the length of the input. We denote the $\{0,1\}$-valued characteristic function of a set, X, by $c_X(.)$.

For sets, X, Y, complexity class, C, and function, r, we say $X \leq^C_{r-red} Y$ if membership in X can be determined by an algorithm of complexity C that makes at most $r(n)$ queries to an oracle, Y, to which the oracle returns values of c_Y for the queried strings. (Recall that n is the length of the input). We refer to the function, r, as the **norm** of the reduction. If the above algorithm formulates a list of all the queries to be made to Y before actually querying Y, then the **bounded query reduction** is **nonadaptive.** Otherwise the reduction is **adaptive.** The **bounded query class,** $C^D_{red}[r]$, over a general **base** class, D, consists of all sets, X, for which $X \leq^C_{r-red} D$, i.e, there is a set $Y \in D$ such that $X \leq^C_{r-red} Y$. If the function, r, above is in fact a constant, k, then $\leq^C_{k-red}$-reductions are called **constant query reductions** and $C^D_{red}[k]$ is called a **constant query class.**

General bounded query nonadaptive reductions, $\leq^C_{r-red}$, are in general no stronger than (many-one, single query) $\leq^C_m$-reductions, and no weaker

than (bounded query Turing) $\leq^C_{r-T}$-reductions, which are adaptive. Recalling that weaker reductions generate larger bounded query classes, we have: $D = C^D_m \subseteq C^D_{red}[r] \subseteq C^D[r]$. The reader should note that we usually omit the subscript, T. Bounded query nonadaptive reductions have been studied since the 1960's in the context of recursion theory ([Ro 67]) and their polynomial-time bounded versions were first studied in [LaLySe 75].

Bounded query nonadaptive reductions fall into various categories. When $X \leq^C_{r-red} Y$, the reduction algorithm that determines membership in X could perform an arbitrary computation in C after Y's answers to a list of parallel queries is obtained. Alternatively, the reduction algorithm could input the list of Y's answers into a Boolean circuit, formula, or full truthtable whose output determines membership in X. We now give formal definitions for the bounded query classes associated with the above types of nonadaptive reductions.

1. $X \in \mathbf{C}^{\mathbf{D}}_{||}[\mathbf{r}] \iff \exists Y \in D$ and an algorithm, M, of complexity, C, such that on input, x, M computes a tuple, $(x_1, \ldots, x_{r(n)})$, of queries to Y and then decides membership of $x \in X$ by an arbitrary computation in C on the tuple, $(c_Y(x_1), \ldots, c_Y(x_{r(n)}))$, of answers returned by Y.

2. $X \in \mathbf{C}^{\mathbf{D}}_{\mathbf{tt}}[\mathbf{r}] \iff \exists Y \in D$ and an algorithm, M, of complexity, C, such that on input, x, M computes a tuple, $(t, x_1, \ldots, x_{r(n)})$, a representation, t, of a Boolean circuit with $r(n)$ inputs, and $r(n)$ queries to Y so that $x \in X \iff$ the circuit (represented by) t outputs 1 on the tuple, $(c_Y(x_1), \ldots, c_Y(x_{r(n)}))$.

A $\leq^{\mathbf{C}}_{\mathbf{btt}}$-reduction is a $\leq^C_{k-tt}$-reduction for some constant k. Hence, a set, X, is $\leq^C_{btt}$-complete for a class, D, if, for all $Y \in D$, there is a constant, k, such that $Y \leq^C_{k-tt} X$.

The next two definitions are obtained from the one above by replacing "Boolean circuit" by "Boolean formula" and "full truthtable" respectively.

3. $X \in \mathbf{C}^{\mathbf{D}}_{\mathbf{bf}}[\mathbf{r}] \iff \exists Y \in D$ and an algorithm, M, of complexity, C, such that on input, x, M computes a tuple, $(t, x_1, \ldots, x_{r(n)})$, consisting of a representation, t, of a Boolean formula in $r(n)$ variables, and $r(n)$ queries to Y so that $x \in X \iff$ the formula (represented by) t evaluates to 1 when its variables assume the values, $c_Y(x_1), \ldots, c_Y(x_{r(n)})$.

4. $X \in \mathbf{C}^{\mathbf{D}}_{\mathbf{ftt}}[\mathbf{r}] \iff \exists Y \in D$ and an algorithm, M, of complexity, C, such that on input, x, M computes a tuple, $(t, x_1, \ldots, x_{r(n)})$, consisting of a representation, t, of a full $2^{r(n)} \times (r(n)+1)$ truthtable in $r(n)$ variables, and $r(n)$ queries to Y so that $x \in X \iff$ the truthtable (represented by) t evaluates to 1 when its variables assume the values, $c_Y(x_1), \ldots, c_Y(x_{r(n)})$.

We defer the formal definition of Boolean hierarchies to Section 1.1. Intuitively, Boolean hierarchies are generated by a simpler type of nonadaptive reduction that the ones defined above. The oracle set, Y, is replaced by a collection of k sets from a base class, C, and the reduction algorithm makes one query to each set in this collection. If membership in X is determined by a *fixed Boolean combination* of the oracles' answers, then X falls into the k^{th} level of a **Boolean hierarchy** over C. Higher levels of these hierarchies are formed by allowing larger collections of oracles and more queries. A natural extension - allowing the number of oracle queries to be a slowly growing function of the input, while still deciding membership as an easily specifiable Boolean combination of the oracles' answers - generates an **extended Boolean hierarchy** over C.

Main result of this thesis.

We give a rigorous and uniform justification of the following statement.

- Various definitions of Boolean hierarchies over fairly general complexity classes are equivalent just as in the case of NP. Furthermore, at least at lower levels, all extended Boolean hierarchies, satisfy these same definitional equivalences. Moreover, the characteristics of bounded query classes defined by polynomial time computations that make a constant number of queries to an oracle in NP, for example, the influence of the number of queries, and the the relationship between adaptive and nonadaptive queries, extend to fairly general computations that query oracles from fairly general complexity classes, C. For instance, C could be RP, $FewP$, RNC, $NEXP$, or the recursively enumerable sets, RE.

The development of this thesis calls for a few general comments. First, since our aim is to provide a uniform treatment of all the complexity classes and computations (reductions) mentioned in the above statement, a sizable portion of this thesis, especially Chapter 2, will be devoted to developing the required machinery, and to formalizing the definitions of "fairly general" complexity classes and computations. We note that these definitions are based on closure under certain basic operations and are quite unrestrictive; for instance, the definition of "fairly general" base classes admits complexity classes that have no complete sets.

Second, a rigorous justification of the above statement involves the generalization of earlier work that is specifically tuned toward polynomial-time computations with NP oracles. We take this opportunity to make a careful and comprehensive review of such earlier studies. Although surveys on this topic already exist in [Wa 88] and [CGHHSWW 88,89], the intent to gener-

alize in a new direction gives our review a substantially different character, by finding a new common thread with which to tie earlier results together. For instance, along with the proof of each earlier result we will analyze those specific assumptions made in the proof that do not hold for general classes or reductions, thus setting the stage for a rigorous separation of superfluous assumptions from those that are, in fact, necessary for the proof to go through. Moreover, we review those earlier, narrower generalizations with a similar view towards reducing superfluity and providing sharper insight. As a consequence, our review yields more "minimalistic" proofs of earlier results.

Third, the development of this thesis will revolve around the classes and hierarchies that are generated by the simplest type of bounded query reductions, namely Boolean hierarchies. Investigations of more complicated bounded query classes will be built on this foundation.

Organization.

This thesis consists of two chapters.

Chapter 1 is divided into six subsections and reviews earlier studies on Boolean hierarchies and relevant bounded query classes in a unified manner that prepares the reader for the generalizations of the following chapter.

Section 1.1 gives the necessary background to familiarize the reader with the Boolean hierarchy over NP.

Section 1.2 gathers from the literature a sizable collection of example problems for the Boolean hierarchy and bounded query classes over NP and presents them under an accessible classification.

Section 1.3 reviews the known properties of the Boolean hierarchy over NP. The proofs of those results that are to be generalized in Chapter 2 are

analyzed carefully to pinpoint the assumptions that are specific to Boolean functions of constant norm and to the base class NP. In addition the notation required in these proofs is carefully set up to facilitate their generalization in Chapter 2.

Section 1.4 reviews earlier generalizations of the Boolean hierarchy over NP to Boolean functions of nonconstant norms, and earlier investigations of the hierarchy's relationship to bounded query classes that are based on various types of reductions. Here again, the assumptions that are specific to the base class NP are pinpointed. This is followed by a review of an earlier study of Boolean hierarchies over general base sets.

Section 1.5 provides a table of results both from earlier work and from this thesis, compares them and thus puts the latter in perspective.

Section 1.6 touches briefly upon various aspects of bounded query classes that have been studied in the literature but are not strongly related to this thesis.

Chapter 2 develops the machinery required for the definition and uniform treatment of (extended) Boolean hierarchies over general complexity classes, formalizes the notions of "fairly general" complexity classes and reductions, generalizes results from Chapter 1 and thereby proves the main result of this thesis.

A more detailed note on the organization of Chapter 2 can be found in Chapter 2 itself.

1.1. The Boolean hierarchy over NP: background and definitions.

Boolean hierarchies are typically formed by taking classes of sets that are closed under union and intersection but not complementation, and then forming a Boolean closure of the classes by iterating closure under operations involving union, intersection *and* complementation. We start with basic definitions.

Definition 1:
Let t_k be any k-ary Boolean function. Then for any collection, C of sets in $\{0,1\}^*$,

$$\mathbf{t_k[C]} =_{def} \{ X : \exists X_1, \ldots, X_k \in C \ [c_X(x) = t_k(c_{X_1}(x), \ldots, c_{X_k}(x)) \ ] \}.$$

•

That is, each set in $t_k[C]$ is a specific Boolean combination, defined by t_k, of the finite collections of sets taken k at time from the class, C. Boolean hierarchies are typically obtained by taking some reasonable collection of Boolean functions, $\{\lambda k \ t_k\}$, that are uniformly specifiable from k, and which are so chosen that the containments

$$t_1[C] \subseteq t_2[C] \subseteq \ldots \subseteq t_i[C] \ldots$$

are obvious. True hierarchies are obtained when the containments turn out to be proper.

¿From another point of view, the k^{th} level of a Boolean hierarchy over some base class is a constant query class defined by a nonadaptive *fixed* truthtable reduction that makes one query to each of k oracles from the base class. The truthtable involved in the reduction is invariant for all the sets in the k^{th} level and is uniformly generated from k. However, the collection of k oracle sets could be different for each set in the k^{th} level.

Notice that for a class like NP that is closed under $\leq_m^P$-reductions (Chapter 2 contains a detailed list of the far less restrictive list closure operations that are, in reality, involved) and has $\leq_m^P$-complete sets, classes $t_k[NP]$ can be alternatively defined as follows.

Definition 2:
For each k, let t_k be a k-ary Boolean function. Then

$$t_k[NP] =_{def} \{ X : \exists Y \in NP \text{ and } f \text{ a function computable in polynomial-time such that } f(x) = (x_1, \ldots, x_k) \text{ and } [\, c_X(x) = t_k(c_Y(x_1), \ldots, c_Y(x_k)) \,] \}.$$

•

In other words, a level of a Boolean hierarchy over a class like NP, is its closure under *fixed* truthtable reduction of constant norm.

As early examples, Boolean hierarchies were studied by Hausdorff in 1914 in the context of descriptive set theory ([Ha 78]), and the Boolean hierarchy over the recursively enumerable sets was defined and studied by Eršov ([Er 68a], [Er 68b], [Er 69]) who in [Er 68b] noted the connections between its transfinite extension through the recursive ordinals and the work done by Putnam, [Pu 65], on "mind changes" of recursive sequences. Together, the work of Putnam on mind changes of recursive sequences and of Eršov on extensions of Boolean hierarchies through recursive ordinals shows that the use of either truthtables or mind changes provides characterizations of the class $\Sigma_2^0 \cap \Pi_2^0$ in the arithmetic hierarchy. Independent of the work of Putnam and Eršov, Boolean hierarchies were also applied to Logic by Addison ([Ad 65]).

A Boolean hierarchy over NP was introduced in [PaYa 82] in order to classify the complexity of certain combinatorial optimization problems. Boolean hierarchies over NP have since been defined in various ways, in [WeWa 85], [CaHe 86], and [Kö 85] and [KöSchWa 87], who chose various sequences of Boolean functions for t_k in Definition 1. These papers study structural properties of the hierarchy and most of the results that appear in them are contained either in an excellent survey by Wagner ([Wa 88]) or in the journal versions, [CGHHSWW 88,89]. For regular Boolean hierarchies over NP, the various definitions used by these authors have all been proven equivalent ([WeWa 85], [KöSchWa 87]; elaborated in Section 1.3, Theorem 5).

Definition 3:

1. The **Difference hierarchy** was defined over NP by Köbler and Schöning ([Kö 85], [KöScWa 87]), and was denoted by them as $Diff_k =_{def} parity_k[NP]$, where

$$parity_k(x_1, \ldots, x_k) = x_1 \oplus \ldots \oplus x_k \qquad (\oplus = \text{xor}).$$

2. The **Wechsung-Wagner hierarchy** over NP was defined by Wechsung and Wagner ([WeWa 85]). They denoted it as $g_k[NP]$, where

$$g_k(x_1, \ldots, x_k) = (\neg x_1 \wedge x_2) \vee \ldots \vee (\neg x_{k-1} \wedge x_k)$$

for k even, and

$$g_k(x_1, \ldots, x_k) = (\neg x_1 \wedge x_2) \vee \ldots \vee (\neg x_{k-2} \wedge x_{k-1}) \vee \neg x_k$$

for k odd.

3. The **Hausdorff hierarchy** $g'_k[C]$, is $g_k[C]$ with the additional requirement that for all i, j such that $j > i$, $x_i = 1 \Rightarrow x_j = 1$. That is,

$$g'_k = g_k \bigwedge_{1 \leq j \leq k} (x_j \Rightarrow x_{j+1}).$$

Hausdorff ([Ha 78]) showed that $g'_k[D] = g_k[D]$ for all classes, D, that are closed under union and intersection.

4. The **Boolean hierarchy** was defined over NP by Cai and Hemachandra ([CaHe 86]). They denoted it as $h_k[NP]$, where

$$h_k(x_1, \ldots, x_k) = (\ldots((x_1 \wedge \neg x_2) \vee x_3) \ldots) \vee x_{k-1}) \wedge \neg x_k$$

for k even, and

$$h_k(x_1, \ldots, x_k) = (\ldots((x_1 \wedge \neg x_2) \vee x_3) \ldots) \wedge \neg x_{k-1}) \vee x_k$$

for k odd. •

In each case, these sequences of truthtables can be used to define hierarchies over general base classes in addition to NP. It is known, (see Theorem 5, this section) that the four collections of Boolean functions defined in 1 - 4 generate the same Boolean hierarchy over NP. In addition in Chapter 2, we will show that this equivalence is independent of the base class NP: it holds for most complexity classes.

1.2. The Boolean hierarchy over NP: example problems.

Example problems in various levels of the Boolean hierarchy over NP are known, and in fact, this hierarchy was initially motivated in [PaYa 82] as being useful for classifying certain versions of combinatorial optimization problems. Since then, many example problems have been found in all the levels of this hierarchy including complete problems for each level and "probably noncomplete" problems in each level, that is, problems that have been shown to be noncomplete in reasonable relativizations [BlGu 82], that is, relativizations in which statements that are strongly believed to be untrue do not become true and vice versa.

The former complete problems fall roughly into three categories: optimization problems ([PaYa 82], [PaWo 85] [Wa 86] [BuHa 88]), critical problems ([PaYa 82], [CaMe 86]), and "definitional" problems i.e., problems that are constructed directly from the various definitions of the hierarchy ([CaHe 86], [KöScWa 87], [BuHa 88]). The latter noncomplete problems are based on counting ([GuWe 86, 87]) and are generalizations of the unique solution problems introduced and studied in [PaYa 82] and [BlGu 82]. We give a short background and description for example problems in each of the categories named above.

NOTE: in the remainder of this section, "completeness" will mean $\leq_m^P$-completeness, and we will denote the k^{th} level of Boolean hierarchies defined by *all* of the truthtables of Definition 3 as $NP[k]$, and its complement as *co*-$NP[k]$. This uniform notation will be justified by Theorem 5.

Optimization problems.

In an early attempt to classify combinatorial optimization problems by complexity, it was demonstrated in [LeMo 81] that the "exact optimum" problems listed below are in $\Delta_2^P - (NP \cup coNP)$ unless $NP = coNP$. Subse-

quently, the class $NP[2]$ was brought into use in [PaYa 82] (and denoted as D^P) when these exact optimum problems and many other natural problems were shown to be not only in, but also complete for this class (see also [PaWo 85]). Below, we list some of these problems.

EXACT OPTIMUM TSP ROUTE.

Instance: An $m \times m$ distance matrix denoting the (integer) weights on the edges of a complete graph, and an integer, M.

Question: Is it true that the optimum *TSP* route has cost M?

Status: Complete for $NP[2]$.

Other exact optimization problems that are complete for $NP[2]$ include *EXACT MAXIMUM CLIQUE*, *EXACT MINIMUM VERTEX COVER* and *EXACT CHROMATIC NUMBER* which are defined in the obvious manner.

Another natural class of problems that was shown (in [PaYa 82] and [PaWo 85]) to be complete for $NP[2]$ is the class of "facets of optimization" problems. Roughly, a combinatorial optimization problem such as *finding* the *OPTIMUM TSP ROUTE* in a n-vertex weighted complete graph, or the *MAXIMUM CLIQUE* or *MINIMUM VERTEX COVER* of a given graph typically requires the minimization of a linear functional over a finite set of points (of cardinality at least exponential in n) in n-space. This turns out to be equivalent to minimization of the functional over the convex hull of the same set of points. Hence the complexity of the problem now rests on how hard it is to list, in a nonredundant fashion, the linear (in)equalities that represent the *facets* of the convex polyhedral feasible region. Thus, it was shown in [PaYa 82] that given an NP-optimization problem and an inequality, the question: "does the given inequality represent a facet of the

convex feasible region for the given optimization problem?" is complete for $NP[2]$. We give an example below in our standard format.

TSP FACETS.

Instance: A complete graph, G, with edge weights and an inequality, I.

Question: Is it true that I represents a facet of the convex feasible region for the *OPTIMUM TSP ROUTE* problem?

Status: Complete for $NP[2]$.

Other problems such as *CLIQUE FACETS* and *VERTEX COVER FACETS* can be defined in an analogous manner, and are also complete for $NP[2]$.

We now turn our attention to systematic generation of complete optimization problems for $NP[k]$, given an arbitrary k. This was achieved in [Wa 86] and [BuHa 88] by generalizing the definition of exact optimum problems as follows.

APPROXIMATE OPTIMUM ASSIGNMENT SAT_k.

Instance: A Boolean formula, F, in variables, $x_1, \ldots, x_n$, and a set, $S \subseteq N$, (resp. $S \subseteq \{$*finite intervals* $[a,b] \ : \ a,b \in N\}$) such that $|S| = k$.

Question: If the satisfying assignments for F are viewed as n-bit binary numbers (whose most significant bit is the assignment for x_1), does the maximum assignment of F belong to S (resp. belong to one of the intervals in S)?

Status: Complete for $NP[2k]$.

The *APPROXIMATE OPTIMUM*$_k$ problems for *VERTEX COVER*, *CLIQUE*, *INDEPENDENT SET*, *TSP*, etc are all defined in the obvious

manner and are all $NP[2k]$-complete. The following sequences of complete optimization problems (from [Wa 86] and [BuHa 88]) have a slightly different flavor.

APPROXIMATE OPTIMUM CLAUSE 3SAT$_k$.

Instance: A *3CNF* formula, F and a set, $S \subseteq N$, such that $|S| = k$.

Question: Is it true that the maximum over all assignments - of the number of satisfiable clauses of F - belongs to S?

Status: Complete for $NP[2k]$.

EVEN MAXIMUM ASSIGNMENT PREFIX$_k$.

Instance: A Boolean formula, F, in variables, $x_1, \ldots, x_n$.

Question: Is it true that the variable with the maximum index among $x_1, \ldots, x_k$ - that is assigned 1 by some satisfying assignment of F - has an even index?

Status: Complete for $NP[k]$.

Critical problems.

Detecting if a given graph is critical with respect to a certain property is a frequently occurring problem in graph theory: a critical graph is one that does not (resp. does) have a certain property, but deleting any node or edge creates a graph that does (resp. does not). To a large extent, a characterization of the class of critical graphs for a property yields a characterization of the property itself. The following examples were shown, in [PaYa 82], to be complete for the class D^P (which was later shown, in [WeWa 85] and [KöScWa 87], to be identical to various other definitions of $NP[2]$ or the second level of the Boolean hierarchy over NP).

MINIMAL HAMILTONIAN GRAPH.

Instance: A graph, G.

Question: Is it true that G is Hamiltonian but removing any single edge from G creates a graph that is not Hamiltonian?

Status: Complete for $NP[2]$.

MINIMAL HAMILTONIAN DIGRAPH.

Instance: A directed graph, G.

Question: Same as above.

Status: Same as above.

CRITICAL INTEGER PROGRAMMING.

Instance: A linear system of inequalities, $T =_{def} Ax \leq b$

Question: Is it true that T has no integer solution but omitting any single inequality from T creates a system that has?

Status: Complete for $NP[2]$.

Notice that the membership of the above problems in $NP[2]$ is rather transparent. Therefore, the major part of the work in [PaYa 82] goes into proving that the above problems are *complete* for $NP[2]$. The same paper stated the following problem as an example in $NP[2]$, but expressed the belief that proving its completeness would be difficult. A proof of completeness appeared later, in [CaMe 86].

MINIMAL 3-UNCOLORABILITY.

Instance: A graph, G.

Question: Is it true that G is not 3-colorable, but omitting any single vertex from G creates a graph that is?

Status: Complete for $NP[2]$.

In addition, it was proved in [CaMe 86] that the above problem is complete for $NP[2]$ even if it is restricted to planar graphs that consist of vertices with degree at most 5.

Definitional problems.

Following a straightforward result in [CGHHSWW 88], and results in [WeWa 85] and [KöScWa 87] about the equivalence of the various hierarchies of Definition 3 (see Section 1.3, Theorem 5), complete problems for each level of the Boolean hierarchy over NP can be systematically generated by just replacing the base sets for any of the hierarchies in Definition 3 by an NP -complete set, as in the next three examples from [CGHHSWW 88], and [KöScWa 87].

GENERIC COMPLETE PROBLEM$_k$.

Instance: A k-tuple, $(x_1, x_2, \ldots, x_k)$.

Question: Let X be an NP-complete set and t_k be any one of the truthtables in Definition 3; does $t_k(c_X(x_1), \ldots, c_X(x_k))$ evaluate to 1?

Status: Complete for $NP[k]$.

In particular, replacing t_k by the $Parity_k$ truthtable, and X by SAT in the above, yields the following complete problem.

$PARITY_k^{SAT}$.

Instance: A k-tuple of Boolean formulas, $(x_1, x_2, \ldots, x_k)$.

Question: Does $Parity_k(c_{SAT}(x_1), \ldots, c_{SAT}(x_k))$ evaluate to 1?

Status: Complete for $NP[k]$.

The following problem is a modification of the above problem by including a *truthtable specification with the input.* The status of the resulting problem follows directly, and is independent of the representation of the truthtable. We state the problem, however, for a full truthtable representation.

FULL TRUTHTABLE$_k^{SAT}$.

Instance: A $(k+1)$-tuple, $(x_t, x_1, x_2, \ldots, x_k)$, where x_t encodes a full truthtable, t, with k variables, and $x_1, \ldots, x_k$ are Boolean formulas.

Question: Does $t(c_{SAT}(x_1), \ldots, c_{SAT}(x_k))$ evaluate to 1?

Status: Hard for $NP[k]$; in $NP[k+1] \cap co\text{-}NP[k+1]$.

The next three examples, from [CaHe 86], and [BuHa 88] are not generic problems, but are more "natural." Their completeness follows in a straightforward fashion from the different definitions of the hierarchy and either from well-known reductions between NP-complete problems (as in *VERTEX COVER*$_k$, below) or the completeness of critical problems ([CaMe 86]) for $NP[2]$ (as in *ODD COLORABILITY*$_k$, below).

ODD COLORABILITY$_k$.

Instance: A graph, G.

Question: Let $k \in N$; does G have an odd chromatic number that lies between $3k$ and $4k$?

Status: Complete for $NP[k]$.

VERTEX COVER$_k$.

Instance: A tuple, (G, l), where G is a graph and l is an integer.

Question: Let $k \in N$, and let k be odd; does G have a minimum vertex cover of size $l + i + 1$ for some $0 \leq i \leq k$? (For k even: does G have a minimum vertex cover whose size is either $\leq l$ or equal to $l + i$ for some $0 \leq i \leq k$?

Status: Complete for $NP[k]$.

ODD SAT PREFIX$_k$.

Instance: A Boolean formula, F, in m variables, $x_1, \ldots, x_m$.

Question: Among the first k variables, $x_1, \ldots, x_k$, are there an odd number that are assigned 1 by some satisfying assignment of F?

Status: Complete for $NP[k]$.

Example problems for nonconstant query classes over *NP*.

The authors of [BuHa 88], [KöScWa 86], and [Wa 86] were quick to point out a by-product of finding sequences of complete problems going up the levels of the Boolean hierarchy over *NP*: the limits or *ω-jumps* of these sequences turn out to be complete problems for different nonconstant bounded query classes over *NP*, for example, $P^{NP} = \Delta_2^P$, $P^{NP}[\log\ n] = P_{tt}^{NP}$ (by Theorem 7, Section 1.3), and $P_{tt}^{NP}[\log\ n]$.

GENERIC COMPLETE PROBLEM$_\omega$.

Instance: A tuple, $(m, x_1, x_2, \ldots, x_m)$.

Question: Let X be an *NP*-complete set and t_m be any one of the truthtables in Definition 3; does $t_m(c_X(x_1), \ldots, c_X(x_m))$ evaluate to 1?

Status: Complete for P_{tt}^{NP}.

PARITY$_{\omega}^{SAT}$.

Instance: A tuple, $(m, x_1, x_2, \ldots, x_m)$, where $x_1, \ldots, x_m$ are Boolean formulas.

Question: Does $Parity_m(c_{SAT}(x_1), \ldots, c_{SAT}(x_m))$ evaluate to 1?

Status: Complete for P_{tt}^{NP}.

The following shows that the ω-jumps of two sequences of problems - both complete for the levels of the Boolean hierarchy over NP- are complete for different nonconstant query classes over NP.

FULL TRUTHTABLE$_{\omega}^{SAT}$.

Instance: A tuple, $(x_t, x_1, x_2, \ldots, x_m)$, where x_t encodes a full truthtable, t_m, with m variables, and $x_1, \ldots, x_m$ are Boolean formulas.

Question: Does $t_m(c_{SAT}(x_1), \ldots, c_{SAT}(x_m))$ evaluate to 1?

Status: Complete for $P_{tt}^{NP}[O(\log\ m)]$

Next we list the ω-jumps of a few optimization problems.

APPROXIMATE OPTIMUM ASSIGNMENT SAT$_{\omega}$.

Instance: A Boolean formula, F, in variables, $x_1, \ldots, x_m$, and a finite set, $S \subseteq N$ (resp. $S \subseteq$ {*finite intervals* $[a, b]$: $a, b \in N$}.)

Question: If the satisfying assignments for F are viewed as m-bit binary numbers whose most significant bit is the assignment for x_1, does the maximum assignment of F belong to S (resp. belong to one of the intervals in S)?

Status: Complete for P_{tt}^{NP}.

The *APPROXIMATE OPTIMUM*$_\omega$ problems for *VERTEX COVER CLIQUE, INDEPENDENT SET*, etc are all defined in the obvious manner and are all P_{tt}^{NP}-complete. In fact, the set, S, in these examples could be infinite in certain special cases. Examples follow.

ODD OPTIMUM ASSIGNMENT SAT$_\omega$.
(or (*0 MOD b*) *OPTIMUM ASSIGNMENT SAT*$_\omega$).

Instance: A Boolean formula, F, in variables, $x_1, \ldots, x_m$.

Question: If the satisfying assignments for F are viewed as m-bit binary numbers whose most significant bit is the assignment for x_1, is the maximum assignment of F odd (or 0 mod b; b fixed)?

Status: Complete for P_{tt}^{NP}.

The *ODD OPTIMUM*$_\omega$ problems for *VERTEX COVER, CLIQUE, INDEPENDENT SET*, etc. are all defined in the obvious manner and are all P_{tt}^{NP}-complete. In addition, the *APPROXIMATE OPTIMUM CLAUSE 3SAT*$_\omega$, *ODD OPTIMUM CLAUSE 3SAT*$_\omega$ and the *EVEN MAXIMUM ASSIGNMENT PREFIX*$_\omega$ problems are the ω-jumps of the corresponding sequences of optimization problems mentioned earlier, and are also complete for P_{tt}^{NP}.

It is interesting that *APPROXIMATE OPTIMUM TSP*$_\omega$ and *ODD OPTIMUM TSP*$_\omega$, however, are P^{NP}-complete, thus providing an instance where the ω-jumps of two different sequences of problems, both complete for the levels of the Boolean hierarchy over NP, can be complete for different nonconstant query classes over NP. The main reason for this is that to determine the cost of the optimum *TSP* route, one needs to perform binary search on the interval in which the cost may lie, and the length of this interval may not be polynomially bounded in the size of the instance. However for the

the problems of the previous paragraph, such a polynomial bound does exist on the length of the search interval.

It should be noted that many of the above problems have been listed in papers that are not specifically concerned with the levels of Boolean hierarchy over NP (e.g., [Ka 87], [Kr 86] [Pa 82]). As a result, their significance - as the limit points of underlying sequences of complete problems for the levels of the Boolean hierarchy over NP - is not explicitly mentioned in these papers.

The next two problems are additional examples from [Pa 82] and [Ka 86] and have a different flavor. These are "unique optimum" problems and although they resemble counting problems (described in the following subsection) they are fundamentally different in that they are complete for nonconstant bounded query classes over NP, while counting problems are generally believed not to be complete. The first of the problems below was mentioned in [Pa 82] and the second in [Ka 87].

UNIQUE OPTIMUM TSP.

Instance: A complete weighted graph, G.

Question: Is there a unique TSP route in G that is shorter than all other TSP routes?

Status: Complete for P^{NP}.

UNIQUE OPTIMUM CLAUSE SATISFIABILITY.

Instance: A CNF formula, F.

Question: Is it true that all assignments that satisfy the maximum set of clauses of F satisfy the same set of clauses?

Status: Complete for P_{tt}^{NP}.

The next problem appears in [Kr 86] as an example of an optimization problem in $P_{tt}^{NP}[\log(\log\ n)+O(1)]$, but the question of its completeness is open.

ODD OPTIMUM BIN PACKING.

Instance: A finite set, F, of items, an integral size for each item, and a positive integer bin capacity, b.

Question: Is it true that the minimum k - such that F can be divided into k disjoint sets each of which has total size at most b - is odd?

Status: In $P_{tt}^{NP}[\log(\log\ n)+O(1)]$.

Counting problems.

Variants of the problem of counting the number of satisfying assignments of a given formula have been studied in different contexts and are scattered over a wide range of complexity classes. In [PaYa 82] the following counting problem was first stated as an example in $NP[2]$.

UNIQUE SAT.

Instance: A Boolean formula, F.

Question: Does F have exactly one satisfying assignment?

Status: Hard for *coNP*; in $NP[2]$, relativizations exist in which the problem is not complete for $NP[2]$.

An interesting result followed in [BlGu 82] showing that there are reasonable relativizations in which *UNIQUE SAT* is not complete for $NP[2]$, and yet $NP = coNP$. In subsequent papers, [CaHe 86], and [GuWe 86,87], the *UNIQUE SAT* problem was generalized to yield counting classes with different "finite acceptance types." These classes are intertwined with the levels of

the Boolean hierarchy over NP and hence provide examples (given below) for all the levels. Furthermore, [GuWe 87] gives a generalization of the result in [BlGu 82] that provides relativized evidence that all of these counting problems are probably not complete for their corresponding levels of the Boolean hierarchy over NP.

SAT ASSIGNMENT COUNT$_k$.

Instance: A Boolean formula, F, and a finite (cofinite) set, $S \subseteq N$, such that $|S| = k$ ($|\overline{S}| = k$).

Question: Does the number of satisfying assignments of F belong to S?

Status: Hard for $coNP$; in $NP[k]$, relativizations exist where the problem is not complete for $NP[k]$.

Counting problems in nonconstant bounded query classes over NP.

Examples of counting problems in the nonconstant bounded query class P_{tt}^{NP} appear in [Ka 87]. These problems fall under the "unique optimum" class of problems some of which were listed earlier in this section as optimization problems that are complete for other nonconstant bounded query classes (e.g., *UNIQUE OPTIMUM TSP* - complete for P^{NP} and *UNIQUE OPTIMUM ClAUSE SATISFIABILITY* - complete for P_{tt}^{NP}). The unique optimum problems listed below, however, are believed to behave more like counting problems as apparent from an open question posed in [Ka 87] that asks whether relativized results analogous to [BlGu 82] can be obtained to give evidence that the following problems are not complete for P_{tt}^{NP}.

UNIQUE OPTIMUM ASSIGNMENT SAT.

Instance: A CNF Boolean formula, F.

Question: Is it true that there is a unique assignment of 0's and 1's to the variables of F that satisfies more clauses of F than any other assignment?

Status: Hard for *coNP*; in P_{tt}^{NP}.

Other problems such as *UNIQUE OPTIMUM CLIQUE, UNIQUE OPTIMUM VERTEX COVER* and *UNIQUE OPTIMUM COLORING*, defined in the obvious way, are also in P_{tt}^{NP} but have not been proven complete. It is probable ([Ka 87]) that relativized evidence exists for their noncompleteness.

To illustrate how different variants of the same unique optimum problem can behave quite differently, we give two examples from [Ka 87]. The first is complete for P_{tt}^{NP}, (reduction to *UNIQUE OPTIMUM CLAUSE SAT*) although it is a variant of the *UNIQUE OPTIMUM CLIQUE* problem. The second behaves like a counting problem (i.e., completeness proofs seem unlikely) but is a variant of *UNIQUE OPTIMUM TSP* which is complete for P^{NP}.

UNIQUE OPTIMUM GROUPED CLIQUE

Instance: A graph, G, partitioned into maximal independent sets.

Question: Is it true that all maximum cliques of G contain vertices from the same collection of independent sets from the given partition?

Status: Complete for P_{tt}^{NP}.

UNIQUE OPTIMUM BOUNDED TSP

Instance: A weighted graph, $G = (V, E)$, with edge weights bounded by a fixed polynomial in $|V|$.

Question: Is it true that there is a unique optimal *TSP* route in G?

Status: Hard for *coNP*; in P_{tt}^{NP}.

1.3. The Boolean hierarchy over *NP*: properties.

In this section we will give proofs for the known properties of the Boolean hierarchy over *NP*, and point out the hidden assumptions in these proofs that are specific to fixed truthtable reducibilities of constant norm, and to the base class *NP*. Furthermore, we will set up some of the notation required in these proofs in a unified manner in order to facilitate the exposition in Chapter 2.

Robustness: equivalence of the various definitions.

The next theorem from [KöScWa 87] and [WeWa 85] shows that the four Boolean hierarchies in Definition 3 are equivalent. More generally, this theorem characterizes the Boolean functions, t, for which the classes, $t[NP]$, are contained in the k^{th} level of the Boolean hierarchy over *NP*. The proof uses the concept of "mind changes" of Boolean functions first introduced in recursion theory by Putnam [Pu 65]. This concept was first used in a complexity theoretic setting by Beigel to relate adaptive and nonadaptive constant query classes ([Be 87a], Theorem 7, this section).

For the following theorem, we will be interested in various *orderings* of the variables of the truthtables. Orderings will be used to define "mind-changes" of truthtables, that in turn are used as a means to find mappings between truthtables. Such mappings will be necessary to establish uniqueness and equivalence of the classes based on these truthtables. We will expend some effort here in setting up the notation carefully in order to facilitate exposition in later sections. A *concrete ordering* of a truthtable with k variables is just a permutation of the set, $\{1, 2, \ldots, k\}$. We use the term "concrete" to differentiate between specific orderings and ordering *functions* that will play

a role in Chapter 2.

We leave the readers to supply their own definition, but we observe that any permutation of $\{1, 2, \ldots, k\}$ can always be realized with a complete table of (bit) length roughly $k \log\ k$.

For any ordering, O, of the variables, $x_1, x_2, \ldots, x_k$, and for any value, $1 \leq i \leq k$, we define $Ord_O(i)$ as the position of the variable, x_i, in O.

Definition 4:

Let t_k be any finite truthtable with k variables and associated ordering O of $\{1, 2, \ldots, k\}$. We denote the length k vectors or k-tuples of all zeroes and of all ones by $\vec{0}_k$ and $\vec{1}_k$, respectively, and often drop the subscript, k, when the context is clear. We denote the function that counts the number of ones in a Boolean vector, $\vec{b}$, by $\#_1(\vec{b})$. Suppose that $\vec{a} = \langle a_1, \ldots, a_k \rangle$ and $\vec{b} = \langle b_1, \ldots, b_k \rangle$ are Boolean vectors with all $a_j \leq b_j$. We say that t_k **changes its mind** from $\vec{a}$ to $\vec{b}$ if $t_k(\vec{a}) \neq t_k(\vec{b})$. In addition we let $\mu_{\mathbf{t_k},\mathbf{O}}(\vec{\mathbf{b}}, \mathbf{i})$ denote the number of mind changes of t_k that have been obtained just after i bits are flipped from 0 to 1 when transforming $\vec{0}$ to $\vec{b}$, where the transformation is accomplished one bit at a time, always flipping a "0" to a "1" in the order specified by the ordering, O. We frequently abbreviate $\mu_{t_k,O}(\vec{b}, \#_1(\vec{b}))$ to simply $\mu_{t_k,O}(\vec{b})$. By convention, for $j \geq \#_1(\vec{b})$, we define $\mu_{\mathbf{t_k},\mathbf{O}}(\vec{\mathbf{b}}, \mathbf{j})$ to be simply $\mu_{\mathbf{t_k},\mathbf{O}}(\vec{\mathbf{b}}, \#_1(\vec{\mathbf{b}}))$.

Perhaps the most important thing to notice about mind changes is that for *any* order, O,

$$t_k(b_1, \ldots, b_k) = t_k(\vec{0}) + \mu_{t,O}(\vec{b}) \mod 2.$$

and furthermore,

$$\forall \vec{b} \;\; \mu_{t_k,O}(\vec{b}) \leq \forall \vec{b} \;\; \mu_{t_k,O}(\vec{1}) \leq k.$$

Since in calculating $\mu_{t_k,O}(\vec{b})$, we always are interested in the pairs, $\langle O, \vec{b} \rangle$, it will be useful to combine such pairs as follows: A **partial concrete ordering** is a function, τ, that maps some number, j, of elements of $\{1, 2, \ldots, k\}$ to 0 and the remaining $k - j$ elements of $\{1, 2, \ldots, k\}$ in a one-one fashion to $\{1, 2, \ldots, k - j\}$. (A concrete ordering is then just the special case in which $j = 0$.) For any partial concrete ordering, τ, we denote by $\vec{\mathbf{a}}_\tau$ that Boolean vector that has its j^{th} bit, $\vec{a}_{\tau,j}$, equal to 0 if $\tau(j) = 0$ and $\vec{a}_{\tau,j}$ equal to 1 if $\tau(j) > 0$. We shall use the notation, $\mu_{t_k,\tau}(\vec{a}_\tau)$, in the obvious manner. That is, $\mu_{t_k,\tau}(\vec{a}_\tau)$ is the number of mind changes of t_k in going from $\vec{0}$ to $\vec{a}_\tau$ in the order determined by (the nonzero portion of) τ. •

It is not hard to see that there are at most $O(2^{k \log\ k})$ partial concrete orderings of $\{1, 2, \ldots, k\}$. For our purposes in the following proofs, these must be coded into strings or integers in some "nice" way, so that the encodings are "easily" recognized. Equally important, we require of these codings that all of the partial concrete orderings of $\{1, 2, \ldots, k\}$ be coded as strings of length roughly less than $2^{k \log\ k}$. By leaving details to the reader, we are now ready to state the theorem. Although we state the theorem for the specific case of the base class NP, in Chapter 2, we will examine the assumptions made in the proof that are specific to NP, show that they can be dispensed with and therefore demonstrate that the theorem holds independent of the base class.

Theorem 5:

1. ([WeWa 85]) Let s and t be arbitrary Boolean functions and let $s(\vec{0}) = t(\vec{0})$, and let τ denote any concrete ordering of the arguments of s and t. Then

$$max_\tau\{\mu_{s,\tau}(\vec{1})\} = max_\tau\{\mu_{t,\tau}(\vec{1})\} \Rightarrow s[NP] = t[NP],$$

and

$$max_\tau\{\mu_{s,\tau}(\vec{1})\} \leq max_\tau\{\mu_{t,\tau}(\vec{1})\} \Rightarrow s[NP] \subseteq t[NP].$$

2. ([KöScWa 87]) Recall the Boolean functions of Definition 3. For all k,

$$Diff_k[NP] = g_k[NP] = g'_k[NP] = h_k[NP].$$

Proof:

We demonstrate a mapping between the truthtables, s and t, under the conditions given on their maximum number of mind changes over all orderings, τ. The proof consists of two parts denoted ♣. The first part shows that there is a special sequence of fixed truthtable reductions, $\{\lambda m\ special_m\}$, such that for any truthtable, s,

$$m \geq max_\tau\{\mu_{s,\tau}(\vec{1})\} \Rightarrow s[NP] \subseteq special_m[NP].$$

The second part shows that for any truthtable, t,

$$m \leq max_\tau\{\mu_{t,\tau}(\vec{1})\} \Rightarrow special_m[NP] \subseteq t[NP].$$

The latter implication in Part 1 of the above theorem follows directly, and the former implication follows simply by conducting the above argument in the reverse direction, from t to s. Part 2 follows since for each of the

truthtables, t_k of Definition 3, $t_k(\vec{0}) = 0$ and $max_\tau\{\mu_{t_k,\tau}(\vec{1})\} = k$ (for illustration, consider the $parity_k$ truthtable: a standard left to right ordering of the variables gives k mind changes from $\vec{0}$ to to $\vec{1}$.

• Let s be any k-ary Boolean function and let X be a set in $s[NP]$. By Definition 2, the set, X, can be defined as:

$$c_X(x) = s(c_{Y'}(f(x,1)),\ldots,c_{Y'}(f(x,k))), \qquad (\sharp)$$

where f is a function computable in polynomial-time and $Y' \in NP$. We will prove that there is a set, Y, such that

1.

$$c_X(x) = \sum_{i=1}^{max_\tau\{\mu_{s,\tau}(\vec{1})\}} [c_Y(i,x)] + s(\vec{0}) \mod 2, \qquad (\sharp\sharp)$$

2.

$$\forall x \forall i\, [c_Y(i+1,x) \leq c_Y(i,x)], \ and$$

3. $Y \in NP$.

All of these requirements will define the special truthtable reduction, $special_m$, that satisfies:

$$m \geq max_\tau\{\mu_{s,\tau}(\vec{1})\} \Rightarrow s[NP] \subseteq special_m[NP].$$

It is clear that to establish Requirement 1, it is adequate to define the set, Y, in such way that for every choice of x there is some concrete ordering, τ', of the arguments, $\{1,2,\ldots,k\}$, of s such that

$$\sum_{i=1}^{max_\tau\{\mu_{s,\tau}(\vec{1})\}} [c_Y(i,x)] = \mu_{s,\tau'}(c_{Y'}(f(x,1)),\ldots,c_{Y'}(f(x,k))).$$

We will define Y to be

$$Y =_{def} \{ (i,x) : \textit{for some concrete ordering, } \tau', \textit{ of } \{1,2,\ldots,k\} \textit{ there exist at least } i \textit{ mind changes of } s \textit{ between } \vec{0} \textit{ and } (c_{Y'}(f(x,1)),\ldots,c_{Y'}(f(x,k)))\}.$$

Notice that Requirement 2, namely the monotonicity of the function, c_Y, is satisfied, and since $c_Y(i,x)$ is always 0 when $i \geq max_\tau\{\mu_{s,\tau}(\vec{1})\}$, the sum in Requirement 1 is justified by the above definition of Y. As an aside, we could in fact have replaced the preceding inequality by

$$i \geq max_\tau\{max_{\vec{b}}\{\mu_{s,\tau}(\vec{b})\}\},$$

to give a more succinct summation, and meet a stronger Requirement 1, since

$$\forall\tau\ [max_\tau\{max_{\vec{b}}\{\mu_{s,\tau}(\vec{b})\}\} \leq max_\tau\{\mu_{s,\tau}(\vec{1})\}.$$

It remains to locate Y in NP. We give the following NP algorithm for Y.

Input: (i,x)

Guess: a concrete *partial* ordering, τ, of $\{1,\ldots,k\}$.

Check:

i. does there exist at least i mind changes of s between $\vec{0}$ and $\vec{a}_\tau$? and

ii. does τ satisfy:

$$\forall j\ [\vec{a}_{\tau,j}= 1 \ \Rightarrow\ c_{Y'}(f(x,j)) = 1]?$$

In other words, for all j such that $\vec{a}_{\tau,j}= 1$, does Y' accept $f(x,j)$?

(###)

The first check takes at most k time steps, and the second check is a positive membership question to Y' and simply requires the running of the NP algorithm for Y'.

• We will now show that for all sets, X, if there is a set, $Y \in NP$, such that the $special_m$-reduction:

$$c_X(x) = \sum_{i=1}^{m} [c_Y(i,x)] + initval \mod 2,$$

where m and $initval \in \{0,1\}$ are constants; and the set, Y, satisfies

$$\forall x \forall i \, [c_Y(i+1,x) \leq c_Y(i,x)],$$

then for any truthtable, t,

$$[m + initval \leq max_\tau\{\mu_{t,\tau}(\vec{1})\} + t(\vec{0})] \Rightarrow X \in t[NP].$$

Assume that t is a l-ary Boolean function, and let O be the (total) ordering (that value of τ) that maximizes $\mu_{t,\tau}(\vec{1})$. We will obtain the above result by performing a simple substitution of the values of c_Y into the arguments of t.

The truthtable t, with its variables ordered by the ordering O has enough mind changes to do a mod 2 count of the values (i,x) for which the value of $c_Y(i,x)$ is "1," since $m \leq max_\tau\{\mu_{t,\tau}(\vec{1})\}$. Hence, if we substitute

$$c_Y(1,x), \ldots, c_Y(1,x), c_Y(2,x), \ldots, c_Y(2,x), \ldots, c_Y(m,x), \ldots, c_Y(m,x)$$

into the variables for t in the order dictated by O, beginning the substitution of the next new variable at the position where the next mind change

of t occurs, then t, with these substitutions, must give a mod 2 count of $\sum_{i=1}^{m} [c_Y(i,x)]$. Since for any vector $\vec{b}$

$$t(\vec{b}) = \mu_{t,O}(\vec{b}, l) + t(\vec{0}) \mod 2,$$

doing the proper substitutions gives us the required result, provided the initial values, *initval* and $t(\vec{0})$, are identical, which is assumed in the statement of the theorem, since in the course of finding a mapping from s to t, we simply substitute $s(\vec{0})$ for *initval.* If the two variables, *initval* and $t(\vec{0})$ are not identical, then we must do enough substitutions to effect one more mind change. ∎

Analysis of specific assumptions.

• Equation (♯) uses Definition 2 for the class, $s[NP]$, that is generated by the truthtable, s. On the surface, the existence of the set, $Y' \in NP$ and the function, f, use the fact that NP has complete sets.

• The NP algorithm (♯♯♯) for the set, Y, explicitly uses nondeterminism.

• Our entire exposition of the above proof is done under the assumption that the truthtables, s and t, are *fixed* truthtables, and therefore are necessarily of constant norm.

•

The above theorem shows that the k^{th} level of the Boolean hierarchy over NP is a unique, robust and well-defined class which we will denote as **NP[k]**. The complement of $NP[k]$ will be denoted, in the usual manner, as **co-NP[k]**.

An examination of the above proof gives the following normal form that characterizes the class $NP[k]$.

Theorem 6: ([WeWa 85])

For each k,

$$NP[k] = \{X \mid \exists Y \in NP\ [\forall x \forall i\ [c_Y(i+1,x) \leq c_Y(i,x)]\ and$$
$$c_X(x) = \Sigma_{i=1}^{k}\ c_Y(i,x)\ \bmod 2]\}.$$

Relationship to other constant query classes over NP.

Recall from the definition of the Boolean hierarchy, and from the fact that NP is closed under $\leq_m^P$-reductions and certain simple operations like finite unions, that sets in its k^{th} level are reducible to a set in NP by a fixed truthtable reduction of constant norm (see Definition 2). The following theorem of Beigel ([Be 87a]) shows that the converse is also true. That is, any set that is reducible to a set in NP by a truthtable reduction of norm k, (here the truthtables are not fixed - they depend on the input), lies in $NP[k+1] \cap co\text{-}NP[k+1]$. In other words the constant truthtable hierarchy over NP, $P_{tt}^{NP}[k]$ for $k \in N$, interleaves the Boolean hierarchy over NP. In addition, the following theorem exhibits a relationship between constant adaptive and nonadaptive query hierarchies over NP.

Before we state the theorem we first note that when dealing with polynomial-time or log-space truthtable reductions to NP that are of constant norm, the actual representation of the truthtable is immaterial: the representation might be a Boolean formula, or a full truthtable.

Theorem 7: ([Be 87a])
For each k,

1. $NP[k] \cup co\text{-}NP[k] \subseteq P_{tt}^{NP}[k] \subseteq NP[k+1] \cap co\text{-}NP[k+1]$.
2. $P_{tt}^{NP}[2^k - 1] = P^{NP}[k]$.

Proof:

1. The first containment in Part 1 is straightforward from Definition 2: sets in $NP[k] \cup co\text{-}NP[k]$ are $\leq^P_{k-tt}$-reducible to sets in NP (using fixed truthtables). The second containment is established as follows: replace the fixed truthtable, s, in the second implication of Theorem 5(1) by a truthtable, s^x_k, that is of constant norm, but is generated in polynomial-time from the input in the course of a $\leq^P_{k-tt}$-reduction. The existence of the set, Y, exactly as in the proof of Theorem 5 (Equation $\sharp\sharp$) follows in a straightforward manner. Furthermore, replace the truthtable, t, in Theorem 5(1) by the fixed truthtables that generate $NP[k]$ (or $co\text{-}NP[k]$.) Then, the second part of the proof of Theorem 5, namely the substitution of the values of c_Y into the arguments of the fixed truthtables that generate $NP[k]$ (resp. $co\text{-}NP[k]$) also follows, *provided* the initial value $s^x_k(\vec{0})$ is 0 (resp. 1) for all x. Since, in general, neither of these requirements are met by all $\leq^P_{k-tt}$-reductions, we need the additional query in the right hand side of the containment, $P^{NP}_{tt}[k] \subseteq NP[k+1] \cap co\text{-}NP[k+1]$.

The containment, $P^C[k] \subseteq P^C_{tt}[2^k - 1]$, is obtained in the straightforward manner by gathering all possible adaptive sequences of queries that correspond to all possible answers from the oracle into a set of $2^k - 1$ non-adaptive queries, an operation that is viable within polynomial time since k is a constant.

The reverse containment is obtained as follows. Consider any set, X, that is reducible to a set, $Y' \in NP$, in terms of the set, Y, as in Equation ($\sharp\sharp$) in the proof of Theorem 5. Since the function, c_Y, is monotone in i, the sum, $\sum_{i=1}^{2^k-1} c_Y(i,x)$, can be computed merely by finding that j such that

$c_Y(j,x) = 1$, but $c_Y(j+1,x) = 0$. However, this "turning point," j, can be located by performing a binary search on the interval, $[1, 2^k - 1]$, and making at most k adaptive queries to Y. The result then follows. ∎

Analysis of specific assumptions.

• The assumptions that are specific to the base class, NP, are identical to those in the proof of Theorem 5.

• While we dealt only with fixed truthtables in the proof of Theorem 5, here we deal with (constant norm) truthtables that vary with the input. However, by applying Definition 2, we could restrict ourselves to deal only with (constant norm) truthtable reductions to a single set, thus allowing an easy transition from fixed to varying truthtables.

• We assume that the conversion of adaptive queries to (a larger number of) nonadaptive queries is a polynomial-time process and hence we need the fact that the reductions that we are dealing with are $\leq^P_{k-tt}$-reductions.

•

The next theorem was proved independently in [BuHa 88] and [Wa 87a] and shows that when the base class is NP, general polynomial-time nonadaptive reductions of constant norm, or $\leq^P_{k-||}$-reductions do not differ from the corresponding truthtable reductions, or $\leq^P_{k-tt}$-reductions. Furthermore, log-space and polynomial-time reductions of constant norm over NP are equivalent.

Theorem 8:

For each k,

1. $P^{NP}_{tt}[k] = P^{NP}_{||}[k] = L^{NP}_{||}[k] = L^{NP}_{tt}[k]$, and

2. $L^{NP}[k] = L^{NP}_{tt}[2^k - 1]$.

Consequences of collapse.

The following theorems in [Ka 88] and [ChKa 90] give evidence that the Boolean hierarchy over NP is proper. They show that if the Boolean hierarchy over NP collapses to any finite level, then the polynomial-time hierarchy collapses to the level: $P^{NP^{NP[log\ n]}}$. The technique used in the first proof is to show that if the Boolean hierarchy over NP collapses, then SAT is reducible to a set in $coNP$ by a $\leq_m^P$-reduction that queries a sparse oracle in the second level of the polynomial-time hierarchy. The collapse of PH follows from of an inductive counting technique similar to the ones used, for instance, in [Ma 82], [Lo 85] and [Ka 87]. Later, the above result was strengthened in [ChKa 90] leading to a corollary that the collapse of the Boolean hierarchy over NP to its k^{th} level implies the collapse of the Boolean hierarchy over Σ_2^P (defined analogously to the hierarchy over NP) to *its* k^{th} level. The various standard definitions of the latter hierarchy are also equivalent, and just as in the case of NP and its k^{th} level and complement are correspondingly denoted $\mathbf{\Sigma_2^P[k]}$ and $\mathbf{co\text{-}\Sigma_2^P[k]}$ respectively.

Theorem 9:

For each k

1. ([Ka 88]) $NP[k] = co\text{-}NP[k] \Rightarrow PH \subseteq P^{NP^{NP[log\ n]}}$.

2. ([ChKa 90]) $NP[k] = co\text{-}NP[k] \Rightarrow PH \subseteq P^{NP^{NP[log\ k+1]}}$, and $NP[k] = co\text{-}NP[k] \Rightarrow \Sigma_2^P[k] = co\text{-}\Sigma_2^P[k]$.

Miscellaneous properties.

The following is a brief sketch of various other properties of the Boolean hierarchy over NP (mainly relative o some oracle) that were proved in [CaHe 86], and appear in journal form in [CGHSWW 88] and [CGHSWW 89].

Theorem 10: ([CGHHSWW 88,89])

1. There exist some relativizations in which the Boolean hierarchy over NP is proper, and others where it collapses to any chosen level that, in addition, coincides with $PSPACE$. In the former relativization, the Boolean hierarchy over NP has no $\leq_m^P$ complete sets, since such a set would be in some finite level of the hierarchy, resulting in a collapse. Notice, however, that any NP-complete set is $\leq_{btt}^P$-complete for the Boolean hierarchy over NP.

2. For a set X and class C we say that X is **C-immune** (**C-bi-immune**) if X (both X and $\overline{X}$) contain(s) no infinite subsets that are in C. Notice that no set in the Boolean hierarchy over NP is $NP[2]$-immune, since any set in $NP[k]$ is a finite union of $NP[2]$ sets from Definition 3(2), or NP- or $coNP$-bi-immune. However, a structural asymmetry between the levels of the Boolean hierarchy over NP and their complements is apparent from the following: there exist relativizations in which $NP[2]$ has $co\text{-}NP[2]$-immune sets. There also exist relativizations in which Δ_2^P contains sets that are immune to the Boolean hierarchy over NP.

3. No proof that relativizes can much improve the result in [KaLi 80] that if NP has sparse $\leq_T^P$-hard sets then $PH = \Sigma_2^P$. More precisely, no proof that relativizes can show that if NP has sparse $\leq_T^P$-hard sets then $PH = \bigcup_{k \in N} NP[k]$.

4. The result in [HaImSe 83] that

$E =_{def} DTIME[2^{O(n)}] = NE =_{def} NTIME[2^{O(n)}] \Longleftrightarrow$

$NP - P$ has no sparse sets.

can be extended to show that

$NP - P$ has no sparse sets $\Longleftrightarrow$

$\forall k\ NP[2k+1] - NP[2k]$ *has no sparse sets* $\Longleftrightarrow$

$\forall k\ NP[k] - P$ *has no tally sets.*

The first equivalence above shows a disparity between odd and even levels of the Boolean hierarchy over NP that is further supported by the following: there are relativizations in which no odd levels of this hierarchy have sparse sets, but all even levels do.

5. While the result in [HaImSe 83] does not differentiate between sparse and tally sets (the existence of sparse and tally sets in $NP - P$ are equivalent), the second equivalence above does: tally sets can be removed from the Boolean hierarchy over NP while still leaving other sparse sets in its even levels. Furthermore, this gap can be narrowed: it is possible to show that *capturable* sets - sets that are subsets of sparse NP sets - behave much like tally sets although they form a richer class.

$NP - P$ *has no sparse sets* $\Rightarrow \forall k\ NP[k] - P$ *has no capturable sets.*

1.4. Previous generalizations of the Boolean hierarchy over NP.

Recall that one of the the goals of this thesis is to investigate the properties of extended Boolean hierarchies - Boolean hierarchies generated by truthtables whose norms are slowly growing functions of the size of the input - over general base classes. In this section, we will will first review those properties of the constant levels of the Boolean hierarchy over NP that have been previously generalized to the extended Boolean hierarchy over NP. Following that, we will review results in [Ch 89] on Boolean hierarchies over general base sets.

The extended Boolean hierarchy over NP.

The first problem that arises in studying extended Boolean hierarchies is to find a consistent and robust definition that is also intuitive. Wagner ([Wa 87a]) uses the following generalization of the normal form of Theorem 6 to define the extended Boolean hierarchy over NP.

Definition 11:

Let $r(|x|)$ be polynomially bounded in $|x|$.

$$\mathbf{NP[r]} =_{def} \{X \mid \exists Y \in NP\ [\forall x \forall i\ [c_Y(i+1,x) \leq c_Y(i,x)]\ and$$
$$c_X(x) = \Sigma_{i=1}^{r(|x|)}\ c_Y(i,x) \mod 2]\ \}.$$

We denote the complement of $NP[r]$ as **co-NP[r].** ●

Before we critique the above "normal form" definition and suggest alternatives, we first list some results that support this definition.

The next theorem states that the various natural hierarchies over NP that are generated - by allowing the Boolean functions in Definition 3 to have nonconstant norms - are not only equivalent to each other, but also coincide with Wagner's extended Boolean hierarchy over NP, thus giving evidence that his definition is robust. The statement of the theorem is a minor variant of one proved by Wagner in [Wa 87a].

Theorem 12:

Let $r(|x|)$ be polynomial-time computable from x and polynomially bounded in $|x|$, and let

$$\mathbf{Diff_r[NP]} =_{def} \{X \mid \exists Y \in NP \textit{ and a function } f \textit{ computable in}$$
$$\textit{polynomial-time such that } f(x) = (x_1, \ldots, x_{r(|x|)})$$
$$\textit{and } c_X(x) = \Sigma_{i=1}^{r(|x|)} c_Y(x_i) \mod 2\}.$$

Define $\mathbf{g_r}$[NP], $\mathbf{g'_r}$[NP], and $\mathbf{h_r}$[NP] (suggested by Cai and Hemachandra in [CaHe 86]) in an analogous manner based on the functions in Definition 3. Then

$$Diff_r[NP] = g_r[NP] = g'_r[NP] = h_r[NP] = NP[r].$$

The proof is a generalization of that of Theorem 5.

Analysis of specific assumptions.

• The generalization from truthtables of constant norm to truthtables of nonconstant, possibly polynomial norms brings to light that our proof for Theorem 5 makes assumptions about NP that are not really necessary. For instance, the NP algorithm ($\sharp\sharp\sharp$) for the set, Y, in the proof of Theorem 5 can be abandoned in favor of a careful construction of the set, Y, through finite (approximately 2^k : recall that k is the norm of the truthtable s) unions and intersections of sets in NP that are $\leq_m^P$-reducible to the set, Y'. However, in generalizing the proof to the above theorem, we allow the truthtable, s, to have polynomial norm, and hence a full use of nondeterminism is required to show that the set, Y, is in NP.

• By being similarly careful, Definition 2 for the constant levels of the Boolean hierarchy over NP can be obtained from Definition 1 without using the fact that NP has $\leq_m^P$-complete sets, but merely by using the fact that NP is closed under reductions simpler than $\leq_m^P$-reductions, such as $\leq_m^L$-reductions, and under finite unions and intersections. However, the use of the single set, Y, as opposed to a sequence of sets in the definition of $Diff_r[NP]$ in the above theorem can be justified only by the fact that NP has a recursive enumeration of machines and has complete sets.

•

The next theorem shows that the interleaving of the constant levels of the Boolean hierarchy and the constant query hierarchies over NP extends also to nonconstant levels.

Theorem 13:

1. $NP[r] \cup co\text{-}NP[r] \subseteq P_{tt}^{NP}[r] \subseteq NP[r+1] \cap co\text{-}NP[r+1]$, for $r(|x|)$ computable in polynomial-time from x, and polynomially bounded in $|x|$.

2. $P_{tt}^{NP}[2^r - 1] = P^{NP}[r]$, for r computable in polynomial-time, and logarithmically bounded.

3. $P_{tt}^{NP} = \bigcup_{k \in N} NP[n^k]$.

The proofs of Parts 1 and 2 are generalizations of the proof of Theorem 6, and utilize Theorem 12 instead of Theorem 5. Part 3 follows directly from Part 1.

The assumptions made in this proof are identical to those made in the proof of Theorem 12.

Wagner extended Kadin's collapse theorem for certain "well-behaved" nonconstant levels of his Boolean hierarchy over NP as follows. ([Wa 87b]).

Theorem 14: ([Wa 87b])
Let $r(|x|)$ be a sublinear function such that $Min_m[r(mn) = m]$ is defined everywhere and computable in polynomial-time (many natural functions r satisfy this property, e.g, $n^{1/k}$, $\log(\log\ n)$). Then

$$NP[r] = co\text{-}NP[r] \Rightarrow \textit{PH collapses to its third level.}$$

We now discuss some of the problems with using Definition 11 for the extended Boolean hierarchies over general complexity classes.

First, the justification for Wagner's normal form rests on several hidden assumptions mentioned after Theorem 12 (and Theorem 13) that may be made in the case of NP, but not in the case of general base classes. It would be preferable if the definition were more philosophically straightforward and made intuitive sense independent of whether or not Theorems 12 and 13 hold for a general base class. Moreover, once such a definition is formalized, it will be of interest to characterize base classes for which the results mentioned above do hold.

Secondly, Definition 11 is counterintuitive even when the base class is NP, for the following reason. The classes $NP[r]$ as in this definition are not closed under $\leq_m^P$-reductions, making them awkward to manipulate. For instance, $\leq_m^P$-complete sets for $NP[g]$ may lie in $NP[f]$ even when the function, f, grows much slower than the function, g, and furthermore, sets in all levels of this extended hierarchy are $\leq_m^P$-reducible to sets in sublinear levels. The reduction just pads the input by a polynomial factor. We discuss this issue in greater detail with te conclusions at the end of Chapter 2.

Cai and Hemachandra's suggestion for defining the r^{th} level of the extended Boolean hierarchy over NP as $h_r[NP]$ (see Theorem 12) partially surmounts at least the first of the above problems. By using their method, it is straightforward to define extended Boolean hierarchies over base classes that have recursive enumerations and complete sets. For general base classes, however, their definition faces both of the above problems.

Other truthtable reducibilities and base sets.

Recall that our goal not only includes a study of extended Boolean hierarchies over general base classes, but also the relationship between Boolean hierarchies and bounded query classes defined over the same base class using

a fairly general type of reduction. Here, we review earlier work in [Wa 87a] and [BuHa 88] on bounded query classes over NP that use log-space instead of a polynomial-time reductions.

The next theorem is a set of results proved independently in [Wag 87a] and [BuHa 88] about the relationship between bounded (but nonconstant) query classes defined using different kinds of nonadaptive reductions, between reductions that use different kinds of truthtable representations and between log-space versus polynomial-time reductions.

Theorem 15:
For $r(|x|)$ polynomial-time computable from x and polynomially bounded in $|x|$,

1. $P_{||}^{NP}[r] = P_{bf}^{NP}[r] = P_{tt}^{NP}[r]$ and

2. $P_{ftt}^{NP} = P_{tt}^{NP}[O(\log\ n)]$ and

For r log-space computable and polynomially bounded,

3. $L_{||}^{NP}[r] = L_{bf}^{NP}[r] = L_{tt}^{NP}[r] = P_{tt}^{NP}[r]$.

For r log-space computable and logarithmically bounded,

4. $P^{NP}[r] = L^{NP}[r] = L_{tt}^{NP}[2^r - 1]$.

5. The class $Diff_{poly}[NP]$ remains unchanged if the function f in its definition (see Theorem 13) is required to be log-space computable.

6. $Diff_{poly}[NP] = \bigcup_{k \in N} NP[n^k] = P_{tt}^{NP} = L_{tt}^{NP} = P^{NP}[O(\log\ n)] = L^{NP}[O(\log\ n)]$.

7. $P_{ftt}^{NP} = L_{ftt}^{NP} = P_{tt}^{NP}[O(\log\ n)] = L_{tt}^{NP}[O(\log\ n)]$.

Notice that the earlier work on Boolean hierarchies that has been reviewed so far deals exclusively with the base class NP. In the latter part of this

subsection, we review a result of Chang ([Ch 89]) that can be viewed as a first step towards investigating Boolean hierarchies over general base classes. He studies Boolean hierarchies over individual sets in NP, with the specific view of generalizing Kadin's collapse result ([Ka 87]; see Theorem 9, this section). His results establish properties of Boolean hierarchies over those subclasses of NP that contain P but do not behave as well as NP, for instance the levels of the low and high hierarchies of Schöning ([Sc 82,83]). We first define the Boolean hierarchy over individual sets in NP.

Definition 16:
Let $X \in NP$. The following sequence of sets represents the Boolean hierarchy of sets over X. For each k,

$$X[2k] =_{def} \{\langle x_1, \ldots, x_{2k}\rangle \mid \langle x_1, \ldots, x_{2k-1}\rangle \in X[2k-1] \ \wedge \ x_{2k} \in \overline{X}\},$$

and

$$X[2k-1] =_{def} \{\langle x_1, \ldots, x_{2k-1}\rangle | \langle x_1, \ldots, x_{2k-2}\rangle \in X[2k-2] \vee \ x_{2k-1} \in X\}.$$

The complements of the above sets are denoted **co-X[k]**. •

Before we state the theorem, we define the low and high hierarchies of Schöning ([Sc 82,83]). Let C^X be the class of sets that are recognized by C-machines with oracle X.

$$\mathbf{high}_k = \{X \mid X \in NP \ \wedge \ \Sigma_k^{P,X} = \Sigma_k^{P,SAT}\}.$$

$$\mathbf{low}_k = \{X \mid X \in NP \ \wedge \ \Sigma_k^{P,X} = \Sigma_k^{P}\}.$$

$$\widehat{\mathbf{low}}_k = \{X \mid X \in NP \ \wedge \ \Delta_k^{P,X} = \Delta_k^{P}\}.$$

Clearly,

$$low_0 \subseteq \widehat{low_1} \subseteq low_1 \subseteq \widehat{low_2} \subseteq low_2 \subseteq \ldots, and$$

$$high_0 \subseteq high_1 \subseteq high_2 \subseteq \ldots.$$

In addition, Schöning shows that all NP-complete sets are high, and

$$low_k = high_k \Rightarrow PH = \Sigma_k^P.$$

It follows from these results that the first statement in the following theorem is a straightforward consequence of the second.

Theorem 17: ([Ch 89])

$$\exists k\ SAT[k] \leq_m^P co\text{-}SAT[k] \Rightarrow PH\ is\ finite.$$

More generally, let $X, Y \in NP$; then

$$X[k] \leq_m^P co\text{-}Y[k] \Rightarrow X \in \widehat{low_3}.$$

1.5. A table of results.

The row and column headings in the following table are self-explanatory. The entries in the table refer to theorems in this thesis: the entries with asterisks refer to generalizations in Chapter 2, and the remaining entries refer to earlier results reviewed in Chapter 1. To explain double entries: while Theorem 12 shows robustness of Wagner's normal form definition of the extended Boolean hierarchy over NP Theorem 34 does the same for a more intuitive definition of the extended Boolean hierarchy over general base classes, and shows the equivalence of this definition to Wagner's definition for the case of NP; and while Theorem 15 considers the special case of log-space nonconstant query reductions to NP, (instead of the usual polynomial-time reductions as in Theorem 13) Theorem 35 considers general nonconstant query reductions to a general base class. Finally, note that Theorem 17 only considers consequences of the collapse of Boolean hierarchies over *base sets* in NP to constant levels, and therefore covers only a small portion of its place in the table.

	BH over *NP*; constant norm, poly-time or log-space reductions.	*EBH* over *NP*; general norm, poly-time or log-space reductions.	*EBH* over a general base class; general norm, general reductions.
Examples, Complete problems.	Section 1.2	Section 1.2	* Section 2.3
Robustness, uniqueness.	5	12 * 34	* 34
Relationship to $\leq_T$ and $\leq_{tt}$.	7	13 15 * 35	15 * 35
Relationship to $\leq_{bf}$ and $\leq_{ftt}$.	8	13 15 * 35	15 * Page 59
Consequence of collapse.	9	14	17
Miscellaneous properties.	10		

1.6. Other related work on bounded query classes.

We list various directions of study all of which deal with bounded query classes and reducibilities, but will not be considered further in this thesis.

- The counting hierarchy over NP was introduced and studied in [GuWe 87] for the purpose of finely classifying the counting problems that are generalizations of the *UNIQUE SAT* problem introduced in [PaYa 82] (see Section 1.2., this chapter). This hierarchy interleaves the Boolean hierarchy over NP and counting problems can be found in all levels of the latter hierarchy. However, there is evidence ([BlGu 82], [GuWe 87]) based on relativizations that counting problems are not complete for the levels of the Boolean hierarchy over NP, and intuitively clear that the Boolean hierarchy is too coarse to capture the essence of counting problems.

- Bounded query *function* classes over NP were popularized in [Kr 86] in the context of classifying optimization problems. These classes are designed for evaluation problems just as the bounded query classes reviewed in this chapter are designed for decision problems. These classes were further studied in [Ga 86], [GaPe 88] and [Be 87a,88c]. While the two hierarchies are defined in a similar manner, they behave quite differently in many aspects.

- P-terse and P-cheatable sets were defined in [AmGa 87] and studied in [Be 87c,88a,88c], [AmBeGa 88], [GaHeHo 90], and [GoJoYo 88a,b]. A set X is **P-terse** (resp. **P-superterse**) if, for all k, it is not possible to answer k polynomial-time computable nonadaptive queries to X by making only $k-1$ polynomial-time computable adaptive queries to X (resp. any set Y). A set X is **P-cheatable** if there is a k, a set Y and a polynomial-time computation that determines membership of 2^k strings

in X by making only k adaptive queries to Y. The papers mentioned above study the relationship of these concepts to traditional concepts in Complexity Theory, such as P-selectiveness and P-closeness, and [Be 88c] considers the consequences of the assumption that NP-hard sets are not P-terse.

- As mentioned earlier, Boolean hierarchies were first introduced by Eršov ([Er 68,69]) in recursion theory, where bounded query classes were already well-studied (see [Ro 67]). More recent work on Boolean hierarchies, bounded query classes, terseness, cheatability, and other related notions in a recursion theoretic context can be found in [Hay 78], [Ep 79], [EpHaKr 81], [BeGaHa 87], [BeGaOw 87], [BGGO 87], [Be 88b], [BeGa 87,88a,b], and [Ow].

- The relationship between bounded query classes over NP, NP computations with sparse NP oracles, and NP computations with restricted access to oracles were studied in [BoLoSe 84,85] and [Lo 85], and later in [ScWa 87]. Similar studies on polynomial space bounded computations can be found in [Bo 81], [BoWr 81], [BoLoSe 84], [Lo 85], and [BaBoSc 85]. Some form of the inductive counting techniques that appeared originally in [Mah 82], and more recently, for instance, in [Ka 87] figure prominently in most of the former papers. These techniques were unified in [ScWa 87] and, as an aside, similar techniques were used in several hierarchy collapse results of [LaJeKi 87], [He 87] and [To 87] that preceded (and were superceded by) the result proved independently in [Im 88] and [Sz 87] that nondeterministic space is closed under complement.

- General properties of constant truthtable reductions to sparse sets was

the subject of study in [BoKo 87] and [Ko 88]. More specific investigations in [Ye 83], [Yap 83], [Wat 88] dealt with the consequence of the assumption that all sets in NP are reducible to a sparse set by constant truthtable reductions. The results in these latter papers were recently superceded by a result in [OgWa 90] that the above assumption is false unless $P = NP$.

Chapter 2

Generalized bounded query hierarchies.

The goal of this chapter is to uniformly establish the relationships between different bounded query classes that are generated by fairly general reductions to fairly general base classes; in other words, a uniform generalization of the core results of Chapter 1. As a substantial portion of this endeavor, we will first define our "fairly general" reductions, $\leq^Q_{r-red}$, by specifying the requirements that they satisfy: this will involve a formal specification of a typical class Q of reduction complexities, and of the various nonadaptive oracle access mechanisms (the only adaptive reduction we will consider is the Turing reduction, $\leq^Q_{r-T}$.) Second, we will specify the requirements that our fairly general base classes satisfy, thus enabling us to study the relationships between the classes Q^C_{r-red} as the oracle access mechanism, *red*, varies.

Unlike most of Chapter 1, which dealt with constant query classes, this chapter will provide a uniform treatment of bounded query classes for all functions, r, that bound the number of oracle queries in the generating reductions, $\leq^Q_{r-red}$.

As in Chapter 1, our investigations will be based on the classes and hierarchies that are generated by the simplest of nonadaptive reductions, namely Boolean hierarchies. Here however, we will deal with extended Boolean hierarchies, or, in other words, we will give a uniform treatment of constant

and nonconstant levels of Boolean hierarchies.

As mentioned earlier, (see Section 1.4), finding a formal definition of extended Boolean hierarchies that is also intuitive provides a modest challenge. The normal form definition of Wagner for the extended Boolean hierarchy over NP (see Definition 11) is not straightforward for arbitrary complexity classes. Intuitively, we aim for a direct extension of Definition 1 to nonconstant levels roughly as follows.

Let t_k be any of the Boolean functions of Definition 3, (Section 1.1). For a base class, C, just as we defined

$$\mathbf{t_k[C]} =_{def} \{ X : \exists X_1, \ldots, X_k \in C \ \forall x \, [c_X(x) = t_k(c_{X_1}(x), \ldots, c_{X_k}(x))\,]\},$$

for constant k, we would ideally like to define

$$\begin{aligned} \mathbf{t_r[C]} =_{def} \{ X : \exists X_1, \ldots, & \textit{ a uniformly specifiable} \\ & \textit{infinite sequence of sets in } C \textit{ such that} \qquad (*) \\ & [c_X(x) = t_{r(|x|)}(c_{X_1}(x), \ldots, c_{X_{r(|x|)}}(x))\,]\}, \end{aligned}$$

for arbitrary functions, r.

To make such a definition consistent and meaningful, we need to specify not only an indexed subclass, $\{X_1, \ldots, X_{r(|x|)}\}$, of the base class, C, but also how the sequence, $\{t_1, \ldots, t_{r(|x|)}\}$, of Boolean functions is generated. Therefore, notational issues that are obvious while defining constant levels of Boolean hierarchies become delicate for extended Boolean hierarchies.

¿From the above discussion, it should be clear to the reader that the substance of this chapter is directed as much towards setting the stage for the generalization, i.e, finding the right definitions, specifying the extent of generalization, sorting out the notational issues, et cetera, as towards the generalization itself.

We note that a very brief abstract of this chapter appears in [BBJSY 89].

Organization.

This chapter consists of 4 sections. As mentioned earlier, the development of this chapter revolves around extended Boolean hierarchies. Studies of classes generated by more complicated reductions are built on this foundation.

Section 2.1 develops the machinery for dealing with general bounded query reductions, and finds a small set of requirements that bounded query reductions must satisfy in order to be able to generalize the results of Chapter 1 on polynomial-time reductions (over NP).

Section 2.2 develops the machinery for dealing with arbitrary base classes (especially for those that do not have complete sets), and formalizes the requirements that base classes must satisfy in order to be able to generalize, up to varying degrees, the results of Chapter 1 that were specific to the base class NP.

Section 2.3 gives a formal definition of extended Boolean hierarchies that is intuitive and independent of the base class. Furthermore, this section gives example problems in the constant levels of the Boolean hierarchy over RP, as a practical motivation for studying Boolean hierarchies over arbitrary base classes.

Section 2.4 generalizes the core results of Chapter 1 to fairly general base classes and fairly general reductions, thus establishing the main result stated in Page 7. Finally, this section concludes with a discussion of certain unsatisfactory gaps that arise from our definition of generalized bounded query classes, suggests an alternative adjusted definition, and points out

inconsistencies that show that even this new definition is inappropriate for specific cases.

2.1. Generalized nonadaptive reductions.

In this section, we formalize what we mean by a fairly general nonadaptive reduction, $\leq^{Q}_{r-red}$, by specifying the basic requirements that are satisfied by a typical class, Q, of reduction algorithms. We aim to find the minimal set of such requirements that still allows generalizations of the core results of Chapter 1. We will see that these requirements are quite nonrestrictive and, in particular, permit the class, Q, to be any reasonable class of complexity higher than log-space.

We begin with a definition of a *minimal class of functions*, Q, and proceed to a definition of *Q-truthtable* reductions followed by a discussion of more general nonadaptive reductions.

Definition 18:

A minimal class of functions is any class of functions, $Q : N \to N$ that:

- is closed under substitution (composition);
- contains a (numerical) successor function that we denote by $+1$;
- contains the constant function, $0(x) = 0$;
- contains the usual pairing functions, $\langle x_1, \ldots, x_n \rangle$, and projection functions, $p^n_i(x_1, \ldots, x_i, \ldots, x_n) = x_i$;
- contains the function, $Test(x, y, z, w)$, that returns z if $x \leq y$ and returns w if $x > y$;

- contains the characteristic function of the predicate that tests, of a string, y, and an input, x, whether y represents a partial concrete ordering of $\{1, 2, \ldots, x\}$.

If Q is any minimal class of functions, a subset $B \subseteq Q$ is called **bounded** if $f \in B$ implies that $2^{f \log(f)}$ is bounded by some function in Q. •

Note that whenever we explicitly take Q to be the set of all functions computable in polynomial-time or inn log-space, a natural choice for B would be the class:

$$\{f : f \in Q \ \& \ \exists \text{ a polynomial } p \text{ such that } f(x) \leq p(|x|)\}.$$

Definition 19:

Let Q be any minimal class of functions,

1. A **Q-truthtable** is a function $t \in Q$ that on input x, produces the representation of a concrete truthtable (Boolean circuit), $t^x(x_1, x_2, \ldots, x_{\nu_t(x)})$, such that
   - the function, $\lambda x \ t^x(\vec{0}_{\nu_t(x)})$, is in the class Q,
   - the function, $\lambda x, \tau \ \mu_{t^x,\tau}(\vec{a}_\tau)$, that computes the number of mind changes of t where the variable τ denotes a partial concrete orderings of $\{1, 2, \ldots, \nu_t(x)\}$, is in the class Q, and
   - the function, $\nu_t(x)$, that simply counts the number of variables in t^x is in the subclass B.

Often we will only be interested in functions, r, such that $\nu_t(x)$ is bounded by $r(|x|)$.

2. For sets, X and Y, we write $X \leq^Q_{r-tt} Y$ and say that X is Q-truthtable reducible to Y with at most $r(|x|)$ (parallel or nonadaptive) queries to

Y if there exists a Q-truthtable, t, and a function, $f \in Q$, such that

$$c_X(x) = t^x(c_Y(f(x,1)),\ldots,c_Y(f(x,\nu_t(x)))),$$

where $\nu_t(x) \leq r(|x|)$. We will use the obvious definition for the non-adaptive $\leq^Q_{r-||^-}$, $\leq^Q_{r-bf^-}$, $\leq^Q_{r-ftt}$-reductions and the (adaptive) Turing reduction, $X \leq^Q_{r-T} Y$, and we will sometimes refer to $\{f(x,i)\}$ as the **set of queries** to the oracle, Y, and $\{c_Y(f(x,i))\}$ as the **set of answers** from Y.

•

Notice that for any minimal representation of truthtables, such as Boolean formulas or circuits, and for any Q-truthtable, t, $\nu_t(x)$ is smaller than the size of the representation of t^x. In most reasonable complexity classes of functions, and certainly for those at least as strong as log-space, if we can compute a function, f in Q then the only thing that stops us from computing 2^f, and placing it in Q, is that 2^f is too long to write out. The relations, $|2^{\nu_t(x)}| \leq |t^x| + 1$, and $t \in Q$, thus motivate the requirement that $\nu_t \in B$.

Furthermore, in any complexity class, Q, of functions at least as strong as log-space, given the concrete truthtable, t^x, from a Q-truthtable, t, one can always calculate both the number of variables in the truthtable and the value, $t^x(\vec{0}_{\nu_t(x)})$. (As in Chapter 1, we will denote the vector or k-tuple of $v(x)$ zeroes or ones, for any function $v \in Q$ as $\vec{0}_{v(x)}$ or $\vec{1}_{v(x)}$ and often drop the subscript when the context is clear). Thus, in classes of functions like log-space the functions, ν_t, will *automatically* be in the bounded subclass B and the function, $\lambda x\ t^x(\vec{0}_{\nu_t(x)})$, will *automatically* be in the class, Q, if the function, t, that produces the concrete truthtable is in the class, Q.

The situation with respect to the function, $\lambda x, \tau\ \mu_{t^x,\tau}(\vec{a}_\tau)$, is only slightly more difficult. The concrete truthtable, t^x, must itself be produced by a log-space calculation and $\nu_t(x) \leq |t^x|$. Thus from the order, τ, we can successively produce the sequence of vector substitutions into t^x and evaluate the truthtable as we do the substitutions. Clearly the count of mind changes can be maintained within log-space. Thus in reasonable complexity classes of functions, Q, and certainly for those at least as strong as log-space, the only necessary requirement for a truthtable to be a Q-truthtable is that the function t^x be a function in Q; the additional conditions "•" are all superfluous. However, as we shall see, special truthtables based on Definition 3, have such nice properties that the calculation of mind changes may be possible by *ad hoc* methods even when the underlying functional class, Q, is not strong enough to permit the successive substitutions just described.

On generalized truthtables.

The above discussion makes clear that for classes, Q, that are at least as strong as log-space, the actual representation of the concrete truthtable in a $\leq^Q_{tt}$-reduction is immaterial. In other words, $\leq^Q_{tt}$- and $\leq^Q_{bf}$-reductions are equivalent. Furthermore, for classes, Q, that are at least as strong as log-space, a $\leq^Q_{r-tt}$-reduction from a set, X, to a set, Y, is equivalent to any nonadaptive a $\leq^Q_{r-||}$-reduction that permits $c_X(x)$ to be any function in Q with $r(|x|)$ answers from Y as arguments. If however, we are interested in *full* Q-truthtable or $\leq^Q_{ftt}$-reductions, the situation is different but easily manageable. Intuitively, full Q-truthtables, $t \in Q$, are functions that on input, x, generate full, redundant (nonminimal) representations of concrete Q-truthtables, t^x: evaluating t^x involves no more than a table look-up, rather than a general computation of complexity, Q. However, it will become

clear that for classes, Q, that are at least as strong as log-space, $\leq^Q_{O(2^r)-ftt}$-reductions - to the fairly general complexity classes that we will consider - are equivalent to $\leq^Q_{O(r)-tt}$-reductions. The readers should convince themselves of this once we have specified the restrictions on our fairly general base classes in Section 2.2.

Therefore, without loss of generality, in the remainder of this thesis, we will only deal with $\leq^Q_{tt}$-reductions. In addition, when we refer to a *truthtable* we *always* have in mind a Q-truthtable for some fixed minimal class of functions, Q. When we mean to specify a concrete truthtable instead of a Q-truthtable, (a truthtable generating function), we will make sure that the context is clear.

We now extend the definition of partial concrete orderings and mind changes for concrete truthtables (Definition 4), to ordering and mind change *functions* for general Q-truthtables.

Definition 20:
Let Q be any minimal class of functions, and let t be any Q-truthtable. Let O be a function that on input, x, produces a (concrete) permutation, O^x, of the set, $\{1, 2, \ldots, \nu_t(x)\}$. We extend the definition of the function, Ord_O, (Definition 4) to the **ordering function**, **O**, by defining $Ord_O(x, i) =_{def} (O^x)^{-1}(i)$, that gives the "position" of the i^{th} variable of t^x in the ordering, O^x. Accordingly, we formalize the notion of mind changes for general Q-truthtables by defining the **mind change function**, $\mu_{\mathbf{t},\mathbf{O}}$, by

$$\mu_{t,O}(x, \vec{b}, i) = \mu_{t^x,O^x}(\vec{b}, i),$$

where the reader will recall that μ_{t^x,O^x} gives the number of mind changes of truthtable, t^x, in going from $\vec{0}_{\nu_t(x)}$ to $\vec{b}$ by flipping i bits in the order

dictated by the order, O^x.

- We say that the ordering function, O, is a **Q-ordering** of the truthtable, t, if the functions, $\lambda x, i\ Ord_O(x,i)$, and $\lambda x, i\ \mu_{t,O}(x, \vec{1}, i)$ are in Q.
- For any Q-truthtable, t, and associated Q-ordering, O, we define the "maximal mind changes" on input, x, as follows:

$$Max\mu_{t,O}(x) =_{def} max_{\{\vec{b}:\ |\vec{b}|\ =\ \nu_t(x)\}} \{\mu_{t,O}(x, \vec{b}, \nu_t(x))\}.$$

•

Recall that the functions, ν_t, and $\lambda x, \tau\ \mu_{t^x,\tau}(\vec{a}_\tau)$, were required to be in Q for the definition of Q-truthtables. However, if we are dealing with a class, Q, that contains all log-space computable functions, if we can compute both t^x and O^x within the class, Q, then it will *automatically* follow that both of the functions, Ord_O, and $\mu_{t,O}$ are in Q. Furthermore, since $\mu_{t,O} \leq \nu_t$, $\mu_{t,O}$ will automatically be in the subclass, B, if it is in Q. Thus in Definitions 19 and 20, the *explicit* assumptions that the functions, ν_t, Ord_O, $t^x(\vec{0}_{\nu_t(x)})$ and the various mind-change functions be in the classes, B or Q are only needed for *very* minimal classes, Q, of functions. *For classes, Q, of functions with reasonable computational power and for any reasonable choice of the subclass, B, there is no way to avoid having these functions in Q or in B.*

With this background, we make some preliminary remarks on extended Boolean hierarchies based on any of the special families $\{\lambda k\ t_k\}$, of Boolean functions from Definition 3.

Example 21:

1. Let $\{\lambda k\ t_k\}$ be any of the sequences of k-ary truthtables of Definition 1.2 and suppose that Q is any minimal class of functions powerful enough that, from the string $1^{|x|}$ we can produce the concrete truthtable, $t_r^x = t_{r(|x|)}$, by a function, t_r, in the class, Q. (For example, demanding that Q contain all log-space computable functions is much more than adequate). Let $\nu_{t_r}(x) = r(|x|)$ be *any* integer valued function in the bounded subclass, B. First observe that $t_r^x(\vec{0})$ is always identically 0. These functions are always in Q. Thus, t_r will be a Q-truthtable provided that the function, $\lambda x, \tau\ \mu_{t_r,\tau}(x, \vec{a}_\tau)$, is also in the class, Q, for any partial concrete ordering, τ, of $\{1, 2, \ldots, r(|x|)\}$. For minimal function classes, Q, at least as large as log-space, we have seen that this condition will always hold.

2. For any of these four choices for the Q-truthtable t_r based on Definition 3, consider the identity ordering, $O^x(i) = i$. It is easy to see that

$$\mu_{t_r,O}(x, \vec{1}, i) = i, \quad \text{for } i \leq r(|x|)$$

(and of course $\mu_{t_r,O}(x, \vec{1}, i) = r(x)$ for $i \geq r(x)$). Furthermore, $Ord_O(x, i) = i$. Thus the functions, $\lambda x, i\ Ord_O(x, i)$, and $\lambda x, i\ \mu_{t_r,O}(x, \vec{1}, i)$ are trivially in Q. Thus for any minimal collection of functions, Q, and any function, $\nu_{t_r} \in B$, the identity ordering is trivially a Q-ordering for the Q-truthtable, t_r, for any choice of t_r based on the truthtables of Definition 3. •

2.2. Generalized base classes.

In this section we specify a small set of requirements that is sufficient for base classes, C, to satisfy in order to permit a generalization of the results

of Chapter 1 to extended Boolean hierarchies over C and to bounded query classes, $Q_{tt}^{C}[r]$ and $Q^{C}[r]$, over C. These requirements are based partly on the desired intuitive definition of extended Boolean hierarchies (see Equation $(*)$, this chapter) over general complexity classes, and partly from the analyses of assumptions made in the proofs of Chapter 1 for constant levels of the Boolean hierarchy over the base class NP.

Indexing.

The straightforward definition of extended Boolean hierarchies (Equation $(*)$, this chapter) requires the substitution of arbitrary sets, $X_1, \ldots, X_{r(|x|)}$, into truthtables that depend on the input, x. For base classes that have a complete set, Y, one can hope to code all of the sets, $X_1, \ldots, X_{r(|x|)}$, into Y through uniform successive reductions that are no more powerful than the associated Q-truthtable. For instance, one could define extended Boolean hierarchies generated by the Q-truthtables in Example 2.4 as

$$t_r[C] =_{def} \{X : \exists Y \ Q\text{-}complete\ for\ C\ and\ \exists f \in Q$$
$$[\ c_X(x) = t_r^x(c_Y(f(x,1)), \ldots, c_Y(f(x, r(|x|)\ )\ )\ )\ ]\ \}.$$

However, for base classes that do not have complete sets, we need a suitable *indexing* of sets in the base class so that substitutions into the truthtable are well-defined.

Definition 22:

An **indexed collection**, $I : S \to C$, is a collection, C, of sets together with a function, I, mapping some subset, S, of $\{0,1\}^*$ onto C and a decomposition of C into $\bigcup C_k$, $C_k \subseteq C_{k+1}$. We use the notation, X_j, for the set $I(j)$ and the notation, S^k, for the set of indices, $I^{-1}[C_k]$, and we often abbreviate the

above notation and just refer to the class, C, itself as an indexed collection.

Intuitively, for machine based complexity classes, the set, S, is normally chosen to be just some canonical set of machines each of which accepts a (not necessarily distinct) member of C, and X_j is just the set accepted by the machine, j. Note that, for many base classes, C, like RP, *Few* P, we can not reasonably demand that the set, S, of machines that define C be easily decidable, or even decidable at all. Then, S^k could just be the slice of machines for sets in C that run within a certain constructible resource bound. For example, for NP, S^k might be the set of machines running nondeterministically within time bound $|x|^k$. For another indexed collection, we might take S_k to be machines running in nondeterministic time $2^{k|x|}$. If we consider a class like the recursively enumerable sets where uniformity is not an issue, then for each k, S_k might simply be *all* Turing machines.

Closure requirements.

¿From the analyses of assumptions that were made for various results in Chapter 1, it is not hard to see that Boolean hierarchies that are built from base classes (such as NP or RP) that are closed under finite intersection, union, and $\leq_m^P$-reductions behave in a similar fashion up to constant levels. In order to generalize these notions, we will need to extend the finite closures under union and intersection to include closure under appropriate slowly growing, but infinite, unions and intersections.

Definition 23:

Let Q be a minimal class of functions with B a bounded subclass of Q. Let $I : S \to C = \bigcup C_k$ be any indexed collection of sets. We say that C is

fully closed under disjunctions (under unions) if for every k and every function, $u \in Q$, with *range* $(u) \subseteq S^k$ and every function, $q \in Q$, the set, X, defined by

$$x \in X \iff \bigvee_{i \leq q(x)} x \in X_{u(i)} (\in C) \qquad (**)$$

is also in C. We shall abuse this notation when the meaning is clear and write X as $\bigcup_{i \leq q(x)} X_{u(i)}$. We say that C is closed under **bounded disjunctions** (unions) if for every function, $u \in Q$, with $range(u) \subseteq S^k$ and every function, $v \in B$, the set, X, defined by $(**)$ with q replaced by v is also in C. Similar definitions hold for closure under full and bounded conjunctions (intersections). We will say that a sequence of sets, $\lambda i\ X_{u(i)}$ is **uniform in** C if $u \in Q$ and $range(u) \subseteq S^k$.

•

In the special cases where the class, Q, is the set of functions computable in polynomial-time or in log-space, we sometimes call full closure under disjunctions simply closure under **exponential** unions, and in this case we call bounded closure under disjunctions closure under **polynomial** unions.

Definition 24:
We say that an indexed collection of sets, $C = \bigcup C_k$, is uniformly closed under **bounded conjunctive reductions** (from the class, Q,) if for all functions, $f \in Q$, for all k and all functions, $u \in Q$, with $range(u) \in S^k$, and for all functions, $v \in B$, there exists a k' and a function, $u' \in Q$ with $range(u') \subseteq S^{k'}$ such that for all j

$$x \in X_{u'(j)} \iff \bigwedge_{i \leq v(x)} f(x, i, j) \in X_{u(i)}.$$

We say that the sequence, $\{\lambda j\ X_{u'(j)}\}$, is **uniformly reducible** to the sequence, $\{\lambda j\ X_{u(j)}\}$, and, as in Definition 23, we often simply say that the sequence, $\{\lambda j\ X_{u'(j)}\}$, is a **uniform sequence** in C. •

Note that any class that is closed under bounded conjunctive reductions is trivially closed under bounded conjunctions or intersections and under $\leq_m^Q$-reductions.

Definition 25:

We say that an indexed collection of sets, C, which contains at least one nontrivial set, that is, a set that is neither empty nor universal, is **uniformly closed** if it is closed both under bounded unions and bounded conjunctive reductions. •

Example 26:

For each uniformly closed class, C, in the following examples, the reader should choose a reasonable bounded subclass, B.

1. For both the function class (Q) of all functions computable in NC and the function class (Q) of all functions computable in log-space, the base class (C) RNC (Random NC, defined analogously to Random polynomial-time, or RP) is uniformly closed.
2. For both the function class (Q) of all functions computable in polynomial-time and the function class (Q) of all functions computable in log-space, the base classes $C = NP$, $NEXP^{linear}$, $NEXP^{poly}$, RP, $FewP$, and RE all are uniformly closed. Furthermore, NP, $NEXP^{linear}$, $NEXP^{poly}$, and RE are also fully closed under (exponential) unions.
3. For the class of functions computable in EXP^{linear}, the base classes $NEXP^{linear}$, $NEXP^{poly}$, and RE are uniformly closed and are fully

closed under unions. For both the function class of all functions computable in EXP^{linear} and the function class of all functions computable in EXP^{poly}, the base classes $NEXP^{poly}$ and RE are uniformly closed and are fully closed under unions.

4. For the class of total recursive functions, the base class, RE, of all recursively enumerable sets is uniformly closed and is fully closed under unions.
5. Take Q to be the class of functions computable in polynomial time and let X be any set in NP. Then the base class, $C \subseteq NP$, of all sets $\leq_m^P X$ is uniformly closed and its closure under full unions is contained in NP. If X is in RP, then the same base class, C, is a uniformly closed subset of RP, but we cannot be sure that its full closure under unions is contained in RP.

We will be most interested in proving theorems about extended Boolean hierarchies built over uniformly closed complexity classes. However, we will point out how to strengthen these theorems for classes like NP that are also fully closed under unions. A major goal will be to delineate differences in Boolean hierarchies built over base classes like RP that are merely uniformly closed and those built over base classes like NP that are also fully closed under unions. As we shall see, the extent to which base classes are closed under more than bounded unions directly influences the ease with which one can build extended hierarchies up to arbitrary levels.

2.3. Generalized Boolean hierarchies: definitions and example problems.

We are finally equipped with sufficient machinery to define extended Boolean hierarchies over general base classes. Following this definition, as a practical motivation for studying such hierarchies, we give example problems in (constant levels of) the Boolean hierarchy over the class, RP, which is not as well-behaved as NP: assuming $NP \neq RP$, RP has no known complete problems, no known recursive enumeration of machines, and is not fully closed under unions, unlike NP.

Definition 27:
Let Q be any minimal collection of functions, and let t be any Q-truthtable, and $I : S \rightarrow C$ an indexed collection of sets. Analogous to Definition 1 for the regular Boolean hierarchy, we build classes in an extended Boolean hierarchy by defining

$$t[C] =_{def} \{ X : \exists k, \exists u \in Q \textit{ with } range(u) \subseteq S^k \forall x [c_X(x) = t^x(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x))] \}.$$

•

As the reader will have observed, we will often not use the functions, $\nu_t(x)$, to define the levels of an extended Boolean hierarchy generated by t, but rather the functions, $r(|x|)$, that tightly bound $\nu_t(x)$. Therefore we formally define the class of functions, r, that correspond to functions, ν_t, either in Q or in the subclass, B, of Q.

Definition 28:
For a minimal class, Q, of functions and its bounded subclass, B, the class, **Qexp,** (resp. **Bexp**) consists of all functions, r, for which $r(|x|) = v(x)$ for some function v in Q (resp. B.) •

Now we are ready to define the r^{th} level of the extended Boolean hierarchy based on the truthtables of Definition 3.

Definition 29:
Let C be a uniformly closed class, Q a minimal class of functions, and B a bounded subclass of Q, and $I : S \rightarrow C$ an appropriate indexing of C. Let $\{\lambda_k\ t_k\}$ be any of the sequences of Boolean functions of Definition 3. Then the $\mathbf{r^{th}}$ **level** of a standard extended Boolean hierarchy over C is denoted $\mathbf{t_r[C]}$, and defined as follows.

$$t_r[C] =_{def} \{X : \exists k\ \exists u \in Q \textit{ with } range(u) \subseteq S^k$$
$$[\ c_X(x) = t_r^x(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(r(|x|))}}(x))\ ]\ \},$$

where t_r^x is the concrete truthtable, $t_{r(|x|)}$. Notice that we have just defined Q-truthtables, t_r, for the Boolean functions of Definition 3, as in Example 21, so that t_r^x is merely the concrete truthtable generated by t_r on input x, and the function $\nu_{t_r}(x)$ is merely the function, $r(|x|)$. •

Example problems.

As a practical motivation for studying Boolean hierarchies over arbitrary complexity classes, we now give natural examples of languages that lie in the constant levels of the Boolean hierarchy over RP. These languages are of particular interest because they represent natural problems that are in BPP but do not seem to lie in RP.

The Boolean hierarchy over RP obviously contains RP but is contained in BPP. It is of particular interest because so little is known about sets that might be in $BPP - RP$. In [SoSt 77], it was shown that primality testing is in $coRP$. More recently, it was claimed in [AdHu 87] that primality testing

is also in RP, and hence primality testing is now believed to be in ZPP. As a consequence, problems such as determining whether a number is perfect and determining whether a number is Carmichael that were previously conjectured to be in $BPP - RP$ ([BaMiSh 86]) are now believed to be in RP. Here, we give examples of problems that are in the Boolean hierarchy over RP but seem not to be in RP, thus contributing to renew the belief that RP is properly contained in BPP.

It follows from results in [BaMiSh 86], [RaSh 88] and the claim that the set of primes is in ZPP that the sets $\{x : x \textit{ is perfect and is the sum of two integer squares}\}$ and $\{x : x \textit{ is perfect and is the sum of three integer squares}\}$ are in RP. These results together with Lagrange's proof of Fermat's well-known conjecture that every integer can be represented as the sum of four (possibly zero) integer squares, show that

1. $\{x : x \textit{ is perfect and is the sum of three integer squares but is not a sum of two integer squares }\}$ is in $RP[2]$, and
2. $\{x : x \textit{ is perfect and is the sum of four integer squares but is not a sum of three integer squares }\}$
   $\cup \{x : x \textit{ is perfect and is the sum of two integer squares }\}$ is in $RP[3]$.

Currently there are no known techniques that place either of these two sets in RP. Of course, we should point out that it is not known that there are infinitely many perfect numbers (although this is believed to be the case), and even if there are infinitely many perfect numbers, either of the sets described above could be empty, although this is believed to be unlikely.

A second, and rather different, collection of problems involving polynomials that are represented by straight line programs can also be shown to be in the Boolean hierarchy over RP. It was shown in [Sch 80] that checking

whether a polynomial, p, represented as a straight line program, is identically zero is a $coRP$ problem. More formally,

Definition 30:

A **straight line program**, Φ, is an ordered sequence of instructions, $I_1, \cdots, I_{l(\Phi)}$, such that an individual instruction has one of the following forms:

$x_k \leftarrow x_j + x_i;$

$x_k \leftarrow x_j - x_i;$

$x_k \leftarrow x_j x_i;$

$x_k \leftarrow 0;$

$x_k \leftarrow 1;$

where x_k, x_j, x_i are indexed variables and the instructions of the form $x_i \leftarrow \ldots$ and $x_j \leftarrow \ldots$ appear earlier in the program than any of the above instructions.

For a straight line program, Φ, we will let x_m denote the variable with greatest index contained in Φ, let x_Φ denote the value assigned to the variable, x_m, at the end of the computation of Φ, and let $l(\Phi)$ denote the number of instructions of Φ. If we allow instructions of the form: $x \leftarrow z$; where z is an indeterminate, then Φ describes a polynomial, $P_\Phi(z)$, of degree at most $2^{l(\Phi)}$. •

With this notation we can now formally describe our next set of problems the first of which was given in [Sch 80] as an example of a problem in $coRP$.

IDENTICALLY ZERO POLYNOMIAL.

Instance: a straight line program, Φ, with an indeterminate, z.

Question: is $P_\Phi(z)$ identically null?

Status: in $coRP$.

EQUIVALENT PROGRAM EVALUATIONS.

Instance : two straight line programs, Φ_1 *and* Φ_2.

Question: is x_{Φ_1} equal to x_{Φ_2}?

Status: in *coRP*.

The second problem can clearly be $\leq_m^P$-reduced to the first, so it too is in *coRP*.

Further problems involving straight line programs for instance finding the GCD of the polynomials represented by two input straight line programs, were shown in [Kal 88] to be in *RP*.

Next we consider a problem involving straight line programs that is in *RP*[2], but is probably not in *RP* or *coRP*. More precisely, it was shown in [BBJSY 89] that the following problem is not in $RP \cup coRP$ unless the above two problems fall into *ZPP*. For a polynomial $p(z) = \sum_{k=0}^{n} a_k z^k$, we will denote the coefficient of z^k in $p(z)$ by $[z^k]p(z)$.

MONOMIAL PROGRAM.

Instance : a straight line program, Φ, with an indeterminate, z.

Question: does Φ represent a monomial? I.e., does there exist exactly one k such that $[z^k]P_\Phi$ is not equal to 0?

Status: in *RP*[2].

2.4. Generalized Boolean hierarchies: properties.

We now generalize some of the results of Chapter 1. First we show that the definition of extended Boolean hierarchies over arbitrary base classes

(Definition 29) is robust, and that one need not have complete sets in the base class in order to give an "indexing free" definition such as Wagner's normal form definition for extended Boolean hierarchies. Second, we show that the relationship between this hierarchy and adaptive and nonadaptive bounded query hierarchies carries over from polynomial-time reductions and the base class NP to all minimal Q-reductions and all uniformly closed base classes, C.

Robustness.

In the following, we give a generalization of Theorem 5 for extended Boolean hierarchies over general base classes. In other words, we show that up to a certain level, Definition 29 is a robust definition of extended Boolean hierarchies over general base classes, that is, extended hierarchies that are based on the truthtables of Definition 3 are all equivalent. Furthermore, we characterize Q-truthtables, t, for which the class, $t[C]$, is contained in any given level of the extended Boolean hierarchy over C.

For base classes that are uniformly closed, the following results characterize and demonstrate robustness of extended Boolean hierarchies up to levels defined by slowly growing functions. However, these results apply to arbitrary levels in the case of classes like NP that are fully closed under unions, thus providing a complete generalization of Theorem 12.

Moreover, these results also show that one need not have complete sets in the base class, C, in order to give an "indexing free" definition of the extended Boolean hierarchy over C. In particular, for fully closed base classes, our definition of extended Boolean hierarchies is equivalent to Wagner's normal form definition (Definition 11) for all levels; for base classes that are merely uniformly closed, this equivalence extends only up to levels that are

defined by slowly growing functions, (for instance, in the case of RP, up to logarithmic levels).

The following lemma proves one direction of the above equivalence by showing that any set, X, in the class, $t[C]$, generated by an arbitrary Q-truthtable, t, is in fact reducible to a single set, Y, by a reduction that is based on the *parity* truthtable. However, the membership of Y in C depends on the extent to which C is closed under unions.

Lemma 31:
Consider any Q-truthtable t, and uniformly closed complexity class C. Then for any set $X \in t[C]$, there exists a set Y such that

1. $$c_X(x) \;=\; \sum_{i=1}^{\nu_t(x)} [c_Y(i,x)] \;+ t^x(\vec{0}_{\nu_t(x)}) \mod 2,$$
2. $c_Y(i,x) \;\geq\; c_Y(i+1,x)$, and $c_Y(\nu_t(x)+1,x) = 0$,
3. Y is in the closure of C under unions bounded by a function in Q, and
4. $Y \in C$ if C is fully closed under unions, or if $2^{\nu_t \log(\nu_t)}$ is bounded by a function in B.

Proof:

Let $I : S \to C$ be an indexing of C. By definition of $t[C]$, the set, X, is definable, using a function, $u \in Q$, with *range* $(u) \subseteq S^k$, as $c_X(x) = t^x(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x))$. Thus we must prove that

$$t^x(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x))$$

$$= \sum_{i=1}^{\nu_t(x)} [c_Y(i,x)] \;+\; t^x(\vec{0}_{\nu_t(x)}) \mod 2$$

for an appropriate set, Y. From this formulation, it is clear that to establish Part 1, it is adequate to define the set, Y, in such way that for every choice of x there is some concrete ordering, τ, of $\{1, 2, \ldots, \nu_t(x)\}$ such that

$$\sum_{i=1}^{\nu_t(x)} [c_Y(i,x)] \;=\; \mu_{t^x,\tau}(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x)).$$

We will define Y to be

$$\begin{aligned} Y \;=_{def}\; \{\, (i,x) : & \textit{ for some concrete ordering, } \tau, \textit{ of } \{1,2,\ldots,\nu_t(x)\} \\ & \textit{there exist at least } i \textit{ mind changes of } t^x \\ & \textit{between } \vec{0}_{\nu_t(x)} \textit{ and} \\ & (c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x)) \,\}. \end{aligned}$$

The definition of the set, Y, makes both 1 and 2 obvious.

It remains only to locate the set, Y, in the closure of the class C under unions bounded by members of Q. We do not know, in general, if the class, C admits the power of nondeterminism and therefore we cannot just define Y by a positive nondeterministic reduction to the set, X, as we could do in Theorem 5 for the case when $C = NP$. Therefore, we perform a rigorous construction of the set, Y, through uniform bounded unions and intersections of sets in C.

We begin by observing that for the sequence $X_{u(1)}, X_{u(2)}, X_{u(3)}, \ldots$ which defines the set X, the related sequence $Z_1, Z_2, Z_3, \ldots$ given by

$$Z_j =_{def} \{1\} \bigcup \{x+2 : x \in X_{u(j)}\}$$

is easily seen to be a uniform sequence in C.

Define a function f in Q by

$$f(j,i,x,\tau) = 0 \qquad \text{if } \mu_{t^x,\tau}(\vec{a}_\tau) < i \quad or \quad \tau \text{ is not a concrete partial ordering of } \{1,2,\ldots,\nu_t(x)\};$$

$$= 1 \qquad \text{if } \mu_{t^x,\tau}(\vec{a}_\tau) \geq i \quad and \quad \vec{a}_{\tau,j} = 0 \quad and \quad \tau \text{ is a concrete partial ordering;}$$

$$= x+2 \quad \text{otherwise.}$$

Next define the sets, Y_τ, by

$$Y_\tau =_{def} \{(i,x) : \textit{the partial ordering, } \tau, \textit{ satisfies } [\vec{a}_{\tau,j} = 1 \Rightarrow c_{X_{u(j)}}(x) = 1], \textit{ and there exist at least } i \textit{ mind changes of } t^x \textit{ between } \vec{0}_{\nu_t(x)} \textit{ and } \vec{a}_\tau \}.$$

Note that for each τ if $(i,x) \in Y_\tau$, then there are at least i mind changes in going from $\vec{0}_{\nu_t(x)}$ to $\vec{a}_\tau$ by flipping only bits in the vector $\vec{a}_\tau$, doing so in the order determined by τ. Furthermore, the definition of Y_τ guarantees that if $(i,x) \in Y_\tau$ and $\vec{a}_{\tau,j} = 1$, then $x \in X_{u(j)}$. Therefore,

$$(i,x) \in Y_\tau \iff \bigwedge_{j \leq \nu_t(x)} f(j,i,x,\tau) \in Z_j.$$

Because the class C is closed under bounded conjunctive reductions, the sequence $\lambda\tau\ Y_\tau$ is a uniform sequence in C, it follows that $Y_\tau \in C$.

Clearly, for all partial concrete orderings, τ, we have that $Y_\tau \subseteq Y$, and furthermore, since the partial concrete ordering, τ, witnessing the containment can be nonconstructively chosen so that $\vec{a}_\tau$ happens to be

$(c_{X_{u(1)}}(x), c_{X_{u(2)}}(x), \ldots, c_{X_{u(\nu_t(x))}}(x))$, the set, Y, can be defined by

$$(i,x) \in Y \iff (i,x) \in \bigcup_{\tau \text{ is a partial ordering of } \{1,2,\ldots,\nu_t(x)\}} Y_\tau. \qquad (\dagger)$$

Since these concrete orderings are all coded as numbers less than, or equal to $2^{\nu_t(x)\log(\nu_t(x))}$, we have that

$$Y = (i,x) \in \bigcup_{\{j \,:\, j \leq 2^{\nu_t(x)\log(\nu_t(x))}\}} Y_j.$$

Thus Y is in the closure of C under unions bounded by a function in Q. This establishes Parts 3 and 4. ∎

Notice that in Equation (†), the union can in fact be taken over all evaluations (concrete orderings), O^x, of the *functions*, $O \in Q$, and furthermore, $c_Y(i,x) = 0$ for all $i \geq max_{O \in Q}\{Max\mu_{t,O}(x)\}$. Therefore the sum in Part 1 can be written more precisely as

$$c_X(x) = \sum_{i=1}^{max_{O \in Q}\{Max\mu_{t,O}(x)\}} [c_Y(i,x)] + t^x(\vec{0}_{\nu_t(x)}) \mod 2.$$

¿From our next lemma, it is easily seen that a full converse to Lemma 31 also holds provided that the set, Y, is actually in the class, C, as it always must be if C is closed under full unions *or* if the function, $2^{\nu_t \log(\nu_t)}$ is bounded by a function in B.

Lemma 32:
Consider any Q-truthtable, t, and uniformly closed complexity class, C. Let v be any integer valued function in Q and *initval* any Boolean valued function

in Q; Y is any nontrivial set in C such that $\forall x \forall i\ [c_Y(i,x) \geq c_Y(i+1,x)]$; and X is any set defined by

$$c_X(x) = \sum_{i=1}^{v(x)} [c_Y(i,x)] + initval(x) \mod 2.$$

Then, for any Q-truthtable, t, and any Q-ordering function, O for t,

$$[\, v(x) \leq \mu_{t,O}(x, \vec{1}_{\nu_t(x)}, \nu_t(x)) + t^x(\vec{0}_{\nu_t(x)}) + initval(x) \mod 2]$$
$$\Longrightarrow X \in t[C].$$

Proof:

We use the function, $Ord_O(x,i)$, that gives the position of the i^{th} variable of t^x in the ordering, O^x, and the function, $\mu_{t,O}(x, \vec{1}_{\nu_t(x)}, i)$, that gives the number of mind changes that have occurred just *after* flipping the i^{th} bit in the determination of $\mu_{t,O}(x, \vec{1}_{\nu_t(x)})$. Because O is a Q-ordering of t, both of these functions must be in the class, Q.

The idea of the proof is the same as that in Theorem 5: the truthtable, t, with its variables ordered by the ordering, O, has enough mind changes to do a mod 2 count of the values (i,x) for which the value of $c_Y(i,x)$ is "1." Hence, if we substitute

$$c_Y(1,x), \ldots, c_Y(1,x), c_Y(2,x), \ldots, c_Y(2,x), \ldots, c_Y(v(x),x), \ldots, c_Y(v(x),x)$$

into the variables for t^x in the order dictated by O, beginning the substitution of the next new variable each time a mind change of t^x occurs, then t^x, with

these substitutions, must give a mod 2 count of $\sum_{i=1}^{v(x)} [c_Y(i,x)]$. Since for any vector $\vec{b}$

$$t^x(\vec{b}) = \mu_{t,O}(x, \vec{b}, \nu_t(x)) + t^x(\vec{0}_{\nu_t(x)})) \mod 2,$$

doing the proper substitutions should give us the required result, provided the initial values $initval(x)$ and $t^x(\vec{0}_{\nu_t(x)})$ are identical. If these two variables are not identical, then we must do enough substitutions to effect one more mind change.

Formally, we proceed as follows. First, we define the function $v'(x) =_{def} v(x) + initval(x) + t^x(\vec{0}_{\nu_t(x)})$. It is easily seen that $v' \in Q$. Next, recall that the function $Ord_O(x,j)$ produces the "position" of the j^{th} variable of t^x in the order determined by O^x. With these two definitions, if we can force

$$c_Y(1,x),\ldots,c_Y(1,x),c_Y(2,x),\ldots,c_Y(2,x),\ldots,c_Y(v(x),x),\ldots,c_Y(v(x),x) \qquad (\dagger\dagger)$$

to be substituted into t^x in this order and then flip the bits in this order, then all of the bit flipping will occur, first in all those places where the Boolean values are assigned the value "1", followed by (attempted, but blocked) bit flipping in all those places where the Boolean values are assigned the value "0." This will force the mind changes in this order for this substitution to reflect the values in the summation $\sum_{i=1}^{v(x)} [c_Y(i,x)]$. Of course, if $initval(x) + t^x(\vec{0}_{\nu_t(x)}) \neq 0$, then we need to force one more mind change, so we use $v'(x)$ instead of $v(x)$ in $(\dagger\dagger)$.

To accomplish these substitutions, note that $\mu_{t,O}(x, \vec{1}_{\nu_t(x)}, Ord_O(x,j)-1)$ is the number of mind changes have occurred just *before* the j^{th} bit is flipped. We should be substituting the j^{th} variable during those periods

when the mind changes that have been witnessed total $j-1$, and we should quit when

$$\mu_{t,O}(x, \vec{1}_{\nu_t(x)}, Ord_O(x,j)) > v'(x).$$

Since the set Y is not trivial, we let m be some fixed member of $\overline{Y}$. To accomplish the substitutions we have just discussed, we define the function

$$\begin{aligned} f(x,j) &= (\mu_{t,O}(x, \vec{1}_{\nu_t(x)}, Ord_O(x,j)-1)+1, x) \text{ if} \\ &\qquad \mu_{t,O}(x, \vec{1}_{\nu_t(x)}, Ord_O(x,j)) \leq r'(x), \\ &= m \qquad \text{otherwise.} \end{aligned}$$

By our assumptions on Q and O, the function f is easily seen to be in Q, and we then have that

$$\begin{aligned} &\mu_{t,O}(x, (c_Y(f(1,x)), c_Y(f(2,x)), \ldots, c_Y(f(\nu_t(x),x))), \nu_t(x)) \\ &= \sum_{i=1}^{v(x)} [c_Y(i,x)] + initval(x) + t^x(\vec{0}_{\nu_t(x)}) \mod 2, \end{aligned}$$

and thus that

$$\begin{aligned} &t^x(c_Y(f(x,1)), c_Y(f(x,2)), \ldots, c_Y(f(x,\nu_t(x)))) = \\ &\sum_{i=1}^{v(x)} [c_Y(i,x)] + initval(x) \mod 2. \end{aligned}$$

It is easy to see that the sets, Y_j, such that

$$x \in Y_j \iff f(x,j) \in Y$$

are in the class, C, since C is closed under bounded conjunctive reductions and therefore trivially under $\leq_m^Q$-reductions. Furthermore, one can find an indexing, $I : S \to C$, of the class, C, and define a function, $u \in Q$, such that $range(u) \in S^k$ for some k and $X_{u(j)} (\in C) = Y_j$. The result then follows. ∎

Now we are ready for a generalization of Theorem 5. The following theorem shows that the number of mind changes of Q-truthtables, t and t', (together with a trivial check of the initial values of t and t') determine whether $t[C] \subseteq t'[C]$, for general base classes, C. For uniformly closed base classes, equivalence and uniqueness of the lower levels of the extended Boolean hierarchies defined by the sequences from Definition 3 will be directly established by a simple count of the mind changes together with a trivial check of the initial values of the truthtables. For fully closed base classes, these equivalence and uniqueness results extend to arbitrary levels. Furthermore, we obtain a generalization of Theorem 6 to give an indexing free, normal form characterization of extended Boolean hierarchies that was used by Wagner to *define* this hierarchy over NP (Definition 11).

Before proceeding, one more piece of notation will prove useful in order to avoid dealing with initial values.

Definition 33:
Let t and t' be any two Q-truthtables. Define the function,

$$\mathbf{adjust_{t,t'}(x)} =_{def} \; t^x(\vec{0}_{\nu_t(x)}) + t'^x(\vec{0}_{\nu_{t'}(x)}) \bmod 2.$$

(Note that $adjust_{t,t'}(x)$ is always in the function class, Q.) •

Theorem 34:
Let Q be any minimal collection of functions, B a bounded subclass of Q, and let C be any uniformly closed complexity class.

1. Let t and t' be Q-truthtables. Suppose either that C is closed under full unions *or* that ν_t satisfies $2^{\nu_t \log(\nu_t)} \in B$. Then, if there exists a

Q-ordering, O' for t', such that

$$max_{O \in Q}\{Max\mu_{t,O}(x)\} \leq \mu_{t',O'}(x, \vec{1}_{\nu_{t'}(x)}, \nu_{t'}(x)) + adjust_{t,t'}(x),$$

then

$$t[C] \subseteq t'[C].$$

2. Let t and ν_t be as in Part 1. Then for any function, $r \in Bexp$, and any Q-truthtable, t_r, built from *any* of the four Boolean functions of Definition 3 (as explained in Definition 29), if

$$\nu_t(x) + t(\vec{0}_{\nu_t(x)}) \leq r(|x|),$$

then

$$t[C] \subseteq t_r[C].$$

3. Let r be any function in $Bexp$ and suppose either that C is closed under full unions or that $2^{r\log(r)}$ is bounded by a function in $Bexp$. Let (following Wagner's definition over NP)

$$C[r] =_{def} \{X : \exists Y \in C\ [\forall x \forall i\ [c_Y(i+1,x) \leq c_Y(i,x)]\ and$$

$$c_X(x) = \sum_{i=1}^{r(|x|)} [\ c_Y(i,x)]\ \mod 2.\ ]\}$$

Then for the Q-truthtables that are based on Definition 3,

$$h_r[C] = g_r[C] = g'_r[C] = Diff_r[C] = C[r].$$

By the same argument, the complements of these classes, which we denote as $co\text{-}h_r[C]$, $co\text{-}g_r[C]$, $co\text{-}g'_r[C]$, $co\text{-}Diff_r[C]$, and $co\text{-}C[r]$ are also equal.

Proof:

1. From Lemma 31(1,2), for any set, $X \in t[C]$, there is a set, Y, such that

$$c_X(x) = \sum_{i=1}^{max_{O \in Q}\{Max\mu_{t,O}(x)\}} c_Y(i,x) + t_x(\vec{0}_{\nu_t(x)}) \mod 2,$$

and $\forall x \forall i\ [c_Y(i+1,x) \leq c_Y(i,x)]$.

Furthermore, from Lemma 31(4) and the conditions on ν_t required in this theorem, it follows that $Y \in C$. We have now established the antecedents to apply Lemma 32, and the result follows.

2. We simply replace t' of Part 1 by the Q-truthtable, t_r, based on Definition 29. From Example 29, we know that in the case of these Q-truthtables, t_r,the identity ordering is trivially a Q-ordering, that gives $r(|x|)$ mind changes between $\vec{0}_{\nu_{t_r}(x)}$ and $\vec{1}_{\nu_{t_r}(x)}$. ($\nu_{t_r}(x)$ is nothing but $r(|x|)$). Therefore, we obtain Part 2 by replacing $\mu_{t',O'}(x, \vec{1}_{\nu_{t'}(x)}, \nu_{t'}(x)) + adjust_{t,t'}(x)$, in Part 1 by $r(|x|)$ and by replacing $max_{O \in Q}\{Max\mu_{t,O}(x)\}$ in Part 1 by the simpler $\nu_t(x)$ plus the initial value, $t(\vec{0}_{\nu_t(x)})$ (since $t_r^x(\vec{0}_{\nu_{t_r}(x)}) = 0$ always, the latter, in effect, replaces $adjust_{t,t_r}(x)$).

3. Part 3 follows directly from Parts 1 and 2. ∎

The above theorem justifies our use, henceforth, of the standard **C[r]** (**co-C[r]**) to denote the r^{th} level of the extended Boolean hierarchy over C up to levels appropriate for the specific choice of C. For instance, for the base class, NP, that is fully closed under unions, r can be any polynomial, whereas for the base class, RP, that is only uniformly closed, $r(|x|)$ must be bounded by

$O(\log(|x|)\log\log(|x|))$, for the above equivalence (and standard notation) to go through.

Relationship to other bounded query classes.

We now give a generalization of Theorems 7 and 8 (and also, in a sense, Theorems 13 and 15), to extended Boolean hierarchies and bounded query hierarchies over arbitrary base classes. Since we have already discussed how to treat $\leq^Q_{r-bf}$, $\leq^Q_{r-||}$, and $\leq^Q_{r-ftt}$, uniformly as $\leq^Q_{tt}$ reductions, (see Page 59, Section 2.1), we will restrict our investigations in this section to the bounded query classes generated by $\leq^Q_{r-tt}$- and $\leq^Q_{r-T}$-reductions. The following theorem holds for *all* bounded query classes, $Q^C_{tt}[r]$, that are generated by reductions that have lower complexity than minimal Q-truthtable reductions. (That is, when the classes, Q, are *stronger* than a minimal class). In particular, when $C = NP$, the classes, $Q^C_{tt}[r]$, can be treated uniformly for all appropriate classes Q that are stronger than log-space. Therefore, the following theorem is not only a generalization of all the results in Chapter 1 that relate the Boolean hierarchy over NP to different bounded query classes, but is also a generalization of those results in Chapter 1 that relate log-space and polynomial-time bounded query classes over NP.

Theorem 35: Let Q be any minimal class of functions stronger than log-space, B a bounded subclass of Q, r a function in $Bexp$, and let C be a uniformly closed class.

1. If either $2^{r\log(r)} \in Bexp$ or C is fully closed under unions, then

$$C[r] \cup co\text{-}C[r] \subseteq Q^C_{tt}[r(|x|)] \subseteq C[r+1] \cap co\text{-}C[r+1].$$

2. If either $2^{2^r \cdot r} \in Bexp$ or C is fully closed under unions and $2^r - 1 \in Bexp$,

then

$$Q^C[r(|x|)] \;=\; Q^C_{tt}[2^{r(|x|)}-1].$$

Proof:
The proof is a straightforward generalization of the arguments in the proof of Theorem 7.

1. The containment, $C[r] \cup co\text{-}C[r] \subseteq Q^C_{tt}[r(|x|)]$, follows easily from the normal form definition of $C[r]$. (See Theorem 34(3)). The containment, $Q^C_{tt}[r(|x|)] \subseteq C[r+1] \cap co\text{-}C[r+1]$ follows from Theorem 34(2): replace the Q-truthtable, t, in 34(2) by the Q-computation that generates a $\leq^Q_{r-tt}$-reduction; since the initial values of these truthtables may differ, we need the leeway of the additional query in the right hand side of the above containment.

2. The stricter bounds on the function, r are required simply because the statement involves truthtable reductions with $2^{r(|x|)}$ queries. The containment, $Q^C[r(|x|)] \subseteq Q^C_{tt}[2^{r(|x|)}-1]$, is obtained in the straightforward manner by gathering all possible adaptive sequences of queries that correspond to all possible answers from the oracle into a set of $2^{r(|x|)}-1$ nonadaptive queries, an operation that is viable within any class, Q, that is at least as strong as log-space.

The reverse containment is obtained as follows. Consider any set, X, that is reducible to a set, $Y' \in C$, by a Q-truthtable reduction:

$$c_X(x) = t^x(c_{Y'}(f(x,1)),\ldots,c_{Y'}(f(x,2^{r(|x|)}))),$$

where t and f are in Q. We apply Lemma 31 to find another set, Y, to replace Y' such that $\forall x \forall i\ [c_Y(i+1,x) \leq c_Y(i,x)]$, and X is reducible to Y by the

standard reduction:

$$c_X(x) = \sum_{i=0}^{2^{r(|x|)}} c_Y(i,x) + t^x(\vec{0}_{r(|x|)}) \mod 2.$$

Since the function, c_Y, is monotone in i, the above sum can be computed merely by finding that j such that $1 \leq j \leq 2^{r(|x|)}$, and $c_Y(j,x) = 1$, but $c_Y(j+1,x) = 0$. However, this "turning point," j, can be located by performing a binary search on the interval, $[1, 2^{r(|x|)}]$, and making at most $r(|x|)$ adaptive queries to Y. The result then follows. ∎

For the case where Q is P and C is NP the above theorem is equivalent to the known result of Wagner (Theorem 13). When C is RP, however, Part 1 of the above theorem holds for $r(|x|)$ being at most $O(\log(|x|)\log\log(|x|))$, and Part 2 holds when $r(|x|)$ is at most $O(\log\log(|x|))$.

Conclusions.

In the process of proving the main result stated in Page 7 of the introduction, we have put forth a general theory of Boolean hierarchies and bounded query classes. Along the way, we have achieved an intuitive, indexing free definition of extended Boolean hierarchies even for base classes that do not have complete sets or recursive enumerations. Although these hierarchies are well-defined and robust only up to lower levels in the case of base classes that are merely uniformly closed, for instance, up to logarithmic levels in the case of *Few P* and *RP*, they extend to arbitrary levels in the case of classes that are fully closed under unions, for instance, *NP*, *NEXP*, *RE*, et cetera.

Notice, however, that the extended Boolean hierarchies as in Definition 29 are unsatisfactory since the levels of these hierarchies are not closed under

$\leq_m^Q$ reductions. For instance, sets in the r^{th} level of the extended Boolean hierarchy over NP may be $\leq_m^P$-reduced to sets in a lower, r'^{th} level by a reduction that simply pads the input and increases its size by a polynomial factor. In other words, as long as there is a polynomial p such that $r'(p(|x|)) = r(|x|)$, sets in the r^{th} level of the extended Boolean hierarchy over NP can be $\leq_m^P$-reduced to sets in the r'^{th} level. Therefore, one could argue that the extended Boolean hierarchy over NP is well-defined only up to sublinear levels as sets in arbitrary polynomial levels can always be reduced to sets in sublinear levels.

More generally, for an arbitrary base class, C, and the corresponding minimal class of functions, Q, define the class, *Qsize*, as consisting of all functions, q, for which $q(|x|) = |f(x)|$ for some function, f in Q. Then, for any function, r in *Qexp*, all sets in the r^{th} level of the extended Boolean hierarchy over C can be $\leq_m^Q$-reduced to sets in the lower, r'^{th} level, if there is a function, $q \in Qsize$ such that $r(|x|) \leq r'(q(|x|))$. In other words, one could argue that the highest level up to which the extended Boolean hierarchy over C is well-defined is determined by the least function, r', that satisfies the following.

$$\forall r \in Qexp\ \exists q \in Qsize\ [\forall x\ r'(q(|x|)) = r(|x|)].$$

We can show however, that this unsatisfactory gap in the definition of extended Boolean hierarchies can be rectified easily and intuitively by defining each level in such a manner that it is closed under $\leq_m^P$ reductions. A precise description follows.

Definition.

Let C be a uniformly closed complexity class with indexing, $I : S \rightarrow C$,

Q a minimal class of functions, and B a bounded subclass of Q. For each function, $r \in Qexp$, define the **Q-closure** of r, denoted $\mathbf{r^Q}$, as the class

$$\{r' : \exists q \in Qsize\ [r(|x|) = r'(q(|x|))\ ]\ \}.$$

Now for each Q-truthtable, t, define the (adjusted) class, $\mathbf{t^*[C]}$, as

$$\begin{aligned} t^*[C] \ =_{def}\ & \{\ X :\ \exists k,\ \exists\ u, f \in Q\ \ with\ range(u) \subseteq S^k \\ & [\ c_X(x)\ =\ t^x(c_{X_{u(1)}}(f(x)), c_{X_{u(2)}}(f(x)), \ldots, c_{X_{u(\nu_t(f(x)))}}(f(x)))\ ]\ \}, \end{aligned}$$

The r^{th} level, $t_r^*[C]$, of the extended Boolean hierarchy over C is now adjusted to be merely a collection of levels from the original Definition 12. More formally,

$$t_r^*[C] = \bigcup_{r' \in r^Q} t_{r'}[C].$$

Furthermore, the adjusted bounded query classes, $Q_{red}^{*C}[r]$, are merely the classes, $\bigcup_{r' \in r^Q} Q_{red}^C[r']$. •

We note that it is not hard to derive all of the results of Chapter 2 for these adjusted definitions. What, then, prevents us from using these adjusted definitions as standard for extended Boolean hierarchies and bounded query classes? To answer this question, recall Wagner's extension of Kadin's collapse result (Theorem 14), that the equivalence of two levels of the extended Boolean hierarchy over NP that are determined by two distinct "well-behaved" functions results in the collapse of PH.

This gives evidence that at least for the extended Boolean hierarchy over NP two distinct "well-behaved" levels, for instance, $NP[(|x|)^{1/k}]$ and $NP[(|x|)^{1/k+1}]$ are distinct. Hence we would like any definition of extended

Boolean hierarchies to distinguish between these two levels in the case of NP. Unfortunately, however, this criterion is not satisfied by the adjusted definition given above: since the Q-closure (in this case, polynomial-time-closure) of both the functions $|x|^{1/k}$ and $|x|^{1/k+1}$ is the class of all polynomials in $|x|$, sets from these two distinct levels of the original definition of the extended Boolean hierarchy over NP are contained in the same level in the adjusted definition. Therefore, while the new definition may seem intuitive on the surface, it is too coarse at least for the base class, NP.

The results in this thesis are to a large extent the strongest that one can obtain about generalized Boolean hierarchies. Stronger questions - for instance the consequences of collapse, and finding the most appropriate definition for nonconstant levels (from the two discussed above) - will have to be specific to particular base classes. However, such questions generate interesting open problems. For instance, it will be informative to find the consequences of the collapse of the Boolean hierarchy over RP, since the only known examples of problems in $BPP - RP$ lie in the constant levels of that hierarchy.

Bibliography.

[Ad 65] J. Addison, "The method of alternating chains," *Symp. Theor. Models,* North-Holland, (1965), 1-16.

[AmBeGa 88] A. Amir, R.J. Beigel, W.I. Gasarch, "Cheatable, P-terse, and P-superterse sets," *Tech. Report, Dept. of Computer Science, University of Maryland-College Park,* 2090, (1988).

[AmGa 87] A. Amir, W.I. Gasarch, "Polynomially terse sets," *Proc.* 2^{nd} *Conf. on Struct. Compl. Theory,* (1987), 22-27.

[AdHu 87] L. Adleman, M. A. Huang, "Recognizing primes in random polynomial time," 19^{th} *Symp. Theor. Computing,* (1987), 462-469.

[BaBoSc 85] J.L. Balcázar, R.V. Book, U. Schöning, "On bounded query machines," *Theor. Comp. Sci.,* 40 (1985), 237-243.

[BaDiGa] J.L. Balcázar, J. Díaz, J. Gabbarró, "Structural Complexity I," *EATCS monographs on Theor. Comp. Sci.,* 11, W. Brauer, G. Rozenberg, A. Salomaa Ed.s, Springer-Verlag.

[BaMiSh 86] E. Bach, G. Miller, J. Shallit, "Sums of divisors, perfect numbers and factoring," *SIAM J. Computing,* 15, (1986), 1143-1154.

[BaGiSo 75] T. Baker, J. Gill and R. Solovay, "Relativizations of the $P =$ *NP* question," *SIAM J. Computing,* 4 (1975), 431-442.

[Be 87a] R.J. Beigel, "Bounded queries to *SAT* and the Boolean hierarchy," *Theor. Comp. Sci.,* to appear.

[Be 87b] R.J. Beigel, "A structural theorem that depends quantitatively on the complexity of *SAT*," *Proc.* 2^{nd} *Conf. on Struct. Compl. Theory,* (1987), 28-32.

[Be 87c] R.J. Beigel, "SAT^A is terse with probability one," *Tech. Report, Dept. of Computer Science, Johns Hopkins University,* 87-04 (1987).

[Be 88a] R.J. Beigel, "Why are counting problems hard to approximate," *Tech. Report, Dept. of Computer Science, Johns Hopkins University,* 88-05 (1988).

[Be 88b] R.J. Beigel, "When are $k+1$ queries better than k," *Tech. Report, Dept. of Computer Science, Johns Hopkins University,* 88-06 (1988).

[Be 88c] R.J. Beigel, "NP-hard sets are P-superterse unless $R = NP$," *Tech. Report, Dept. of Computer Science, Johns Hopkins University,* 84-4 (1988).

[BeGa 87] R.J. Beigel, W.I. Gasarch, "Binary search is optimal for recursive graph theory," *Tech. Report, Dept. of Computer Science, University of Maryland-College Park,* P804, (1987).

[BeGa 88a] R.J. Beigel, W.I. Gasarch, "On the complexity of finding the chromatic number of a recursive graph I: the bounded case," *Annals of Pure and Applied Logic,* to appear.

[BeGa 88b] R.J. Beigel, W.I. Gasarch, "On the complexity of finding the chromatic number of a recursive graph II: the unbounded case," *Annals of Pure and Applied Logic,* to appear.

[BeGaHa 87] R.J. Beigel, W.I. Gasarch, L. Hay, "Bounded query classes and the difference hierarchy," *Archiv. Math. Logic,* 29 (1989) 69-84.

[BGGO 87] R.J. Beigel, W.I. Gasarch, J. Gill, J.C. Owings, "Terse, superterse and verbose sets," *Tech. Report, Dept. of Computer Science, University of Maryland-College Park,* 1806, (1987).

[BeGaOw 87] R.J. Beigel, W.I. Gasarch, J.C. Owings, "Nondeterministic bounded query reducibilities," *Annals of Pure and Applied Logic,* to appear.

[BeGi 81] C. Bennet and J. Gill, "Relative to a random oracle A, $P^A \neq NP^A \neq coNP^A$ with probability one," *SIAM J. Computing,* 10 (1981), 96-113.

[BeHeWe 89] R. Beigel, L. Hemachandra and G. Wechsung, "On the power of probabilistic polynomial time: $P^{NP[\log]} \subseteq PP$," *Proc. 4th Conf. Struct. Compl. Theory,* (1989), to appear.

[BlGu 82] A. Blass, Y. Gurevich, "On the unique satisfiability problem," *Information and Control,* 55, (1982), 80-88.

[BBJSY 89] A. Bertoni, D. Bruschi, D. Joseph, M. Sitharam, P. Young, "Generalized Boolean hierarchies and Boolean hierarchies over RP," *Proc. Conf. on Fund. Comput. Theory,* Lect. Notes Comp. Sci. 380 (1989).

[Bo 81] R.V. Book, "Bounded query machines: on NP and $PSPACE$," *Theor. Comp. Sci.,* 15 (1981), 27-39.

[BoKo 87] R.V. Book, K-I. Ko, "On sets truth-table reducible to sparse sets," *Proc. 2nd Conf. on Struct. Compl. Theory,* (1987), 147-154; also *SIAM J. Comput.,* 17 (1988) 903-919.

[BoLoSe 84] R.V. Book, T.J. Long, A.L. Selman, "Quantitative relativizations of complexity classes," *SIAM J. Comput.,* 13 (1984), 461-487.

[BoLoSe 85] R.V. Book, T.J. Long, A.L. Selman, "Qualitative relativizations of complexity classes," *J. Comput. System Sciences,* 30 (1985), 395-413.

[BoWr 81] R.V. Book, C. Wrathall, "Bounded query machines: on $NP()$ and $NPQUERY()$," *Theor. Comp. Sci.,* 15 (1981), 41-50.

[BrJoYo 89] D. Bruschi, D. Joseph, P. Young, "Strong separation of the Boolean hierarchy over RP," *Proc 3rd Italian Conf. on Theor. Comp. Sci.,* published by World Scientific Press, London, (Nov. 1989), 13 pages.

[BuHa 88] S. Buss, L. Hay, "On truth-table reducibility to SAT and the difference hierarchy," *Proc.* 3^{rd} *Conf. on Struct. Compl. Theory,* (1988), 224-233.

[CGHHSWW 88] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K.W. Wagner, G. Wechsung, "The Boolean hierarchy I: structural properties," *SIAM J. Comput,* 6 (1988), 1232-1252.

[CGHHSWW 89] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K.W. Wagner, G. Wechsung, "The Boolean hierarchy II: applications," *SIAM J. Comput,* 7 (1989), 95-111.

[CaHe 86] J. Cai, L. Hemachandra, "The Boolean hierarchy: hardware over NP," *Proc.* 1^{st} *Conf. Struct. Compl. Theory,* (1986), 105-124.

[CaMe 85] J. Cai, G.E. Meyer, "Graph minimal uncolorability is D^P complete," *SIAM J. Computing,* 16 (1987) No. 2.

[Ch 89] R. Chang, "On the structure of bounded queries to arbitrary NP sets," *Proc.* 4^{th} *Conf. Struct. Compl. Theory,* (1989), 250-258.

[ChKa 90] R. Chang, J. Kadin, "The Boolean hierarchy and the polynomial hierarchy: a closer connection," To appear in *Proc.* 5^{th} *Conf. Struct. Compl. Theory,* (1990), 250-258.

[EpHaKr 81] R.L. Epstein, R. Haas, R.L. Kramer, "Hierarchies of sets and degrees below $0'$," *Logic Year 1979-80,* Lect. Notes Math. 859 (1981) 32-48.

[Ep 79] R.L. Epstein, "Degrees of unsolvability: structure and theory," Lect. Notes Math. 759 (1979).

[Er 68a] Y. Eršov, "A hierarchy of sets, I," *Algebra and Logic,* 7 (1968), 25-43.

[Er 68b] Y. Eršov, "A hierarchy of sets, II," *Algebra and Logic,* 7 (1968), 15-47.

[Er 69] Y. Eršov, "A hierarchy of sets, III," *Algebra and Logic,* 9 (1969), 20-31.

[GeGr 86] J. Geske and J. Grollmann, "Relativizations of unambiguous and random polynomial time classes," *SIAM J. Computing,* 15 (1986), 511-519.

[Ga 86] W.I. Gasarch, "The complexity of optimization functions," *Tech. Report, Dept. of Computer Science, Univ. of Maryland-College Park,* 1652 (1986).

[GaHeHo 90] W.I. Gasarch, L.A. Hemachandra, A. Hoene, "On checking versus evaluation of multiple queries," *Tech. Report, Dept. of Computer Science, Univ. of Rochester,* 223, (1990).

[GaPe 88] W.I. Gasarch, S.R. Pearlman, "The complexity of optimization problems related to partition," *Tech. Report, Dept. of Computer Science, Univ. of Maryland-College Park,* 2028 (1988).

[GoJoYo 88a] J.A. Goldsmith, D.A. Joseph, P. Young, "Bi-immunity and P-closeness of P-cheatable sets in $P/poly$," *Tech. Report, Dept. of Computer Science, Univ. of Wisconsin-Madison,* to appear *J. Comp. Sys. Sci.*

[GoJoYo 88b] J.A. Goldsmith, D.A. Joseph, P. Young, "Using self-reduciblility to characterize polynomial-time," *Tech. Report, Dept. of Computer Science, Univ. of Wisconsin-Madison,* to appear *Inf. and Control.*

[GuWe 86] T. Gundermann and G. Wechsung, "Nondeterministic Turing machines with modified acceptance," *Proc. Conf. on Math. Found. Comp. Sci.,*, Lect. Notes Comp. Sci. 233 (1986), 396-404.

[GuWe 87] T. Gundermann and G. Wechsung, "Counting classes with finite acceptance types," *Computers and Artificial Intelligence,* 6 (1987), No. 5, 395-409.

[Ha 78] F. Hausdorff, *Set Theory,* Chelsea, 3rd ed., (1978).

[HaHe 86] J. Hartmanis, L. Hemachandra, "Complexity classes without machines: on complete sets for *UP*," *Proc.* 13^{th} *Intl. Conf. Automata. Lang. Prog.*, Lect. Notes Comp. Sci. (1986), 123-135.

[HaImSe 83] J. Hartmanis, N. Immerman, V. Sewelson, "Sparse sets in $NP - P$: *EXPTIME* versus *NEXPTIME*," *Proc.* 15^{th} *Symp. Theor. Computing*, (1983), 382-391.

[He 87] L.A. Hemachandra, "The strong exponential hierarchy collapses," *Proc.* 19^{th} *Symp. Theor. Computing*, (1987), 110-122.

[Hay 78] L. Hay, "Convex subsets of 2^n and bounded truth-table reducibility," *Discr. Math.*, 21 (1978), 31-46.

[HiZa 84] P. Hinman and S. Zachos, "Probabilistic machines, oracles and quantifiers," *Proc. Recursion Theory Week*, Lect. Notes Math. 1141 (1984), 159-192.

[Im 88] N. Immerman, "Nondeterministic space is closed under complement," *Proc.* 3^{rd} *Conf. on Struct. Compl. Theory*, (1988).

[Ka 88] J. Kadin, "The polynomial time hierarchy collapses if the Boolean hierarchy collapses," *Proc.* 3^{rd} *Conf. on Struct. Compl. Theory*, (1988), 278-292.

[Ka 87] J. Kadin, "$P^{NP}[\log\ n]$ and sparse Turing complete sets for *NP*," *Proc.* 2^{nd} *Conf. Struct. Compl. Theory*, (1987), 33-40.

[Kal 88] E. Kaltofen, "Greatest common divisors of polynomials given by straight line programs," *J. Assoc. Comp. Mach.*, 35, 1 (1988), 231-264.

[KaLi 80] R.M. Karp, R.J. Lipton, "Some connections between nonuniform and uniform complexity," *Proc.* 12^{th} *Symp. Theor. Computing*, (1980), 302-309.

[Ko 82] K-I. Ko, "Some observations on probabilistic algorithms and *NP* -hard problems," *Inf. Proc. Let.,* 14 (1982), 39-43.

[Ko 88] K-I. Ko, "Distinguishing bounded reducibilities by sparse sets," *Proc.* 3^{rd} *Conf. Struct. Compl. Theory,* (1988), 181-192.

[KöScWa 87] J. Köbler, U. Schöning and K.W. Wagner, "The difference and truth-table hierarchies for *NP*," *RAIRO,* 21 (1987), 419-435.

[Kr 86] M.W. Krentel, "The complexity of optimization problems," *Proc.* 18^{th} *Symp. Theor. Computing,* (1986), 69-76.

[Lac 65] A.H. Lachlan, "Some notions of reducibility and productiveness," *Zeitsch. Math. Logik. Grundlag. Math.,* 11 (1965), 17-44.

[La 83] C. Lautemann, "*BPP* and the polynomial hierarchy," *Inf. Proc. Let.,* 17 (1983), 215-217.

[LaJeKi 87] K.J. Lange, B. Jenner, B. Kirsig, "The logarithmic alternating hierarchy collapses: etc," *Proc.* 14^{th} *Intl. Conf. Automata. Lang. Prog.,* Lect. Notes Comp. Sci. 267 (1987) 531-541.

[LaLySe 75] R.E. Ladner, N.A. Lynch, A.L. Selman, "A comparison of polynomial time reducibilities," *Theor. Comp. Sci.,* 1 (1975), 103-123.

[LeMo 81] E.W. Leggett, D.J. Moore, "Optimization problems and the polynomial hierarchy," *Theor. Comp. Sci.,* 15 (1981), 279-289.

[Lo 85] T.J. Long, "On restricting the size of oracles when compared to restricting access to oracles," *SIAM J. Comput.,* 14 (1985), 585-597.

[Ma 82] S. Mahaney, "Sparse complete sets for *NP* : solution to a conjecture of Berman and Hartmanis," *J. Comput. Sys. Sci.,* 25 (1982), 130-143.

[OgWa 90] M. Ogiwara, O. Watanabe, "On polynomial-time bounded truthtable reducibility of *NP* sets to sparse sets.," *Proc.* 22^{nd} *Symp. Theor.*

Computing, (1990), 457-467.

[Ow] J.C. Owings, "A cardinality version of Beigel's nonspeedup theorem," *J. Sym. Logic,* To appear.

[Pa 82] C.H. Papadimitriou, "On the complexity of unique solutions," *Proc.* 23^{rd} *Found. of Comput. Symp.,* (1982) 14-20.

[PaWo 85] C.H. Papadimitriou, D. Wolfe, "The complexity of facets resolved," *Proc.* 26^{th} *Found. of Comput. Symp.,* (1985) 74-78.

[PaYa 82] C.H. Papadimitriou, M. Yannakakis, "The complexity of facets and some facets of complexity," *Proc.* 23^{rd} *Found. of Comput. Symp.,* (1982).

[PaZa 82] C.H. Papadimitriou, S. Zachos, "Two remarks on the power of counting," *Proc.* 6^{th} *GI Conf. on Theor. Comp. Sci.,* Lect. Notes Comp. Sci. 145 (1983) 269-276.

[Pu 65] H. Putnam, "Trial and error predicates and a solution to a problem of Mostowski," *J. Sym. Logic,* 30 (1965), 49-57.

[Ra 80] M. Rabin, "Probabilistic algorithm for testing primality," *J. Number Theory,* 12 (1980), 128-138.

[RaSh 88] M. Rabin and J. Shallit, "Randomized algorithms in number theory," *Comm. Pure Appl. Math.,* 39, (1986), S239-S256.

[Rac 82] C. Rackoff, "Relativized questions involving probabilistic algorithms," *JACM,* 29 (1982), 261-268.

[Ro 67] H. Rogers, "Theory of recursive functions and effective computability," *McGraw-Hill, New York.*

[Sc 82] U. Schöning, "A uniform approach to obtain diagonal sets in complexity classes," *Theor. Comp. Sci.,* 18 (1982) 95-103.

[**Sc**] U. Schöning, "Complexity and structure," *LNCS* 211, (Goos, Hartmanis ed.,) Springer-Verlag.

[**Sc 83**] U. Schöning, "A low and a high hierarchy in *NP*," *J. Comput. Sys. Sci.*, 27 (1983) 14-28.

[**ScWa 87**] U. Schöning, K.W. Wagner, "Collapsing oracle hierarchies, census functions and logarithmically many queries," *Tech. Report, Inst. of Math., Univ. of Augsburg,* 140 (1987).

[**Sch 80**] J. Schwartz, "Fast probabilistic algorithms for the verification of polynomial identities," *JACM,* 27 (1980), 701-717.

[**Si 83**] M. Sipser, "A complexity theoretic approach to randomness," *Proc.* 15^{th} *Symp. Theor. Computing,* (1983), 330-335.

[**SoSt 77**] R. Solovay and V. Strassen, "A fast Monte-Carlo test for primality," *SIAM J. Comput* 6 (1977), 84-85; [Erratum: 7 (1978), 118].

[**Sz 87**] R. Szelepcsényi, "The method for forcing nondeterministic automata," *Bull. EATCS,* 33 (1987), 96-99.

[**To 87**] S. Toda, "$\Sigma_2^{SPACE(n)}$ is closed under complement," *J. Comput. Sys. Sci.*, 35 (1987), 145-152.

[**Uk 83**] E. Ukkonen, "Two results on polynomial time truth-table reductions to sparse sets," *SIAM J. Comput.,* 12 (1983), 580-590.

[**Wa 86**] K.W. Wagner, "More complicated questions about maxima and minima, and some closure properties of *NP*," *Intl. Conf. Automata. Lang. Prog.,* Lect. Notes Comp. Sci., 226 (1986), 434-443.

[**Wa 87a**] K.W. Wagner, "Bounded query classes," *Tech. Report, Institute of Mathematics, University of Augsburg,* 157 (1987), 1-23; also "On restricting the access to an *NP* oracle," *Proc* 15^{th} *Intl. Conf. Automata. Lang. Prog.*,

(1988).

[Wa 87b] K.W. Wagner, "Number-of-query hierarchies," *Tech. Report, Institute of Mathematics, University of Augsburg,* 158 (1987), 1-22.

[Wa 87c] K.W. Wagner, "Log-query classes," *Tech. Report, Institute of Mathematics, University of Augsburg,* 145, (1987).

[Wa 88] K.W. Wagner, "Bounded query computations," *Proc.* 3^{rd} *Conf. on Struct. Compl. Theory,* (1988), 260-277.

[Wat 88] O. Watanabe, "On $\leq^P_{1-tt}$-sparseness and nondeterministic complexity classes," *Proc* 15^{th} *Intl. Conf. Automata. Lang. Prog.*, (1988).

[WeWa 85] G. Wechsung and K.W. Wagner, "On the Boolean closure of *NP*," *Proc. Conf. Fundament. Comput. Theory,* Lect. Notes Comp. Sci. 199 (1985), 485-493.

[Ya 83] C. Yap, "Some consequences on non-uniform conditions on uniform classes," *Theor. Comp. Sci.,* 26 (1983) 283-300.

[Ye 83] Y. Yesha, "On certain polynomial time truth-table reducibilities of complete sets to sparse sets," *SIAM J. Comput.,* 12 (1983), 411-425.

[ZaHe 84] S. Zachos and H. Heller, "A decisive characterization of *BPP*," *Information and Control,* 69 (1986), 125-135.

[Za 86] S. Zachos, "Probabilistic quantifiers, adversaries and complexity classes: an overview," *Proc.* 1^{st} *Conf. Struct. Compl. Theory,* (1986), 383-400.