INFORMATIQUE THÉORIQUE ET APPLICATIONS

JOHANNES KÖBLER
UWE SCHÖNING
KLAUS W. WAGNER

The difference and truth-table hierarchies for NP

Informatique théorique et applications, tome 21, n° 4 (1987), p. 419-435.

<http://www.numdam.org/item?id=ITA_1987__21_4_419_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 21, n° 4, 1987, p. 419 à 435)

THE DIFFERENCE AND TRUTH-TABLE HIERARCHIES FOR NP(*)

by Johannes KÖBLER (1), Uwe SCHÖNING (2) and Klaus W. WAGNER (3)

Communicated by K. MEHLHORN

Abstract. – *Two hierarchies of complexity classes are defined. Both, the difference hierarchy and the truth-table hierarchy for* **NP** *are located between* **NP** *as bottom class and* $\Delta_2^{\mathbf{P}}$. *We give examples for complete sets in both hierarchies and investigate their interrelationships. It turns out that the "ω-jump" of the truth-table hierarchy depends on the encodings of Boolean functions that we use.*

Résumé. – *Nous définissons deux hiérarchies des classes de complexité. La hiérarchie de différences et la hiérarchie du tableau de vérité pour* **NP** *sont localisés entre* **NP** *comme la classe de base et* $\Delta_2^{\mathbf{P}}$. *Nous présentons des exemples d'ensembles complets dans les deux hiérarchies, et nous examinons leurs corrélations. Il apparait que le « ω-bond » de la hiérarchie du tableau de vérité dépend du chiffrement utilisé des formules de Boole.*

1. INTRODUCTION

The class $\Delta_2^{\mathbf{P}} - \mathbf{NP}$ has received a certain current research interest since sets in this class arise from many **NP**-complete problems by imposing certain uniqueness or optimality conditions (*see* [1, 6, 7, 8, 13]). For example, consider the well known **NP**-complete set CLIQUE, i.e. the set of pairs (G, k) such that G is an undirected graph containing a clique of (at least) k vertices. Then MAXCLIQUE – the problem of deciding whether k is the maximum clique size of G – is in $\Delta_2^{\mathbf{P}} - \mathbf{NP}$, provided $\mathbf{NP} \neq \mathbf{coNP}$ (*see* [9]). In [9] also other **NP**-complete problems are described where this phenomenon occurs.

(*) Received in June 1986.
This paper is a revised and extended version of the diploma thesis of the first author (*see* [6]).

(1) Institut für Informatik, Universität Stuttgart, Azenbergstraße 12, D-7000 Stuttgart.
(2) Erziehungswissenschaftliche Hochschule Rheinland-Pfalz, Abteilung Koblenz, Seminar für Informatik, Rheinau 3-4, D-5400 Koblenz.
(3) Institut für Mathematik, Universität Augsburg, Memminger Straße 6, D-8900 Augsburg.

Papadimitriou and Yannakakis [11] introduced a new class $\mathbf{D}^{\mathbf{P}}$ located between $\mathbf{NP}$ and $\Delta_2^{\mathbf{P}}$ which seems appropriate to give an exact characterization for some of the problems occuring in $\Delta_2^{\mathbf{P}} - \mathbf{NP}$, since some of those problems, like MAXCLIQUE, turn out to be $\mathbf{D}^{\mathbf{P}}$-complete. In [10], Papadimitriou also presents some similar new problems that are even complete for $\Delta_2^{\mathbf{P}}$. In [3], [6] and [14] hierarchies have been introduced and investigated independently which are located between $\mathbf{NP}$ as bottom class and $\Delta_2^{\mathbf{P}}$. The union of each of these hierarchies is the Boolean closure of $\mathbf{NP}$.

In the present paper which should be considered as a revised and extended version of [6] we show that all these hierarchies either coincide or are very closely related to each other. For each level of these hierarchies we give examples of complete sets, and we present conditions causing these hierarchies to "collapse".

Finally, we study the complexities of the "ω-jumps" of these hierarchies, and we obtain the somewhat surprising result that the answer depends heavily on the succinctness of the encodings of the Boolean functions that we use. If, for example, the Boolean functions are represented by Boolean circuits then it is very likely that the "ω-jump" has a higher complexity than in the case when the Boolean functions are represented by Boolean formulas.

2. NOTATION

All our sets are languages over some fixed alphabet Σ, say $\Sigma = \{0, 1\}$. For a string $w \in \Sigma^*$, $|w|$ denotes its length, and for a set $A \subseteq \Sigma^*$, $|A|$ denotes its cardinality. The symmetric difference of two sets A and B is defined by $A \Delta B = (A - B) \cup (B - A)$. For a set $A \subseteq \Sigma^*$, let $\bar{A} = \Sigma^* - A$ be its complement, and for classes $\mathbf{A}$ and $\mathbf{B}$ of sets let $\mathbf{co}\,\mathbf{A} = \{\bar{A} : A \in \mathbf{A}\}$, $\mathbf{A} \wedge \mathbf{B} = \{A \cap B : A \in \mathbf{A}$ and $B \in \mathbf{B}\}$, $\mathbf{A} \vee \mathbf{B} = \{A \cup B : A \in \mathbf{A}$ and $B \in \mathbf{B}\}$, and let $\mathbf{BA}(\mathbf{A})$ denote the Boolean algebra generated by $\mathbf{A}$, i.e. the smallest class that contains $\mathbf{A}$ and is closed under union, intersection and complementation.

Our model of computation is the multi-tape Turing machine. For a Turing machine M, let $L(M)$ denote the set accepted by M, and for an oracle Turing machine and an oracle set A, let $L(M, A)$ be the set accepted by M when using the oracle A. Let $\mathbf{P}$ ($\mathbf{NP}$) be the class of sets acceptable by deterministic (nondeterministic) polynomial-time bounded Turing machines. A Turing machine is said to be polynomial-time bounded if there is a polynomial p such that each computation of M on inputs of size n has length at most $p(n)$. Similarly, let $\mathbf{P}^A$ ($\mathbf{NP}^A$) be the class of sets acceptable by deterministic (nondeterministic) polynomial-time bounded oracle machines when using the

oracle set A. The class $\bigcup\{\mathbf{P}^A: A\in\mathbf{NP}\}$ is commonly called $\mathbf{\Delta_2^P}$ due to its membership in a more general structure, the polynomial-time hierarchy (*see* [13]). If $A\in\mathbf{P}^B$ then we also say that A is polynomial-time Turing reducible to B (for short: $A\leqq_T^{\mathbf{P}} B$). A more restrictive but more commonly used reducibility is the polynomial-time many-one reducibility: A is polynomial-time many-one reducible to B (for short $A\leqq_m^{\mathbf{P}} B$) if there is a function f: $\Sigma^*\to\Sigma^*$ computable in polynomial time such that $f^{-1}(B)=A$. For a set A and a class $\mathbf{C}$ of sets, A is called $\mathbf{C}$-hard (w. r. t. $\leqq_m^{\mathbf{P}}$) if $B\leqq_m^{\mathbf{P}} A$ for each $B\in\mathbf{C}$. Further, A is called $\mathbf{C}$-complete if A is $\mathbf{C}$-hard and $A\in\mathbf{C}$. A well known **NP**-complete set is SAT, the set of (encodings of) satisfiable Boolean formulas.

Finally, for a set A let c_A: $\Sigma^*\to\{0, 1\}$ denote its characteristic function.

3. THE DIFFERENCE HIERARCHY

In this section we introduce and investigate a hierarchy of classes located between **NP** as the bottom class and $\mathbf{\Delta_2^P}=\bigcup\{\mathbf{P}^A: A\in\mathbf{NP}\}$. The definition is based on the symmetric difference operation and is motivated from the definition of the "*n-r. e.*" sets in recursive function theory (*see* [12], p. 67). The second level of the difference hierarchy turns out to be Papadimitriou and Yannakakis' class $\mathbf{D^P}$ [11], and the union of the difference hierarchy is the Boolean algebra generated by the sets in **NP**. We show that the difference hierarchy essentially coincides with the Boolean **NP**-hierarchy introduced in [14] and with the Boolean hierarchy introduced in [3]. Thus this hierarchy has been introduced and investigated independently in [3], [6] and [14]. Further, we define complete sets for each level of this hierarchy and study conditions under which the hierarchy "collapses", i. e. it consists only of finitely many levels.

DEFINITION 3.1: The *difference hierarchy* (for **NP**) is the sequence of classes $\{\mathbf{DIFF_k}\}_{k\geqq 1}$ where $\mathbf{DIFF_1}=\mathbf{NP}$ and $\mathbf{DIFF_{k+1}}=\{A\,\Delta B: A\in\mathbf{DIFF_k}$ and $B\in\mathbf{NP}\}$ for each $k\geqq 1$. Further, let $\mathbf{DH}=\mathbf{DIFF_1}\cup\mathbf{DIFF_2}\cup\mathbf{DIFF_3}\cup\ldots$.

Clearly, we have $\mathbf{NP}=\mathbf{DIFF_1}\subseteq\mathbf{DIFF_2}\subseteq\ldots\subseteq\mathbf{DH}\subseteq\mathbf{\Delta_2^P}$. However, it is not known whether any of these inclusions is strict. If any of these inclusions is strict then $\mathbf{NP}\neq\mathbf{co\,NP}$. Next we summarize some elementary properties of this hierarchy.

PROPOSITION 3.2: *For every* $k \geq 1$,

(i) $\mathbf{DIFF_k} \cup \mathbf{co\,DIFF_k} \subseteq \mathbf{DIFF_{k+1}} \cap \mathbf{co\,DIFF_{k+1}}$.

(ii) $\mathbf{DIFF_k}$ *is closed under polynomial-time many-one reducibility.*

(iii) $\mathbf{DIFF_k} = \mathbf{DIFF_{k+1}}$ *implies* $\mathbf{DIFF_k} = \mathbf{DH}$.

Proof: (i) Trivial because of $\overline{A \Delta B} = \bar{A} \Delta B$, $A = A \Delta \varnothing$ and $\bar{A} = A \Delta \Sigma^*$.

(ii) Suppose $A \leq_m^{\mathbf{P}} B$ via some polynomial-time computable function f, and suppose that $B \in \mathbf{DIFF_k}$. then there exist sets $C_1, \ldots, C_k$ in **NP** such that $B = C_1 \Delta \ldots \Delta C_k$. Now we have

$$A = f^{-1}(B) = f^{-1}(C_1 \Delta \ldots \Delta C_k) = f^{-1}(C_1) \Delta \ldots \Delta f^{-1}(C_k).$$

Since $f^{-1}(C_i) \in \mathbf{NP}$ for all $i = 1, \ldots, k$, we have $A \in \mathbf{DIFF_k}$.

(iii) By induction on n it can easily be seen that $\mathbf{DIFF_{k+n}} = \mathbf{DIFF_k}$ for each $n \geq 1$. ■

It turns out that the difference hierarchy essentially coincides with the Boolean **NP**-hierarchy introduced in [14] as the sequence of classes $\{\mathbf{C_k^{NP}}, \mathbf{D_k^{NP}}\}_{k \geq 1}$ and with the Boolean hierarchy introduced in [3] as the sequence of classes $\{\mathbf{NP(k)}\}_{k \geq 1}$.

DEFINITION 3.3: (i) [14] For every $k \geq 1$,

$$\mathbf{C_{2k-1}^{NP}} = \mathbf{co\,NP} \vee \bigvee_{k-1} (\mathbf{NP} \wedge \mathbf{co\,NP}), \qquad \mathbf{D_{2k-1}^{NP}} = \mathbf{NP} \vee \bigvee_{k-1} (\mathbf{NP} \wedge \mathbf{co\,NP})$$

$$\mathbf{C_{2k}^{NP}} = \bigvee_{k} (\mathbf{NP} \wedge \mathbf{co\,NP}), \qquad \mathbf{D_{2k}^{NP}} = \mathbf{NP} \vee \mathbf{co\,NP} \vee \bigvee_{k-1} (\mathbf{NP} \wedge \mathbf{co\,NP})$$

(ii) [3] $\mathbf{NP(1)} = \mathbf{NP}$ and, for every $k \geq 1$,

$$\mathbf{NP(2k)} = \mathbf{NP(2k-1)} \wedge \mathbf{co\,NP}, \quad \mathbf{NP(2k+1)} = \mathbf{NP(2k)} \vee \mathbf{NP}.$$

To prove the relationships between the Boolean **NP**-hierarchy, the difference hierarchy and the Boolean hierarchy we need the following definition.

DEFINITION 3.4: *Let* f be a k-ary Boolean function. We define: $A \in f(\mathbf{NP})$ iff there exist $B_1, \ldots, B_k \in \mathbf{NP}$ such that $c_A(x) = f(c_{B_1}(x), \ldots, c_{B_k}(x))$ for all $x \in \Sigma^*$.

It has been shown that for every Boolean function f the class $f(\mathbf{NP})$ coincides with one of the classes $\mathbf{C_k^{NP}}$ or $\mathbf{D_k^{NP}}$. To answer the question of which $\mathbf{C_k^{NP}}$ or $\mathbf{D_k^{NP}}$ is equal to $f(\mathbf{NP})$ for a given Boolean function f the non-numerical measure c for Boolean functions has been introduced. Let $\sqsubseteq$ be the initial word relation, i. e. $u \sqsubseteq w$ iff there exists a v such that $uv = w$, for

u, $w \in \Sigma^*$. Further, for $a_1, \ldots, a_k, b_1, \ldots, b_k \in \{0, 1\}$ we define $(a_1, \ldots, a_k) \leqq (b_1, \ldots, b_k)$ iff $a_1 \leqq b_1, \ldots, a_k \leqq b_k$. Finally, set $\underline{a}_i = (a_{i\,1}, \ldots, a_{ik})$, and define $c(f) = \max \sqsubseteq \{f(\underline{a}_1) f(\underline{a}_2) \ldots f(\underline{a}_r) : r \geqq 1, f(\underline{a}_i) \neq f(\underline{a}_{i+1})$ and $\underline{a}_1 \leqq \underline{a}_2 \leqq \ldots \leqq \underline{a}_r\}$ (it is obvious that this maximum exists) and $m(f) = |c(f)| - 1$.

THEOREM 3.5 [14]: *For every Boolean function f,*

$$f(\mathbf{NP}) = \begin{cases} \mathbf{C}^{\mathbf{NP}}_{m(f)} & \text{if } c(f) \text{ ends with 0} \\ \mathbf{D}^{\mathbf{NP}}_{m(f)} & \text{if } c(f) \text{ ends with 1.} \end{cases} \blacksquare$$

Using the above theorem it is not hard to prove.

THEOREM 3.6 [14]: *For every* $k \geqq 1$,

(i) $\mathbf{C}^{\mathbf{NP}}_{k+1} = \mathbf{D}^{\mathbf{NP}}_k \wedge \mathbf{co\,NP}$,

(ii) $\mathbf{D}^{\mathbf{NP}}_{k+1} = \mathbf{C}^{\mathbf{NP}}_k \vee \mathbf{NP}$,

(iii) $\mathbf{C}^{\mathbf{NP}}_k = \mathbf{co}\,\mathbf{D}^{\mathbf{NP}}_k$,

(iv) $\mathbf{C}^{\mathbf{NP}}_k \cup \mathbf{D}^{\mathbf{NP}}_k \subseteq \mathbf{C}^{\mathbf{NP}}_{k+1} \cap \mathbf{D}^{\mathbf{NP}}_{k+1}$,

(v) $\bigcup_{k \geqq 1} \mathbf{C}^{\mathbf{NP}}_k = \bigcup_{k \geqq 1} \mathbf{D}^{\mathbf{NP}}_k = \mathbf{BA(NP)}$. $\blacksquare$

Note that the above results are proved in [14] not only for **NP** but also for all classes which are closed under union and intersection.

From Theorem 3.6 (i) and (ii) we obtain immediately the relationships between the Boolean **NP**-hierarchy from [14] and the Boolean hierarchy from [3].

COROLLARY 3.7: *For every* $k \geqq 1$,

(i) $\mathbf{NP(2k-1)} = \mathbf{D}^{\mathbf{NP}}_{2k-1}$,

(ii) $\mathbf{NP(2k)} = \mathbf{C}^{\mathbf{NP}}_{2k}$. $\blacksquare$

Theorem 3.5 enables us to establish the result on the relationship between the difference hierarchy and the Boolean **NP**-hierarchy.

THEOREM 3.8: *For every* $k \geqq 1$,

(i) $\mathbf{DIFF}_{2k-1} = \mathbf{D}^{\mathbf{NP}}_{2k-1}$,

(ii) $\mathbf{DIFF}_{2k} = \mathbf{C}^{\mathbf{NP}}_{2k}$.

Proof: Obviously, $\mathbf{DIFF}_k = \mathbf{d}_k\mathbf{(NP)}$ where

$$d_k(x_1, x_2, \ldots, x_k) = x_1 \oplus x_2 \oplus \ldots \oplus x_k,$$

where $\oplus$ stands for addition modulo 2. Furthermore, because of

$$d_k(0,0,0,\ldots,0)\, d_k(1,0,0,\ldots,0)\, d_k(1,1,0,\ldots,0) \ldots d_k(1,1,1,\ldots,1) = 0101\ldots$$

and since

$$(0,0,0,\ldots,0)\leqq(1,0,0,\ldots,0)\leqq(1,1,0,\ldots,0)\leqq\ldots\leqq(1,1,1,\ldots,1)$$

is a chain of maximum length we have $c(d_k)=0101\ldots$ By Theorem 3.5 we have, for $k\geqq 1$,

$$\mathbf{d_{2k-1}(NP)}=\mathbf{D^{NP}_{2k-1}} \qquad \text{and} \qquad \mathbf{d_{2k}(NP)}=\mathbf{C^{NP}_{2k}}. \blacksquare$$

COROLLARY 3.9: (i) *For every* $k\geqq 1$, $\mathbf{DIFF_k}=\mathbf{NP(k)}$.
(ii) *The sequence* $\{\mathbf{DIFF_k}, \mathbf{co\,DIFF_k}\}_{k\geqq 1}$ *is the Boolean* **NP***-hierarchy.*
(iii) $\mathbf{DH}=\mathbf{BA(NP)}$.
(iv) *For every* $k\geqq 1$, $\mathbf{DIFF_k}=\mathbf{co\,DIFF_k}$ *implies* $\mathbf{DIFF_k}=\mathbf{DH}$.

Proof: Only (iv) deserves a proof. Assume $\mathbf{DIFF_k}=\mathbf{co\,DIFF_k}$. Let $k=2m$. We conclude

$$\mathbf{co\,DIFF_{k+1}}=\mathbf{co\,DIFF_{2m+1}}=\mathbf{C^{NP}_{2m+1}}=\bigvee_m(\mathbf{NP}\wedge\mathbf{co\,NP})\vee\mathbf{co\,NP}=\mathbf{C^{NP}_{2m}}\vee\mathbf{co\,NP}$$

$$=\mathbf{DIFF_k}\vee\mathbf{co\,NP}=\mathbf{co\,DIFF_k}\vee\mathbf{co\,NP}=\mathbf{D^{NP}_{2m}}\vee\mathbf{co\,NP}$$

$$=\bigvee_{m-1}(\mathbf{NP}\wedge\mathbf{co\,NP})\vee\mathbf{NP}\vee\mathbf{co\,NP}\vee\mathbf{co\,NP}=\mathbf{D^{NP}_{2m}}=\mathbf{co\,DIFF_k}.$$

Consequently, $\mathbf{DIFF_k}=\mathbf{DIFF_{k+1}}$, and by Proposition 3.2 (iii) we have $\mathbf{DIFF_k}=\mathbf{DH}$. The case $k=2m+1$ is treated analogously. ■

Now we consider sets which are complete for every level of the difference hierarchy.

DEFINITION 3.10: Let L be an arbitrary language and let $k\geqq 1$. Define

$$L^{k-\Delta}=\{(x_1,\ldots,x_k):|\{i:x_i\in L\}|\text{ is odd}\} \qquad \text{and} \qquad L^{\omega-\Delta}=\bigcup_{k\geqq 1}L^{k-\Delta}.$$

THEOREM 3.11: (i) *For each* $k\geqq 1$, $\mathrm{SAT}^{k-\Delta}$ *is* $\mathbf{DIFF_k}$*-complete.*
(ii) $\mathrm{SAT}^{\omega-\Delta}$ *is* **DH**-hard.

Proof: (i) First note that

$$\mathrm{SAT}^{k-\Delta}=\Delta_{i=1}^{k}(\Sigma^*\times\Sigma^*\times\ldots\times\Sigma^*\times\mathrm{SAT}\times\Sigma^*\times\ldots\times\Sigma^*)$$

where SAT is occuring at position i.

Since all the sets in parenthesis are in **NP** it follows that $\mathrm{SAT}^{k-\Delta}$ is in $\mathbf{DIFF_k}$. Now let A be in $\mathbf{DIFF_k}$. Then there exist sets $A_1,\ldots,A_k$ in **NP** and

polynomial-time computable functions $f_1, \ldots, f_k$ such that $A = A_1 \Delta \ldots \Delta A_k$ and $f_i^{-1}(\mathrm{SAT}) = A_i$ for $i = 1, \ldots, k$. Consequently,

$$\begin{aligned} x \in A \text{ iff } & x \in A_1 \Delta \ldots \Delta A_k \\ \text{iff } & \left| \{i : x \in A_i\} \right| \text{ is odd} \\ \text{iff } & \left| \{i : f_i(x) \in \mathrm{SAT}\} \right| \text{ is odd} \\ \text{iff } & (f_1(x), \ldots, f_k(x)) \in \mathrm{SAT}^{k-\Delta}. \end{aligned}$$

(ii) follows from (i). ■

Similar $\mathbf{DIFF_k}$-complete sets have been considered independently in [3]. Note that $\mathrm{SAT}^{2-\Delta}$ is essentially the set SAT-UNSAT which has been defined and shown to be $\mathbf{D^P}$-complete in [11]. In that paper also some other $\mathbf{D^P}$-complete sets have been exhibited. In [16] several natural problems are shown to be complete for every level of the Boolean **NP**-hierarchy (and thus also for every level of the difference hierarchy). For example, the set $\{(G, n_1, \ldots, n_k)$: the chromatic number of the graph G is in $\{n_1, \ldots, n_k\}\}$ is complete in $\mathbf{C_{2k}^{NP}} = \mathbf{co\,DIFF_{2k}}$.

It is not likely that $\mathrm{SAT}^{\omega-\Delta}$ is in **DH**, and therefore **DH**-complete. Moreover, it is even not likely that **DH** has any complete set. We consider $\mathrm{SAT}^{\omega-\Delta}$ again in Section 5.

COROLLARY 3.12: *The difference hierarchy is finite if and only if* **DH** *has a complete set.*

Proof: The forward direction follows from Theorem 3.11 (i). For the backward direction let A be a **DH**-complete set. Then $A \in \mathbf{DIFF_k}$ for some $k \geq 1$, and by Proposition 3.2 (ii) we have $\mathbf{DH} = \mathbf{DIFF_k}$, hence the difference hierarchy is finite. ■

4. THE TRUTH-TABLE HIERARCHY

In what follows we use a fixed natural, uncontrieved encoding of all Boolean circuits with $\wedge$, $\vee$ and $\neg$ gates. If z is such a encoding of a Boolean circuit then h_z denotes the Boolean function realized by this circuit.

From several equivalent definitions for the polynomial-time truth-table reducibility given in [8] and [2] we choose the following.

DEFINITION 4.1: A set A is polynomial-time truth-table reducible to a set B ($A \leq_{tt}^{\mathbf{P}} B$) iff there exists a polynomial-time computable function f such that $c_A(x) = h_z(c_B(x_1), \ldots, c_B(x_m))$ for all $x \in \Sigma^*$ where $f(x) = (z, x_1, \ldots, x_m)$ and

z is the encoding of a Boolean circuit with $\wedge$, $\vee$ and $\neg$ gates. Note that m depends on x.

(ii) A set A is polynomial-time k-bounded truth-table reducible to B ($A \leqq^{\mathbf{P}}_{k-tt} B$) iff $A \leqq^{\mathbf{P}}_{tt} B$ via a function f having at most $(k+1)$-tuples as values. Equivalently, $A \leqq^{\mathbf{P}}_{k-tt} B$ iff there exist polynomial-time computable functions $g, f_1, \ldots, f_k$ such that $c_A(x) = h_{g(x)}(c_B(f_1(x)), \ldots, c_B(f_k(x)))$ for all $x \in \Sigma^*$ where $g(x)$ is the encoding of a Boolean circuit with $\wedge$, $\vee$ and $\neg$ gates.

(iii) A set A is polynomial-time bounded truth-table reducible to B ($A \leqq^{\mathbf{P}}_{btt} B$) iff $A \leqq^{\mathbf{P}}_{k-tt} B$ for some $k \geqq 1$.

(iv) For a class of sets $\mathbf{C}$ and a reducibility α let

$$\alpha(\mathbf{C}) = \{A: \text{there exists a set } B \in \mathbf{C} \text{ such that } A \alpha B\}.$$

Note that in [2] a different terminology is used and $\leqq^{\mathbf{P}}_{tt}$ is denoted by $\leqq^{\text{P.UNIV.ALL}}$

In the preceding definition of $\leqq^{\mathbf{P}}_{tt}$, $\leqq^{\mathbf{P}}_{k-tt}$ ($k \geqq 1$) and $\leqq^{\mathbf{P}}_{btt}$ Boolean functions are represented by Boolean circuits. We also consider the reducibilities $\leqq^{\mathbf{P}}_{bf}$, $\leqq^{\mathbf{P}}_{k-bf}$ ($k \geqq 1$) and $\leqq^{\mathbf{P}}_{bbf}$ ($\leqq^{\mathbf{P}}_{ftt}$, $\leqq^{\mathbf{P}}_{k-ftt}$ ($k \geqq 1$) and $\leqq^{\mathbf{P}}_{bftt}$) which are defined in the same way as $\leqq^{\mathbf{P}}_{tt}$, $\leqq^{\mathbf{P}}_{k-tt}$ ($k \geqq 1$) and $\leqq^{\mathbf{P}}_{btt}$ but using Boolean formulas with operations $\wedge$, $\vee$ and $\neg$ (full truth-tables) instead of Boolean circuits with $\wedge$, $\vee$ and $\neg$ gates. The following relationships between these reducibilities can easily be proved.

PROPOSITION 4.2: *Let A, B be arbitrary sets, and let $k \geqq 1$.*

(i) *$A \leqq^{\mathbf{P}}_{ftt} B$ implies $A \leqq^{\mathbf{P}}_{bf} B$, and $A \leqq^{\mathbf{P}}_{bf} B$ implies $A \leqq^{\mathbf{P}}_{tt} B$. Consequently,*

$$\leqq^{\mathbf{P}}_{ftt}(\mathbf{NP}) \subseteq \leqq^{\mathbf{P}}_{bf}(\mathbf{NP}) \subseteq \leqq^{\mathbf{P}}_{tt}(\mathbf{NP})(^1).$$

(ii) *$A \leqq^{\mathbf{P}}_{k-ftt} B$ iff $A \leqq^{\mathbf{P}}_{k-bf} B$ iff $A \leqq^{\mathbf{P}}_{k-tt} B$. Consequently,*

$$\leqq^{\mathbf{P}}_{k-ftt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{k-bf}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}).$$

---

(1) *Note added in proof*: Though we had conjectured that $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP}) \neq \leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$ it turned out recently that these classes coincide. *In* L. A. HEMACHANDRA, *The Strong Exponential Hierarchy Collapses*, Proc. 19th A.C.M. S.T.O.C., 1987, it is proved that $\leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$ is included in the class $\mathbf{P}^{\mathbf{NP}}[O(\log n)]$ of all sets which ave polynomial-time computable with $O(\log n)$ queries to an NP-oracle. But the classes $\mathbf{P}^{\mathbf{NP}}[O(\log n)]$ and $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$ coincide because they hare the common complete set ODD CLIQUE. The latter is proved in [16] and M. W. KRENTEL, *The Complexity of Optimization Problems*, Proc. 18th A.C.M. S.T.O.C., 1986, pp. 69-76. For a further proof of the coincidence of $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$ and $\leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$ see the very recent paper R. J. BEIGEL, *Bounded Queries to SAT and the Boolean Hierarchy*, manuscript, 1987.

(iii) $A \leqq^{\mathbf{P}}_{bftt} B$ *iff* $A \leqq^{\mathbf{P}}_{bbf} B$ *iff* $A \leqq^{\mathbf{P}}_{btt} B$. *Consequently,*

$$\leqq^{\mathbf{P}}_{bftt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{bbf}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{btt}(\mathbf{NP}).$$

Proof([1]) : Every full truth-table can be converted in polynomial time into an equivalent Boolean formula, and every Boolean formula can be converted in polynomial time into an equivalent Boolean circuit. The converse is also true if the number of Boolean variables or the number of input nodes of the Boolean circuits, resp., is bounded to a fixed number $k \geqq 1$. Of course, the polynomial time bound depends on k. ■

Clearly, for every class $\mathbf{C}$ of languages we have

$$\mathbf{C} \subseteq \leqq^{\mathbf{P}}_{1-tt}(\mathbf{C}) \subseteq \leqq^{\mathbf{P}}_{2-tt}(\mathbf{C}) \subseteq \ldots \subseteq \leqq^{\mathbf{P}}_{btt}(\mathbf{C}) \subseteq \leqq^{\mathbf{P}}_{ftt}(\mathbf{C}) \subseteq \leqq^{\mathbf{P}}_{bf}(\mathbf{C}) \subseteq \leqq^{\mathbf{P}}_{tt}(\mathbf{C}) \subseteq \leqq^{\mathbf{P}}_{T}(\mathbf{C}).$$

We call the sequence $\mathbf{NP}$, $\leqq^{\mathbf{P}}_{1-tt}(\mathbf{NP})$, $\leqq^{\mathbf{P}}_{2-tt}(\mathbf{NP}), \ldots$ the (bounded) *truth-table hierarchy* for $\mathbf{NP}$.

PROPOSITION 4.3: *For every* $k \geqq 1$,

(i) $\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \mathbf{co} \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$.

(ii) $\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$ *is closed under polynomial-time many-one reducibility.*

Proof: (i) is obvious; (ii) is proved as Proposition 3.2 (ii). ■

Like the difference hierarchy for $\mathbf{NP}$, this hierarchy is located between $\mathbf{NP}$ as bottom class and $\Delta^{\mathbf{P}}_2$. Moreover, there exist strong relationships between these hierarchies. To establish our main result in this direction (Theorem 4.5) we need a characterization of polynomial-time k-bounded truth-table reducibility in terms of the Boolean algebra generated by certain sets. The following theorem is a first step in this direction.

THEOREM 4.4: *Let* $k \geqq 1$, *and let* A *and* B *be arbitrary sets. The following statements are equivalent:*

(i) $A \leqq^{\mathbf{P}}_{k-tt} B$

(ii) *there exist sets* $Q_1, \ldots, Q_k$ *which are* $\leqq^{\mathbf{P}}_m$*-reducible to* B *such that*

$$A \in \mathbf{BA}(\mathbf{P} \cup \{Q_1, \ldots, Q_k\}),$$

(iii) *there exist sets* $Q_1, \ldots, Q_k$ *which are* $\leqq^{\mathbf{P}}_m$*-reducible to* B *and sets* $P_I \in \mathbf{P}$ *for every* $I \subseteq \{1, \ldots, k\}$ *such that*

$$A = \bigcup_{I \subseteq \{1, \ldots, k\}} \left(P_I \cap \bigcap_{i \in I} Q_i \cap \bigcap_{i \notin I} \bar{Q}_i\right)$$

Proof: Suppose $A \leqq^{\mathbf{P}}_{k-tt} B$. By definition there exist polynomial-time computable functions $g, f_1, \ldots, f_k$ such that

$$c_A(x) = h_{g(x)}(c_B(f_1(x)), \ldots, c_B(f_k(x)))$$

for all $x \in \Sigma^*$ where $g(x)$ is the encoding of a Boolean circuit with $\wedge$, $\vee$ and $\neg$ gates. It is a obvious fact from Boolean function theory that for every k-ary Boolean function h with the variables $x_1, \ldots, x_k$,

$$h(x_1, \ldots, x_k) = \bigvee_{a_1, \ldots, a_k \in \{0,1\}} (h(a_1, \ldots, a_k) \wedge \bigwedge_{a_i = 1} x_i \wedge \bigwedge_{a_i = 0} \neg x_i)$$

$$= \bigvee_{I \subseteq \{1, \ldots, k\}} (h(c_I(1), \ldots, c_I(k)) \wedge \bigwedge_{i \in I} X_i \wedge \bigwedge_{i \notin I} \neg x_i).$$

Consequently,

$$c_A(x) = \bigvee_{I \subseteq \{1, \ldots, k\}} (h_{g(x)}(c_I(1), \ldots, c_I(k)) \wedge \bigwedge_{i \in I} c_B(f_i(x)) \wedge \bigwedge_{i \notin I} \neg c_B(f_i(x))).$$

Defining $Q_i = f_i^{-1}(B)$ for $i = 1, \ldots, k$ and

$$P_I = \{x: h_{g(x)}(c_I(1), \ldots, c_I(k)) = 1\} \text{ for } I \subseteq \{1, \ldots, k\}$$

we obtain $Q_i \leqq^{\mathbf{P}}_m B$, $P_I \in \mathbf{P}$ and

$$c_A(x) = \bigvee_{I \subseteq \{1, \ldots, k} (c_{P_I}(x) \wedge \bigwedge_{i \in I} c_{Q_i}(x) \wedge \bigwedge_{i \notin I} \neg c_{Q_i}(x))$$

i. e.

$$A = \bigcup_{I \subseteq \{1, \ldots, k\}} (P_I \cap \bigcap_{i \in I} Q_i \cap \bigcap_{i \notin I} \bar{Q}_i).$$

The other two directions of the proof are obvious. ■

The preceding theorem gives an equivalent condition for the $\leqq^{\mathbf{P}}_{k-tt}$-reduction from a set A to a specific set B. We obtain a stronger result if we consider the $\leqq^{\mathbf{P}}_{k-tt}$-reduction from A to an arbitrary **NP**-set.

THEOREM 4.5: *For every $k \geqq 1$ there exists a $(k+1)$-ary Boolean function h_k such that the following statements are equivalent:*

(i) $A \in \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$,

(ii) *there exist an $m \geqq 0$, an $(m+k)$-ary Boolean function f, sets $A_1, \ldots, A_m \in \mathbf{P}$ and sets $B_1, \ldots, B_k \in \mathbf{NP}$ such that*

$$c_A(x) = f(c_{A_1}(x), \ldots, c_{A_m}(x), c_{B_1}(x), \ldots, c_{B_k}(x)),$$

(iii) *There exist a set* $A_1 \in \mathbf{P}$ *and sets* $B_1, \ldots, B_k \in \mathbf{NP}$ *such that*

$$c_A(x) = h_k(c_{A_1}(x), c_{B_1}(x), \ldots, c_{B_k}(x)).$$

Proof: The proof of Theorem 2 in [14] shows that, for fixed $k \geqq 1$, there exists polynomial-time computable functions $a_1, \ldots, a_k$ such that for every k-ary Boolean function h_z (that is the Boolean function described by the Boolean formula z) and every $D_1, \ldots, D_k$:

- $h_{a_i(z)}$ is a k-ary monotonic Boolean function,
- $h_z(c_{D_1}(x), \ldots, c_{D_k}(x)) = \begin{cases} f_k(c_{B_1}(x), \ldots, c_{B_k}(x)) \text{ if } c(h_z) \text{ ends with 0} \\ g_k(c_{B_1}(x), \ldots, c_{B_k}(x)) \text{ if } c(h_z) \text{ ends with 1,} \end{cases}$

where $c_{B_i}(x) = h_{a_i(z)}(c_{D_1}(x), \ldots, c_{D_k}(x))$, for $i = 1, \ldots, k$,

$$f_k(\alpha_1, \ldots, \alpha_k) = \begin{cases} \bigvee_{i=1}^{(k-1)/2} (\alpha_{2i-1} \wedge \neg\alpha_{2i}) \vee \neg\alpha_k \text{ if } k \text{ is odd} \\ \bigvee_{i=1}^{k/2} (\alpha_{2i-1} \wedge \neg\alpha_{2i}) \text{ if } k \text{ is even,} \end{cases}$$

$$g_k(\alpha_1, \ldots, \alpha_k) = \begin{cases} \bigvee_{i=1}^{(k-1)/2} (\alpha_{2i-1} \wedge \neg\alpha_{2i}) \vee \alpha_k \text{ if } k \text{ is odd} \\ \bigvee_{i=1}^{(k-2)/2} (\alpha_{2i-1} \wedge \neg\alpha_{2i}) \vee \alpha_{k-1} \vee \neg\alpha_k \text{ if } k \text{ is even.} \end{cases}$$

Now let $A \in \leqq_{k-tt}^{\mathbf{P}}(\mathbf{NP})$. By definition there exist polynomial-time computable functions $r, s_1, \ldots, s_k$ and a set $B \in \mathbf{NP}$ such that

$$c_A(x) = h_{r(x)}(c_B(s_1(x)), \ldots, c_B(s_k(x))).$$

Obviously, the sets $D_i = s_i^{-1}(B)$ are also in **NP** ($i = 1, \ldots, k$) and we obtain

$$c_A(x) = h_{r(x)}(c_{D_1}(x), \ldots, c_{D_k}(x)) = \begin{cases} f_k(c_{B_1}(x), \ldots, c_{B_k}(x)) \text{ if } c(h_{r(x)}) \text{ ends with 0} \\ g_k(c_{B_1}(x), \ldots, c_{B_k}(x)) \text{ if } c(h_{r(x)}) \text{ ends with 1,} \end{cases}$$

where $c_{B_i}(x) = h_{a_i(r(x))}(c_{D_1}(x), \ldots, c_{D_k}(x))$, for $i = 1, \ldots, k$.

Since the Boolean functions $h_{a_i(r(x))}$ are monotonic and $D_1, \ldots, D_k$ are in **NP**, the sets $B_1, \ldots, B_k$ are also in **NP**.

Furthermore, the set $A_1 = \{x : c(h_{r(x)}) \text{ ends with } 0\}$ is in **P**, because all $h_{r(x)}$ are k-ary Boolean functions.

Defining $h_k(\alpha_0, \alpha_1, \ldots, \alpha_k) = (\alpha_0 \wedge f_k(\alpha_1, \ldots, \alpha_k)) \vee (\neg\alpha_0 \wedge g_k(\alpha_1, \ldots, \alpha_k))$ we obtain

$$c_A(x) = (c_{A_1}(x) \wedge f_k(c_{B_1}(x), \ldots, c_{B_k}(x))) \vee (\neg c_{A_1}(x) \wedge g_k(c_{B_1}(x), \ldots, c_{B_k}(x)))$$
$$= h_k(c_{A_1}(x), c_{B_1}(x), \ldots, c_{B_k}(x)). \blacksquare$$

Note that in the above theorem we can replace **NP** by any class of sets which is closed under $\leqq^{\mathbf{P}}_{bf}$-reducibility via Boolean formulas with operations $\vee$ and $\wedge$.

COROLLARY 4.6: (i) *For every* $k \geqq 1$,

$$\mathbf{DIFF_k} \cup \mathbf{co\,DIFF_k} \subseteq \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) \subseteq \mathbf{DIFF_{k+1}} \cap \mathbf{co\,DIFF_{k+1}}.$$

(ii) $\leqq^{\mathbf{P}}_{btt}(\mathbf{NP}) = \mathbf{BA}(\mathbf{NP})$.

(iii) *For every* $k \geqq 1$,

$$\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{(k+1)-tt}(\mathbf{NP}) \textit{ implies } \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{btt}(\mathbf{NP}).$$

(iv) *If the truth-table hierarchy is infinite then* $\leqq^{\mathbf{P}}_{btt}(\mathbf{NP})$ *is strictly included in* $\leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$.

Proof: (i) It is obvious that $\mathbf{DIFF_k} \cup \mathbf{co\,DIFF_k} \subseteq \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$.

Let $A \in \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$. By Proposition 4.3 (i) we have also $\bar{A} \in \leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP})$, and by Theorem 4.5 we have A, $\bar{A} \in h_k(\mathbf{NP})$. Since h_k is $(k+1)$-ary we can conclude $m(h_k) \leqq k+1$. By Theorem 3.5 there holds $h_k(\mathbf{NP}) \subseteq \mathbf{C}^{\mathbf{NP}}_{k+1}$ or $h_k(\mathbf{NP}) \subseteq \mathbf{D}^{\mathbf{NP}}_{k+1}$. Consequently, A, $\bar{A} \in \mathbf{C}^{\mathbf{NP}}_{k+1}$ or A, $\bar{A} \in \mathbf{D}^{\mathbf{NP}}_{k+1}$. From $\mathbf{D}^{\mathbf{NP}}_{k+1} = \mathbf{co\,C}^{\mathbf{NP}}_{k+1}$ [Theorem 3.6 (iii)] we conclude $A \in \mathbf{C}^{\mathbf{NP}}_{k+1} \cap \mathbf{D}^{\mathbf{NP}}_{k+1}$.

(ii) is an immediate consequence of (i) and Corollary 3.9 (ii).

(iii) By (i) the equality $\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{(k+1)-tt}(\mathbf{NP})$ implies $\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \mathbf{DIFF_{k+1}} = \mathbf{co\,DIFF_{k+1}}$. From Corollary 3.9 (iv) we conclude $\mathbf{DIFF_{k+1}} = \mathbf{DH}$, and Corollary 3.9 (iii) and Corollary 4.6 (ii) yield $\leqq^{\mathbf{P}}_{k-tt}(\mathbf{NP}) = \mathbf{DIFF_{k+1}} = \mathbf{DH} = \mathbf{BA}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{btt}(\mathbf{NP})$.

(IV) Obviously, $\mathrm{SAT}^{\omega-\Delta} \leqq^{\mathbf{P}}_{tt} \mathrm{SAT}$. Therefore, if $\leqq^{\mathbf{P}}_{btt}(\mathbf{NP}) = \leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$ then $\mathrm{SAT}^{\omega-\Delta} \in \leqq^{\mathbf{P}}_{btt}(\mathbf{NP}) = \mathbf{BA}(\mathbf{NP}) = \mathbf{DH}$. By Theorem 3.11 (ii) $\mathrm{SAT}^{\omega-\Delta}$ is **DH**-complete. Using Corollary 3.12 it follows that the difference hierarchy, and consequently also the truth-table hierarchy, is finite. $\blacksquare$

Thus the union of the difference hierarchy and the union of the truth-table hierarchy are the same. The analogous fact in recursive function theory has already been observed (*see* [12]).

Next we define a sequence of sets which will subsequently be shown to be complete for the several levels of the truth-table hierarchy.

DEFINITION 4.7: For each set L and $k \geqq 1$ define

(i) $L^{k-tt} = \{ (z, x_1, \ldots, x_k)$: z encodes a Boolean circuit with k input nodes and $h_z(c_L(x_1), \ldots, c_L(x_k)) = 1\}$,

(ii) $L^{\omega-tt} = \bigcup_{k \geqq 1} L^{k-tt}$,

(iii) $L^{k-bf} = \{(z, x_1, \ldots, x_k)$: z encodes a Boolean formula with k variables and $h_z(c_L(x_1), \ldots, c_L(x_k)) = 1\}$,

(iv) $L^{\omega-bf} = \bigcup_{k \geqq 1} L^{k-bf}$,

(v) $L^{k-ftt} = \{(z, x_1, \ldots, x_k)$: z encodes a full truth-table with k variables and $h_z(c_L(x_1), \ldots, c_L(x_k)) = 1\}$,

(vi) $L^{\omega-ftt} = \bigcup_{k \geqq 1} L^{k-ftt}$.

PROPOSITION 4.8: *For each set L and $k \geqq 1$,*

(i) $L^{k-ftt} \equiv_m^{\mathbf{P}} L^{k-bf} \equiv_m^{\mathbf{P}} L^{k-tt}$.

(ii) $L^{\omega-ftt} \leqq_m^{\mathbf{P}} L^{\omega-bf} \leqq_m^{\mathbf{P}} L^{\omega-tt}$.

Proof: Use the same arguments as for Proposition 4.2 ■

We do not know whether $L^{\omega-ftt} \equiv_m^{\mathbf{P}} L^{\omega-bf}$ or $L^{\omega-bf} \equiv_m^{\mathbf{P}} L^{\omega-tt}$.

THEOREM 4.9: (i) *For each $k \geqq 1$,*

$$\mathrm{SAT}^{k-ftt}, \mathrm{SAT}^{k-bf} \text{ and } \mathrm{SAT}^{k-tt} \text{ are } \leqq_{k-tt}^{\mathbf{P}}(\mathbf{NP})\text{-complete.}$$

(ii) $\mathrm{SAT}^{\omega-ftt}$, $\mathrm{SAT}^{\omega-bf}$ *and* $\mathrm{SAT}^{\omega-tt}$ *are* $\mathbf{BA}(\mathbf{NP})$*-hard.*

Proof: (i) Because of Proposition 4.8 we restrict ourselves to SAT^{k-tt}. Clearly, $\mathrm{SAT}^{k-tt} \leqq_{k-tt}^{\mathbf{P}} \mathrm{SAT}$. Thus, $\mathrm{SAT}^{k-tt} \in \leqq_{k-tt}^{\mathbf{P}}(\mathbf{NP})$.

Suppose $A \in \leqq_{k-tt}^{\mathbf{P}}(\mathbf{NP})$. Then there exist a $B \in \mathbf{NP}$ and polynomial-time computable functions $g, f_1, \ldots, f_k$ such that

$$c_A(x) = h_{g(x)}(c_B(f_1(x)), \ldots, c_B(f_k(x))).$$

Because of $B \in \mathbf{NP}$ there exists a polynomial-time computable function g' such that $c_B(x) = c_{\mathrm{SAT}}(g'(x))$ for all $x \in \Sigma^*$. Consequently,

$$\begin{aligned} x \in A \text{ iff } & h_{g(x)}(c_B(f_1(x)), \ldots, c_B(f_k(x))) = 1 \\ \text{iff } & h_{g(x)}(c_{\mathrm{SAT}}(g'(f_1(x))), \ldots, c_{\mathrm{SAT}}(g'(f_k(x)))) = 1 \end{aligned}$$

$$\text{iff } (g(x), g'(f_1(x)), \ldots, g'(f_k(x))) \in \mathrm{SAT}^{k-tt}.$$

(ii) follows from (i). ■

5. BEYOND THE HIERARCHIES

In this section we consider the exact complexity of the sets $\mathrm{SAT}^{\omega-\Delta}$, $\mathrm{SAT}^{\omega-ftt}$, $\mathrm{SAT}^{\omega-bf}$ and $\mathrm{SAT}^{\omega-tt}$ defined in sections 3 and 4. That is, we are interested in the complexity of the different natural "ω-jumps" of our hierarchies. Note that one answer for the analogous question concerning the polynomial-time hierarchy is that the ω-jump B_ω defined in [13] is **PSPACE**-complete. However, in [5] another ω-jump K^ω of the polynomial-time hierarchy has been considered, and it is an open question whether $B_\omega \equiv_m^{\mathbf{P}} K^\omega$.

In Proposition 4.8 we have shown that

$$\mathrm{SAT}^{\omega-ftt} \leqq_m^{\mathbf{P}} \mathrm{SAT}^{\omega-bf} \leqq_m^{\mathbf{P}} \mathrm{SAT}^{\omega-tt}.$$

It is not very likely that these sets are $\equiv_m^{\mathbf{P}}$-equivalent. Thus we have the somewhat surprising suspicion that the complexity of the ω-jumps of the truth-table hierarchy depends heavily on the succinctness of the encodings of Boolean functions that we use. On the other hand, it turns out that $SAT^{\omega-\Delta} \equiv_m^{\mathbf{P}} \mathrm{SAT}^{\omega-bf}$, i.e. that the ω-jump of the difference hierarchy has the same complexity as one of the ω-jumps of the truth-table hierarchy.

THEOREM 5.1: (*i*) $SAT^{\omega-tt}$ *is* $\leqq_{tt}^{\mathbf{P}}(\mathbf{NP})$*-complete.*

(ii) $\mathrm{SAT}^{\omega-bf}$ *is* $\leqq_{bf}^{\mathbf{P}}(\mathbf{NP})$*-complete.*

(iii) $\mathrm{SAT}^{\omega-ftt}$ *is* $\leqq_{ftt}^{\mathbf{P}}(\mathbf{NP})$*-complete.*

Proof: As for Theorem 4.9 (i). ■

Let us emphasize once more that it is not likely that $\leqq_{ftt}^{\mathbf{P}}(\mathbf{NP}) = \leqq_{bf}^{\mathbf{P}}(\mathbf{NP})$ or $\leqq_{ftt}^{\mathbf{P}}(\mathbf{NP}) = \leqq_{tt}^{\mathbf{P}}(\mathbf{NP})$ because these problems are closely related to the problems of whether there exist polynomial-time algorithms converting Boolean formulas or Boolean circuits into equivalent truth-tables. A similar dependency of the complexity of problems from the input presentation has been observed by other authors before. Sometimes an exponential increase of complexity can be observed if one chooses a succint encoding of the instances of the problems (*see* [4] and [15]).

Next we show that $\mathrm{SAT}^{\omega-\Delta}$ is also $\leqq_{bf}^{\mathbf{P}}(\mathbf{NP})$-complete. This is a consequence of a result in [16] where a sufficient condition is given for a set A to be complete in $\leqq_{bf}^{\mathbf{P}}(\mathbf{NP})$. Note that this class is denoted there by $\mathbf{P}_{bf}^{\mathbf{NP}}$.

THEOREM 5.2 [16]: *Let D be* **NP***-complete and let $A \in \leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$. If there exist a polynomial-time computable function f such that*

$$\left|\{i : x_i \in D\}\right| \text{ is odd} \leftrightarrow f(x_1, \ldots, x_k) \in A$$

for all $k \geqq 1$, $x_1, \ldots, x_k \in \Sigma^*$ such that $c_D(x_1) \geqq c_D(x_2) \geqq \ldots \geqq c_D(x_k)$ then A is $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$-complete. ■

COROLLARY 5.3: $\mathrm{SAT}^{\omega - \Delta}$ *is $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$-complete.* ■

In [16] many natural problems are shown to be $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$-complete, for example the problem $\{(G, n_1, \ldots, n_k) : k \geqq 1$ and the chromatic number of the graph G is one of the numbers $n_1, \ldots, n_k\}$.

It should be noted that our results do not depend particularly on **NP** and SAT as **NP**-complete problem. Just as well similar results can be developed for other classes having complete sets and being closed under union and intersection.

Finally let us mention some relationships between two classes investigated in [2] and the classes $\leqq^{\mathbf{P}}_{bf}(\mathbf{NP})$ and $\leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$.

For a deterministic oracle Turing machine M with oracle set B and input x, let $Q(M, B, x)$ be the set of all queries which are asked by M to B during its computation on x. Further, let

$$Q(M, x) = \bigcup \{Q(M, B, x) : B \text{ oracle set}\},$$

and

$$QA(M, x) = \bigcup \{Q(M, B, x) : B \text{ oracle set and } M \text{ accepts } x \text{ with oracle } B\}.$$

For a set B, the set A is in **P.UNIV.ALL** (B) [**P.UNIV.ACC** (B)] if and only if there exist two deterministic polynomial-time bounded Turing machines M and M' such that for every input x:

- $x \in A$ iff M accepts x using oracle B,
- M' computes on input x the set $Q(M, x)$ $[QA(M, x)]$.

Furthermore, define

$$\mathbf{P.UNIV.ALL} = \bigcup \{\mathbf{P.UNIV.ALL}(B) : B \in \mathbf{NP}\}$$

and

$$\mathbf{P.UNIV.ACC} = \bigcup \{\mathbf{P.UNIV.ACC}(B) : B \in \mathbf{NP}\}.$$

It follows from a result in [8] that $\mathbf{P.UNIV.ALL} = \leqq^{\mathbf{P}}_{tt}(\mathbf{NP})$. Moreover,

THEOREM 5.4 [2], [6]: For every set B,

$$\mathbf{P.UNIV.ALL}(B) = \mathbf{P.UNIV.ACC}(B).$$

Proof: For the inclusion "$\supseteq$" *see* [2].

For the other inclusion we reproduce the proof from [6]. Let M_1 be a deterministic polynomial-time bounded oracle Turing machine accepting the set A with oracle B, and let M_2 be a deterministic polynomial-time bounded Turing machine computing on input x the set $Q(M_1, x)$. Now we choose an arbitrary word b from B, and we construct a deterministic polynomial-time bounded oracle Turing machine M_1' such that

- M_1' accepts the set A with oracle B,
- $QA(M_1', x) = Q(M_1, x) \cup \{b\}$ for all inputs x.

This implies $A \in \mathbf{P.UNIV.ACC}(B)$.

The Turing machine M_1' works (with an arbitrary oracle C) as follows: On input x it computes first $Q(M_1, x)$ by simulating M_2. Then it asks all $y \in Q(M_1, x) \cup \{b\}$ to the oracle C. Finally it simulates M_1 on input x. If x is accepted by M_1 then it is also accepted by M_1'. If x is rejected by M_1 then x is rejected by M_1' if and only if $b \in C$.

Evidently, for $C = B$ the machines M_1 and M_1' accept the same set of words, namely A.

On the other hand, it is obvious that

$$QA(M_1', x) \subseteq Q(M_1, x) \cup \{b\} \subseteq Q(M_1', \Sigma^* - \{b\}, x) \subseteq QA(M_1', x). \blacksquare$$

REFERENCES

1. A. BLASS and Y. GUREVICH, *On the Unique Satisfiability Problem, Information and Control*, Vol. 55, 1982, pp. 80-88.
2. R. V. BOOK, T. J. LONG and A. L. SELMAN, *Quantitative Relativizations of Complexity Classes*, S.I.A.M. J. Comput., Vol. 13, 1984, pp. 461-487.
3. J. CAI and L. HEMACHANDRA, *The Boolean Hierarchy: Hardware Over NP*, Proc. of the Structure in Complexity Theory Conference, Berkeley 1986 (to appear).
4. H. GALPERIN and A. WIGDERSON, *Succinct representation of graphs*, Information and Control, Vol. 56, 1986, pp. 183-198.
5. H. HELLER, *Relativized Polynomial Hierarchies Extending Two Levels*, Math. Syst. Theory, Vol. 17, 1984, pp. 71-84.
6. J. KÖBLER, *Untersuchung verschiedener polynomieller Reduktionsklassen von NP*, diploma thesis, University of Stuttgart, 1985.
7. R. E. LADNER, *The Circuit Value Problem is log Space Complete for P*, SIGACT News, Vol. 7, 1975, pp. 18-20.

8. R. E. LADNER, N. A. LYNCH and A. L. SELMAN, *A Comparison of Polynomial Time Reducibilities*, Theor. Comput. Sci., Vol. 1, 1975, pp. 103-123.
9. E. W. LEGETT and D. J. MOORE, *Optimization Problems and the Polynomial Hierarchy*, Theor. Comput. Sci., Vol. 15, 1981, pp. 279-289.
10. C. H. PAPADIMITRIOU, *On the Complexity of Unique Solutions*, Proc. 23rd Symp. Found of Comput. Sci, 1982, pp. 14-20.
11. C. H. PAPADIMITRIOU and M. YANNAKAKIS, *The Complexity of Facets (and Some Facets of Complexity)*, Proc. 14th A.C.M. Symp. Theory of Comput., 1982, pp. 255-260.
12. D. B. POSNER, *A Survey of Non-re Degress* $\leqq 0'$, in DRAKE/WAINER Eds, Recursion Theory: its Generalisations and Applications, Cambridge University Press, 1980, pp. 52-109.
13. L. J. STOCKMEYER, *The Polynomial-Time Hierarchy*, Theor. Comput. Sci., Vol. 3, 1977, pp. 1-22.
14. G. WECHSUNG and K. W. WAGNER, *On the Boolean closure of NP*, submitted for publication (extended abstract as: G. WECHSUNG, On the Boolean closure of NP; L.N.C.S., Vol. 199, 1985, pp. 485-493).
15. K. W. WAGNER, *The Complexity of Problems Concerning Graphs with Regularities*, Proc. Symp. Math. Found. of Comput. Sci., 1984, Lecture Notes in Computer Science, Vol. 176, 1984, pp. 544-552.
16. K. W. WAGNER, *Maximum and Minimum Problems, and Some Closures of NP*, Proc. 13th Intern Coll. Autom., Lang. and Progr., 1986 (to appear).