Theoretical Computer Science 119 (1993) 293–310
Elsevier

If not empty, NP − P is topologically large

Marius Zimand

Department of Computer Science, University of Rochester, Rochester, NY 14627, USA

Communicated by A. Salomaa
Received August 1991

Zimand, M., If not empty, NP − P is topologically large, Theoretical Computer Science 119 (1993) 293–310.

In the classical Cantor topology or in the superset topology, NP and, consequently, classes included in NP are *meagre*. However, in a natural combination of the two topologies, we prove that NP − P, if not empty, is a *second category* class, while NP-complete sets form a *first category* class. These results are extended to different levels in the polynomial hierarchy and to the low and high hierarchies. P-immune sets in NP, NP-simple sets, P-bi-immune sets and NP-effectively simple sets are all second category (if not empty). It is shown that if C is any of the above second category classes, then for all $B \in \mathrm{NP}$ there exists an $A \in C$ such that A is arbitrarily close to B infinitely often.

1. Introduction

Since there is strong evidence that NP ≠ P, it is natural to ask how "large" is NP − P. There are many results which indicate that this class is rich. For example, in [15] the existence of sets in NP which are neither in P nor NP-complete is shown, and in [4] the existence of infinite families of incomparable sets in NP − P under polynomial-time reductibilities is proved, provided that NP ≠ P. In this paper we intend a more direct investigation of the abundance of sets in NP − P. There are usually two concepts that are used to tackle such questions: measure and Baire category [18, p. 269]. The first one is not relevant in our case. One can easily see that NP itself has null measure and, thus, NP − P is trivially of measure zero. There are some difficulties also from the point of view of Baire category. In the classical setting, a set in a topological space is rare if its closure does not contain any nonempty set. A set is of the first Baire category if it is a finite or denumerable union of rare sets and it is of the

Correspondence to: M. Zimand, Department of Computer Science, University of Rochester, Rochester, NY 14627, USA. Email: zimand@cs.rochester.edu.

second Baire category if it is not of the first category. As we deal with algorithmical objects, it is natural to consider recursive variants of these definitions (see [17]). The standard procedure goes along the following lines [9]. Let $X=\{0,1\}$. For $i\geqslant 1$, let a_i be the ith string in the lexicographical order of X^*. Thus, $X^*=\{a_1<a_2\cdots<a_n\ldots\}$, where $<$ is the lexicographical order on X^*. A class is a set of languages. We identify a recursive set A included in X^* by its characteristic sequence $A(1)A(2)\ldots A(n)\ldots$, where, for each positive integer i, $A(i)=1$ if $a_i\in A$ and $A(i)=0$ otherwise. By this codification, $A\in X^{\infty}$, where X^{∞} denotes the set of infinite words over the alphabet X. For v in X^{∞}, we define its support by $\operatorname{supp}(v)=\{i\geqslant 1 \mid v(i)=1\}$, where $v(i)$ is the ith bit of v. If v in X^{∞} has finite support then $|v|=\max\{i \mid i\in\operatorname{supp}(v)\}$ is the length of v. We identify a class A of sets with the family of the characteristic sequences of the sets in the class. In this way, all the classes we are going to study are included in X^{∞}. Since we need a constructive topology on X^{∞}, it is necessary for the basic neighborhoods to admit a finite characterization; hence, the basic neighborhoods will be generated by strings in X^{∞} with finite support. Let F be the set of the strings in X^{∞} with finite support and consider the set of basic neighborhoods U_v, v in F. A set A included in X^{∞} is recursively rare if there exists a recursive function f which, for every v in F, produces a witness w in F with $w\in U_v-\operatorname{closure}(A)$, i.e. w proves that U_v is not included in the closure of A. There are mainly two ways largely used in the literature for defining U_v, v in X^* (once again we identify F with X^*, by associating with each v in F, the finite string $v(1)v(2)\ldots v(|v|)$ in X^*). The first way [17] consists in defining, for every v in X^*,

$$U'_v=\{w\in X^{\infty} \mid v \text{ is a prefix of } w\}.$$

This corresponds to extensions of initial finite segments, an operation which is widely used in computational theory [18]. Unfortunately, under this definition, NP is of the first category and, in consequence, NP − P is trivially of the first category. The second method [8, 9] consists in defining, for every v in X^*,

$$U_v=\{w\in X^{\infty} \mid \forall i\leqslant |v|\ [v(i)=1 \Rightarrow w(i)=1]\},$$

where $|v|$ denotes the length of the finite string $v\in X^*$. This definition corresponds to extensions of sets. If we simply require that the witness function produces for every v in X^* a string w of finite support in U_v, the resulting definitions are again inadequate for our purposes. In this case, one can see that $\mathrm{NP}-X^*$ is a class of the first category; so, NP − P is, again trivially, of the first category. The two methods described above are too rough in order to distinguish, from a topological point of view, the size of some interesting classes included in NP. Hence, we propose a mixing of the two variants. The basic neighborhoods are the ones from the second method, but w has the supplementary property that v is a prefix of w, as in the first method. The family (U_v), v in X^*, is closed under the intersection; so, in the topology generated by it, the open sets consist of arbitrary unions of basic neighborhoods. If A is included in X^{∞}, one can see that $\operatorname{closure}(A)=\{w\in X^{\infty} \mid \exists v\in A, \operatorname{supp}(w)\leqslant\operatorname{supp}(v)\}$. After some obvious computations and keeping into account the above remarks, we obtain the following definitions.

Definition 1.1. A class A is called recursively rare if there is a recursive function $f: X^* \to X^*$ such that, for every v in X^*,

(i) v is a prefix of $f(v)$, and

(ii) $A \cap U_{f(v)} = \emptyset$.

Definition 1.2. (a) A class A is recursively of the first Baire category if there is a decomposition $A = \bigcup A_i$ and a recursive function $f: N \times X^* \to X^*$ such that, for every $i \in N$ and for every $v \in X^*$,

(i) v is a prefix of $f(i, v)$, and

(ii) $A_i \cap U_{f(i,v)} = \emptyset$.

(b) A class A is recursively of the second Baire category if it is not recursively of the first Baire category.

For the sake of brevity, we shall simply say first or second Baire category, or even first or second category.

Under these definitions we prove that NP − P is of the second Baire category provided that it is not empty. This is the main result of this paper and is stated in Section 2. In the same section we extend this result to the other levels of the polynomial hierarchy and to the low hierarchy [19, 20]. We also show here that the class of NP-hard sets as well as all the levels of the high hierarchy are of the first category. Since the finite union of first category classes is still a class of the first category, it follows that the class of sets in NP that are neither in P nor NP-complete is of the second category. In Section 3 we obtain similar results for classes of sets which realize stronger forms of separation between NP − P. It is shown that the classes of P-immune sets in NP, of NP-simple sets or of P-bi-immune sets in NP, if not empty, are all of the second Baire category. Moreover, the class of effectively NP-simple sets is also either empty or of the second Baire category. This result contrasts strongly with the situation in classical recursive function theory, where all effectively simple sets are complete for the class of r.e. sets [22, pp. 79–87]. Many approaches in structural complexity theory have origins in recursive function theory, by observing the analogy between sets in NP and r.e. sets on one side and between sets in P and recursive sets on the other. Our result supplies an example where this analogy does not work and it suggests that our methods may be used in the investigation of some structural properties of complexity classes. In fact, Section 4 provides such applications to the study of the density of sets in NP − P. It is shown that if NP ≠ P, then for an arbitrarily slowly increasing, unbounded and recursive function r and for any B in NP, there exists A in NP − P such that $|((A - B) \cup (B - A))^{\leqslant n}| \leqslant r(n)$ for infinitely many n. Similarly, one can show [23] that for any recursive set B and for any increasing and recursive function r, there exists a set in NP − P such that $|(B - A)^{\leqslant n}| \leqslant r(n)$ for infinitely many n. Similar results hold for all the other classes shown to be of the second Baire category in Sections 2 and 3. We also find an intrinsic characterization of P which is similar in spirit with the one found by Ambos-Spies [2] in terms of measure theory. In [2] it is shown that a set A is in P if and only if $\{B \mid A \leqslant_m^p B\}$ has measure 1 (for other related results see [7]). We prove that A is in P if and only if $\{B \mid A \leqslant_{\mathrm{T}}^{\mathrm{P}} B\}$ is of the second Baire category. In our view, these results validate the proposed topology for our investigation of sizes of complexity classes based on Baire category. However, note that an approach based on the topology generated by the system of basic neighborhoods (U_v') has been considered in [16], by restricting the complexity of the witness function from Definition 1.1.

The rest of this section is dedicated to the presentation of some more notation. In this paper, if not specified otherwise, all sets A are of strings over the alphabet $X = \{0, 1\}$. For a set A, $\bar{A}$ denotes the complement of A. For a finite set included in X^*, let $|A|$ denote the number of strings in A, and let $A^{\leqslant n}$ denote the set of strings in A such that $|x| \leqslant n$. By ":=" we denote the assignation. We assume the reader is familiar with Turing machines, nondeterministic Turing machines, their time complexity, the classes P, NP and the polynomial hierarchy [5, 19]. We fix enumerations $\{P_k\}_k$ and $\{\mathrm{NP}_k\}_k$ of the polynomial-time deterministic and nondeterministic machines. Sometimes P_k (NP_k) will also denote the language accepted by the machine P_k (NP_k). From the context it will be clear whether the meaning of P_k (NP_k) is the kth machine in the above enumeration or the language accepted by it.

2. NP − P, the low and the high hierarchies

We present first a rather technical lemma that will be the core of all our proofs showing various classes of sets to be of the second Baire category. All these results rely on the π_2-hardness principle brought in evidence mostly by Hartmanis [12, 13]. We show that if A is a class of the first Baire category, then A is included in a set D which is in the Σ_2 level of Kleene's arithmetical hierarchy and such that Co-Fin is included in the complement of D. It follows that if A is a class which has a property similar to being π_2-hard (i.e. Tot $\leqslant_{\mathrm{m}}(A, \text{Co-Fin})$; see below) then A is necessarily of the second Baire category.

Let $(W_i)_{i \in N}$ be an enumeration of the class of recursively enumerable sets of strings and $W_{x,s}$ be the set of the strings enumerated in W_x within the first s steps of computation [22]. Let $\text{Tot} = \{x \mid W_x = X^*\}$ and $\text{Co-Fin} = \{x \mid \bar{W}_x \text{ is finite}\}$. For a class A of recursively enumerable sets, we say that Tot $\leqslant_{\mathrm{m}}(A, \text{Co-Fin})$ if there is a recursive function $s: N \to N$ such that $i \in \text{Tot}$ implies $W_{s(i)} \in A$ and $i \notin \text{Tot}$ implies $s(i) \in \text{Co-Fin}$.

Lemma 2.1. *Let A be a class of recursively enumerable sets of strings with* Tot $\leqslant_{\mathrm{m}}$ $(A, \text{Co-Fin})$. *Then A is of the second category.*

Proof. Suppose A is of the first category. This means that there is a decomposition $A = \bigcup_{i \geqslant 0} A_i$ and a recursive function $f: N \times X^* \to X^*$ such that, for every integer i and every string $w \in X^*$, w is a prefix of $f(i, w)$ and $\mathrm{U}_{f(i,w)} \cap A_i = \emptyset$.

Let $D = \{j \in N \mid (\exists i)(\forall n)(\forall s)(\exists k \in N,\ n < k \leqslant |f(i, 0^n)|,\ a_k \notin W_{j,s})\}$. Clearly, Co-Fin is included in $\bar{D}$ and D is in the Σ_2 level of the arithmetic hierarchy [18, 22]. Observe that $W_j \in A$ implies $j \in D$. Indeed, if $W_j \in A$ then there exists some i such that $W_j \in A_i$. Also note that if, for some n, it holds true that, for all k with $n < k \leqslant |f(i, 0^n)|$, $a_k \in W_j$, then we can conclude that $W_j \in U_{f(i,0^n)}$, which contradicts the fact that $A_i \cap U_{f(i,0^n)} = \emptyset$ for all integers i and n. Now it follows that

$i \in \mathrm{Tot}$ implies $W_{s(i)} \in A$, which implies $s(i) \in D$, and

$i \in \mathrm{Tot}$ implies $s(i) \in \text{Co-Fin} \subset \bar{D}$.

Consequently, $\mathrm{Tot} \leqslant_m D$, which is a contradiction because Tot is π_2-complete [22] and D is in Σ_2. □

We can now proceed to the main result of this paper, i.e. the fact that if $\mathrm{NP} - \mathrm{P} \neq \emptyset$, then $\mathrm{NP} - \mathrm{P}$ is of the second category. By Lemma 2.1 we only have to show that $\mathrm{Tot} \leqslant_m (\mathrm{NP} - \mathrm{P}, \text{Co-Fin})$. This follows by a standard argument (see [12]) which uses the delayed diagonalization method of Ladner [15].

Lemma 2.2. *If* $\mathrm{NP} - \mathrm{P} \neq \emptyset$, *then* $\mathrm{Tot} \leqslant_m (\mathrm{NP} - \mathrm{P}, \text{Co-Fin})$.

Proof. Let $\{M_i\}_{i \in N}$ be an enumeration of all deterministic Turing machines such that $W_i = \mathrm{dom}(M_i)$. For every Turing machine M_i we define a nondeterministic polynomial-time Turing machine $\mathrm{NP}_{s(i)}$ such that if $i \in \mathrm{Tot}$, then $\mathrm{NP}_{s(i)} \in \mathrm{NP} - \mathrm{P}$, and if $i \notin \mathrm{Tot}$, then the language accepted by $\mathrm{NP}_{s(i)}$ is co-finite. The computation of $\mathrm{NP}_{s(i)}$ is performed in stages, starting with Stage = 0. The machine $\mathrm{NP}_{s(i)}$ has two kinds of objectives depending on the parity of Stage. Thus, if Stage $= 2e$, $\mathrm{NP}_{s(i)}$ tries to find, in polynomial time, whether M_i accepts a_e. If and when this happens, Stage is incremented to $2e+1$. In case $\mathrm{NP}_{s(i)}$ does not succeed in determining whether M_i accepts a_e, the current input is accepted. In this way, if $i \notin \mathrm{Tot}$, then clearly $\mathrm{NP}_{s(i)}$ remains stuck in an even stage and it accepts a co-finite set. On the other hand, if Stage $= 2e+1$, $\mathrm{NP}_{s(i)}$ looks for a string y such that $\mathrm{NP}_{s(i)}(y) \neq \mathrm{P}_e(y)$. If no such y is found, the current input x is accepted if and only if $x \in \mathrm{SAT}$. Consequently, if repeated failures occur, then $\mathrm{NP}_{s(i)}$ begins to look like SAT. Hence, $\mathrm{NP}_{s(i)}$ will eventually find a string y such that $\mathrm{NP}_{s(i)}(y) \neq \mathrm{P}_e(y)$, because, otherwise, SAT would be in P, which contradicts $\mathrm{NP} \neq \mathrm{P}$. When this happens, Stage is incremented to the value $2e+2$.

The construction of $\mathrm{NP}_{s(i)}$: Initially, Stage:= 0. On input $x \in X^*$ of length n, the computation of $\mathrm{NP}_{s(i)}$ proceeds as follows:

(a) For n steps, $\mathrm{NP}_{s(i)}$ simulates deterministically the previous computations $\mathrm{NP}_{s(i)}(a_1)$, $\mathrm{NP}_{s(i)}(a_2)$... and determines how many stages have been completed so far. The variable Stage contains the value of the stage that $\mathrm{NP}_{s(i)}$ has been able to find within the allowed time.

(b) *Case* 1: Stage $= 2e$. For n steps $\mathrm{NP}_{s(i)}$ simulates M_i on input a_e. If M_i does not accept a_e in due time, then x is accepted. Otherwise Stage:= $2e+1$, and x is accepted.

Case 2: Stage $= 2e+1$. For n steps $\mathrm{NP}_{s(i)}$ looks for a string $y \leqslant x$ such that $\mathrm{NP}_{s(i)}(y) \neq \mathrm{P}_e(y)$. In doing this $\mathrm{NP}_{s(i)}$, on input y, is simulated deterministically. If such a y is found then Stage:= $2e+2$ and x is accepted. Otherwise x is accepted if and only if $x \in \mathrm{SAT}$. End of construction of $\mathrm{NP}_{s(i)}$.

The following claims settle this lemma.

Claim 2.3. $\mathrm{NP}_{s(i)}$ *is a polynomial-time nondeterministic algorithm.*

Proof. The only nondeterministic step in the computation of $\mathrm{NP}_{s(i)}$ on input x occurs in (b), case 2. In this case $\mathrm{NP}_{s(i)}$ has to determine if $x \in \mathrm{SAT}$, which can be done in a polynomial-time nondeterministic computation. All the other computations are performed in deterministic polynomial time (in fact, linear time). □

Claim 2.4. *If at a certain moment in the construction of* $\mathrm{NP}_{s(i)}$, $\mathrm{Stage} = 2e$ *and* M_i *accepts* a_e, *then there exists a later moment when* $\mathrm{Stage} := 2e+1$.

Proof. For sufficiently long input x, $\mathrm{NP}_{s(i)}$ has enough time to find that M_i accepts a_e. □

Claim 2.5. *If at a certain moment in the construction of* $\mathrm{NP}_{s(i)}$, $\mathrm{Stage} = 2e+1$, *then* $\mathrm{NP}_{s(i)} \neq \mathrm{P}_e$ *and there exists a later moment when Stage*$:= 2e+2$.

Proof. By the argument presented before the detailed construction of $\mathrm{NP}_{s(i)}$. □

Claim 2.6. *If* $i \notin \mathrm{Tot}$, *then* $\mathrm{NP}_{s(i)}$ *accepts a co-finite set.*

Proof. Let a_e be the minimal string such that M_i does not accept a_e. By Claims 2.4 and 2.5, $\mathrm{Stage} = 2e$ is reached. Clearly, from this moment on, $\mathrm{NP}_{s(i)}$ accepts every input string. □

Claim 2.7. *If* $i \notin \mathrm{Tot}$, *then* $\mathrm{NP}_{s(i)} \in \mathrm{NP} - \mathrm{P}$.

Proof. By Claim 2.3, $\mathrm{NP}_{s(i)} \in \mathrm{NP}$. Taking into account that $i \in \mathrm{Tot}$ and Claim 2.4, it follows that the assertion in Claim 2.5 holds for every integer e. □

Theorem 2.8. *If* $\mathrm{NP} - \mathrm{P} \neq \emptyset$, *then* $\mathrm{NP} - \mathrm{P}$ *is of the second Baire category.*

Proof. Immediate, from Lemmas 2.1 and 2.2. □

Minor changes in the proof of Lemma 2.2 yield the following generalization of Theorem 2.8 to higher levels in the polynomial hierarchy.

Theorem 2.9. (a) *For all* $k \geqslant 0$, *if* $\Sigma_{k+1}^{\mathrm{P}} - \Sigma_k^{\mathrm{P}} \neq \emptyset$, *then* $\Sigma_{k+1}^{\mathrm{P}} - \Sigma_k^{\mathrm{P}}$ *is of the second Baire category.*

(2) *If* $\mathrm{PSPACE} - \mathrm{PH} \neq \emptyset$, *then* $\mathrm{PSPACE} - \mathrm{PH}$ *is of the second Baire category.*

All these results could lead someone to the idea that everything is of the second category. This is not true. For example the class of NP-complete sets is of the first category. Denote by $\mathrm{Comp} = \{A \in \mathrm{NP} \mid A \text{ is } \leqslant_{\mathrm{T}}^{\mathrm{P}} \text{ complete}\}$.

Theorem 2.10. *If* $\mathrm{NP} - \mathrm{P} \neq \emptyset$, *then* Comp *is of the first Baire category.*

Proof. Recall that $\{P_k\}$ is an enumeration of polynomial-time deterministic oracle machines. Without loss of generality, we assume that the time complexity of P_k on an input of length n is bounded by n^k+k. Clearly, $\mathrm{Comp}=\bigcup_{i\geqslant 0}\mathrm{Comp}_i$, where $\mathrm{Comp}_i=\{A\in\mathrm{NP}\mid \mathrm{SAT}=L(P_i^A)\}$. Using this decomposition, we show that Comp is of the first Baire category. Note that for every integer i and for every string $w\in X^*$, there exists a string y such that $P_i^{w1^{|y|^i+i}}(y)\neq\mathrm{SAT}(y)$ (otherwise SAT would be in P), where $w1^{|y|^i+i}$ denotes the concatenation of w with a number of $|y|^i+i$ 1's and P^z, $z\in X^*$ means that the oracle is the finite set encoded by string z. Consider the recursive function $f\colon N\times X^*\to X^*$ which, on input (i, w), acts as follows: for all $v\in X^*$, with $|v|=|w|$ it finds a corresponding $y(v)$ such that $P_i^{v1^{|y(v)|^i+i}}(y(v))\neq\mathrm{SAT}(y(v))$, and then it selects the longest such $y(v)$ over all v with $|v|=|w|$, let it be $y(0)$. We set $f(i,w)=w1^{|y(0)|^i+i}$. Clearly, w is a prefix of $f(i,w)$. Also $U_{f(i,w)}\cap\mathrm{Comp}_i=\emptyset$. Indeed, let $A\in U_{f(i,w)}$ be an infinite set and let v be the initial segment of length $|w|$ of the characteristic sequence of A. There is a string $y(v)$, $|y(v)|\leqslant|y(0)|$ such that $P_i^{v1^{|y(v)|^i+i}}(y(v))\neq\mathrm{SAT}(y(v))$. As P_i, on input $y(v)$, does not ask the oracle for a string longer than $|y(v)|^i+i$ and since $v1^{|y(v)|^i+i}$ is a prefix of the characteristic sequence of A, it follows that $P_i^A(y(v))\neq\mathrm{SAT}(y(v))$. Consequently, $A\notin\mathrm{Comp}_i$. We have shown that Comp_i is recursively rare; hence Comp is of thc first category. □

By inspecting the proof, the result in Theorem 2.10 holds for the class of NP-hard sets. As a corollary, we obtain a stronger form of a well-known result of Ladner [15] stating the existence of a set which is neither in P- nor NP-complete, under the hypothesis that $\mathrm{NP}\neq\mathrm{P}$. The following corollary shows that there exist many sets with the above property.

Corollary 2.11. *If* $\mathrm{NP}\neq\mathrm{P}$, *then* $(\mathrm{NP}-\mathrm{P})-\mathrm{Comp}$ *is a class of the second category.*

Proof. Suppose that $(\mathrm{NP}-\mathrm{P})-\mathrm{Comp}$ is of the first category. One can easily show that the union of two classes of first category is still a class of the first category. Then by Theorem 2.10, it would follow that $\mathrm{NP}-\mathrm{P}$ is of the first category. But this contradicts Theorem 2.5. □

We investigate next the topological size of the low and high hierarchies [20], two hierarchies related to the polynomial hierarchy. For each $k\leqslant 0$, $L_k=\{A\in\mathrm{NP}\mid\Sigma_k^{\mathrm{P}}(A)\subset\Sigma_k^{\mathrm{P}}\}$ is the low hierarchy in NP and $H_k=\{A\in\mathrm{NP}\mid\Sigma_{k+1}^{\mathrm{P}}\subset\Sigma_k^{\mathrm{P}}(A)\}$ is the high hierarchy in NP. It can be seen that the sets lying in the low hierarchy are more or less (it depends on the level k) similar to sets in P, while the sets in the high hierarchy are similar to NP-complete sets. For example, $L_0=\mathrm{P}$, $L_1=\mathrm{NP}\cap\text{Co-NP}$, $H_0=\{A\mid A$ is $\leqslant_{\mathrm{T}}^{\mathrm{P}}$-complete for NP$\}$, $H_1=\{A\mid A$ is $\leqslant_{\mathrm{T}}^{\mathrm{sn}}$-complete for NP$\}$, where $A\leqslant_{\mathrm{T}}^{\mathrm{sn}}B$ means $A\in\mathrm{NP}(B)\cap\text{Co-NP}(B)$.

Theorem 2.12. *For all* $k\geqslant 0$, *if* $L_{k+1}-L_k\neq\emptyset$, *then* $L_{k+1}-L_k$ *is a class of the second category.*

Proof. By Lemma 2.1 it is sufficient to show that $\mathrm{Tot} \leqslant_{\mathrm{m}} (\mathrm{L}_{k+1} - \mathrm{L}_k, \text{Co-Fin})$. In order to achieve this, let $\{M_i\}_{i \in N}$ be an enumeration of all deterministic Turing machines such that $W_i = \mathrm{dom}(M_i)$, $\{R_j\}_{j \in N}$ be an enumeration of all Σ_k^{p}-machines (i.e. k-alternating machines starting in an existential state [10]) and A be a fixed set in $\mathrm{L}_{k+1} - \mathrm{L}_k$. Thus, $\Sigma_{k+1}^{\mathrm{p}}(A) = \Sigma_{k+1}^{\mathrm{p}}$ and $\Sigma_k^{\mathrm{p}}(A) \neq \Sigma_k^{\mathrm{p}}$. For any set B, let $C_k(B) = \{i \# x \# 0^{|x|^i + i} \mid$ the oracle Σ_k^{p}-machine R_i accepts x, when working with oracle $B\}$, i.e. $C_k(B)$ is the canonical $\leqslant_{\mathrm{m}}^{\mathrm{p}}$-complete set for $\Sigma_k^{\mathrm{p}}(B)$. It is clear that $\Sigma_k^{\mathrm{p}}(B) \neq \Sigma_k^{\mathrm{p}}$ implies $C_k(B) \in \Sigma_k^{\mathrm{p}}(B) - \Sigma_k^{\mathrm{p}}$. For every deterministic Turing machine M_i we define a nondeterministic polynomial-time Turing machine $\mathrm{NP}_{s(i)}$, such that if $i \in \mathrm{Tot}$ then $\mathrm{NP}_{s(i)} \in \mathrm{L}_{k+1} - \mathrm{L}_k$ and if $i \notin \mathrm{Tot}$ then $\mathrm{NP}_{s(i)}$ accepts a co-finite language. The condition $\mathrm{NP}_{s(i)} \in \mathrm{L}_{k+1} - \mathrm{L}_k$ is realized by imposing to $\mathrm{NP}_{s(i)}$ to satisfy two requirements: (i) $\mathrm{NP}_{s(i)} \leqslant_{\mathrm{T}}^{\mathrm{p}} A$, and (ii) $C_k(\mathrm{NP}_{s(i)}) \notin \Sigma_k^{\mathrm{p}}$. In this way, $\Sigma_{k+1}^{\mathrm{p}}(\mathrm{NP}_{s(i)}) \subset \Sigma_{k+1}^{\mathrm{p}}(A) = \Sigma_{k+1}^{\mathrm{p}}$ and $\Sigma_k^{\mathrm{p}}(\mathrm{NP}_{s(i)}) \neq \Sigma_k^{\mathrm{p}}$, because $C_k(\mathrm{NP}_{s(i)}) \in \Sigma_k^{\mathrm{p}}(\mathrm{NP}_{s(i)}) - \Sigma_k^{\mathrm{p}}$. Hence, $\mathrm{NP}_{s(i)} \in \mathrm{L}_{k+1} - \mathrm{L}_k$. The construction of $s(i)$ is based on the same technique used in Lemma 2.2. The $\mathrm{NP}_{s(i)}$ performs its computations in stages, starting with Stage 0. On input x, of length n, $\mathrm{NP}_{s(i)}$ runs as follows:

(a) For n steps, $\mathrm{NP}_{s(i)}$ simulates in a deterministic way the previous computations and determines how many stages have been completed so far.

(b) *Case* 1: Stage $= 2e$. For n steps, $\mathrm{NP}_{s(i)}$ simulates M_i on input a_e. If $M_i(a_e)$ does not stop within the allowed time or M_i rejects a_e then $\mathrm{NP}_{s(i)}$ accepts x and stops. Otherwise, x is accepted and Stage $:= 2e + 1$.

Case 2: Stage $= 2e+1$. Let U be the oracle Σ_k^{p}-machine that accepts $C_k(\mathrm{NP}_{s(i)})$ by the help of the oracle $\mathrm{NP}_{s(i)}$ and let T be the machine that deterministically simulates U and that replaces oracle queries by deterministic computation of the sets $\mathrm{NP}_{s(i)}$ and A (we need A because $\mathrm{NP}_{s(i)}$ depends on A by the final phase of its construction). For n steps, $\mathrm{NP}_{s(i)}$ looks for a y such that $T(y)$ stops (this implicitly means that the deterministic computation of $\mathrm{NP}_{s(i)}$ stops on all strings queried by U) and $T(y) \neq R(y)$. If such a y is found, then the current stage becomes $2e+1$, $\mathrm{NP}_{s(i)}$ accepts x and it stops. If no such y is found in due time, then x is accepted if and only if $x \in A$. This completes the construction.

Clearly, $\mathrm{NP}_{s(i)} \leqslant_{\mathrm{T}}^{\mathrm{p}} A$. One can see that if $i \in \mathrm{Tot}$ then $C_k(\mathrm{NP}_{s(i)}) \notin \Sigma_k^{\mathrm{p}}$. Indeed, if the current stage stabilizes itself at $2e+1$, then from this stage on $\mathrm{NP}_{s(i)}$ looks like A. However $C_k(A)$ differs from R_e in infinitely many points since otherwise we get that $C_k(A) \in \Sigma_k^{\mathrm{p}}$ which implies $\Sigma_k^{\mathrm{p}}(A) = \Sigma_k^{\mathrm{p}}$. Hence, $\mathrm{NP}_{s(i)}$ finds eventually a string y that satisfies the condition from (b) Case 2. But in this case the value of the current stage is incremented which contradicts our assumption. On the other hand, if $i \notin \mathrm{Tot}$, then it is easy to observe that $\mathrm{NP}_{s(i)}$ accepts a co-finite set. □

What is the topological size of the classes H_k, $k \geqslant 0$, from the high hierarchy? As expected, they behave like Comp (which in fact is just H_0).

Theorem 2.13. *For all* $k \geqslant 0$, *if* $\Sigma_{k+1}^{\mathrm{p}} \neq \Sigma_k^{\mathrm{p}}$, *then* H_k *is a class of the first category.*

Proof. For all $k \geqslant 0$, let C_k be a complete set for Σ_k^{p}. Then $\mathrm{H}_k = \{A \in \mathrm{NP} \mid \Sigma_k^{\mathrm{p}}(A) = \Sigma_{k+1}^{\mathrm{p}}\} \subset \bigcup_{(i(1), i(2), \ldots, i(k)) \in N^k} \{A \in \mathrm{NP} \mid C_{k+1} = \mathrm{NP}_{i(1)}(\mathrm{NP}_{(i(2))} \ldots (\mathrm{NP}_{(i(k))}(A) \ldots)\}$ (the union is taken over all k-tuples $(i(1), i(2), \ldots, i(k))$ in N^k) because if $A \in \mathrm{H}_k$ then $C_{k+1} \in \Sigma_{k+1}^{\mathrm{p}} = \Sigma_k^{\mathrm{p}}(A)$. Now, by performing some slight and obvious modifications in the proof of Theorem 2.10, one can see that H_k is indeed of the first category. □

Corollary 2.14. *For all $k \geqslant 0$, if $\Sigma_{k+1}^{\mathrm{p}} \neq \Sigma_k^{\mathrm{p}}$ then $\mathrm{NP} - (\mathrm{L}_k \cup \mathrm{H}_k)$ is a class of the second category.*

Proof. If $\Sigma_{k+1}^{\mathrm{p}} \neq \Sigma_k^{\mathrm{p}}$, then $\mathrm{NP} \neq \mathrm{P}$ [5]. In this case $\mathrm{SAT} \in \mathrm{NP} - \mathrm{L}_k$. Taking SAT in the place of A in the proof of Theorem 2.12, one can show that $\mathrm{NP} - \mathrm{L}_k$ is of the second category. By Theorem 2.13, H_k is a class of the first category. Hence $\mathrm{NP} - (\mathrm{L}_k \cup \mathrm{H}_k)$ is of the second category. □

3. Immunity and simplicity

In this section we investigate the topological size of some classes of sets achieving the separation of P from NP in a stronger form like the class of P-immune sets or NP-simple sets. For a class C of sets, a set A is C-immune if A is an infinite set and no infinite subset of A is in C. A set is C-simple if A belongs to C and the complement of A is C-immune. A C-simple set A is called effectively C-simple if there exists a recursive function $f: N \rightarrow N$ such that C_k included in $\bar{A}$ implies $|C_k| \leqslant f(k)$, where $C = (C_k)_{k \in N}$ is an effective enumeration of C. If C is the class of r.e. sets, we simply say immune, simple or effectively simple instead of r.e.-immune a.s.o. All these notions are inherited from classical recursive function theory [22, 18] and the versions where C is P or NP have been intensively studied in structural complexity theory [21, 3, 6].

Theorem 3.1. *If there exists a* P*-immune set in* NP, *then the class of* P*-immune sets in* NP *is of the second category.*

Proof. Let $A = \{B \in \mathrm{NP} \mid B \text{ is P-immune}\}$. By hypothesis, A is not empty, so we fix a set H in A. Our intention is to construct for each i a nondeterministic machine $\mathrm{NP}_{s(i)}$ such that if $i \in \mathrm{Tot}$ then $\mathrm{NP}_{s(i)} \in A$ and if $i \notin \mathrm{Tot}$ then the language accepted by $\mathrm{NP}_{s(i)}$ is co-finite. Then by Lemma 2.1 the result follows. During the construction we use the variables Stage, Next-Candidate and a list, called List, which stores indexes of polynomial-time deterministic machines that have been considered but have not yet been diagonalized over. List is thought as an array of integers, so we can speak about the first element of List, about the second a.s.o. Insertions in List are made at the bottom of List (in the last position), deletions can be made anywhere in List, but after a deletion the List is compactified so as not to contain any gap. We shall take care not to increase the size of List too much so that insertions and deletions can be realized in linear time. The variable Next-Candidate keeps the index j of the next polynomial-time deterministic machine P_j that we attempt to introduce in List. Stage is a variable that records the current stage in the construction of $\mathrm{NP}_{s(i)}$. If Stage has an even value, Stage$=2e$, then $\mathrm{NP}_{s(i)}$ tries to find if M_i accepts a_e. When this happens, Stage is incremented to the value $2e+1$. If M_i does not accept a_e, then Stage remains perpetually at the value $2e$, causing $\mathrm{NP}_{s(i)}$ to accept all its further inputs. If Stage has an odd value, then $\mathrm{NP}_{s(i)}$ tries to find for each k in List a string z such that $z \in \mathrm{P}_k \cap \overline{\mathrm{NP}}_{s(i)}$. In this attempt, $\mathrm{NP}_{s(i)}$ offers to each element of List a time which is proportional to its position in List, without exceeding n steps for the whole operation, where n is the length of the current input of $\mathrm{NP}_{s(i)}$. Specifically, if k is in position j, $n/2^j$ steps are spent in the effort of finding the desired string z. However, as $\mathrm{NP}_{s(i)}$ processes increasingly longer inputs, if P_k is infinite, there will be eventually enough time to discover a z such that $z \in \mathrm{P}_k \cap \overline{\mathrm{NP}}_{s(i)}$. The existence of such a string z follows by the P-immunity of H and by the mechanism of delayed diagonalization which makes $\mathrm{NP}_{s(i)}$ resemble H in case of the failure of discovering the suitable string z.

The construction of $\mathrm{NP}_{s(i)}$: $\mathrm{NP}_{s(i)}$ performs its computations in stages. Initially Stage:$=0$, Next-Candidate:$=0$, List:$=0$. On input $x \in X^*$, of length n, $\mathrm{NP}_{s(i)}$ runs as follows:

(a) For n steps $\mathrm{NP}_{s(i)}$ simulates deterministically the previous computations (i.e. it computes $\mathrm{NP}_{s(i)}(a_1)$, $\mathrm{NP}_{s(i)}(a_2)\cdots$for as many inputs the time bound allows) and determines the current values of Stage, Next-Candidate and the content of List.

(b) *Case* 1: Stage$=2e$. The $M_i(a_e)$ is simulated for n steps. If M_i accepts a_e in the specified time then x is accepted, Stage:$=2e+1$ and $\mathrm{NP}_{s(i)}$ stops.

Case 2: Stage$=2e+1$. Let $m=|\mathrm{List}|$, i.e. m is the number of elements in List. If $m>n$ then x is rejected and $\mathrm{NP}_{s(i)}$ stops. Otherwise Next-Candidate is inserted in List and Next-Candidate:$=$ Next-Candidate$+1$.

For $j \in \overline{1,m}$, let List$[j]$ denote the value of the jth element in List. Then for every $j \in \overline{1,m}$, for $n/2^j$ steps, $\mathrm{NP}_{s(i)}$ looks for a string z such that $z \in \mathrm{P}_{\mathrm{List}[j]}$ and $z \notin \mathrm{NP}_{s(i)}$. In doing this, $\mathrm{NP}_{s(i)}$ is simulated in a deterministic way.

Case 2.1: The search succeeds for some j. Then for all such j's, List$[j]$ is deleted from List, x is accepted and Stage:$=2e+2$.

Case 2.2: The search fails for all j. Then x is accepted if and only if $x \in H$.

End of construction of $\mathrm{NP}_{s(i)}$.

Now the proof follows by the next series of Claims.

Claim 3.2. $\mathrm{NP}_{s(i)}$ *is a polynomial-time nondeterministic machine.*

Proof. The only nondeterministic step in the computation of $\mathrm{NP}_{s(i)}(x)$ occurs in (b) case 2.2. This step is realized by simulating the polynomial-time nondeterministic machine that accepts H. All the other operations are performed in a deterministic way in polynomial time (in fact linear time). Observe that the size of List is not allowed to increase too much, so that the operations of insertion and deletion can be realized in linear time in the size of the input $|x|$. □

Claim 3.3. *Suppose that at a certain moment in the construction of* $\mathrm{NP}_{s(i)}$, Stage $=2e$ *and M_i accepts a_e. Then there exists a later moment when* Stage$:=2e+1$.

Proof. This follows immediately because for a long enough string x $\mathrm{NP}_{s(i)}$ has sufficient time to simulate the accepting computation of M_i on input a_e. □

Claim 3.4. *Suppose that at a certain moment in the construction of* $\mathrm{NP}_{s(i)}$, Stage $=$ $2e+1$. *Then there exists a later moment when* Stage *is increased to* $2e+2$.

Proof. Let us suppose the contrary. Clearly, there exists a moment when a k, such that P_k is infinite, is inserted in List. Since H is P-immune, $\mathrm{P}_k \cap \bar{H} \neq \emptyset$. Moreover, $\mathrm{P}_k \cap \bar{H}$ is an infinite set. Indeed if $\mathrm{P}_k \cap \bar{H} = B$, B finite, then $\mathrm{P}_k - B \subset H$, but P_k is an infinite set in P. This contradicts the P-immunity of H. By our assumption, it follows that there exists a string x_0 such that for $x \geqslant x_0$, $\mathrm{NP}_{s(i)}$ accepts x iff $x \in H$. Hence $\mathrm{NP}_{s(i)} = H$ a.e. In this case for a sufficiently long input string x, $\mathrm{NP}_{s(i)}$ has enough time to discover a string z such that $z \in \mathrm{P}_k \cap \overline{\mathrm{NP}}_{s(i)}$. But this implies the incrementation of Stage to the value $2e+2$. □

Claim 3.5. *If* $i \in \mathrm{Tot}$, *then* $\mathrm{NP}_{s(i)}$ *accepts a co-finite language.*

Proof. Let e be the minimal with the property that M_i does not accept a_e. By Claims 3.3 and 3.4, it follows that Stage reaches the value $2e$. It is clear that starting with this moment $\mathrm{NP}_{s(i)}$ accepts every input. □

Claim 3.6. *If* $i \in \mathrm{Tot}$ *and* P_k *is infinite, then* $\mathrm{P}_k \cap \overline{\mathrm{NP}}_{s(i)} \neq \emptyset$.

Proof. Since $i \in \mathrm{Tot}$, it follows by Claims 3.3 and 3.4 that there exists a moment when k is inserted in List. The reasoning in the proof of Claim 3.4 shows that $\mathrm{P}_k \cap \overline{\mathrm{NP}}_{s(i)} \neq \emptyset$. □

Claim 3.7. $\mathrm{NP}_{s(i)}$ *is infinite.*

Proof. If $i \notin \mathrm{Tot}$, we use Claim 3.5. If $i \in \mathrm{Tot}$, then Claims 3.3 and 3.4 show that the value of Stage passes through all positive integers. But any increase of Stage from an even value is done in (b), case 1 and implies also the acceptance of the current input string x. □

The analogous result for the case of NP-simple sets can be derived more simply.

Theorem 3.8. *If there exists an* NP-*simple set, then the class of* NP-*simple sets is a class of the second Baire category.*

Proof. The proof relies again on Lemma 2.1. We fix an NP-simple set H and then for each integer i we define $\mathrm{NP}_{s(i)}$ such that (a) $i \in \mathrm{Tot}$ implies H is included in $\mathrm{NP}_{s(i)}$ and $\overline{\mathrm{NP}}_{s(i)}$ is infinite, and (b) $i \notin \mathrm{Tot}$ implies $\mathrm{NP}_{s(i)}$ accepts a co-finite set. Note that situation (a) implies that $\mathrm{NP}_{s(i)}$ is NP-simple, because if NP_k is infinite then $\mathrm{NP}_k \cap H \neq \emptyset$, so $\mathrm{NP}_k \cap \mathrm{NP}_{s(i)} \neq \emptyset$.

Construction of $\mathrm{NP}_{s(i)}$: (a) For n steps $\mathrm{NP}_{s(i)}$ simulates deterministically the previous computations $\mathrm{NP}_{s(i)}(a_1)$, $\mathrm{NP}_{s(i)}(a_2), \ldots$ for as many inputs the time bound allows. It may happen that on some inputs a_k the simulation of $\mathrm{NP}_{s(i)}(a_k)$ finds different values for the variable Stage on the originally nondeterministic branches of the computation of $\mathrm{NP}_{s(i)}(a_k)$. If this is the case the least such value is selected for Stage.

(b) *Case* 1: Stage $= 2e$. Then Stage$:= 2e+1$ and the polynomial-time nondeterministic machine N that accepts H is started on input x. If a computation of $N(x)$ that accepts x is discovered, then $\mathrm{NP}_{s(i)}$ accepts x and Stage comes back to the value $2e$. Otherwise $\mathrm{NP}_{s(i)}$ rejects x. After these operations $\mathrm{NP}_{s(i)}$ stops.

Case 2: Stage $= 2e+1$. For n steps $\mathrm{NP}_{s(i)}$ simulates M_i on input a_e. If M_i accepts a_e in the specified time, then x is accepted, Stage$:= 2e+2$ and $\mathrm{NP}_{s(i)}$ stops. Otherwise $\mathrm{NP}_{s(i)}$ accepts x and stops.

This completes the construction.

Suppose that $i \in \mathrm{Tot}$. Then the value of the variable Stage passes through all positive integers k. This is proved by induction on k. Indeed, suppose $k = 2e$. There exists a moment when the input x is such that $x \notin H$, because $\bar{H}$ is infinite. At this moment, Stage becomes $2e+1$ and is never decreased later. In case $k = 2e+1$, since $i \in \mathrm{Tot}$, we conclude that for a sufficiently long input x, $\mathrm{NP}_{s(i)}$ has enough time to discover that M_i accepts a_e and, consequently, to increase Stage to the value $2e+2$. But any permanent increase of Stage in (b) case 1 implies that $\mathrm{NP}_{s(i)}$ rejects the current input. Hence, $\overline{\mathrm{NP}}_{s(i)}$ is infinite. On the other hand, H is included in $\mathrm{NP}_{s(i)}$ because $\mathrm{NP}_{s(i)}$ rejects an input x only in case $x \notin H$ (see (b) case 1). Clearly, $\mathrm{NP}_{s(i)}$ is computed by a polynomial-time nondeterministic machine. We conclude that $\mathrm{NP}_{s(i)}$ is NP-simple.

In case $i \notin \mathrm{Tot}$, it is easy to see that Stage stabilizes itself to the value $2e+1$, where a_e is the minimal string that M_i does not accept, and from that moment on $\mathrm{NP}_{s(i)}$ accepts all further input strings. Hence, in this case $\mathrm{NP}_{s(i)}$ is co-finite. □

We can strengthen the above results by considering bi-immune for P-sets [5]. An infinite co-infinite set A is bi-immune for P if both A and its complement are P-immune.

Theorem 3.9. *If there exists a bi-immune for* P *set in* NP, *then the class of bi-immune for* P *sets in* NP *is a class of the second category.*

Proof. By a slight modification of the proof of Theorem 3.1, which we sketch here. This time we use two lists, List1 and List2. The construction of $\mathrm{NP}_{s(i)}$ is done in stages and three situations occur depending on whether the current stage k is of the form $3e$, $3e+1$ or $3e+2$. In case $k=3e$, $\mathrm{NP}_{s(i)}$ simulates $M_i(a_e)$ as in (b) case 1 in the proof of Theorem 3.1. In case $k=3e+1$, $\mathrm{NP}_{s(i)}$ looks for a z such that $z \in \mathrm{P}_{\mathrm{m}} \cap \overline{\mathrm{NP}_{s(i)}}$, for m in List1, whereas in case $k=3e+2$, $\mathrm{NP}_{s(i)}$ looks for a z such that $z \in \mathrm{P}_{\mathrm{m}} \cap \mathrm{NP}_{s(i)}$ for m in List2. The other details are left to the reader. □

One can observe that the technique used in Theorem 3.1 for the case of P-immune sets would have worked also for the case of NP-simple sets. We have preferred to present another proof (which incidentally is more simple) because the method used in Theorem 3.2 can be applied for the case of effectively NP-simple sets.

Proposition 3.10. *If there exists an effectively* NP*-simple set in* NP, *then the class of effectively* NP*-simple sets in* NP *is of the second category.*

Proof. Practically, the proof of Theorem 3.8 works here too. □

Proposition 3.10 has the following interesting consequence. Either there is no effectively NP-simple set in NP or there are effectively NP-simple sets that are not NP-complete, because the latter class is of the first Baire category. This statement contrasts sharply with the situation in recursive function theory where effectively simple sets are complete for the class of r.e. sets [22, pp. 79–87]. This result shows that our topological analysis has some interesting applications in the study of the structural properties of some important complexity classes. For example, by combining Theorem 3.1 with Theorem 2.10, we obtain that if there exists P-immune sets in NP, then there exists such a set that is not NP-complete. By Theorem 3.8, a similar result holds for NP-simple sets also. Note that the problem of whether there exist NP-complete sets that are NP-simple or NP-hard sets that are NP-immune has been investigated in [14]. It is shown in [14] that if NP is exponentially hard (which is very likely), no NP-hard set can be NP-immune and if NP$\cap$Co-NP is exponentially hard (which again is very likely), NP-complete NP-simple sets do not exist.

4. Some applications

One may argue that the topology proposed in this paper is exotic and that the results obtained in the previous sections are not relevant outside the point of view of this topology. In this section we want to counteract this claim. We show that all classes C of sets shown to be of second category have a very interesting property: for every B in NP there exists $A \in C$ such that for an arbitrarily slowly increasing, unbounded and recursive function $r: N \to N$, there exists $A \in C$ such that

$|(A \Delta B)^{\leqslant n}| \leqslant r(n)$, for infinitely many n, where $A \Delta B$ is $(A-B) \cup (B-A)$. We believe that this property shows that the classes of the second Baire category are really large and, consequently, the proposed topology is validated by intuitive results. Moreover, this property is obtained by using a topology similar to the one in the previous sections.

For the rest of this section we fix a set B in NP. For every $w \in X^*$ we define a basic neighborhood as

$$U_w^B = \{v \in X^\infty \mid \forall i \leqslant |w| \, (a_i \in B,\ w(i)=1 \Rightarrow v(i)=1) \text{ and }$$
$$(a_i \notin B,\ w(i)=0 \Rightarrow v(i)=0))\}.$$

Note that $U_w = U_w^{X^*}$, so that the topology used in the previous sections is just a particular case of the topology introduced here. The following definition is motivated as in the Introduction.

Definition 4.1. (a) A class C included in X^∞ is recursively of the first Baire category relative to B if there is a decomposition $C = \bigcup_{i \geqslant 0} C_i$ and a recursive function $f: N \times X^* \to X^*$ such that, for every natural i and for every $w \in X^*$,

(i) w is a prefix of $f(i, w)$ and

(ii) $U^B_{f(i,w)} \cap C_i = \emptyset$.

(b) A class C included in X^∞ is recursively of the second Baire category relative to B if it is not recursively of the first Baire category relative to B.

As before, we shall drop the words "recursively" and "Baire" in the further use of this terminology.

A result similar to Lemma 2.1 holds. Indeed, let C included in NP be a class of sets of the first category relative to B via a function $f: N \times X^* \to X^*$. Hence, if $A \in C$, there is an integer i such that $A \in C_i$ and, consequently, for every w, $A \notin U^B_{f(i,w)}$. If $\bar{B}(\leqslant n)$ is the string $\bar{B}(a_1)\bar{B}(a_2)\ldots\bar{B}(a_n)$, we deduce that for every $n \in N$ there exists $x \in N$, $n < x \leqslant |f(i, \bar{B}(\leqslant n))|$ such that $a_x \in (A \Delta B)$, because, otherwise, $A \in U^B_{f(i, \bar{B}(\leqslant n))}$. Consequently, if $A \in C$, then

$$A \in D_f = \{j \mid (\exists i \in N)(\forall n \in N)(\exists x \in N,\ n < x \leqslant |f(i, \bar{B}(\leqslant n))|,\ a_x \in \mathrm{NP}_j \Delta B)\}.$$

Here NP_j denotes the language accepted by the machine NP_j. Observe that $D_f \in \Sigma_2$ in the Kleene's arithmetic hierarchy. In consequence, similarly to Lemma 2.1, we obtain the following lemma.

Lemma 4.2. *Let C be a class included in* NP. *If for every recursive function $f: N \times X^* \to X^*$* Tot $\leqslant_m (C, \bar{D}_f)$, *then C is a class of sets of the second category relative to B.*

All the classes shown to be of the second category in the previous sections are in fact of the second category relative to any B in NP. We sketch the proof for the case $A = \mathrm{NP} - \mathrm{P}$, but the same modification works in all the other cases.

Lemma 4.3. *Let $f: N \times X^* \to X^*$ be a recursive function. Then* Tot $\leqslant_{\mathrm{m}}(\mathrm{NP}-\mathrm{P}, \bar{D}_f)$, *provided that* $\mathrm{NP} \neq \mathrm{P}$.

Proof. For every Turing machine M_i we construct a polynomial-time nondeterministic machine $\mathrm{NP}_{s(i)}$ such that $i \in \mathrm{Tot}$ implies $\mathrm{NP}_{s(i)} \in \mathrm{NP}-\mathrm{P}$ and $i \notin \mathrm{Tot}$ implies $s(i) \in \bar{D}_f$. The construction of $\mathrm{NP}_{s(i)}$ is performed in stages. On an even Stage $= 2e$, we simulate M_i on input a_e and, if M_i does not accept a_e in the allowed time, then we try to satisfy, for some $h \in N$, the following requirement:

$$R_h\colon\ (\exists n \in N)(\forall x \in N,\ n < x \leqslant |f(h, \bar{B}(\leqslant n))|,\ a_x \notin (\mathrm{NP}_{s(i)} \Delta B).$$

To this aim we use the variable Hcrt which stores the value h for which we try to satisfy R_h. For such an h, we store the attempted value of n which appear in R_h in the variable Ncrt. If Ncrt $= 0$, we have to start satisfying the requirement R_h from the very beginning, i.e. for all a_x with $n < x \leqslant f(h, \bar{B}(\leqslant n))$. On an odd stage, Stage $= 2e+1$, we satisfy $\mathrm{NP}_{s(i)} \neq \mathrm{P}_e$ exactly as in Lemma 2.2, by delayed diagonalization.

The construction of $\mathrm{NP}_{s(i)}$: Initially, Stage:$=0$, Hcrt:$=0$, Ncrt:$=0$. On input $x \in X^*$, $x = a_m$, $|x| = n$, $\mathrm{NP}_{s(i)}$ acts as follows:

(a) For n steps, $\mathrm{NP}_{s(i)}$ simulates deterministically the previous computations $\mathrm{NP}_{s(i)}(a_1)$, $\mathrm{NP}_{s(i)}(a_2), \ldots$ and determines the values of the variables Stage. Hcrt and Ncrt.

(b) *Case* 1: Stage $= 2e$. For n steps, M_i is simulated on input a_e.

If M_i does not accept a_e within n steps then $\mathrm{NP}_{s(i)}$ performs the following operations:

(b.1) If Ncrt $= 0$ then Ncrt:$= m$.

(b.2) $\mathrm{NP}_{s(i)}$ computes $f(\mathrm{Hcrt}, \bar{B}(\leqslant \mathrm{Ncrt}))$. The computation of B is simulated in a deterministic way. If this computation stops within n steps and $m > f(\mathrm{Hcrt}, \bar{B}(\leqslant \mathrm{Ncrt}))$, then Hcrt:$=$ Hcrt $+ 1$, Ncrt:$= 0$ and $\mathrm{NP}_{s(i)}$ accepts and stops. Otherwise, x is accepted if and only if $x \in B$.

In case M_i accepts a_e within n steps then Hcrt:$=0$, Ncrt:$=0$ and Stage:$=2e+1$.

Case 2: Stage $= 2e+1$. $\mathrm{NP}_{s(i)}$ looks for a $z < x$ such that $\mathrm{NP}_{s(i)} \neq \mathrm{P}_e(z)$ for n steps. In doing this the computation of $\mathrm{NP}_{s(i)}$ on various inputs z is simulated in a deterministic way. If such a z is found then x is accepted and Stage:$= 2e+2$. Otherwise, x is accepted if and only if $x \in \mathrm{SAT}$.

This completes the construction of $\mathrm{NP}_{s(i)}$.

Claim 4.4. $\mathrm{NP}_{s(i)}$ *is computed in nondeterministic polynomial time.*

Proof. Nondeterministic computations occur for simulating B and SAT on the input string x. All other steps are done in linear deterministic time. □

Claim 4.5. *If* $i \notin \mathrm{Tot}$, *then for all naturals* h *the requirement* R_h *is fulfilled.*

Proof. Let e be minimal such that M_i does not accept a_e. It is easily seen that, starting with the moment when $\mathrm{NP}_{s(i)}$ enters for the first time in Stage $= 2e$ after the computation of (a), $\mathrm{NP}_{s(i)}$ satisfies, one after another, all the requirements R_h. $\square$

Claim 4.6. *If* $i \in \mathrm{Tot}$, *then* $\mathrm{NP}_{s(i)} \in \mathrm{NP} - \mathrm{P}$.

Proof. If $i \in \mathrm{Tot}$, then the variable Stage takes the value $2e+1$ for all $e \in N$. At the Stage $= 2e+1$, the requirement $\mathrm{NP}_{s(i)} \neq \mathrm{P}_e$ is fulfilled. Taking into consideration Claim 4.4, it follows that $\mathrm{NP}_{s(i)} \in \mathrm{NP} - \mathrm{P}$. $\square$

Corollary 4.7. *For any* $B \in \mathrm{NP}$, $\mathrm{NP} - \mathrm{P}$ *is a class of the second category relative to* B, *provided* $\mathrm{NP} \neq \mathrm{P}$. $\square$

For any infinite set D included in X^*, we define the principal function of D, $\#D: N_+ \to N_+$ by the relation (see [22, p. 81]): $D = (a_{\#D(1)} < a_{\#D(2)} < \cdots < a_{\#D(n)} < \cdots\}$. Observe that for all integer n, among the first $\#D(n)$ strings in X^*, there are exactly n strings that belong to D. Principal functions are similar to ranking functions [1, 11].

Theorem 4.8. *Let* $B \in \mathrm{NP}$ *and* C *included in* NP *be a class of the second category relative to* B. *Then either there exists* A *in* C *such that* $A \,\Delta\, B$ *is finite or for any recursive function* $f: N \to N$ *there exists* A *in* C *such that* $\#(A \,\Delta\, B)(n) \geqslant f(n)$ *for infinitely many* n *in* N.

Proof. Suppose that for any A in C, $A \,\Delta\, B$ is infinite and there exists a recursive function $f: N \to N$ such that $\#(A \,\Delta\, B)(n) < f(n)$ for sufficiently large n. Let $C_i = \{A \in C \mid \#(A \,\Delta\, B)(n) < f(n),\ \forall n \geqslant i\}$. Then $C = \bigcup_{i \geqslant 0} C_i$. For $n \geqslant i$, in the set $\{a_1, a_2, \ldots, a_{\#(A \Delta B)(n)}\}$ there are exactly n strings from $A \,\Delta\, B$ for any A in C_i. Hence, for $n \geqslant i$, in the set $\{a_{n+1}, a_{n+2}, \ldots, a_{\#(A \Delta B)(n+1)}, \ldots a_{f(n+1)}\}$ there exists at least one element from $A \,\Delta\, B$, for any A in C_i. Now we define $g: N \times X^* \to X^*$ by

$$g(i, w) = \begin{cases} wB(|w|+1)B(|w|+2)\ldots B(f(|w|+1)), & \text{if } |w| > i \\ w1^{i-|w|}B(i+1)\ldots B(f(i+1)), & \text{if } |w| \leqslant i. \end{cases}$$

By the above observation $C_i \cap U^B_{g(i,w)} = \emptyset$. This contradicts the fact that C is of the second category relative to B. $\square$

Corollary 4.9. *Let* $B \in \mathrm{P}$ *and* C *included in* NP *be a class of the second category relative to* B. *Then for every* $r: N \to N$ *recursive, increasing and unbounded there exists* $A \in C$ *such that* $|(A \,\Delta\, B)^{\leqslant n}| \leqslant r(n)$ *for infinitely many* n.

Proof. If there exists A in C such that $A \,\Delta\, B$ is finite, the conclusion is immediate. In the opposite case, we consider the function r' defined by $r'(n) = r'(n-1)$ if $r(n) = r(n-1)$

and $r'(n)=r'(n-1)+1$ otherwise (i.e. $r(n)>r(n-1)$). It is clear that $r': N \to N$ is surjective and $r(n) \geqslant r'(n)$ for all naturals n. We also take a recursive function f such that $f(r'(n)) \geqslant 2^{n+1}-1$, for all n. From Theorem 4.8 we know that there exists a set A in C such that $\#(A \Delta B)(n) \geqslant f(n)$ for infinitely many n, hence $\#(A \Delta B)(r'(n)) \geqslant f(r'(n))$ for infinitely many n. By the definition of the principal function in the set $\{a_1, a_2, \ldots, a_{\#(A\Delta B)(r'(n))}\}$ there are exactly $r'(n)$ strings from $A \Delta B$, hence the cardinal of $\{a_1, a_2, \ldots, a_{f(r'(n))}\} \cap (A \Delta B)$ is less or equal than $r'(n) \leqslant r(n)$ for infinitely many n. Since $f(r'(n)) \geqslant 2^{n+1}-1$, the conclusion follows. □

Note that if $B=\emptyset$ and $C=\mathrm{NP}-\mathrm{P}$, we obtain the result that there exists a set in NP − P which is infinitely often sparse. By varying the class C, we deduce the existence of such a set that is P-immune or NP-simple.

The following theorem is the Baire category analogue of a result of Ambos-Spies [2], which characterizes the class P in terms of measure theory. It provides one additional reason for the adequacy of our definitions for the investigation of the topological size of various complexity classes.

Theorem 4.9. *A set A is in* P *if and only if the class $\{B \mid A \in \mathrm{P}(B)\}$ is of the second category.*

Proof. If A is in P then $\{B \mid A \in \mathrm{P}(B)\}$ is a class of the second category since it includes P. If A is not in P, then as in Theorem 2.10, one can show that $\{B \mid A \in \mathrm{P}(B)\}$ is a class of the first category.

Acknowledgments

I would like to thank Elvira Mayordomo for hints and valuable criticism, and Cristian Calude for lots of suggestions and encouragements.

References

[1] E. Allender and R. Rubinstein, P-printable sets, *SIAM J. Comput.* **17** (1988) 1193–1202.
[2] K. Ambos-Spies, Randomness, relativizations and polynomial reducibilities, in: *Proc. 1st Conf. Structure in Complexity Theory*, Lecture Notes in Computer Science, Vol. 223 (Springer, Berlin, 1986) 23–34.
[3] J. Balcazar, Simplicity, relativizations, and nondeterminism, *SIAM J. Comput.* **14** (1985) 148–157.
[4] J. Balcazar and J. Diaz, A note on a theorem of Ladner, *Inform. Process. Lett.* **15** (1982) 84–86.
[5] J. Balcazar, J. Diaz and J. Gabarro, *Structural Complexity, Vol. I* (Springer, Berlin, 1988).
[6] J. Balcazar and U. Schöning, Bi-immune sets for complexity classes, *Math. Systems Theory* **18** (1985) 1–10.
[7] R. Book, Comparing complexity classes: some observations on intrinsic characterizations, restricted relativizations, and uniform witnesses, Tech. Report, Univ. of California, Santa Barbara, 1989.

[8] C. Calude, Topological size of sets of partial recursive functions, *Z. Math. Logik Grundlag. Math.* **28** (1982) 455–462.

[9] C. Calude, *Theories of Computational Complexity* (North-Holland, Amsterdam, 1988).

[10] A. Chandra, D. Kozen and L. Stockmeyer, Alternation, *J. ACM* **28** (1981) 114–133.

[11] A. Goldberg and M. Sipser, Compression and ranking, in: *Proc. 25th Ann. ACM STOC* (1985) 440–448.

[12] J. Hartmanis, On the importance of being π_2-hard, *Bull. EATCS* **37** (1989) 117–127.

[13] J. Hartmanis, Independence results about context-free languages and lower bounds, *Inform. Process. Lett.* **20** (1985) 241–248.

[14] J. Hartmanis, M. Li and Y. Yesha, Containment, separation, complete sets, and immunity of complexity classes, in: *Proc. ICALP*, Lecture Notes in Computer Science, Vol. 226 (Springer, Berlin, 1986) 136–145.

[15] R. Ladner, On the structure of polynomial time reducibility, *J. ACM* **22** (1975) 155–171.

[16] J. Lutz, Category and measure in complexity classes, *SIAM J. Comput.* **19** (1990) 1100–1131.

[17] K. Mehlhorn, On the size of sets of computable functions, in: *Proc. Ann. IEEE Symp. on Switching and Automata Theory* (1973) 190–196.

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

[19] U. Schöning, *Complexity and Structure*, Lecture Notes in Computer Science, Vol. 211 (Springer, Berlin, 1985).

[20] U. Schöning, A low and a high hierarchy within NP, *J. Comput. Systems Sci.* **27** (1983) 14–28.

[21] U. Schöning and R. Book, Immunity, relativizations, and nondeterminism, *SIAM J. Comput.* **13** (1984) 329–337.

[22] R. Soare, *Recursively Enumerable Sets and Degrees* (Springer, Berlin, 1987).

[23] M. Zimand, Structural complexity theory: positive relativization and Baire classification, Ph.D. Thesis, Univ. of Bucharest, 1991; *Bull. EATCS* **45** (1991) 475–478.