University of Szeged
Research Group on Artificial Intelligence

Learnability and Characterization Results for Classes of Boolean Functions

Summary of the PhD Dissertation

by

Balázs Szörényi

Advisor:

György Turán

Szeged
2007

Introduction

The present dissertation, in its first part, considers theoretical results from the field of theory revision. Theory revision, as part of learning theory, is interested in reconstructing some unknown function $f_{\rm trg}$ (called the **target concept**), acquiring information about it via some protocol, specified by the given learning model. However, as opposed to the general learning problem, it is assumed that the learner is not new to the given task, but it initially has a hypotheses that is assumed to be some rough approximation of the unknown function. As an analogous real-world example, one can consider an initial version of an expert system provided by an expert, which needs to be refined using further examples or other information available. Having some initial hypotheses available should make the learning problem easier to solve—making the relevance of the model apparent, and motivating its analysis from the theoretical point of view.

The theory revision results in the present dissertation all consider some Boolean formula class; read-once, threshold and projective DNF formulas are analyzed from the point of view of efficient revisability.

In the second part characterizational results are presented; all showing equivalence between some syntactical and some semantical properties of some classes of Boolean functions. The syntactic properties involve Boolean formula classes, like DNFs satisfying some syntactic irredundancy notion, Horn formulas (one of the most studied formula class in artificial intelligence), disjoint DNFs (DNFs with pairwise conflicting terms) and decision trees (another very important object in computer science—which can also be thought of as a subclass of DNFs). The semantic properties include restrictions given for partitioning the n-dimensional cube with subcubes, special local restrictions given for a Boolean function on its domain, extensions of the truth set of some function fulfilling some special criteria, and finally some extremal properties.

The numbering of the theorems and lemmas, etc in this booklet is the same as that of the Dissertation. However, for the sake of keeping the presentation as simple as possible the formulation occasionally differs.

Theory Revision Results

Descriptions of theory revision systems are given, for example, in [17; 27; 28; 34; 35]. One of the first papers studying revision from a theoretical aspect is due to Mooney [25]. He assumed that the target can be obtained from the initial hypotheses by using **revision operators**, which are simple, predefined syntactic modifications, such as the deletion or the addition of a literal, and gave bounds for the the number of random examples needed in the PAC model for revision in terms of the number of these modifications necessary. Greiner [11] considered the computational complexity of hypothesis finding in a related framework.

The models for theory revision used in the present dissertation are extensions of Mooney's approach to two popular learning models: the query model and the mistake bounded model. In these settings the **revision distance** of the initial hypotheses φ and the target concept $f_{\rm trg}$, denoted $\mathrm{dist}(\varphi, f_{\rm trg})$, is defined to be the minimal number of revision operators needed to apply on φ to obtain a formula equivalent with $f_{\rm trg}$.

In the query learning model (introduced by Angluin [3]) an oracle is assumed to answer (in constant time) questions of the learner via some query protocol. These questions are typically of the form of a **membership query**, querying the value of the target concept on some assignment, or an **equivalence query**, asking whether some formula, constructed by the learner is equivalent to the target concept.

Definition 3.1 (Theory revision in the query learning model) *Given some formula class $\mathcal{R}$, an algorithm is a* **revision algorithm** *for $\mathcal{R}$ with* **query complexity** *p, if, given any function f_{trg}—called* **target concept**—*representable by some formula in $\mathcal{R}$, on input $\varphi \in \mathcal{R}$—called* **initial formula**—*the algorithm outputs some representation for f_{trg} using at most $p(\hat{e}, \log n)$ queries about f_{trg}, where $\hat{e} = \text{dist}(\varphi, f_{\text{trg}})$. The algorithm is said to be an* **efficient revision algorithm** *for $\mathcal{R}$, if p is a polynomial and the running time can also be bounded by a polynomial of the size of φ, the number of variables and $\hat{e}$. It is said that the query complexity of $\mathcal{R}$ is at least q, if any revision algorithm for $\mathcal{R}$ is of query complexity $\Omega(q)$.*

It is often also interesting whether both types of queries are necessary for efficient revision of a given formula class. The thesis considers this problem for both formula classes for which efficient revision is provided in the query learning model. Finally an equivalence query is called **proper**, if the formula in the query is an element of $\mathcal{R}$.

The mistake bounded model (see e.g. [22]) is defined in an on-line setting. In this model the learning proceeds in a sequence of rounds. In each round the learner receives first an instance of the domain (i.e., on which f_{trg} is defined), then produces a prediction of its classification, and finally receives a label (which, in a noise-free model is the correct classification—i.e., what f_{trg} evaluates on it). If the predicted classification and the received label disagree then the learner made a **mistake**. The **mistake bound** of the learning algorithm is the maximal number of mistakes, taken over all possible runs, (that is, sequences of instances), depending on the **size** of f_{trg}.

Definition 3.2 (Theory revision in the mistake bounded model) *Given some formula class $\mathcal{R}$, an algorithm is a* **revision algorithm** *for $\mathcal{R}$ with* **mistake bound** *p, if, given any function f_{trg}—called* **target concept**—*representable by some formula in $\mathcal{R}$, on input $\varphi \in \mathcal{R}$—called* **initial formula**—*the algorithm makes at most $p(\hat{e}, \log n)$ mistakes on instances classified by f_{trg}, where $\hat{e} = \text{dist}(\varphi, f_{\text{trg}})$. The algorithm is said to be an* **efficient revision algorithm** *for $\mathcal{R}$, if p is a polynomial and the running time in each round can also be bounded by a polynomial of the size of φ, the number of variables and $\hat{e}$.*

Additional results on theory revision (not discussed in the present dissertation) are given in the papers [8–10].

Read-once formulas

In Chapter 4 read-once functions are considered. A formula is **read-once formula**, if every variable occurs in it at most once and a function is **read-once function** if it is representable with a read-once formula. Results presented in this chapter appeared in the paper [9]. The importance of this formula class is rather theoretical, being a nontrivial subclass of Boolean formulas that is tractable from several different aspects, and having a nice semantic characterization [12; 16; 26]. This class is shown to be efficiently learnable in the query model using membership and equivalence queries [5], which motivated the research aimed to construct an efficient revision algorithm for it. The main result of this chapter is a revision algorithm for this class in the deletions-only case (Algorithm `ReviseReadOnce`), which is shown to be an efficient revision algorithm. (The **deletions-only case** means that the target concept can be represented by some formula obtained from the initial formula by removing some subformulas from it.)

Theorem 4.7 *Let φ be a read-once formula over n variables, and assume that the target concept f_{trg} can be represented by some formula obtained from φ by removing some subformulas from it. Then* `ReviseReadOnce`(φ)*, using at most $O(\hat{e}\log n)$ queries (where $\hat{e} = \mathrm{dist}(\varphi, f_{\mathrm{trg}})$), outputs some representation for f_{trg}.*

Additionally it was proved that the query complexity of the algorithm is close to the optimal.

Theorem 4.11 *The query complexity of revising read-once formulas in the deletions-only model is $\Omega(\hat{e}\log(n/\hat{e}))$, where n is the number of variables in the initial formula and $\hat{e}$ is the revision distance between the initial formula and the target formula.*

Finally it is shown that both type of query used by Algorithm `ReviseReadOnce` is necessary for the efficiency.

Theorem 4.13 *The query complexity of revising read-once formulas in the deletions-only model with* proper *equivalence queries alone is at least $\lfloor n/2 \rfloor - 1$ (where n is the number of relevant variables in the initial formula), even when the revision distance is only one.*

Theorem 4.14 *Denote the revision distance between the initial formula and the target formula by $\hat{e}$, and assume that the learner is allowed to ask arbitrarily many membership queries, but only at most $\hat{e} - 1$ equivalence queries. Under this restriction the query complexity of revising read-once formulas in the deletions-only model is at least $n - \hat{e}$, where n is the number of relevant variables in the initial formula.*

Threshold Formulas

In Chapter 5 the revisability of Boolean threshold functions is considered. Results in this section appeared in the paper [31]. (A Boolean function is said to be a **threshold function** if it can be represented by a set of variables R and a threshold θ, such that it evaluates 1 on exactly those assignments which assign 1 to at least θ of the variables in R.) Threshold functions (although in a more general form) are famous for being the basic ingredient of neural networks and support vector machines—and have several other applications as well. Boolean threshold functions are also known to be efficiently learnable in the query learning model [14] (however the learning algorithm presented in [14] uses only membership functions). The main result is again an algorithm (Algorithm `ReviseThreshold`) which is an efficient revision algorithm for the class of Boolean threshold functions in the query model.

Theorem 5.5 *Let φ be the initial formula and f_{trg} be the target concept, where both are n-variable threshold funtions. Then* `ReviseThreshold`(φ)*, using $O(\hat{e}\log n)$ queries, outputs some representation for f_{trg}, where $\hat{e} = \mathrm{dist}(\varphi, f_{\mathrm{trg}})$.*

Again, this query complexity turns out to be close to the optimal.

Proposition 5.8 *The query complexity of revising threshold formulas with membership and equivalence queries is* $\Omega(\hat{e}\log(n/\hat{e}))$*, where* n *is the number of variables and* $\hat{e}$ *is the revision distance between the initial formula and the target formula.*

In view of that the learning algorithm of Hegedűs for this class uses only membership queries, the question whether both type of queries are necessary for the efficient revision seems even more appropriate. The answer turns out to be affirmative again.

Theorem 5.6 *Assume that both the initial formula and the target formula have threshold value* t*, and that the learner is allowed to ask equivalence queries only for threshold functions also having threshold value* t*. (On the other hand, no restrictions are set on the membership queries.) Under this restriction, the query complexity of revising threshold formulas is at least* $n-1$ *(where* n *is the number of variables in the universe in scope), even when the revision distance is only one.*

Theorem 5.7 *The query complexity of revising threshold formulas with equivalence queries alone is at least* $n-1$ *(where* n *is the number of variables in the universe in scope), even when the revision distance is only one.*

Finally it is shown that the natural extension of Algorithm `Winnow` [22] does not give an efficient revision algorithm for the class of threshold formulas.

Proposition 5.9 `Winnow` *is not an efficient revision algorithm for threshold functions. More precisely, for any weight vector representing the initial threshold function* $\mathrm{Th}^1_{v_1,\ldots,v_n}$*,* `Winnow` *can make* n *mistakes when the target function is* $\mathrm{Th}^2_{v_1,\ldots,v_n}$*.*

This is interesting in view of that this algorithm is famous for learning some formula classes highly efficiently using some (general) threshold function representation.

Projective DNF Formulas

As a closure of the first part dealing with theory revision, in Chapter 6 the revisability of projective DNFs is considered, discussing the corresponding results appeared in [30].

Definition 6.1 *A DNF formula* φ *is a* k-**projective DNF***, or* k-**PDNF** *if it is of the form* $\varphi = \rho_1 t_1 \vee \cdots \vee \rho_\ell t_\ell$*, where, for* $i = 1,\ldots,\ell$ *it holds that* ρ_i *is a* k*-conjunction,* t_i *is an arbitrary conjunction and that* $\rho_i\varphi \equiv \rho_i t_i$*. The class of* n*-variable functions representable by some* k*-projective formula is denoted* k-PDNF$_n$*.*

Projective DNF formulas form a subclass of the disjunctive normal form formulas, introduced recently by Valiant [33]. (The motivation for considering *subclasses* of the DNFs has substantially grown after the recent result of Alekhnovich *et al.* proving that, unless RP = NP, the class of DNFs is not efficient learnable [2].) This class was found by Valiant to be suitable for a special form of learning, called projective learning, the general idea behind it being that learning, similarly to other biological processes, should be carried out on multiple levels in a distributed manner. The main result of this chapter is that Algorithm `RevWinn` (a natural extension of Valiant's algorithm) is an efficent revision algorithm for the class of k-projective DNFs in the mistake bounded model.

Theorem 6.3 *Suppose that the initial formula is the k-PDNF$_n$ formula φ and that the target concept is also k-PDNF$_n$, and that $\hat{e} = \mathrm{dist}(\varphi, f_{\mathrm{trg}})$. Then algorithm* `Rev-`$k$`-PDNF` *is efficient, and makes $O(\hat{e}k \log n)$ mistakes.*

The algorithm (just like the one used by Valiant [33]) consists of two levels. On the lower level simple learning algorithms are run, each concentrating on just a small part (or restriction) of the function to be learned. On the upper level another simple algorithm is run, which, on one hand, learns how to (re)combine the output of the algorithms on the lower level, and, on the other hand, it filters the information forwarded to these algorithms such that each one receives only that part of the information which is supposed to be relevant for it.

In the second part of the chapter first it is shown that the definition of k-PDNFs is basically purely semantical. For this the following notions need to be introduced. Given a Boolean function f, its **truth set**, denoted $\mathcal{T}(f)$, consists of exactly those assignments on which f evaluates 1. A **cube** is the truth set of some term.

Lemma 6.5

(a) *A function f is k-projective if and only if for every $\mathbf{x} \in \mathcal{T}(f)$ there is a k-conjunction ρ such that $\mathbf{x} \in \mathcal{T}(\rho)$ and $\mathcal{T}(f) \cap \mathcal{T}(\rho)$ is a cube.*

(b) *If for every $\mathbf{x} \in \mathcal{T}(f)$ there is a k-conjunction ρ such that $\mathcal{T}(f) \cap \mathcal{T}(\rho) = \{\mathbf{x}\}$, then f is k-projective.*

Then a learnability related parameter, the so called exclusion dimension of the class is examined. (The analysis extensively used the above observation formulated by Lemma 6.5.) This parameter is known to be related to the query complexity of the best learning algorithms for a given class (see [4; 15]) which, combined with the result derived for the exclusion dimension implies lower and upper bounds for the query complexity of this class.

Proposition 6.9 *The class k-PDNF$_n$ can be learned with $O\left(n\, 2^k \binom{n}{k}^2\right)$ membership and proper equivalence queries. On the other hand the query complexity of this class is at least $\binom{\lfloor n/4 \rfloor}{k} - 1$.*

Characterization Results

Characterization results appear (and are applied) in several forms in mathematics and in computer science; like giving a semantic description for some object defined in a syntactic way (e.g. that a number, written in decimal form, is divisable by 5 if and only if its last digit is either 0 or 5), or to give an alternative syntactic description for some object defined in a syntactic way, and so on. Actually, it is one of the fundamental tools in the analysis of some mathematical object (like, say, a function, set, formula class, etc) to give an alternative description or representation for it, and work with that. It can, on one hand provide more insight on the given object—which, in turn, can help solving the given problem—and, on the other hand (as is usual), it can provide more intriguing questions. A prominent example for this is the Fourier transform of functions—giving an alternative representation for functions as a linear combination of some orthonormal system—, which is of invaluable importance, both in case of the real world applications and also on the theoretical level.

In this part of the thesis various characterization results are shown. First, motivated by one of the revision algorithms, a structural description of a class of projective DNF is given. Then k-term DNFs are considered, giving a complete description of those formulas which have the largest number of prime implicants. This completes a series of well-known results on this class. A related characterization result is given for a class of DNF tautologies with a distance condition. Finally, motivated by a problem in belief revision (an area related to, but distinct from, theory revision), a criterion is given for the existence of a complement of a Horn formula.

1-projective DNF Formulas

In Chapter 7 a further characterization result is presented for projective DNF formulas, discussing a result which appeared in the paper [30]. Projective DNFs are defined in a rather semantic way (which is more apparent from part (a) of Lemma 6.5), however the main result of this chapter is a simple syntactic description for a subclass of this class, the 1-projective DNFs (Theorem 7.3). To make the formulation (and the proof) easier, a special irredundancy notion was introduced (consisting of simple, natural syntactic irredundancy restrictions). $\mathrm{Lit}(t)$ denotes the set of literals occuring in t, $\mathrm{Var}(t)$ the set of variables occuring in t.

Definition 7.1 *A 1-PDNF formula $\varphi = \rho_1 t_1 \vee \cdots \vee \rho_\ell t_\ell$ is* **p-irredundant** *if the following conditions all hold:*

(a) $\mathrm{Lit}(\rho_i t_i) \not\subseteq \mathrm{Lit}(\rho_j t_j)$ *for each distinct* $i, j \in \{1, \ldots, \ell\}$,

(b) $\rho_i, \overline{\rho_i} \notin \mathrm{Var}(t_i)$ *for every* $1 \leq i \leq \ell$,

(c) *if* $\ell \geq 3$ *then* $\rho_i \neq \overline{\rho_j}$ *for each distinct* $i, j \in \{1, \ldots, \ell\}$.

Otherwise, φ *is called* **p-redundant**.

It was also discussed that any 1-PDNF can be easily transformed into p-irredundant form.

Theorem 7.3 *A formula φ is a p-irredundant 1-PDNF formula if and only if it is either of the form*

$$\varphi = \bigvee_{i=1}^{s} (\rho_{i,1} t_i \vee \cdots \vee \rho_{i,\ell_i} t_i),$$

where $\rho_{i,r} \notin \mathrm{Var}(t_i)$ and $\overline{\rho_{i,r}} \in \mathrm{Lit}(t_j)$ for every distinct $i, j \in \{1, \ldots, s\}$ and $1 \leq r \leq \ell_i$, and furthermore the projections are all based on different variables, or it is of the form

$$\varphi = vt \vee \overline{v}t' \ ,$$

where $v \notin \mathrm{Var}(t)$ and $\overline{v} \notin \mathrm{Var}(t')$.

k-term-DNF Formulas with Largest Number of Prime Implicants

A term t is an **implicant** of some Boolean function f, if any assignment satisfying t also satisfies f, while t is said to be a **prime implicant** of f if, in addition, this does not hold for any term obtained from t by removing some literals from it. In Chapter 8 the relation between the number of terms in a DNF and the number of prime implicants of it is considered, discussing results which appeared in [29].

First, in Section 8.3 previously known results on the topic are discussed. If some DNF consists of K terms, then it has at most $2^K - 1$ prime implicants [6; 21; 24]. It is also mentioned that this bound is known to be sharp [19; 21; 24]. The main result of the chapter is a complete characterization of the DNFs that have as many prime implicants as this bound allows; thereby completing the above results. (DNFs fulfilling this extremal property are called **maximal DNF**s.)

To formulate this result, some notions must be introduced. A DNF tautology is called **BT-DNF** if it possess a tree-like structure: there is some variable v occuring in each term in it; there is some variable u occuring in each term containing v unnegated and there is some variable w containing v negated; and so on. A DNF tautologz is called **ND** if it is a BT-DNF in which every term conflicts with every other term in exactly on variable. (Two terms **conflict** in some variable, if that variable occurs unnegated in one of the terms, and occurs negated in the other.) A DNF is a **NUD** if it satisfies that removing each variable from it that occurs either only negated or only unnegated in it, the resulting DNF is an ND. Now we state the main result of the chapter.

Theorem 8.1 *A DNF is maximal DNF if and only if it is a NUD.*

The theorem was proved by reducing the problem to the following one, stating that from the definition of ND the condition "is a BT-DNF" can be omitted.

Lemma 8.11 (ND Lemma [18]) *If in a DNF tautologz every term conflicts with every other term in exactly on variable, then it is an ND.*

Disjoint DNF Tautologies with Conflict Bound Two

Chapter 9 considers a generalization of the ND-lemma, discussing the results appeared in [32]. More precisely it is shown that if in some DNF tautology each pair of terms conflict in at least one but at most two variables, then it also posesses a tree-like structure (i.e., it is a BT-DNF).

Theorem 9.1 *If in a DNF tautology each term conflicts with every other term in one or two variables, then the DNF is a* BT-DNF.

In the chapter it is also discussed how this result relates to various generalizations motivated by semantic resp. syntactic considerations. Demonstrated by an example it is also shown that further relaxing the bound given for the conflict of the terms to three, the above mentioned tree-like structure does not hold always. This problem is also a special case of a problem considered in [23]: given a DNF tautology, the task is to construct a decision tree such that for each term of the DNF generated by it there is a term of the tautology that is a subterm of it. They have shown that even for some very simple DNFs this problem requires a decision tree with extremely big complexity; however the result presented in this chapter implies that for each DNF in the above mentioned restricted class there exists always some simple decision tree.

Decomposable Horn Formulas

Finally, in Chapter 10 decomposable **Horn formulas** are considered (conjunctive normal form formulas in which every clause contains at most one unnegated variable), discussing the results from [20]. For this first the notion of Horn complement is introdcued. (Notation $f \leq g$ means that f implies g; $f \leq g$ means that $f \leq g$, but $f \neq g$.)

Definition 10.5 (f-complement) *For Horn functions f and g such that $f \leq g$, a Horn function h is an f-**complement of** g iff $f \lneqq h$ and $f = (g \wedge h)$.*

Definition 10.6 (decomposable Horn function) *A Horn function f is **decomposable** if every Horn consequence $g \neq 1$ of f has an f-complement.*

Horn formulas, being an expressive class which also allows for polynomial time inference, and indeed is generally computationally tractable, play a central role in artificial intelligence and in computer science. The notion of decomposability comes from belief revision [1], a field interested in revising knowledge base in such a manner that satisfies some "reasonability" properties, that are typically formulated in the form of postulates. Decomposability was introduced for general logics in [7], where it was also shown to be equivalent to the existence of some revision operator satisfying the AGM postulates [1]—one of the most popular postulates used in belief revision. The main result of the chapter is Theorem 10.10, showing characterizations for the existence of a complement of a Horn function consequence of another Horn function.

Theorem 10.10 *Let $\varphi \not\equiv 1$ be a Horn formula, and ψ be a Horn consequence of φ. Then the following are equivalent:*

(a) *ψ has a φ-complement,*

(b) *$\hat{\varphi} \not\leq \psi$,*

(c) *for some clause D of ψ it holds that $\mathcal{B}_\varphi(D) \not\leq D$.*

We also use the notion of almost anti-monotone function. (A function is **anti-monotone** if it is monotonically decreasing in the usual sense.)

Definition 10.3 (almost anti-monotone function) *a function is* **almost anti-monotone** *if it is either anti-monotone, or there is an anti-monotone function g such that $\mathcal{T}(f) = \mathcal{T}(g) \cup \{\mathbf{1}\}$,*

Theorem 10.10, in turn is used to give a complete description of decomposable Horn formulas.

Theorem 10.13 *A Boolean function is a decomposable Horn function if and only if it is almost anti-monotone.*

[1] Belief revision is related to theory revision (at least in its topic); thus—as a closure—the two main topics of the dissertation meet again.

The characterizations lead to efficient algorithms for the construction of a complement whenever it exists (which is in contrast with a related, but somewhat more stringent complement notion of [13], the existence of which is NP-complete to decide). The result, is purely combinatiorial, but was meant in [20] as a first step towards what is referred to as "Horn-to-Horn belief revision": revision of Horn knowledge bases where the revised knowledge base is also required to be Horn; integrating hopefully efficient revision (the central notion in theory revision) and common sense reasoning (as a main goal in belief revision).

References

[1] C. E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: partial meet functions for contraction and revision. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, and T. Pitassi. Learnability and automatizability. In *Proc. 45th Symposium on Foundations of Computer Science (FOCS 2004)*, pages 621–630. IEEE Computer Society Press, 2004.

[3] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[4] D. Angluin. Queries revisited. *Theoretical Compututer Science*, 313(2):175–194, 2004. Earlier version appeared in 12th ALT, 2001.

[5] D. Angluin, L. Hellerstein, and M. Karpinski. Learning read-once formulas with queries. *Journal of the ACM*, 40(1):185–210, 1993.

[6] A. K. Chandra and G. Markowsky. On the number of prime implicants. *Discrete Mathematics*, 24:7–11, 1978.

[7] G. Flouris, D. Plexousakis, and G. Antoniou. On generalizing the AGM postulates: preliminary results and applications. In *Proc. 10th Workshop on Non-Monotonic Reasoning (NMR 2004)*, pages 171–179, 2004.

[8] J. Goldsmith and R. H. Sloan. New Horn Revision Algorithms. *Journal of Machine Learning Research*, 6:1919–1938, 2005.

[9] J. Goldsmith, R. H. Sloan, B. Szörényi, and Gy. Turán. Theory revision with queries: Horn, read-once, and parity formulas. *Artificial Intelligence*, 156:139–176, 2004.

[10] J. Goldsmith, R. H. Sloan, and Gy. Turán. Theory revision with queries: DNF formulas. *Machine Learning*, 47(2–3):257–295, 2002.

[11] R. Greiner. The complexity of theory revision. *Artificial Intelligence*, 107(2):175–217, 1999.

[12] V. Gurvich. On repetition-free Boolean functions. *Uspekhi Matematicheskikh Nauk*, 32(1):183–184, 1977. (In Russian).

[13] P. L. Hammer and A. Kogan. Essential and Redundant Rules in Horn Knowledge Bases. In *Proc. 28th Hawaii International Conference on System Sciences (HICSS 1995*, 209–218, 1995.

[14] T. Hegedűs. On training simple neural networks and small-weight neurons. In *Proc. 1st Eurpean Conference on Computational Learning Theory (EuroColt 1993)*, pages 69–82. Oxford University Press, 1994.

[15] L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, and D. Wilkins. How many queries are needed to learn? *Jourrnal of the ACM*, 43(5):840–862, 1996.

[16] M. Karchmer, N. Linial, I. Newman, M. Saks, and A. Wigderson. A combinatorial characterization of read-once formulae. *Discrete Mathematics*, 114:275–282, 1993.

[17] M. Koppel, R. Feldman, and A. M. Segre. Bias-driven revision of logical domain theories. *Journal of Artificial Intelligence Research*, 1:159–208, 1994.

[18] O. Kullmann. The combinatorics of conflicts between clauses. In *Proc. 6th Conference on Theory and Applications of Satisfiability Testing (SAT 2003)*, vol.2919 of LNCS, pages 426–440. Springer, 2003.

[19] J.-M. Laborde. Sur le cardinal maximum de la base complète d'une fonction booléenne, en fonction du nombre de conjunctions de l'une de ses formes normales. *Discrete Mathematics*, 32:209–212, 1980.

[20] M. Langlois, R. H. Sloan, B. Szörényi, and Gy. Turán. Horn formulas, decomposability and Belief Revision. Manuscript.

[21] A. A. Levin. Comparative complexity of disjunctive normal forms. *Metody Discret. Analiz.*, 36:23–38, 1981. (In Russian)

[22] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[23] L. Lovász, M. Naor, I. Newman, and A. Wigderson. Search problems in the decision tree model. *SIAM Journal on Discrete Mathematics*, 8:119–132: 1995.

[24] C. McMullen and J. Shearer. Prime implicants, minimum covers, and the complexity of logic simplification. *IEEE Transactions on Computers*, 35(8):761–762, 1986.

[25] R. J. Mooney. A preliminary PAC analysis of theory revision. In *Computational Learning Theory and Natural Learning Systems, volume III: Selecting Good Models*, pages 43–53. MIT Press, 1995.

[26] D. Mundici. Functions computed by monotone Boolean formulas with no repeated variables. *Theoretical Compututer Science*, 66:113–114, 1989.

[27] D. Ourston and R. J. Mooney. Theory refinement combining analytical and empirical methods. *Artificial Intelligence*, 66(2):273–309, 1994.

[28] B. L. Richards and R. J. Mooney. Automated refinement of first-order Horn-clause domain theories. *Machine Learning*, 19(2):95–131, 1995.

[29] R. H. Sloan, B. Szörényi, and Gy. Turán. On k-term DNF with the maximal number of prime implicants. Accepted for publication at *SIAM Journal on Discrete Mathematics*. Earlier version appeared as Electronic Colloquium on Computational Complexity (ECCC) Technical Report TR05-023.

[30] R. H. Sloan, B. Szörényi, and Gy. Turán. Projective DNF Formulae and Their Revision. Accepted for publication at *Discrete Applied Mathematics*. Earlier version appeared in 16th COLT, 2003.

[31] R. H. Sloan, B. Szörényi, and Gy. Turán. Revising threshold functions. *Theoretical Computer Science*, 382(3):198–208, 2007.

[32] B. Szörényi. Disjoint DNF Tautologies with Conflict Bound Two. Accepted for publication at *Journal on Satisfiability, Boolean Modeling and Computation*.

[33] L. G. Valiant. Projection learning. *Machine Learning*, 37(2):115–130, 1999.

[34] S. Wrobel. *Concept Formation and Knowledge Revision*. Kluwer, 1994.

[35] S. Wrobel. First order theory refinement. In *Advances in ILP*, pages 14–33. IOS Press, Amsterdam, 1995.