Algorithmic properties of autarkies

Victor Marek marek@cs.uky.edu
Mirosław Truszczyński mirek@cs.uky.edu
Department of Computer Science
University of Kentucky
Lexington, KY 40506-0046

Abstract

Autarkies arise in studies of satisfiability of CNF theories. In this paper we extend the notion of an autarky to arbitrary propositional theories. We note that in this general setting autarkies are related to the 3-valued logic. Most of our results are concerned with algorithmic properties of autarkies. We prove that the problem of the existence of autarkies is NP-complete and that, as in the case of SAT, if an autarky exists then it can be computed by means of polynomially many calls to an oracle for the decision version of the problem. We also prove that, while intractable in general, the problem of the existence of autarkies is in P if we restrict the class of autarkies of interest to those that are consistent with a fixed complete and consistent set of literals, or if we restrict the class of theories to 2CNF, Horn, and affine theories. In particular we present normal form results for autarkies of theories of special types.

KEYWORDS: *Autarky, algorithms, complexity*

1. Introduction

Autarkies arise in studies of propositional satisfiability. They were introduced in [7] in order to establish sufficient conditions for pruning the search for a satisfying interpretation of a CNF theory.

Let T be a collection of propositional clauses (a CNF theory). A nonempty and consistent set v of literals is an *autarky* for T if every clause $C \in T$ that contains a dual of a literal from v contains also a literal from v. Pure literals are simplest examples of autarkies. Namely, if a literal l is *pure* in a CNF theory T, that is, T contains no occurrence of the dual literal to l, then the set $\{l\}$ is an autarky for T.

Let us denote by T_v^- the set of all clauses in T that contain neither a literal from v nor the dual of a literal in v. The following simple result gives a fundamental property of autarkies that makes them useful in satisfiability research.

Theorem 1.1. *Let T be a CNF theory. If v is an autarky for T then T is satisfiable if and only if T_v^- is satisfiable.*

Theorem 1.1 implies that if v is an autarky for a CNF theory T then testing whether T is satisfiable can be reduced to testing whether T_v^- is satisfiable. This latter task is easier as T_v^- has at least $|v|$ fewer atoms than T. We note that if v consists of a pure literal, the simplification described by Theorem 1.1 is known as the *pure-literal* pruning rule.

Using Theorem 1.1, researchers designed algorithms testing satisfiability of 3CNF theories with the worst-case running times exponentially better than the trivial bound of $O(2^n)$, where n is the number of atoms in the input theory[1]. The first such algorithm, with the worst-case running time of $O(1.619^n)$, was presented in [7]. The line of research it started culminated with an algorithm running in time $O(1.497^n)$, described in [9, 4].

A most direct use of autarkies to decide satisfiability of a theory consists of repeatedly computing an autarky and using its literals to reduce the theory. The problem with this pruning mechanism is that computing autarkies is hard. The corresponding decision problem was reported to be NP-complete in [5]. To circumvent that problem [5] introduced the notion of a *linear* autarky, defined in terms of a certain linear programming problem. Linear autarkies can be computed in polynomial time. Using linear autarkies in place of general ones makes the reduction method described above polynomial. Moreover, [5] shows that the class of theories for which the method actually decides satisfiability contains, in particular, some well-known classes of theories for which the satisfiability problem is polynomial: 2CNF theories and Horn theories.

In this paper we study general autarkies. We first show that the concept of an autarky can be extended to the case of theories consisting of arbitrary propositional formulas. That generalization emphasizes and exploits a connection to 3-valued logic, already present in the original setting of CNF theories but obscured by the syntactic simplicity of clauses. We then focus on algorithmic properties of autarkies and show that the problem to decide the existence of autarkies is NP-complete, a fact reported without proof in [5]. We also show explicitly the property of self-reducibility — the existence of a reduction from a search problem for autarkies to its decision version. We investigate the problem of existence of autarkies consistent with a given complete and consistent set of literals. We show that for every such set v of literals, the problem to decide whether a finite theory possesses an autarky consistent with v is in P. In particular, the problems of existence of positive and negative autarkies are polynomial. Next, we prove that for several classes of theories, for which the satisfiability problem is in the class P, the existence of autarkies can also be decided in polynomial time. In addition, we obtain results concerning the structure of the set of autarkies of theories in these classes. Finally, in the conclusions we offer some more comments on the role of the 3-valued logic for the concept of an autarky.

The fact that computing autarkies is hard limited their role in the design of satisfiability solvers (and as we noted, prompted research of special autarkies that can be computed efficiently). The situation may be different, however, when we consider the problem of deciding the truth of a quantified boolean formula (QBF). This problem is PSPACE-complete in general and even those pruning techniques that require exponential time may be beneficial, as demonstrated in [8]. Autarkies may provide such pruning techniques, as we have the following general version of Lemma 2.4 from [1], concerned with simplifications by pure literals whose atoms are existentially quantified.

Lemma 1.2. *Let $Q_1x_1\ldots Q_nx_nE$ be a QBF, where E is a formula in CNF. If v is an autarky for E such that every atom that appears in v is existentially quantified, then $Q_1x_1\ldots Q_nx_nE$ is true if and only if $Q_1x_1\ldots Q_nx_nE^-_v$ is true.*

The theory E^-_v contains no atoms that appear in v and the corresponding quantifiers can be dropped from the prefix. Thus, the QBF $Q_1x_1\ldots Q_nx_nE^-_v$ constitutes a simplification of the original one. If the cost of finding autarkies can be offset by gains in the search time resulting from better pruning, autarkies will prove useful in the design of fast QBF solvers and deserve further study.

1. We provide worst-case estimates of the running times of satisfiability solvers up to a polynomial factor.

2. Preliminaries

We consider the language of propositional logic determined by a set of atoms At, two constants $\bot$ and $\top$, and the boolean connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$ and $\oplus$ (the last one denoting the *exclusive or*).

A *literal* is an atom or the negation of an atom. In the first case, the literal is called *positive* and in the second case — *negative*. Given a literal l there is an *underlying atom*, $|l|$. Thus $|\neg p| = p$. The *dual* of literal l, denoted $\bar{l}$, is $\neg p$ if $l = p$ and p is an atom, and p if $l = \neg p$. A *clause* is a disjunction of literals. Thus, clauses do not contain constants $\bot$ and $\top$. We identify the empty clause with the constant $\bot$.

For a formula φ, we write $At(\varphi)$ for the set of atoms that appear in φ and $Lit(\varphi)$ for the set of literals one can built of these atoms. We extend this notation to sets of literals and theories.

A *3-valued interpretation* of a set of atoms At is a function $v : At \rightarrow \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$, where **t**, **f** and **u** represent truth values *true*, *false* and *unknown*. There is a one-to-one correspondence between 3-valued interpretations and consistent sets of literals. It maps a 3-valued interpretation v to the set of literals

$$\{p : v(p) = \mathbf{t}\} \cup \{\neg p : v(p) = \mathbf{f}\}.$$

Therefore, we identify 3-valued interpretations and consistent sets of literals, and use the same symbols (typically v and w) to denote them. A *complete* interpretation is a 3-valued interpretation that assigns only values **t** and **f**. When we identify such interpretation with a set of literals, the resulting set v is *complete* and *consistent*, that is, for every literal l either l or $\bar{l}$ belongs to v.

We define the truth value of a formula φ in a 3-valued interpretation v, which we denote by $[v(\varphi)]_3$, in a standard way by using the 3-valued truth tables for the logical connectives in the language [3, Section 64]. They are shown in Table 1. When $[v(\varphi)]_3 = \mathbf{t}$, we say that v *3-satisfies* φ.

p	$\neg p$
f	**t**
t	**f**
u	**u**

p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \oplus q$
f	**f**	**f**	**f**	**t**	**f**
f	**u**	**f**	**u**	**t**	**u**
f	**t**	**f**	**t**	**t**	**t**
u	**f**	**f**	**u**	**u**	**u**
u	**u**	**u**	**u**	**u**	**u**
u	**t**	**u**	**t**	**t**	**u**
t	**f**	**f**	**t**	**f**	**t**
t	**u**	**u**	**t**	**u**	**u**
t	**t**	**t**	**t**	**t**	**f**

Figure 1. Truth tables for the 3-valued logic of Kleene.

When v is a complete interpretation (that is, when v is a complete and consistent set of literals), the truth value of every formula φ is the same under v, regardless of whether we view v as a 3-valued or a 2-valued interpretation. In such case, whenever $[v(\varphi)]_3 = \mathbf{t}$ (which is precisely when $v(\varphi) = \mathbf{t}$ in the 2-valued logic), we say that v satisfies φ.

There is a natural ordering $\preceq_k$ (called *knowledge ordering*) on the truth values $\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ of three-valued logic. In this ordering $\mathbf{u} \preceq \mathbf{t}$, and $\mathbf{u} \preceq \mathbf{f}$ (while **t** and **f** are non comparable). This ordering of truth values extends (via product ordering) to 3-valued interpretations. When we identify 3-valued interpretations with consistent sets of literals, the ordering $\preceq_k$ becomes the inclusion ordering of the family of consistent sets of literals.

In the paper we need a fundamental property of 3-valued interpretations established by Kleene.

Proposition 2.1 (Kleene). *Let φ be a propositional formula, and let v, v' be two 3-valued interpretations such that $v \preceq_k v'$ (that is, $v \subseteq v'$). Then $[v(\varphi)]_3 \preceq [v'(\varphi)]_3$.*

We will now introduce autarkies of arbitrary propositional theories. We say that v *touches* φ if $At(\varphi) \cap At(v) \neq \emptyset$.

Definition 2.2. *Let T be a set of propositional formulas. A consistent set v of literals is an* autarky *for T if every $\varphi \in T$ that is touched by v is 3-satisfied by v.*

Our general definition of autarkies, when limited to clauses, is equivalent to the definition we presented in the introduction. Indeed, a consistent set v of literals 3-satisfies a clause C if and only if C contains a literal from v. In addition, we can extend to the general case the fundamental property of autarkies, Theorem 1.1. Let v be a consistent set of literals and T a set of formulas. We define T_v^- to be the set of all formulas in T that are not touched by v (contain no atom from $At(v)$). This notation is a direct extension of the notation we introduced for CNF theories in the introduction. We now have the following result.

Proposition 2.3. *Let v be a consistent set of literals and T a set of formulas. If v is an autarky for T, then T is satisfiable if and only if T_v^- is satisfiable.*

Next, we state some basic properties of autarkies that we refer to later. The proofs are straightforward and we omit them.

Proposition 2.4. *Let T be a propositional theory.*

1. *If v is a consistent and complete set of literals that satisfies T then v is an autarky for T*
2. *If v an autarky for T then for every set of formulas $T' \subseteq T$, $v \cap Lit(T')$ is an autarky for T'.*

Finally, we state and prove a result, which allows us to reduce a theory when searching for its autarkies. Let φ be a formula of propositional logic and let $A \subseteq At(\varphi)$. We denote by φ_A the formula obtained from φ by replacing all positive occurrences of atoms from A with $\perp$ and all negative occurrences of atoms from A with $\top$. The formula φ_A *underestimates* φ by making the atoms from A to contribute to the satisfaction of φ as little as possible. There is also a dual notion. We define φ^A to be the formula obtained from φ by replacing all positive occurrences of atoms from A with $\top$, all negative occurrence of atoms from A with $\perp$ and, as before, by simplifying the constants away. This formula *overestimates* φ.

We have the following general property of 3-valued logic.

Proposition 2.5. *Let φ be a propositional formula, v a consistent set of literals and A a set of atoms such that $A \cap At(v) = \emptyset$. Then:*

1. $[v(\varphi)]_3 = \mathbf{t}$ *if and only if* $[v(\varphi_A)]_3 = \mathbf{t}$
2. $[v(\varphi)]_3 = \mathbf{f}$ *if and only if* $[v(\varphi^A)]_3 = \mathbf{f}$

Proof: First, we observe that positive occurrences of atoms in a formula φ are negative occurrences in $\neg\varphi$, and similarly, negative occurrences of atoms in φ are positive occurrences in $\neg\varphi$. Then, assuming that $\psi = \neg\varphi$, we have:

$$\psi_A = \neg\varphi^A \text{ and } \psi^A = \neg\varphi_A. \tag{1}$$

Now we proceed by simultaneous induction on the complexity of formula φ.
Base Case: The formula φ is an atom, say p. There are two subcases.
(a) $p \notin A$. In this case, $p_A = p^A = p$ and both equivalences hold.
(b) $p \in A$. In this case $p \notin At(v)$ and so, $[v(p)]_3 = \mathbf{u}$. Moreover, $p_A = \bot$ and $p^A = \top$. We have $[v(p)]_3 \neq \mathbf{t}$ and $[v(p_A)] \neq \mathbf{t}$. Thus, the first equivalence holds. Similarly, $[v(p)]_3 \neq \mathbf{f}$ and $[v(p_A)] \neq \mathbf{f}$. Hence the second equivalence holds, as well.
Inductive step: Let ψ be a formula of length at least 2. The cases when the main functor in ψ is $\wedge$ and $\vee$ are obvious. So let $\psi = \neg\varphi$. We have that $[v(\psi)]_3 = \mathbf{t}$ if and only if $[v(\varphi)]_3 = \mathbf{f}$. By the inductive hypothesis, the second equivalence holds for φ. Thus, $[v(\varphi)]_3 = \mathbf{f}$ if and only if $[v(\varphi^A)]_3 = \mathbf{f}$. Next, we have that $[v(\varphi^A)]_3 = \mathbf{f}$ if and only if $[v(\neg\varphi^A)]_3 = \mathbf{t}$. By (1), $[v(\neg\varphi^A)]_3 = \mathbf{t}$ if and only if $[v(\psi_A)]_3 = \mathbf{t}$. Thus, the first equivalence holds for $\psi = \neg\varphi$. The second equivalence can be argued in a similar way.

Since the cases of $\rightarrow$ and $\oplus$ reduce to $\neg, \wedge$, and $\vee$, the inductive step is complete and the assertion follows. $\Box$

We extend the mapping $\varphi \mapsto \varphi_A$ to theories. Given a propositional theory T and a set of atoms $A \subseteq At(T)$, we define $T_A = \{\varphi_A \colon \varphi \in T\}$. We have now the following reduction result.

Proposition 2.6. *Let T be a set of formulas, $A \subseteq At(T)$ a set of atoms and v a set of literals such that $At(v) \cap A = \emptyset$. Then v is an autarky for T if and only if v is an autarky for T_A.*

Proof: If v is an autarky for T then v is nonempty and consistent. Let us assume that v touches a formula $\psi \in T_A$. We have $\psi = \varphi_A$, for some formula $\varphi \in T$. Since $At(\psi) \subseteq At(\varphi)$, v touches φ and consequently, as v is an autarky for T, v 3-satisfies φ. By Proposition 2.5, v 3-satisfies $\varphi_A = \psi$. Thus, v is an autarky for T_A (as ψ was chosen arbitrarily). The converse implication can be proved similarly, once we observe that if a set v of literals such that $At(v) \cap A = \emptyset$ touches a formula $\varphi \in T$ then it touches the formula $\varphi_A \in T_A$. $\Box$

In several places in the paper we will use a symmetry argument applied to theories obtained by replacing some literals with others. A *renaming* is a permutation of the set of literals $Lit(At)$ such that for every literal l, $\pi(\bar{l}) = \overline{\pi(l)}$.

If π is a renaming and v is a *consistent* set of literals (a 3-valued interpretation), $\pi(v)$ is also a consistent set of literals (a 3-valued interpretation).

The so called *permutation lemma* is another useful property. Let π be a renaming and let φ be a propositional formula. We will now define a formula $\pi(\varphi)$ obtained by applying π to φ. To this end, we view φ as a tree with atoms in the leaves and boolean operators in the internal nodes. We define $\pi(\varphi)$ to be the formula obtained by replacing in the tree of φ every subtree representing a literal l with the subtree for the dual literal.

We note that this type of renaming of literals preserves clauses. That is, if C is a clause and π is a renaming, $\pi(C)$ is a clause, too.

We have the following property of renamings.

Lemma 2.7 (Permutation Lemma). *For every 3-valued interpretation v, for every renaming π of literals, and for every formula φ of L,*

$$[\pi(v)(\pi(\varphi)]_3 = [v(\varphi)]_3.$$

For every renaming π, 3-valued interpretation v touches a formula φ if and only if $\pi(v)$ touches $\pi(\varphi)$. Let us define $\pi(T) = \{\pi(\varphi) : \varphi \in T\}$. As a direct consequence of Lemma 2.7, we obtain a "symmetry" result for autarkies.

Proposition 2.8. *Let v be a 3-valued interpretation and T a set of propositional formulas. Then v is an autarky for T if and only if $\pi(v)$ is an autarky for $\pi(T)$.*

3. Decision and search problems for autarkies

The main objective of this section is to establish the complexity of the problem of the existence of autarkies. We will also consider a *search* version of the problem (to compute an autarky or determine that none exists).

Definition 3.1. AUTARKY EXISTENCE*: Given a propositional theory T, decide whether T has an autarky.*

First, we note the following obvious property that follows directly from the definition of an autarky.

Proposition 3.2. *Let T be a propositional theory and v a consistent set of literals, $v \subseteq Lit(T)$. The question whether v is an autarky for T can be decided in polynomial time in the size of T.*

Proposition 3.2 implies that the AUTARKY EXISTENCE problem is in the class NP. Our goal now is to show that it is NP-complete.

Proposition 3.3. *The* AUTARKY EXISTENCE *problem is NP-complete.*

Proof: By the comments above, we focus on the NP-hardness only. The proof is by the reduction from a variant of the propositional satisfiability problem, in which we restrict input theories to those that do not contain the empty clause nor tautologies. Clearly this decision problem is also NP-complete.

Let T be a CNF theory and let p_i, $0 \leq i \leq n-1$, be all atoms that appear in T. We introduce n *new* atoms q_i, $0 \leq i \leq n-1$, and define a CNF theory $A(T)$ to consist of three groups of clauses:

1. all clauses in T
2. clauses $p_i \vee q_i$ and $\neg p_i \vee \neg q_i$, where $0 \leq i \leq n-1$
3. clauses $\neg p_i \vee p_{i+1} \vee q_{i+1}$, $p_i \vee p_{i+1} \vee q_{i+1}$, $\neg q_i \vee p_{i+1} \vee q_{i+1}$, and $q_i \vee p_{i+1} \vee q_{i+1}$, where $0 \leq i \leq n-1$, and the addition of indices is modulo n.

The theory $A(T)$ can be constructed in linear time in the size of T. We will show that T is satisfiable if and only if $A(T)$ has an autarky.

($\Rightarrow$) Since T is satisfiable, there is a set $v \subseteq Lit(T)$ such that for every i, $0 \leq i \leq n-1$, exactly one of p_i and $\neg p_i$ belongs to v, and v satisfies T (indeed, each complete interpretation satisfying T can be represented by such set of literals). We define v' as follows:

$$v' = v \cup \{\neg q_i \colon p_i \in v,\ i = 0, 1, \ldots, n-1\} \cup \{q_i \colon \neg p_i \in v,\ i = 0, 1, \ldots, n-1\}.$$

We will show that v' is an autarky for $A(T)$. To this end, it is enough to show that every clause in $A(T)$ contains a literal from v'.

Since v satisfies T and T consists of non-tautological clauses, every clause in T contains a literal from v and so, also a literal from v'. By the definition of v', every clause of type (2) contains a literal from v', as well. Since all clauses of type (3) are subsumed by clauses of type (2), every clause of type (3) also contains a literal from v'.
($\Leftarrow$) Let us assume that v' is an autarky for $A(T)$. By the definition, v' is consistent and contains at least one literal. Due to the symmetry of the clauses of types (2) and (3), without loss of generality we can assume that it is one of $p_0, q_0, \neg p_0$, or $\neg q_0$. Since the proof in each case is the same, let us assume that $p_0 \in v'$. Since $\neg p_0 \vee \neg q_0$ is in $A(T)$ and is touched by v', it follows that $\neg q_0 \in v'$. Let us consider the clause

$$\neg p_0 \vee p_1 \vee q_1$$

from $A(T)$. It is touched by v'. Consequently, it is satisfied by v', which in turn implies that v' contains p_1 or q_1. In the first case, since v' touches and so, satisfies the clause $\neg p_1 \vee \neg q_1$, $\neg q_1 \in v'$. In the second case, for the same reasons, $\neg p_1 \in v'$. Continuing this argument, we show that v' is a complete set of literals over $At(A(T))$.

Let $v = v' \cap Lit(T)$. Let us consider a clause $C \in T$. It follows that $C \in A(T)$. Since T does not contain the empty clause and since v' is a complete set of literals over $At(A(T))$, v' touches C. Consequently, v' contains a literal from C. Since every literal in C belongs to $Lit(T)$, v contains a literal from C. Thus, v satisfies C and so T, as well (as C is an arbitrary close from T). $\Box$

We will now show that the AUTARKY SEARCH problem, where the goal is to *compute* an autarky or determine that none exists, can be solved directly by means of polynomially many calls to an algorithm for the AUTARKY EXISTENCE problem. While every NP-complete search problem can be solved by means of polynomially many calls to an oracle for its decision version, we show here an *explicit* reduction of AUTARKY SEARCH to AUTARKY EXISTENCE. Our reduction is based on two lemmas of separate interest.

Lemma 3.4. *Let T be a CNF theory and v a consistent set of literals.*

1. *If $a \in At(T)$, then v is an autarky for T and $a, \neg a \notin v$ if and only if v is an autarky for $T \cup \{a, \neg a\}$*
2. *If for every $a \in At(T)$, $T \cup \{a, \neg a\}$ has no autarkies then every autarky for T is a complete set of literals over $At(T)$.*

Proof: Part (1) of the assertion follows directly from the definition of an autarky.
(2) Let v be an autarky for T. By (1) it follows that for every $a \in At(T)$, $a \in v$ or $\neg a \in v$. Thus, v is a complete set of literals. $\Box$

Lemma 3.5. *Let T be a CNF theory such that every autarky for T is a complete set of literals over $At(T)$. Then, for every literal $l \in Lit(T)$, a set of literals $v \subseteq Lit(T)$ is an autarky for $T \cup \{l\}$ if and only if v is an autarky for T and $l \in v$.*

Proof: ($\Leftarrow$) Since v is an autarky for T and $l \in v$, v is an autarky for $T \cup \{l\}$.
($\Rightarrow$) Conversely, let us assume that v is an autarky for $T \cup \{l\}$. Then v is an autarky for T (Proposition 2.4(2)). Thus, v is a complete set of literals over $At(T)$ and so, it touches the unit clause l. Consequently, v contains l. □

We are now ready to show how a procedure to decide the existence of autarkies can be used to compute them. Let T be an input CNF theory

1. If T has no autarkies, output 'no autarkies' and terminate.

2. As long as there is an atom $a \in At(T)$ such that $T \cup \{a, \neg a\}$ has an autarky, we replace T by the theory obtained from $T_{\{a\}}$ by removing $\bot$ and $\neg\top$ from every clause of $T_{\{a\}}$. This operation preserves autarkies and ensures that the resulting theory is a collection of disjunctions of literals. We then continue. We denote by T' the theory we obtain when the process terminates.

3. We fix an enumeration of atoms in $At(T')$, say $At(T) = \{a_1, \ldots, a_n\}$, and define $T_0 := T'$. For $i = 1, \ldots, n$, we proceed as follows. If $T_{i-1} \cup \{a_i\}$ has an autarky, we set $l_i := a_i$. Otherwise, we set $l_i := \neg a_i$. We then set $T_i := T_{i-1} \cup \{l_i\}$. When the loop terminates, we set $v = \{l_1, \ldots, l_n\}$ and output it as an autarky of T.

Let us analyze Step 2. Let $a \in At(T)$ be an atom such that $T \cup \{a, \neg a\}$ has an autarky. Then, by Lemma 3.4(1), T has an autarky that contains neither a nor $\neg a$. By Proposition 2.6, $T_{\{a\}}$ has an autarky and every autarky of $T_{\{a\}}$ is an autarky of T (and this property holds also for the modification of $T_{\{a\}}$, as described Step 2). Since the input theory T has an autarky (we moved past Step 1), T' has an autarky and every autarky of T' is an autarky for T. Moreover, for no atom $a \in At(T')$, $T' \cup \{a, \neg a\}$ has an autarky. Thus, by Lemma 3.4(2), every autarky of T' is a complete set of literals. Using that fact, we find one autarky of T' in Step 3 of the algorithm. As we noted it is also an autarky for T.

We prove the correctness of Step 3 by showing that for every i, $1 \leq i \leq n$, T_i has an autarky, that every autarky of T_i is a complete set of literals over $At(T')$, and that every autarky of T_i is an autarky of T_{i-1}. In particular, the claim implies that T_n has a complete autarky. Since T_n contains unit clauses $l_1, \ldots, l_n$, $v = \{l_1, \ldots, l_n\}$ is an autarky for T_n. By the claim, it is also an autarky for T' and so, for T.

To prove the claim, we note that the claim holds for $i = 1$. Indeed, $T_0 = T'$ and so, T_0 has an autarky and every autarky for T_0 is a complete set of literals. Thus, every autarky for T_0 contains a_1 or $\neg a_1$. By Lemma 3.5, it follows that T_1 has an autarky. Moreover, since $T_0 \subseteq T_1$, every autarky for T_1 is an autarky for T_0. It also follows then that every autarky for T_1 is a complete set of literals. Assuming that the claim holds for some i, $1 \leq i < n$, we prove in the same way as in the case of $i = 1$, that the claim holds for $i + 1$. Thus, the claim follows by induction.

It is clear that the method described above requires linear number of calls to a procedure deciding the AUTARKY EXISTENCE problem.

We now discuss the relation of Theorem 3.3 with one of the results of [6].

Let S be a set of clauses. A clause $C \in S$ is *lean in S* if for some resolution refutation tree $\mathcal{T}$ with premises from S, C is one of premises (leaves) of $\mathcal{T}$. A subset L of S is *lean* in S if it consists of clauses that are lean in S. Clearly, for every set S of clauses, S has a largest lean subset; it consists of all clauses that are lean in S. We denote this set by L_S.

A *nonempty* subset $A \subseteq S$ is an *autark* of S with a *witness* v if v is an autarky for S and A is the set of all clauses touched (thus satisfied) by v. There is an operation $\circ$ on the set of 3-valued interpretations. This operation is defined by

$$v_1 \circ v_2 = v_1 \cup \{l : l \in v_2 \text{ and } \bar{l} \notin v_1\}$$

One can check that if both v_1, v_2 are autarkies for S then so is $v_1 \circ v_2$. Moreover, if A_i is an autark subset for which v_i is a witness, $i = 1, 2$, then $v_1 \circ v_2$ is a witness for $A_1 \cup A_2$.

We also note that the collection of autarkies of S is closed under the unions of increasing chains. Thus, if S has autarkies, it has maximal autarkies. Let v be a maximal autarky of S and let A be the set of all clauses in S touched by v. Clearly, A is an autark of S (v is its witness). We claim that A is a largest autark in S. Indeed, let A' be an autark in S and let v' be its witness. By our comments above, $v \circ v'$ is an autarky of S. Since v is a subset of $v \circ v'$, the maximality of v implies that $v \circ v' = v$. Consequently, v is a witness of the fact that $A \cup A'$ is an autark. In particular, $A \cup A'$ consists of all clauses in S touched by v. By the definition of A, $A \cup A' = A$ and so, $A' \subseteq A$.

This argument shows that if S has autarks, it has a largest autark. We denote this largest autark of S by A_S. In the case when S has no autarks, we set $A_S = \emptyset$. Since autarks are nonempty, S has autarks if and only if $A_S \neq \emptyset$. In [6] Kullmann shows the following elegant result.

Proposition 3.6 ([6]). *For every set of clauses S, $A_S \cup L_S = S$ and $A_S \cap L_S = \emptyset$.*

The definitions imply that $A_S \neq \emptyset$ if and only if S has an autarky. But, of course, by Proposition 3.6, $A_S \neq \emptyset$ if and only if $S \neq L_S$. Now, let LEAN be the language consisting of those sets of clauses for which $S = L_S$. Then Kullmann's result implies that for every finite set of clauses S, $S \in$ AUTARKY EXISTENCE if and only if $S \notin$ LEAN. Since AUTARKY EXISTENCE is NP-complete (Theorem 3.3), we get the following result of Kullmann from [6], Lemma 5.7.

Proposition 3.7 ([6]). *The problem* LEAN *is co-NP-complete.*

We note, however, that by the same observation (complementarity of languages AUTARKY EXISTENCE and LEAN), Proposition 3.7 can be used as an alternative argument to show Theorem 3.3.

4. Autarkies consistent with a given interpretation

In the previous section we demonstrated that the problem to decide the existence of an autarky is NP-complete. In this section, we will show that versions of that problem, in which we are interested in autarkies of some particular types are easier. Let w be a complete interpretation (that is, a complete and consistent set of literals). We say that a 3-valued interpretation v is *consistent* with w if $v \subseteq w$ (that is, since w is complete, if $v \cup w$ is consistent). We will now study the problem of testing if a theory T has an autarky consistent with a complete 3-valued interpretation w. We will first show that for a specific interpretation w consisting of atoms only the corresponding problem is polynomial and then use a symmetry argument to extend that result to arbitrary complete 3-valued interpretations.

Formally we say that an autarky v for T is *positive* if v consists of atoms. Let $v_{\mathbf{t}}$ be a complete interpretation defined by

$$v_{\mathbf{t}} = At.$$

Clearly, positive autarkies are precisely those autarkies that are consistent with $v_{\mathbf{t}}$.

We call a formula $\varphi \in L$ a *generalized constraint* (*constraint*, for short) if $At(\varphi) \neq \emptyset$ and the interpretation $v_{\mathbf{t}}$ does *not* satisfy φ ($v_{\mathbf{t}}(\varphi) \neq \mathbf{t}$).

It is common to refer to clauses consisting entirely of negative literals as *constraints*. One can check that a clause C is a constraint if and only if $v_{\mathbf{t}}(C) \neq \mathbf{t}$. The condition $v_{\mathbf{t}}(\varphi) \neq \mathbf{t}$ generalizes a key property characterizing constraint clauses, which justifies our choice of terminology.

Given a set of propositional formulas T, we define $T^c = \{\varphi \in T : \varphi \text{ is a constraint}\}$. and $A^c_T = At(T^c)$. The subscript T is dropped if T is clear from the context. We have the following properties of positive autarkies.

Proposition 4.1. *Let T be a set of formulas. Let $v \subseteq At(T)$ be a set of atoms. Then v is an autarky for T if and only if $v \subseteq At(T_{A^c})$ and v is an autarky for T_{A^c}.*

Proof: Let us assume that v is an autarky for T. We need to show that $v \cap A^c = \emptyset$ and that v is an autarky for T_{A^c}.

Let $\varphi \in T^c$. Let us assume that v touches φ. Since v is an autarky for T, v 3-satisfies φ. Since $v \preceq_k v_{\mathbf{t}}$, by Proposition 2.1 we have that $v_{\mathbf{t}}$ 3-satisfies φ, a contradiction with the fact that φ is a constraint. It follows that $v \cap A^c = v \cap At(T^c) = \emptyset$ and, consequently, $v \subseteq At(T_{A^c})$.

Next, let us consider a formula $\psi \in T_{A^c}$ such that v touches ψ. There is a formula $\varphi \in T$ such that $\psi = \varphi_{A^c}$. Since v touches ψ, v touches φ as well. Consequently, v 3-satisfies φ. By Proposition 2.5 (1), v 3-satisfies φ_{A^c}, that is, v 3-satisfies ψ. Since ψ was arbitrary, v is an autarky for T_{A^c}.

Conversely, let us assume that v is a set of atoms, $v \subseteq At(T_{A^c})$, and that v is an autarky for T_{A^c}. From the first assumption, it follows that $v \cap A^c = \emptyset$. Let us consider a formula $\varphi \in T$ such that v touches φ. Since $v \cap A^c = \emptyset$, v touches φ_{A^c}. Consequently, v 3-satisfies φ_{A^c}. By Proposition 2.5 (1) again, v 3-satisfies φ, and since φ is an arbitrary formula in T, the other implication follows, as well. □

Proposition 4.1 entails an algorithm that decides if a set T of formulas has a positive autarky and if so, computes it. To this end, the algorithm computes a sequence of pairs $\langle T_n, A_n\rangle$, starting with $T_0 = T$, and $A_0 = A^c_T$. If after the iteration k, $A_k = \emptyset$, the computation of the sequence terminates. Otherwise, the algorithm proceeds to the iteration $k+1$ and computes $T_{k+1} = (T_k)_{A_k}$, and $A_{k+1} = A^c_{T_k}$. This construction terminates because in every iteration the number of atoms in the theory decreases. We denote by $\langle T_+, A_+\rangle$ the last element in the sequence. Before we continue the description of the algorithm, we note the following consequence of Proposition 4.1.

Corollary 4.2. *Let T be a set of formulas. Let $v \subseteq At(T)$ be a set of atoms. Then v is an autarky for T if and only if $v \subseteq At(T_+)$ and v is an autarky for T_+.*

We return to the algorithm. By the definition, $A_+ = \emptyset$. Two cases are possible.

Case 1. $T_+ = \emptyset$. In this case, T_+ has no autarkies and, in particular, no positive autarkies. By Proposition 4.1 (and by induction), T has no positive autarkies.

Case 2. $T_+ \neq \emptyset$. Then, by the definition of T_+, $At(T_+) \neq \emptyset$. Since $A_+ = \emptyset$, $v_{\mathbf{t}}$ satisfies every formula in T_+ which contains at least one atom. Consequently, each such formula is 3-satisfied by $v = At(T_+)$. Since $v \neq \emptyset$, v is an autarky for T_+ and so, by Corollary 4.2, also for T. Thus we get the following corollary.

Corollary 4.3. *The problem $\{T : T$ is a finite propositional theory and T possesses a positive autarky$\}$ is polynomial.*

To extend the Corollary 4.3 to the case of arbitrary interpretations, we need an additional concept and a lemma. A *shift* is a renaming π such that for all l, $|l| = |\pi(l)|$ (we recall that $|l|$ denotes the atom of the literal l). Thus shift does not change the underlying atom, but can only change the sign.

Lemma 4.4. *Let w be a complete interpretation. Then there is a unique shift π_w such that $w = \pi_w(v_{\mathbf{t}})$. The shift π_w can be computed from w in linear time.*

Let w be a complete interpretation. We define P_w as the problem consisting of those finite theories T which possess an autarky consistent with w.

Proposition 4.5. *For every complete interpretation w, the problem P_w is polynomial.*

Proof: Given a finite theory T, let us apply π_w to T. Then, by Proposition 2.8 the resulting theory T' possesses a positive autarky if and only if T possesses an autarky consistent with w. But we have a polynomial-time algorithm for testing if T' possesses a positive autarky, and one can compute T' from T in polynomial time. Thus the assertion follows. □

One interesting interpretation is $v_{\mathbf{f}}$ defined by:

$$v_{\mathbf{f}} = \{\neg p \colon p \in At\},$$

A negative autarky for T is an autarky for T consisting of negative literals. It is quite clear that a negative autarky is one that is consistent with $v_{\mathbf{f}}$. Let us call a formula $\varphi \in L$ a *generalized dual-constraint* (abbreviated simply to *dual-constraint*) if $At(\varphi) \neq \emptyset$ and the interpretation $v_{\mathbf{f}}$ defined by does *not* satisfy φ. Dual-constraints generalize the notion of a *positive* clause.

Our algorithm for finding positive autarkies allows us to define an algorithm for finding negative autarkies. We can do this in either of two ways. One is to use Permutation Lemma and Proposition 2.8. But there is another, direct way. We define $T^d = \{\varphi \in T : \varphi \text{ is a dual-constraint}\}$, and $A^d_T = At(T^d)$. We then have by a reasoning following that of the proof of Proposition 4.1 the following fact.

Proposition 4.6. *Let T be a set of formulas. Let v be a set of negative literals. Then v is an autarky for T if and only if $At(v) \subseteq At(T_{A^d})$ and v is an autarky for T_{A^d}.*

Now, it is clear that we can follow the algorithm for computing positive autarkies almost verbatim; all we need to do is to consider T^d instead of T^c. The next corollary follows from Proposition 4.5, or directly via the reasoning outlined above.

Corollary 4.7. *The problem $\{T : T$ is a finite propositional theory and T possesses a negative autarky$\}$ is polynomial.*

We will use the results of this section in the next section, when studying the issue of autarkies for Horn theories.

5. Classes of theories for which AUTARKY EXISTENCE is easy

It is well known that the SAT problem is in P for the following classes of theories:

1. 2CNF theories
2. Horn theories, dual-Horn theories and renameable-Horn theories

3. Affine theories

We will show that for each of these classes the problem of the existence of autarkies is also in P. In some cases, we will also identify minimal autarkies and characterize the structure of the family of autarkies of a theory. This forms a partial solution to a general problem (*How the structure of a set of formulas F is reflected in its collection of autarkies?*) formulated in Section 9 of [6].

5.1 The class of 2CNF theories

The results of this section are related to the results from [5], because one can show that every autarky of a 2CNF theory is a linear autarky. Here we study the connection of autarkies with boolean constraint propagation and obtain results on the structure of the set of autarkies of 2CNF theories.

Let T be a CNF theory and let l be a literal. The key tool in studying autarkies of 2CNF theories is a version of the well-known boolean constraint (or unit) propagation. Let T be a CNF theory and let l be a literal, $l \in Lit(T)$. We set $L_0 := \{l\}$. We define L_{i+1} to consist of those literals l' that are in L_i or that can be derived by resolving literals in L_i with a clause in T. If the resolution results in the empty clause $\perp$, we include it in L_{i+1}, too. We set $BCP(T, l) = \bigcup_{i=0}^{\infty} L_i$. We note that in the version of unit-propagation we presented here we do not include in $BCP(T, l)$ literals that form unit clauses in T. In order to include a literal other than l in $BCP(T, l)$, it must be derived from a non-unit clause in T by resolving it against literals included in $BCP(T, l)$ earlier.

Proposition 5.1. *Let T be a 2CNF theory and v an autarky for T. If $l \in v$ then $BCP(T, l) \subseteq v$.*

Proof: We use the notation introduced above. By the definition, $L_0 \subseteq v$. Let us assume that $L_i \subseteq v$.

First, let us assume that $\perp \in L_{i+1}$. Since $L_i \subseteq v$, $\perp \notin L_i$. Thus, there is a literal $l \in L_i$ such that $C = \bar{l}$ is a clause in T. Since $l \in v$, v touches C but, being consistent, contains no literal in C. This contradicts the fact that v is an autarky for T. Thus, $\perp \notin L_{i+1}$.

Next, let us consider a literal l' such that $l' \in L_{i+1} \setminus L_i$. It follows that there is a literal $l \in L_i$ such that the clause $C = l' \vee \bar{l}$ belongs to T. Since $l \in v$, v touches C. Thus, v contains a literal from C. Since $\bar{l} \notin v$, it follows that $l' \in v$. Consequently, $L_{i+1} \subseteq v$. By induction, $BCP(T, l) \subseteq v$. □

Proposition 5.2. *Let T be a 2CNF theory and let $l \in Lit(T)$. If $BCP(T, l)$ is consistent then it is an autarky of T.*

Proof: Since $BCP(T, l)$ is consistent, it is a set of literals (that is, it does not contain $\perp$). Moreover, by the definition, $BCP(T, l) \neq \emptyset$. Let C be a clause touched by a literal $l' \in BCP(T, l)$. If l' is a literal of C, $BCP(T, l)$ contains a literal from C. So, let us assume that $\bar{l'}$ is a literal of C. Since $\perp \notin BCP(T, l)$, C contains a literal l'' that is different from $\bar{l'}$. It follows that $l'' \in BCP(T, l)$ and so, $BCP(T, l)$ contains a literal from C in this case, too. □

These two results form the basis for a necessary and sufficient condition for the existence of autarkies for 2CNF theories, and for a characterization of minimal autarkies. Specifically, Propositions 5.1 and 5.2 imply the following result.

Proposition 5.3. *Let T be a 2CNF theory.*

1. *T has an autarky if and only if for some literal $l \in Lit(T)$ the set $BCP(T, l)$ is consistent*

2. *Every autarky of T is the union of a nonempty family of autarkies of the form $BCP(T, l)$.*

It is now clear that in order to decide whether a 2CNF theory T has an autarky, it is enough to compute $BCP(T, l)$ for every literal $l \in Lit(T)$. If in at least one case, we obtain a consistent set of literals, this set is an autarky for T. Otherwise, T has no autarkies. This method can be implemented to run in polynomial time in the size of T.

Theorem 5.3 also implies a method to compute minimal autarkies of a 2CNF theory T. To this end, we observe that minimal autarkies are precisely minimal consistent sets of the form $BCP(T, l)$. To compute them all we need to do is to identify minimal elements in the family of *consistent* sets of the form $BCP(T, l)$, which can be accomplished in polynomial time.

5.2 The class of Horn theories

We now consider the case of Horn theories. As in the previous section, the results we present here are related to those presented in [5]. Unlike [5] however, our focus is on the structure of autarkies and we do not impose restrictions on the class of Horn theories that we consider.

A clause is *Horn* if it contains at most one non-negated literal. A Horn clause is *definite* if it contains exactly one non-negated literal. Otherwise, it is an *indefinite* clause or a *constraint*. A Horn clause is a *fact* if it is a positive unit clause (consists of a single literal and this literal is an atom).

A *Horn theory* is a collection of Horn clauses. We denote the set of constraints and the set of facts of a Horn theory T by T^c and T^f, respectively. Facts are the only dual constraints a Horn theory may contain. Thus, $T^f = T^d$.

If T contains no constraints ($T^c = \emptyset$), it is *definite*. If T contains no facts ($T^f = \emptyset$), it is *dual definite*. We note that the set of all atoms of a definite Horn theory is a 2-valued model of that theory. Similarly, the set of all literals obtained by negating all atoms appearing in a dual definite Horn theory is a 2-valued model of that theory.

Given a set of literals v, we define v^+ as the set of positive literals (that is, atoms) in v, and v^- as the set of negative literals (negated atoms) in v. We then have the following fact.

Lemma 5.4. *Let T be a Horn theory. Let v be an autarky for T such that $v^- \neq \emptyset$. Then v^- is an autarky for T.*

Proof: Let C be a clause in T such that v^- touches C. Since $v^- \subseteq v$, v touches C and, consequently, v contains a literal from C. Let us assume that v does not contain a negative literal from C. It follows that C is definite, say $C = \neg p_1 \vee \ldots \vee \neg p_k \vee q$, and $q \in v$. Since v is consistent, $\neg q \notin v$ and so, $\neg q \notin v^-$. Further, for every i, $1 \leq i \leq k$, $\neg p_i \notin v$ and so, $\neg p_i \notin v^-$. This is a contradiction with the fact that v^- touches C. Thus, v contains a negative literal from C, and consequently, v^- contains a literal from C. It follows that v^- is an autarky for T. □

Corollary 5.5. *Let T be a Horn theory. If T has an autarky then it has a positive autarky or a negative autarky.*

Proof: Let us assume that T does not have a positive autarky. Let v be an autarky for T. By our assumption, $v^- \neq \emptyset$ and, by Lemma 5.4, v^- is a negative autarky for H. □

Corollary 5.6. *The problem of the existence of an autarky for Horn theories is in* P.

Proof: To decide whether a Horn theory T has an autarky we use first the algorithm described in Section 4 to find a positive autarky of T. If we succeed, T has an autarky and we stop. Otherwise, we use the dual version of this algorithm that finds a negative autarky (if one exists). If we succeed, T has an autarky and we stop. Otherwise, we return that T has no autarkies and stop. Corollary 5.5 implies that the algorithm is correct. It is evident that it can be implemented to run in polynomial time. $\Box$

We now turn attention to minimal autarkies of Horn theories. We have the following result.

Proposition 5.7. *If v is a minimal autarky of a Horn theory T then v is positive or v is negative.*

Proof: Let us assume that v is not positive. It follows that $v^- \neq \emptyset$. Thus, v^- is an autarky for T (Lemma 5.4). Since v is a minimal autarky for T, $v = v^-$, that is, v is a negative autarky for T. $\Box$

Positive autarkies of Horn theories have a characterization based on a certain efficient computational procedure with a flavor of a bottom-up constraint propagation. Let T be a Horn theory and let a be an atom. We set $A_0 = \{a\}$. Next, given a set of atoms A_i, we define A_{i+1} to contain every atom from A_i and in addition, every atom b such that there is a clause $C = b \vee \neg b_1 \vee \ldots \vee \neg b_k$ in T, with at least one b_j in A_i. We then set $AP(T, a) = \bigcup_{i=0}^{\infty} A_i$ (AP stands for *autarky propagation*). We have the following basic result. We use in it the notation T_+, which was introduced in Section 4.

Proposition 5.8. *Let T be a Horn theory and a an atom in $At(T_+)$.*

1. *The set of atoms $AP(T_+, a)$ is an autarky for T*
2. *If v is a positive autarky for T and $a \in v$ then $AP(T_+, a) \subseteq v$*
3. *Every positive autarky of T is the union of sets of the form $AP(T_+, a)$.*

Proof: (1) We recall that if $T_+ \neq \emptyset$, then theories T and T_+ have the same positive autarkies (Corollary 4.2). Consequently, since $AP(T_+, a)$ consists of atoms only, it suffices to show that $AP(T_+, a)$ is an autarky for T_+. Let C be a clause in T_+ such that $AP(T_+, a)$ touches C. Let us assume that $C = b \vee \neg b_1 \vee \ldots \vee \neg b_k$ (we recall that T_+ is definite). If $b \in AP(T_+, a)$, then $AP(T_+, a)$ contains a literal from C. So, let us assume that $b_j \in AP(T_+, a)$ and, more specifically that $b_j \in A_i$, for some non-negative integer i. By the definition, $b \in A_{i+1}$ and so, $b \in AP(T_+, a)$. Thus, $AP(T_+, a)$ contains a literal from C in this case, too.
(2) Let v be a positive autarky for T. Then, v is an autarky for T_+ (Corollary 4.2). Since $a \in v$, $A_0 = \{a\} \subseteq v$ (we use the notation introduced above). Let us assume that $A_i \subseteq v$ and let us consider an atom $b \in A_{i+1} \setminus A_i$. It follows that there is a clause $C \in T_+$ such that $C = b \vee \neg b_1 \vee \ldots \vee \neg b_k$ and for some j, $b_j \in A_i$. Hence $b_j \in v$ and so, v touches C. Since v is an autarky for T_+, v contains a literal from C. Since v consists of atoms, $b \in v$. Thus, $A_{i+1} \subseteq v$ and, by induction, $AP(T_+, a) \subseteq v$.
(3) This part of the assertion is a direct consequence of (2). $\Box$

We note that our results imply a polynomial-time algorithm to compute minimal positive autarkies of Horn theories. Indeed to accomplish that task, one needs to compute all sets $AP(T_+, a)$, where $a \in At(T_+)$, and then select the minimal ones.

Let us observe that similar effective characterizations of negative autarkies of Horn theories are unlikely to exist, as negative autarkies are related to hitting sets of hypergraphs, a connection that implies the following result.

Proposition 5.9. *The following problem is NP-complete: given a Horn theory T and an integer k, decide whether T has a negative autarky with no more than k elements.*

Proof: The membership in the class NP is evident. To prove NP-hardness, we construct a reduction from the *hitting set problem*: given a family $\mathcal{H}$ of finite sets and an integer k, decide whether $\mathcal{H}$ has a hitting set with at most k elements. This problem is known to be NP-complete [2]. Let $X = \bigcup \mathcal{H}$. We define a Horn theory $T(\mathcal{H})$ as follows. For every $a \in X$, we include in $T(\mathcal{H})$ all clauses of the form

$$a \vee \neg a_1 \vee \ldots \vee \neg a_m \tag{2}$$

where $\{a_1, \ldots, a_m\}$ is a set in $\mathcal{H}$. We observe that the theory $T(\mathcal{H})$ can be constructed in polynomial time.

Let H be a hitting set for $\mathcal{H}$. Then, $v = \{\neg h \colon h \in H\}$ is an autarky for $T(\mathcal{H})$, as it 3-satisfies $T(\mathcal{H})$. Conversely, let v be a negative autarky for $T(\mathcal{H})$. Then, $v = \{\neg h \colon h \in H\}$, for some $H \subseteq X$. Let us choose $a \in H$ (since v is an autarky, H is not empty). Then, v touches and so, satisfies all clauses of the form (2), where $\{a_1, \ldots, a_m\}$ ranges over all sets in $\mathcal{H}$. Since v is negative, it follows that H is a hitting set for $\mathcal{H}$.

Thus, $\mathcal{H}$ has a hitting set with at most k elements if and only if $T(\mathcal{H})$ has a negative autarky with at most k elements, and the hardness follows. □

We conclude with a result on autarkies of a certain subclass of Horn theories.

Proposition 5.10. *Every autarky of a Horn theory consisting of facts and constraints contains a pure literal.*

Proof: Let v be an autarky for such a theory, say T. If v contains a negative literal l, l does not touch facts in T. Thus, l is pure in T.

So let us suppose that v consists of positive literals (atoms) only. Then v does not satisfy any constraint in T, and so all literals in v are pure. □

5.3 Dual-Horn theories

The results we obtained for Horn theories in Section 5.2 extend to the cases of dual-Horn theories. A clause is *dual-Horn* if it contains at most one negative literal. On analogy with Horn clauses, we have *facts*, definite dual-Horn clauses and dual-constraints. Facts in current context are negative units.

We state in a single proposition properties of autarkies for dual-Horn theories. The proofs of these results are similar to those of Section 5.2.

Proposition 5.11.

1. *Let G be a dual-Horn theory. Let v be an autarky for G such that $v^+ \neq \emptyset$. Then v^+ is an autarky for G.*

2. *Let G be a dual-Horn theory. If G has an autarky then it has a positive autarky or a negative autarky.*

3. *The problem of existence of an autarky for dual-Horn theories is polynomial.*

4. *If v is a minimal autarky of a dual-Horn theory G then v is positive or v is negative.*

5. *Every autarky of a dual-Horn theory consisting of facts and dual-constraints contains a pure literal.*

5.4 Renameable-Horn theories

A theory T is a *renameable-Horn theory* if there is a renaming π such that $\pi(T)$ is a Horn theory. It is well known that renameable-Horn theories can be recognized in polynomial time and, given a renameable-Horn theory, an appropriate renaming can be constructed in polynomial time, too. Of course the dual-Horn theories are renameable-Horn, but the class of renameable-Horn formulas is wider than dual-Horn. The property of Horn and dual-Horn theories that whenever they have autarkies then they have positive or negative autarkies does not lift to renameable-Horn theories. For instance, the theory $\{p \vee q, \neg p \vee \neg q\}$, which is renameable-Horn, has no positive and no negative autarkies. Nevertheless, as in the case of Horn theories, the problem of the existence of autarkies for renameable-Horn is easy. Applying a shift transforming a renameable-Horn theory R to a Horn theory H_R, checking if the theory R_H possesses an autarky, and if so, shifting the autarky back (via the same shift) gives a polynomial-time method to compute autarkies of renameable-Horn theories. Thus, we have the following fact.

Proposition 5.12. *The problem of existence of an autarky for renameable-Horn theories is polynomial.*

5.5 Affine theories

In this subsection, we study affine theories. These theories do not consist of clauses and our generalization of autarkies to the case of arbitrary theories becomes essential.

A propositional formula is *affine* if it is of the form

$$C = x_1 \oplus x_2 \oplus \ldots \oplus x_k$$

where $x_1, \ldots x_{k-1}$ are propositional variables and x_k is a propositional variable or a boolean constant $\top$ or $\bot$.

Let v be a set of literals. Let us observe that v 3-satisfies a affine formula $\varphi = x_1 \oplus x_2 \oplus \ldots \oplus x_k$ if and only if $At(\varphi) \subseteq At(v)$ and v satisfies φ in 2-valued logic.

Let T be a affine theory. A set of atoms $X \subseteq At(T)$ is a *component* of T if X is a minimal nonempty subset of At such that for every formula $C \in T$, $At(C) \subseteq X$ or $At(C) \cap X = \emptyset$. Alternatively, let $G(T)$ be the graph with the vertex set $At(T)$, in which two vertices are connected with an edge if they appear in the same formula of T. Then, components of T are precisely the vertex sets of connected components of $G(T)$. It follows that components of a affine theory T form a partition of the set $At(T)$.

Proposition 5.13. *Let T be a affine theory and let v be an autarky for T. Then*

1. *For every $C \in T$, either $At(C) \subseteq At(v)$ or $At(C) \cap At(v) = \emptyset$*
2. *For every component X of T, either $X \subseteq At(v)$ or $X \cap At(v) = \emptyset$.*
3. *For every component X of T such that $X \subseteq At(v)$, v satisfies $\{C\colon At(C) \subseteq X\}$ (in 2-valued logic).*

Proof: (1) Let $C \in T$. If v does not touch C then $At(C) \cap At(v) = \emptyset$. If v touches C then, v 3-satisfies C. By our earlier observation, it follows that $At(C) \subseteq At(v)$.

(2) Let us assume that $X \setminus At(v) \neq \emptyset$ and $X \cap At(v) \neq \emptyset$. Then $X \setminus At(v)$ is a nonempty proper subset of X and so, it is not a component. Thus, there is a formula C such that $At(C) \cap (X \setminus At(v)) \neq \emptyset$ and $At(C) \setminus (X \setminus At(v)) \neq \emptyset$. Since $At(C) \cap X \neq \emptyset$, $At(C) \subseteq X$. Thus, $At(C) \cap At(v) \neq \emptyset$. It follows that C is touched by v but not 3-satisfied by v, a contradiction with (1).
(3) Since v touches every formula in $\{C\colon At(C) \subseteq X\}$, v 3-satisfies every formula in $\{C\colon At(C) \subseteq X\}$. By our earlier observation, v satisfies every formula in $\{C\colon At(C) \subseteq X\}$ in 2-valued logic. □

Corollary 5.14. *Let T be a affine theory.*

1. *If for every component X the theory $\{C \in T\colon At(C) \subseteq X\}$ is unsatisfiable (in 2-valued logic) then T has no autarkies*
2. *If there is a component X such that $\{C \in T\colon At(C) \subseteq X\}$ is satisfiable (in 2-valued logic), then the set v of literals such that $At(v) = X$ and v satisfies $\{C \in T\colon At(C) \subseteq X\}$ is a minimal autarky for T*
3. *Every autarky for T is the union of minimal autarkies of T of the kind described in (2).*

Thus, to decide the existence of autarkies of a affine theory T we first find all components of T (one can accomplish that in polynomial time, as finding connected components of graphs is in P) and we use a polynomial-time algorithm deciding satisfiability of affine theories, to find a component X such that $\{C \in T\colon At(C) \subseteq X\}$ is satisfiable (in 2-valued logic). If none exists, T has no autarkies. Otherwise, T has a satisfiable component and, by Corollary 5.14, has an autarky.

6. Conclusions

The contribution of this paper is twofold. First, we studied computational properties of autarkies. We proved that the existence problem for autarkies is NP-complete. We have shown a direct reduction of the search version of the problem to the decision version in a linear number of calls to the decision version. We found a new class of *easy* autarkies - those that are consistent with a given complete interpretation. We also found several classes of theories for which the problem of autarky existence can be solved in polynomial time. More importantly, in each of these cases we classified autarkies and obtained the results on their structure in terms of minimal autarkies. Our results complement those of [5] and [6].

Second, we generalized autarkies to the case of arbitrary propositional theories by exploiting the concept of satisfiability in 3-valued logic. The choice of the logic warrants some comments. Let us call a set of literals a *weak autarky* of a theory T if for every formula $\varphi \in T$ that is touched by v, v entails φ (in 2-valued logic). It is well known that if $[v(\varphi)]_3 = \mathbf{t}$ then v entails φ (in 2-valued logic). Thus, every autarky is a weak autarky. In addition, a fundamental property of autarkies, Theorem 2.3, holds for weak autarkies, as well. Why then not to use weak autarkies rather than autarkies? In the case of clausal theories, there is no essential difference. Both concepts coincide if we exclude tautological clauses from considerations, a typical assumption in the satisfiability research. However, in the general case, the difference is significant. One can verify whether v 3-satisfies φ in polynomial time, while the problem to verify whether v entails φ is co-NP-complete in general (we stress that v is not necessarily a complete set of literals). Thus, the choice of logic in extending the notion of an autarky to the case of arbitrary theories is closely tied to the difficulty of recognizing autarkies.

Acknowledgments

This research was supported by the National Science Foundation under Grants No. 0097278 and 0325063.

References

[1] M. Cadoli, M. Schaerf, A. Giovanardi, and M. Giovanardi. An algorithm to evaluate quantified boolean formulae and its experimental evaluation. *Journal of Automated Reasoning*, 28:101–142, 2002.

[2] M.R. Garey and D.S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W.H. Freeman and Co., San Francisco, Calif., 1979.

[3] S. C. Kleene. *Introduction to Metamathematics*. North-Holland, 1967. Fifth reprint.

[4] O. Kullmann. New methods for 3-SAT decision and worst-case analysis. *Theoretical Computer Science*, 223:1–72, 1999.

[5] O. Kullmann. Investigations on autark assignments. *Discrete Applied Mathematics*, 107:99–137, 2000.

[6] O. Kullmann. Lean clause-sets: Generalizations of minimally unsatisfiable clause sets. *Discrete Applied Mathematics*, 130:209–249, 2003.

[7] B. Monien and E. Speckenmeyer. Solving satisfiability in less than 2^n steps. *Discrete Applied Mathematics*, 10:287–295, 1985.

[8] J. Rintanen. Partial implicit unfolding in the davis-putnam procedure for quantified boolean formulae. In R. Nieuwenhuis and A. Voronkov, editors, *Proceedings of the International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR01)*, volume 2250 of *LNCS*, pages 362–376. Springer, 2001.

[9] I. Schiermeyer. Pure literal lookahead: an $o(1.497^n)$ 3-satisfiability algorithm. In *Workshop on the Satisfiability Problem*, pages 127–136. Universität zu Köln, Technical Report No. 93-230, 1996.