Handbook of Satisfiability
Armin Biere, Marijn Heule, Hans van Maaren and Toby Walsh (Eds.)
IOS Press, 2009

Chapter 11

Minimal Unsatisfiability and Autarkies

Hans Kleine Büning and Oliver Kullmann

The topic of this chapter is the study of certain forms of "redundancies" in propositional conjunctive normal forms and generalisations. In Sections 11.1 - 11.7 we study "minimally unsatisfiable conjunctive normal forms" (and generalisations), unsatisfiable formulas which are irredundant in a strong sense, and in Sections 11.8 - 11.13 we study "autarkies", which represent a general framework for considering redundancies. Finally in Section 11.14 we collect some main open problems.

11.1. Introduction

A literal is a variable or a negated variable. Let X be a set of variables, then $\mathrm{lit}(X)$ is the set of literals over the variables in X. Clauses are disjunctions of literals. Clauses are also considered as sets of literals. A propositional formula in conjunctive normal form (CNF) is a conjunction of clauses. CNF formulas will be considered as multi-sets of clauses. Thus, they may contain multiple occurrences of clauses. The set of all variables occurring in a formula φ is denoted as $\mathrm{var}(\varphi)$. Next we introduce formally the notion of *minimal unsatisfiability*. Please note that in some of the older papers minimal unsatisfiable formulas are sometimes called "critical satisfiable".

Definition 11.1.1. A set of clauses $\{f_1, \ldots, f_n\} \in \mathrm{CNF}$ is called *minimal unsatisfiable* if $\{f_1, \ldots, f_n\}$ is unsatisfiable and for every clause f_i $(1 \leq i \leq n)$ the formula $\{f_1, \ldots, f_{i-1}, f_{i+1}, \ldots, f_n\}$ is satisfiable. The set of minimal unsatisfiable formulas is denoted as MU.[1]

The complexity class D^P is the set of problems which can be described as the difference of two NP-problems, i.e., a problem Z is in D^P iff $Z = X - Y$ where X and Y are in NP. The D^P completeness of MU has been shown by a reduction of the D^P-complete problem UNSAT-SAT [PW88]. UNSAT-SAT is the set of pairs (F, G) for which F is unsatisfiable and G is satisfiable.

Theorem 11.1.1. *[PW88]* MU *is D^P-complete.*

[1] Depending on the author, instead of "minimal unsatisfiable formulas" also "minimally unsatisfiable formulas" can be used (or is preferable).

The idea of the proof is as follows. At first, we show that MU is in D^P. Let Y be the set of satisfiable CNF and let X be the set of CNF such that after removal of any clause they are satisfiable. Then we obtain $\mathrm{MU} = X - Y$ and therefore MU is in D^P. For the hardness we assign to every pair of 3-CNF formulas (F, G) a formula $H(F, G)$, such that (F, G) is in UNSAT-SAT if and only if $H(F, G)$ is minimal unsatisfiable:

- Let $F = \{f_1, \ldots, f_n\}$ be in 3-CNF with $f_i = (L_{i,1} \vee L_{i,2} \vee L_{i,3})$.
- For new variables $x_1, \ldots, x_n$ let $\pi_i = (x_1 \vee \cdots \vee x_{i-1} \vee x_{i+1} \vee \cdots \vee x_n)$.
- We define

$$H_1(F) = \bigwedge_{1 \le i \le n} (f_i \vee \pi_i) \wedge \bigwedge_{1 \le i \le n, 1 \le j \le 3} (\neg L_{i,j} \vee \pi_i \vee \neg x_i) \wedge \bigwedge_{1 \le i < j \le n} (\neg x_i \vee \neg x_j).$$

- Let $G = \{g_1, \ldots, g_m\}$ be in 3-CNF with $g_i = (L'_{i,1} \vee L'_{i,2} \vee L'_{i,3})$.
- For new variables $y_1, \ldots, y_m$ let $\varphi_i = (y_1 \vee \cdots \vee y_{i-1} \vee y_{i+1} \vee \cdots \vee y_n)$.
- We define

$$H_2(G) = \bigwedge_{1 \le i \le n} (g_i \vee \varphi_i) \wedge \bigwedge_{1 \le i \le m, 1 \le j \le 3} (\neg L'_{i,j} \vee \varphi_i \vee \neg x_i) \wedge \bigwedge_{1 \le i < j \le m} (\neg y_i \vee \neg y_j) \wedge (y_1 \vee \cdots \vee y_m).$$

- Then we have $H_2(G)$ is unsatisfiable. Moreover, F is unsatisfiable if and only if $H_1(F) \in \mathrm{MU}$, and G is satisfiable if and only if $H_2(G) \in \mathrm{MU}$.
- For $H_1(F) = \bigwedge_i \sigma_i$ and $H_2(G) = \bigwedge_j \tau_j$ we define $H(F, G) = \bigwedge_{i,j} (\sigma_i \vee \tau_j)$.
- Then (F, G) is in UNSAT-SAT if and only if $H(F, G)$ is in MU.

11.2. Deficiency

A very interesting measure for the complexity of formulas is CNF is the so-called "deficiency" which indicates the difference between the number of clauses and the number of variables.

Definition 11.2.1. Let F be a formula in CNF with n clauses and k variables. Then the *deficiency* of F is defined as $\mathrm{d}(F) = n - k$. Further, we define the *maximal deficiency* as $\mathrm{d}^*(F) = \max\{\mathrm{d}(G) \mid G \subseteq F\}$.

Let k-CNF be the set of CNF-formulas with deficiency (exactly) k. Then the satisfiability problem for k-CNF is NP-complete. That can be shown easily by a reduction to SAT. Similarly, let $\mathrm{CNF}^*(k)$ be the set of formulas with maximal deficiency k, i.e. $\{F \in \mathrm{CNF} : \mathrm{d}^*(F) = k\}$. In [FKS02] it has been shown that the satisfiability problem for these classes is solvable in polynomial time.

The set of minimal unsatisfiable formulas with deficiency k is denoted as $\mathrm{MU}(k)$. It has been shown in [AL86] that any minimal unsatisfiable formula has a deficiency greater than 0, that means, every minimal unsatisfiable formula contains more clauses than variables. With respect to deficiency, one of the main results is the following

Theorem 11.2.1. *[FKS02] For fixed integer k the problem* $\mathrm{MU}(k)$ *is solvable in polynomial time.*

Instead of asking whether a formula is in MU(k) we may be interested in whether a formula contains a MU(k)-subformula. This problem (i.e., the set of formulas) is denoted by sup-MU(k) and is NP-complete [Sze01, KZ02a].

Formulas in conjunctive normal form can be represented as matrices ("variable-clause matrices"). The columns are the clauses and for each variable there is a row.

Example 11.2.1. For example, the formula $F = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_2)$ can be written as

$$\begin{pmatrix} x_1 & x_1 & & \neg x_1 & \neg x_1 \\ x_2 & & \neg x_2 & x_2 & \neg x_2 \\ x_3 & \neg x_3 & x_3 & & \neg x_3 \end{pmatrix}$$

If the order of the variables is fixed then sometimes we replace the positive occurrence of a variable by the symbol $'+'$ and the negative occurrence by the symbol $'-'$. For the fixed order x_1, x_2, x_3 and the formula given above, we have

$$\begin{pmatrix} + & + & & - & - \\ + & & - & + & - \\ + & - & + & & - \end{pmatrix}.$$

The transposed of the variable-clause matrix is called the "clause-variable matrix", and is studied further (as a matrix over $\{-1, 0, +1\}$) in [Kul03]; see Subsection 11.11.3.2 for further details on matrix representations of formulas.

11.2.1. Splitting

For a formula $F \in$ MU and a variable x we can assign the value true to x and reduce the formula by removing any clause with x and deleting any occurrence of $\neg x$. The formula we obtain is unsatisfiable and therefore contains at least one minimal unsatisfiable formula. If we perform the assignment also for $\neg x$ then we get two MU-formulas (a choice for each branch), and in this sense the assignment and the reduction splits the formula F into two MU-formulas. The splitting can be used to investigate the structure of MU-formulas and is a helpful tool for proofs; for an early reference see [KB99].

Definition 11.2.2. For $F \in$ MU and a variable x of F there exist formulas $B_x, C, B_{\neg x}$ in which neither x nor $\neg x$ occur such that

1. the formula F can be represented as

$$F = \{x \vee f_1, \ldots, x \vee f_s\} \cup B_x \cup C \cup B_{\neg x} \cup \{\neg x \vee g_1, \ldots, \neg x \vee g_t\};$$

2. $\{f_1, \ldots, f_s\} \cup B_x \cup C$, denoted by F_x, is in MU, and $\{g_1, \ldots, g_t\} \cup B_{\neg x} \cup C$, denoted by $F_{\neg x}$, is in MU. F_x and $F_{\neg x}$ may consist of the empty clause only.

We call $(F_x, F_{\neg x})$ a *splitting* of F on x.

Example 11.2.2. Continuing Example 11.2.1, we obtain the splitting

$$F_{\neg x_1} = (x_2 \vee x_3) \wedge (\neg x_3) \wedge (\neg x_2 \vee x_3)$$
$$F_{x_1} = (\neg x_2 \vee x_3) \wedge (x_2) \wedge (\neg x_2 \vee \neg x_3).$$

The corresponding matrix representations are

$$\begin{pmatrix} x_2 & \neg x_2 \\ x_3\ \neg x_3 & x_3 \end{pmatrix}, \begin{pmatrix} \neg x_2\ x_2\ \neg x_2 \\ x_3 \qquad \neg x_3 \end{pmatrix}.$$

For all k, all $F \in \mathrm{MU}(k)$, all variables x in F and all splittings $(F_x, F_{\neg x})$ of F on x we have:

1. $\mathrm{d}(F_x), \mathrm{d}(F_{\neg x}) \leq k$ (see Corollary 7.10 in [Kul03], considering the larger class of "matching lean" clause-sets as discussed later in Subsection 11.11.2).
2. If all variables occur positively as well as negatively at least twice, then $\mathrm{d}(F_x), \mathrm{d}(F_{\neg x}) < k$ (Theorem 2 in [KB00]; see Lemma 3.9 in [Kul00a] for a somewhat stronger statement, also considering the larger class of "lean" clause-sets as discussed later in Subsection 11.8.3).
3. If $k \geq 2$, then (by Lemma 3.10 in [Kul00a]) there is some variable x' such that for some splitting $(F_{x'}, F_{\neg x'})$ we have $\mathrm{d}(F_{x'}) < k$ and $\mathrm{d}(F_{\neg x'}) < k$. Such a variable can be computed in polynomial time.

That by splitting the deficiency can be reduced has to do with the "expansion" of variables, and is discussed further in Section 4.4 in [Kul07a]. If the common part of both formulas, C, is empty, we call the splitting *disjunctive*. We say a formula has a *unique splitting* for a variable x, if there exists exactly one pair of splitting formulas $(F_x, F_{\neg x})$. In general, splittings are not unique. If a MU-formula F has a disjunctive splitting on x then however the disjunctive splitting on x is unique. See Section 3 in [KZ03] for further investigations into splittings.

11.2.2. Structure of MU(1) and MU(2)

A class of MU-formulas with deficiency 1 is the set Horn-MU of minimal unsatisfiable Horn formulas. That together with the linear-time decidability can easily be shown by an induction on the number of variables. But MU(1) contains more formulas. The structure of MU(1)-formulas is well understood. Every MU(1)-formula can be represented as a *basic matrix* and vice versa [DDKB98]. Basic matrices are defined inductively as follows:

1. $(+\ -)$ is a basic matrix.
2. If B is a basic matrix then
$$\begin{pmatrix} B & 0 \\ b_1 & + \end{pmatrix}, \begin{pmatrix} B & 0 \\ b_2 & - \end{pmatrix}$$
are basic matrices, where b_1, b_2 are non-null row vectors without $+$ resp. without $-$.
3. If B_1 and B_2 are basic matrices and b_1, b_2 are as above, then
$$\begin{pmatrix} B_1 & 0 \\ b_1 & b_2 \\ 0 & B_2 \end{pmatrix}$$

is a basic matrix.

Here, " $+$ " represents a positive literal, and " $-$ " a negative literal. We usually omit zeros and write them as blank fields. The rows belong to the variables and the columns to the clauses.

Example 11.2.3. The formula $x \wedge \neg x$ has the basic matrix $\begin{pmatrix} + & - \end{pmatrix}$. For the formula $(x_1 \vee x_2) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_2 \vee \neg x_3)$ we have the basic matrix

$$\begin{pmatrix} + & - & \\ + & & - & - \\ & & + & - \end{pmatrix}.$$

Since basic matrices have a disjunctive splitting, we can always find a disjunctive splitting for MU(1)-formulas. Moreover, it is easy to see that any minimal unsatisfiable Horn formula is in MU(1), whereas for 2-CNF minimal unsatisfiable formulas with an arbitrary deficiency can be constructed. For an approach based on a tree representation of the elements of MU(1) see [Kul00a]. We remark that the class MU(1) has been studied rather extensively from the point of view of "qualitative matrix analysis"; see Subsection 11.12.1 for some general introduction.

With respect to the structure of the matrices for greater deficiencies only very few results are known. For example, every MU(2)-formula in which each literal occurs at least twice has the following form except for renaming [KB00]:

$$\begin{pmatrix} x_1 & \neg x_1 & \neg x_2 & \cdots & \neg x_{n-1} & \neg x_n & \neg x_1 \\ x_2 & x_2 & x_3 & \cdots & x_n & x_1 & \neg x_2 \\ x_3 & & & & & & \neg x_3 \\ \vdots & & & & & & \vdots \\ x_n & & & & & & \neg x_n \end{pmatrix}$$

See [SD97] for some initial study on MU(3) and MU(4).

11.3. Resolution and Homomorphism

It is well-known that in the worst-case resolution refutations of unsatisfiable formula require super-polynomially many resolution steps. For fixed deficiency, for each $F \in \mathrm{MU}(k)$ there is a resolution refutation with not more than $2^{k-1}n$ resolution steps. The proof follows immediately by splitting property 3 from Section 11.2.1 together with the characterisation of MU(1) from Subsection 11.2.2.

It is easy to see that $\mathrm{MU}(k)$ is closed under $(1,*)$*–resolution* (also called *singular DP-resolution*) for every k. That means, if $F = \{L \vee f, \neg L \vee g_1, \neg L \vee g_2, \cdots, \neg L \vee g_s\} + F' \in \mathrm{MU}(k)$ and L as well as $\neg L$ do not occur in F', then $\{f \vee g_1, f \vee g_2, \ldots, f \vee g_s\} + F' \in \mathrm{MU}(k)$. Please note that the literal L occurs only once in the formula, and that we simultaneously resolve on all clauses with literal L. In that sense, $(1,*)$-resolution is a restricted version of hyperresolution.

We say a formula is *read-once refutable* if the formula has a resolution refutation tree in which each input clause is used at most once, and such a refutation

is called a *read-once refutation*. ROR is the class of read-once refutable formulas and known to be NP-complete [IM95]. The following example is a formula for which a read–once refutation exists, no proper subformula has a read-once refutation, but the formula is not minimal unsatisfiable.

$$\begin{pmatrix} z & u & \neg x & \neg x & \neg a & \neg a & a & y & \neg z & \neg u \\ & & \neg y & \neg b & \neg y & \neg b & x & b & y & a \\ & & & & & & & & b & \neg x \end{pmatrix}.$$

Let ROR-MU(k) be the set of minimal unsatisfiable formulas with deficiency k for which a read-once refutation exists. ROR-MU(k) is decidable in polynomial time for any fixed k [KZ02a]. For the class Sup-ROR-MU(k) $:= \{F \mid \exists G \in$ ROR-MU(k) : $G \subseteq F\}$ it has been shown in [KZ02a, Sze01] that the decision problem is NP-complete. In [FSW06] it has been shown that the *Short Resolution Refutation* (SRR) and the *Small Unsatisfiability Subset* (SUS) problems are likely not fixed-parameter tractable, however the restrictions to planar formulas or to formulas with fixed bounds on clause-length and variable-occurrence are fixed-parameter tractable. Here SRR is the problem whether for a parameter t a refutation with at most t steps exists, and SUS is the problem of deciding whether a formula contains an unsatisfiable sub-formula with at most t clauses.

In [Sze01] homomorphisms for CNF-formulas have been introduced as a tool for proving the unsatisfiability of formulas. For CNF-formulas H and F, a mapping $\phi : \mathrm{lit}(H) \longrightarrow \mathrm{lit}(F)$ is called a *homomorphism* from H to F if $\phi(\neg x) = \neg\phi(x)$ for every variable x, and $\phi(H) = \{\phi(h) \mid h \in H\} \subseteq F$, where $\phi(h) = \{\phi(L_1), \ldots, \phi(L_m)\}$ for a clause $h = \{L_1, \ldots, L_m\}$. According to [Sze01], for every tree resolution proof T one can find in polynomial time a formula $H \in \mathrm{MU}(1)$ and a homomorphism $\phi : H \to \mathrm{pre}(T)$, where $\mathrm{pre}(T)$ is the set of all premises of T (i.e., the clauses labelling the leaves) such that $\phi(H) = \mathrm{pre}(T)$, and $|H|$ equals the number of leaves of T.

Let $\mathcal{C}$ be a class of CNF-formulas. We define $\mathcal{HOM}(\mathcal{C})$ as the set of pairs (F, G) of CNF-formulas such that a homomorphism from F to G exists with image all of G. The problem $\mathcal{HOM}$(Horn-MU) is NP-complete. Clearly, the problem is in NP. The completeness can be shown by a reduction to the 3-colouring problem for graphs. Furthermore, for fixed $k \geq 1$ the problem $\mathcal{HOM}(\mathrm{MU}(k))$ is NP-complete, even we allow another fixed parameter $t \geq 1$ and consider all pairs (F, G) with $F \in \mathrm{MU}(k)$ and $G \in \mathrm{MU}(t)$ [KX05].

We say (H, ϕ) is a *representation* of F if $\phi : H \to F$ is a homomorphism with $\phi(H) = F$; so $\mathcal{HOM}(\mathcal{C})$ is the set of all pairs (F, G) such that F can be made a representation of G. Let (H, ϕ) be a representation of F, where F and H are minimal unsatisfiable formulas. If the shortest resolution refutation of F requires m steps then the shortest refutation for H needs at least m steps [KZ02b]. For all $k, t \geq 1$ and for all $F \in \mathrm{MU}(k)$ a representation of F by a MU(t)-formula can be computed in polynomial time.

A homomorphism ϕ with $\phi(H) = F$ is termed *clause–preserving* for H and F if H and F have the same number of clauses. As a consequence of results shown in [Sze01] and [KZ02b], we have that for every formula F for which a disjunctive splitting tree exists, we can find a formula $H \in \mathrm{MU}(1)$ for which a clause-preserving homomorphism ϕ with $\phi(H) = F$ exists, and vice versa.

11.4. Special Classes

In this section we present some MU-problems for restricted classes of formulas and some additional constraints.

11.4.1. 2-CNF-MU

For every $k \geq 1$ there is a formula $F \in$ 2-CNF-MU(k), the set of minimal unsatisfiable formulas with deficiency k consisting of 2-clauses. Whether a 2-CNF-formula is in MU can be decided in linear time. In a 2-CNF-MU formula, each literal occurs at most twice. Every 2-CNF-MU formula can be reduced by iterative (1,*)-resolution in linear time to a 2-CNF-MU formula in which each literal occurs exactly twice. These formulas have the following structure except for renaming

$$\begin{pmatrix} x_1 & \neg x_2 & \cdots & \neg x_{n-1} & \neg x_n & \neg x_1 & x_2 & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_n & x_1 & \neg x_2 & \neg x_3 & \cdots & \neg x_n & \neg x_1 \end{pmatrix}.$$

11.4.2. Hitting Formulas

Let HIT $:= \{F \in \text{CNF} \mid \forall f, g \in F, f \neq g\, \exists L \in f : \neg L \in g\}$ denote the set of "hitting formulas", formulas in which every pair of clauses has a complementary pair of literals. Let HIT-MU be the set of minimal unsatisfiable hitting formulas. In [Iwa89], a satisfiability algorithm has been introduced which counts for a given formula the number $\phi(F)$ of falsifying truth assignments (total assignments which for at least one clause falsify every literal in it). If the number is $\phi(F) = 2^n$, where n is the number of variables, then the formula is unsatisfiable, and otherwise satisfiable. The idea of this counting is based on "independent" sets of clauses. A set of clauses S is termed *independent* if every pair of (different) clauses contains no complementary pair of literals. The calculation of $\phi(F)$ can be expressed as follows for a formula $F = \{f_1, \cdots, f_m\}$:

$$\phi(F) = \sum_{1 \leq i \leq m} \sum_{S \in \text{IND}_i(F)} (-1)^{i-1} \cdot 2^{n-|\text{var}(S)|}$$

where $\text{IND}_i(F) := \{S \subseteq F \mid S \text{ independent and } |S| = i\}$. F is a hitting formula if and only if $\text{IND}_i(F) = \emptyset$ for $i \geq 2$; thus for $F \in$ HIT we have

$$\phi(F) = \sum_{f \in F} 2^{n-|f|}.$$

It follows for F in HIT:

1. $\sum_{f \in F} 2^{-|f|} \leq 1$;
2. $\sum_{f \in F} 2^{-|f|} < 1 \iff F \in \text{SAT}$;
3. $\sum_{f \in F} 2^{-|f|} = 1 \iff F \in \text{MU}$.

A special class of hitting clause-sets are k-regular hitting clause-sets, where between two different clauses always *exactly* k complementary (or "clashing") pairs of literals exist. A k-regular hitting clause-set for $k \neq 1$ is satisfiable, due to the

completeness of resolution. The unsatisfiable 1-regular hitting clause-sets have been characterised in [Kul04a] as those elements of MU(1) which are "saturated" or "maximal" (see below); an alternative combinatorial proof (not using methods from linear algebra as in [Kul04a]) has been given in [SST07].

11.4.3. Stronger forms of minimal unsatisfiability

Let $A \leq_p B$ denote the polynomial reducibility between A and B (i.e., A can be polynomially reduced to B), while $A =_p B$ is an abbreviation for $A \leq_p B$ and $B \leq_p A$. It is known that MARG-MU, MAX-MU and Almost-Unique-MU are D^P–complete, while Unique-MU $=_p$ Unique-SAT $\leq_p$ Dis-MU, where these classes are defined as follows:

1. *Maximal formulas* MAX-MU: A clause f of an MU-formula F is called *maximal* in F if for any literal L occurring neither positively nor negatively in f the formula obtained from F by adding L to f is satisfiable. Then $F \in$ MAX-MU if f is maximal in F for any $f \in F$. Another notion used in the literature is "saturated clause-sets"
2. *Marginal formulas* MARG-MU: A formula $F \in$ MU is called *marginal* w.r.t. a literal L if removing an arbitrary occurrence of L from F produces a non-minimal unsatisfiable formula. We say F is *marginal* if F is marginal w.r.t. all literals occurring in F. Or in other words, a formula is marginal if and only if removing an arbitrary literal always leads to a non-minimal unsatisfiable formula. The set of marginal formulas is denoted as MARG-MU.
3. *Unique minimal unsatisfiable formulas* Unique-MU: Based on the well-known concept of Unique-SAT, the set of formulas having exactly one satisfying truth assignment, we define the class Unique-MU as the class of MU-formulas F for which for any clause $f \in F$ we have $F - \{f\} \in$ Unique-SAT.
4. *Almost unique formulas* Almost-Unique-MU: A weaker notion which demands that except for one clause f, the reduced formula $F - \{f\}$ must be in Unique-SAT.
5. *Disjunctive Formulas* Dis-MU: We define $F \in$ Dis-MU if $F \in$ MU and for any $x \in \mathrm{var}(F)$ every splitting of F on x is disjunctive. Please note, that in case of a disjunctive splitting on a variable x the splitting on x is unique.

See [KZ07a] for more information on these classes.

11.4.4. Regarding the number of literal occurrences

For fixed $k \geq 3$, the problem "k-CNF-MU", recognising minimal unsatisfiable formulas in k-CNF, remains D^P-complete. The D^P-completeness of k-CNF-MU follows from the D^P-completeness of MU. We only have to replace longer clauses with shorter clauses by introducing new variables and by splitting the long clauses. Let k-CNF^p be the class of k-CNF-formulas in which each literal occurs at least p times. It has been shown in [KBZ02] that 3-CNF-MU^3, and for any fixed $k \geq 4$

and any $p \geq 2$ the classes k-CNF-MU^p are D^P-complete. But whether a formula in 3-CNF-MU exists in which each literal in F occurs at least 5 times is not known. More results can be found for example in [KZ02a].

11.4.5. Complement-invariant formulas

A formula F is *complement-invariant* if for every clause $\{L_1, \ldots, L_k\} \in F$ also the complemented clause is in F, that is $\{\neg L_1, \ldots, \neg L_k\} \in F$. A complement-invariant F, which additionally consists only of positive and negative clauses, can be identified with the underlying variable-hypergraph $\mathfrak{V}(F)$, whose vertices are the variables of F while the hyperedges are the variable-sets of clauses of F. It is easy to see that such F is satisfiable if and only if $\mathfrak{V}(F)$ is 2-colourable (i.e., vertices can be coloured using two colours such that no hyperedge is monochromatic), while F is minimally unsatisfiable if and only if $\mathfrak{V}(F)$ is minimally non-2-colourable (or "critically 3-colourable"), i.e., G is not 2-colourable but removing any hyperedge renders it 2-colourable. In this way critical-3-colourability of hypergraphs is embedded into the study of minimal unsatisfiable formulas. Complement-invariant formulas have been studied at various places, but apparently the first systematic treatment is in [Kul07b]. The deficiency of complement-invariant formulas F is less useful here, but the *reduced deficiency* $\mathrm{d_r}(F) := \mathrm{d}(\mathfrak{V}(F))$ is the central notion, where the deficiency $\mathrm{d}(G)$ of a hypergraph is the difference of the number of hyperedges and the number of vertices. As shown in [Sey74] (translated to our language), the reduced deficiency of minimally unsatisfiable $F(G)$ is at least 0. Solving a long outstanding open question, in [RST99, McC04] it was shown (again, in our language) that the decision problem whether a complement-invariant formula is minimally unsatisfiable with reduced deficiency 0 is decidable in polynomial time; more on the strong connections to the satisfiability problem the reader finds in [Kul07b], while in Subsection 11.12.2 of this chapter we further discuss this problem from the point of view of autarky theory.

11.4.6. Weaker forms of minimal unsatisfiability

The question whether an arbitrarily given formula can be transformed into a minimal unsatisfiable formula by replacing some literals with their complements has been studied in [Sze05]. It has been shown in [Sze05] that the problem whether a formula is satisfiable and remains satisfiable under such replacements is Π_2^P-complete.

The class of formulas which are the union of minimally unsatisfiable formulas has been studied at different places; see for example [KLMS06, Kul07a]. A generalisation of minimal unsatisfiability based on autarkies is discussed in Subsection 11.4.7.

11.4.7. Generalising minimal unsatisfiability for satisfiable formulas

Similarly to minimal unsatisfiable formulas, where removing a clause leads to a satisfiable formula, we say a formula F is *clause-minimal* (or "irredundant") if removing a clause f from F results in a formula not equivalent to F, i.e.

$F - \{f\} \not\equiv F$. Let CL-MIN := $\{F \in \text{CNF} \mid \forall f \in F : F - \{f\} \not\equiv F\}$ and CL-MIN(k) the set of CL-MIN-formulas with deficiency k. The problems CL-MIN and SAT $\cap$ CL-MIN are known to be NP-complete [PW88]. Furthermore, we have MU = UNSAT $\cap$ CL-MIN, and for any fixed k the problem CL-MIN(k) is NP-complete [KZ05]. Further results one finds in [Kul07a].

11.4.8. Extending minimal unsatisfiable formulas

The extension problem is the problem of determining whether for pairs of formulas (F, H) there is a formula G for which $F + G \equiv H$ and $\text{var}(G) \subseteq \text{var}(H + F)$. Without any restriction this question is equivalent to whether $H \models F$ (that is, whether H implies F). The latter problem is known to be coNP-complete. Now we are looking for extensions for MU-formulas. Let MU-EXT = $\{F \mid \exists G : F + G \in \text{MU}$ and $\text{var}(G) \subseteq \text{var}(F)\}$. Then we have MU-EXT = CL-MIN. Suppose, a CNF-formula F can be extended to a minimal unsatisfiable formula. Then it is easy to see that F is clause minimal. For the inverse direction suppose $F \in$ CL-MIN is satisfiable. Let $\{t_1, \cdots, t_r\}$ be the set of satisfying (total) truth assignments for F and let G consist of the r clauses exactly falsified by the assignments t_i; then $F + G$ is in MU. More details and further classes can be found in [KZ05]

11.5. Extension to non-clausal formulas

In this section, we will extend the notion of "minimal unsatisfiability" and "deficiency" to non-clausal propositional formulas, where for the sake of a simplified representation we only allow the operations "binary and", "binary or" and "not". Furthermore we study only formulas in "negation normal form", where we have negations only at the leaves, i.e., we study binary nested and's and or's of literals. The length of a propositional formula is the number of occurrences of literals. More formally, the length can be defined as $\ell(L) = 1$ for a literal L and $\ell(F \wedge G) = \ell(F \vee G) = \ell(F) + \ell(G)$. The number of occurrences of $\wedge$-symbols in a formula F is denoted by $\#_\wedge(F)$, whereas $\#_\vee(F)$ is the number of $\vee$-symbols. Obviously, we have $\ell(F) = 1 + \#_\wedge(F) + \#_\vee(F)$ (since the operations are binary). Following [KZ07b], we define:

Definition 11.5.1. Let F be a propositional formula in negation normal form with $n(F)$ variables. The *cohesion* $\text{D}(F)$ is defined as $\text{D}(F) = 1 + \#_\wedge(F) - n(F)$.

Example 11.5.1. The formula $F = (x \wedge y) \vee (\neg x \wedge (z \vee \neg y))$ has the cohesion $\text{D}(F) = 1 + \#_\wedge(F) - n(F) = 1 + 2 - 3 = 0$. An application of the distributive law $F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$ may change the cohesion. For example, from $F = z \wedge (x \vee y)$ we obtain the formula $F' = (z \wedge x) \vee (z \wedge y)$. The cohesion of F is -1 and the cohesion of F' is 0.

For propositional formulas in negation normal form the so-called "Tseitin procedure" generates satisfiability-equivalent formulas in CNF. Two formulas F and G are satisfiability-equivalent, if F is satisfiable if and only if G is satisfiable. For F a propositional formula in negation normal form, the Tseitin procedure

replaces step by step subformulas of the form $(F_1 \wedge F_2) \vee F_3$ by the formula $(z \vee F_1) \wedge (z \vee F_2) \wedge (\neg z \vee F_3)$ for a new variable z. Let $\mathrm{ts}(F)$ denote a formula in CNF obtained from F by this Tseitin procedure, eliminating the non-determinism in the procedure in some (arbitrary) way. Because the Tseitin procedure adds as many $\wedge$-symbols as new variables, we obtain $\mathrm{D}(\mathrm{ts}(F)) = \mathrm{D}(F)$.

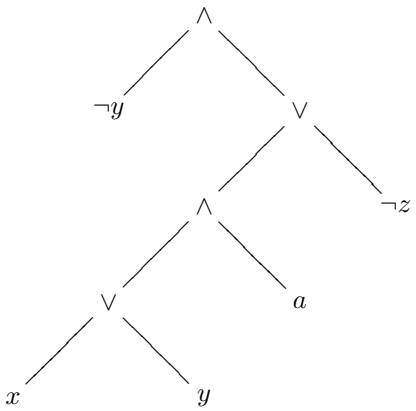
Lemma 11.5.1. *Let F be a propositional formula in negation normal form. Then we have $\mathrm{D}(F) = \ell(F) - \#_\vee(F) - n(F)$. and if $F \in$ CNF, then $\mathrm{D}(F) = \mathrm{d}(F)$.*

Minimal unsatisfiability is defined for formulas in conjunctive normal form. The formulas are unsatisfiable, and after eliminating an arbitrary clause the formulas are satisfiable. In non-clausal formulas, instead of the deletion of a clause we will remove so-called "or-subformulas" based on a representation of formulas as trees:

- If the formula is a literal L, the associated tree T_L is a node labelled with L.
- For a formula $F \wedge G$ (resp. $F \vee G$) let T_F and T_G be the associated trees. Then we have a root labelled with $\wedge$ (resp. $\vee$) and the subtrees T_F and T_G. That means, the successor nodes of the $\wedge$-node (resp. $\vee$-node) are the roots of T_F and T_G.

Please notice that the leaves of the tree are labelled with the literals of the formula. Let T be the associated tree of a formula. An *or-subtree* of T is a subtree T', whose root is either an $\vee$-node or a literal which is successor of an $\wedge$-node. An or-subtree T' of T is again a representation of a propositional formula, say $F_{T'}$, which is a subformula of the propositional formula of F. The formula $F_{T'}$ is called an *or-subformula* of F. In case of formulas in conjunctive normal form, the or-subformulas are exactly the clauses.

Example 11.5.2. For the formula $(\neg y) \wedge (((x \vee y) \wedge a) \vee \neg z)$ the associated tree is



and the four or-subformulas are $(\neg y)$, $((x \vee y) \wedge a) \vee (\neg z)$, $(x \vee y)$, and a.

In case of a propositional formula F (as always in negation normal form), an or-subformula of F is a subformula of F. For formulas in conjunctive normal

form, that is a conjunction of clauses, a satisfiable formula remains satisfiable after the deletion of any clause. Similarly, after the deletion of an or-subtree in a satisfiable formula the formula is still satisfiable. That follows from the definition of or-subtrees, because the predecessor node of an or-subtree is an $\wedge$-node, and so the deletion of an or-subtree does not decrease the truth-value of the formula at that $\wedge$-node, while the formulas are in negation normal form and thus all operations are monotonically increasing.

Definition 11.5.2. A propositional formula F in negation normal form is in MU^* if F is unsatisfiable and eliminating an arbitrary or-subformula yields a satisfiable formula. The set of formulas in MU^* with cohesion (exactly) k is denoted by $\mathrm{MU}^*(k)$.

It can easily be shown that for a CNF-formula being minimally unsatisfiable and being element of MU^* are equivalent properties. By the property of the Tseitin procedure, and the results known for MU and $\mathrm{MU}(k)$ the following theorem holds [KZ07b].

Theorem 11.5.2. *The following holds for the notion of minimal unsatisfiable propositional formulas according to Definition 11.5.2:*

1. MU^* *is* D^P*-complete.*
2. *For* $F \in \mathrm{MU}^*$ *we have* $\mathrm{D}(F) \geq 1$.
3. *For fixed* k, $\mathrm{MU}^*(k)$ *is decidable in polynomial time.*

There is an alternative definition of the class MU^*, based on the complementation of literal occurrences. Let MU_L be the set of unsatisfiable formulas F such that for all literal occurrences of F its replacement by its complement renders the formula satisfiable. Obviously, for formulas in CNF, any formula in MU_L is in MU and vice versa. For the general case, $\mathrm{MU}^* = \mathrm{MU}_L$ can be shown by an analysis of the Tseitin procedure $\mathrm{ts}(F)$.

11.6. Minimal Falsity for QBF

The concept of minimal unsatisfiability for CNF can be extended to QCNF, the class of closed quantified Boolean formulas in prenex form and with matrix in CNF; see Chapter 7 in [KBL99] for a general introduction to quantified propositional logic. All formulas F in QCNF have the form $F = Q_1 x_1 \cdots Q_n x_n f$, where $Q_i \in \{\exists, \forall\}$ and f is a propositional formula in conjunctive normal form using (only) the variables $x_1, \ldots, x_n$. $Q_1 x_1 \cdots Q_n x_n$ is the *prefix* of F, and f is called the *matrix* of F. The set of existential variables is denoted by $\mathrm{var}_\exists(F)$, while $\mathrm{var}(F)$ is the set of all variables. Let $F = Q_1 x_1 \cdots Q_n x_n f$, $F' = Q_1 x_1 \cdots Q_n x_n f'$ be two QCNF formulas with the same prefix. We say that F' is a *subformula* of F, denoted by $F' \subseteq F$, if f' is a subformula of f. Let Qf be a formula in QCNF with prefix Q. Then $f_{|\exists}$ is the conjunction of clauses we obtain after the deletion of all occurrences of literals with universal variables from f. Please note, that the propositional formula $f_{|\exists}$ may contain multiple occurrence of clauses. A formula $Q(f_1 \wedge \cdots \wedge f_n)$ in QCNF is called *minimal false*, if the formula is false and for

any clause f_i the formula $Q(f_1 \wedge \cdots \wedge f_{i-1} \wedge f_{i+1} \wedge \cdots \wedge f_n)$ is true. The class of minimal false formulas is denoted as MF.

The minimal falsity problem MF is PSPACE-complete [KZ06]. That can easily be shown by means of the PSPACE-completeness of the satisfiability problem for QCNF. Since the satisfiability problem for quantified Boolean Horn formulas (QHORN) and formulas with matrix in 2-CNF (Q2-CNF) is solvable in polynomial time, the minimal falsity problems for both classes are solvable in polynomial time, too. The notion of deficiency and maximal deficiency can be extended to QCNF, taking only the existential variables into account.

Definition 11.6.1. [KZ06]

1. Let $F = Qf \in$ QCNF with m clauses. The deficiency is defined as $\mathrm{d}(F) = m - |\mathrm{var}_\exists(f)|$.
2. The maximal deficiency of F is defined as $\mathrm{d}^*(F) := \max\{\mathrm{d}(F') : F' \subseteq F\}$.
3. Let k be fixed. The set of minimal false formulas with deficiency k is defined as $\mathrm{MF}(k) = \{F : F \in \mathrm{MF} \text{ and } \mathrm{d}(F) = k\}$.

Some of the methods and techniques well-known for MU(k) can be adapted in order to prove similar results. For example, a formula F is termed *stable* if for any proper subformula $F' \subset F$, $\mathrm{d}(F') < \mathrm{d}(F)$. The notion of stable QCNF formulas can be understood as a generalisation of matching lean CNF formulas [Kul03]. A formula $F \in$ CNF is matching lean if and only if $\mathrm{d}(F') < \mathrm{d}(F)$ for each proper subformula $F' \subset F$; see Subsection 11.11.2. By Definition 11.6.1, F is stable if and only if $F_{|\exists}$ is matching lean. Corollary 7.12 in [Kul03] says that the largest matching lean subformula can be computed in polynomial time. This implies that the largest stable subformula can also be computed in polynomial time.

Theorem 11.6.1. *[KZ08a] Let $F = Qf$ be a formula in* QCNF*.*

1. *If $f_{|\exists}$ is satisfiable or $\mathrm{d}^*(F) \leq 0$, then F is true.*
2. *If $F \in$ MF, then F is stable.*
3. *Any minimal false formula has deficiency strictly greater than 0.*

Besides MF(1), which is solvable in polynomial time as shown later in this section, the computational complexity of MF(k) is open, where only some non-trivial upper bounds are known.

Theorem 11.6.2. *[KZ06] Let k be fixed.*

1. *The satisfiability problem for* QCNF *with maximal deficiency k is in* NP*.*
2. *The minimal falsity problem* MF(k) *for* QCNF *is in D^P.*

The proof of Theorem 11.6.2 makes use of the size of models of satisfiable formulas in QCNF with maximal deficiency k. Let $F = \forall x_1 \exists y_1 \forall x_2 \exists y_2 \cdots \forall x_k \exists y_k f$ be a formula in QCNF. The formula is true if and only if there are Boolean functions (formulas) $\rho_i : \{0,1\}^i \to \{0,1\}$ for $i \in \{1,\ldots,k\}$ (depending on $x_1,\ldots,x_i$) such that

$$\forall x_1 \cdots \forall x_k f[y_1/\rho_1(x_1),\ldots,y_k/\rho_k(x_1,\ldots,x_k)]$$

is true. $f[y_1/\rho_1, \ldots, y_k/\rho_k]$ denotes the formula obtained by simultaneously replacing in the matrix the occurrences of the existential variables y_i by the formulas ρ_i. In general a sequence of Boolean functions $M = (\rho_1, \cdots, \rho_k)$ is called a *model* for F if the formula is true for these functions. We assume that the Boolean functions ρ_i are represented as propositional formulas. If the Boolean functions ρ_i are given as CNF-formulas, then we call M a *CNF-model* for F.

Theorem 11.6.3. *[KZ08a] For any $k \geq 1$ and any true* QCNF *formula $F = Qf$ with maximal deficiency $\mathrm{d}^*(F) = k$, there is a set U with at most $2^{4k/3}$ universal variables such that F has a* CNF*-model $M = (\rho_1, \ldots, \rho_m)$ where the formulas ρ_i are constants or fulfil $\mathrm{var}(\rho_i) \subseteq U$ and have at most 2^k clauses.*

The size of the model functions only depends on k. By guessing a set U of size less than or equal to $2^{4k/3}$ and model formulas ρ_i, and replacing x_i by ρ_i, we obtain a universally quantified formula with universal variables in U. Since k is fixed, the satisfiability of these formulas can thus be decided in polynomial time.

QEHORN (QE2-CNF respectively) is the set of formulas in QCNF with $F = Qf$ for which $f_{|\exists}$ is a Horn formula (2-CNF formula respectively). That means, the existential part of the matrix is in HORN (2-CNF respectively). The satisfiability problem for both classes is PSPACE-complete. Through unfolding these formulas by setting the universal variables to true and false, we get an existentially quantified Horn formula (2-CNF formula respectively). Since any true QCNF formula with maximal deficiency $k \geq 0$ has a model over at most $2^{4k/3}$ universal variables, the length of the unfolded formula can be bounded polynomially in the length of the initial formula. Since the satisfiability of Horn and 2-CNF formulas is decidable in polynomial time, we obtain the following result.

Lemma 11.6.4. *Let k be fixed.*

1. *The satisfiability problem for formulas in* QEHORN *and* QE2-CNF *with maximal deficiency k is solvable in polynomial time.*
2. *The minimal falsity problem for formulas in* QEHORN *and* QE2-CNF *with deficiency k can be decided in polynomial time.*

Formulas in MF(1) are closely related to formulas in MU(1). Every formula $F = Qf$ in MF(1) has a matrix $f_{|\exists} \in$ MU(1). But the other direction does not hold. A simple counterexample is as follows.

Example 11.6.1. Consider

$$F = \forall x \exists y\, (\neg x \vee y) \wedge (x \vee \neg y).$$

$((\neg x \vee y) \wedge (x \vee \neg y))_{|\exists} = (\neg x \wedge x) \in$ MF(1), but the formula F is satisfiable (by the model $y = x$).

Theorem 11.6.5. *[KZ08a] The minimal falsity problem* MF(1) *is solvable in polynomial time.*

The polynomial-time algorithm is based on the following observations:

1. The satisfiability problem for QCNF formulas with maximal deficiency 1 is solvable in polynomial time.

2. For any true QCNF formula with $\mathrm{d}^*(F) = 1$, there exists at most one universal variable y, such that there is a model in which the model functions are either the constants 0 or 1, or the formulas y or $\neg y$.
3. The clauses of a MF(1)-formula must satisfy some properties on the connectivity of clauses.

11.7. Applications and Experimental Results

The core problems of relevant applications in this area is to extract minimal unsatisfiable sub-formulas ("MUS's") of a CNF-formula, where more precisely three levels can be distinguished:

1. finding some "small" unsatisfiable sub-formula;
2. finding some minimal unsatisfiable sub-formula (MUS);
3. finding a smallest minimal unsatisfiable sub-formula.

Additionally we are also interested in finding all MUS's. Regarding finding "small" unsatisfiable sub-formulas, [BS01, Bru03] enhance and modify DPLL-solvers so that they target "hard clauses", from which the unsatisfiable sub-formula is obtained. [ZM04] also only finds "small" unsatisfiable sub-formulas, but uses the resolution refutation found by conflict-driven SAT solvers. Regarding finding MUS's, a natural and simple algorithm is given in [vW08], while learning of conflict-driven SAT solvers is exploited in [OMA$^+$04], and [GMP06, GMP07a] use the information obtained by local search solvers on maximally satisfiable sub-formulas (see [GMP07b] for generalisations regarding constraint satisfaction problems). For finding smallest MUS's, in [Bru05] ideas from linear programming are applied to special classes of CNF-formulas, while a general branch-and-bound algorithm is given in [MLA$^+$05]. Regarding finding all MUS's, [LS05] exploit the duality between MUS's and maximally satisfiable sub-formulas ("MSS's"; the generalisation for non-boolean variables and irredundant clause-sets is given in Lemma 6.11 in [Kul07a]). Finally, connecting MUS's with "autarkies" (as a weaker form of redundancies), we mention [KLMS06, LS08].

11.8. Generalising satisfying assignments through "autarkies"

At several places did we already encounter the notion of an "autarky". In the subsequent sections we give an overview on the emerging theory of autarkies.

11.8.1. Notions and notations

In the previous section the "logical aspect" of SAT has been emphasised, and thus the basic objects of study were called "formulas". Autarky theory takes a more combinatorial turn and needs to take full account of "syntactical details", and thus we will now speak of *clause-sets*, which are (here only finite) sets of *clauses*, where a clause is a finite and complement-free set of literals. Some useful notions:

- The *empty clause* is denoted by $\bot := \emptyset$, while the *empty clause-set* is denoted by $\top := \emptyset$.

- For a literal x we denote the underlying variable by $\mathrm{var}(x)$, while for a clause C we define $\mathrm{var}(C) := \{\mathrm{var}(x) : x \in C\}$, and $\mathrm{var}(F) := \bigcup_{C \in F} \mathrm{var}(C)$ for a clause-set F.
- The number of clauses of a clause-set F is $c(F) := |F| \in \mathbb{N}_0$, while the number of variables is $n(F) := |\mathrm{var}(F)| \in \mathbb{N}_0$.

An important step is to emancipate the notion of a *partial assignment*, which is a map $\varphi : V \to \{0,1\}$ where V is some finite set of variables (possibly the empty set); we use $\mathrm{var}(\varphi) := V$ to denote the domain of φ.[2] Using $\varphi(\overline{v}) = \overline{\varphi(v)}$, partial assignments are extended to literals (over their domain). The application of a partial assignment φ to a clause-set F is denoted by $\varphi * F$, and is defined as the clause-set obtained from F by first removing all clauses satisfied by φ, and then removing from the remaining clauses all literal occurrences which are falsified by φ. Partial assignments φ, ψ can be combined by the operation $\varphi \circ \psi$ which is the partial assignment with domain the union of the domains of φ and ψ, while $(\varphi \circ \psi)(v)$ is defined as $\psi(v)$ if possible and otherwise $\varphi(v)$. Using $\langle\rangle$ for the empty partial assignment, we have the following fundamental laws:

$$(\varphi \circ \psi) \circ \theta = \varphi \circ (\psi \circ \theta)$$
$$\varphi \circ \langle\rangle = \langle\rangle \circ \varphi = \varphi$$
$$\varphi * (\psi * F) = (\varphi \circ \psi) * F$$
$$\langle\rangle * F = F$$
$$\varphi * \top = \top$$
$$\varphi * (F_1 \cup F_2) = \varphi * F_1 \cup \varphi * F_2$$
$$\bot \in F \Rightarrow \bot \in \varphi * F.$$

A clause-set F is satisfiable if it has a satisfying assignment φ, that is, fulfilling $\varphi * F = \top$ (while a "falsifying assignment" fulfils $\bot \in \varphi * F$). We need some specific notions to fix the variables involved in satisfying assignments:

Definition 11.8.1. For a clause-set F and a set of variables V with $\mathrm{var}(F) \subseteq V$ let $\mathrm{mod}_V(F)$ by the set of satisfying assignments φ with domain exactly V, while we use $\mathrm{mod}(F) := \mathrm{mod}_{\mathrm{var}(F)}(F)$.

Satisfying partial assignments in the sense of Definition 11.8.1, with a fixed domain including all variables in F, are called "total satisfying assignments". We denote the restriction of a partial assignment φ to the domain $V \cap \mathrm{var}(\varphi)$ by $\varphi|V$ for a set V of variables. Finally, for a partial assignment φ we denote by C^ε_φ for $\varepsilon \in \{0,1\}$ the clause consisting of exactly the literals x with $\varphi(x) = \varepsilon$; thus C^0_φ consists of the literals falsified by φ, the "CNF representation" of φ, while C^1_φ consists of the literals satisfied by φ, the "DNF representation" of φ.

Partial assignments assign truth values $0, 1$ to values, and their operation on clause-sets F by $\varphi * F$ handles literals accordingly; a simpler operation is that by a (finite) set V of variables, where now literals involving these variables are simply crossed out, and we write $V * F$ for this operation. While for partial assignments

[2] Often partial assignments are identified with clauses, interpreting them as setting their literals to 1 (for CNF); for a systematic treatment however this convenient "trick" is better avoided, and we put emphasis on the different roles played by clauses and partial assignments.

the above laws just reflect, that we have an operation of the monoid of partial assignments (see Subsection 11.9.1) on the (upper) semilattice of clause-sets, here now we have the operation of finite sets of variables as (upper) semilattice, with set-union as composition, on the (upper) semilattice of clause-sets, with the laws $V*(W*F) = (V \cup W)*F$, $\emptyset * F = F$, $V * \top = \top$, $V * (F_1 \cup F_2) = V * F_1 \cup V * F_2$, and also $\bot \in F \Rightarrow \bot \in V * F$.

In order to give some examples, we need some notation for specific partial assignments, and we use for example "$\langle x \to 1, y \to 0\rangle$" to denote the partial assignment which sets variable x to 1 (i.e., true) and y to 0 (i.e., false); instead of variables also literals can be used here.

Example 11.8.1. Using four (different) variables a, b, x, y and the clause-set $F := \{\{a, b\}, \{x, \overline{a}\}, \{y, a, b\}, \{\overline{x}, y, b\}\}$ we have $\langle x \to 1, y \to 0\rangle * F = \{\{a, b\}, \{b\}\}$ and $\{a, x\} * F = \{\{b\}, \bot, \{y, b\}\}$.

Often we need to select clauses containing some variables from a given set of variables, and we do this by $F_V := \{C \in F : \mathrm{var}(C) \cap V \neq \emptyset\}$ for clause-sets F and sets of variables V. Finally we need to *restrict* clause-sets F to some set V of variables, and, generalising a standard notion from hypergraph theory, we use $F[V]$ for this operation, defined by

$$F[V] := (\mathrm{var}(F) \setminus V) * F_V.$$

Note that we have $F[V] = ((\mathrm{var}(F) \setminus V) * F) \setminus \{\bot\}$. Finally we remark that for more combinatorially-oriented investigations often clauses need to occur several times, and thus (at least) *multi*-clause-sets are needed; the reader finds the generalisation of the considerations here to multi-clause-sets in [Kul07a], while in Subsection 11.11.3.2 we discuss the more general notion of "labelled clause-sets".

11.8.2. Autarkies

Now we come to the fundamental notion of an "autarky". The term was coined by [MS85] for a partial assignment whose application produces a sub-clause-set, while implicitly used in works like the similar [Luc84] or in the earlier [EIS76]. Since these works took a more "logical" point of view (they were just concerned with deciding satisfiability, not with more detailed structural investigations), the underlying notion of an "autarky" was ambiguous w.r.t. whether contractions of clauses after application of partial assignments was taken into account or not. This ambiguity is resolved by distinguishing between "weak autarkies" and "(normal) autarkies":

Definition 11.8.2. A partial assignment φ is called a **weak autarky for F** if $\varphi * F \subseteq F$ holds, while φ is an **autarky for** F if φ is a weak autarky for all sub-clause-sets F' of F.

Every satisfying assignment for F is an autarky for F. Thus every partial assignment is an autarky for $\top$.

Example 11.8.2. The partial assignment $\langle b \to 0\rangle$ is a weak autarky for $F := \{\{a\}, \{a, b\}\}$, but not an autarky for F. The partial assignment $\langle b \to 1\rangle$ is an autarky for F.

The basic fact about (weak) autarkies is the observation:

If φ is a weak autarky for F, then $\varphi * F$ is satisfiability equivalent to F.

(For every partial assignment φ we have that from $\varphi * F$ being satisfiable follows F being satisfiable, while the other direction, that if F is satisfiable then $\varphi * F$ is satisfiable, follows from $\varphi * F \subseteq F$.) A well-known reduction for SAT-solving is the elimination of pure literals, and this is a special case of an "autarky reduction" (see Subsection 11.8.3 for the treatment of full autarky reduction).

Example 11.8.3. A *pure literal* (also called "monotone literal") for a clause-set F is a literal x such that $\overline{x}$ does not occur in F (i.e., $\overline{x} \notin \bigcup F$). Every pure literal x for F yields an autarky $\langle x \to 1 \rangle$ for F. In Example 11.8.2, literals a, b are pure literals for F. See Subsection 11.11.1 for more on "pure autarkies".

Autarkies are easily characterised by

$$\begin{array}{c}\varphi \text{ is an autarky for } F \text{ if and only if for every clause } C \in F\\ \text{either } \varphi \text{ does not "touch" } C, \text{ i.e., } \mathrm{var}(\varphi) \cap \mathrm{var}(C) = \emptyset,\\ \text{or } \varphi \text{ satisfies } C, \text{ i.e., } \varphi * \{C\} = \top.\end{array}$$

Weak autarkies are allowed to touch clauses without satisfying them, if the resulting clause (after removal of falsified literals) was already present in the original clause-set. Since we regard it as essential that an autarky for a clause-set is also an autarky for every sub-clause-set, the fundamental notion is that of an "autarky"; see Subsection 11.8.4 for a discussion. The basic characterisations of autarkies in terms of satisfying assignments are given by the following equivalent statements (for arbitrary clause-sets F and partial assignments φ):

1. φ is an autarky for F
2. iff φ is a satisfying assignment for $F_{\mathrm{var}(\varphi)}$
3. iff φ is a satisfying assignment for $F[\mathrm{var}(\varphi)]$.

The following properties of autarkies are considered to be the most basic ones (leading to an "axiomatic" generalisation in Section 11.11), for arbitrary clause-sets F, partial assignments φ and sets V of variables:

1. $\langle\rangle$ is an autarky for F. In general, φ is an autarky for F iff the restriction $\varphi|\,\mathrm{var}(F)$ is an autarky for F, and φ is called a *trivial autarky* for F if $\mathrm{var}(\varphi) \cap \mathrm{var}(F) = \emptyset$ (i.e., iff $\varphi|\,\mathrm{var}(F) = \langle\rangle$).
2. If φ is an autarky for F, then φ is also an autarky for $F' \subseteq F$.
3. If φ, ψ are autarkies for F then also $\psi \circ \varphi$ is an autarky for F. More generally, if φ is an autarky for F and ψ is an autarky for $\varphi * F$, then $\psi \circ \varphi$ is an autarky for F.
4. If $\mathrm{var}(\varphi) \cap V = \emptyset$, then φ is an autarky for $V * F$ iff φ is an autarky for F.
5. φ is an autarky for F iff φ is an autarky for $F \cup \{\bot\}$.
6. If φ is an autarky for F, and F' is isomorphic to F (by renaming variables and flipping polarities; compare Subsection 11.9.5), then applying the same renamings and flipping of polarities to φ we obtain an autarky φ' for F'.

An *autark subset* (or "autark sub-clause-set") of F is some $F' \subseteq F$ which is (exactly) satisfied by some autarky φ for F, i.e., $F' = F \setminus (\varphi * F)$. $\top$ is an

autarky sub-clause-set of F, and if F_1, F_2 are autark sub-clause-sets of F, then so is $F_1 \cup F_2$. It follows that there is the *largest autark sub-clause-set* of F.

11.8.3. Autarky reduction and lean clause-sets

Given an autarky φ for clause-set F, we can reduce F satisfiability-equivalent to $\varphi * F$. A clause-set F is called *lean* if no autarky-reduction is possible, that is, if every autarky for F is trivial. This notion was introduced in [Kul00b], and further studied in [Kul03, Kul07a]. Examples for lean clause-sets are $\top$, all minimally unsatisfiable clause-sets, and clause-sets obtained from lean clause-sets by the extension rule of extended resolution (see Lemma 3.2 in [Kul00b]). Except of $\top$, every lean clause-set is unsatisfiable, and F is lean iff $F \cup \{\bot\}$ is lean. If F is lean, then so is $V * F$ and the restriction $F[V]$. The union of lean clause-sets is again lean, and thus, as with autark subsets, the lean sub-clause-sets of a clause-set form a set stable under union, with smallest element $\top$, while the largest element is called the **lean kernel** and denoted by $\mathrm{N_a}(F)$.

Another possibility to define the lean kernel is by using that autarky reduction is confluent: By repeating autarky-reduction for a clause-set F as long as possible, i.e., until we arrive at a lean sub-clause-set $F' \subseteq F$, we obtain the lean kernel $\mathrm{N_a}(F) = F'$ (here the letter "N" stands for "normal form"). The lean kernel can be characterised in many other ways:

1. $\mathrm{N_a}(F)$ is the largest lean sub-clause-set of F.
2. $F \setminus \mathrm{N_a}(F)$ is the largest autark subset of F.
3. The decomposition $F = \mathrm{N_a}(F) \cup (F \setminus \mathrm{N_a}(F))$ is characterised as the unique decomposition $F = F_1 \cup F_2$, $F_1 \cap F_2 = \emptyset$, such that
   (a) F_1 is lean, and
   (b) $\mathrm{var}(F_1) * F_2$ is satisfiable.
4. So $\mathrm{N_a}(F)$ is the unique $F' \subseteq F$ such that F' is lean and $\mathrm{var}(F') * (F \setminus F')$ is satisfiable.

The operator $\mathrm{N_a}$ is a kernel operator, that is, we have $\mathrm{N_a}(F) \subseteq F$, $F \subseteq G \Rightarrow \mathrm{N_a}(F) \subseteq \mathrm{N_a}(G)$, and $\mathrm{N_a}(\mathrm{N_a}(F)) = \mathrm{N_a}(F)$; furthermore we have $\mathrm{N_a}(F) = \top$ iff F is satisfiable, while $\mathrm{N_a}(F) = F$ iff F is lean.

We now consider the fundamental *duality* between autarkies and resolution. A precursor of this duality in the context of tableaux calculi has been established in [Van99] (see especially Theorems 5.2, 5.3 there); in Subsection 11.10.2 we will further discuss this work. We recall the resolution rule, which allows for "parent clauses" C, D which clash in exactly one literal x, i.e., $C \cap \overline{D} = \{x\}$, to derive the "resolvent" $R := (C \setminus \{x\}) \cup (D \setminus \{\overline{x}\})$. A resolution refutation for F is a (rooted) binary tree labelled with clauses, such that the leaves are labelled with clauses from F, every inner node is labelled by a resolvent of its two children, and the root is labelled with $\bot$. F is unsatisfiable iff it has a resolution refutation. We remark that such resolution refutations do not allow for "dead ends", and that handling "full resolution", i.e., resolution in dag-form, where clauses can be parent-clauses for several resolution steps, can be achieved in this framework by not counting nodes of the tree but only counting different clauses. A basic observation is that

a clause $C \in F$ containing a pure literal cannot be used (i.e., label a leaf) in any resolution refutation, since we can never get rid off this pure literal. This can be generalised to the fact that if φ is an autarky for F satisfying some leaf clause of a resolution tree, then φ also satisfies every clause on the path from this leaf to the root (the autarky condition prevents the possibility that all satisfied literals vanish together). Thus a clause satisfied by some autarky cannot be part of any resolution refutation. In Section 3.3 in [Kul00b] it was shown that also the reverse direction holds.

Theorem 11.8.1. *For every clause-set F the lean kernel $\mathrm{N_a}(F)$ consists of exactly the clauses from F which can be used in some (tree) resolution refutation of F (without dead-ends). In other words, there exists an autarky satisfying some clause $C \in F$ if and only if C cannot be used in any resolution refutation of F.*

Theorem 11.8.1 yields a method for computing the lean kernel, which will be discussed in Subsection 11.10.3.

11.8.4. Remarks on autarkies vs. weak autarkies

A weak autarky φ for a clause-set F is not an autarky for F iff there exists some $F' \subset F$ such that φ is not a weak autarky for F'; e.g., in Example 11.8.2 $\langle b \to 0\rangle$ is not a weak autarky for $\{\{a,b\}\} \subset F = \{\{a\},\{a,b\}\}$. So the property of being a weak autarky is not inherited by sub-clause-sets, which makes weak autarkies a more fragile, "accidental" or "non-algebraic" concept. Nevertheless, weak autarkies share many properties with autarkies:

1. If φ is a weak autarky for F, and ψ a weak autarky for $\varphi * F$, then $\psi \circ \varphi$ is a weak autarky for F (this follows by definition).
2. Consider a weak autarky ψ for F; as mentioned, if ψ is not an autarky for F then ψ is not a weak autarky for some sub-clause-set F' of F. However ψ is a weak autarky for each $F' \subseteq F$ such that there is a weak autarky φ for F with $F' = \varphi * F$ (since if ψ is not a weak autarky for F' anymore, then the "absorbing" clause $C \in F$ for some "new" clause $\psi * D = C$, $D \in F'$, created by ψ, must have vanished, which in this case means that $\varphi * \{C\} = \top$, implying $\varphi * \{D\} = \top$, and thus also D vanished).
3. It follows that the composition of weak autarkies is again a weak autarky. So the weak autarkies for a clause-set F form a sub-monoid of $(\mathcal{PASS}, \circ, \langle\rangle)$, containing the autarky monoid $\mathrm{Auk}(F)$ (see Section 11.9 for the autarky monoid).
4. Furthermore reduction by weak autarkies is confluent, yielding a (well-defined) sub-clause-set of $\mathrm{N_a}(F)$.

11.9. The autarky monoid

We have already mentioned that the composition of autarkies yields again an autarky. This fundamental observation was made in [Oku00], and it initiated the study of the *autarky monoid* in [Kul00b], continued in [KMT08]. In this subsection we discuss the basic features of the autarky monoid. Some prerequisites

from algebra are used, and the following literature might help to provide the missing (always elementary) definitions: [Lau06] in Chapters 1 - 4 provides basic material on algebraic structures and hull ("closure") systems; [Bou89], Sections §I.1 - §I.5, presents the basics on monoids, groups and operations in a systematic way, while Chapter I in [Lan02] presents such material (including categories) in a somewhat more advanced fashion; regarding category theory an accessible introduction is given by the first two chapters of [Pie91].

11.9.1. The monoid of partial assignments

Since "autarky monoids" are sets of autarkies together with the composition of partial assignments, we need to take a closer look at the monoid $\mathcal{PASS}$ of all partial assignments. Starting with an arbitrary set $\mathcal{VA}$ of "variables", we obtain the set $\mathcal{LIT}$ of literals (uncomplemented and complemented variables), the set $\mathcal{CL}$ of clauses (finite and clash-free sets of literals) and the set $\mathcal{CLS}$ of clause-sets (finite sets of clauses), while $\mathcal{PASS}$ is the set of maps $\varphi : V \to \{0,1\}$ for some finite $V \subseteq \mathcal{VA}$.[3] The monoid $(\mathcal{PASS}, \circ, \langle\rangle)$ of partial assignments (together with the composition of partial assignments, and the empty partial assignment as the neutral element) has the following properties:

1. $\mathcal{PASS}$ is generated by the elementary partial assignments $\langle v \to \varepsilon\rangle$ for variables $v \in \mathcal{VA}$ and $\varepsilon \in \{0,1\}$. The defining relations between elementary partial assignments (those relations which specify the composition of $\mathcal{PASS}$) are

   (a) $\langle v \to \varepsilon\rangle \circ \langle v \to \varepsilon'\rangle = \langle v \to \varepsilon'\rangle$;
   (b) $\langle v \to \varepsilon\rangle \circ \langle w \to \varepsilon'\rangle = \langle w \to \varepsilon'\rangle \circ \langle v \to \varepsilon\rangle$ for $v \neq w$.

   For a partial assignment $\varphi \in \mathcal{PASS}$ we have the unique representation $\varphi = \circ_{v \in \mathrm{var}(\varphi)} \langle v \to \varphi(v)\rangle$ (using the commutativity of elementary partial assignments for different variables).
2. The closure (hull) of a finite set $P \subseteq \mathcal{PASS}$ of partial assignments under composition (i.e., the generated monoid) is again finite (since every partial assignments involves only finitely many variables).
3. $\mathcal{PASS}$ is idempotent, i.e., for all $\varphi \in \mathcal{PASS}$ we have $\varphi \circ \varphi = \varphi$.
4. The following assertions are equivalent for $\varphi, \psi \in \mathcal{PASS}$:

   (a) φ, ψ commute (i.e., $\varphi \circ \psi = \psi \circ \varphi$);
   (b) φ, ψ are compatible (i.e., $\forall\, v \in \mathrm{var}(\varphi) \cap \mathrm{var}(\psi) : \varphi(v) = \psi(v)$);
   (c) there exists a partial assignment $\theta \in \mathcal{PASS}$ with $\varphi \subseteq \theta$ and $\psi \subseteq \theta$.

[3] The set $\mathcal{VA}$ may be some finite set; more useful would be the choice $\mathcal{VA} = \mathbb{N}$, while actually for a theoretical study the choice of $\mathcal{VA}$ as a universe of set theory (a set stable under all set-theoretical operations) is most appropriate. Literals are best represented via $\mathcal{LIT} := \mathcal{VA} \times \{0,1\}$, where 0 stands for "uncomplemented". And a map f with domain X is a set $\{(x, f(x)) : x \in X\}$ of ordered pairs, so that a partial assignment is formally the same as a clause, namely a set of pairs (v, ε), only that now the interpretation is the opposite: while a literal $(v, \varepsilon) \in C$ in a clause C means "v shall not get value ε", an assignment $(v, \varepsilon) \in \varphi$ for a partial assignment φ means that v gets value ε; this duality reflects that clauses are part of a CNF representation of the underlying boolean function, while (satisfying) partial assignments are part of a DNF representation.

For a partial assignment $\varphi \in \mathcal{PASS}$ the submonoid $\mathcal{PASS}(\varphi) := \{\psi \in \mathcal{PASS} : \psi \subseteq \varphi\}$ of partial assignments contained in φ is commutative, and for every finite commutative submonoid M of $\mathcal{PASS}$ there exists a partial assignment φ with $M \subseteq \mathcal{PASS}(\varphi)$.

5. The partial order $\subseteq$ between partial assignments[4] has the following properties:
   (a) $\langle\rangle$ is the smallest element; there is no largest element iff $\mathcal{VA}$ is not empty, and there are no maximal elements iff $\mathcal{VA}$ is infinite.
   (b) The infimum of any set $\emptyset \neq P \subseteq \mathcal{PASS}$ of partial assignments is the intersection of the elements of P.
   (c) $P \subseteq \mathcal{PASS}$ has an upper bound iff P is finite and all elements of P are pairwise compatible, and in this case P has a supremum (namely the union).
   (d) The order is right-compatible with the composition, that is, for partial assignments $\varphi, \psi, \theta \in \mathcal{PASS}$ with $\varphi \subseteq \psi$ we have $\varphi \circ \theta \subseteq \psi \circ \theta$.
   However, the order is not left-compatible with the composition, so for example we have $\langle v \to 0\rangle \supset \langle\rangle$, but $\langle v \to 1\rangle \circ \langle v \to 0\rangle = \langle v \to 0\rangle \not\supseteq \langle v \to 1\rangle \circ \langle\rangle = \langle v \to 1\rangle$.
   (e) The order is identical with the natural right-compatible order of the monoid $\mathcal{PASS}$, that is, $\varphi \subseteq \psi \Leftrightarrow \psi \circ \varphi = \psi$ (while $\mathrm{var}(\varphi) \subseteq \mathrm{var}(\psi) \Leftrightarrow \varphi \circ \psi = \psi$).
   (f) Since the smallest element is the neutral element, $\mathcal{PASS}$ is "right non-negative", and we always have $\varphi \subseteq \psi \circ \varphi$ (while we only have $\mathrm{var}(\varphi) \subseteq \mathrm{var}(\varphi \circ \psi)$).
6. The formation of the domain of partial assignments, i.e., the map var, is a (surjective) homomorphism from the monoid $\mathcal{PASS}$ to the monoid of finite subsets of $\mathcal{VA}$ together with union, that is, $\mathrm{var}(\varphi \circ \psi) = \mathrm{var}(\varphi) \cup \mathrm{var}(\psi)$ and $\mathrm{var}(\langle\rangle) = \emptyset$. Restricted to the commutative submonoid $\mathcal{PASS}(\varphi)$ for some $\varphi \in \mathcal{PASS}$, var is an isomorphism from $\mathcal{PASS}(\varphi)$ to $\mathbb{P}(\mathrm{var}(\varphi))$ (the powerset of the domain of φ).

A natural representation of $\mathcal{PASS}$ as a transformation monoid is obtained as follows: Assume $0, 1 \notin \mathcal{VA}$, let $\mathcal{VA}' := \mathcal{VA} \cup \{0, 1\}$, and define the operation $* : \mathcal{PASS} \times \mathcal{VA}' \to \mathcal{VA}'$ by $\varphi * v := \varphi(v)$ for $v \in \mathrm{var}(\varphi)$, and $\varphi * x := x$ otherwise. Now we have $\langle\rangle * x = x$ and $(\varphi \circ \psi) * x = \varphi * (\psi * x)$ for $x \in \mathcal{VA}'$. Furthermore the operation is faithful, i.e., for $\varphi \neq \psi$ there is $x \in \mathcal{VA}'$ with $\varphi * x \neq \psi * x$. The corresponding injective monoid-morphism from $\mathcal{PASS}$ into the set of transformations of $\mathcal{VA}'$ represents partial assignments φ and their composition by transformations $(\varphi * x)_{x \in \mathcal{VA}'} : \mathcal{VA}' \to \mathcal{VA}'$ and their composition.

11.9.2. Autarky monoid and autarky semigroup

As already mentioned, the composition of autarkies is again an autarky, which can be easily seen as follows: Consider autarkies φ, ψ for F, that is, for all $F' \subseteq F$

[4] $\varphi \subseteq \psi$ holds iff $\mathrm{var}(\varphi) \subseteq \mathrm{var}(\psi)$ and $\forall\, v \in \mathrm{var}(\varphi) : \varphi(v) = \psi(v)$; we have $\varphi \subseteq \psi$ iff $C^0_\varphi \subseteq C^0_\psi$ iff $C^1_\varphi \subseteq C^1_\psi$

we have $\varphi * F' \subseteq F'$ and $\psi * F' \subseteq F'$. Now we have $(\varphi \circ \psi) * F' = \varphi * (\psi * F') \subseteq \varphi * F'' \subseteq F''$ for $F'' := \psi * F' \subseteq F'$, and thus $(\varphi \circ \psi)$ is an autarky for F.

Definition 11.9.1. For a clause-set $F \in \mathcal{CLS}$ the **autarky monoid**, denoted by $\mathrm{Auk}(F)$, is the sub-monoid of $\mathcal{PASS}$ given by all autarkies for F, while the **autarky semigroup** is $\mathrm{Auks}(F) := \mathrm{Auk}(F) \setminus \{\langle\rangle\}$. For performing computations, the **restricted** versions $\mathrm{Auk}^{\mathrm{r}}(F) := \{\varphi \in \mathrm{Auk}(F) : \mathrm{var}(\varphi) \subseteq \mathrm{var}(F)\}$ and $\mathrm{Auks}^{\mathrm{r}}(F) := \mathrm{Auk}^{\mathrm{r}}(F) \setminus \{\langle\rangle\}$ are preferable.

Remarks:

1. $\mathrm{Auk}(F), \mathrm{Auk}^{\mathrm{r}}(F)$ are submonoids of $\mathcal{PASS}$, where $\mathrm{Auk}^{\mathrm{r}}(F)$ is always finite, while $\mathrm{Auk}(F)$ is infinite iff $\mathcal{VA}$ is infinite. $\mathrm{Auks}(F), \mathrm{Auks}^{\mathrm{r}}(F)$ are subsemigroups of $\mathcal{PASS} \setminus \{\langle\rangle\}$.
2. $\mathrm{Auk}(F)$ is obtained from the elements of $\mathrm{Auk}^{\mathrm{r}}(F)$ by extending them arbitrarily on variables not in $\mathrm{var}(F)$.
3. For every set X the "right-null semigroup" is the semigroup $(X, *)$ with $x * y := y$. Now the set $\mathrm{mod}(F)$ of total satisfying assignments for a clause-set F is a right-null sub-semigroup of $\mathrm{Auks}^{\mathrm{r}}(F)$, and if F is satisfiable, then this set equals the set of "maximal autarkies", as discussed below.

Some simple examples, where we know the full autarky monoid:

1. F is lean iff $\mathrm{Auk}^{\mathrm{r}}(F) = \{\langle\rangle\}$ iff $\mathrm{Auks}^{\mathrm{r}}(F) = \emptyset$.
2. Consider the unique clause-set F with $\mathrm{var}(F) = \{v_1, \ldots, v_n\}$ $(n \geq 1)$ which is equivalent (as CNF) to the condition $v_1 \oplus \cdots \oplus v_n = \varepsilon \in \{0, 1\}$, where "$\oplus$" is exclusive-or ($F$ consists of 2^{n-1} clauses, each involving all variables). Now every non-trivial autarky for F is a satisfying assignment, and every satisfying assignment involves all variables, and so the restricted autarky semigroup of F is the right-null semigroup on the set of the 2^{n-1} satisfying assignments.

Basic properties of the autarky monoid are as follows:

1. Let $\mathrm{var}(\mathrm{Auk}(F)) := \mathrm{var}(\mathrm{Auk}^{\mathrm{r}}(F)) := \bigcup_{\varphi \in \mathrm{Auk}^{\mathrm{r}}(F)} \mathrm{var}(\varphi)$. Then the largest autark subset of F is $F_{\mathrm{var}(\mathrm{Auk}(F))}$.
2. Since $\mathrm{Auk}^{\mathrm{r}}(F)$ is finite, $\mathrm{Auk}^{\mathrm{r}}(F)$ has maximal elements, called *maximal autarkies*, and for every maximal autarky φ we have $\mathrm{var}(\varphi) = \mathrm{var}(\mathrm{Auk}(F))$, while $F \setminus (\varphi * F)$ is the largest autark subset of F. We denote the set of maximal autarkies of F by $\mathrm{Auk}{\uparrow}(F)$. Note that F is lean iff $\mathrm{Auk}{\uparrow}(F) = \{\langle\rangle\}$.
3. The maximal autarkies of F are exactly the total satisfying assignments for $F[\mathrm{var}(\mathrm{Auk}(F))]$. So $\mathrm{Auk}{\uparrow}(F)$ is a right-null sub-semigroup of $\mathrm{Auks}^{\mathrm{r}}(F)$. If F is satisfiable, then $\mathrm{Auk}{\uparrow}(F) = \mathrm{mod}(F)$.
4. The minimal elements of the restricted autarky semigroup $\mathrm{Auks}^{\mathrm{r}}(F)$ are called *minimal autarkies*. F has a minimal autarky iff F is not lean, and the minimal autarkies are exactly the atoms of the partial order $\mathrm{Auk}^{\mathrm{r}}(F)$. (One could thus speak of "atomic autarkies", but it seems that the slightly different contexts for "maximal" and "minimal" autarkies do not cause confusion.) We denote the set of minimal autarkies of F by $\mathrm{Auk}{\downarrow}(F)$ (so F is lean iff $\mathrm{Auk}{\downarrow}(F) = \emptyset$).

If only the autark *subsets* are of interest (not their certificates, i.e., their associated autarkies), then the (upper) semilattice (with zero)

$$\mathrm{Auk}^{\mathrm{s}}(F) := \{F' \subseteq F \mid \exists\, \varphi \in \mathrm{Auk}(F) : \varphi * F = F \setminus F'\}$$

of autark subsets is to be studied (the operation is just set-union, the neutral element is $\top$). The canonical homomorphism $\Phi : \mathrm{Auk}(F) \to \mathrm{Auk}^{\mathrm{s}}(F)$ given by $\varphi \mapsto F \setminus (\varphi * F)$ is surjective, and the inverse image of $\top$ is the set of trivial autarkies. Also the restriction $\Phi : \mathrm{Auk}^{\mathrm{r}}(F) \to \mathrm{Auk}^{\mathrm{s}}(F)$ is surjective, and in general also the restriction is not injective.

11.9.3. Finding (semi-)maximal autarkies

Given any nontrivial autarky, we can efficiently find some (contained) minimal autarky by the observation made in [KMT08], that "within" a partial assignment autarky search is easy:

Definition 11.9.2. For any submonoid M resp. subsemigroup S of $\mathcal{PASS}$ and for a partial assignment $\varphi \in \mathcal{PASS}$ we denote by $M(\varphi) := M \cap \mathcal{PASS}(\varphi)$ resp. $S(\varphi) := S \cap \mathcal{PASS}(\varphi)$ the submonoid resp. subsemigroup of M resp. S obtained by restriction to partial assignments contained in φ.

Lemma 11.9.1. *Consider a clause-set $F \in \mathcal{CLS}$ and a partial assignment $\varphi \in \mathcal{PASS}$. Then $\mathrm{Auk}(F)(\varphi)$ has a (unique) largest element φ_0, which can be efficiently computed as follows:*

1. *Let $\varphi_0 := \varphi$.*
2. *If φ_0 is an autarky for F, then stop. Otherwise, choose $C \in F$ with $\mathrm{var}(\varphi) \cap \mathrm{var}(C) \neq \emptyset$ and $\varphi * \{C\} \neq \top$, choose $v \in \mathrm{var}(\varphi) \cap \mathrm{var}(C)$, let φ_0 be the restriction of φ_0 to $\mathrm{var}(\varphi_0) \setminus \{v\}$, and repeat this step.*

For a given clause-set F with $V := \mathrm{var}(F)$, the map $\varphi \in \{\varphi \in \mathcal{PASS} : \mathrm{var}(\varphi) \subseteq V\} \mapsto \varphi_0 \in \mathrm{Auk}^{\mathrm{r}}(F)$, which assigns to every partial assignment the contained largest autarky, is obviously surjective. Such autarkies φ_0 which are largest elements of some $\mathrm{Auk}(F)(\varphi)$ for total assignments φ (i.e., $\mathrm{var}(\varphi) = \mathrm{var}(F)$) are called *semi-maximal autarkies*. In Subsection 11.10.1 some applications are discussed.

Corollary 11.9.2. *Given a nontrivial autarky φ for a clause-set F, we can efficiently compute some minimal autarky $\varphi_0 \in \mathrm{Auk}{\downarrow}(F)$ (with $\varphi_0 \subseteq \varphi$).*

Proof. Try removing assignments from φ and obtaining a smaller nontrivial autarky by Lemma 11.9.1 until a minimal autarky is obtained. □

11.9.4. Generating autarkies

Now we are considering generating sets of the autarky monoid, i.e., subsets E of $\mathrm{Auk}^{\mathrm{r}}(F)$ such that the generated submonoid is $\mathrm{Auk}^{\mathrm{r}}(F)$ itself. The autarky monoid has the potential to give a succinct representation of a large set of satisfying assignments: Though there are even more autarkies than satisfying assignments, a generating set might nevertheless be much smaller.

Example 11.9.1. For $n \geq 0$ we consider the (well-known) satisfiable clause-set $F := \{\{v_1, w_1\}, \ldots, \{v_n, w_n\}\}$ with $2n$ variables (well-known for not having a short DNF-representation). The structure of $\mathrm{Auk}^{\mathrm{r}}(F)$ is as follows:

- There are $2n$ minimal autarkies $\langle v_i \to 1\rangle, \langle w_i \to 1\rangle$.
- $|\mathrm{mod}(F)| = 3^n$, where the satisfying (total) assignments, which are also the maximal autarkies here, are all combinations of the three satisfying assignments for the clauses $\{v_i, w_i\}$.
- $|\mathrm{Auk}^{\mathrm{r}}(F)| = 6^n$, where the autarkies are all combinations of the six autarkies for the clauses $\{v_i, w_i\}$.
- Using $E_i := \{\langle v_i \to 1\rangle, \langle w_i \to 1\rangle, \langle v_i \to 1, w_i \to 0\rangle, \langle w_i \to 1, v_i \to 0\rangle\}$, the set $E := \bigcup_{i=1}^n E_i$ is a generating set for $\mathrm{Auk}^{\mathrm{r}}(F)$ of size $4n$.

So a basic question is to obtain interesting notions of generating sets for autarky monoids. Only at first glance it might appear that the minimal autarkies are sufficient to compute at least the learn kernel, but the following example shows that this is clearly not the case.

Example 11.9.2. Minimally unsatisfiable Horn clause-sets have deficiency 1; moreover, the Horn clause-sets with $n \geq 0$ variables, which are even saturated minimally unsatisfiable, are up to the names of the variables exactly the clause-sets $H_n := H'_n \cup \{\{\overline{v_1}, \ldots, \overline{v_n}\}\}$, where

$$H'_n := \{\{v_1\}, \{\overline{v_1}, v_2\}, \{\overline{v_1}, \overline{v_2}, v_3\}, \ldots, \{\overline{v_1}, \ldots, \overline{v_{n-1}}, v_n\}\}.$$

For example $H_0 = \{\bot\}$, $H_1 = \{\{v_1\}, \{\overline{v_1}\}\}$ and $H_2 = \{\{v_1\}, \{\overline{v_1}, v_2\}, \{\overline{v_1}, \overline{v_2}\}\}$. It is not hard to see that

$$\mathrm{Auk}^{\mathrm{r}}(H'_n) = \{\langle\rangle, \langle v_n \to 1\rangle, \langle v_{n-1}, v_n \to 1\rangle, \ldots, \langle v_1, \ldots, v_n \to 1\rangle\}.$$

So $\mathrm{Auk}^{\mathrm{r}}(H'_n)$ as a partial order is a linear chain with $n+1$ elements, with $\langle\rangle$ as smallest element and with $\varphi_n := \langle v_1, \ldots, v_n \to 1\rangle$ as largest element; for example $\mathrm{Auk}^{\mathrm{r}}(H'_2) = \{\langle\rangle, \langle v_2 \to 1\rangle, \langle v_1, v_2 \to 1\rangle\}$. For $n \geq 1$ the only minimal autarky is $\langle v_n \to 1\rangle$.

So what can we say about a generating set G of the (restricted) autarky semigroup $\mathrm{Auks}^{\mathrm{r}}(F)$? Clearly every minimal autarky for F must be element of G, and, more generally, every *directly indecomposable* autarky $\varphi \in \mathrm{Auks}^{\mathrm{r}}(F)$ must be in G, which are characterised by the condition that whenever $\varphi = \psi_1 \circ \psi_2$ for $\psi_1, \psi_2 \in \mathrm{Auk}^{\mathrm{r}}(F)$ holds, then $\psi_1 = \varphi$ or $\psi_2 = \varphi$ must be the case. The directly indecomposable autarkies for Example 11.9.1 are exactly the elements of the generating set E given there, but the following example shows that the directly indecomposable autarkies in general do not generate all autarkies.

Example 11.9.3. Let $F := \{\{v_1, v_2, v_3\}, \{v_1, \overline{v_2}, \overline{v_3}\}\}$. For a partial assignment φ and a set of variables V with $V \cap \mathrm{var}(\varphi) = \emptyset$ let us denote by $[\varphi]_V$ the set of all partial assignments ψ with $\varphi \subseteq \psi$ and $\mathrm{var}(\psi) \subseteq \mathrm{var}(\varphi) \cup V$. Using $\varphi_1 := \langle v_1 \to 1\rangle$, $\varphi_2 := \langle v_2 \to 1, v_3 \to 0\rangle$ and $\varphi_3 := \langle v_2 \to 0, v_3 \to 1\rangle$ we have $\mathrm{Auks}^{\mathrm{r}}(F) = [\varphi_1]_{\{v_2,v_3\}} \cup [\varphi_2]_{\{v_3\}} \cup [\varphi_3]_{\{v_3\}}$, whence $|\mathrm{Auks}^{\mathrm{r}}(F)| = 3^2 + 3 + 3 - 1 - 1 = 13$. There are three minimal autarkies, namely $\varphi_1, \varphi_2, \varphi_3$, and four further directly

indecomposable autarkies, namely $\langle v_1 \to 1, v_2 \to \varepsilon\rangle$ and $\langle v_1 \to 1, v_3 \to \varepsilon\rangle$ for $\varepsilon \in \{0,1\}$. These seven elements generate eleven of the thirteen (non-trivial) autarkies, leaving out $\varphi_2 \circ \langle v_1 \to 0\rangle$ and $\varphi_3 \circ \langle v_1 \to 0\rangle$.

Given a set $E \subseteq \mathrm{Auk}^{\mathrm{r}}(F)$ of autarkies, instead of allowing all compositions of elements of E it makes sense to only allow "commutative compositions", that is we only consider subsets $E' \subseteq E$ of pairwise commuting (pairwise compatible) partial assignments and their composition $\circ_{\varphi\in E'}\varphi$; note that due to the imposed commutativity here we do not need to take care of the order, and due to idempotency also no exponents need to be considered. We call E a *commutatively generating set* for $\mathrm{Auks}^{\mathrm{r}}(F)$ if the set of commutative compositions of E is $\mathrm{Auks}^{\mathrm{r}}(F)$, while we speak of a *commutatively semi-generating set*, if the commutative compositions of E at least yield all semi-maximal autarkies (different from $\langle\rangle$; recall Subsection 11.9.3).[5] The generating set E in Example 11.9.1 is also a commutatively generating set; Example 11.9.1 is just a combination of variable-disjoint clauses, and we conclude this subsection by discussing the autarky-monoid and its generation for singleton clause-sets (the smallest non-trivial examples).

Example 11.9.4. Consider a clause C, and let $F := \{C\}$ and $n := n(F) = |C|$.

- There are n minimal autarkies (i.e., $|\mathrm{Auk}{\downarrow}(F)| = n$).
- $|\mathrm{Auk}^{\mathrm{r}}(F)| = 3^n - (2^n - 1)$ (the non-autarkies are all partial assignments using only value 0, except of the empty partial assignment).
- For $n \geq 1$ there are $2^n - 1$ maximal autarkies (i.e., $|\mathrm{Auk}{\uparrow}(F)| = 2^n - 1$), which are also the satisfying assignments, while if $n = 0$, then F is unsatisfiable and lean.
- Every semi-maximal autarky is also a maximal autarky.
- There are $n + n(n-1)$ directly indecomposable autarkies, namely the minimal autarkies plus the partial assignments $\langle x \to 1, y \to 0\rangle$ for $x \in C$ and $y \in C \setminus \{x\}$.
- The set I of directly indecomposable autarkies is a (smallest) generating set, and it is also a commutatively generating set.
- If we replace the n minimal autarkies in I by the single autarky $\langle x \to 1 : x \in C\rangle$, obtaining I', then I' is a commutatively semi-generating set (of size $n(n-1)+1$).

11.9.5. The autarky monoid as a functor

In [Sze03] the basic form of categories of clause-sets has been introduced (recall Section 11.3), called $\mathfrak{CLS}$ here, and which is defined as follows:

1. The set of objects of $\mathfrak{CLS}$ is $\mathcal{CLS}$.
2. For a clause-set F let $\mathrm{lit}(F) := \mathrm{var}(F) \cup \overline{\mathrm{var}(F)}$ be the set of literals over F (which occur in at least one polarity; for a set L of literals we use $\overline{L} := \{\overline{x} : x \in L\}$).
3. Now a morphism $f : F \to G$ from clause-set F to clause-set G is a map $f : \mathrm{lit}(F) \to \mathrm{lit}(G)$ with the following two properties:

[5] We could also speak of "compatibly generating". Using "sign vectors" instead of partial assignments, according to Definition 5.31 in [BK92] we could also speak of "conformally generating".

(a) f preserves complements, i.e., for $x \in \mathrm{lit}(F)$ we have $f(\overline{x}) = \overline{f(x)}$.
(b) f preserves clauses, i.e., for $C \in F$ we have $f(C) \in G$, where $f(C) := \{f(x) : x \in C\}$.

The isomorphisms in $\mathfrak{CLS}$ are the standard isomorphisms between clause-sets, allowing the renaming of variables accompanied with flipping polarities. Extending [Sze03] (Section 3) we can now recognise $F \in \mathcal{CLS} \mapsto \mathrm{Auk}(F)$ as a contravariant functor from $\mathfrak{CLS}$ to the category $\mathfrak{MON}$ of monoids (objects are monoids, morphisms are homomorphisms of monoids):

Lemma 11.9.3. *The formation of the autarky monoid is a contravariant functor* $\mathrm{Auk} : \mathfrak{CLS} \to \mathfrak{MON}$*, where for a morphism* $f : F \to G$ *between clause-sets the homomorphism* $\mathrm{Auk}(f) : \mathrm{Auk}(G) \to \mathrm{Auk}(F)$ *assigns to the autarky* $\varphi \in \mathrm{Auk}(G)$ *the partial assignment* $\mathrm{Auk}(f)(\varphi)$ *given as follows:*

1. *the domain of* $\mathrm{Auk}(f)(\varphi)$ *is the set of variables* $v \in \mathrm{var}(F)$ *such that* $\mathrm{var}(f(v)) \in \mathrm{var}(\varphi)$*;*
2. *for* $v \in \mathrm{var}(\mathrm{Auk}(f)(\varphi))$ *we set* $\mathrm{Auk}(f)(\varphi)(v) := \varphi(f(v))$.

Proof. First for $\varphi \in \mathrm{Auk}(G)$ and $\psi := \mathrm{Auk}(f)(\varphi)$ we need to show $\psi \in \mathrm{Auk}(F)$. So consider a clause $C \in F$ touched by ψ. Thus there is $x \in C$ such that $\varphi(f(x))$ is defined. Now φ touches $f(C)$, so there is $x' \in C$ with $\varphi(f(x')) = 1$, i.e., $\psi(x') = 1$. To see that Auk preserves identities we just note that $\mathrm{id}_F = \mathrm{id}_{\mathrm{lit}(F)}$, and then $\mathrm{Auk}(\mathrm{id}_F) = \mathrm{id}_{\mathrm{Auk}(F)}$. Finally consider morphisms $f : F \to G$ and $g : G \to H$ in $\mathfrak{CLS}$; we have to show $\mathrm{Auk}(g \circ f) = \mathrm{Auk}(f) \circ \mathrm{Auk}(g)$, and this follows directly from the definitions. □

Also $\mathrm{Auk}^{\mathrm{r}}$ yields a functor. We remark that the coproduct in $\mathfrak{CLS}$ is the variable-disjoin sum, and the functor $\mathrm{Auk}^{\mathrm{r}}$ anti-preserves coproducts, i.e., maps the variable-disjoint sum of clause-sets to the product of their autarky-monoids.

11.9.6. Galois correspondences

Finally we consider the Galois correspondence given by the autarky notion. Consider the set $\mathcal{CL}$ of clauses and the set $\mathcal{PASS}$ of partial assignments. We have two fundamental relations between clauses and partial assignments:

1. S_0 is the set of pairs $(C, \varphi) \in \mathcal{CL} \times \mathcal{PASS}$ such that φ is a satisfying assignment for $\{C\}$, that is, $\varphi * \{C\} = \top$.
2. S_1 is the set of pairs $(C, \varphi) \in \mathcal{CL} \times \mathcal{PASS}$ such that φ is an autarky for $\{C\}$, that is, $\varphi * \{C\} \in \{\top, \{C\}\}$.

If we use the alternative representation $\mathcal{PASS}' := \mathcal{CL}$ of partial assignments by clauses containing the literals satisfied by the partial assignment, i.e., a partial assignment φ corresponds to the clause C^1_φ, then we obtain the corresponding relations $S'_0 = \{(C, D) \in \mathcal{CL} \times \mathcal{PASS}' : C \cap D \neq \emptyset\}$ and $S'_1 = \{(C, D) \in \mathcal{CL} \times \mathcal{PASS}' : C \cap \overline{D} \neq \emptyset \Rightarrow C \cap D \neq \emptyset\}$.

Given the "polarities" S_0, S_1, we obtain (as usual) Galois correspondences (X_0, Y_0) resp. (X_1, Y_1) between the partial orders $(\mathbb{P}(\mathcal{CL}), \subseteq)$ and $(\mathbb{P}(\mathcal{PASS}), \subseteq)$,

given as $X_i : \mathbb{P}(\mathcal{CL}) \to \mathbb{P}(\mathcal{PASS})$ and $Y_i : \mathbb{P}(\mathcal{PASS}) \to \mathbb{P}(\mathcal{CL})$ defined by

$$X_i(F) := \{\varphi \in \mathcal{PASS} \mid \forall\, C \in F : S_i(C, \varphi)\}$$
$$Y_i(P) := \{C \in \mathcal{CL} \mid \forall\, \varphi \in P : S_i(C, \varphi)\}.$$

for $F \subseteq \mathcal{CL}$ and $P \subseteq \mathcal{PASS}$. Using $\mathrm{mod}^*(F) := \{\varphi \in \mathcal{PASS} : \varphi * F = \top\}$ for the set of all satisfying partial assignments for F, we see that

$$X_0(F) = \mathrm{mod}^*(F), \quad X_1(F) = \mathrm{Auk}(F)$$

for sets of clauses F.[6] As it is well known, $Z_i := Y_i \circ X_i$ for $i \in \{0, 1\}$ yields a closure operator on $\mathbb{P}(\mathcal{CL})$ (i.e., $Z_i(F) \supseteq F$, $Z_i(Z_i(F)) = Z_i(F)$ and $F \subseteq G \Rightarrow Z_i(F) \subseteq Z_i(G)$). It is a well-known generalisation of the fundamental duality-law for the formation of the transversal hypergraph, that $Z_0(F)$ is just the set of all clauses C which follow logically from F, i.e., $Z_0(F) = \{C \in \mathcal{CL} : F \models C\}$. Using $\mathcal{PASS}'$ instead of $\mathcal{PASS}$, we have $Y_i = X_i$ for $i \in \{0, 1\}$, and thus the semantical closure $Z_0(F)$ can be understood as $Z_0(F) = \mathrm{mod}^*(\mathrm{mod}^*(F)) =: \mathrm{mod}^{**}(F)$. And analogously we can understand the *autarky closure* $Z_1(F)$, the largest set of clauses with the same autarky monoid as F, as $Z_1(F) = \mathrm{Auk}(\mathrm{Auk}(F))$; thus we write the autarky closure as $\mathrm{Auk}^2(F) := Z_1(F)$. Some simple properties of the autarky closure are (for all $F \subseteq \mathcal{CL}$):

1. $\mathrm{Auk}^2(\top) = \top$ (also $\mathrm{mod}^{**}(\top) = \top$).
2. $\bot \in \mathrm{Auk}^2(F)$, and thus $\mathrm{Auk}^2(F) \supseteq F \cup \{\bot\}$ (while $\bot \in \mathrm{mod}^{**}(F)$ iff F is unsatisfiable).
3. We have $\mathrm{var}(\mathrm{Auk}^2(F)) = \mathrm{var}(F)$. This is quite different from mod^{**}, where for $F \neq \top$ we have $\mathrm{var}(\mathrm{mod}^{**}(F)) = \mathcal{VA}$, since $\mathrm{mod}^{**}(F)$ is stable under addition of super-clauses.
4. A clause-set F is lean iff $\mathrm{Auk}^2(F) = \{C \in \mathcal{CL} : \mathrm{var}(C) \subseteq \mathrm{var}(F)\}$, that is, iff $\mathrm{Auk}^2(F)$ contains all possible clauses over $\mathrm{var}(F)$ (in other words, iff $\mathrm{Auk}^2(F)$ is as large as possible).
5. $\mathrm{Auk}^2(F)$ is stable under resolution, that is, if $C, D \in F$ with $C \cap \overline{D} = \{x\}$, then $R \in \mathrm{Auk}^2(F)$ for the resolvent $R := (C \setminus \{x\}) \cup (D \setminus \{\overline{x}\})$.
6. $\mathrm{Auk}^2(F)$ is stable under the composition of partial assignments considered as clauses, that is, if $C, D \in \mathrm{Auk}^2(F)$ then also $(C \setminus \overline{D}) \cup D \in \mathrm{Auk}^2(F)$.

11.10. Finding and using autarkies

In this section we discuss the methods which have been applied to find autarkies, and what use has been made of those autarkies. First some remarks on the complexity of the most basic algorithmic problems. The class of lean clause-sets has been shown in [Kul03] to be coNP-complete, and thus finding a non-trivial autarky is NP-complete. By Corollary 11.9.2 also finding a minimal autarky is NP-complete, while by contrast deciding whether for a given clause-set F and partial assignment φ it is φ a maximal autarky for F is coNP-complete (if φ is an autarky for F, then φ is a maximal autarky iff $F \setminus (\varphi * F)$ is lean). As shown in [KMT08], the "autarky existence problem" is self-reducible, that is, given an

[6] This includes, of course, clause-sets, which we defined as *finite* sets of clauses.

oracle for deciding whether a clause-set is lean, one can efficiently find a maximal autarky. Lemma 8.6 in [Kul03] shows how (just) to compute the lean kernel given such an oracle. In [KMT08] one finds complexity characterisations of other basic tasks related to autarkies. In Subsection 11.11.6 we will discuss a refined list of the basic algorithmic problems of autarky theory.

11.10.1. Direct approaches

At first sight it might appear that time $\tilde{\mathrm{O}}(3^{n(F)})$ is needed to exhaustively search for a non-trivial autarky in a clause-set F, since there are $3^{n(F)}$ partial assignments. However, we can find all semi-maximal autarkies (recall Subsection 11.9.3) in time $\tilde{\mathrm{O}}(2^{n(F)})$ by running through all total assignments φ and determining the maximal autarky $\varphi_0 \subseteq \varphi$ for F (this was first observed in [KMT08]). Thus the lean kernel can be computed in time $\tilde{\mathrm{O}}(2^{n(F)})$ (for arbitrary clause-sets, using linear space). See Subsection 11.10.8 for applications of this method in local search solvers (while in Corollary 11.10.2 the bound $\tilde{\mathrm{O}}(2^{n(F)})$ is improved for restricted clause-length).

The direct backtracking approach searching for an autarky is closely related to the tableau calculus, when focusing on a specific clause to be satisfied by some autarky, and we further comment on this subject in Subsection 11.10.2. From a somewhat more general perspective, searching for an autarky can be considered as a constraint satisfaction problems where variables have 3-valued domains; yet there are no experiences with this approach, but in Subsection 11.10.4 we discuss the reduction of the autarky search problem to SAT.

11.10.2. Autarkies and the tableau-calculus

A (standard) tableau-based SAT solver starts with a "top" clause $C \in F$, chooses a literal $x \in C$, and tries recursively to refute that $\langle x \to 1\rangle$ could be extended to a satisfying assignment for F (this is then repeated for all literals in C, with the goal to conclude unsatisfiability of F at the end). It has been observed in [Van99] that this process should better be considered as the natural form of searching for an autarky for F satisfying C, since in case the solver gets stuck, that is, cannot finish the unsatisfiability-proof based on x (and thus cannot proceed to the other literals in C), we actually found an autarky $\langle x \to 1\rangle \subseteq \varphi$ for F, so that we can repeat the whole process with $\varphi * F \subset F$, and either finally find F unsatisfiable (that is, find some subset $F' \subseteq F$ unsatisfiable with $\mathrm{N_a}(F) \subseteq F'$), or can compose the various autarkies φ found in the repetitions of this process to a (global) satisfying assignment. This view of the (standard, propositional) tableau calculus yields an explanation why SAT solvers are usually faster: Exhaustively searching for an autarky is a somewhat harder task than just (exhaustively) searching for a satisfying assignment (especially to determine that there is none), and so for the clause-based branching performed by a tableau-solver the top clause C is split into $|C|$ branches $\langle x \to 1\rangle$ for $x \in C$, while a SAT solver can use some order $C = \{x_1, \ldots, x_k\}$ and set in branch i additionally to $x_i \to 1$ also $x_j \to 0$ for $j < i$ (where for the autarky search the x_j stay unassigned). Roughly spoken, SAT only needs to search through 2^n (total) assignments, while tableau search somehow needs to search through 3^n partial assignments. However for instances

having "nice" autarkies (not too small, not too big), the autarky approach should have merits. The approach from [Van99] has been extended by parallel processing in [Oku00], exploiting that the composition of autarkies again is an autarky (and so we can combine autarkies found in parallel), while applications are discussed in [VO99].

11.10.3. Computing the lean kernel by the autarky-resolution duality

Based on the duality between autarkies and resolution (recall Theorem 11.8.1), the following meta-theorem has been given (in variations) in [Kul01, Kul07a] for computing the lean kernel:

Theorem 11.10.1. *Consider a class $\mathcal{C} \subseteq \mathcal{CLS}$ of clause-sets, an "upper bound" function $f : \mathcal{C} \to \mathbb{R}_{\geq 0}$ and an algorithm $\mathfrak{A}$ defined on inputs $F \in \mathcal{C}$ with the following properties:*

1. *$\mathcal{C}$ is stable under restriction, that is, for all (finite) $V \subseteq \mathcal{VA}$ and all $F \in \mathcal{C}$ we have $F[V] \in \mathcal{C}$.*
2. *The running time of $\mathfrak{A}$ on input $F \in \mathcal{C}$ is $\tilde{\mathrm{O}}(f(F))$.*
3. *f is monotone w.r.t. restriction, that is, for $V \subseteq \mathcal{VA}$ and $F \in \mathcal{C}$ we have $f(F[V]) \leq f(F)$.*
4. *$\mathfrak{A}$ for input F either returns a satisfying partial assignment φ_F for F or a non-empty set of variables $\emptyset \neq V_F$ used in some resolution refutation F.*

Then for input $F_0 \in \mathcal{C}$ we can compute the lean kernel as well as an autarky realising the lean kernel (a "quasi-maximal autarky") in time $\tilde{\mathrm{O}}(f(F_0))$ as follows:

- *Let $F := F_0$.*
- *Run $\mathfrak{A}$ on F. If the output is a satisfying assignment φ, then return (F, φ), while if the output is a set V of variables then let $F := F[V]$ and repeat.*

For the pair (F, φ) computed we have $F = \mathrm{N_a}(F_0)$, $\varphi \in \mathrm{Auk}(F_0)$ and $\varphi * F_0 = F$.

Given a quasi-maximal autarky, we can determine the variables which are in the lean kernel but not in the largest autark subset, and then by arbitrary extension we can also compute a maximal autarky. It is not too complicated to provide a backtracking SAT solver (look-ahead or conflict-driven) without much space- or time-overhead with the ability to output in case of unsatisfiability the set of variables involved in the resolution refutation found; note that we do not need the resolution refutation itself, which can require exponential space, but only the set of variables involved, computed recursively bottom-up from the leaves, cutting off branches if the branching variable is not used as resolution variable.[7] For example, we can use $\mathcal{C} = \mathcal{CLS}$, $f = b^{n(F)}$ where b depends on the maximal clause-length of F, and for $\mathfrak{A}$ the classical CNF algorithm from [MS85, Luc84] (branching on a shortest clauses, and using the autarky found when no clause is shortened; see Subsection 11.10.5), and we obtain:

Corollary 11.10.2. *For a clause-set $F \in \mathcal{CLS}$ let $\mathrm{rk}(F) \in \mathbb{N}_0$ denote the maximal clause-length. Then a maximal autarky for input $F \in \mathcal{CLS}$ can be computed*

[7] This realises "intelligent backtracking" also for conflict-driven solvers, and is implemented in the `OKsolver-2002`.

in time $\tilde{\mathrm{O}}(\tau^{n(F)}_{\mathrm{rk}(F)-1})$*, where* $\tau_{\mathrm{rk}(F)-1} = \tau(1,\ldots,\mathrm{rk}(F)-1)$ *is the solution* $x > 1$ *of* $\sum_{i=1}^{\mathrm{rk}(F)-1} x^{-i} = 1$ *(see Subsection 7.3.2 in Chapter 7 on branching heuristics). For inputs restricted to 3-CNF we have* $\mathrm{rk}(F) \leq 3$*, where* $\tau(1,2) = 1.618\ldots$*, that is, a maximal autarky for inputs* F *in 3-CNF can be computed in time* $O(1.619^{n(F)})$*.*

It is not known to what extend the improved k-CNF bounds can be applied: Closest is [Kul99b], but as shown in [Kul99c], the use of blocked clauses cannot be simulated by resolution, so the meta-theorem is not applicable (at least prima facie). All other algorithms are based on some form of probabilistic reasoning, and thus in case of unsatisfiability do not yield (in some form) resolution refutations, so also here the meta-theorem (Theorem 11.10.1) is not applicable.

An application of the computation of the lean kernel to the problem of finding "unsatisfiable cores" in clause-sets is discussed in [KLMS06] (concentrating especially on the different roles of clauses, like "necessary", "potentially necessary", "usable" or "unusable" clauses).

11.10.4. Reduction to SAT

A translation of the autarky search problem to SAT has been introduced in [LS08], considering applications similar to those mentioned in [KLMS06]; the basic method used for the translation is the use of "clause indicator variables" for clauses satisfied by the autarky, and using also "variable indicator variables" to indicate whether an original variable is used by the autarky. The experimental results suggest that this works complementary to the method used in [KLMS06] (see the previous Subsection 11.10.3): Computation of the lean kernel via the autarky-resolution duality, based on a conflict-driven solver here, tends to use many iterations, while the algorithm of [LS08] searches directly for the largest autark subset (to express this, the clause-indicator variables are used, together with "AtMost"-constraints).

There are many variations possible for such translations, but the basic translation, encoding autarkies for clause-set F exactly as satisfying assignments of $A(F)$, is simple: For every variable $v \in \mathrm{var}(F)$ introduce three variables $a_{v,\varepsilon}$ for $\varepsilon \in \{0,1,*\}$, where "$*$" stands for "unassigned", and $a_{v,\varepsilon}$ means that v shall get "value" ε; so we have $n(A(F)) = 3n(F)$. A literal x over $\mathrm{var}(F)$ is interpreted by the positive literal $a(x)$ over $\mathrm{var}(A(F))$, where for positive literals v we set $a(v) := a_{v,1}$, while for negative literals $\overline{v}$ we set $a(\overline{v}) := a_{v,0}$. Now a clause $C \in F$ yields $|C|$ many clauses of length $|C|$, namely for each $x \in C$ the "autarky clause" $C_x := \{\overline{a(\overline{x})}\} \cup \{a(y) : y \in C \setminus \{x\}\}$ (which requires that if one literal x is set to false, then some other literal y must be set to true). Then $A(F)$ consists of the clauses C_x for $C \in F$ and $x \in C$ together with the ALO-clauses, that is, for every variable $v \in \mathrm{var}(F)$ the (positive, ternary) clause $\{a_{v,0}, a_{v,1}, a_{v,*}\}$, and together with the AMO-clauses, that is for each $v \in \mathrm{var}(F)$ and $\varepsilon, \varepsilon' \in \{0,1,*\}$, $\varepsilon \neq \varepsilon'$ the (negative, binary) clause $\{\overline{a_{v,\varepsilon}}, \overline{a_{v,\varepsilon'}}\}$. Obviously we have a natural bijection from $\mathrm{mod}(A(F))$ to $\mathrm{Auk}^{\mathrm{r}}(F)$ (note that every satisfying (partial) assignment for $A(F)$ must actually be total). A problem now is that to express non-triviality of the autarky, we need the long clause $\{\overline{a_{v,*}} : v \in \mathrm{var}(F)\}$ (containing *all* variables); to

avoid this clause of maximal length, and to obtain more control over the autarky obtained, [LS08] uses the above mentioned indicator variables.

11.10.5. Autarkies for worst-case upper bounds

Apparently the first application of "autarkies" (without the name, which was invented 10 years later) can be found in [EIS76], where they are used to show that 2-SAT is decidable in polynomial time. This very simple method works as follows for a clause-set F with maximal clause-length $\mathrm{rk}(F) \leq 2$: Consider any literal x in F, and let the partial assignment φ_x be obtained from $\langle x \to 1\rangle$ by adding all inferred unit-clause propagations for F. If $\bot \in \varphi_x * F$, then F is satisfiability-equivalent to $\langle x \to 0\rangle * F$ and we eliminated one variable. Otherwise we know that all clauses in $\varphi_x * F$ must have length at least 2, and thus, due to $\mathrm{rk}(F) \leq 2$, φ_x is an autarky for F — so in this case F is satisfiability equivalent to $\varphi_x * F$, and again at least one variable has been eliminated.

If F has unrestricted clause-length, then in case of $\bot \notin \varphi_x * F$ and where furthermore φ_x is not an autarky for F (which can be easily checked), we know that F must contain a clause C of length $2 \leq |C| \leq \mathrm{rk}(F)-1$ (note that this generalises the above argument). Applying this argument to both branches $\varphi_x, \varphi_{\overline{x}}$ has been exploited in [MS85] for improved k-SAT upper bounds (and in this article also the notion of an "autark assignment" has been introduced). Basically the same argument, but in a different form, has been used in [Luc84]; see [KL97, KL98] for discussions (the point is while [MS85] checks whether $\varphi_x, \varphi_{\overline{x}}$ are autarkies for the current "residual" clause-set (at each node of the search tree), [Luc84] only checks the special cases which are actually needed to prove the bound).

These ideas have been systematically extended in [Kul99b], on the one hand generalising autarkies to "Br-autarkies", allowing for the application of an arbitrary reduction and the possible elimination of blocked clauses afterwards. And on the other hand, the notion of a *weak k-autarky* is introduced (where "weak" is added now according to the general distinction between "weak autarkies" and "autarkies") for $k \geq 0$, which is a partial assignment φ such that $|(\varphi * F) \setminus F| \leq k$; note that weak 0-autarkies are exactly the weak autarkies. For $k \geq 1$ in general $\varphi * F$ is no longer satisfiability-equivalent to F, however the point is that, using $N := (\varphi * F) \setminus F$ for the set of new clauses created by φ, for every partial assignment ψ with $\mathrm{var}(\psi) \subseteq \mathrm{var}(N)$ (thus $\mathrm{var}(\psi) \cap \mathrm{var}(\varphi) = \emptyset$) and $\psi * N = \top$ (i.e., ψ satisfies N) the partial assignment φ is a weak autarky for $\psi * F$.[8] In [Kul99b] this k-autarky principle has been applied by considering a complete branching $(\psi_1, \ldots, \psi_m)$ for N (i.e., covering all total assignments), where then those branches ψ_i satisfying N can be augmented by φ. In Subsection 11.13.2 we discuss another possibility to exploit this situation, by adding "look-ahead autarky clauses".

Finally we mention the "black and white literals principle" ([Hir00]), which generalises the "generalised sign principle" from [KL97], and which reformulates the "autarky principle" (that applying an autarky yields a satisfiability-equivalent

[8] The proof is: $(\varphi * (\psi * F)) \setminus (\psi * F) = ((\varphi \circ \psi) * F) \setminus (\psi * F) = ((\psi \circ \varphi) * F) \setminus (\psi * F) = (\psi * (\varphi * F)) \setminus (\psi * F) \subseteq (\psi * (N \cup F)) \setminus (\psi * F) = ((\psi * N) \cup (\psi * F)) \setminus (\psi * F) = \top$.

clause-set) in such a way that it becomes easily applicable as a source for reductions in worst-case upper-bound proofs (see [KL98] for a discussion).

11.10.6. Incomplete autarky search in complete SAT solvers

The basic idea from [MS85], as explained in Subsection 11.10.5, is to check the assignments φ_x (obtained by adding all inferred unit-clause propagation to a test-assignment $x \to 1$) for being an autarky (in which case there is no need to branch). This has been implemented in the `OKsolver`-2002 (see [Kul98, Kul02])[9], incorporating the ideas from [Kul99b]:

- Now arbitrary binary splitting are considered (and not clause-branching as in [MS85]).
- The assignments φ_x occur naturally when performing "failed-literal reduction".
- The branching heuristics is based on counting new clauses, and the autarky-test is actually necessary to avoid branches without "progress" (i.e., with zero new clauses).

See Subsection 7.7.4.1 in Chapter 7 on the general theory of branching heuristics for more information. This type of "ad-hoc" and "enforced" autarky search has also been applied (with variations) in the `march` solvers.

11.10.7. Employing autarkies in generalised unit-clause-propagation

Unit-clause propagation can be strengthened to failed-literal reduction, and further to arbitrary levels (corresponding to levels of generalised Horn formulas); see [Kul99a] for a thorough treatment and an overview on the literature. There, through the use of oracles, special poly-time procedures for finding autarkies (e.g., finding "linear autarkies"; see Subsection 11.10.9) are employed, enhancing the hierarchies by adding more satisfiable instances to each level.

This approach only uses autarkies at the leaves (to ensure confluence); in a more ad-hoc manner in [DE92] the assignments found when searching for enforced assignments at the various levels of the hierarchy (generalising the approach from Subsection 11.10.6 in a certain way) are directly checked for the autarky property.

11.10.8. Local search

Local search solvers search through total assignments, and each such total assignment can be inspected for being an autarky, according to Lemma 11.9.1. A first implementation one finds in `UnitMarch` (enhancing the local search solver UnitWalk; see [Hv08]). Combinations with other local search solvers might become interesting in the future (especially when adapting the heuristics to the new goal of finding autarkies). Direct local search through partial assignments should also be interesting, but has not been tried yet.

[9] this "historic" solver is now part of the `OKlibrary` (`http://www.ok-sat-library.org`)

11.10.9. Polynomial-time cases

Most classes of clause-sets, where basic tasks like autarky finding or computation of the lean kernel can be performed in polynomial time, are handled by special types of autarkies, which is the subject of the following Section 11.11. Here we only comment on the most prominent cases, and especially how they can be handled by the techniques discussed in the current section (considering general autarkies); see [KMT08] for a more elaborated study.

We already mentioned in Subsection 11.10.5 that the consideration of clause-sets F in 2-CNF (every clause contains at most two literals) was at the beginning of "autarky theory". And it is easy to see that actually if F is not lean then there must be some literal x such that φ_x (which was the extension of $\langle x \to 1\rangle$ by unit-clause propagations) is an autarky (see [KMT08] for more details). So the autarky-existence problem is in P for 2-CNF, and (since 2-CNF is stable under the application of partial assignments) also the lean kernel is computable in polynomial time. This method for finding autarkies is covered by the method discussed in Subsection 11.10.6 for finding "ad-hoc enforced autarkies", and thus is actually implemented in look-ahead solvers like the `OKsolver`-2002 or the `march`-solvers (see Subsection 11.10.6). Another way to handle 2-CNF is by Theorem 11.10.1, where for finding a resolution-refutation one could also apply the stability of 2-CNF under the resolution rule (though finding a resolution refutation by a backtracking solver is more efficient). Finally every autarky for F in 2-CNF is a special case of a "linear autarky", as discussed in Subsection 11.11.3, and thus can be handled also by the special methods there.

In a certain sense "dual" to 2-CNF is the class of clause-sets F such that every variable occurs at most once positively and at most once negatively. Minimally unsatisfiable such F are exactly the marginal minimally unsatisfiable clause-sets of deficiency 1. It is well known (and trivial) that by DP-resolution (un)satisfiability for this class is decidable in polynomial time, so we can find easily a resolution refutation, while by using self-reduction we can also find a satisfying assignment; so again by Theorem 11.10.1 we obtain that the computation of the lean kernel (and the computation of a maximal autarky) can be done in polynomial time. Another approach is given by the observation that every autarky for such F must be a "matching autarky", as discussed in Subsection 11.11.2, and thus we can use the (poly-time) methods from there. Finally, minimal autarkies of F correspond to the cycles in the conflict multigraph of F (compare [Kul04a] for the notion of the conflict (multi-)graph), and thus we can find them by finding cycles.

Finally we mention that also for Horn clause-sets we can compute the lean kernel in polynomial time by Theorem 11.10.1. Again this can be handled also by linear autarkies (see Subsection 11.11.3).

11.10.10. Final remarks on the use of autarkies for SAT solving

Until now, the applications of autarky search to practical SAT solving have been rather restricted, mostly focusing on searching autarkies directly in the input, while autarky search at each node of the search tree, as discussed in Subsection 11.10.6, just checks whether "by chance" the partial assignments at hand are

actually autarkies. Now in special cases autarky search as preprocessing will help, but stronger results can only be expected when *systematically* searching for autarkies at *each* node of the search tree. We close this section by showing (by example) that under "normal circumstances" relevant autarkies are created deeper in the search tree. Consider an input clause-set F, and we ask how many autarkies we obtain after application of a partial assignment φ. To simplify the discussion, we assume that F is unsatisfiable (these are also the relevant hard cases created at least during search). The basic intuition is that we obtain the more autarkies the more "redundancies" are contained in F. By definition an unsatisfiable clause-set is "irredundant" iff it is minimally unsatisfiable, and minimally unsatisfiable clause-sets are lean. Now there are stronger "degrees k of irredundancy", achieved by requiring that for every partial assignment φ with $n(\varphi) \leq k$ for some given k we have that $\varphi * F$ is minimally unsatisfiable. Minimally unsatisfiable F are characterised by $k = 0$, saturated minimally unsatisfiable F by $k = 1$, and unsatisfiable hitting clause-sets F by $k = \infty$; see Chapter 6 in [Kul07a] for investigations of these issues. So from the autarky-perspective unsatisfiable hitting clause-sets are the most intractable cases — whatever partial assignment is applied, never any autarky arises. On the other end of the spectrum we have marginal minimally unsatisfiable clause-sets F of deficiency 1 — here already for $n(\varphi) = 1$ the clause-set $\varphi * F$ decomposes in variable-disjoint components, one of them minimally unsatisfiable (again marginal minimally unsatisfiable of deficiency 1), while all other components are satisfiable (and actually "matching-satisfiable"; see Subsection 11.11.2), and so we have (easy, relevant) autarkies. For practical applications, instances will not even be minimally unsatisfiable, and typically also are assembled from several parts, and thus the appearance of relevant autarkies not too deep down the search tree is to be expected.

11.11. Autarky systems: Using weaker forms of autarkies

In this section we consider restricted notions of autarkies, which allow to find such restricted autarkies in polynomial time, while still sharing many of the properties with general autarkies. The first example for a special form of autarkies, "linear autarkies", was introduced in [Kul00b], and the general properties of the formation $F \mapsto \mathrm{Auk}(F)$ of the (full) autarky monoid have been parallelled with the properties of the sub-monoid $F \mapsto \mathrm{LAuk}(F) \leq \mathrm{Auk}(F)$ of linear autarkies. Based on these investigations, in [Kul03] various forms of "autarky systems" have been introduced, axiomatising the basic properties. Some preliminary study on further generalisations can be found in [Kul01], while the notions have been revised and adapted to clause-sets with non-boolean variables in [Kul07a].

In the light of Lemma 11.9.3, an appropriate definition of an "autarky system" would be that of a contravariant functor $\mathcal{A} : \mathfrak{CLS} \to \mathfrak{MON}$ such that the canonical inclusion (of sub-monoids) is a natural transformation $\mathcal{A} \to \mathrm{Auk}$. But since the study of such categorical notions in the context of SAT has not really started yet, we refrain here from such functorial formulations, and call an **autarky system** a map $\mathcal{A}$ defined on all clause-sets such that $\mathcal{A}(F)$ is a sub-monoid of $\mathrm{Auk}(F)$ for all $F \in \mathcal{CLS}$, and such that for $F_1 \subseteq F_2$ we have $\mathcal{A}(F_2) \subseteq \mathcal{A}(F_1)$. The most basic examples of autarky systems are $\mathcal{A} = \mathrm{Auk}$, the full autarky system,

and $\mathcal{A} = (\{\langle\rangle\})_{F \in \mathcal{CLS}}$, the trivial autarky system. From an autarky system $\mathcal{A}$ we obtain the *restricted system* $\mathcal{A}^{\rm r}(F) := \mathcal{A}(F) \cap {\rm Auk}^{\rm r}(F)$, which is again an autarky system if $\mathcal{A}$ is "standardised" (that is, variables not it F do not matter; see below for the definition). The restricted version is especially important for practical considerations, however for theoretical investigations mostly the "full" versions (also using variables not in F) are more convenient. As before, we call the minimal elements of $\mathcal{A}^{\rm r}(F) \setminus \{\bot\}$ *minimal $\mathcal{A}$-autarkies*, and the maximal elements of $\mathcal{A}^{\rm r}(F)$ *maximal $\mathcal{A}$-autarkies*. If $\mathcal{A}^{\rm r}$ is an autarky system, then all maximal autarkies have the same domain (use the same variables).

A clause-set F is called *$\mathcal{A}$-lean* if $\mathcal{A}(F)$ contains only trivial autarkies. Using $\mathcal{A}$-autarky reduction instead of autarky reduction, i.e., reduction $F \rightsquigarrow \varphi * F$ for $\varphi \in \mathcal{A}(F)$, generalising Subsection 11.8.3 we obtain the *$\mathcal{A}$-lean kernel* ${\rm N}_{\mathcal{A}}(F) \subseteq F$, the largest $\mathcal{A}$-lean sub-clause-set of F; note that due to the anti-monotonicity condition for an autarky system, $\mathcal{A}$-autarky reduction still is confluent. Again we have that ${\rm N}_{\mathcal{A}}(F) = F$ holds iff F is $\mathcal{A}$-lean, while in case of ${\rm N}_{\mathcal{A}}(F) = \top$ we call F *$\mathcal{A}$-satisfiable*. The lean kernel formation $F \mapsto {\rm N}_{\mathcal{A}}(F)$ is again a kernel operator. The union of $\mathcal{A}$-lean clause-sets is $\mathcal{A}$-lean, and ${\rm N}_{\mathcal{A}}(F)$ is the union of all $\mathcal{A}$-lean sub-clause-sets of F.

If there exists $\varphi \in \mathcal{A}(F)$ with $\varphi * F = \top$, then F is $\mathcal{A}$-satisfiable, but the reverse direction does not hold in general. In other words, using *$\mathcal{A}$-autark subsets* of F as sub-clause-sets satisfied by some $\mathcal{A}$-autarky for F, we again have that the union of $\mathcal{A}$-autark subsets is again an $\mathcal{A}$-autark subsets, and thus there is a largest $\mathcal{A}$-autark subsets, where the largest $\mathcal{A}$-autark subset is disjoint with the largest lean subset ${\rm N}_{\mathcal{A}}(F)$ — however we do not obtain a partition of F in this way, and additional properties of autarky systems are needed. We do not want to discuss various forms of strengthened autarky systems, but we only consider the strongest form, called a **normal autarky system**, which comprises the six basic properties of autarkies listed in Subsection 11.8.2. More precisely, we require the following five additional properties (for all partial assignments φ, ψ, all clause-sets F and all finite sets V of variables):

standardised $\varphi \in \mathcal{A}(F) \Leftrightarrow \varphi|\,{\rm var}(F) \in \mathcal{A}(F)$;
iterative if $\varphi \in \mathcal{A}(F)$ and $\psi \in \mathcal{A}(\varphi * F)$, then $\psi \circ \varphi \in A(F)$;
invariant under variable elimination if ${\rm var}(\varphi) \cap V = \emptyset$, then $\varphi \in \mathcal{A}(F) \Leftrightarrow \varphi \in \mathcal{A}(V * F)$;
$\bot$-invariant $\mathcal{A}(F) = \mathcal{A}(F \cup \{\bot\})$;
invariant under renaming if F' is isomorphic to F, then the same isomorphism turns ${\rm Auk}(F)$ into ${\rm Auk}(F')$.

(Note that for the functorial definition of an autarky system mentioned above, the property of invariance under renaming is automatically fulfilled.) If $\mathcal{A}$ is a normal autarky system, then the restricted system $\mathcal{A}^{\rm r}$ is an autarky system which fulfils all additional properties except of being standardised (that is, $\mathcal{A}^{\rm r}$ is iterative, invariant under variable elimination, $\bot$-invariant and invariant under renaming). If $\mathcal{A}_1, \mathcal{A}_2$ are (normal) autarky systems, then also $F \mapsto \mathcal{A}_1(F) \cap \mathcal{A}_2(F)$ is a (normal) autarky system.[10]

[10] More generally, arbitrary intersections of (normal) autarky systems are again (normal) autarky systems, and we obtain the complete lattice of autarky systems and the complete

We assume from now on that $\mathcal{A}$ is a normal autarky system, which can be justified by the fact that every autarky system has a unique strengthening which is a normal autarky system (just by adding the "missing" autarkies). Now for all clause-sets F there is an $\mathcal{A}$-maximal autarky φ with $\varphi * F = \mathrm{N}_{\mathcal{A}}(F)$, especially F is $\mathcal{A}$-satisfiable iff there is $\varphi \in \mathcal{A}(F)$ with $\varphi * F = \top$. Regarding the full autarky system, from an autarky φ with $\varphi * F = \mathrm{N}_{\mathrm{a}}(F)$ we can easily obtain a maximal autarky φ', by adding arbitrary assignments for variables in $\mathrm{var}(F \setminus \varphi * F)$ which do not already have a value (and also variables not in F need to be removed). However for arbitrary (normal) autarky systems this might not be possible, and so we call $\mathcal{A}$-autarkies φ with $\varphi * F = \mathrm{N}_{\mathcal{A}}(F)$ *quasi-maximal*. Every $\mathcal{A}$-maximal autarky is $\mathcal{A}$-quasi-maximal.

If F is $\mathcal{A}$-lean, then also $V * F$ and $F[V]$ are $\mathcal{A}$-lean for (finite) sets V of variables, and every clause-set $F' \supseteq F$ with $\mathrm{var}(F') = \mathrm{var}(F)$ is $\mathcal{A}$-lean as well.[11] The $\mathcal{A}$-autark subsets of F, i.e., the subsets $F_{\mathrm{var}(\varphi)}$ for $\varphi \in \mathcal{A}(F)$, can be characterised as the $\mathcal{A}$-satisfiable subsets F_V for $V \subseteq \mathrm{var}(F)$, where furthermore F_V is $\mathcal{A}$-satisfiable iff $F[V]$ is $\mathcal{A}$-satisfiable. The set of $\mathcal{A}$-autark subsets of F is closed under union, with smallest element $\top$ and largest element $F \setminus \mathrm{N}_{\mathcal{A}}(F)$. The partition of F into $\mathrm{N}_{\mathcal{A}}(F)$ and $F \setminus \mathrm{N}_{\mathcal{A}}(F)$ (where some part can be empty) has, as before, several characterisations; it is the partition of F into the largest $\mathcal{A}$-lean and the largest $\mathcal{A}$-autark subset, or, equivalently, if $F = F_1 \cup F_2$, where F_1 is $\mathcal{A}$-lean and $\mathrm{var}(F_1) * F_2$ is $\mathcal{A}$-satisfiable, then $F_1 = \mathrm{N}_{\mathcal{A}}(F)$ and $F_2 = F \setminus \mathrm{N}_{\mathcal{A}}(F)$ (and also the other way around; note that disjointness of F_1, F_2 actually follows from $\mathrm{var}(F_1) * F_2$ being satisfiable). $\mathcal{A}$-satisfiability is "self-reducible" in the sense that (in the same way as for the full autarky system) from an oracle deciding $\mathcal{A}$-leanness we obtain efficiently a way for computing the lean kernel (see Lemma 8.6 in [Kul03]); however for also computing an $\mathcal{A}$-quasi-maximal autarky (as in [KMT08]) additional conditions on the autarky system $\mathcal{A}$ are needed.

11.11.1. Pure autarkies

In Example 11.8.3 we have already seen that every pure literal x for a clause-set F induces an autarky $\langle x \to 1\rangle \in \mathrm{Auk}(F)$. This is not yet an autarky system, since we do not have closure under composition. So we call a partial assignment φ a *simple pure autarky* for F if for all literals x with $\varphi(x) = 1$ we have $\overline{x} \notin \bigcup F$; the set of all simple pure autarkies is denoted by $\mathrm{PAut}_0(F)$. In this way we obtain an autarky system $F \mapsto \mathrm{PAut}_0(F)$, which also is standardised, invariant under variable elimination, $\bot$-invariant, and invariant under renaming, however not iterative (applying a simple pure autarky can create new pure literals).

Example 11.11.1. For $F := \{\{a, b\}, \{\overline{b}\}\}$ we have $\mathrm{PAut}_0(F) = \{\langle\rangle, \langle a \to 1\rangle\}$, and so the unique PAut_0-maximal autarky is $\langle a \to 1\rangle$. However this is not a satisfying assignment for F, despite the fact that F is PAut_0-satisfiable.

---

lattice of normal autarky systems, and the corresponding closures (hulls; as always in algebra).

[11] We remark here that for *formal clause-sets*, which correspond to the notion of a hypergraph and are pairs (V, F) for sets V of variables and clause-sets over V (so that now "formal variables" are allowed, i.e., elements of $V \setminus \mathrm{var}(F)$), $\mathcal{A}$-leanness of (V, F) implies $V = \mathrm{var}(F)$, since otherwise a non-trivial autarky would exist.

To obtain a normal autarky system, according to the general theory we consider $\mathrm{PAut}(F)$, which is the closure of $\mathrm{PAut}_0(F)$ under "iteration" of autarkies. So the elements of $\mathrm{PAut}(F)$ are the partial assignments $\varphi_m \circ \cdots \circ \varphi_1$, where φ_i is a simple pure autarky for F_{i-1}, where $F_0 := F$ and $F_{i+1} := \varphi_{i+1} * F_i$. The elements of $\mathrm{PAut}(F)$ are called *pure autarkies*.

Given any autarky system $\mathcal{A}$, it is natural to study which random clause-sets (say, in the constant-density model, fixing a clause-length $k \geq 2$, and considering all k-uniform clause-lists of length c over n given variables as equally likely) are $\mathcal{A}$-satisfiable. Experimental evidence seems to suggest that at least the "natural" autarky systems considered here all show a threshold behaviour (below a certain threshold density every clause-list is almost surely $\mathcal{A}$-satisfiable, above that density almost surely none). Only for pure autarkies this has been finally rigorously established in [Mol05], proving that for all $k \geq 2$ a threshold exists, at the density $\frac{c}{n} = 1$ for $k = 2$ and at $\frac{c}{n} = 1.636\ldots$ for $k = 3$.

11.11.2. Matching autarkies

The notion of "matching autarkies" is based on the work in [FV03], which introduced "matched clause-sets", that is, clause-sets F with $\mathrm{d}^*(F) = 0$ (recall Definition 11.2.1[12]), which are the *matching satisfiable clause-sets* using our systematic terminology for autarky systems (that is, clause-sets satisfiable by matching autarkies). Perhaps the earliest work where matching autarkies are implicitly used is [AL86]. Matching satisfiable sub-clause-sets of a clause-set F as the independent sets of a transversal matroid have been used in [Kul00a] to show that SAT for clause-sets with bounded maximal deficiency can be decided in polynomial time. The first thorough study of matching lean clause-sets (clause-sets without non-trivial matching autarkies) one finds in [Kul03], completed (and extended to clause-sets with non-boolean variables) in [Kul07a]. As shown there, the basic tasks like finding a maximal matching autarkies can be performed in polynomial time. The structure of the matching autarky monoid has not been investigated yet.

As introduced in [Kul03], "matching autarkies" are partial assignments which satisfy in every clause they touch a literal with underlying *unique* variable.

Definition 11.11.1. A partial assignment φ is called a *matching autarky* for a clause-set F if there is an injection $v : F_{\mathrm{var}(\varphi)} \to \mathrm{var}(\varphi)$ such that for all clauses $C \in F_{\mathrm{var}(\varphi)}$ there is a literal $x \in C$ with $v(C) = \mathrm{var}(x)$ and $\varphi(x) = 1$.

Clearly every matching autarky is in fact an autarky. The set of matching autarkies is denoted by $\mathrm{MAuk}(F)$ resp. $\mathrm{MAuk}^{\mathrm{r}}(F)$ (considering only matching autarkies with variables in F).

Example 11.11.2. If every literal occurs only once, then every autarky is a matching autarky; see for example Example 11.9.1. As the simplest example where some autarkies are left out consider $F := \{\{v_1\}, \{v_1, v_2\}\}$; we have $\mathrm{MAuk}^{\mathrm{r}}(F) = \{\langle\rangle, \langle v_2 \to 1\rangle, \langle v_1 \to 1, v_2 \to 1\rangle\}$ (note that $\langle v_1 \to 1\rangle$ is an autarky for F but not a matching autarky). While F is still matching satisfiable, the clause-set $\{\{v_1, v_2\}, \{v_1, \overline{v_2}\}, \{\overline{v_1}, v_2\}\}$ is satisfiable but matching lean.

[12] we remark that also the notion of "deficiency" was introduced in [FV03]

Using *multi*-clause-sets F instead of clause-sets, a partial assignment φ is a matching autarky for F iff $F[\mathrm{var}(\varphi)]$ is matching satisfiable (we have to use multi-clause-sets in order to avoid the contraction of clauses).

Example 11.11.3. $\langle v \to 1 \rangle$ is *not* a matching autarky for $F := \{\{v\}, \{w\}, \{v, w\}\}$, and accordingly as multi-clause-set $F[\mathrm{var}(\varphi)]$ consists of two occurrences of the clause $\{v\}$, and thus is not matching satisfiable. But as a clause-set we would obtain $F[\mathrm{var}(\varphi)] = \{\{v\}\}$ which is matching satisfiable.

The operator $F \mapsto \mathrm{MAuk}(F)$ yields a normal autarky system, but only for multi-clause-sets; for clause-sets we have an autarky system which fulfils all properties of normal autarky systems except of being invariant under variable elimination. The matching satisfiable clause-sets are exactly the clause-sets with $\mathrm{d}^*(F) = 0$, while the matching lean clause-sets are exactly the clause-sets F with $\forall\, F' \subset F : \mathrm{d}(F') < \mathrm{d}(F)$ (thus $\mathrm{d}^*(F) = \mathrm{d}(F)$, and if $F \neq \top$, the $\mathrm{d}(F) \geq 1$). This generalises the fact from [AL86] that minimally unsatisfiable clause-sets have deficiency at least 1. More on matching autarkies one finds in Section 7 of [Kul03], and in Section 4 of [Kul07a].

11.11.3. Linear autarkies

The notion of a "linear autarky", introduced in [Kul00b], was motivated by the applications of linear programming to SAT solving in [vW00] (see Theorem 2 there). The basic notion is that of "simple linear autarkies", yielding an autarky system which fulfils all conditions of a normal autarky system except of being iterative, which then is repaired by the notion of a "linear autarky". Let a *variable weighting* be a map $w : \mathcal{VA} \to \mathbb{Q}_{>0}$, that is, every variable gets assigned a positive rational number (real numbers could also be used). We extend the weighting to literals and clauses, relative to a partial assignment φ, as follows (using an arithmetisation of "false" by -1 and "true" by $+1$):

- For a literal x let $\mathrm{sgn}_\varphi(x) := 0$ if $\mathrm{var}(x) \notin \mathrm{var}(\varphi)$, otherwise $\mathrm{sgn}_\varphi(x) := +1$ if $\varphi(x) = 1$, while $\mathrm{sgn}_\varphi(x) := -1$ if $\varphi(x) = 0$.
- Now for a literal x let $w_\varphi(x) := \mathrm{sgn}_\varphi(x) \cdot w(\mathrm{var}(v))$.
- And for a clause C we define $w_\varphi(C) := \sum_{x \in C} w_\varphi(x)$,

Note that for clauses C we have $w_\varphi(\overline{C}) = -w_\varphi(C)$. Now we can state the definition of "simple linear autarkies" (as introduced in [Kul00b]):

Definition 11.11.2. A partial assignment φ is a *simple linear autarky* for a clause-set F if there exists a weighting $w : \mathcal{VA} \to \mathbb{Q}_{>0}$ such that for all $C \in F$ we have $w_\varphi(C) \geq 0$.

Clearly a simple linear autarky is an autarky, and also clearly we only need to consider the variables actually occurring in F. Defining *linear autarkies* as the closure of simple linear autarkies under iteration we obtain a normal autarky system.

11.11.3.1. Comparison to pure and matching autarkies

Example 11.11.4. Every pure literal x for a clause-set F yields a simple linear autarky $\langle x \to 1\rangle$, certified by any constant weighting. The partial assignment $\langle v_1 \to 1, v_2 \to 2\rangle$ is a (satisfying) simple linear autarky for the matching lean clause-set $\{\{v_1, v_2\}, \{v_1, \overline{v_2}\}, \{\overline{v_1}, v_2\}\}$ from Example 11.11.2, again certified by any constant weighting. The partial assignment $\varphi := \langle v_1, v_2, v_3 \to 1\rangle$ is not a simple linear autarky for $F := \{\{v_1, \overline{v_2}, \overline{v_3}\}, \{\{\overline{v_1}, v_2\}\}, \{v_3\}\}$, since if it had a certificate w, then $w(\overline{v_3}) < 0$ in the first clause, so from $0 \le w(\{v_1, \overline{v_2}, \overline{v_3}\}) = w(\{v_1, \overline{v_2}\}) + w(\overline{v_3})$ we obtain $w(\{v_1, \overline{v_2}\}) \ge -w(\overline{v_3}) > 0$, but $w(\{\overline{v_1}, v_2\}) = -w(\{v_1, \overline{v_2}\})$, and thus the second clause gets a negative weight. On the other hand, φ is a matching autarky for F.

So it might seem that simple linear autarkies and matching autarkies are incomparable in strength, however this is misleading: every clause-set which has a non-trivial matching autarky also has a non-trivial simple linear autarky, as shown in [Kul00b] (by basic linear algebra one shows that a matching satisfiable clause-set $F \neq \top$ must have a non-trivial simple linear autarky, and thus F must be linearly satisfiable (by iteration); using invariance under variable elimination this is lifted to arbitrary non-matching-lean F). So the linearly-lean kernel of a clause-set is always contained in the matching-lean kernel. In Example 11.11.4 the clause-set F has the simple linear autarky $\langle v_1 \to 1, v_2 \to 1\rangle$, and while the partial assignment φ is not a simple linear autarky, it is actually a linear autarky.

Example 11.11.5. Consider $F := \{\{v_1, \overline{v_2}, \overline{v_3}\}, \{\overline{v_1}, v_2, \overline{v_3}\}, \{\overline{v_1}, \overline{v_2}, v_3\}\}$. The partial assignment $\varphi := \langle v_1, v_2, v_3 \to 1\rangle$ is a matching autarky for F but not a simple linear autarky; furthermore φ is a minimal autarky for F and thus φ is also not a linear autarky. So we see that, although the reduction power of linear autarkies is as least as big as the reduction power of matching autarkies (in this example, $\langle v_1, v_2 \to 0\rangle$ is a simple linear autarky for F), there are matching autarkies which do not contain any non-trivial linear autarkies.

11.11.3.2. Matrix representations of clause-sets

Now we turn to the question of finding linear autarkies in polynomial time. This is best done by translating the problem into the language of linear programming. For a clause-set F consider the clause-variables matrix $M(F)$ of size $c(F) \times n(F)$. Let us emphasise that this is different to the use of matrices in Example 11.2.1, where matrices only are used for an economic representation, while here now "mathematical" matrices are used, with entries from $\{-1, 0, +1\} \subset \mathbb{Z}$, where the rows correspond to the clauses of F (in some order), and the columns correspond to the variables of F (in some order). Now F is linearly lean if and only if $M(F) \cdot \vec{x} \ge 0$ has the only solution $\vec{x} = 0$, and furthermore every solution $\vec{x}$ corresponds to a simple linear autarky by the same arithmetisation as used before, i.e., 0 corresponds to "not assigned", $+1$ corresponds to "true" and -1 corresponds to "false". Thus by linear programming we can find a non-trivial linear autarky (if one exists).

Via linear autarkies, several basic cases of poly-time SAT decision are covered. Using a constant weighting, we see that every autarky for a clause-set F in 2-CNF (every clause has length at most 2) is a simple linear autarky, and thus every

satisfiable 2-CNF is linearly satisfiable. [van00] applies linear autarkies to q-Horn formulas, while in [Kul99a] it is shown how satisfiable generalised Horn clause-sets (in a natural hierarchy of generalised Horn clause-sets, covering all clause-sets) can be captured by linearly satisfiable clause-sets (used as an oracle to amplify the general hierarchy studied in [Kul99a]). For more on linear autarkies see [Kul00b, Kul03], while [Kul06] contains more on "balanced" linear autarkies (see Subsection 11.11.4). In general the field of linear autarkies appears still largely unexplored; for example there seems to be a threshold for linear satisfiability of random 3-CNF at around density 2 (i.e., having twice as many clauses as variables).

Since matrix representations of satisfiability problems are at least of great theoretical value, we conclude this subsection on linear autarkies by some general comments on technical problems arising in this context. First, what is a "matrix"?! The appropriate definition of the notion of a (general) matrix is given in [Bou89], where a *matrix over a set* V is defined as a triple (R, C, f), where R, C are arbitrary sets (the "row indices" and the "column indices"), and $f : R \times C \to V$ gives the entries of the matrix. It is important to note that for row and column indices arbitrary objects can be used, and that no order is stipulated on them. In this context it is appropriate to introduce *labelled clause-sets* (corresponding to general hypergraphs) as triples (V, F_0, F), where V, F_0 are arbitrary sets (the "variables" and the "clause-labels"), while $F : F_0 \to \mathcal{CLS}(V)$ maps every clause-label to a clause over V. Note that labelled clause-sets can have *formal variables*, i.e., variables $v \in V$ not occurring in the underlying clause-set ($v \notin \mathrm{var}(F(F_0))$), and that clauses can occur multiple times (furthermore clauses have "names"). Now we have a perfect correspondence between finite matrices over $\{-1, 0, +1\}$ and finite labelled clause-sets, more precisely we have two ways to establish this correspondence, either using the *clause-variable matrix*, which assigns to (V, F_0, F) the matrix (F_0, V, F') (where $F'(C, v)$ is ± 1 or 0, depending on whether v occurs in $F(C)$ positively, negatively or not at all), or using the *variable-clause matrix* (V, F_0, F'') (with $F''(v, C) := F'(C, v)$). *Standardised matrices* use row-sets $R = \{1, \ldots, n\}$ and column-sets $C = \{1, \ldots, m\}$ for $n, m \in \mathbb{N}_0$, and in this way also "automatically" an order on the rows and columns is established (so that standardised matrices can be identified with "rectangular schemes of values"). Ironically this less appropriate notion of a matrix (with its ambiguous treatment of order) is common in the field of combinatorics, so that, in the spirit of [BR91], in this context we call the above matrices "combinatorial matrices". In the context of the autarky part of this chapter, clause-variable matrices (as well as its transposed, variable-clause-matrices) are combinatorial matrices, so that we save us the ordeal of having to determine somehow an order on variables and clauses.[13]

Clause-sets do not allow formal variables nor multiple clauses, and thus their clause-variable matrices don't have zero columns nor identical rows. Thus when we speak of a matrix corresponding to a clause-set, then by default we consider clause-variable matrices, and the matrices don't contain zero columns or identical

[13] In Subsection 11.12.2 we will consider the determinant of (square) clause-variable matrices, and the determinant of a combinatorial matrix is only determined up to its sign — fortunately actually only the absolute value of the determinant is needed there!

rows (while when considering labelled clause-sets no such restrictions are applied).

11.11.4. Balanced autarkies

We now consider the important (normal) autarky system of "balanced autarkies" which, unlike pure, matching and linear autarkies, has an NP-complete search problem (as the full autarky system): the main motivation behind balanced autarkies perhaps can be understood as providing a bridge from hypergraph theory to satisfiability theory.

Definition 11.11.3. A partial assignment φ is a *balanced autarky* for a clause-set F if φ an autarky for F as well as for $\overline{F} = \{\overline{C} : C \in F\}$, or, in other words, if in every clause $C \in F$ touched by φ there are satisfied literals as well as falsified literals.

Some basic facts about balanced autarkies are as follows:

1. Balanced autarkies yield a normal autarky system.
2. Balanced-satisfiable clause-sets, i.e., clause-sets satisfiable by balanced autarkies, are exactly the instances of NAESAT ("not-all-equal-sat"), that is, clause-sets which have a satisfying assignment which in every clause also falsifies some literals.
3. φ is a balanced autarky for F iff φ is an autarky for $F \cup \overline{F}$.
4. For *complement-invariant* clause-sets, i.e., where $F = \overline{F}$ holds, the balanced autarkies are exactly the autarkies. Especially, the satisfying partial assignments of complement-invariant clause-sets are exactly the balanced-satisfying assignments.
5. A complement-invariant clause-set F is satisfiable iff the underlying variable-hypergraph $\mathfrak{V}(F) = (\mathrm{var}(F), \{\mathrm{var}(C) : C \in F\})$ is 2-colourable.[14]

An example for a class of balanced lean clause-sets (having no non-trivial balanced autarky) is the class of clause-sets $F \setminus \{C\}$ for $F \in \mathrm{MU}(1)$ and $C \in F$ (note that every such $F \setminus \{C\}$ is matching satisfiable).[15]

Regarding the intersection of the autarky system of balanced autarkies with the autarky systems presented until now, *balanced linear autarkies* seem most important (that is, linear autarkies which are also balanced, constituting a strong autarky system).

[14] A *hypergraph* is a pair (V, E), where V is the set of "vertices", while E, the set of "hyperedges", is a set of subsets of V. A 2-colouring of a hypergraph (V, E) is a map $f : V \to \{0, 1\}$ such that no hyperedge is "monochromatic", i.e., the 2-colourings of F correspond exactly to the satisfying assignments for the (complement-invariant) clause-set $F_2((V, E)) := E \cup \overline{E}$ (using the vertices as variables).

[15] That every such $F \setminus \{C\}$ is balanced lean can be shown by induction on $n(F)$. The assertion is trivial for $n(F) = 0$, so assume $n(F) > 0$. F must contain a variable v occurring positively and negatively only once. If $v \notin \mathrm{var}(C)$, then we can apply the induction hypothesis to F' obtained from F by (singular) DP-resolution on v. If on the other hand there is no such v, then F is renamable to a Horn clause-set, so, since F is unsatisfiable, F must contain a unit-clause $\{x\} \in F$, and we can apply the induction hypothesis to F'' obtained from F by (singular) DP-resolution on $\mathrm{var}(x)$.

Example 11.11.6. The clause-set $F = \{\{v_1, \overline{v_2}, \overline{v_3}\}, \{\overline{v_1}, v_2, \overline{v_3}\}, \{\overline{v_1}, \overline{v_2}, v_3\}\}$ from Example 11.11.5 is balanced satisfiable (for example by setting all variables to 1 or setting all variables to 0), while F is balanced linearly lean (has no non-trivial balanced lean autarky).

Balanced linear autarkies for a clause-set F can be found easier than general linear autarkies, by just solving a system of linear equations (not linear inequalities), namely they correspond to the solution of $M(F) \cdot x = 0$ (recall Subsection 11.11.3). As already alluded to in Subsection 11.4.5, they have been implicitly used in [Sey74]: While matching autarkies are used to show that minimally unsatisfiable clause-sets have a deficiency at least 1, balanced linear autarkies are used to show that minimally non-2-colourable hypergraphs have deficiency at least 0 (where we define the deficiency of a hypergraph (V, E) as $|E| - |V|$, the difference of the number of hyperedges and the number of vertices). Slightly more general, any clause-set F which is balanced linearly lean has $\mathrm{d}(F) \geq 0$, which follows directly from the matrix characterisation of balanced linear autarkies; actually for such F we have the stronger property $\mathrm{d}^*(F) = \mathrm{d}(F)$ (Lemma 7.2 in [Kul06]). For more on the relation of balanced autarkies to combinatorics see Subsections 11.12.1, 11.12.2.

11.11.5. Generalising minimally unsatisfiable clause-sets

In [Kul07b] the notion of **minimally $\mathcal{A}$-unsatisfiable** clause-sets F has been introduced for arbitrary (normal) autarky systems $\mathcal{A}$, which are clause-sets F which are $\mathcal{A}$-unsatisfiable but where every strict sub-clause-set is $\mathcal{A}$-satisfiable. The motivation is as follows: The hypergraph 2-colouring problem for a hypergraph $G = (V, E)$ (where V is a finite set of vertices, and E is a set of subsets of V) can be naturally expressed as the satisfiability problem for the clause-set $F_2(G)$ (as already mentioned in Subsections 11.4.5 11.11.4), using the elements of V as variables, while the clauses are the hyperedges $H \in E$ itself (which become now "positive clauses") together with the complements $\overline{H}$ (which become "negative clauses"). The notion of *minimally non-2-colourable hypergraphs* (or *critically 3-colourable hypergraphs*) has been studied intensively in the literature (see for example Chapter 15 in [JT95]). It is easy to see that G is minimally non-2-colourable if and only if $F_2(G)$ is minimally unsatisfiable. Analogously to clause-sets, for hypergraphs G we can consider the *deficiency* $\mathrm{d}(G) := |E(G)| - |V(G)|$. It has been shown in [Sey74] that $\mathrm{d}(G) \geq 0$ holds for minimally non-2-colourable hypergraphs. The situation is very similar to $\mathrm{d}(F) \geq 1$ for minimally unsatisfiable clause-sets, and indeed, while the proof of this property for clause-sets relies on matching autarkies, for hypergraphs we use balanced linear autarkies instead (thus replacing matching theory by linear algebra; see [Kul06] for some initial investigations on this subject). Deciding the class MU(1) in polynomial time has been discussed in Subsection 11.2.2, while the analogous problem for hypergraphs, deciding whether a *square hypergraph* (of deficiency 0) is minimally 2-colourable, is actually a much more difficult problem, which was finally resolved in [RST99, McC04] (see Subsection 11.12.2). Now satisfiability of the elements of MU(1) after removal of a clause is of a very special kind, namely they are *matching satisfiable*, and analogously satisfiability of square minimally non-2-colourable

hypergraphs after removal of a hyperedge is covered by *balanced linear satisfiability* (regarding the translation $F_2(G)$). So we have several natural and important cases of a restricted kind of minimal unsatisfiability, namely where satisfiability after removal of a clause yields an $\mathcal{A}$-satisfiable clause-set for some (feasible) autarky system $\mathcal{A}$. And instead of requiring that the whole clause-set must be unsatisfiable, we only need it to be $\mathcal{A}$-unsatisfiable; thus minimal $\mathcal{A}$-unsatisfiability at the same time strengthens and weakens the notion of minimal unsatisfiability.

Regarding the full autarky system Auk, minimal Auk-unsatisfiability is just minimal unsatisfiability. And every minimally unsatisfiable clause-set is minimally $\mathcal{A}$-unsatisfiable for every autarky system $\mathcal{A}$. The minimally matching unsatisfiable (i.e., minimally MAuk-unsatisfiable) clause-sets F are exactly the matching lean clause-sets of deficiency 1, which includes MU(1), while all other such F are satisfiable.

We have already seen another generalisation of (ordinary) minimal unsatisfiability, namely the notion of $\mathcal{A}$-lean clause-sets. Now minimal unsatisfiable clause-sets F are not just lean, but they are "minimally lean" in the sense that every strict sub-clause-set except of $\top$ is not lean, so one could introduce the notion of "minimally $\mathcal{A}$-lean clause-sets" — however the reader can easily convince himself that a clause-set F is "minimally $\mathcal{A}$-lean" iff F is minimally $\mathcal{A}$-unsatisfiable. So to arrive at a new notion, we have to be more careful, and instead of demanding that every strict sub-clause-set of F except of $\top$ is not $\mathcal{A}$-lean, we require that after removal of any *single* clause we obtain a non-$\mathcal{A}$-lean clause-set. We have arrived at the notion of a **barely $\mathcal{A}$-lean** clause-set, which generalises the notion of a "barely L-matrix" in [BS95] (see Subsection 11.12.1 below).

Generalising the notion of an "L-indecomposable matrix" in [BS95] (again, compare Subsection 11.12.1), in [Kul07b] also the notion of an **$\mathcal{A}$-indecomposable clause-set** has been introduced. This notion applies to $\mathcal{A}$-lean clause-sets, where an $\mathcal{A}$-lean clause-set F by definition is *$\mathcal{A}$-decomposable* if the clause-variable matrix $M(F)$ can be written as $\left(\begin{smallmatrix} A & 0 \\ * & B \end{smallmatrix}\right)$ for non-empty matrices A, B (having at least one row and one column) corresponding to clause-sets F_1, F_2 which are $\mathcal{A}$-lean (then the pair (F_1, F_2) yields an *$\mathcal{A}$-decomposition* of F). Note that since we are dealing here with clause-sets, B may not contain repeated rows (as in Definition 4 of [Kul07b]), which is justified when only Theorem 11.11.1 is considered, since if F is barely $\mathcal{A}$-lean, then no contraction can occur in an $\mathcal{A}$-decomposition.

Example 11.11.7. Consider two (different) variables a, b and the clause-set $F := F_1 \cup F_2'$, where $F_1 := \{\{a\}, \{\overline{a}\}\}$ and $F_2' := \{\{a, b\}, \{\overline{a}, \overline{b}\}\}$. We consider the full autarky system. Now F is barely lean, but not minimally unsatisfiable, and an autarky-decomposition of F is given by F_1, F_2 for $F_2 := F[\{b\}] = \{\{b\}, \{\overline{b}\}\}$. The clause-set $F' := F_1 \cup F_2''$ for $F_2'' := F_2' \cup \{\{a, \overline{b}\}\}$ has no autarky-decomposition as a clause-set (but see below), and it is lean but not barely lean.

The basic theorem now is ([Kul07b]):

Theorem 11.11.1. *Consider a normal autarky system $\mathcal{A}$. Then a clause-set F with at least two clauses is minimally $\mathcal{A}$-unsatisfiable if and only if F is barely $\mathcal{A}$-lean and $\mathcal{A}$-indecomposable.*

If the autarky system $\mathcal{A}$ is sensitive to repetition of clauses (as are matching

autarkies; recall Example 11.11.3), then multi-clause-sets or even labelled clause-sets (which additionally allow for formal variables and for clause-labels; recall Subsection 11.11.3.2) are needed. For the above definition of $\mathcal{A}$-(in)decomposability then labelled clause-sets are used instead of clause-sets, and so the submatrices A, B may contain zero columns and repeated rows (but note that if A or B contains a zero column then the corresponding labelled clause-set cannot be $\mathcal{A}$-lean).

Example 11.11.8. Considering $F' = \{\{a\}, \{\overline{a}\}, \{a, b\}, \{\overline{a}, \overline{b}\}, \{a, \overline{b}\}\}$ from Example 11.11.7 as a multi-clause-set, it has now the autarky-decomposition consisting of $\{a\} + \{\overline{a}\}$ and $\{b\} + 2 \cdot \{\overline{b}\}$. And the clause-set $\{\{a\}, \{\overline{a}\}, \{a, b\}, \{\overline{a}, b\}\}$ is matching lean (but not barely matching lean), and as a clause-set it has no matching-autarky-decomposition, while as a multi-clause-set it has the matching autarky decomposition consisting of $\{a\} + \{\overline{a}\}$ and $2 \cdot \{b\}$.

An $\mathcal{A}$-lean labelled clause-set $F = (V, F_0, F^*)$ has an $\mathcal{A}$-autarky decomposition if and only if there exists $\emptyset \subset V' \subset V$ such that the clause-label-set $F_0' := \{C \in F_0 : \mathrm{var}(F^*(C)) \subseteq V'\}$ is not empty and the labelled clause-set $F_1 := (V', F_0', F^*|F_0')$ is $\mathcal{A}$-lean (this corresponds to the matrix A in the above definition, while $F_2 = F[V \setminus V']$; note that leanness of F_2 is implied by leanness of F, and that in general zero rows in B can be moved to A).

11.11.6. The basic algorithmic problems

To conclude this section on autarky systems, we collect now the basic algorithmic problems concerning autarkies. First we need notations for the various classes of clause-sets involved; consider a class $\mathbb{C} \subseteq \mathcal{CLS}$ of clause-sets and a normal autarky system $\mathcal{A}$. By $\mathcal{A}$-SAT($\mathbb{C}$) resp. $\mathcal{A}$-UNSAT($\mathbb{C}$) we denote the $\mathcal{A}$-satisfiable resp. $\mathcal{A}$-unsatisfiable clause-sets in $\mathbb{C}$, by $\mathcal{A}$-LEAN($\mathbb{C}$) the $\mathcal{A}$-lean clause-sets in $\mathbb{C}$, by $\mathcal{A}$-MU($\mathbb{C}$) the minimally $\mathcal{A}$-unsatisfiable clause-sets in $\mathbb{C}$, and finally by $\mathcal{A}$-BRLEAN($\mathbb{C}$) resp. $\mathcal{A}$-INDEC($\mathbb{C}$) the barely $\mathcal{A}$-lean clause-sets resp. the $\mathcal{A}$-indecomposable clause-sets in $\mathbb{C}$. If $\mathcal{A}$ is not mentioned, then the default is the full autarky system, and if $\mathbb{C}$ is not mentioned, then the default is the class $\mathcal{CLS}$ of all clause-sets. Thus $\mathrm{MU}(k) = \mathrm{MU}(\{F \in \mathcal{CLS} : \mathrm{d}(F) = k\})$, while LEAN is the set of all lean clause-sets.

The basic *decision problems* are now as follows, as "promise problems", that is, where $F \in \mathbb{C}$ is already given:

1. The Autarky Existence Problem for $\mathcal{A}$ and $\mathbb{C}$ is to decide whether $F \in \mathcal{CLS} \setminus \mathcal{A}$-LEAN($\mathbb{C}$) holds, while its negation, the Leanness Problem, is to decide $F \in \mathcal{A}$-LEAN($\mathbb{C}$).
2. The Satisfiability Problem for $\mathcal{A}$ and $\mathbb{C}$ is to decide whether $F \in \mathcal{A}$-SAT($\mathbb{C}$) holds, while its negation, the Unsatisfiability Problem, is to decide $F \in \mathcal{A}$-UNSAT($\mathbb{C}$).
3. The Minimal Unsatisfiability Problem for $\mathcal{A}$ and $\mathbb{C}$ is to decide whether $F \in \mathcal{A}$-MU($\mathbb{C}$) holds.
4. The Barely Leanness Problem is to decide $F \in \mathcal{A}$-BRLEAN($\mathbb{C}$).
5. The Indecomposability Problem is to decide $F \in \mathcal{A}$-INDEC($\mathbb{C}$).

If we can solve the satisfiability problem "efficiently", and $\mathbb{C}$ is stable under formation of sub-clause-sets (i.e., if $F \in \mathbb{C}$ and $F' \subseteq F$, then also $F' \in \mathbb{C}$), then

by definition we can solve "efficiently" the minimal unsatisfiability problem. If we can solve "efficiently" the autarky existence problem, and $\mathbb{C}$ is stable under formation of sub-clause-sets, then by definition the barely leanness problem can also be solved "efficiently".

The most basic *functional problems* are as follows (again for input $F \in \mathbb{C}$):

1. The LEAN KERNEL PROBLEM for $\mathcal{A}$ and $\mathbb{C}$ is the problem of computing the lean kernel $\mathrm{N}_{\mathcal{A}}(F)$.
2. The NON-TRIVIAL AUTARKY PROBLEM for $\mathcal{A}$ and $\mathbb{C}$ is the autarky existence problem together with the computation of some non-trivial autarky in case it exists.
3. The QUASI-MAXIMAL AUTARKY PROBLEM for $\mathcal{A}$ and $\mathbb{C}$ is the problem of computing some quasi-maximal autarky.

If we can solve the non-trivial autarky problem "efficiently", and $\mathbb{C}$ is stable under formation of sub-clause-sets, then by iteration we can solve "efficiently" the quasi-maximal autarky problem. Obviously solving the quasi-maximal autarky problem implies solving the non-trivial autarky problem and the lean kernel problem, and solving the lean kernel problem implies solving the satisfiability problem and the autarky existence problem. Call $\mathbb{C}$ *stable* if $\mathbb{C}$ is stable under formation of sub-clause-sets and under crossing out variables (that is, if $F \in \mathbb{C}$, then for any set V of variables we have $V * F \in \mathbb{C}$); stability implies stability under application of partial assignments and under restrictions.[16] Now the proof of Lemma 8.6 in [Kul03] yields an algorithm, which for stable classes allows to derive from an "efficient" solution of the autarky existence problem an "efficient" solution of the lean kernel problem; however, as already mentioned at the end of the introduction to Section 11.11, in order to solve also the non-trivial autarky problem further conditions on $\mathcal{A}$ are needed.

Theorem 11.10.1 yields a method for computing maximal (general) autarkies for classes $\mathbb{C}$ which are stable under restriction and for which we can solve the satisfiability problem and the unsatisfiability problem in a functional sense. It is not known whether Theorem 11.10.1 can be generalised to more general autarky systems. Regarding the full problems (unrestricted autarkies and unrestricted clause-sets), as already remarked in the introduction to Section 11.10, the autarky existence problem is NP-complete and the leanness problem is coNP-complete. Of course, the (full) satisfiability problem is NP-complete, the (full) unsatisfiability problem coNP-complete, and the full minimal unsatisfiability problem is D^P-complete. Finally, (full) barely leanness decision is D^P-complete ([KMT08]), while for the (full) indecomposability problem it is only known that it is in Π_2 (the second level of the polynomial hierarchy).

Considering the classes $\mathcal{CLS}(k) := \{F \in \mathcal{CLS} : \mathrm{d}^*(F) = k\}$ (and still unrestricted autarkies; note that these classes are stable under sub-clause-formation), the satisfiability problem is not only polynomial-time solvable for fixed k (recall Section 11.2), but it is also fixed-parameter tractable in k ([Sze04]), and fur-

[16] Expressed in terms of clause-variable matrices, stability of $\mathbb{C}$ corresponds to the property of a class of $\{-1, 0, +1\}$-matrices to be stable under formation of submatrices (i.e., under elimination of rows and columns). So "stability" is (the in our opinion essential) half of the conditions on a *semicentral* class of clause-sets in the sense of [Tru98] (Section 5.2).

thermore also finding a satisfying assignment resp. a tree resolution refutation is fixed-parameter tractable. On the other hand, the lean kernel problem is known to be decidable in polynomial time for fixed k (Theorem 4.2 in [Kul00a]), but it is not known whether this problem is also fixed-parameter tractable. Whether solving the non-trivial autarky problem is decidable in polynomial time is not known (only for $k = 1$ we know how to compute a quasi-maximal autarky, by first computing a quasi-maximal matching autarky, and then using that a matching-lean clause-set of deficiency 1 is minimally unsatisfiable iff it is unsatisfiable). The basic problem with the classes $\mathcal{CLS}(k)$ is that they are not stable under crossing out variables. Nothing is known here about the indecomposability problem.

Further complexity results regarding (the autarky system of) balanced autarkies (and unrestricted clause-sets) one finds in Subsection 11.12.2.

11.12. Connections to combinatorics

We need some good notions for some important properties of clause-sets in relation to standard notions from combinatorics: The *degree of a literal* l in a clause-set F is the number of clauses $C \in F$ with $l \in C$, while the *degree of a variable* v in a clause-set F is the sum of the degrees of the literals v and $\overline{v}$. Note that these degrees are the degrees of literals and variables in the *bipartite literal-clause graph* resp. the *bipartite variable-clause graph*, while the length of a clause is its degree in any of these two most basic graphs related to clause-sets. Regarding a whole clause-set, one needs to distinguish between *minimal literal-degree* and *maximal literal-degree* and between *minimal variable-degree* and *maximal variable-degree*. A clause-set F is called *literal-regular* if every literal in $\mathrm{var}(F) \cup \overline{\mathrm{var}(F)}$ has the same (literal-)degree, while F is called *variable-regular* if every variable in $\mathrm{var}(F)$ has the same (variable-)degree. Finally F is called *uniform* if all clauses of F have the same length.

A well-known field of activity relating SAT and combinatorics concerns the question how low the maximal variable-degree can be pushed in a k-uniform minimally unsatisfiable clause-set. These investigations started with [Tov84], with the most recent results in [HS05, HS06]. A related extremal problem is considered in [SZ08], where instead of the maximal variable-degree the number of edges in the *conflict graph* of a k-uniform minimally unsatisfiable clause-set is minimised, where the conflict graph of a clause-set F has the clauses of F as vertices, with an edge joining two vertices iff they have at least one conflict.[17]

Further differentiations are relevant here: Instead of considering arbitrary minimally unsatisfiable clause-sets, on the one hand we can restrict attention to hitting clause-sets (i.e., the conflict graph is a complete graph), and on the other hand we can require variable-regularity or, stronger, literal-regularity. Another regularity condition is *conflict-regularity*, where the conflict-multigraph (now allowing parallel edges) is required to be regular. First investigations regarding hitting clause-sets are found in [SS00, SST07], continued in [HK08]. Here we cannot go further into this interesting field, but we will only consider relations to combinatorics which are related to the notion of *deficiency*.

[17] Note that we take the notion of a "graph" in the strict sense, namely where an edge is a 2-subset of the vertex set, and thus loops or parallel edges are not possible.

Especially for hitting clause-sets (and generalisations) the field concerning biclique partitioning (and the hermitian rank) is of interest, relating the notion of deficiency to eigenvalues of graphs and quadratic forms; for first explorations of the connections to SAT see [Kul04a, GK05] (further extended in [Kul07a, HK08]). Again due to space constraints we cannot go further in this direction, but we will restrict our attention to relations which directly concern autarkies.

11.12.1. Autarkies and qualitative matrix analysis

"Qualitative matrix analysis" ("QMA"; see [BS95] for a textbook) is concerned with matrices over the real numbers, where matrices A, B are considered equivalent if they have the same sign matrices, i.e., $\mathrm{sgn}(A) = \mathrm{sgn}(B)$ (these are $\{-1, 0, +1\}$-matrices), and where questions from matrix analysis (starting with solvability of systems of linear equations) are considered "modulo" this equivalence relation.

As an example consider the notion of a non-singular (square) matrix A (over $\mathbb{R}$): A is non-singular iff $\det(A) \neq 0$, while A is *sign-non-singular* (short: an *SNS-matrix*) iff every matrix B with $\mathrm{sgn}(B) = \mathrm{sgn}(A)$ is non-singular (for example, $\left(\begin{smallmatrix} 1 & -1 \\ 1 & 1 \end{smallmatrix}\right)$ is an SNS-matrix, while $\left(\begin{smallmatrix} 1 & 2 \\ 1 & 1 \end{smallmatrix}\right)$ is not). We remark that from $\det(A) = \det(A^{\mathrm{t}})$ it follows that A is sign-non-singular iff its transposition A^{t} is sign-non-singular. In the subsequent Subsection 11.12.2 we will discuss the connections between SNS-matrices and SAT in more detail.

Motivated by [DD92], in [Kul03] the basic translations between SAT and QMA have been outlined; it seems justified to call QMA "SAT in disguise", but with a rather different point of view. Recall from Subsections 11.11.3, 11.11.4, that linear autarkies resp. balanced linear autarkies of a clause-set F correspond to the solutions of $M(F) \cdot \vec{x} \geq 0$ resp. $M(F) \cdot \vec{x} = 0$. Taking these systems of (in)equalities only "qualitatively", that is, the entries of $M(F)$ can be changed when only their signs are preserved, we arrive at exactly all autarkies resp. all balanced autarkies for F (restricted to $\mathrm{var}(F)$). This is the basic connection between SAT and QMA; more precisely, the details are as follows.

The class of all real matrices B with $\mathrm{sgn}(B) = \mathrm{sgn}(A)$ is denoted by $\mathfrak{Q}(A)$ (the "qualitative class of A"). Recall from Subsection 11.11.3 that linear autarkies φ for clause-sets F correspond to solutions of $M(F) \cdot \vec{x} \geq 0$, where $M(F)$ is the clause-variable matrix, via the association $\vec{x} \mapsto \varphi_{\vec{x}}$ for partial assignments with $\mathrm{var}(\varphi_{\vec{x}}) \subseteq \mathrm{var}(F)$ given by $\mathrm{sgn}_{\varphi_{\vec{x}}}(l) = \mathrm{sgn}(\vec{x}_{\mathrm{var}(l)})$ for literals l over F (that is, $l \in \mathrm{var}(F) \cup \overline{\mathrm{var}(F)}$). By the same translation from real vectors to partial assignments, (arbitrary) autarkies of F correspond to the solutions of $A \cdot \vec{x} \geq 0$ for $A \in \mathfrak{Q}(M(F))$ (all together), while balanced autarkies of F correspond to the solutions of $A \cdot \vec{x} = 0$ for $A \in \mathfrak{Q}(M(F))$. Furthermore, satisfying (partial) assignments for F correspond to the solutions of $A \cdot \vec{x} > 0$ for $A \in \mathfrak{Q}(M(F))$.[18]

Now the basic dictionary between SAT and QMA is as follows, where, corresponding to the convention in [BS95], clause-sets are represented by their variable-clause-matrix, the transposition of the clause-variable matrix (often characterisations in QMA are based on the characterisations of the above qualitative systems

[18] Where "$a > b$" for vectors a, b (of the same length) means that $a_i > b_i$ for all indices i.

$A \cdot \vec{x} \geq 0$, $A \cdot \vec{x} = 0$ and $A \cdot \vec{x} > 0$ via Gordan's transposition theorem (and its variations)):

- Balanced lean clause-sets correspond to *L-matrices* (matrices A such that all $B \in \mathfrak{Q}(A)$ have linearly independent rows, that is, from $\vec{y}^t \cdot B = 0$ follows $\vec{y} = 0$).
- Lean clause-sets correspond to *L^+-matrices* (matrices A such that for all $B \in \mathfrak{Q}(A)$ from $\vec{y}^t \cdot B \geq 0$ follows $\vec{y} = 0$).
- Unsatisfiable clause-sets correspond to *sign-central matrices* (matrices A such that for all $B \in \mathfrak{Q}(A)$ there is $\vec{x} \geq 0$, $\vec{x} \neq 0$ with $B \cdot x = 0$)[19], while minimally unsatisfiable clause-sets correspond to *minimal sign-central matrices* (sign-central matrices, where elimination of any column destroys sign-centrality).
- Minimally unsatisfiable clause-sets of deficiency 1 (i.e., the elements of MU(1)) correspond to *S-matrices* (matrices A with exactly one column more than rows, which are L-matrices, stay L-matrices after elimination of any column, and where there exists $\vec{x} > 0$ with $A \cdot \vec{x} = 0$). The saturated elements of MU(1) correspond to *maximal S-matrices* (where changing any 0 to ± 1 renders the matrix a non-S-matrix).
- Thus SNS-matrices are exactly the square L-matrices. And, as already mentioned in Subsection 11.11.4, from every $F \in \mathrm{MU}(1)$ we obtain an SNS-matrix by choosing $C \in F$ and considering the variable-clause matrix (or the clause-variable matrix) of $F \setminus \{C\}$.

For some more details on these correspondences see [Kul03]. The exploration of the connections between SAT and QMA has only been started yet, and interesting collaborations are to be expected in this direction.

11.12.2. Autarkies and the even cycle problem

Consider the following decision problems:

- Is a square hypergraph minimally non-2-colourable? (the "square critical hypergraph decision problem")
- Is a square matrix an SNS-matrix? (the "SNS-matrix decision problem")
- By multiplying some entries of a square matrix A by -1, can one reduce the computation of the permanent of A to the computation of the determinant of A? ("Polya's permanent problem")
- Does a directed graph contain a directed cycle of even length? (the "even-cycle decision problem")

All these problems are closely linked by direct translations (see [McC04] for a systematic account, containing many more equivalent problems[20]). Whether any (thus all) of these problems is solvable in polynomial time was a long-standing open problem, finally (positively) solved in [RST99, McC04]. Now by putting SAT in the centre, the equivalence of these problems becomes very instructive.

[19] the equivalence to unsatisfiability follows directly from Gordan's transposition theorem

[20] regarding the traditional combinatorial point of view, which unfortunately excludes SAT

11.12.2.1. The equivalence of the basic problems

We already mentioned hypergraph colouring in Subsections 11.4.5 and 11.11.5, introducing the deficiency $\mathrm{d}(G)$ of a hypergraph and the *reduced deficiency* $\mathrm{d_r}(F)$ of a complement-invariant clause-set (as $\mathrm{d_r}(F) := \mathrm{d}(\mathfrak{V}(F))$, where $\mathfrak{V}(F)$ is the variable-hypergraph of F). So every minimally unsatisfiable complement-invariant clause-set F has $\mathrm{d_r}(F) \geq 0$, and the square critical hypergraph decision problem is up to a different encoding the same as the problem of deciding whether a complement-invariant clause-set F with $\mathrm{d_r}(F) = 0$ is minimally unsatisfiable, but where F is a "PN-clause-set", that is, every clause is either positive or negative. Now the standard reduction of arbitrary clause-sets F to PN-clause-sets (introducing a new variable and a new binary clause for each "mixed" literal-occurrence), symmetrised by using the same variable for a clause C and its complement $\overline{C}$, translates complement-invariant clause-sets into complement-invariant PN-clause-sets while maintaining the reduced deficiency, and thus we see that by this simple transformation the square critical hypergraph decision problem is the same problem as to decide whether an (arbitrary) complement-invariant clause-sets of reduced deficiency 0 is minimally unsatisfiable.

By the results of Subsection 11.12.1, the SNS-matrix decision problem is (nearly) the same problem as to decide whether a clause-set F with $\mathrm{d}(F) = 0$ is balanced lean[21], which is the same problem as to decide whether a complement-invariant clause-set F with $\mathrm{d_r}(F) = 0$ is lean. So we need to find out for complement-invariant clause-set F with $\mathrm{d_r}(F) = 0$, what the precise relationship is between the problem of deciding leanness and deciding minimal unsatisfiability. Since a clause-set F is minimally balanced unsatisfiable iff $F \cup \overline{F}$ is minimally unsatisfiable, this is the same as the relationship between the problem of deciding balanced leanness and deciding minimal balanced unsatisfiability for clause-sets F_0 with $\mathrm{d}(F_0) = 0$,

For clause-sets F with deficiency $\mathrm{d}(F) = 1$, minimal unsatisfiability is equivalent to (that is, is implied by) being lean. Analogously one could expect for complement-invariant clause-sets F with $\mathrm{d_r}(F) = 0$, that leanness implies minimal unsatisfiability, or equivalently, that balanced leanness for F_0 with $\mathrm{d}(F_0) = 0$ implies minimal balanced unsatisfiability — however this is not the case, as the trivial examples $F = \{\{a\}, \{\overline{a}\}, \{b\}, \{\overline{b}\}\}$ resp. $F_0 = \{\{a\}, \{b\}\}$ shows (the reason is roughly that $1+1 = 2$ while $0+0 = 0$, and so a balanced lean F_0 with deficiency zero can contain a non-empty balanced lean strict sub-clause-set with deficiency zero). The first key here is Theorem 11.11.1, which yields that a clause-set F_0 is minimally balanced unsatisfiable iff F is barely balanced lean and balanced indecomposable. The second key is that for balanced lean F_0 with $\mathrm{d}(F_0) = 0$ the autarky decompositions (F_1, F_2) w.r.t. balanced autarkies are exactly the $\mathcal{A}_0$-autarky decompositions (F_1, F_2) with $\mathrm{d}(F_1) = \mathrm{d}(F_2) = 0$ of F_0 w.r.t. the smallest normal autarky system $\mathcal{A}_0$ (note that every clause-set is $\mathcal{A}_0$-lean; see Theorem 13, Part 3 in [Kul07b]). Now these $\mathcal{A}_0$-decompositions are the decompositions associated to the notion of a "partly decomposable matrix" (see Theorem 13, Part 4 in [Kul07b], and Subsection 4.2 in [BR91]), and thus can be found in polynomial time. So we can conclude now: Given on the one hand an algorithm for deciding

[21] there are trivial differences due to clause-sets not having multiple clause-occurrences

whether F_0 with $\mathrm{d}(F_0) = 0$ is balanced lean, to decide whether F_0 with $\mathrm{d}(F_0) = 0$ is minimally balanced unsatisfiable, we first check whether F_0 is balanced lean, and if this is the case then we check whether $M(F_0)$ is not partly decomposable (that is, whether $M(F_0)$ is "fully indecomposable") — F_0 is minimally balanced unsatisfiable iff both tests are passed. On the other hand, assume now that an algorithm is given for deciding whether F_0 with $\mathrm{d}(F_0) = 0$ is minimally balanced unsatisfiable. Here one needs to add the observation that if (F_1, F_2) is an $\mathcal{A}_0$-autarky decomposition of F, then F is $\mathcal{A}$-lean iff both F_1, F_2 are $\mathcal{A}$-lean (for any normal autarky system $\mathcal{A}$). So for given F_0 first we check whether F_0 is minimally balanced unsatisfiable — if this is the case then F_0 is balanced lean. Otherwise we search for an $\mathcal{A}_0$-decomposition (A_1, A_2) with $\mathrm{d}(A_1) = \mathrm{d}(A_2) = 0$ — if no such decomposition exists then F_0 is not balanced lean while otherwise F_0 is balanced lean iff both F_1, F_2 are balanced lean (applying the whole procedure recursively).

Considering the determinant computation via the Leibniz expansion, it is quite straightforward to show that a square matrix A is an SNS-matrix iff $\det(A) \neq 0$ and $\mathrm{per}(|A|) = |\det(A)|$ holds, where $|A|$ is obtained from A by taking absolute values entrywise, and $\mathrm{per}(A) = \sum_{\pi \in S_n} \prod_{i=1}^{n} A_{i,\pi(i)}$. This establishes the basic connection to Polya's permanent problem. We remark that for a clause-set F with $\mathrm{d}(F) = 0$ the number of matching satisfying assignments for F is $\mathrm{per}(|M(F)|)$. Finally, by considering the decomposition of permutations into cycles, it is also straightforward to show that a reflexive digraph G (every vertex has a loop) has no even (directed) cycle iff its adjacency matrix (a square $\{0,1\}$-matrix, with all entries on the main diagonal equal to 1 due to reflexivity of G) is an SNS-matrix.[22]

11.12.2.2. Generalisations and strengthenings

Let us now recapitulate the poly-time decision results in the light of the general algorithmic problems from Subsection 11.11.6. We introduced already the convention that for a class $\mathcal{C} \subseteq \mathcal{CLS}$ of clause-sets, $\mathcal{C}(k)$ is the class of $F \in \mathcal{C}$ with $\mathrm{d}^*(F) = k$ (note that for $F \in \mathrm{MU}$ we have $\mathrm{d}^*(F) = \mathrm{d}(F)$). In general, for some function $f : \mathcal{C} \to \mathbb{R}$ we write $\mathcal{C}_{f=k} := \{F \in \mathcal{C} : f(F) = k\}$, and similarly $\mathcal{C}_{f \leq k}$ and so on; so $\mathrm{MU}(k) = \mathrm{MU}_{\mathrm{d}=k}$. The class of complement-invariant clause-sets is denoted by $\mathrm{CI} := \{F \in \mathcal{CLS} : F = \overline{F}\}$, while the autarky system of balanced autarkies is denoted by the prefix "B".

Let us denote by $\mathrm{MUCI} := \mathrm{MU}(\mathrm{CI})$ the class of all complement-invariant minimally unsatisfiable clause-sets F, and so by $\mathrm{MUCI}_{\mathrm{d_r}=k}$ for $k \in \mathbb{N}_0$ the subclass of clause-sets of reduced deficiency k is denoted. We know

$$\mathrm{MUCI} = \bigcup_{k \in \mathbb{N}_0} \mathrm{MUCI}_{\mathrm{d_r}=k},$$

where $\mathrm{MUCI}_{\mathrm{d_r}=0}$ is decidable in polynomial time (as discussed in Subsection 11.12.2.1). Furthermore we denote by $\mathrm{BMU} := \text{B-MU}$ the class of minimally

[22] If we have given an arbitrary $\{0,1\}$-matrix A (possibly with zeros on the diagonal), and we want to reduce the SNS-decision problem for A to the even-cycle problem, then we use that if $\det(A) \neq 0$ then by row-permutation of A we can transform A into a matrix with 1-constant main diagonal (while this transformation does not alter the SNS-property).

balanced unsatisfiable clause-sets F, and thus by $\mathrm{BMU}(k)$ for $k \in \mathbb{N}_0$ the subclass given by $\mathrm{d}(F) = k$ is denoted. So

$$\mathrm{BMU} = \bigcup_{k \in \mathbb{N}_0} \mathrm{BMU}(k),$$

where $\mathrm{BMU}(0)$ is decidable in polynomial time (again, as discussed in Subsection 11.12.2.1). Note that BMU can be phrased as the class of "minimally not-all-equal-unsatisfiable clause-sets". And note that, as with $\mathrm{MU}(k)$, for $F \in \mathrm{BMU}$ we have $\mathrm{d}^*(F) = \mathrm{d}(F)$. Analogously to the $\mathrm{MU}(k)$-hierarchy, the following conjecture seems natural:

Conjecture 11.12.1. The MINIMAL UNSATISFIABILITY PROBLEM for classes of complement-invariant clause-sets with bounded reduced deficiency is solvable in polynomial time. That is, for every $k \in \mathbb{N}_0$ the class $\mathrm{MUCI}_{\mathrm{d_r}=k}$ is decidable in polynomial time. Equivalently, all $\mathrm{BMU}(k)$ for $k \in \mathbb{N}_0$ are decidable in polynomial time. Another equivalent formulation is that for all fixed $k \in \mathbb{N}_0$ the problem whether a hypergraph with deficiency k is minimally non-2-colourable is decidable in polynomial time.

Similarly we define LEANCI := LEAN(CI) as the class of complement-invariant lean clause-sets, while BLEAN := B-LEAN denotes the class of balanced lean clause-sets (recall that for $F \in \mathrm{BLEAN}$ we have $\mathrm{d}^*(F) = \mathrm{d}(F)$). So $\mathrm{LEANCI} = \bigcup_{k \in \mathbb{N}_0} \mathrm{LEANCI}_{\mathrm{d_r}=k}$ and $\mathrm{BLEAN} = \bigcup_{k \in \mathbb{N}_0} \mathrm{BLEAN}(k)$. As discussed in Subsection 11.12.2.1, for $k = 0$ in both cases we have poly-time decision.

Conjecture 11.12.2. The AUTARKY EXISTENCE PROBLEM for classes of complement-invariant clause-sets with bounded reduced deficiency is solvable in polynomial time. That is, for every $k \in \mathbb{N}_0$ the class $\mathrm{LEANCI}_{\mathrm{d_r}=k}$ is decidable in polynomial time. Equivalently, all $\mathrm{BLEAN}(k)$ for $k \in \mathbb{N}_0$ are decidable in polynomial time. Another equivalent formulation is that for all fixed $k \in \mathbb{N}_0$ the problem whether a matrix over $\{-1, 0, +1\}$ (or, equivalently, over $\{0, 1\}$) with exactly k more columns than rows is an L-matrix.

It needs to be investigated whether the reductions from Subsection 11.12.2.1 can be generalised, so that we obtain analogously the equivalence of Conjecture 11.12.2 to Conjecture 11.12.1. Finally we note that for $\mathrm{MU}(k)$ we actually have fixed-parameter tractability, and so we also conjecture the strengthening:

Conjecture 11.12.3. The decision problems in Conjecture 11.12.1 are not just poly-time decidable for each (fixed) $k \in \mathbb{N}_0$, but they are also fixed-parameter tractable in this parameter, i.e., there exists a time bound $f(\ell, k)$ for decision of MUCI depending on k and the number ℓ of literal occurrences in the input, which is of the form $f(\ell, k) = \alpha(k) \cdot \ell^\beta$ for some function α and some constant β.

The analogous problem to Conjecture 11.12.2, namely to decide $\mathrm{LEAN}(k)$, is not known to be fixed-parameter tractable, and so we refrain from conjecturing that also for the problems in Conjecture 11.12.2 we have fixed-parameter tractability.

11.12.2.3. Solving SAT problems by autarky reduction

While in the previous Subsection 11.12.2.2 we considered the basic decision problems as outlined in Subsection 11.11.6, we now turn to the functional problems as discussed there. Recall that for the classes $\mathcal{CLS}(k)$ we do not just have minimally unsatisfiability decision, but stronger satisfiability decision in polynomial time (even fixed-parameter tractability), and furthermore we can solve the lean-kernel problem in polynomial time (but currently it is not known whether the non-trivial autarky problem, or, here equivalent, the quasi-maximal autarky problem, can be solved in polynomial time). Different from the approaches used for $\mathcal{CLS}(k)$, which where satisfiability- or unsatisfiability-based, here now our approach is autarky-based, fundamentally relying on the following "self-reducibility conjecture".

Conjecture 11.12.4. Given a square non-SNS matrix A over $\{-1, 0, +1\}$, one can find in polynomial time a "witness" matrix $B \in \mathfrak{Q}(A)$ over $\mathbb{Z}$ (with the same sign pattern as A) which is singular.

There are two avenues for proving Conjecture 11.12.4: "constructivising" the proofs in [RST99, McC04], or refining the method from [KMT08] for finding an autarky when only given a decision procedure for leanness (the problem here is, similar to the classes $\mathcal{CLS}(k)$, that we do not have stability under crossing out variables nor under addition of unit-clauses). A witness according to Conjecture 11.12.4 yields a non-trivial balanced autarky for the clause-set corresponding to A. In order to make use of this, analogously to d^* we need to introduce the *maximal reduced deficiency* $\mathrm{d}^*_\mathrm{r}(F)$ for a complement-invariant clause-set F, defined as the maximum of $\mathrm{d}_\mathrm{r}(F')$ for complement-invariant $F' \subseteq F$; by definition we have $\mathrm{d}^*_\mathrm{r}(F) \geq 0$. Now [Kul07b] shows the following, using SATCI := SAT(CI) for the satisfiable complement-invariant clause-sets and BSAT := B-SAT for the balanced satisfiable clause-sets (i.e., not-all-equal satisfiable clause-sets).

Theorem 11.12.1. *Given Conjecture 11.12.4, we obtain poly-time decision of* $\mathrm{SATCI}_{\mathrm{d}^*_\mathrm{r}=0}$*, that is, SAT decision for complement-invariant clause-sets with maximal reduced deficiency* 0*. Equivalently, we obtain poly-time decision of* $\mathrm{BSAT}(0)$*, that is, NAESAT decision in poly-time for clause-sets* F *with* $\mathrm{d}^*(F) = 0$*.*

Stronger, we can compute a quasi-maximal autarky for complement-invariant clause-sets F *with* $\mathrm{d}^*_\mathrm{r}(F) = 0$*. And, equivalently, we can compute a quasi-maximal balanced autarky for clause-sets* F *with* $\mathrm{d}^*(F) = 0$*.*

Given the positive solution of Polya's problem, Theorem 11.12.1 (that is, the possibility of computing a quasi-maxima autarky) actually is equivalent to Conjecture 11.12.4. The natural generalisation of Conjecture 11.12.4 thus is as follows.

Conjecture 11.12.5. The QUASI-MAXIMAL AUTARKY PROBLEM for classes of complement-invariant clause-sets with bounded maximal reduced deficiency is solvable in polynomial time, that is, for every $k \in \mathbb{N}_0$ there exists a poly-time procedure for computing for complement-invariant clause-sets F with $\mathrm{d}^*_\mathrm{r}(F) = k$ a quasi-maximal autarky. Equivalently, quasi-maximal balanced autarkies can be computed in poly-time for $\mathrm{d}^*(F) = k$. Or, equivalently, there exists a polynomial-time procedure for deciding whether a matrix A over $\{-1, 0, +1\}$ with (exactly)

k more columns than rows is an L-matrix, and computing in the negative case a singular matrix $B \in \mathfrak{Q}(A)$ over $\mathbb{Z}$.

As discussed in general in Subsection 11.11.6, on the one hand Conjecture 11.12.5 implies poly-time decision for all $\mathrm{SATCI}_{\mathrm{d}_r^*=k}$, and, equivalently, poly-time decision of all $\mathrm{BSAT}(k)$ (that is, NAESAT decision for clause-sets with maximal deficiency k), and thus implies Conjecture 11.12.1. And on the other hand Conjecture 11.12.5 implies also Conjecture 11.12.2.

Finally we remark that by the process of "PN-separation", as discussed at the beginning of Subsection 11.12.2.1, every complement-invariant clause-set F can be transformed into a complement-invariant PN-clause-set F' (i.e., a hypergraph 2-colouring problem) such that F' is equivalent to F w.r.t. the properties of being satisfiable, minimally unsatisfiable and lean, and where $\mathrm{d}_r(F') = \mathrm{d}(F)$ and $\mathrm{d}_r^*(F') = \mathrm{d}_r^*(F)$. Actually it is easier to perform this transformation on one "core half" of F, a clause-set F_0 with $F = F_0 \cup \overline{F_0}$: then the properties being preserved by $F_0 \rightsquigarrow F_0'$ are balanced satisfiability, minimal balanced unsatisfiability and balanced leanness, and we have $\mathrm{d}(F_0') = \mathrm{d}(F_0)$ and $\mathrm{d}^*(F_0') = \mathrm{d}^*(F_0)$. Thus the various problems discussed in this subsection can be phrased as hypergraph-2-colouring problems, and the matrices occurring can be restricted to $\{0,1\}$-matrices.

11.13. Generalisations and extensions of autarkies

In Subsection 11.8.4 we have already commented on the (slight) generalisation given by the notion of a "weak autarky" (compared to (ordinary) autarkies). In this final section we will discuss more substantial generalisations, either concerning the autarky notion itself (in Subsections 11.13.1, 11.13.2), or concerning more general frameworks (in Subsections 11.13.3, 11.13.4).

11.13.1. Safe partial assignments

The most general case of "safe assignments" are *satisfiability-preserving partial assignments* φ, i.e., where $\varphi * F$ is satisfiability equivalent to F. This poses very mild restrictions on φ: if F is unsatisfiable, then every φ is satisfiability-preserving, while φ is satisfiability-preserving for satisfiable F iff there exists a satisfying (total) assignment extending φ. If φ is satisfiability-preserving for F, then also every $\varphi' \subseteq \varphi$ is satisfiability-preserving for F, and for satisfiable F the satisfiability-preserving partial assignments are exactly the $\varphi' \subseteq \varphi$ for the total satisfying assignments φ.

A satisfiability-preserving partial assignment amounts to specifying certain additional constraints which do not destroy satisfiability. The analogous notion of a *satisfiability-preserving clause* for a clause-set F is a clause C such that $F \cup \{C\}$ is satisfiability-equivalent to F. Examples for satisfiability-preserving clauses for F are clauses which follow logically from F. If C is satisfiability-preserving for F, then also every super-clause of C is satisfiability-preserving for F. Any C is a satisfiability-preserving clause for any unsatisfiable F, while C is satisfiability-preserving for a satisfiable F if and only if there exists $x \in C$ such that $\langle x \to 1 \rangle$

is satisfiability-preserving for F. So for a literal x the clause $\{x\}$ is satisfiability-preserving for F iff the partial assignment $\langle x \to 1\rangle$ is satisfiability-preserving for F, and satisfiability-preservation of clauses in general weakens these cases, while satisfiability-preservation of partial assignments strengthens them.

Lemma 11.13.1. *Consider a clause-set F and compatible partial assignments φ, ψ such that φ is satisfiability-preserving for $\psi * F$. Then for every literal $x \in C^1_\varphi$ the clause $C^0_\psi \cup \{x\}$ is satisfiability-preserving for F.*

Proof. If $\psi * F$ is unsatisfiable, then C^0_ψ follows from F, and thus C^0_ψ is satisfiability-preserving for F. If $\psi * F$ is satisfiable, then also $(\varphi \circ \psi) * F$ is satisfiable, thus $\varphi \circ \psi = \psi \circ \varphi$ is satisfiability-preserving for F, and so is $\langle x \to 1\rangle$. □

The composition of satisfiability-preserving partial assignments in general is not again satisfiability-preserving. The notion of a *safe partial assignment* provides a restriction having this property, where φ is safe for a clause-set F if for every partial assignment ψ with $\psi * F = \top$ we have $\psi * (\varphi * F) = \top$ as well (i.e., if ψ is a satisfying assignment, then so is $\psi \circ \varphi$). Every weak autarky is safe, for unsatisfiable clause-sets again every partial assignment is safe, and more generally every enforced partial assignment φ (i.e., every satisfying total assignment for F is an extension of φ) is safe. Every safe partial assignment is satisfiability-preserving. The safe partial assignments for F yield a monoid (with $\mathrm{Auk}(F)$ as sub-monoid), as is easily checked (using just the fundamental properties of the operation of $\mathcal{PASS}$ on $\mathcal{CLS}$).

11.13.2. Look-ahead autarky clauses, and blocked clauses

Recall the notion of a weak k-autarky φ for F from Subsection 11.10.5, defined by the condition that for the set $N(\varphi, F) := (\varphi * F) \setminus F$ of new clauses created by the application of φ to F we have $|N(\varphi, F)| \le k$. For any partial assignment ψ with $\psi * N = \top$ and $\mathrm{var}(\psi) \subseteq \mathrm{var}(N)$ and for every $x' \in C^1_\varphi$ then by Lemma 11.13.1 the *look-ahead autarky clause* $C^0_\psi \cup \{x'\}$ is satisfiability-preserving for F. In the special case of $\varphi = \varphi_x$, as discussed in Subsection 11.10.5, the choice $x' = x$ is appropriate. For a look-ahead autarky clause we actually can allow a slightly more general case, just given arbitrary partial assignments ψ, φ with $\mathrm{var}(\varphi) \cap \mathrm{var}(\psi) = \emptyset$ such that φ is a weak autarky for $\psi * F$: In this case ψ must satisfy all clauses of $N(\varphi, F)$ except of those which after application of ψ are already contained in $\psi * F$.

Recall the notion of a *blocked clause* for clause-set F w.r.t. $x \in C$ (introduced in [Kul99b], and further studied in [Kul99c]), that is, for every clause D in F with $\overline{x} \in D$ there is $y_D \in C \setminus \{x\}$ with $\overline{y_D} \in D$. Let $C_0 := C \setminus \{x\}$, and consider the partial assignment $\varphi_{C_0} := \langle y \to 0 : y \in C_0\rangle$. Now the blocking-condition is equivalent to the literal x being pure for $\varphi_{C_0} * F$, and thus C is a look-ahead autarky clause for F. So look-ahead autarky clauses generalise blocked clauses, where for the latter it is shown in [Kul99c] that their addition to clause-sets cannot be simulated by (full) resolution (even if we only use C without new variables).

Thus addition of look-ahead autarky clauses by a SAT solver could be an interesting feature; though one would expect that the full power of extended

resolution is only unleashed by allowing blocked clauses with new variables, nevertheless the special case of look-ahead autarky clauses considered above, given by some partial assignment φ_x we naturally have "at hand" in a SAT solver, could become interesting in the future. In the `OKsolver`-2002 the special case of a weak 1-autarky has been implemented as an experimental feature: if $C = \{a_1, \ldots, a_k\}$ ($k \geq 2$) is the (single) new clause, then for $1 \leq i \leq k$ the k binary clauses $\{\overline{a_i}, x\}$ can be added.

Finally we should remark that addition of look-ahead autarky clauses is a form of "local learning", that is, the added clauses are valid only at the current ("residual") node in the backtracking tree, and need to be removed when backtracking.

11.13.3. Beyond clause-sets

Autarkies for generalised clause-sets, using non-boolean variables v and literals "$v \neq \varepsilon$" for (single) values ε in the domain of v, are studied in [Kul07a], and some first applications to hypergraph colouring one finds in [Kul06]. As we have seen in Section 11.12.2, the hypergraph 2-colouring problem is captured by considering complement-invariant clause-sets, which can be reduced to their core halfs, that is, to the underlying hypergraphs, by considering balanced autarkies. Now for k-colouring, variables with uniform domain $\mathbb{Z}_k = \{0, \ldots, k-1\}$ are considered, and *weak hypergraph-colouring* (no hyperedge is monochromatic) is captured by *weakly-balanced autarkies*, which are autarkies where every touched clause contains at least two different assigned values, while *strong hypergraph-colouring* (every hyperedge contains all colours) is captured by *strongly-balanced autarkies*, autarkies where every touched clause contains all possible k assigned values.

Another route of generalisation is to consider boolean functions (in some representation, for example BDD's as in [FKS$^+$04]) instead of clauses, and, more generally, arbitrary sets of constraints. Just replacing "clause" by "constraint" in the basic condition "every touched clause is satisfied", and allowing partial assignments for constraints (satisfying the constraint iff all total extensions satisfy it), we obtain a straight-forward generalisation.

Regarding QBF, a powerful framework is here to consider QCNF-problems as satisfiability problems, that is the universal variables are "eliminated" while we seek substitutions for the existential variables by boolean functions in the preceeding universal variables such that a tautology is created (that is, every clause becomes a tautology). Now an autarky in this setting is a partial substitution of existential variables by boolean functions (depending on the universal variables on which the existential variable depends) such that the substitution either does not touch a clause or makes it a tautology. The simplest case is to consider only constant functions, that is we search for standard CNF-autarkies of the matrix where all universal variables are crossed out.

11.13.4. An axiomatic theory

A first approach towards an "axiomatic theory", extending [Kul04b], is given in [Kul01], (in principle) comprising all extensions mentioned in Subsections 11.13.1

and 11.13.3. The basic idea is to consider the operation $*$ of a monoid $(\mathcal{I}, \circ, e)$ of "instantiators" (like partial assignments) on a lower semilattice $(\mathcal{P}, \wedge, \top)$ of "problem instances" (like clause-sets with union). We obtain the induced partial order $P_1 \leq P_2$ for problem instances given by $P_1 \leq P_2 :\Leftrightarrow P_1 \wedge P_2 = P_1$. Now *weak autarkies* for problem instances $P \in \mathcal{P}$ are instantiators $\varphi \in \mathcal{I}$ with $\varphi * P \leq P$, while *autarkies* fulfil the condition $\varphi * P' \leq P'$ for all $P' \leq P$. Given the multitude of notions of autarkies, such a general theory should become indispensable at some time in the future.

11.14. Conclusion

An important structural parameter for satisfiability problems about clause-sets (CNF-formulas) and QBF-formulas is the (maximal) deficiency. Various hierarchies based on the notion of (maximal) deficiency emerged, with some poly-time results established, and most open:

1. For deciding minimally unsatisfiability see Section 11.2, while more generally in Subsection 11.11.6 satisfiability decision (and more) is discussed, and a general framework is given (with still open problems in this context).
2. Deciding the tautology/falsity property for quantified boolean formulas is discussed in Section 11.6.
3. "Not-all-equal satisfiability" (with the normal deficiency), or, equivalently, complement-invariant clause-sets with the reduced deficiency, is considered in Section 11.12.2 (see especially Subsection 11.12.2.3 for open problems and conjectures).

References

[AL86] R. Aharoni and N. Linial. Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. *Journal of Combinatorial Theory*, A 43:196–204, 1986.

[BG06] A. Biere and C. P. Gomes, editors. *Theory and Applications of Satisfiability Testing - SAT 2006*, volume 4121 of *Lecture Notes in Computer Science*. Springer, 2006. ISBN 3-540-37206-7.

[BK92] A. Bachem and W. Kern. *Linear Programming Duality: An Introduction to Oriented Matroids*. Universitext. Springer-Verlag, Berlin, 1992. ISBN 3-540-55417-3.

[Bou89] N. Bourbaki. *Algebra I: Chapters 1-3*. Elements of Mathematics. Springer-Verlag, Berlin, 1989. Second printing, ISBN 3-540-19373-1.

[BR91] R. A. Brualdi and H. J. Ryser. *Combinatorial Matrix Theory*, volume 39 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1991. ISBN 0-521-32265-0.

[Bru03] R. Bruni. Approximating minimal unsatisfiable subformulae by means of adaptive core search. *Discrete Applied Mathematics*, 130:85–100, 2003.

[Bru05] R. Bruni. On exact selection of minimally unsatisfiable subformulae. *Annals for Mathematics and Artificial Intelligence*, 43:35–50, 2005.

[BS95] R. A. Brualdi and B. L. Shader. *Matrices of sign-solvable linear systems*, volume 116 of *Cambridge Tracts in Mathematics*. Cambridge University Press, 1995. ISBN 0-521-48296-8.

[BS01] R. Bruni and A. Sassano. Restoring satisfiability or maintaining unsatisfiability by finding small unsatisfiable subformulae. In H. Kautz and B. Selman, editors, *LICS 2001 Workshop on Theory and Applications of Satisfiability Testing (SAT 2001)*, volume 9 of *Electronic Notes in Discrete Mathematics (ENDM)*. Elsevier Science, June 2001.

[BW05] F. Bacchus and T. Walsh, editors. *Theory and Applications of Satisfiability Testing 2005*, volume 3569 of *Lecture Notes in Computer Science*, Berlin, 2005. Springer. ISBN 3-540-26276-8.

[DD92] G. Davydov and I. Davydova. Tautologies and positive solvability of linear homogeneous systems. *Annals of Pure and Applied Logic*, 57:27–43, 1992.

[DDKB98] G. Davydov, I. Davydova, and H. Kleine Büning. An efficient algorithm for the minimal unsatisfiability problem for a subclass of CNF. *Annals of Mathematics and Artificial Intelligence*, 23:229–245, 1998.

[DE92] M. Dalal and D. W. Etherington. A hierarchy of tractable satisfiability problems. *Information Processing Letters*, 44:173–180, 1992.

[EIS76] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal Computing*, 5(4):691–703, 1976.

[FKS02] H. Fleischner, O. Kullmann, and S. Szeider. Polynomial–time recognition of minimal unsatisfiable formulas with fixed clause–variable difference. *Theoretical Computer Science*, 289(1):503–516, November 2002.

[FKS+04] J. Franco, M. Kouril, J. Schlipf, S. Weaver, M. Dransfield, and W. M. Vanfleet. Function-complete lookahead in support of efficient SAT search heuristics. *Journal of Universal Computer Science*, 10(12):1655–1692, 2004.

[FSW06] M. R. Fellows, S. Szeider, and G. Wrightson. On finding short resolution refutations and small unsatisfiable subsets. *Theoretical Computer Science*, 351(3):351–359, 2006.

[FV03] J. Franco and A. Van Gelder. A perspective on certain polynomial-time solvable classes of satisfiability. *Discrete Applied Mathematics*, 125:177–214, 2003.

[GK05] N. Galesi and O. Kullmann. Polynomial time SAT decision, hypergraph transversals and the hermitian rank. In H. H. Hoos and D. G. Mitchell, editors, *Theory and Applications of Satisfiability Testing 2004*, volume 3542 of *Lecture Notes in Computer Science*, pages 89–104, Berlin, 2005. Springer. ISBN 3-540-27829-X.

[GMP06] É. Grégoire, B. Mazure, and C. Piette. Tracking MUSes and strict inconsistent covers. In *Proceedings of the Formal Methods in Computer Aided Design (FMCAD)*, pages 39–46. IEEE Computer Society, 2006.

[GMP07a] É. Grégoire, B. Mazure, and C. Piette. Boosting a complete technique to find MSS and MUS thanks to a local search oracle. In M. M. Veloso,

editor, *Proceedings of IJCAI*, pages 2300–2305, 2007.

[GMP07b] É. Grégoire, B. Mazure, and C. Piette. MUST: Provide a finer-grained explanation of unsatisfiability. In *Principles and Practice of Constraint Programming (CP)*, pages 317–331, 2007.

[GT04] E. Giunchiglia and A. Tacchella, editors. *Theory and Applications of Satisfiability Testing 2003*, volume 2919 of *Lecture Notes in Computer Science*, Berlin, 2004. Springer. ISBN 3-540-20851-8.

[Hir00] E. Hirsch. New worst-case upper bounds for SAT. *Journal of Automated Reasoning*, 24(4):397–420, 2000.

[HK08] M. Henderson and O. Kullmann. Multiclique partitions of multigraphs, and conflict-regular satisfiability problems with non-boolean variables. In preparation, August 2008.

[HS05] S. Hoory and S. Szeider. Computing unsatisfiable k-SAT instances with few occurrences per variable. *Theoretical Computer Science*, 337:347–359, 2005.

[HS06] S. Hoory and S. Szeider. A note on unsatisfiable k-CNF formulas with few occurrences per variable. *SIAM Journal on Discrete Mathematics*, 20(2):523–528, 2006.

[Hv08] M. J. Heule and H. van Maaren. Parallel SAT solving using bit-level operations. *Journal on Satisfiability, Boolean Modeling and Computation*, 4:99–116, 2008.

[IM95] K. Iwama and E. Miyano. Intractability of read-once resolution. In *Proceedings Structure in Complexity Theory, Tenth Annual Conference*, pages 29–36. IEEE, 1995.

[Iwa89] K. Iwama. CNF satisfiability test by counting and polynomial average time. *SIAM Journal on Computing*, 18(2):385–391, April 1989.

[JT95] T. R. Jensen and B. Toft. *Graph Coloring Problems*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, 1995. ISBN 0-471-02865-7.

[KB99] H. Kleine Büning. An upper bound for minimal resolution refutations. In G. Gottlob, E. Grandjean, and K. Seyr, editors, *Computer Science Logic 12th International Workshop, CSL'98*, volume 1584 of *Lecture Notes in Computer Science*, pages 171–178. Springer, 1999.

[KB00] H. Kleine Büning. On subclasses of minimal unsatisfiable formulas. *Discrete Applied Mathematics*, 107:83–98, 2000.

[KBL99] H. Kleine Büning and T. Lettmann. *Propositional Logic: Deduction and Algorithms*. Cambridge University Press, 1999.

[KBZ02] H. Kleine Büning and X. Zhao. Minimal unsatisfiability: Results and open questions. Technical Report tr-ri-02-230, Series Computer Science, University of Paderborn, University of Paderborn, Department of Mathematics and Computer Science, 2002. `wwwcs.uni-paderborn.de/cs/ag-klbue/de/research/MinUnsat/`.

[KL97] O. Kullmann and H. Luckhardt. Deciding propositional tautologies: Algorithms and their complexity. Preprint, 82 pages; the ps-file can be obtained at `http://cs.swan.ac.uk/~csoliver/`, January 1997.

[KL98] O. Kullmann and H. Luckhardt. Algorithms for SAT/TAUT decision based on various measures. Preprint, 71 pages; the ps-file can be

obtained from http://cs.swan.ac.uk/~csoliver/, December 1998.

[KLMS06] O. Kullmann, I. Lynce, and J. P. Marques-Silva. Categorisation of clauses in conjunctive normal forms: Minimally unsatisfiable subclause-sets and the lean kernel. In Biere and Gomes [BG06], pages 22–35. ISBN 3-540-37206-7.

[KMT08] O. Kullmann, V. W. Marek, and M. Truszczyński. Computing autarkies and properties of the autarky monoid. In preparation, October 2008.

[Kul98] O. Kullmann. Heuristics for SAT algorithms: A systematic study. In *SAT'98*, March 1998. Extended abstract for the Second workshop on the satisfiability problem, May 10 - 14, 1998, Eringerfeld, Germany.

[Kul99a] O. Kullmann. Investigating a general hierarchy of polynomially decidable classes of CNF's based on short tree-like resolution proofs. Technical Report TR99-041, Electronic Colloquium on Computational Complexity (ECCC), October 1999.

[Kul99b] O. Kullmann. New methods for 3-SAT decision and worst-case analysis. *Theoretical Computer Science*, 223(1-2):1–72, July 1999.

[Kul99c] O. Kullmann. On a generalization of extended resolution. *Discrete Applied Mathematics*, 96-97(1-3):149–176, 1999.

[Kul00a] O. Kullmann. An application of matroid theory to the SAT problem. In *Fifteenth Annual IEEE Conference on Computational Complexity (2000)*, pages 116–124. IEEE Computer Society, July 2000.

[Kul00b] O. Kullmann. Investigations on autark assignments. *Discrete Applied Mathematics*, 107:99–137, 2000.

[Kul01] O. Kullmann. On the use of autarkies for satisfiability decision. In H. Kautz and B. Selman, editors, *LICS 2001 Workshop on Theory and Applications of Satisfiability Testing (SAT 2001)*, volume 9 of *Electronic Notes in Discrete Mathematics (ENDM)*. Elsevier Science, June 2001.

[Kul02] O. Kullmann. Investigating the behaviour of a SAT solver on random formulas. Technical Report CSR 23-2002, Swansea University, Computer Science Report Series (available from http://www-compsci.swan.ac.uk/reports/2002.html), October 2002.

[Kul03] O. Kullmann. Lean clause-sets: Generalizations of minimally unsatisfiable clause-sets. *Discrete Applied Mathematics*, 130:209–249, 2003.

[Kul04a] O. Kullmann. The combinatorics of conflicts between clauses. In Giunchiglia and Tacchella [GT04], pages 426–440. ISBN 3-540-20851-8.

[Kul04b] O. Kullmann. Upper and lower bounds on the complexity of generalised resolution and generalised constraint satisfaction problems. *Annals of Mathematics and Artificial Intelligence*, 40(3-4):303–352, March 2004.

[Kul06] O. Kullmann. Constraint satisfaction problems in clausal form: Autarkies, minimal unsatisfiability, and applications to hypergraph inequalities. In N. Creignou, P. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, number 06401 in Dagstuhl Sem-

inar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006. http://drops.dagstuhl.de/opus/volltexte/2006/803.

[Kul07a] O. Kullmann. Constraint satisfaction problems in clausal form: Autarkies and minimal unsatisfiability. Technical Report TR 07-055, Electronic Colloquium on Computational Complexity (ECCC), June 2007.

[Kul07b] O. Kullmann. Polynomial time SAT decision for complementation-invariant clause-sets, and sign-non-singular matrices. In J. P. Marques-Silva and K. A. Sakallah, editors, *Theory and Applications of Satisfiability Testing - SAT 2007*, volume 4501 of *Lecture Notes in Computer Science*, pages 314–327. Springer, 2007. ISBN 978-3-540-72787-3.

[KX05] H. Kleine Büning and D. Xu. The complexity of homomorphisms and renamings for minimal unsatisfiable formulas. *Annals of Mathematics and Artificial Intelligence*, 43(1-4):113–127, 2005.

[KZ02a] H. Kleine Büning and X. Zhao. The complexity of read-once resolution. *Annals of Mathematics and Artificial Intelligence*, 36(4):419–435, 2002.

[KZ02b] H. Kleine Büning and X. Zhao. Polynomial time algorithms for computing a representation for minimal unsatisfiable formulas with fixed deficiency. *Information Processing Letters*, 84(3):147–151, November 2002.

[KZ03] H. Kleine Büning and X. Zhao. On the structure of some classes of minimal unsatisfiable formulas. *Discrete Applied Mathematics*, 130:185–207, 2003.

[KZ05] H. Kleine Büning and X. Zhao. Extension and equivalence problems for clause minimal formulae. *Annals of Mathematics and Artificial Intelligence*, 43:295–306, 2005.

[KZ06] H. Kleine Büning and X. Zhao. Minimal false quantified boolean formulas. In Biere and Gomes [BG06], pages 339–352. ISBN 3-540-37206-7.

[KZ07a] H. Kleine Büning and X. Zhao. The complexity of some subclasses of minimal unsatisfiable formulas. *Journal on Satisfiability, Boolean Modeling and Computation*, 3:1–17, 2007.

[KZ07b] H. Kleine Büning and X. Zhao. An extension of deficiency and minimal unsatisfiability of quantified boolean formulas. *Journal on Satisfiability, Boolean Modeling and Computation*, 3:115–123, 2007.

[KZ08a] H. Kleine Büning and X. Zhao. Computational complexity of quantified Boolean formulas with fixed maximal deficiency. *Theoretical Computer Science*, 407(1-3):448–457, 2008.

[KZ08b] H. Kleine Büning and X. Zhao, editors. *Theory and Applications of Satisfiability Testing - SAT 2008*, volume 4996 of *Lecture Notes in Computer Science*. Springer, 2008. ISBN 978-3-540-79718-0.

[Lan02] S. Lang. *Algebra*, volume 211 of *Graduate Texts in Mathematics*. Springer, New York, third edition, 2002. ISBN 0-387-95385-X; QA154.3.L3 2002.

[Lau06] D. Lau. *Function Algebras on Finite Sets: A Basic Course on Many-Valued Logic and Clone Theory*. Springer Monographs in Mathematics. Springer, 2006. ISBN 3-540-36022-0.

[LS05] M. H. Liffiton and K. A. Sakallah. On finding all minimally unsatisfiable subformulas. In Bacchus and Walsh [BW05], pages 173–186. ISBN 3-540-26276-8.

[LS08] M. Liffiton and K. Sakallah. Searching for autarkies to trim unsatisfiable clause sets. In Kleine Büning and Zhao [KZ08b], pages 182–195. ISBN 978-3-540-79718-0.

[Luc84] H. Luckhardt. Obere Komplexitätsschranken für TAUT-Entscheidungen. In *Frege Conference 1984, Schwerin*, pages 331–337. Akademie-Verlag Berlin, 1984.

[McC04] W. McCuaig. Pólya's permanent problem. *The Electronic Journal of Combinatorics*, 11, 2004. #R79, 83 pages.

[MLA+05] M. N. Mneimneh, I. Lynce, Z. S. Andraus, J. P. Marques-Silva, and K. A. Sakallah. A branch and bound algorithm for extracting smallest minimal unsatisfiable formulas. In Bacchus and Walsh [BW05], pages 467–474. ISBN 3-540-26276-8.

[Mol05] M. Molloy. Cores in random hypergraphs and boolean formulas. *Random Structures and Algorithms*, 27(1):124–135, 2005.

[MS85] B. Monien and E. Speckenmeyer. Solving satisfiability in less than 2^n steps. *Discrete Applied Mathematics*, 10:287–295, 1985.

[Oku00] F. Okushi. Parallel cooperative propositional theorem proving. *Annals of Mathematics and Artificial Intelligence*, 26:59–85, 2000.

[OMA+04] Y. Oh, M. N. Mneimneh, Z. S. Andraus, K. A. Sakallah, and I. L. Markov. AMUSE: a minimally-unsatisfiable subformula extractor. In *Proceedings of the 41st annual conference on Design automation (DAC)*, pages 518–523, 2004.

[Pie91] B. C. Pierce. *Basic Category Theory for Computer Scientists*. Foundations of Computing. The MIT Press, 1991. ISBN 0-262-66071-7.

[PW88] C. H. Papadimitriou and D. Wolfe. The complexity of facets resolved. *Journal of Computer and System Sciences*, 37:2–13, 1988.

[RST99] N. Robertson, P. D. Seymour, and R. Thomas. Permanents, Pfaffian orientations, and even directed circuits. *Annals of Mathematics*, 150:929–975, 1999.

[SD97] M. Shaohan and L. Dongmin. A polynomial-time algorithm for reducing the number of variables in MAX SAT problem. *Science in China (Series E)*, 40(3):301–311, June 1997.

[Sey74] P. D. Seymour. On the two-colouring of hypergraphs. *The Quarterly Journal of Mathematics (Oxford University Press)*, 25:303–312, 1974.

[SS00] P. Savický and J. Sgall. DNF tautologies with a limited number of occurrences of every variable. *Theoretical Computer Science*, 238:495–498, 2000.

[SST07] R. H. Sloan, B. Sörényi, and G. Turán. On k-term DNF with the largest number of prime implicants. *SIAM Journal on Discrete Mathematics*, 21(4):987–998, 2007.

[SZ08] D. Scheder and P. Zumstein. How many conflicts does it need to be

unsatisfiable? In Kleine Büning and Zhao [KZ08b], pages 246–256. ISBN 978-3-540-79718-0.

[Sze01] S. Szeider. NP-completeness of refutability by literal-once resolution. In R. Gore, A. Leitsch, and T. Nipkow, editors, *IJCAR 2001, Proceedings of the International Joint Conference on Automated Reasoning*, volume 2083 of *Lecture Notes in Artificial Intelligence*, pages 168–181. Springer-Verlag, 2001.

[Sze03] S. Szeider. Homomorphisms of conjunctive normal forms. *Discrete Applied Mathematics*, 130(2):351–365, 2003.

[Sze04] S. Szeider. Minimal unsatisfiable formulas with bounded clause-variable difference are fixed-parameter tractable. *Journal of Computer and System Sciences*, 69(4):656–674, 2004.

[Sze05] S. Szeider. Generalizations of matched CNF formulas. *Annals of Mathematics and Artificial Intelligence*, 43(1-4):223–238, 2005.

[Tov84] C. A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8:85–89, 1984.

[Tru98] K. Truemper. *Effective Logic Computation*. A Wiley-Interscience Publication, 1998. ISBN 0-471-23886-4.

[Van99] A. Van Gelder. Autarky pruning in propositional model elimination reduces failure redundancy. *Journal of Automated Reasoning*, 23(2):137–193, 1999.

[van00] H. van Maaren. A short note on some tractable cases of the satisfiability problem. *Information and Computation*, 158(2):125–130, May 2000.

[VO99] A. Van Gelder and F. Okushi. A propositional theorem prover to solve planning and other problems. *Annals of Mathematics and Artificial Intelligence*, 26:87–112, 1999.

[vW00] H. van Maaren and J. Warners. Solving satisfiability problems using elliptic approximations — effective branching rules. *Discrete Applied Mathematics*, 107:241–259, 2000.

[vW08] H. van Maaren and S. Wieringa. Finding guaranteed MUSes fast. In Kleine Büning and Zhao [KZ08b], pages 291–304. ISBN 978-3-540-79718-0.

[ZM04] L. Zhang and S. Malik. Extracting small unsatisfiable cores from unsatisfiable Boolean formula. In Giunchiglia and Tacchella [GT04], pages 239–249. ISBN 3-540-20851-8.