CSPs in clausal form: Autarkies, minimal unsatisfiability, hypergraphs

Oliver Kullmann
Computer Science Department
University of Wales Swansea

Dagstuhl 2006
October 4, 2006

Generalised SAT problems:
A link to combinatorics

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (http://www-compsci.swan.ac.uk/reports/2006.html), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Overview

1. Generalised clause-sets
2. Matching structure
3. The boolean translation
4. Minimally unsatisfiable clause-sets
5. Hypergraph inequalities
6. Colouring clause-sets
7. Conclusions

See University of Wales Swansea Computer Science Report Series (`http://www-compsci.swan.ac.uk/reports/2006.html`), CSR 13-2006

Good generalisations of CNFs

The first task is to find

a generalisation of boolean CNFs,
allowing non-boolean variables,
but keeping the good properties.

Motivation: **Colouring problems** ($>$ 2 colours).

"Good":

We are interested in for example studying

combinatorial properties of colouring problems,

and thus we are (mainly) interested in the good

combinatorial properties of boolean CNFs.

Good generalisations of CNFs

The first task is to find

a generalisation of boolean CNFs,
allowing non-boolean variables,
but keeping the good properties.

Motivation: **Colouring problems** ($>$ 2 colours).

"Good":

We are interested in for example studying

combinatorial properties of colouring problems,

and thus we are (mainly) interested in the good

combinatorial properties of boolean CNFs.

Variables and signed literals

- A variable v has a **domain** D_v (non-empty, and finite for us).
- **Boolean variables** have domain $\{0, 1\}$.
- A **signed literal** is of the form "$v \in S$" for some $\emptyset \neq S \subseteq D_V$. (Such literals for logical formulas (like CNF and DNF) have been studied in CSP and AI for more than a decade.)

Signed literals seems to be too general to obtain interesting combinatorial properties (at this time).

For us literals are "negative monosigned literals", where $S = D_v \setminus \{\varepsilon\}$ for some $\varepsilon \in D_v$; i.e.

$$\text{“}v \neq \varepsilon\text{”}.$$

This literal x has $\mathrm{var}(x) = v$ and $\mathrm{val}(x) = \varepsilon$.

Variables and signed literals

- A variable v has a **domain** D_v (non-empty, and finite for us).
- **Boolean variables** have domain $\{0, 1\}$.
- A **signed literal** is of the form "$v \in S$" for some $\emptyset \neq S \subseteq D_V$. (Such literals for logical formulas (like CNF and DNF) have been studied in CSP and AI for more than a decade.)

Signed literals seems to be too general to obtain interesting combinatorial properties (at this time).

For us literals are "negative monosigned literals", where $S = D_v \setminus \{\varepsilon\}$ for some $\varepsilon \in D_v$; i.e.

$$\text{“}v \neq \varepsilon\text{”}.$$

This literal x has $\mathrm{var}(x) = v$ and $\mathrm{val}(x) = \varepsilon$.

Clauses, clause-sets, partial assignments

1. Two literals x, y **clash** if $\mathrm{var}(x) = \mathrm{var}(y)$ and $\mathrm{val}(x) \neq \mathrm{val}(y)$.
2. A **clause** is a finite set of non-clashing literals.
3. A **clause-set** is a finite set of clauses.

A **partial assignment** assigns to some variables a value of their respective domains.

Fundamental the 1-1 correspondence between clauses and partial assignments:

Clauses C are interpreted as **no-goods** C_φ of a partial assignment φ (the clause is the disjunction of the falsified literals for all the variables in the domain of φ).

Autarkies as generalised satisfied assignments

A partial assignment φ is an **autarky** for a clause-set F if every clause of F touched by φ is satisfied by φ.

Clauses satisfied by an autarky can be removed satisfiability-equivalently (for that purpose the notion of "autarky" was introduced in 1984 by Monien and Speckenmeyer).

Autarkies have many nice properties. For example reduction by autarkies is confluent.

The result for F of this reduction process is the **lean kernel** $N_a(F)$.

A clause-set F is **lean** if it has no autarkies assigning a value to at least one variable of F.

Autarkies as generalised satisfied assignments

A partial assignment φ is an **autarky** for a clause-set F if every clause of F touched by φ is satisfied by φ.

Clauses satisfied by an autarky can be removed satisfiability-equivalently (for that purpose the notion of "autarky" was introduced in 1984 by Monien and Speckenmeyer).

Autarkies have many nice properties. For example reduction by autarkies is confluent.

The result for F of this reduction process is the **lean kernel** $N_a(F)$.

A clause-set F is **lean** if it has no autarkies assigning a value to at least one variable of F.

Autarkies as generalised satisfied assignments

A partial assignment φ is an **autarky** for a clause-set F if every clause of F touched by φ is satisfied by φ.

Clauses satisfied by an autarky can be removed satisfiability-equivalently (for that purpose the notion of "autarky" was introduced in 1984 by Monien and Speckenmeyer).

Autarkies have many nice properties. For example reduction by autarkies is confluent.

The result for F of this reduction process is the **lean kernel** $N_a(F)$.

A clause-set F is **lean** if it has no autarkies assigning a value to at least one variable of F.

Autarkies as generalised satisfied assignments

A partial assignment φ is an **autarky** for a clause-set F if every clause of F touched by φ is satisfied by φ.

Clauses satisfied by an autarky can be removed satisfiability-equivalently (for that purpose the notion of "autarky" was introduced in 1984 by Monien and Speckenmeyer).

Autarkies have many nice properties. For example reduction by autarkies is confluent.

The result for F of this reduction process is the **lean kernel** $N_a(F)$.

A clause-set F is **lean** if it has no autarkies assigning a value to at least one variable of F.

Autarkies as generalised satisfied assignments

A partial assignment φ is an **autarky** for a clause-set F if every clause of F touched by φ is satisfied by φ.

Clauses satisfied by an autarky can be removed satisfiability-equivalently (for that purpose the notion of "autarky" was introduced in 1984 by Monien and Speckenmeyer).

Autarkies have many nice properties. For example reduction by autarkies is confluent.

The result for F of this reduction process is the **lean kernel** $\mathrm{N_a}(F)$.

A clause-set F is **lean** if it has no autarkies assigning a value to at least one variable of F.

Restricted versions of autarkies

The class of lean clause-sets is co-NP complete (also for 2-clause-sets, since we can have more than 2 values).

Autarky systems allow for poly-time manageable special cases, while maintaining the good properties of general autarkies.

Examples of autarky systems:

1. **pure autarkies** (given by variables with "missing values")
2. **matching autarkies** (discussed later)
3. **linear autarkies** (closely related to linear programming in the boolean case).

Splitting trees and resolution

Deciding SAT via backtracking (for a variable v we have m branches $v \to \varepsilon_1, \ldots, v \to \varepsilon_m$ for $D_v = \{\varepsilon_1, \ldots, \varepsilon_m\}$) yields splitting trees.

Via the correspondence of partial assignments and clauses, all the paths from the root to some leaf in such a splitting tree correspond to clauses.

Splitting trees thus give us tree-like resolution, and via counting only different clauses also full resolution.

We have developed a theory for this (generalised) resolution (allowing also oracles) with general upper and lower bounds.

However, in this context, these bounds do not play a role, but only certain algorithmic and structural properties.

Splitting trees and resolution

Deciding SAT via backtracking (for a variable v we have m branches $v \to \varepsilon_1, \dots, v \to \varepsilon_m$ for $D_v = \{\varepsilon_1, \dots, \varepsilon_m\}$) yields splitting trees.

Via the correspondence of partial assignments and clauses, all the paths from the root to some leaf in such a splitting tree correspond to clauses.

Splitting trees thus give us tree-like resolution, and via counting only different clauses also full resolution.

We have developed a theory for this (generalised) resolution (allowing also oracles) with general upper and lower bounds.

However, in this context, these bounds do not play a role, but only certain algorithmic and structural properties.

Splitting trees and resolution

Deciding SAT via backtracking (for a variable v we have m branches $v \to \varepsilon_1, \dots, v \to \varepsilon_m$ for $D_v = \{\varepsilon_1, \dots, \varepsilon_m\}$) yields splitting trees.

Via the correspondence of partial assignments and clauses, all the paths from the root to some leaf in such a splitting tree correspond to clauses.

Splitting trees thus give us tree-like resolution, and via counting only different clauses also full resolution.

We have developed a theory for this (generalised) resolution (allowing also oracles) with general upper and lower bounds.

However, in this context, these bounds do not play a role, but only certain algorithmic and structural properties.

Splitting trees and resolution

Deciding SAT via backtracking (for a variable v we have m branches $v \to \varepsilon_1, \dots, v \to \varepsilon_m$ for $D_v = \{\varepsilon_1, \dots, \varepsilon_m\}$) yields splitting trees.

Via the correspondence of partial assignments and clauses, all the paths from the root to some leaf in such a splitting tree correspond to clauses.

Splitting trees thus give us tree-like resolution, and via counting only different clauses also full resolution.

We have developed a theory for this (generalised) resolution (allowing also oracles) with general upper and lower bounds.

However, in this context, these bounds do not play a role, but only certain algorithmic and structural properties.

Splitting trees and resolution

Deciding SAT via backtracking (for a variable v we have m branches $v \to \varepsilon_1, \dots, v \to \varepsilon_m$ for $D_v = \{\varepsilon_1, \dots, \varepsilon_m\}$) yields splitting trees.

Via the correspondence of partial assignments and clauses, all the paths from the root to some leaf in such a splitting tree correspond to clauses.

Splitting trees thus give us tree-like resolution, and via counting only different clauses also full resolution.

We have developed a theory for this (generalised) resolution (allowing also oracles) with general upper and lower bounds.

However, in this context, these bounds do not play a role, but only certain algorithmic and structural properties.

Autarkies and resolution

Lemma *For every clause-set F and every clause $C \in F$ there exists an autarky for F satisfying C (i.e., C is not in the lean kernel of F) if and only if C cannot be used in any resolution refutation of F.*

Based on this lemma, the lean kernel of F can be computed by a series of calls to any SAT solver which computes

- satisfying assignments (in the SAT case)
- the set of variables used in the resolution refutation it (implicitly) found (in the UNSAT case).

(See the SAT06 contribution with Inês Lynce and João Marques-Silva for the boolean case.)

More on computing the lean kernel

- The decision problem, whether for inputs F', F with $F' \subseteq F$ we have $N_a(F) = F'$ is D^P-complete (the same complexity class as deciding minimally unsatisfiable clause-sets).
- Since the clauses satisfied by an autarky are contained in every maximally satisfiable sub-clause-set, this computation of the lean kernel might be an interesting pre-processing for MAXSAT.

A fundamental autarky system based an matching theory

In the boolean case **matchings** in the (bipartite) clause-variable graph of F play an important role.

We now set out to generalise these results (connected to the notion of **deficiency**, a notion introduced in the boolean case by John Franco and Allen van Gelder).

Some of these generalisations are more or less straight-forward, but

- for some notions different versions had to be examined
- for some results the proofs are more difficult than in the boolean case.

A fundamental autarky system based an matching theory

In the boolean case **matchings** in the (bipartite) clause-variable graph of F play an important role.

We now set out to generalise these results (connected to the notion of **deficiency**, a notion introduced in the boolean case by John Franco and Allen van Gelder).

Some of these generalisations are more or less straight-forward, but

- for some notions different versions had to be examined
- for some results the proofs are more difficult than in the boolean case.

Generalising the deficiency I

For a boolean clause-set F the **deficiency** is

$$\boldsymbol{\delta(F)} := c(F) - n(F),$$

the difference of the number of clauses and the number of variables.

Now the **weight** of a variable v is $\mathrm{wn}(v) := |D_v| - 1$, and

$$\mathbf{wn}(\boldsymbol{F}) := \sum_{v \in \mathrm{var}(F)} \mathrm{wn}(v).$$

The idea is, that for v we can satisfy any given $\mathrm{wn}(v)$ variables by choosing an unused value for v.

Generalising the deficiency I

For a boolean clause-set F the **deficiency** is

$$\boldsymbol{\delta(F)} := c(F) - n(F),$$

the difference of the number of clauses and the number of variables.

Now the **weight** of a variable v is $\text{wn}(v) := |D_v| - 1$, and

$$\textbf{wn}(\boldsymbol{F}) := \sum_{v \in \text{var}(F)} \text{wn}(v).$$

The idea is, that for v we can satisfy any given $\text{wn}(v)$ variables by choosing an unused value for v.

Generalising the deficiency II

Now

$$\delta(F) := c(F) - \mathrm{wn}(F) \in \mathbb{N}$$
$$\delta^*(F) := \max_{F' \subseteq F} \delta(F') \in \mathbb{N}_0.$$

F is **matching satisfiable** if and only if $\delta^*(F) = 0$.

F is matching satisfiable iff there is a matching in generalised clause-variable graph $\boldsymbol{B(F)}$ covering all clause-nodes.

Generalising the boolean case, now in $B(F)$ every variable v exists in $\mathrm{wn}(v)$ many copies.

$c(F) - \delta^*(F)$ is the size of a maximum matching satisfiable sub-clause-set of F.

Generalising the deficiency II

Now

$$\delta(F) := c(F) - \mathrm{wn}(F) \in \mathbb{N}$$
$$\delta^*(F) := \max_{F' \subseteq F} \delta(F') \in \mathbb{N}_0.$$

F is **matching satisfiable** if and only if $\delta^*(F) = 0$.

F is matching satisfiable iff there is a matching in generalised clause-variable graph $\boldsymbol{B(F)}$ covering all clause-nodes.

Generalising the boolean case, now in $B(F)$ every variable v exists in $\mathrm{wn}(v)$ many copies.

$c(F) - \delta^*(F)$ is the size of a maximum matching satisfiable sub-clause-set of F.

Generalising the deficiency II

Now

$$\delta(F) := c(F) - \mathrm{wn}(F) \in \mathbb{N}$$
$$\delta^*(F) := \max_{F' \subseteq F} \delta(F') \in \mathbb{N}_0.$$

F is **matching satisfiable** if and only if $\delta^*(F) = 0$.

F is matching satisfiable iff there is a matching in generalised clause-variable graph $\boldsymbol{B(F)}$ covering all clause-nodes.

Generalising the boolean case, now in $B(F)$ every variable v exists in $\mathrm{wn}(v)$ many copies.

$c(F) - \delta^*(F)$ is the size of a maximum matching satisfiable sub-clause-set of F.

Generalising the deficiency II

Now

$$\delta(F) := c(F) - \mathrm{wn}(F) \in \mathbb{N}$$
$$\delta^*(F) := \max_{F' \subseteq F} \delta(F') \in \mathbb{N}_0.$$

F is **matching satisfiable** if and only if $\delta^*(F) = 0$.

F is matching satisfiable iff there is a matching in generalised clause-variable graph $\boldsymbol{B(F)}$ covering all clause-nodes.

Generalising the boolean case, now in $B(F)$ every variable v exists in $\mathrm{wn}(v)$ many copies.

$c(F) - \delta^*(F)$ is the size of a maximum matching satisfiable sub-clause-set of F.

Poly-time SAT decision in the maximal deficiency

Based on a question of Hans Kleine Büning posed at SAT 1998, Herbert Fleischner, Stefan Szeider and me showed, that for boolean clause-sets SAT is decidable in polynomial time for bounded $\delta^*(F)$.

The generalisation works smoothly.

Stefan Szeider later showed for boolean clause-sets, that SAT is FPT in $\delta^*(F)$: The exponent in the polynomial does not depend on $\delta^*(F)$ (only the coefficient).

This is an example where the generalisation (of the proof) does not work out so smoothly. We need here also the tool of the *boolean translation*.

Poly-time SAT decision in the maximal deficiency

Based on a question of Hans Kleine Büning posed at SAT 1998, Herbert Fleischner, Stefan Szeider and me showed, that for boolean clause-sets SAT is decidable in polynomial time for bounded $\delta^*(F)$.

The generalisation works smoothly.

Stefan Szeider later showed for boolean clause-sets, that SAT is FPT in $\delta^*(F)$: The exponent in the polynomial does not depend on $\delta^*(F)$ (only the coefficient).

This is an example where the generalisation (of the proof) does not work out so smoothly. We need here also the tool of the *boolean translation*.

Matching autarkies

A **matching autarky** is an autarky which matching satisfies the clauses it satisfies. We obtain:

F is matching lean if and only if for every $F' \subset F$ we have $\delta(F') < \delta(F)$.

Thus if F is matching lean (has no non-trivial matching autarkies), then $\delta(F) \geq 1$ holds.

Especially, minimally unsatisfiable clause-sets F fulfil $\delta(F) \geq 1$.

Matching autarkies

A **matching autarky** is an autarky which matching satisfies the clauses it satisfies. We obtain:

F is matching lean if and only if for every $F' \subset F$ we have $\delta(F') < \delta(F)$.

Thus if F is matching lean (has no non-trivial matching autarkies), then $\delta(F) \geq 1$ holds.

Especially, minimally unsatisfiable clause-sets F fulfil $\delta(F) \geq 1$.

Good properties of the boolean translation

The translation of (generalised) F into a boolean clause-set $\theta(F)$ is straight-forward (nearly):

1. For every literal "$v \neq \varepsilon$" take a (distinct) boolean variable v_ε.
2. Translate the clauses of F directly by replacing the literals with their corresponding boolean variables.
3. Add the ALO clauses, expressing that every v gets at least one value (i.e., $\neg v_{\varepsilon_1} \vee \neg v_{\varepsilon_2} \vee \ldots \vee \neg v_{\varepsilon_m}$).

Is is F satisfiable, minimally unsatisfiable, lean iff $\theta(F)$ is.

The catch here is not to use the AMO clauses:

$$\delta(\theta(F)) = \delta(F).$$

(The restriction to *negative* monosigned literals is essential for this — for (full) monosigned literals we need the AMO clauses, and deficiency preservation is lost).

Good properties of the boolean translation

The translation of (generalised) F into a boolean clause-set $\theta(F)$ is straight-forward (nearly):

1. For every literal "$v \neq \varepsilon$" take a (distinct) boolean variable v_ε.
2. Translate the clauses of F directly by replacing the literals with their corresponding boolean variables.
3. Add the ALO clauses, expressing that every v gets at least one value (i.e., $\neg v_{\varepsilon_1} \vee \neg v_{\varepsilon_2} \vee \dots \vee \neg v_{\varepsilon_m}$).

Is is F satisfiable, minimally unsatisfiable, lean iff $\theta(F)$ is.

The catch here is not to use the AMO clauses:

$$\delta(\theta(F)) = \delta(F).$$

(The restriction to *negative* monosigned literals is essential for this — for (full) monosigned literals we need the AMO clauses, and deficiency preservation is lost)

FPT

The translation behaves more complicated regarding

- maximal deficiency
- matching satisfiability
- matching leanness

however, it is enough to prove FPT of SAT in $\delta^*(F)$ by using the boolean translation.

Some remarks:

- At least in the boolean case, the practical applications of SAT decision for low deficiency seem to be rather restricted. However there is more hope (and non-trivial examples) for QBF.
- Stefan Szeider showed (in the boolean case), that the hierarchies regarding maximal deficiency and incidence treewidth are incomparable.

FPT

The translation behaves more complicated regarding

- maximal deficiency
- matching satisfiability
- matching leanness

however, it is enough to prove FPT of SAT in $\delta^*(F)$ by using the boolean translation.

Some remarks:

- At least in the boolean case, the practical applications of SAT decision for low deficiency seem to be rather restricted. However there is more hope (and non-trivial examples) for QBF.
- Stefan Szeider showed (in the boolean case), that the hierarchies regarding maximal deficiency and incidence treewidth are incomparable.

Remark on splitting

For SAT solving of (generalised) clause-sets splitting on the binary alternative

$$\text{“}v = \varepsilon\text{” or “}v \neq \varepsilon\text{”}$$

is more powerful than splitting on $v = \varepsilon_1, \ldots, v = \varepsilon_m$ (shown by David Mitchell and Joey Hwang).

So, if we want to use *learning*, then we also need "positive" literals "$v = \varepsilon$" besides our negative literals, and things get more complicated (a main reason why generalising SAT to non-boolean variables proved to be difficult in the past).

Irredundant clause-set

- A clause-set F is **irredundant** if removing any clause changes the set of satisfying assignments.
- So the unsatisfiable irredundant clause-sets are exactly the minimally unsatisfiable ones.

We are interested in

1. preservation of (ir)redundancy under application of partial assignments
2. irredundant clause-sets with regular “conflict patterns” (the graph of clashes).
3. minimally unsatisfiable clause-sets of low deficiency.

Hitting clause-sets

A **hitting** clause-set is a clause-set where every two different clauses clash (i.e., the conflict graph is complete).

Lemma *A clause-set F is hitting if and only if for every partial assignment φ (including the empty partial assignment, of course) the clause-set $\varphi * F$ is irredundant.*

Thus hitting clause-sets are the "most irredundant" clause-sets.

SAT decision for hitting clause-sets is in poly-time (by counting). Can we generalise this to a larger class of clause-sets?!

Hitting clause-sets

A **hitting** clause-set is a clause-set where every two different clauses clash (i.e., the conflict graph is complete).

Lemma *A clause-set F is hitting if and only if for every partial assignment φ (including the empty partial assignment, of course) the clause-set $\varphi * F$ is irredundant.*

Thus hitting clause-sets are the "most irredundant" clause-sets.

SAT decision for hitting clause-sets is in poly-time (by counting). Can we generalise this to a larger class of clause-sets?!

Multihitting clause-sets

A clause-set F is **(k)-multihitting** if we can divide the clauses of F into (k) parts such that

1. clauses from the same part never clash
2. clauses from different parts always clash.

(I.e., the conflict graph of F is complete (k)-multipartite.)

Lemma *A multihitting clause-set F has a unique irredundant core (a sub-clause-set which is irredundant and has the same set of satisfying assignments as F). This irredundant core is computed by subsumption elimination.*

So every unsatisfiable multihitting clause-set has a unique minimally unsatisfiable sub-clause-set, which can be computed by subsumption elimination.

Multihitting clause-sets

A clause-set F is **(k)-multihitting** if we can divide the clauses of F into (k) parts such that

1. clauses from the same part never clash
2. clauses from different parts always clash.

(I.e., the conflict graph of F is complete (k)-multipartite.)

Lemma *A multihitting clause-set F has a unique irredundant core (a sub-clause-set which is irredundant and has the same set of satisfying assignments as F). This irredundant core is computed by subsumption elimination.*
So every unsatisfiable multihitting clause-set has a unique minimally unsatisfiable sub-clause-set, which can be computed by subsumption elimination.

On bihitting clause-sets

SAT decision for bihitting (2-hitting) clause-sets is the same problems as the **hypergraph transversal** problem:

1. So SAT decision for bihitting clause-sets can be done in quasi-polynomial time.
2. And its a long-standing open problem whether SAT decision for bihitting clause-sets is in polynomial time.

What about SAT decision for 3-hitting clause-sets ?!

MUs of minimal deficiency

The generalisation of the characterisation of boolean MUs of low deficiency, as pioneered by Hans Kleine Büning and Zhao Xishun, is complicated by the breakdown of important tools.

We managed to characterise the deficiency-one-case by using the boolean translation:

Lemma *Minimally unsatisfiable clause-sets of deficiency* 1 *have the same tree structure as in the boolean case, and the characterisation of the subclasses of saturated and marginal clause-sets also carry over.*

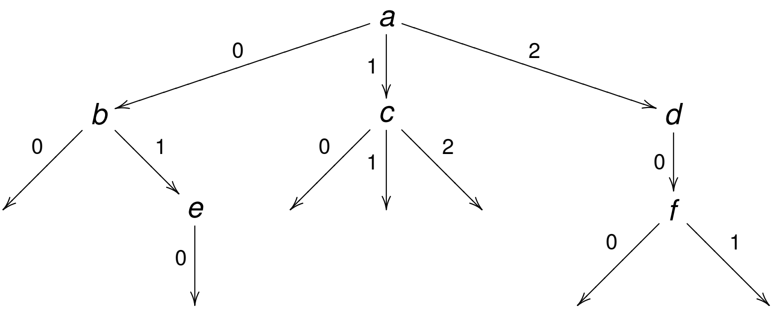
MUs of minimal deficiency

The generalisation of the characterisation of boolean MUs of low deficiency, as pioneered by Hans Kleine Büning and Zhao Xishun, is complicated by the breakdown of important tools.

We managed to characterise the deficiency-one-case by using the boolean translation:

Lemma *Minimally unsatisfiable clause-sets of deficiency* 1 *have the same tree structure as in the boolean case, and the characterisation of the subclasses of saturated and marginal clause-sets also carry over.*

The tree structure exemplified



$$F = \{ \{a \neq 0, b \neq 0\}, \{a \neq 0, b \neq 1, e \neq 0\},$$
$$\{a \neq 1, c \neq 0\}, \{a \neq 1, c \neq 1\}, \{a \neq 1 c \neq 2\},$$
$$\{a \neq 2, d \neq 0, f \neq 0\}, \{a \neq 2, d \neq 0, f \neq 1\} \}.$$

Translating hypergraph colouring

Consider a hypergraph G (a set of "vertices" with a set of "hyperedges", which are just sets of vertices).

Consider also a domain D, here interpreted as "colours".

Obtain the clause-set $\boldsymbol{F_{[D]}(G)}$ as

$$F_{[D]}(G) = \bigcup_{\varepsilon \in D} F_{[\{\varepsilon\},D]}(G),$$

where $F_{[\{\varepsilon\},D]}(G)$ is obtained from G by translating every hyperedge $H = \{v_1, \dots, v_k\}$ of G into a clause

$$v_1 \neq \varepsilon \vee v_2 \neq \varepsilon \vee \dots \vee v_k \neq \varepsilon.$$

The satisfying assignments for $F_{[D]}(G)$ correspond 1-1 to the ("weak") D-colourings of G.

At least as many hyperedges as vertices

Theorem *Consider a hypergraph G without non-covered vertices. If there exists a domain D with $|D| \geq 2$ such that $F_{[D]}(G)$ is (linearly) lean, then there is a matching in $B(G)$ covering all vertex nodes, whence $|E(G)| \geq |V(G)|$.*

The proof exploits linear algebra (basically the "rank argument") in the form of "balanced linear autarkies" together with matching autarkies.

At least as many hyperedges as vertices

Theorem *Consider a hypergraph G without non-covered vertices. If there exists a domain D with $|D| \geq 2$ such that $F_{[D]}(G)$ is (linearly) lean, then there is a matching in $B(G)$ covering all vertex nodes, whence $|E(G)| \geq |V(G)|$.*

The proof exploits linear algebra (basically the “rank argument”) in the form of “balanced linear autarkies” together with matching autarkies.

Fisher's inequality

To show the range of potential applications, we re-derive:

Every hypergraph G, such that two vertices are always on exactly λ blocks for some fixed λ (a "pairwise balanced design"), has $|E(G)| \geq |V(G)|$ (except for trivial situations).

The "new" proof just rephrases the known one based on linear algebra, but it puts it into the context of showing lower bounds on some form of deficiency:

It is known in combinatorics that the underlying ideas of linear algebra are usable in many situations, but yet there is no common framework. Perhaps autarky theory can provide this framework.

Fisher's inequality

To show the range of potential applications, we re-derive:

Every hypergraph G, such that two vertices are always on exactly λ blocks for some fixed λ (a "pairwise balanced design"), has $|E(G)| \geq |V(G)|$ (except for trivial situations).

The "new" proof just rephrases the known one based on linear algebra, but it puts it into the context of showing lower bounds on some form of deficiency:

It is known in combinatorics that the underlying ideas of linear algebra are usable in many situations, but yet there is no common framework. Perhaps autarky theory can provide this framework.

Critically colourable

Two notions: A hypergraph G is **minimally non-k-colourable** if G is non-k-colourable, but removing any hyperedges renders it k-colourable. And G is **critical $(k+1)$-colourable** if G is $(k+1)$-colourable and minimally non-k-colourable.

A central observation:
*A hypergraph G is **minimally non-k-colourable** if and only if $F_{[k]}(G)$ is minimally unsatisfiable.*

If G is critical $(k+1)$-colourable, then G is not critical $(k'+1)$-colourable for $k' < k$. **But** $F_{[k]}(G)$ is minimally unsatisfiable, **thus** lean, and leanness is inherited to k'. So for applications where we are actually need only leanness, we can go from k to k'.

Critically colourable

Two notions: A hypergraph G is **minimally non-k-colourable** if G is non-k-colourable, but removing any hyperedges renders it k-colourable. And G is **critical $(k+1)$-colourable** if G is $(k+1)$-colourable and minimally non-k-colourable.

A central observation:
*A hypergraph G is **minimally non-k-colourable** if and only if $F_{[k]}(G)$ is minimally unsatisfiable.*

If G is critical $(k+1)$-colourable, then G is not critical $(k'+1)$-colourable for $k' < k$. **But** $F_{[k]}(G)$ is minimally unsatisfiable, **thus** lean, and leanness is inherited to k'. So for applications where we are actually need only leanness, we can go from k to k'.

Seymour's inequality

Seymour proved that a critical 3-colourable hypergraph has at least as many hyperedges as vertices.

By the meta theorem we immediately get this for every $k \geq 3$.

The condition $|E(G)| \geq |V(G)|$ translates into

$$\delta(F_{[D]}(G)) \geq n(F).$$

So every minimally unsatisfiable **colouring clause-set** F has $\delta(F) \geq n(F)$.

("Colouring" F are completely determined by the uniform domain D and the variable hypergraph G.)

Seymour's inequality

Seymour proved that a critical 3-colourable hypergraph has at least as many hyperedges as vertices.

By the meta theorem we immediately get this for every $k \geq 3$.

The condition $|E(G)| \geq |V(G)|$ translates into

$$\delta(F_{[D]}(G)) \geq n(F).$$

So every minimally unsatisfiable **colouring clause-set** F has $\delta(F) \geq n(F)$.

("Colouring" F are completely determined by the uniform domain D and the variable hypergraph G.)

Classifying minimally unsatisfiable colouring clause-sets

Finally we set out to classify all minimally unsatisfiable colouring clause-sets F of "minimal deficiency" $\delta(F) = n(F)$.

It is not clear at all whether this might be feasible. So we restrict F further to be **multihitting**.

Classifying minimally unsatisfiable colouring clause-sets

Finally we set out to classify all minimally unsatisfiable colouring clause-sets F of “minimal deficiency” $\delta(F) = n(F)$.

It is not clear at all whether this might be feasible. So we restrict F further to be **multihitting**.

Classification

Theorem The multihitting minimally unsatisfiable colouring clause-sets F with $\delta(F) = n(F)$ are as follows:

1. The only examples with domain size greater than two (that is, more than two multihitting parts) are the trivial minimally unsatisfiable clause-sets with $n = 1$.
2. If F does not contain unit or binary clauses, then F is the encoding of the 2-colouring of the Fano plane (the unique projective incidence plane of order 2); thus $n = 7$, $c = 14$, and all clauses have size 3.
3. Otherwise F is $\neg a \leftrightarrow b \wedge b \leftrightarrow \neg c \wedge \neg c \leftrightarrow a$ (trying to 2-colour the cycle of length 3), or some (iterated) "2-extension" of it (introducing a literal equivalent some already existing literal, keeping always at least one binary clause, never introducing a unit clause, and always having at least one ternary clause).

Classification II

We can draw many interesting corollaries from the classification theorem (which relies heavily on the classification by Seymour of all intersecting square minimally non-2-colouring hypergraphs), for example:

Corollary *A multihitting colouring clause-set F with $\delta(F) = n(F)$, where every clause has at least size four, must be satisfiable.*

Corollary *The satisfiability problem for multihitting colouring clause-sets F with $\delta^*(F) \leq n(F)$ is decidable in polynomial time.*

Summary

- introduced deficiency for generalised clause-sets
- matching lean F fulfil $\delta(F) \geq 1$
- showed SAT to be FPT w.r.t. the maximal deficiency
- classified all minimally unsatisfiable F with $\delta(F) = 1$
- irredundancy; hitting and multihitting clause-sets
- a general meta theorem for showing $|E(G)| \geq |V(G)|$ for hypergraphs G
- two applications: inequalities of Fisher and Seymour
- classification of all minimally unsatisfiable multihitting colouring clause-sets of minimal deficiency.

End