Variable Independence in Linear Real Arithmetic

Alexander Mayorov

University of Kaiserslautern–Landau (RPTU)

July 13, 2023

Table of Contents

Motivation: query evaluation

Consider a database with relations $X(x_1, x_2)$ and $Y(y_1, y_2)$.

$X(x_1, x_2)$	
x_1	x_2
1	$1/3$
2	1
1	0
...	...

$Y(y_1, y_2)$	
y_1	y_2
-50	11
7	5
1	$1/2$
...	...

Suppose we want to evaluate a query

$$q \leftarrow X(x_1, x_2), Y(y_1, y_2), \varphi(x_1, x_2, y_1, y_2)$$

where

$$\varphi(x_1, x_2, y_1, y_2) := 4x_1 - 6x_2 + y_1 + 5y_2 < 8 \wedge 2x_1 - 3x_2 = 1$$

is a formula expressing the desired tuples.

Naive query evaluation

Consider all possible combinations of entries and model check.

$X(x_1, x_2)$	
x_1	x_2
1	1/3
2	1
1	0
...	...

$\times$

$Y(y_1, y_2)$	
y_1	y_2
−50	11
7	5
1	1/2
...	...

$=$

$X(x_1, x_2) \times Y(y_1, y_2)$				φ sat?
x_1	x_2	y_1	y_2	
1	1/3	−50	11	yes
1	1/3	7	5	no
1	1/3	1	1/2	yes
2	1	−50	11	yes
2	1	7	5	no
2	1	1	1/2	yes
1	0	−50	11	no
1	0	7	5	no
1	0	1	1/2	no
...	...	...	...	...

$\rightsquigarrow$ $\Theta(n^2)$ worst-case running time (unfortunately)

Efficient query evaluation

However, what if we know that

$$\varphi \equiv \underbrace{2x_1 - 3x_2 = 1}_{\text{Uses only } x_1, x_2} \wedge \underbrace{y_1 + 5y_2 < 6}_{\text{Uses only } y_1, y_2}$$

is a Boolean combination of predicates never using x_i and y_j at the same time? Then we independently check (and possibly using parallelization):

$X(x_1, x_2)$		$2x_1 - 3x_2 = 1$ sat?
x_1	x_2	
1	1/3	yes
2	1	yes
1	0	no
...	...	...

$Y(y_1, y_2)$		$y_1 + 5y_2 < 6$ sat?
y_1	y_2	
-50	11	yes
7	5	no
1	1/2	yes
...	...	...

$\rightsquigarrow$ The desired query result is the set of all yes-yes combinations of rows from the two tables.

$\rightsquigarrow$ Same query evaluated in $O(n)$ time!

Motivation – continued

This motivates the following intuitive question:

> When can predicates appearing in some logical formula be "torn apart" while preserving equivalence?

We have just seen that x_1, x_2 can be "torn apart" from y_1, y_2 in φ:

$$\varphi(x_1, x_2, y_1, y_2) = \overbrace{4x_1 - 6x_2 + y_1 + 5y_2 < 8}^{\text{Uses } x_i \text{ and } y_j \text{ at the same time}} \wedge 2x_1 - 3x_2 = 1$$
$$\equiv \underbrace{y_1 + 5y_2 < 6 \wedge 2x_1 - 3x_2 = 1}_{\text{Never uses } x_i \text{ and } y_j \text{ at the same time}}$$

This is known as Π-*decomposability* where $\Pi := \{\{x_1, x_2\}, \{y_1, y_2\}\}$.

Capturing variable independence rigorously

More precisely, fix a first-order structure $\mathcal{M}$ and let all formulas be over the quantifier-free first-order theory generated by $\mathcal{M}$.

Definition

Let $\varphi(x_1, \ldots, x_n)$, $\psi(x_1, \ldots, x_n)$ be formulas and Π be a partition of $\{x_1, \ldots, x_n\}$. We say that

ψ is a Π-decomposition	if ψ is a Boolean combination of formulas each having its free variables within some block of Π
φ is Π-decomposable	if there exists a Π-decomposition ψ such that $\varphi \equiv \psi$

Intuition: Π specifies the allowed and forbidden "connections" between variables; Π-decomposability captures the intuitive question above.

Example

Example ($\mathcal{M} = (\mathbb{Q}, +, <, =, 0, 1)$)

Let $\Pi := \{\{x\}, \{y\}\}$. The formula

$$\varphi := (x + y \neq 2 \vee x - y \neq 0) \wedge x > 0$$

is Π-decomposable because

$$\varphi \equiv (x \neq 1 \vee y \neq 1) \wedge x > 0$$

On the other hand, $\varphi := x < y$ is not Π-decomposable.

The variable decomposition problem

Fix $\mathcal{M} := (\mathbb{Q}, +, <, =, 0, 1)$, i.e., consider linear real arithmetic.

Problem (The variable decomposition problem)

Given a formula φ and a binary partition Π, either compute a Π-decomposition of φ or determine that φ is not Π-decomposable.

- Efficient algorithms for the variable decomposition problem can be applied to speedup query evaluation in databases (see above)!
- Many other applications, including string solving, symbolic transducers, quantifier elimination and automata theory.

Best known algorithm: runs in double-exponential time;

Open problem

What is the precise complexity of the variable decomposition problem (over linear real arithmetic)?

Main result

Complete answer to the open question:

Theorem

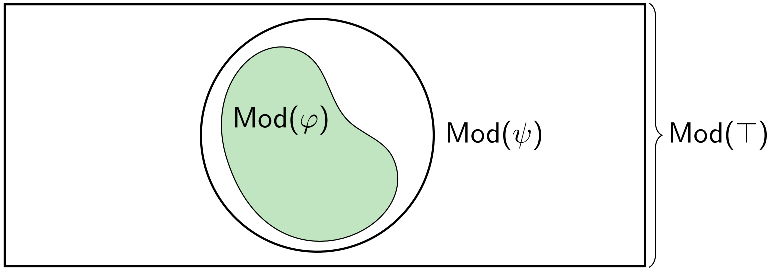
Over linear real arithmetic, the variable decomposition problem can be solved in exponential time. The decision version is coNP-complete.

- Overcomes a fundamental barrier between decomposability over *discrete* theories (for which many techniques are known) and over *non-discrete* ones (for which known techniques either do not work, or yield inefficient algorithms)
- Implies previously unknown bounds on sizes of decompositions
- Relies only on fundamental properties of the theory and thus has the potential of being generalized to other theories
- Exponential-time algorithm is optimal in the setting of deterministic algorithms because the problem is coNP-hard
- Furthermore, the algorithm is implementable and efficient in practice (as experiments have shown)

Another unique feature: partial variable independence

- Suppose we want to apply the algorithm to optimize database queries.
- If the formula φ appearing in a query is Π-decomposable, then we can optimize the query by fully avoiding joining tables and by taking advantage of parallelism.
- However, what if φ is not Π-decomposable?

Idea: find a "minimal" Π-decomposition ψ such that $\varphi \models \psi$ and use ψ instead of φ to optimize the query:



Whenever ψ is not valid, we say that φ has *partial variable independence*.

Example application of partial variable independence

Consider a real-world database containing information about student apartments in Kaiserslautern and Berlin[1]:

KL	
Address	s_{KL}
Richard-Wagner-Straße 88	14
Friedrich-Engels-Straße 5	13
Apfelstraße 6	19
...	...

Berlin	
Address	s_{Berlin}
Hermannstraße 151	19
Lacknerstraße 5	24
Friedelstraße 23	18
...	...

- s_{KL} and s_{Berlin} are areas of corresponding apartments (in m^2).
- Let $r_{\text{KL}} := 18.24$ and $r_{\text{Berlin}} := 27.17$ be the (average) rental prices per square meter in Kaiserslautern and Berlin, respectively.

[1] Source: `https://www.wg-gesucht.de`

Example of application – continued

- Suppose we want to:

> Find apartments from Kaiserslautern and Berlin that have approximately the same area but differ in price significantly, say, the price of the apartment in Berlin must be at least double the price of the apartment in Kaiserslautern

- More precisely, we regard two apartments as having approximately the same area whenever these areas differ by at most 5 sq. meters.

Finding such apartments corresponds to evaluating the conjunctive query

$$q \leftarrow \mathrm{KL}(\_, s_{\mathrm{KL}}), \mathrm{Berlin}(\_, s_{\mathrm{Berlin}}), \varphi(s_{\mathrm{KL}}, s_{\mathrm{Berlin}})$$

where

$$\varphi(s_{\mathrm{KL}}, s_{\mathrm{Berlin}}) := |s_{\mathrm{KL}} - s_{\mathrm{Berlin}}| \leq 5 \land r_{\mathrm{Berlin}} s_{\mathrm{Berlin}} \geq 2 r_{\mathrm{KL}} s_{\mathrm{KL}}$$
$$\land\, s_{\mathrm{KL}} > 0 \land s_{\mathrm{Berlin}} > 0$$

Naive query evaluation

Consider all possible combinations of entries and model check.

KL × Berlin				φ sat?
Address	s_{KL}	Address	s_{Berlin}	
Richard-Wagner-Straße 88	14	Hermannstraße 151	19	yes
Richard-Wagner-Straße 88	14	Lacknerstraße 5	24	no
Richard-Wagner-Straße 88	14	Friedelstraße 23	18	no
Friedrich-Engels-Straße 5	13	Hermannstraße 151	19	no
Friedrich-Engels-Straße 5	13	Lacknerstraße 5	24	no
Friedrich-Engels-Straße 5	13	Friedelstraße 23	18	yes
Apfelstraße 6	19	Hermannstraße 151	19	no
Apfelstraße 6	19	Lacknerstraße 5	24	no
Apfelstraße 6	19	Friedelstraße 23	18	no
...	...	...	...	...

$\leadsto$ Again, we unfortunately get $\Theta(n^2)$ worst-case running time!

Efficient query evaluation

- Fix $\Pi := \{\{s_{\mathsf{KL}}\}, \{s_{\mathsf{Berlin}}\}\}$.
- Unfortunately, φ is not Π-decomposable!
- However, using our algorithm, it is possible to automatically compute

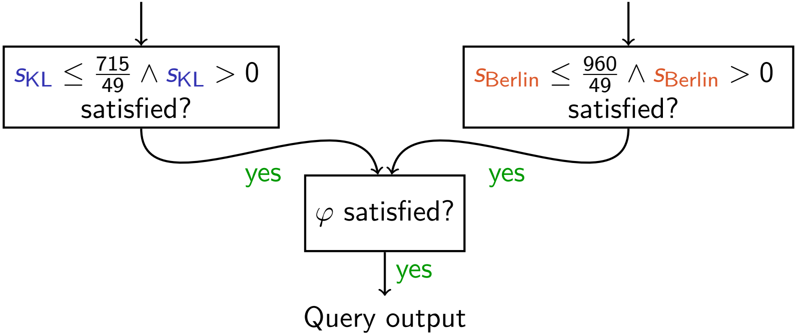
$$\psi := s_{\mathsf{KL}} \leq \frac{715}{49} \wedge s_{\mathsf{KL}} > 0 \wedge s_{\mathsf{Berlin}} \leq \frac{960}{49} \wedge s_{\mathsf{Berlin}} > 0$$

  which is a best-possible approximation of a Π-decomposition for φ.
- The algorithm guarantees $\varphi \models \psi$.
- In other words, there is partial variable independence in φ.
- It can be exploited to optimize the query (see next slide)!

Efficient query evaluation – continued

KL	
Address	s_{KL}
Richard-Wagner-Straße 88	14
Friedrich-Engels-Straße 5	13
Apfelstraße 6	19
...	...

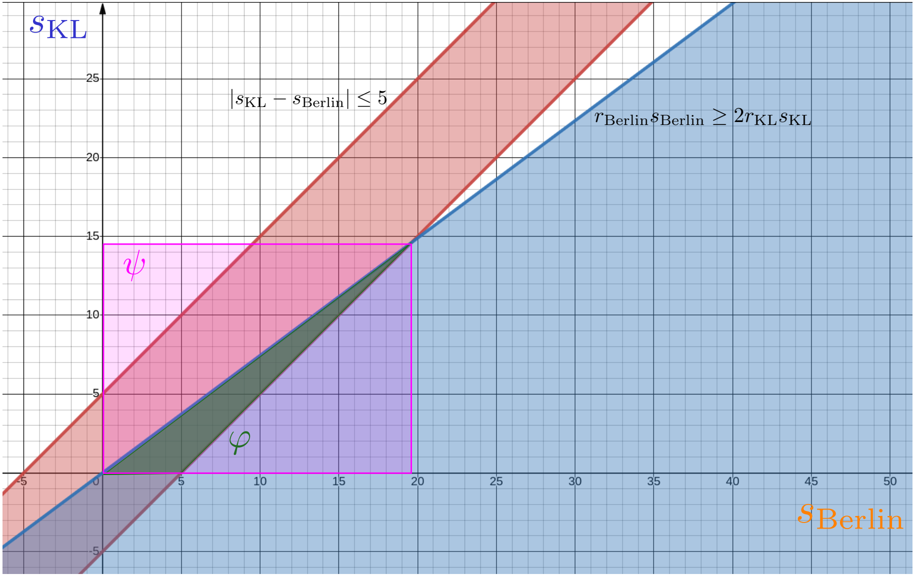
Berlin	
Address	s_{Berlin}
Hermannstraße 151	19
Lacknerstraße 5	24
Friedelstraße 23	18
...	...



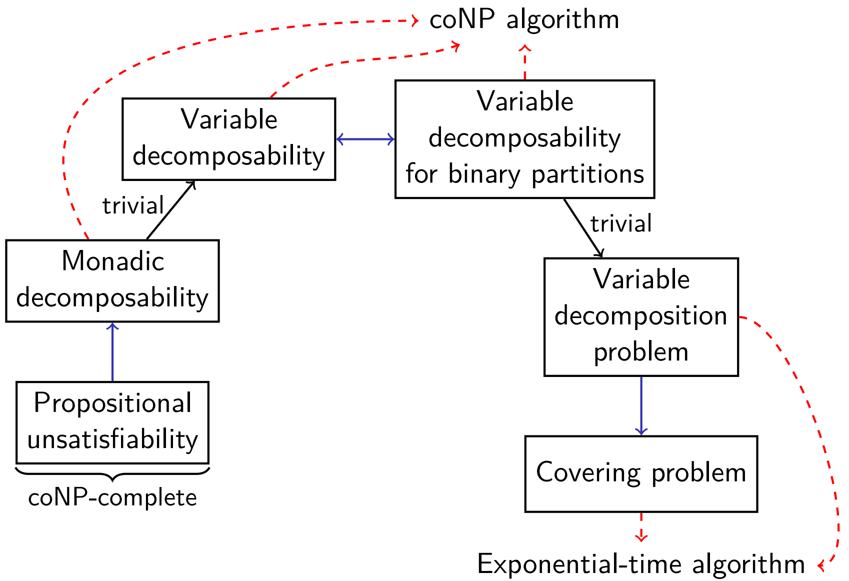
$s_{KL} \leq \frac{715}{49} \wedge s_{KL} > 0$ satisfied?
$s_{Berlin} \leq \frac{960}{49} \wedge s_{Berlin} > 0$ satisfied?
yes
yes
φ satisfied?
yes
Query output

$\rightsquigarrow$ Same query evaluated in $O(n)$ time!

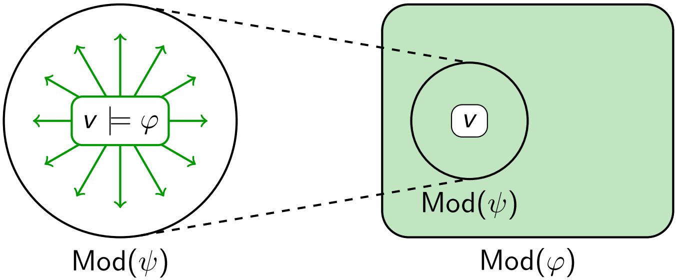
Geometric interpretation



Reductions & algorithms underlying the results


coNP algorithm
Variable decomposability
Variable decomposability for binary partitions
trivial
Monadic decomposability
trivial
Variable decomposition problem
Propositional unsatisfiability
coNP-complete
Covering problem
Exponential-time algorithm

Model flooding formulas



Intuition: when does the Π-decomposability of φ cause a single model of $\psi \wedge \varphi$ to make all models of ψ satisfy φ?

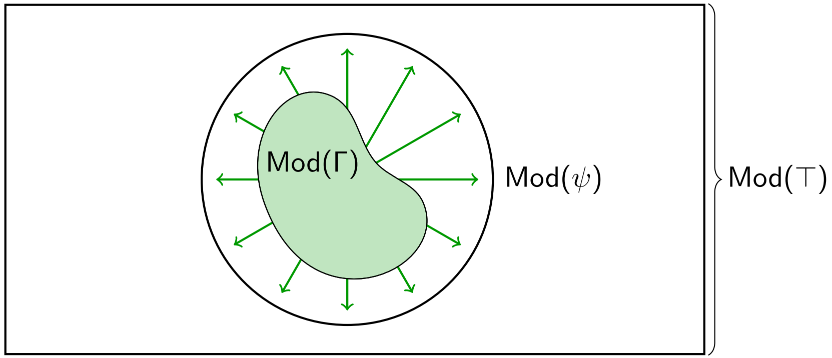
Definition ((φ, Π)-MFF)

Let φ, ψ be formulas. We say that ψ is a (φ, Π)-model-flooding formula ((φ, Π)-MFF) if

$$\varphi \text{ is } \Pi\text{-decomposable} \Rightarrow (\psi \models \varphi \vee \psi \models \neg\varphi)$$

The covering problem

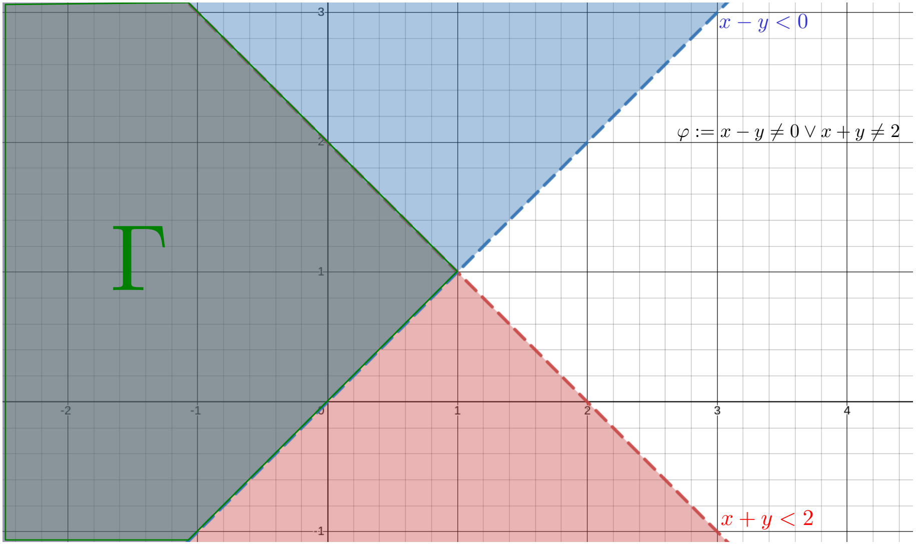
Intuition: compute a Π-decomposition ψ defining a set containing Mod(Γ), such that model flooding occurs in that set.



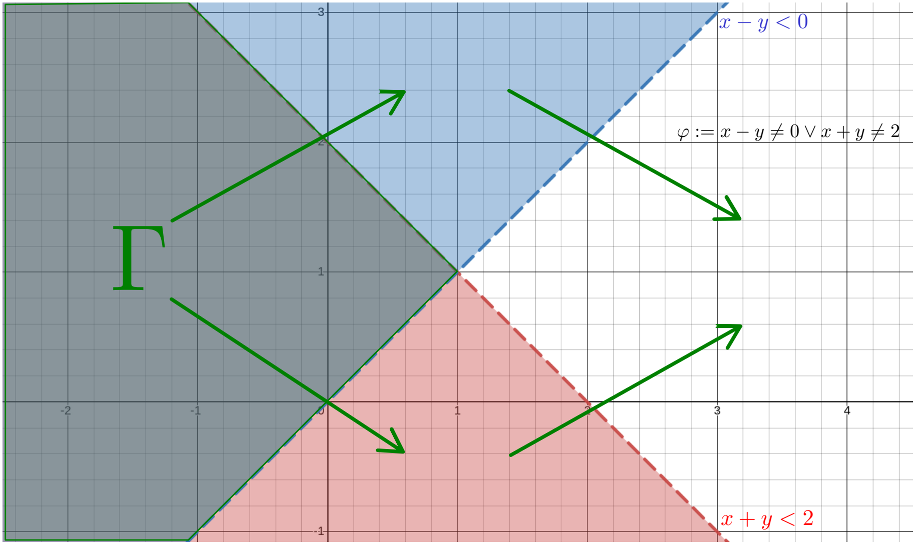
Problem (Covering problem)

Given a formula φ, a binary partition Π and a predicate set $\Gamma \in \mathrm{DNF}_\varphi[\top]$, compute a (φ, Π)-MFF ψ such that $\Gamma \models \psi$ and ψ is a Π-decomposition.

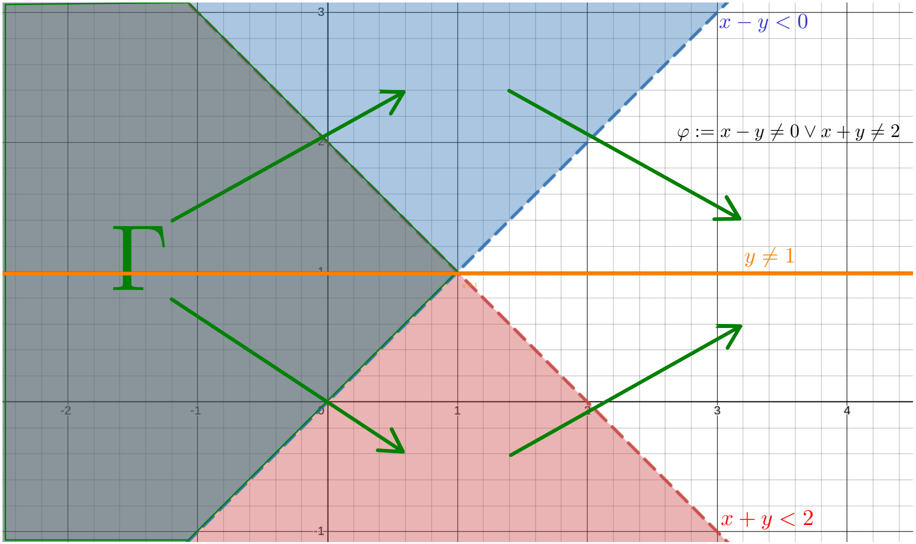
Solving the covering problem



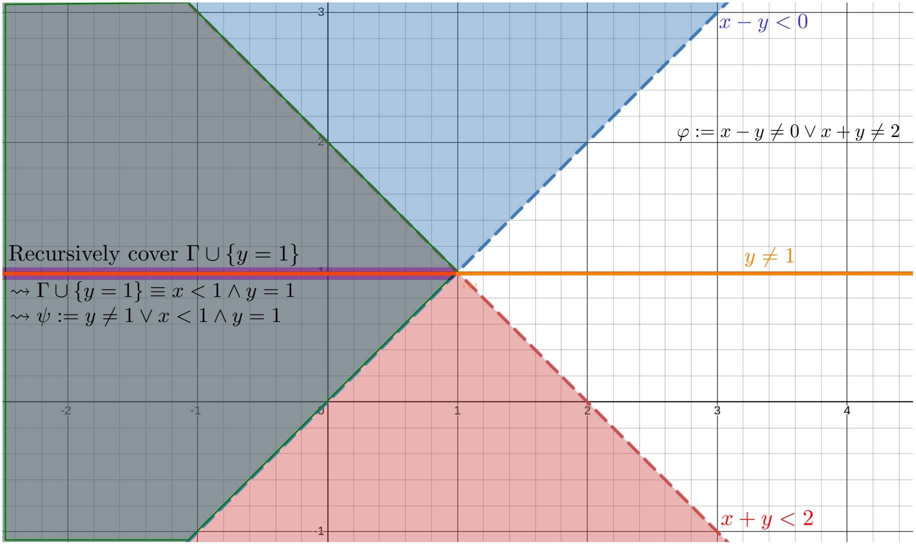
Solving the covering problem



Solving the covering problem



Solving the covering problem



Recursively cover $\Gamma \cup \{y = 1\}$

$\rightsquigarrow \Gamma \cup \{y = 1\} \equiv x < 1 \land y = 1$

$\rightsquigarrow \psi := y \neq 1 \lor x < 1 \land y = 1$

Conclusions

Nontrivial topological properties of the set of logical formulas can be used to significantly speedup algorithms.

- In particular, we improved the best known algorithm for the problem of computing variable decompositions or determining that none exists!
- First algorithm that works in non-discrete settings!
- Main novel technique: **model flooding**
- It has the potential of being adapted to other fragments of first-order logic to obtain similar results.

Open problem

Two given free variables x_i and x_j appearing in a formula $\varphi(x_1, \dots, x_n)$ are independent, if φ is Π-decomposable for some Π containing x_i and x_j in distinct blocks.

Corollary

Over linear real arithmetic, deciding the independence of two given variables is coNP-hard with respect to conjunctive truth-table reductions and is in $\Sigma_2^p = \mathrm{NP}^{\mathrm{NP}}$.

Proof.

Guess appropriate partition Π and run the coNP algorithm. $\square$

Open problem

What is the precise complexity of deciding whether two given variables are independent (over linear real arithmetic)?

Monadic decomposability & further result

A well-studied and important special case of variable decompositions:

Definition

A formula $\varphi(x_1, \dots, x_n)$ is said to be monadically decomposable whenever φ is Π-decomposable for $\Pi := \{\{x_i\} \mid i \in \{1, \dots, n\}\}$.

Example

The formula $\varphi := x + y = 2 \wedge x = 1$ is monadically decomposable, whereas $\varphi := x < y \wedge y < 1$ is not.

Further result:

Theorem

Given a linear real arithmetic formula φ, it is coNP-complete to decide the monadic decomposability of φ.

Efficient reduction to decomposability for binary partitions

Variable decomposability can be efficiently reduced to the case $|\Pi| = 2$:

Theorem

For any (non-unary) partition Π there exists a collection S of binary partitions, such that for any formula φ over an arbitrary theory:

$$\varphi \text{ is } \Pi\text{-decomposable} \Leftrightarrow \forall \Pi' \in S : \varphi \text{ is } \Pi'\text{-decomposable}$$

Moreover, $|S| \in O(\log(|\Pi|))$ and S can be computed in polynomial time.

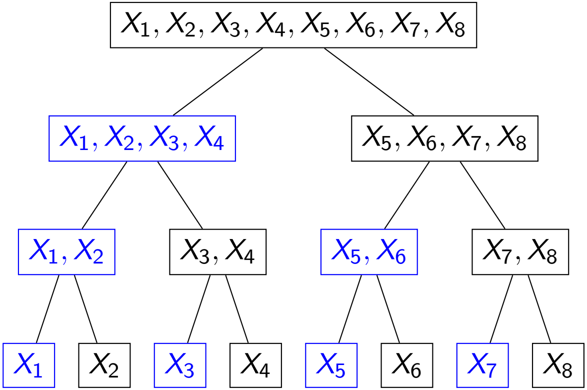
Note: It was already known that testing $\{X_1, \ldots, X_n\}$-decomposability of φ is equivalent to checking that φ is $\{X_i, \bigcup_{j \neq i} X_j\}$-decomposable for all $1 \leq i \leq n$.

$\rightsquigarrow$ The novelty of the above theorem is in the size of S – we improve it from $O(|\Pi|)$ to $O(\log(|\Pi|))$.

Key technique: parallel binary search

Proof sketch

Intuition: express the binary search of every $X_i \in \Pi$ in the language of binary partition sets, using the meet operation $\sqcap$ of the partition lattice.



$$\Pi_{1,1} := X_1 \cup X_2 \cup X_3 \cup X_4$$
$$\Pi_{1,2} := X_5 \cup X_6 \cup X_7 \cup X_8$$

$$\Pi_{2,1} := X_1 \cup X_2 \cup X_5 \cup X_6$$
$$\Pi_{2,2} := X_3 \cup X_4 \cup X_7 \cup X_8$$

$$\Pi_{3,1} := X_1 \cup X_3 \cup X_5 \cup X_7$$
$$\Pi_{3,2} := X_2 \cup X_4 \cup X_6 \cup X_8$$

$$S := \{\{\Pi_{i,1}, \Pi_{i,2}\} \mid i \in \{1, \ldots, \lceil \log(|\Pi|) \rceil\}\} \Rightarrow \bigsqcap_{\Pi' \in S} \Pi' = \Pi$$

Novel characterization of Π-decomposable formulas

Lemma

Fix a formula φ and an oracle solving the covering problem. Let ψ_Γ be the solution to the covering problem for Γ, produced by the oracle. Then

$$\varphi \textit{ is } \Pi\textit{-decomposable} \Leftrightarrow \forall \Gamma \in \mathsf{Sat}(\mathsf{DNF}_\varphi[\varphi]) : \psi_\Gamma \models \varphi$$

Proof.

"$\Rightarrow$": Since $\mathsf{Sat}(\mathsf{DNF}_\varphi[\varphi]) \ni \Gamma \models \psi_\Gamma$, it follows that $\psi_\Gamma \wedge \varphi$ is satisfiable for all Γ. Hence, by definition of (φ, Π)-MFF, $\psi_\Gamma \models \varphi$ for all Γ.
"$\Leftarrow$": The formula φ is Π-decomposable because

$$\varphi \equiv \bigvee_{\Gamma \in \mathsf{Sat}(\mathsf{DNF}_\varphi[\varphi])} \psi_\Gamma$$

The $\models$ entailment follows from the fact that $\Gamma \models \psi_\Gamma$, whereas the converse entailment holds by the assumption that $\psi_\Gamma \models \varphi$. □

$\rightsquigarrow$ The variable decomposition problem reduces to the covering problem.

How to solve the covering problem?

Problem (Covering problem (restated))

Given a formula φ, *a binary partition* Π *and a predicate set* $\Gamma \in \mathrm{DNF}_{\varphi}[\top]$, *compute a* (φ, Π)*-MFF* ψ *such that* $\Gamma \models \psi$ *and* ψ *is a* Π*-decomposition.*

Key question:

How can we ensure that the solution ψ is indeed a (φ, Π)-MFF?

Key intuition:

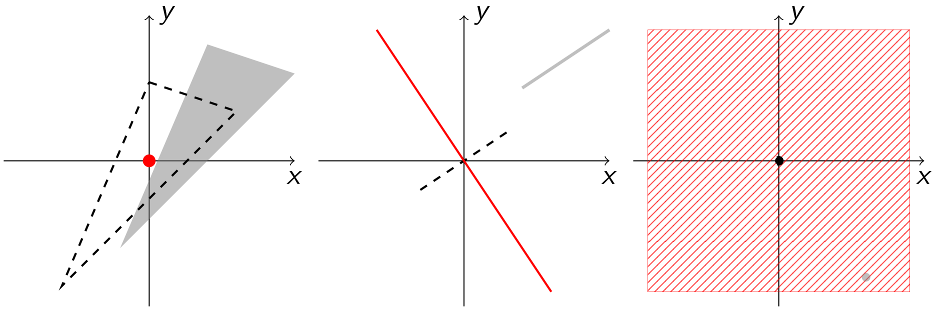
If all $\Lambda \in \mathrm{DNF}_{\varphi}[\top]$ agreeing on some model with ψ enforce same "connections" between variables that are expressible in the language of Π-decompositions, then ψ is a (φ, Π)-MFF.

$\rightsquigarrow$ This leads us to the idea of performing "unification" at the level of disjuncts in $\mathrm{DNF}_{\varphi}[\top]$, with respect to a metric that compares connections between variables expressible using Π-decompositions.

Important: This "unification" must ensure that $\Gamma \models \psi$.

Metric for comparing connections between variables

Consider a satisfiable predicate set Λ.
Question: what "connections" between variables does Λ enforce?
Intuition: think about the semantics $\mathrm{Mod}(\Lambda)$ of Λ



Key insight: every "connection" between variables in Λ corresponds to an element of

$$\{U \leq \mathbb{Q}^n \mid \mathrm{Mod}(\Lambda) - v \subseteq U^{\perp}\}$$

where $v \models \Lambda$ is fixed.

Metric for comparing connections between variables

Problem: We are interested only in those connections between variables that are expressible in the language of Π-decompositions, whereas

$$\{U \leq \mathbb{Q}^n \mid \mathrm{Mod}(\Lambda) - v \subseteq U^\perp\}$$

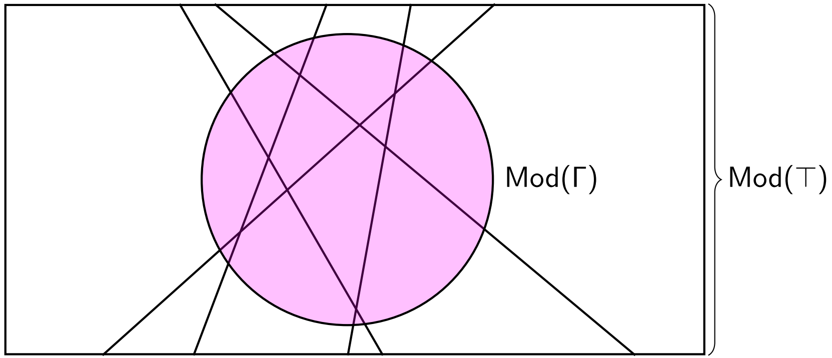
captures all possible dependencies of variables.

Solution: Measure connections between variables separately with respect to every $Z \in \Pi$, by additionally applying the projection homomorphism $\pi_Z : \mathbb{Q}^n \to \mathbb{Q}^{|Z|}$ to the translation of $\mathrm{Mod}(\Lambda)$:

$$\mathrm{LinDep}_{\pi_Z,v}(\Lambda) := \{U \leq V \mid \pi_Z(\mathrm{Mod}(\Lambda) - v) \subseteq U^\perp\}$$

We call elements of $\mathrm{LinDep}_{\pi_Z,v}(\Lambda)$ *Z-dependencies*.

Disjuncts of a predicate set



Definition (Disjuncts of a predicate set)

For a formula φ and a set of predicates Γ, we call

$$\mathrm{DisjOf}_{\varphi}(\Gamma) := \mathrm{Sat}(\{\Gamma \cup \Omega \mid \Omega \in \mathrm{DNF}_{\varphi}[\top]\})$$

the set of disjuncts of Γ.

The covering algorithm – high level overview

Let $\Gamma \in \mathrm{Sat}(\mathrm{DNF}_{\varphi}[\varphi])$ be the given predicate set.

1. If Γ has only one satisfying assignment, return the decomposition of Γ.
2. Let $\Theta := \{p \in \Gamma \mid p \text{ is a } \Pi\text{-decomposition}\}$.
3. Compute a set of predicates together enforcing all Z-dependencies of Γ for every $Z \in \Pi$, and add that set to Θ.
4. Initialize the covering to be Θ.
5. While some disjunct
   $$\Omega \in \mathrm{DisjOf}_{\varphi}(\Theta)$$
   has more Z-dependencies than Γ for some $Z \in \Pi$:
   - Synthesize a *separating predicate q* expressing the Z-dependency present in Ω but absent in Γ (in particular, $\Omega \models q$).
   - Add $\neg q$ as a conjunct to the covering.
   - Recursively cover $\Gamma \cup \{q\}$ and add the result as a disjunct to the covering.
6. Convert the covering constructed so far into a disjunction over a set of predicate sets and remove those disjuncts that are unsatisfiable if taken in conjunction with Γ, and return the resulting covering.