Schaefer's theorem for graphs

Why to consult the infinite at times

Michael Pinsker

Université Diderot - Paris 7

Tel Aviv University, May 2012

Outline

Outline

- **Part I**
  Graph-SAT problems

Outline

- **Part I**

  Graph-SAT problems

- **Part II**

  Making the finite infinite

  CSPs of reducts of the random graph

Outline

- **Part I**

  Graph-SAT problems

- **Part II**

  Making the finite infinite

  CSPs of reducts of the random graph

- **Part III**

  Making the infinite finite

  Ramsey theory and canonical functions

Outline

- **Part I**

  Graph-SAT problems

- **Part II**

  Making the finite infinite

  CSPs of reducts of the random graph

- **Part III**

  Making the infinite finite

  Ramsey theory and canonical functions

- **Part IV**

  The Graph-SAT dichotomy

Outline

- **Part I**

  Graph-SAT problems

- **Part II**

  Making the finite infinite

  CSPs of reducts of the random graph

- **Part III**

  Making the infinite finite

  Ramsey theory and canonical functions

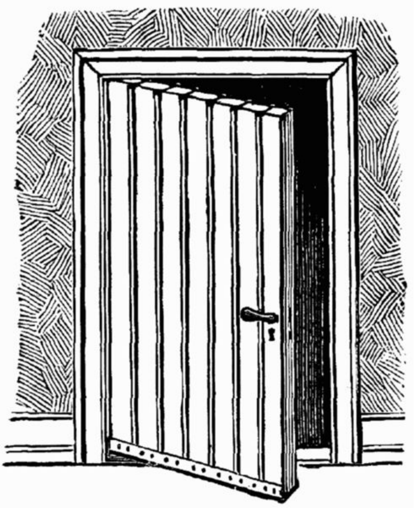
- **Part IV**

  The Graph-SAT dichotomy

- **Part V**

  The future

  CSPs over homogeneous structures



Part I

Graph-SAT problems

Boolean satisfiability problems

Boolean satisfiability problems

Let Ψ be a finite set of propositional formulas.

Boolean satisfiability problems

Let Ψ be a finite set of propositional formulas.

Computational problem: Boolean-SAT(Ψ)

INPUT:

- A set W of propositional variables, and
- statements $\phi_1, \ldots, \phi_n$ about the variables in W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable?

Boolean satisfiability problems

Let Ψ be a finite set of propositional formulas.

Computational problem: Boolean-SAT(Ψ)

INPUT:

- A set W of propositional variables, and
- statements $\phi_1, \ldots, \phi_n$ about the variables in W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable?

Computational complexity depends on Ψ. Always in NP.

Boolean satisfiability problems

Let Ψ be a finite set of propositional formulas.

Computational problem: Boolean-SAT(Ψ)

INPUT:

- A set W of propositional variables, and
- statements $\phi_1, \ldots, \phi_n$ about the variables in W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable?

Computational complexity depends on Ψ. Always in NP.

Theorem (Schaefer STOC'78)

Boolean-SAT(Ψ) is either in P or NP-complete, for all Ψ.

Boolean satisfiability problems

Let Ψ be a finite set of propositional formulas.

Computational problem: Boolean-SAT(Ψ)

INPUT:

- A set W of propositional variables, and
- statements $\phi_1, \ldots, \phi_n$ about the variables in W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable?

Computational complexity depends on Ψ. Always in NP.

Theorem (Schaefer STOC'78) 1139 citations on google scholar

Boolean-SAT(Ψ) is either in P or NP-complete, for all Ψ.

Graph satisfiability problems

Graph satisfiability problems

Let E be a binary relation symbol.

(Imagine: edge relation of an undirected graph.)

Let Ψ be a finite set of quantifier-free $\{E\}$-formulas.

Graph satisfiability problems

Let E be a binary relation symbol.

(Imagine: edge relation of an undirected graph.)

Let Ψ be a finite set of quantifier-free $\{E\}$-formulas.

Computational problem: Graph-SAT(Ψ)

INPUT:

- A finite set W of variables (vertices), and
- statements $\phi_1, \ldots, \phi_n$ about the elements of W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable in a graph?

Graph satisfiability problems

Let E be a binary relation symbol.

(Imagine: edge relation of an undirected graph.)

Let Ψ be a finite set of quantifier-free $\{E\}$-formulas.

Computational problem: Graph-SAT(Ψ)

INPUT:

- A finite set W of variables (vertices), and
- statements $\phi_1, \ldots, \phi_n$ about the elements of W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable in a graph?

Computational complexity depends on Ψ. Always in NP.

Graph satisfiability problems

Let E be a binary relation symbol.

(Imagine: edge relation of an undirected graph.)

Let Ψ be a finite set of quantifier-free $\{E\}$-formulas.

Computational problem: Graph-SAT(Ψ)

INPUT:

- A finite set W of variables (vertices), and
- statements $\phi_1, \ldots, \phi_n$ about the elements of W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable in a graph?

Computational complexity depends on Ψ. Always in NP.

Question

For which Ψ is Graph-SAT(Ψ) tractable?

Graph-SAT: Examples

Graph-SAT: Examples

Example 1 Let Ψ_1 only contain

$$\psi_1(x,y,z) := (E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ \vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ \vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z)) \ .$$

Graph-SAT: Examples

Example 1 Let Ψ_1 only contain

$$\begin{aligned}\psi_1(x,y,z) :=&(E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z)) \,.\end{aligned}$$

Graph-SAT(Ψ_1) is NP-complete.

Graph-SAT: Examples

Example 1 Let Ψ_1 only contain

$$\begin{aligned}\psi_1(x,y,z) :=& (E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ & \vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ & \vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z))\ .\end{aligned}$$

Graph-SAT(Ψ_1) is NP-complete.

Example 2 Let Ψ_2 only contain

$$\begin{aligned}\psi_2(x,y,z) :=& (E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ & \vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ & \vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z)) \\ & \vee (E(x,y) \wedge E(y,z) \wedge E(x,z))\ .\end{aligned}$$

Graph-SAT: Examples

Example 1 Let Ψ_1 only contain

$$\begin{aligned}\psi_1(x,y,z) :=&(E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z)) \ .\end{aligned}$$

Graph-SAT(Ψ_1) is NP-complete.

Example 2 Let Ψ_2 only contain

$$\begin{aligned}\psi_2(x,y,z) :=&(E(x,y) \wedge \neg E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge E(y,z) \wedge \neg E(x,z)) \\ &\vee (\neg E(x,y) \wedge \neg E(y,z) \wedge E(x,z)) \\ &\vee (E(x,y) \wedge E(y,z) \wedge E(x,z)) \ .\end{aligned}$$

Graph-SAT(Ψ_2) is in P.

Part II

Making the finite infinite

CSPs over the random graph

Graph formulas and reducts of the random graph

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

- universal, i.e., all finite graphs are induced subgraphs of G;

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

- universal, i.e., all finite graphs are induced subgraphs of G;
- homogeneous, i.e.,
  For all finite $A, B \subseteq G$, for all isomorphisms $i : A \to B$
  there exists $\alpha \in \mathrm{Aut}(G)$ extending i.

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

- universal, i.e., all finite graphs are induced subgraphs of G;
- homogeneous, i.e.,
  For all finite $A, B \subseteq G$, for all isomorphisms $i : A \to B$
  there exists $\alpha \in \mathrm{Aut}(G)$ extending i.

For a graph formula $\psi(x_1, \ldots, x_n)$, define a relation

$$R_\psi := \{(a_1, \ldots, a_n) \in V^n : \psi(a_1, \ldots, a_n)\}.$$

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

- universal, i.e., all finite graphs are induced subgraphs of G;
- homogeneous, i.e.,
  For all finite $A, B \subseteq G$, for all isomorphisms $i : A \to B$
  there exists $\alpha \in \mathrm{Aut}(G)$ extending i.

For a graph formula $\psi(x_1, \ldots, x_n)$, define a relation

$$R_\psi := \{(a_1, \ldots, a_n) \in V^n : \psi(a_1, \ldots, a_n)\}.$$

For a set Ψ of graph formulas, define a structure

$$\Gamma_\Psi := (V; (R_\psi : \psi \in \Psi)).$$

Graph formulas and reducts of the random graph

Let $G = (V; E)$ denote the random graph, i.e.,
the unique countably infinite graph which is

- universal, i.e., all finite graphs are induced subgraphs of G;
- homogeneous, i.e.,
  For all finite $A, B \subseteq G$, for all isomorphisms $i : A \to B$
  there exists $\alpha \in \mathrm{Aut}(G)$ extending i.

For a graph formula $\psi(x_1, \ldots, x_n)$, define a relation

$$R_\psi := \{(a_1, \ldots, a_n) \in V^n : \psi(a_1, \ldots, a_n)\}.$$

For a set Ψ of graph formulas, define a structure

$$\Gamma_\Psi := (V; (R_\psi : \psi \in \Psi)).$$

Γ_Ψ is a *reduct of* the random graph, i.e.,
a structure with a first-order definition in G.

Graph-SAT as CSP of a reduct of G

Graph-SAT as CSP of a reduct of G

An instance

- $W = \{w_1, \ldots, w_m\}$
- $\phi_1, \ldots, \phi_n$

of Graph-SAT(Ψ) has a positive solution $\leftrightarrow$
the sentence $\exists w_1, \ldots, w_m. \bigwedge_i \phi_i$ holds in Γ_Ψ.

Graph-SAT as CSP of a reduct of G

An instance

- $W = \{w_1, \ldots, w_m\}$
- $\phi_1, \ldots, \phi_n$

of Graph-SAT(Ψ) has a positive solution $\leftrightarrow$
the sentence $\exists w_1, \ldots, w_m. \bigwedge_i \phi_i$ holds in Γ_Ψ.

The decision problem
whether or not a given primitive positive sentence holds in Γ_Ψ
is called the Constraint Satisfaction Problem of Γ_Ψ (or CSP(Γ_Ψ)).

Graph-SAT as CSP of a reduct of G

An instance

- $W = \{w_1, \ldots, w_m\}$
- $\phi_1, \ldots, \phi_n$

of Graph-SAT(Ψ) has a positive solution $\leftrightarrow$
the sentence $\exists w_1, \ldots, w_m. \bigwedge_i \phi_i$ holds in Γ_Ψ.

The decision problem
whether or not a given primitive positive sentence holds in Γ_Ψ
is called the Constraint Satisfaction Problem of Γ_Ψ (or CSP(Γ_Ψ)).

So Graph-SAT(Ψ) and CSP(Γ_Ψ) are one and the same problem.

Why the random graph?

Why the random graph?

We have seen:

Classifying the complexity of all Graph-SAT problems is the same as classifying the complexity of CSPs of all reducts of G.

Why the random graph?

We have seen:

Classifying the complexity of all Graph-SAT problems is the same as classifying the complexity of CSPs of all reducts of G.

Note:

Could have used any universal graph!

Why the random graph?

We have seen:

Classifying the complexity of all Graph-SAT problems is the same as classifying the complexity of CSPs of all reducts of G.

Note:

Could have used any universal graph!

But:

G is the nicest universal graph.

Why the random graph?

We have seen:

Classifying the complexity of all Graph-SAT problems is the same as classifying the complexity of CSPs of all reducts of G.

Note:

Could have used any universal graph!

But:

G is the nicest universal graph.

Let's study CSP(Γ) for reducts Γ of G!

Primitive positive (pp) definability and polymorphisms

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Easy observation.

If $\Gamma \leq_{pp} \Delta$, then CSP(Γ) has a polynomial-time reduction to CSP(Δ).

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Easy observation.

If $\Gamma \leq_{pp} \Delta$, then CSP(Γ) has a polynomial-time reduction to CSP(Δ).

For finite $n \geq 1$, a function $f : \Gamma^n \to \Gamma$ is a *polymorphism* of Γ iff
for all relations R of Γ and all $r_1, \ldots, r_n \in R$ we have $f(r_1, \ldots, r_n) \in R$.

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Easy observation.

If $\Gamma \leq_{pp} \Delta$, then CSP(Γ) has a polynomial-time reduction to CSP(Δ).

For finite $n \geq 1$, a function $f : \Gamma^n \to \Gamma$ is a *polymorphism* of Γ iff
for all relations R of Γ and all $r_1, \ldots, r_n \in R$ we have $f(r_1, \ldots, r_n) \in R$.

Generalization of endomorphism, automorphism.

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Easy observation.

If $\Gamma \leq_{pp} \Delta$, then CSP(Γ) has a polynomial-time reduction to CSP(Δ).

For finite $n \geq 1$, a function $f : \Gamma^n \to \Gamma$ is a *polymorphism* of Γ iff
for all relations R of Γ and all $r_1, \ldots, r_n \in R$ we have $f(r_1, \ldots, r_n) \in R$.

Generalization of endomorphism, automorphism.

We write Pol(Γ) for the set of polymorphisms of Γ.

"Polymorphism clone of Γ"

Primitive positive (pp) definability and polymorphisms

For reducts Γ, Δ, set $\Gamma \leq_{pp} \Delta$ iff
every relation of Γ has a pp-definition from Δ.

Easy observation.

If $\Gamma \leq_{pp} \Delta$, then CSP(Γ) has a polynomial-time reduction to CSP(Δ).

For finite $n \geq 1$, a function $f : \Gamma^n \to \Gamma$ is a *polymorphism* of Γ iff
for all relations R of Γ and all $r_1, \ldots, r_n \in R$ we have $f(r_1, \ldots, r_n) \in R$.

Generalization of endomorphism, automorphism.

We write Pol(Γ) for the set of polymorphisms of Γ.

"Polymorphism clone of Γ"

Theorem (Bodirsky, Nešetřil '03). $\Gamma \leq_{pp} \Delta \;\leftrightarrow\; \mathrm{Pol}(\Delta) \subseteq \mathrm{Pol}(\Gamma)$.

The polymorphism strategy

The polymorphism strategy

Larger reducts $\rightarrow$ harder CSP

$\Gamma \leq_{pp} \Delta \quad \rightarrow \quad \mathrm{CSP}(\Gamma) \leq_{Poltime} \mathrm{CSP}(\Delta)$

The polymorphism strategy

Larger reducts $\rightarrow$ harder CSP

$\Gamma \leq_{pp} \Delta \quad \rightarrow \quad \mathrm{CSP}(\Gamma) \leq_{Poltime} \mathrm{CSP}(\Delta)$

Larger polymorphism clones $\rightarrow$ easier CSP

$\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Delta) \quad \rightarrow \quad \mathrm{CSP}(\Delta) \leq_{Poltime} \mathrm{CSP}(\Gamma)$

The polymorphism strategy

Larger reducts $\rightarrow$ harder CSP

$\Gamma \leq_{pp} \Delta \quad \rightarrow \quad \mathrm{CSP}(\Gamma) \leq_{\mathit{Poltime}} \mathrm{CSP}(\Delta)$

Larger polymorphism clones $\rightarrow$ easier CSP

$\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Delta) \quad \rightarrow \quad \mathrm{CSP}(\Delta) \leq_{\mathit{Poltime}} \mathrm{CSP}(\Gamma)$

Strategy:

(i) Prove hardness for certain reducts;

(ii) Prove that all reducts which do not pp-define any of these hard reducts are tractable.

The polymorphism strategy

Larger reducts $\rightarrow$ harder CSP

$\Gamma \leq_{pp} \Delta \quad \rightarrow \quad \mathrm{CSP}(\Gamma) \leq_{\mathit{Poltime}} \mathrm{CSP}(\Delta)$

Larger polymorphism clones $\rightarrow$ easier CSP

$\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Delta) \quad \rightarrow \quad \mathrm{CSP}(\Delta) \leq_{\mathit{Poltime}} \mathrm{CSP}(\Gamma)$

Strategy:

(i) Prove hardness for certain reducts;
(ii) Prove that all reducts which do not pp-define any of these hard reducts are tractable.

Reducts of (ii) have polymorphisms violating the relations of (i).

Polymorphisms provide algorithms.

Part III

Making the infinite finite

Canonical polymorphisms

Canonical functions

We have seen: Polymorphisms should prove tractability.

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0,1\}$ (Schaefer).

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0, 1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0, 1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Definition. A function $f : G \to G$ is *canonical* $\leftrightarrow$
whenever two pairs $(x, y), (u, v) \in G^2$ have the the same *type*,
then $(f(x), f(y))$ and $(f(u), f(v))$ have the same type as well.

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0, 1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Definition. A function $f : G \to G$ is *canonical* $\leftrightarrow$
whenever two pairs $(x, y), (u, v) \in G^2$ have the the same *type*,
then $(f(x), f(y))$ and $(f(u), f(v))$ have the same type as well.

Examples

- Function which switches edges and non-edges.

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0, 1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Definition. A function $f : G \to G$ is *canonical* $\leftrightarrow$
whenever two pairs $(x, y), (u, v) \in G^2$ have the the same *type*,
then $(f(x), f(y))$ and $(f(u), f(v))$ have the same type as well.

Examples

- Function which switches edges and non-edges.
- Injection onto complete subgraph of G.

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0,1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Definition. A function $f : G \to G$ is *canonical* $\leftrightarrow$
whenever two pairs $(x, y), (u, v) \in G^2$ have the the same *type*,
then $(f(x), f(y))$ and $(f(u), f(v))$ have the same type as well.

Examples

- Function which switches edges and non-edges.
- Injection onto complete subgraph of G.
- Constant function.

Canonical functions

We have seen: Polymorphisms should prove tractability.

True for CSP of finite structures, e.g. max on $\{0, 1\}$ (Schaefer).

How can we use an *infinite* polymorphism $f : \Gamma^n \to \Gamma$ in an algorithm?

Definition. A function $f : G \to G$ is *canonical* $\leftrightarrow$
whenever two pairs $(x, y), (u, v) \in G^2$ have the the same *type*,
then $(f(x), f(y))$ and $(f(u), f(v))$ have the same type as well.

Examples

- Function which switches edges and non-edges.
- Injection onto complete subgraph of G.
- Constant function.

Generalization of *canonical* to functions $f : G^n \to G$ possible.

Example. edge-max: $G^2 \to G$.

Canonical functions theorem

We wish to work with canonical polymorphisms.

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

For all finite graphs H
there exists a finite graph S such that
whenever the edges of S are colored with two colors
then there exists a copy of H in S on which the coloring is constant.

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

For all finite graphs H
there exists a finite graph S such that
whenever the edges of S are colored with two colors
then there exists a copy of H in S on which the coloring is constant.

Every function $f : G \to G$ induces a coloring of the edges of G.
Exploiting this further, one obtains:

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

For all finite graphs H
there exists a finite graph S such that
whenever the edges of S are colored with two colors
then there exists a copy of H in S on which the coloring is constant.

Every function $f : G \to G$ induces a coloring of the edges of G.
Exploiting this further, one obtains:

Theorem (roughly). If a polymorphism of Γ violates a relation R, then there exists a canonical polymorphism of Γ which violates R.

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

For all finite graphs H
there exists a finite graph S such that
whenever the edges of S are colored with two colors
then there exists a copy of H in S on which the coloring is constant.

Every function $f : G \to G$ induces a coloring of the edges of G.
Exploiting this further, one obtains:

Theorem (roughly). If a polymorphism of Γ violates a relation R,
then there exists a canonical polymorphism of Γ which violates R.

General modern proof uses topological dynamics, i.e.,
continuous group actions on compact topological spaces.

Canonical functions theorem

We wish to work with canonical polymorphisms.

Fact. G has the following Ramsey-type property:

For all finite graphs H
there exists a finite graph S such that
whenever the edges of S are colored with two colors
then there exists a copy of H in S on which the coloring is constant.

Every function $f : G \to G$ induces a coloring of the edges of G.
Exploiting this further, one obtains:

Theorem (roughly). If a polymorphism of Γ violates a relation R,
then there exists a canonical polymorphism of Γ which violates R.

General modern proof uses topological dynamics, i.e.,
continuous group actions on compact topological spaces.

Canonical functions are finite objects: functions on types!

Part IV

The Graph-SAT dichotomy

Complexity of CSP for reducts of G

Complexity of CSP for reducts of G

Theorem (Bodirsky, MP '10)

Let Γ be a reduct of the random graph. Then:

- Either Γ has one out of 17 canonical polymorphisms, and CSP(Γ) is tractable,
- or CSP(Γ) is NP-complete.

Complexity of CSP for reducts of G

Theorem (Bodirsky, MP '10)

Let Γ be a reduct of the random graph. Then:

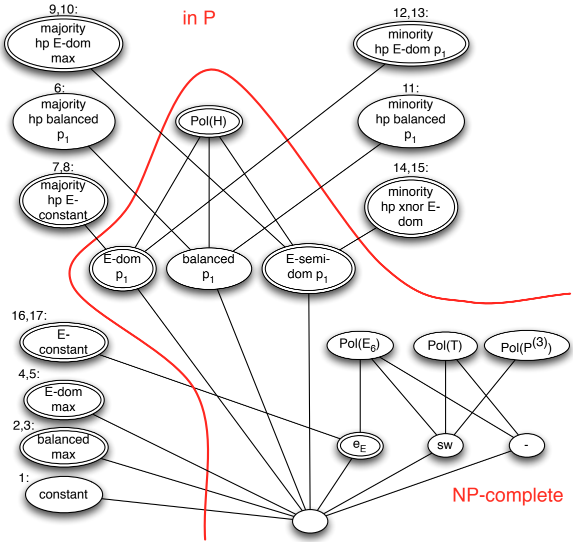
- Either Γ has one out of 17 canonical polymorphisms, and CSP(Γ) is tractable,
- or CSP(Γ) is NP-complete.

Theorem (Bodirsky, MP '10)

Let Γ be a reduct of the random graph. Then:

- Either Γ pp-defines one out of 4 hard relations, and CSP(Γ) is NP-complete,
- or CSP(Γ) is tractable.

The Graph-SAT dichotomy visualized



Theorem

The following 17 distinct clones are precisely the minimal tractable closed clones containing Aut(G):

1. The clone generated by a constant operation.
2. The clone generated by a balanced binary injection of type max.
3. The clone generated by a balanced binary injection of type min.
4. The clone generated by an E-dominated binary injection of type max.
5. The clone generated by an N-dominated binary injection of type min.
6. The clone generated by a function of type majority which is hyperplanely balanced and of type projection.
7. The clone generated by a function of type majority which is hyperplanely E-constant.
8. The clone generated by a function of type majority which is hyperplanely N-constant.
9. The clone generated by a function of type majority which is hyperplanely of type max and E-dominated.
10. The clone generated by a function of type majority which is hyperplanely of type min and N-dominated.

The Meta Problem

The Meta Problem

Meta-Problem of Graph-SAT(Ψ)

INPUT: A finite set Ψ of graph formulas.

QUESTION: Is Graph-SAT(Ψ) in P?

The Meta Problem

Meta-Problem of Graph-SAT(Ψ)

INPUT: A finite set Ψ of graph formulas.

QUESTION: Is Graph-SAT(Ψ) in P?

Theorem (Bodirsky, MP '10)

The Meta-Problem of Graph-SAT(Ψ) is decidable.

Graph satisfiability problems

Graph satisfiability problems

Let Ψ be a finite set of graph formulas.

Computational problem: Graph-SAT(Ψ)

INPUT:

- A set W of variables (vertices), and
- statements $\phi_1, \ldots, \phi_n$ about the elements of W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable in a graph?

Graph satisfiability problems

Let Ψ be a finite set of graph formulas.

Computational problem: Graph-SAT(Ψ)

INPUT:

- A set W of variables (vertices), and
- statements $\phi_1, \ldots, \phi_n$ about the elements of W, where each ϕ_i is taken from Ψ.

QUESTION: Is $\bigwedge_{1 \leq i \leq n} \phi_i$ satisfiable in a graph?

Theorem (Bodirsky, MP '10)

Graph-SAT(Ψ) is either in P or NP-complete, for all Ψ.

Part V

The future

CSPs over homogeneous structures

Amalgamation classes

Amalgamation classes

Graph-SAT(Ψ): Is there a finite graph such that... (graph constraints)

Amalgamation classes

Graph-SAT(Ψ): Is there a finite graph such that... (graph constraints)

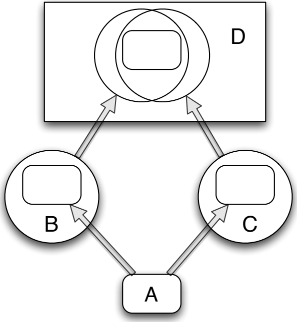
Linorder-SAT(Ψ): Is there a linear order such that... (order constraints, "temporal constraints")

Amalgamation classes

Graph-SAT(Ψ): Is there a finite graph such that... (graph constraints)

Linorder-SAT(Ψ): Is there a linear order such that... (order constraints, "temporal constraints")

The classes of finite graphs and linear orders are *amalgamation classes*.



Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders
- Metric spaces with finite set of distances

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders
- Metric spaces with finite set of distances
- Tournaments

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders
- Metric spaces with finite set of distances
- Tournaments
- K_n-free graphs

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders
- Metric spaces with finite set of distances
- Tournaments
- K_n-free graphs
- Ordered graphs

Amalgamation classes have homogeneous limit

Theorem (Fraïssé)

- If $\mathcal{C}$ is a countable class of structures closed under substructures which has amalgamation, then there exists a unique structure $\mathfrak{C}$ with age $\mathcal{C}$ which is homogeneous.
- The age of a homogeneous structure is an amalgamation class.

$\mathfrak{C}$ is called the Fraïssé limit of $\mathcal{C}$. Example $(\mathbb{Q}, <)$.

Further amalgamation classes.

- Partial orders
- Metric spaces with finite set of distances
- Tournaments
- K_n-free graphs
- Ordered graphs
- Permutations

General method for amalgamation classes

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).
4. Tractability is implied by presence of polymorphisms in Pol(Γ).

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).
4. Tractability is implied by presence of polymorphisms in Pol(Γ).
5. If $\mathfrak{C}$ is Ramsey, then even implied by canonical polymorphisms. These are essentially functions on finite sets.

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).
4. Tractability is implied by presence of polymorphisms in Pol(Γ).
5. If $\mathfrak{C}$ is Ramsey, then even implied by canonical polymorphisms. These are essentially functions on finite sets.
6. Adaptations of the algorithms for these finite functions.

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).
4. Tractability is implied by presence of polymorphisms in Pol(Γ).
5. If $\mathfrak{C}$ is Ramsey, then even implied by canonical polymorphisms. These are essentially functions on finite sets.
6. Adaptations of the algorithms for these finite functions.
7. Hardness proofs: by reduction of known finite CSPs.

General method for amalgamation classes

1. Given amalgamation class $\mathcal{C}$, consider all $\mathcal{C}$-SAT problems.
2. Every problem $\mathcal{C}$-SAT(Ψ) translates into CSP(Γ_Ψ), where Γ_Ψ is a reduct of the (homogeneous infinite) Fraïssé limit $\mathfrak{C}$ of $\mathcal{C}$.
3. For each reduct Γ of this limit $\mathfrak{C}$, the complexity of CSP(Γ) is captured by the polymorphism clone Pol(Γ).
4. Tractability is implied by presence of polymorphisms in Pol(Γ).
5. If $\mathfrak{C}$ is Ramsey, then even implied by canonical polymorphisms. These are essentially functions on finite sets.
6. Adaptations of the algorithms for these finite functions.
7. Hardness proofs: by reduction of known finite CSPs.

   Modern method: exposing a continuous homomorphism from Pol(Γ) to the projection clone on $\{0, 1\}$. *Topological Birkhoff.*

Future research

Future research

(a) Find (improve "making finite"):
Meta-method for translating *tractability of the type function* of a canonical function into *tractability of the canonical function*.

Future research

(a) Find (improve "making finite"):
Meta-method for translating *tractability of the type function* of a canonical function into *tractability of the canonical function*.

(b) Prove (complete "making finite"):
If the dichotomy / tractability conjecture for finite structures holds, then it holds for all reducts of homogeneous Ramsey structures.

Future research

(a) Find (improve “making finite”):
Meta-method for translating *tractability of the type function* of a canonical function into *tractability of the canonical function*.

(b) Prove (complete “making finite”):
If the dichotomy / tractability conjecture for finite structures holds, then it holds for all reducts of homogeneous Ramsey structures.

(c) Answer (improve “making infinite”):
Can all homogeneous structures be made Ramsey by adding finitely many relations?

Future research

(a) Find (improve “making finite”):
Meta-method for translating *tractability of the type function* of a canonical function into *tractability of the canonical function*.

(b) Prove (complete “making finite”):
If the dichotomy / tractability conjecture for finite structures holds, then it holds for all reducts of homogeneous Ramsey structures.

(c) Answer (improve “making infinite”):
Can all homogeneous structures be made Ramsey by adding finitely many relations?

(d) Apply method to:
- finite partial orders – **Poset-SAT(Ψ)**
- finite Boolean algebras – “set constraints”
etc.

References

Graph-SAT dichotomy:

Schaefer's theorem for graphs

by Manuel Bodirsky and Michael Pinsker,

Proceedings of STOC, 2011. Full version on arXiv.

Canonical functions method:

Reducts of Ramsey structures

by Manuel Bodirsky and Michael Pinsker,

AMS Contemporary Mathematics, 2011. Preprint on arXiv.

Modern hardness proofs:

Topological Birkhoff

by Manuel Bodirsky and Michael Pinsker,

Preprint on arXiv, 2012.