The Complexity of the Constraint Satisfaction Problem.

ISG-CS Mini-Conference

Professor David Cohen

Department of Computer Science
Royal Holloway College,
University of London

RHUL University, March, 2011.

Outline

1. **Introducing the CSP**
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. **CSP Complexity**
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. **Hybrid Tractability**
   - Tractable Independent Set Problem
   - A General Theory
4. **CSP Becomes VCSP**

Outline

1. **Introducing the CSP**
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. **CSP Complexity**
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. **Hybrid Tractability**
   - Tractable Independent Set Problem
   - A General Theory
4. **CSP Becomes VCSP**

What is a Constraint?

A constraint has two parts:

- A *list of variables* that are constrained
- A *set of allowed combinations of values* for these variables

What is a Constraint?

A constraint has two parts:

- A *list of variables* that are constrained
- A *set of allowed combinations of values* for these variables

What is a Constraint?

A constraint has two parts:

- A *list of variables* that are constrained
  - which we call the scope
- A *set of allowed combinations of values* for these variables

What is a Constraint?

A constraint has two parts:

- A *list of variables* that are constrained
  - which we call the scope
- A *set of allowed combinations of values* for these variables
  - which we call the relation

Examples of Constraints

Two constraints:

- with their scopes
- and relations

A traditional graph theory constraint

Variables x and y must be different colours

An important temporal constraint

The CSP talk must finish before the next talk

Examples of Constraints

Two constraints:

- with their scopes
- and relations

A traditional graph theory constraint

Variables x and y must be different colours

An important temporal constraint

The CSP talk must finish before the next talk

Examples of Constraints

Two constraints:

- with their scopes
- and relations

A traditional graph theory constraint

Variables x and y must be different colours

An important temporal constraint

The CSP talk must finish before the next talk

Outline

1. **Introducing the CSP**
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. **CSP Complexity**
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. **Hybrid Tractability**
   - Tractable Independent Set Problem
   - A General Theory
4. **CSP Becomes VCSP**

The classical definition of the CSP

Definition

An *instance* of the constraint satisfaction problem (CSP) is a triple (V, D, C) where:

- V is a set of variables
- D is a set of possible domain values
- C is a a set of constraints

Each constraint $c \in C$ is a pair (s, R) where:

- s is the *scope* (list of variables constrained)
- R is the *relation* (set of allowed combinations of values)

A *solution* is a mapping $\sigma : V \to D$ which satisfies all the constraints in $C : \forall (s, R) \in C, \sigma(s) \in R$

The classical definition of the CSP

Definition

An *instance* of the constraint satisfaction problem (CSP) is a triple (V, D, C) where:

- V is a set of variables
- D is a set of possible domain values
- C is a a set of constraints

Each constraint $c \in C$ is a pair (s, R) where:

- s is the *scope* (list of variables constrained)
- R is the *relation* (set of allowed combinations of values)

A *solution* is a mapping $\sigma : V \to D$ which satisfies all the constraints in $C : \forall (s, R) \in C, \sigma(s) \in R$

The classical definition of the CSP

Definition

An *instance* of the constraint satisfaction problem (CSP) is a triple (V, D, C) where:

- V is a set of variables
- D is a set of possible domain values
- C is a a set of constraints

Each constraint $c \in C$ is a pair (s, R) where:

- s is the *scope* (list of variables constrained)
- R is the *relation* (set of allowed combinations of values)

A *solution* is a mapping $\sigma : V \to D$ which satisfies all the constraints in C : $\forall (s, R) \in C, \sigma(s) \in R$

The classical definition of the CSP

Definition

An *instance* of the constraint satisfaction problem (CSP) is a triple (V, D, C) where:

- V is a set of variables
- D is a set of possible domain values
- C is a a set of constraints

Each constraint $c \in C$ is a pair (s, R) where:

- s is the *scope* (list of variables constrained)
- R is the *relation* (set of allowed combinations of values)

A *solution* is a mapping $\sigma : V \to D$ which satisfies all the constraints in $C : \forall (s, R) \in C, \sigma(s) \in R$

Splitting a CSP into two parts

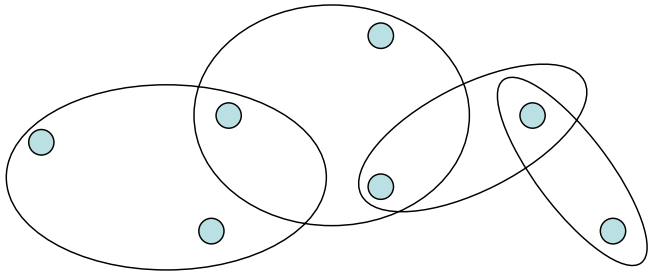
Any CSP instance can be split into two parts:

The structure of the constraints

The nature of the constraints

Splitting a CSP into two parts

Any CSP instance can be split into two parts:

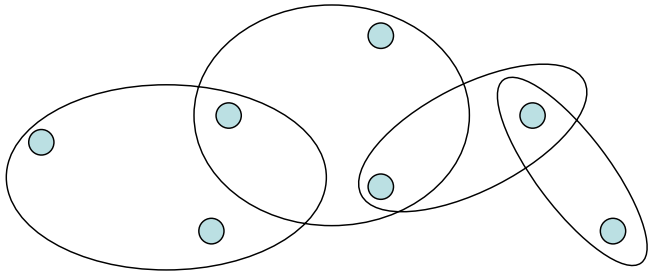


The nature of the constraints

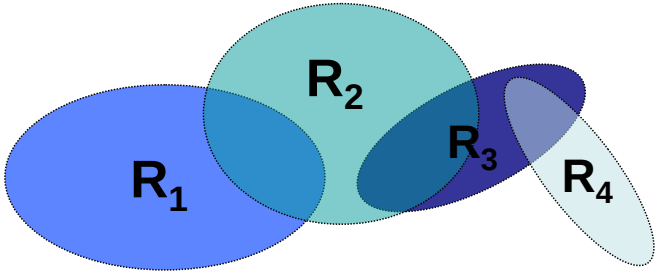
The structure of the constraints

Splitting a CSP into two parts

Any CSP instance can be split into two parts:

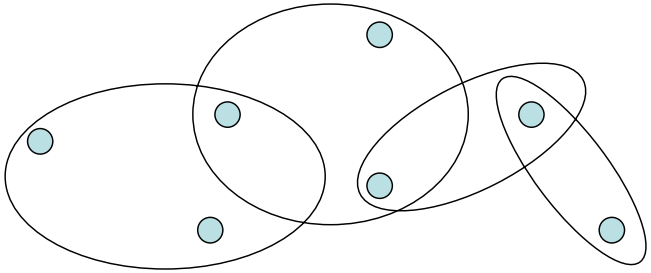


The structure of the constraints



The nature of the constraints

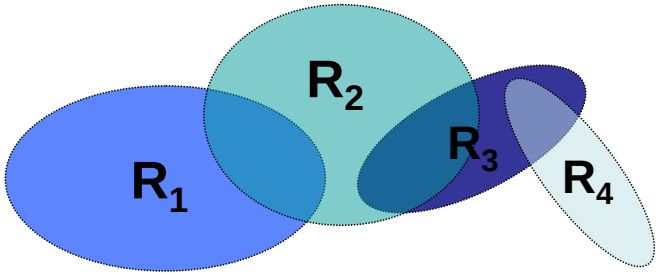
Another definition of a CSP



The structure of the constraints

- this is a *relational structure*

$$(V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints

- this is a *relational structure*

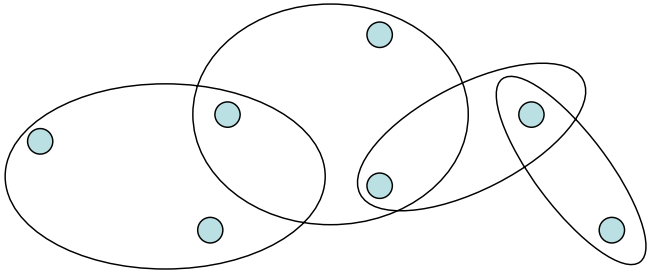
$$(D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

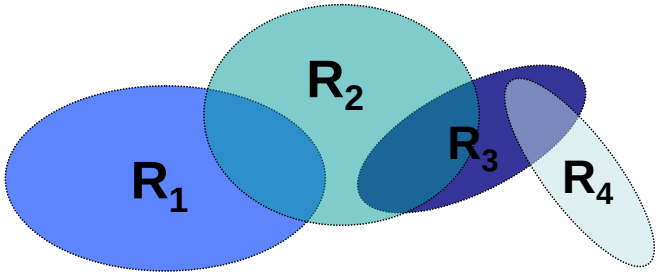
A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

Another definition of a CSP



The structure of the constraints
- this is a *relational structure*

$$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints
- this is a *relational structure*

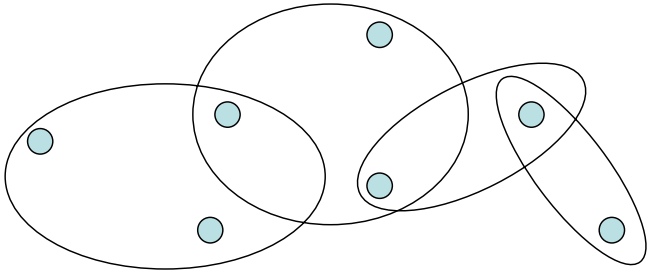
$$\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

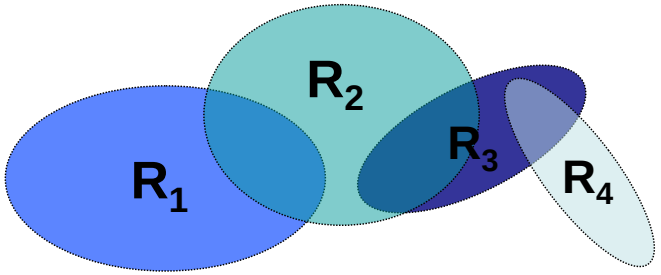
A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

Another definition of a CSP



The structure of the constraints
- this is a *relational structure*

$$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints
- this is a *relational structure*

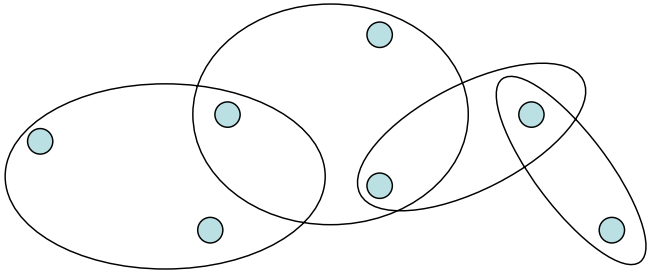
$$\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

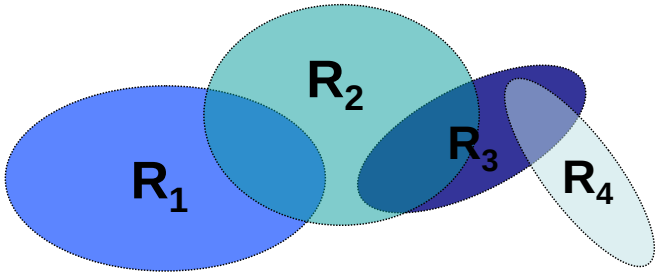
A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

Another definition of a CSP



The structure of the constraints
- this is a *relational structure*

$$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints
- this is a *relational structure*

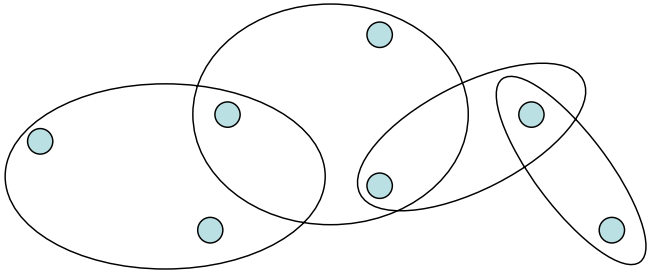
$$\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

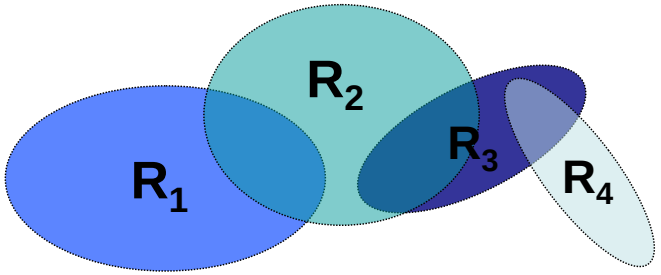
A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

Another definition of a CSP



The structure of the constraints
- this is a *relational structure*

$$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints
- this is a *relational structure*

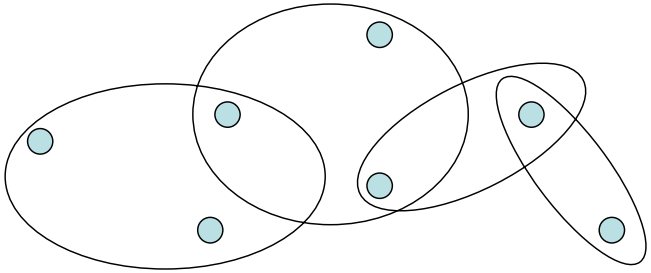
$$\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

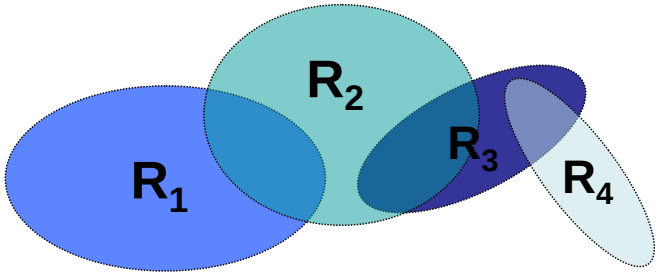
A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

Another definition of a CSP



The structure of the constraints
- this is a *relational structure*

$$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$$



The nature of the constraints
- this is a *relational structure*

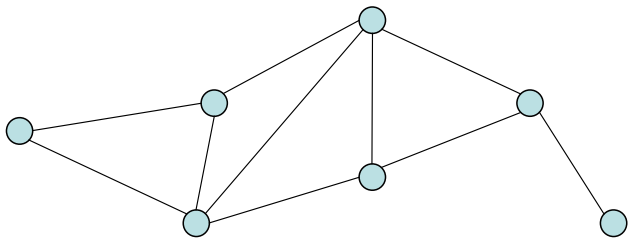
$$\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$$

Definition

An *instance* of the constraint satisfaction problem (CSP) is a pair of similar relational structures $(\mathcal{S}_1, \mathcal{S}_2)$

A *solution* is a homomorphism $h : \mathcal{S}_1 \to \mathcal{S}_2$

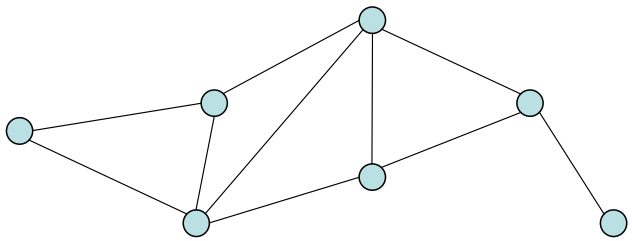
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

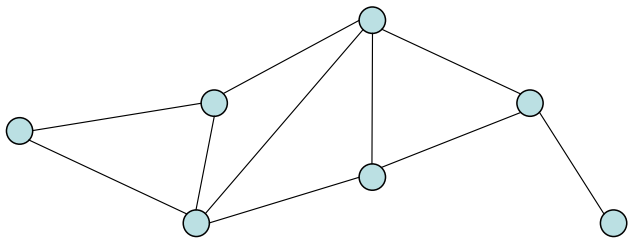
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

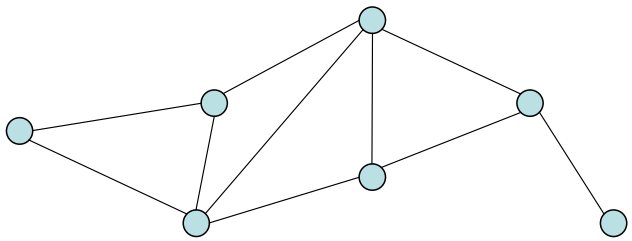
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

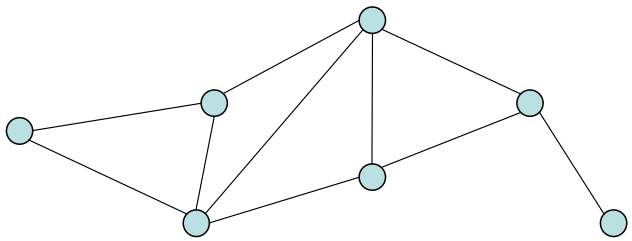
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

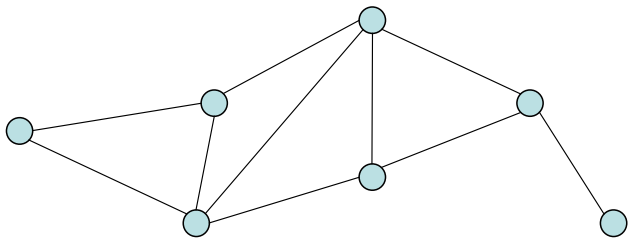
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP...

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

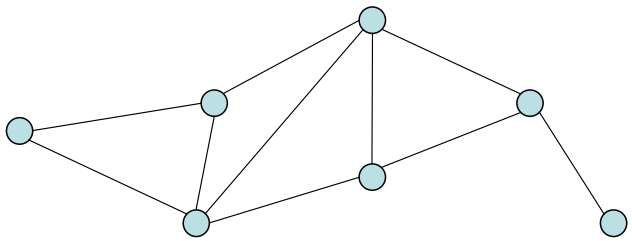
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?}$

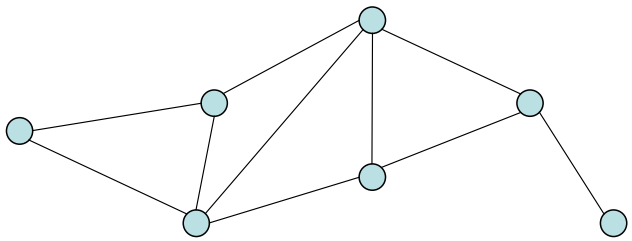
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?} (\{1, \ldots, k\}, \neq)$

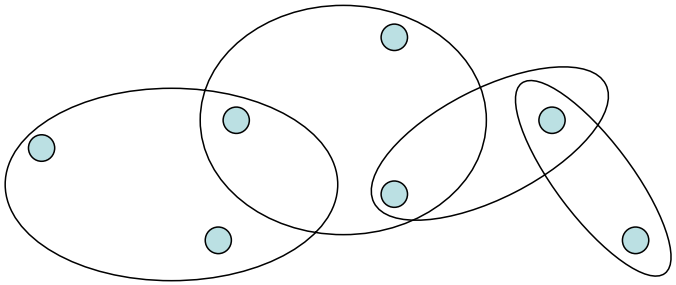
An example: Graph colouring as a CSP



The problem of finding a proper k-colouring for the vertices of a graph (N, E) can be expressed as a CSP…

- $(V, D, C) = (N, \{1, \ldots, k\}, \{((v_1, v_2), \neq) \mid \{v_1, v_2\} \in E\})$
- $(N, E) \xrightarrow{?} K_k$

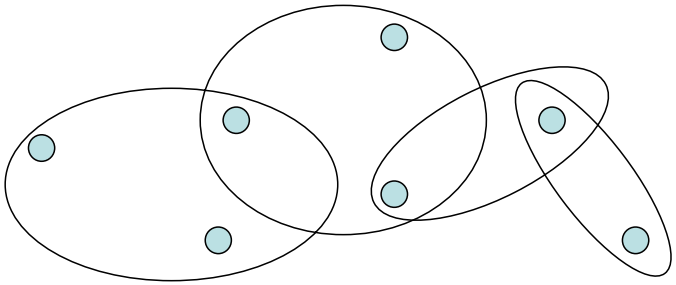
An example: Satisfiability as a CSP



The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

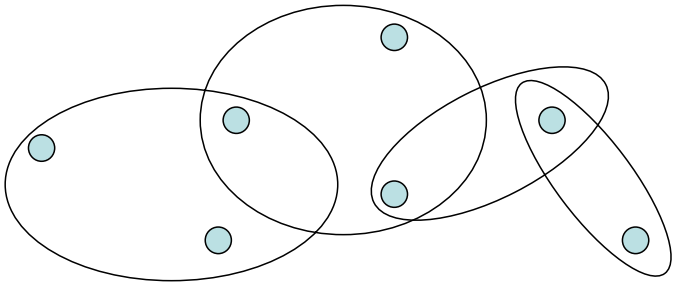
An example: Satisfiability as a CSP



The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

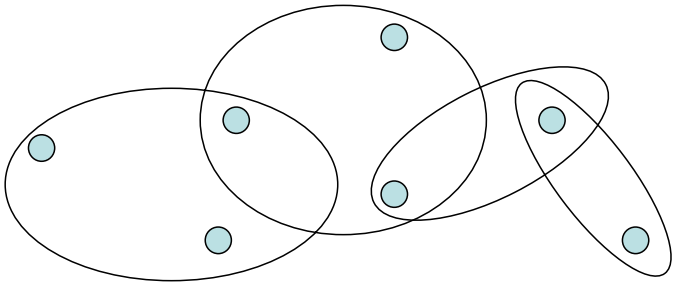
An example: Satisfiability as a CSP



The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

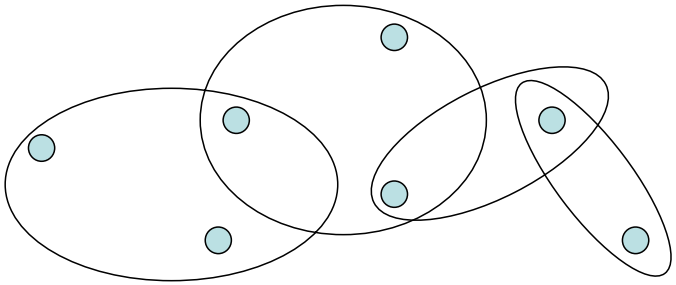
An example: Satisfiability as a CSP



The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

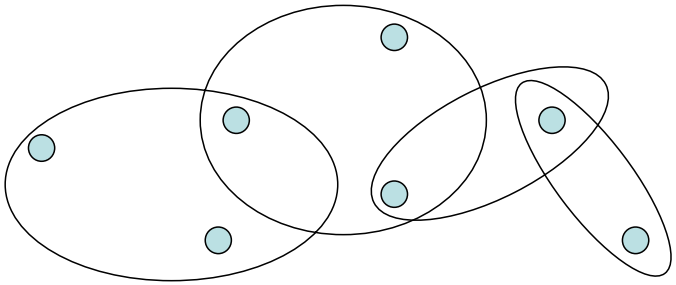
An example: Satisfiability as a CSP



The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

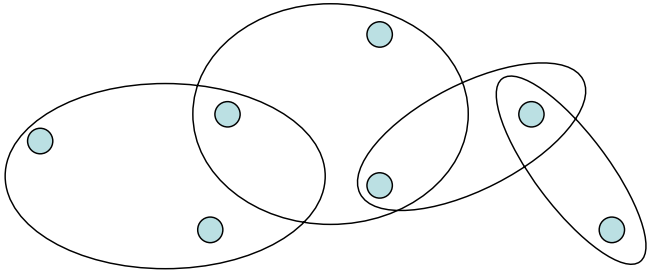
An example: Satisfiability as a CSP

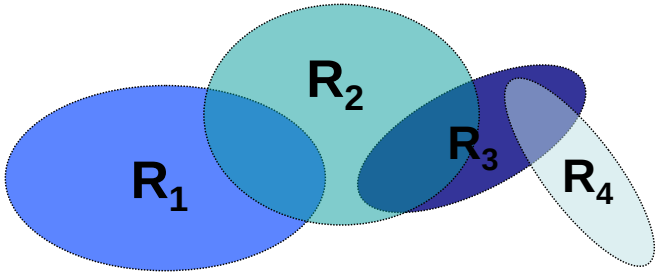


The problem of finding a satisfying assignment for a set of Boolean clauses $c_1, \ldots, c_m$ can be expressed as a CSP...

- $(V, D, C) = (\mathrm{Vars}(c_1, \ldots, c_m), \{0, 1\}, \{(\mathrm{Vars}(c_i), \mathrm{Models}(c_i)) \mid i = 1, \ldots, m\})$
- $(\mathrm{Vars}(c_1, \ldots, c_m), E_1, E_2, \ldots, E_8) \xrightarrow{?} (\{0, 1\}, R_1, R_2, \ldots, R_8)$

Combining the two parts of a CSP





$\mathcal{S}_1 = (V, E_1, E_2, \ldots, E_m)$ $\mathcal{S}_2 = (D, R_1, R_2, \ldots, R_m)$

To allow a wider range of descriptive possibilities we combine these two structures to form their product: $\mathcal{S}_1 \times \mathcal{S}_2$

If we extend both structures with the equality relation, and complement the relations of $\mathcal{S}_2$, then their product is a structure called the microstructure complement

The microstructure complement of a CSP

Definition

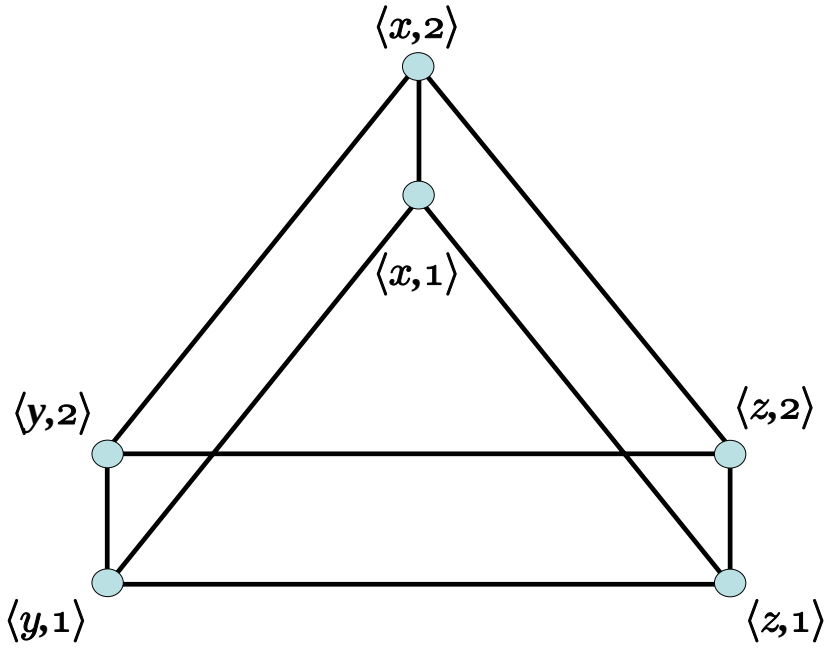
The microstructure complement of a CSP (V, D, C) is the relational structure $(V \times D, E)$ where $((v_1, d_1), (v_2, d_2), \ldots, (v_k, d_k)) \in E$ if and only if:

- $(v_1, v_2, \ldots, v_k)$ is the scope of some constraint $c \in C$ which *disallows* the assignment $(d_1, d_2, \ldots d_k)$; *or*
- $k = 2$, $v_1 = v_2$ and $d_1 \neq d_2$.

An example: Graph 2-colouring

$$F = (x \neq y) \wedge (y \neq z) \wedge (x \neq z)$$

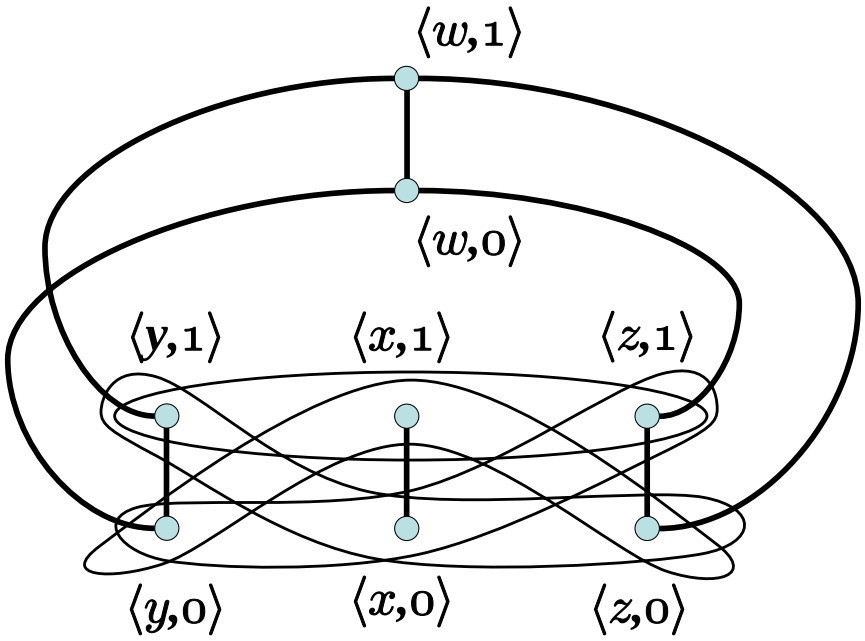
The microstructure complement of this CSP instance is:



An example: Linear equations modulo 2

$$F = (x + y + z \equiv 0) \wedge (w + y \equiv 1) \wedge (w + z \equiv 0) \pmod 2$$

The microstructure complement of this CSP instance is:



Outline

1. Introducing the CSP
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. CSP Complexity
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. Hybrid Tractability
   - Tractable Independent Set Problem
   - A General Theory
4. CSP Becomes VCSP

The CSP is NP-Hard

Testing is easy

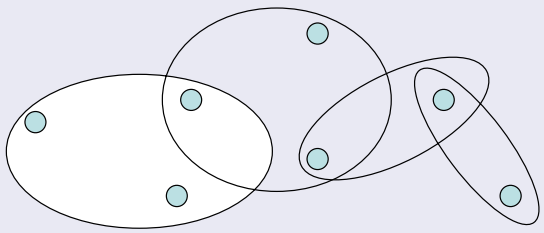
We can test any putative homomorphism (variable assignment) to see it meets all constraints in linear time.

The CSP is NP-Hard

Testing is easy

We can test any putative homomorphism (variable assignment) to see it meets all constraints in linear time.

CSP restricted to domain size 2 is NP-hard



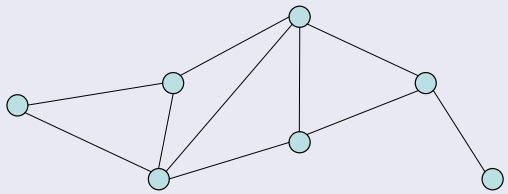
3-SAT: The problem of finding a satisfying assignment for a set of Boolean 3-clauses can be expressed as a CSP…

The CSP is NP-Hard

Testing is easy

We can test any putative homomorphism (variable assignment) to see it meets all constraints in linear time.

CSP restricted to arity 2 constraints is NP-hard



3-Colouring: The problem of finding a proper 3-colouring for the vertices of a graph can be expressed as a CSP…

Outline

1. Introducing the CSP
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. CSP Complexity
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. Hybrid Tractability
   - Tractable Independent Set Problem
   - A General Theory
4. CSP Becomes VCSP

CSP Languages

Restricting the Right Hand Side

- A set Γ of *allowed relations* is called a constraint language.
- Some (quite small) languages are sufficient to express interesting problems (e.g. Linear Equations, 3-SAT, 3-Colouring, Scheduling, Line drawing interpretation,Temporal Logics, Interval Logics. . .).
- For some languages we find effective algorithms.
- For each such algorithm we characterise the (maximal) language that it works on, to build an “algorithm chooser”.

CSP Languages

RHS == Language == Clones == Algebra

- We observed (some 15 years ago) that several maximal tractable languages. . . are characterised by very simple algebraic properties (clones of polymorphisms).
- We proved that every tractable language has this property: So we characterise tractable algebras instead.
- This is a simpler problem because so much is known about algebras!

Languages are Done

We are so close to finishing. . . just one polymorphism left

The following result shows that it remains to prove that Γ is tractable when it has a cyclic (or a Siggers') polymorphism

Theorem

(Siggers, 2009) A finite algebra has a WNU term operation iff it has a 4-ary idempotent term operation f satisfying

$$f(y, x, y, z) = f(x, y, z, x).$$

Outline

1. Introducing the CSP
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. CSP Complexity
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. Hybrid Tractability
   - Tractable Independent Set Problem
   - A General Theory
4. CSP Becomes VCSP

CSP Structure

Restricting the Left Hand Side

- We can limit the hypergraph (or relational) structures that are allowed.
- Such structural restrictions are natural and reasonable in practical applications.
- All tractable structural classes are based upon some notion of the *width of a relational structure*.

Structures are Done

All interesting structural classes are known

The following results show us everything we might want to know:

treewidth For classes characterised by *Primal graphs* (Grohe, 2003)

hypertree width For *tree decompositions* (Gottlob,2007)

treewidth of core For *bounded arity* (Grohe,2007)

submodular width For *FPT* (parameter: no. vars) (Marx, 2010)

The Structural Gap

Between hypertree and submodular Width

The most general known tractable structural property is bounded fractional hypertree width (Grohe, Marx, 2007).
This is better than hypertree width and worse than submodular width.
Where exactly are the *boundaries of structural tractability?*

The Structural Gap

Between hypertree and submodular Width

The most general known tractable structural property is bounded fractional hypertree width (Grohe, Marx, 2007).
This is better than hypertree width and worse than submodular width.
Where exactly are the *boundaries of structural tractability?*

Key Motivation/Timeliness

We believe the following

- Languages (RHS) are mostly done and its gotten too hard/esoteric.
- Structure (LHS) is mostly done and its gotten too hard/esoteric.
- Every tractable class has a definition in terms of its microstructure.
- These are (very like) graph structures.
- We can bring in a whole lot of new theory.

Outline

1. Introducing the CSP
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. CSP Complexity
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. Hybrid Tractability
   - Tractable Independent Set Problem
   - A General Theory
4. CSP Becomes VCSP

Perfect Graphs, Salamon 2008

Motivation

- Solving CSPs is equivalent to finding independent sets (cliques) in their microstructure (complement).
- The independent set and clique problems are tractable for triangulated graphs.
- This explains why CSP instances whose microstructure (complement) is triangulated are tractable

Definition

A class of CSP instances whose microstructure complements have a tractable independent set problem are tractable.

This is a generalised explanation for the tractability of instances of the AllDiff+Unary class.

Outline

1. Introducing the CSP
   - What is a Constraint?
   - What is the Constraint Satisfaction Problem (CSP)?
2. CSP Complexity
   - Hardness
   - Tractable Languages
   - Structural Tractability
3. Hybrid Tractability
   - Tractable Independent Set Problem
   - A General Theory
4. CSP Becomes VCSP

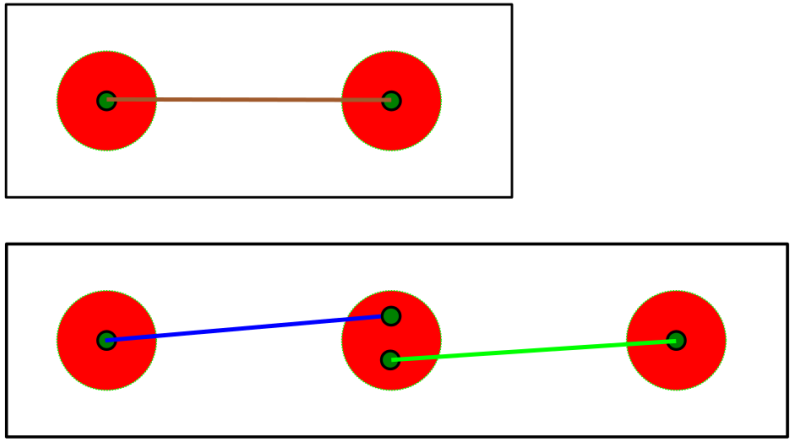
Forbidding certain embeddings in the microstructure complement

Negative Patterns

Can we capture tractable classes by specifying forbidden homomorphisms from certain microstructure complements. Here we insist that distinct variables map to distinct variables.

Example

Of course we can just forbid the single edge from embedding, or a pair of linked edges.



Pivots: Characterising Tractability

Theorem

A set $\mathcal{X}$ of connected patterns is tractable only if there is some $\chi \in \mathcal{X}$ that embeds into some $\textsc{Pivot}(r)$.

Theorem

$\textsc{Pivot}(1)$ *is tractable and any disjoint union of* $\textsc{SepPivot}(r)$ *patterns is tractable.*

Conjecture

A finite set $\mathcal{X}$ of connected patterns is tractable if and only if there is some $\chi \in \mathcal{X}$ that embeds into some $\textsc{Pivot}(r)$.

Key Motivation/Timeliness

We believe the following

- Languages (RHS) are mostly done and its gotten too hard/esoteric.
- Structure (LHS) is mostly done and its gotten too hard/esoteric.
- Every tractable class has a definition in terms of its microstructure.
- These are (very like) graph structures.
- We can bring in a whole lot of new theory.

Formally: A CSP becomes a VCSP

Definition

A CSP is a triple $\langle V, D, C \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \rho \rangle$ where
  - $\sigma \in V^*$ is the scope
  - $\rho \subseteq D^{|\sigma|}$ is the relation

Formally: A CSP becomes a VCSP

Definition

A CSP is a triple $\langle V, D, C \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \rho \rangle$ where
  - $\sigma \in V^*$ is the scope
  - $\rho \subseteq D^{|\sigma|}$ is the relation

Definition

An assignment is a mapping $s : V \mapsto D$.

Formally: A CSP becomes a VCSP

Definition

A CSP is a triple $\langle V, D, C \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \rho \rangle$ where
  - $\sigma \in V^*$ is the scope
  - $\rho \subseteq D^{|\sigma|}$ is the relation

Definition

A solution is an assignment that satisfies all constraints.
Formally:

$$\forall \langle \sigma, \rho \rangle \in C, s(\sigma) \in \rho.$$

Formally: A CSP becomes a VCSP

Definition

A CSP is a triple $\langle V, D, C \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \rho \rangle$ where
  - $\sigma \in V^*$ is the scope
  - ρ is a mapping from $D^{|\sigma|}$ to $\{0, \infty\}$ is the relation

Definition

A solution is an assignment that satisfies all constraints.
Formally:

$$\forall \langle \sigma, \rho \rangle \in C, \rho(s(\sigma)) = 0.$$

Formally: A CSP becomes a VCSP

Definition

A VCSP is a four tuple $\langle V, D, C, \Omega \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \phi \rangle$ where
  - $\sigma \in V^*$ is the scope
  - ϕ is a mapping from $D^{|\sigma|}$ to Ω called the cost function

Formally: A CSP becomes a VCSP

Definition

A VCSP is a four tuple $\langle V, D, C, \Omega \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \phi \rangle$ where
  - $\sigma \in V^*$ is the scope
  - ϕ is a mapping from $D^{|\sigma|}$ to Ω called the cost function

Definition

The valuation structure Ω, is ordered, with a 0 and ∞ and an AC aggregation $\oplus$, where $\alpha \geq \beta, \gamma \in \Omega$ and $\alpha \oplus \gamma \geq \beta \oplus \gamma$.

Formally: A CSP becomes a VCSP

Definition

A VCSP is a four tuple $\langle V, D, C, \Omega \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \phi \rangle$ where
  - $\sigma \in V^*$ is the scope
  - ϕ is a mapping from $D^{|\sigma|}$ to Ω called the cost function

Definition

An assignment is a mapping $s : V \mapsto D$. Its cost is given by

$$\mathrm{Cost}_p(s) = \bigoplus_{\langle \sigma, \phi \rangle \in C} \phi(s(\sigma)).$$

Formally: A CSP becomes a VCSP

Definition

A VCSP is a four tuple $\langle V, D, C, \Omega \rangle$ where

- V is a finite set of variables
- D is a finite domain
- C is a finite set of constraints.
  Each constraint is a pair $\langle \sigma, \phi \rangle$ where
  - $\sigma \in V^*$ is the scope
  - ϕ is a mapping from $D^{|\sigma|}$ to Ω called the cost function

Definition

A solution is an assignment with minimal cost.

Complexity, Expressibility, Galois, Submodularity

We have begun

There is a close association between the **complexity/expressibility** of VCSP languages and a so called fractional clone.
This **association** is via a well defined Galois connection - just as the *lattice of clones* is associated with the complexity/expressibility of CSP languages.
We are trying to use this theory to understand the *structure/expressibility* of **submodular optimisation problems**.

Complexity, Expressibility, Galois, Submodularity

Good Work

- Takhanov 2009: A dichotomy theorem for the general minimum cost homomorphism problem.
- Cohen et al: Binary submodular languages are cubic.
- Cohen et al: A dichotomy for Boolean VCSP languages.
- Cooper et al: Arc-consistency for VCSPs
- etc.,