The Constraint Satisfaction Problem

Libor Barto, Matt Valeriote, Ross Willard

Charles University, McMaster University, University of Waterloo

4 July, 2011

Sudoku

	6	3		4	5		2	9
5				9	2		1	8
			6					
	8			7	9	1		
							8	2
		2				4	6	
2								1
3		6			8	5		
			1		3		7	

Sudoku

x_{11}	6	3	x_{14}	4	5	x_{17}	2	9
5	x_{22}	x_{23}	x_{24}	9	2	x_{27}	1	8
x_{31}	x_{32}	x_{33}	6	x_{35}	x_{36}	x_{37}	x_{38}	x_{39}
x_{41}	8	x_{43}	x_{44}	7	9	1	x_{48}	x_{49}
x_{51}	x_{52}	x_{53}	x_{54}	x_{55}	x_{56}	x_{57}	8	2
x_{61}	x_{62}	2	x_{64}	x_{65}	x_{66}	4	6	x_{69}
2	x_{72}	x_{73}	x_{74}	x_{75}	x_{76}	x_{77}	x_{78}	1
3	x_{82}	6	x_{84}	x_{85}	8	5	x_{88}	x_{89}
x_{91}	x_{92}	x_{93}	1	x_{95}	3	x_{97}	7	x_{99}

Sudoku

An alternate formulation as a decision problem

Is there a way to assign elements from $\{1,2,3,4,5,6,7,8,9\}$ to each variable x_{ij} so that for each i:

$$\begin{aligned}
(x_{i1}, x_{i2}, \ldots, x_{i9}) &\in Sym\{1,2,3,4,5,6,7,8,9\} \\
(x_{1i}, x_{2i}, \ldots, x_{9i}) &\in Sym\{1,2,3,4,5,6,7,8,9\} \\
(x_{11}, x_{12}, \ldots, x_{33}) &\in Sym\{1,2,3,4,5,6,7,8,9\} \\
(x_{14}, x_{15}, \ldots, x_{36}) &\in Sym\{1,2,3,4,5,6,7,8,9\} \\
&\vdots \\
(x_{77}, x_{78}, \ldots, x_{99}) &\in Sym\{1,2,3,4,5,6,7,8,9\} \\
x_{12} &\in \{6\} \\
x_{13} &\in \{3\} \\
&\vdots
\end{aligned}$$

Boolean Satisfiability

Problem:

Given a propositional formula Φ, is there a truth assignment that satisfies Φ?

Boolean Satisfiability

Problem:

Given a propositional formula Φ, is there a truth assignment that satisfies Φ?

Example

$$\Phi = x_1 \wedge (x_3 \vee x_4) \wedge \neg x_4 \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_3 \vee x_4)$$

Boolean Satisfiability

Problem:

Given a propositional formula Φ, is there a truth assignment that satisfies Φ?

Example

$$\Phi = x_1 \wedge (x_3 \vee x_4) \wedge \neg x_4 \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_3 \vee x_4)$$

An Equivalent Formulation

Is there a way to assign values from $\{0, 1\}$ to the variables x_1, x_2, x_3, x_4 so that:

$$\begin{aligned} x_1 &\in \{1\} \qquad x_4 \in \{0\} \\ (x_3, x_4) &\in \{(0,1), (1,0), (1,1)\} \\ (x_1, x_2) &\in \{(0,0), (0,1), (1,1)\} \\ (x_1, x_3, x_4) &\in \{(0,0,0), (0,0,1), (0,1,0), (0,1,1) \\ &\quad (1,0,0), (1,0,1), (1,1,1)\} \end{aligned}$$

The Constraint Satisfaction Problem

Instance

A triple $P = (V, A, \mathcal{C})$ with

- V a nonempty, finite set of variables,
- A a nonempty, finite domain,
- $\mathcal{C}$ a set of constraints $\{C_1, \ldots, C_q\}$ where each C_i is a pair $(\vec{s}_i, R_i)$ with
  - $\vec{s}_i$ a tuple of variables of length m_i, called the scope of C_i, and
  - R_i a subset of A^{m_i}, called the constraint relation of C_i.

The Constraint Satisfaction Problem

Instance

A triple $P = (V, A, \mathcal{C})$ with

- V a nonempty, finite set of variables,
- A a nonempty, finite domain,
- $\mathcal{C}$ a set of constraints $\{C_1, \ldots, C_q\}$ where each C_i is a pair $(\vec{s}_i, R_i)$ with
  - $\vec{s}_i$ a tuple of variables of length m_i, called the scope of C_i, and
  - R_i a subset of A^{m_i}, called the constraint relation of C_i.

Question

Is there a solution to P, i.e., is there a function $f : V \to A$ such that for each $i \leq q$, the m_i-tuple $f(\vec{s}_i) \in R_i$?

NP problems

NP vs P

NP problems

NP vs P

- One feature of a CSP is that it is easy to determine if a given assignment is actually a solution or not.

NP problems

NP vs P

- One feature of a CSP is that it is easy to determine if a given assignment is actually a solution or not.
- Problems of this sort can be solved by non-deterministic polynomial time algorithms and are said to belong to the class NP.

NP problems

NP vs P

- One feature of a CSP is that it is easy to determine if a given assignment is actually a solution or not.
- Problems of this sort can be solved by non-deterministic polynomial time algorithms and are said to belong to the class NP.
- In contrast, problems that can be solved quickly, say in time bounded by a polynomial function of the problem size, are said to belong to the class P.

NP problems

NP vs P

- One feature of a CSP is that it is easy to determine if a given assignment is actually a solution or not.
- Problems of this sort can be solved by non-deterministic polynomial time algorithms and are said to belong to the class NP.
- In contrast, problems that can be solved quickly, say in time bounded by a polynomial function of the problem size, are said to belong to the class P.
- The class of CSPs constitutes a maximally hard problem class within NP.

NP problems

NP vs P

- One feature of a CSP is that it is easy to determine if a given assignment is actually a solution or not.
- Problems of this sort can be solved by non-deterministic polynomial time algorithms and are said to belong to the class NP.
- In contrast, problems that can be solved quickly, say in time bounded by a polynomial function of the problem size, are said to belong to the class P.
- The class of CSPs constitutes a maximally hard problem class within NP.

Problem

Identify natural subclasses of the CSP for which there are efficient algorithms for solving them.

An approach via algebra

Remarks

An approach via algebra

Remarks

- *If the constraints that appear in a given set of instances have a nice underlying algebraic structure, then this may imply the existence of a fast algorithm for solving such instances.*

Remarks

- *If the constraints that appear in a given set of instances have a nice underlying algebraic structure, then this may imply the existence of a fast algorithm for solving such instances.*
- *For example, by using vector space operations, one may quickly solve a system of linear equations over some finite field.*

An approach via algebra

Remarks

- *If the constraints that appear in a given set of instances have a nice underlying algebraic structure, then this may imply the existence of a fast algorithm for solving such instances.*
- *For example, by using vector space operations, one may quickly solve a system of linear equations over some finite field.*

Problem

Identify algebraic properties of sets of constraint relations that can be used to construct good algorithms for solving CSPS over those relations.

Some Constraint Relations

Fact

There is a fast algorithm for solving instances of the CSP whose constraint relations come from the following set:

$$\{(0,5,2),(0,8,2),(0,9,5),(1,8,2),(1,8,5),(4,7,6)\}$$

$$\{(0,2,6),(0,3,2),(0,8,1),(0,9,0),(1,2,2),(1,3,1),(1,3,2),(1,3,5),(4,2,4)\}$$

$$\{(0,0,1),(0,1,3),(0,1,8),(1,1,3),(1,2,1),(1,2,3),(1,5,3),(1,6,1),(4,2,3),(4,4,3),(4,5,3)\}$$

$$\{(5,2,5),(5,2,6),(7,2,0),(7,2,1),(7,2,4),(8,2,2),(9,5,2),(9,6,2)\}$$

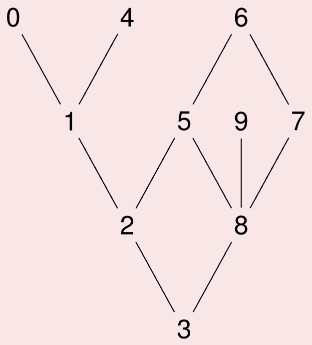
$$\{(5,5,9,1),(7,6,8,8),(8,2,3,3),(8,5,3,2),(8,5,3,3),(8,5,8,3),(9,5,2,2)\}$$

Some Constraint Relations, continued

Hidden Structure

The constraint relations on the previous slide are compatible with the following order and associated binary operation. It follows that the constraint language is tractable.

An Ordering of the Domain

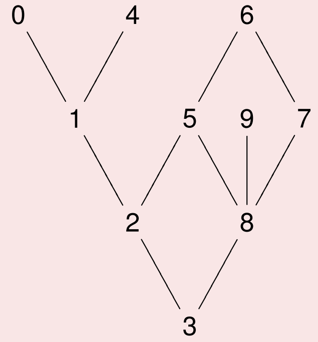


Some Constraint Relations, continued

Hidden Structure

The constraint relations on the previous slide are compatible with the following order and associated binary operation. It follows that the constraint language is tractable.

An Ordering of the Domain



A Compatible Operation

- For $x, y \in \{0, 1, 2, \ldots, 9\}$, define $x \wedge y = \text{g.l.b.}(x, y)$.
- Then for all x, y, z,
  - $x \wedge x = x$,
  - $x \wedge y = y \wedge x$,
  - $x \wedge (y \wedge z) = (x \wedge y) \wedge z$.