



# Representating constraint satisfaction

András Z. Salamon

Computing Laboratory, University of Oxford  
(supervisor: Peter Jeavons)

22 November 2007

# CSP



CSP = Communicating Sequential Processes

# CSP



~~CSP = Communicating Sequential Processes~~

CSP = constraint satisfaction problem

# Cakes talk



Please eat the cakes!



<http://www.pimpthatsnack.com/>

# CSP examples



- ▶ allocating frequencies to mobile phone cells
- ▶ checking if a logical formula is satisfiable
- ▶ laying out components on circuit board
- ▶ fitting a protein structure to measurements
- ▶ finding DNA sequence from set of contigs
- ▶ drawing up a timetable
- ▶ solving system of linear equations

# Constraint satisfaction

What is it?



assign values to variables to satisfy constraints

# Constraint satisfaction

What is it?



assign **values** to variables to satisfy constraints

# Constraint satisfaction

What is it?



assign values to **variables** to satisfy constraints

# Constraint satisfaction

What is it?



assign values to variables to **satisfy constraints**

# What is a constraint?



each constraint has two parts:

- ▶ scope: ordered list of variables
- ▶ relation: allowed combinations of values
- ▶ relation of arity  $r$  over a domain  $D$ :  $R \subseteq D^r$

# Nonogram



		2				
		1	1	3	3	3
1						
1	1					
3						
3						
4						

# Nonogram

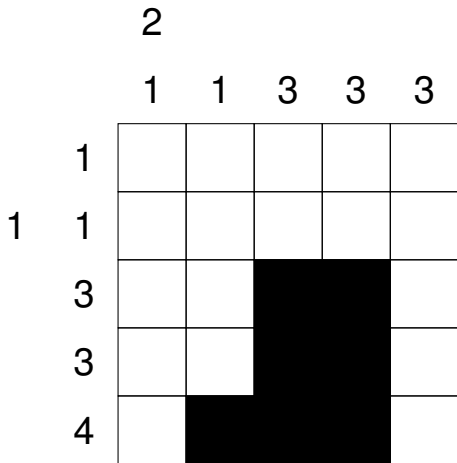


		2				
		1	1	3	3	3
1						
1	1					
3						
3						
4						

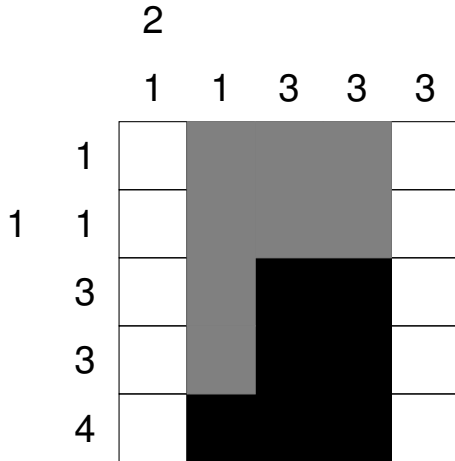
← easy



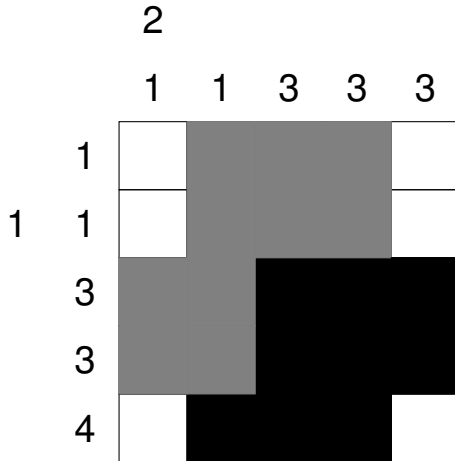
# Nonogram



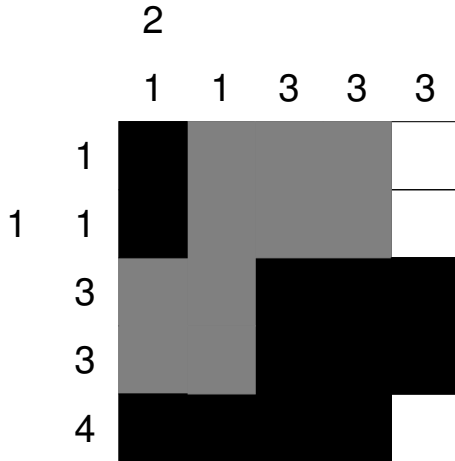
# Nonogram



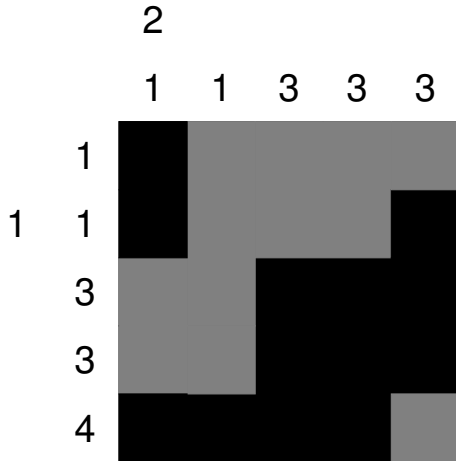
# Nonogram



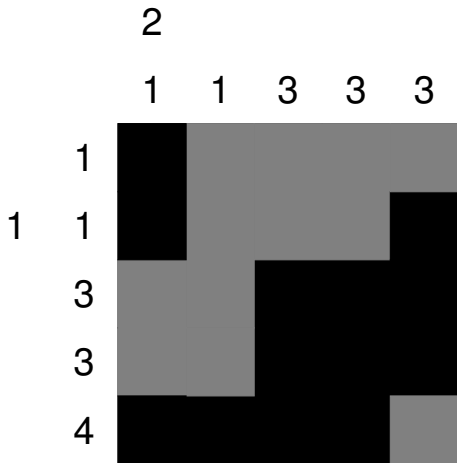
# Nonogram



# Nonogram



# Nonogram (CSP)



# Variable-value representation



## Constraint satisfaction problem **instance**

- ▶ variables: set  $V$  of size  $|V| = s$
- ▶ domain: set  $D$  of size  $|D| = t$
- ▶ constraints: set  $C$
- ▶ solution: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

# Variable-value representation



## Constraint satisfaction problem instance

- ▶ **variables**: set  $V$  of size  $|V| = s$
- ▶ domain: set  $D$  of size  $|D| = t$
- ▶ constraints: set  $C$
- ▶ solution: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

# Variable-value representation



## Constraint satisfaction problem instance

- ▶ variables: set  $V$  of size  $|V| = s$
- ▶ **domain**: set  $D$  of size  $|D| = t$
- ▶ constraints: set  $C$
- ▶ solution: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

# Variable-value representation



## Constraint satisfaction problem instance

- ▶ variables: set  $V$  of size  $|V| = s$
- ▶ domain: set  $D$  of size  $|D| = t$
- ▶ **constraints**: set  $C$
- ▶ solution: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

# Variable-value representation



## Constraint satisfaction problem instance

- ▶ variables: set  $V$  of size  $|V| = s$
- ▶ domain: set  $D$  of size  $|D| = t$
- ▶ constraints: set  $C$
- ▶ **solution**: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

# Variable-value representation



## Constraint satisfaction problem instance

- ▶ variables: set  $V$  of size  $|V| = s$
- ▶ domain: set  $D$  of size  $|D| = t$
- ▶ constraints: set  $C$
- ▶ solution: function  $f: V \rightarrow D$  such that if  $((v_1, v_2, \dots, v_r), R) \in C$  then  $(f(v_1), f(v_2), \dots, f(v_r)) \in R$

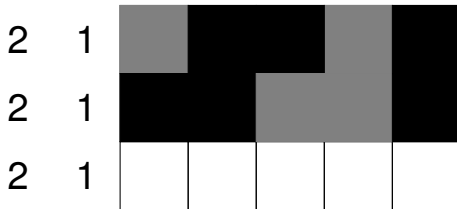
**instance**  $(V, D, C)$

# Nonogram as CSP



2	1				
2	1				
2	1				

# Nonogram as CSP



# Nonogram as CSP



# Nonogram as CSP



$v_{00}$   $v_{01}$   $v_{02}$   $v_{03}$   $v_{04}$  **0** **1** **1** **0** **1**

# Nonogram as CSP



					0	1	1	0	1
$v_{00}$	$v_{01}$	$v_{02}$	$v_{03}$	$v_{04}$	1	1	0	0	1

# Nonogram as CSP



					0	1	1	0	1
					1	1	0	0	1
$v_{00}$	$v_{01}$	$v_{02}$	$v_{03}$	$v_{04}$	1	1	0	1	0

# Nonogram as CSP



					0	1	1	0	1
					1	1	0	0	1
$v_{00}$	$v_{01}$	$v_{02}$	$v_{03}$	$v_{04}$	1	1	0	1	0
scope					relation				

# All-different constraint



all-different constraint of arity  $r$  over domain  $D$

- ▶ scope: variables  $(v_1, v_2, \dots, v_r)$
- ▶ relation: all tuples  $(d_1, d_2, \dots, d_r)$  over  $D$  such that  $i \neq j \Rightarrow d_i \neq d_j$

all-different relation is denoted  $\neq_D^r$  or just  $\neq$

# Sudoku



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

# Sudoku



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

← hard

# Sudoku as CSP



Given a  $9 \times 9$  grid, divided into  $3 \times 3$  regions, containing numbers in some of the blocks, fill in the remaining blocks with digits 1 to 9 so that

- ▶ each row contains each digit
- ▶ each column contains each digit
- ▶ each region contains each digit

use all-different constraint

# Sudoku as CSP



Given a  $9 \times 9$  grid, divided into  $3 \times 3$  regions, containing numbers in some of the blocks, fill in the remaining blocks with digits 1 to 9 so that

- ▶ each row contains each digit
- ▶ each column contains each digit
- ▶ each region contains each digit

use all-different constraint

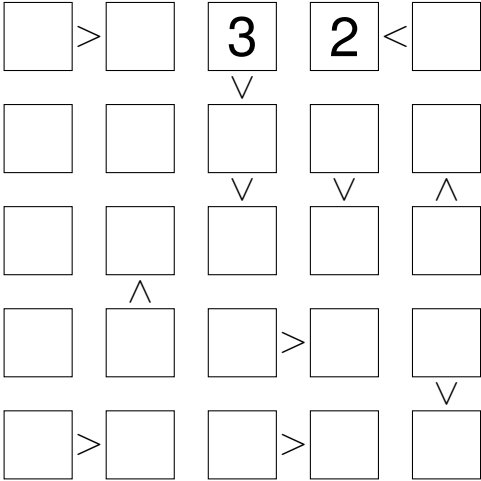
# Constraint relations



relations over domain  $D$  of arity  $r$

- ▶ all-different ( $\neq$ )
- ▶ anything-goes ( $*$ )
- ▶ all-equals ( $\equiv$ )
- ▶ not-all-equals ( $\neq$ )
- ▶ linear order ( $<$ ), arity 2

# Futoshiki





# Relational structure



tuple  $(D, (R_i)_{i \in I})$

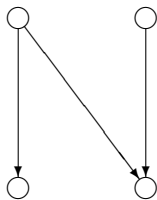
- ▶ domain  $D$
- ▶ tuple  $(R_i)_{i \in I}$  of relations
- ▶ ordered index set  $I$
- ▶ relations  $R_i$  over  $D$

# Graph



loop-free directed graph:  $(V, (E_i)_{i \in \{1\}})$

- ▶ set of vertices  $V$
- ▶ directed edges  $E_1$  (arity 2)
- ▶ no loops so irreflexive:  $\forall x \in V. (x, x) \notin E$
- ▶  $E$  is **symmetric**: if  $(u, v) \in E$  then  $(v, u) \in E$

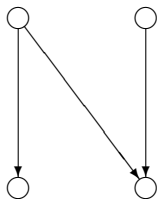


# Graph



loop-free directed graph:  $(V, E)$

- ▶ set of vertices  $V$
- ▶ directed edges  $E$  (arity 2)
- ▶ no loops so irreflexive:  $\forall x \in V. (x, x) \notin E$
- ▶  $E$  is **symmetric**: if  $(u, v) \in E$  then  $(v, u) \in E$

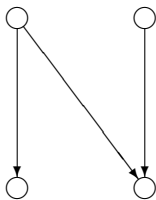


# Graph



loop-free directed graph:  $(V, E)$

- ▶ set of vertices  $V$
- ▶ directed edges  $E$  (arity 2)
- ▶ no loops so irreflexive:  $\forall x \in V. (x, x) \notin E$
- ▶  $E$  otherwise arbitrary

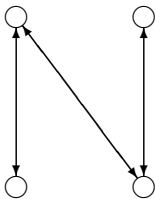


# Graph



loop-free **un**directed graph:  $(V, E)$

- ▶ set of vertices  $V$
- ▶ directed edges  $E$  (arity 2)
- ▶ no loops so irreflexive:  $\forall x \in V. (x, x) \notin E$
- ▶  $E$  is **symmetric**: if  $(u, v) \in E$  then  $(v, u) \in E$

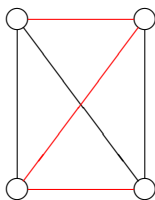


# Graph as relational structure



complemented representation:  $(V, (E_i)_{i \in \{1,2\}})$

- ▶  $E_1$  symmetric irreflexive relation over  $V$  (arity 2)
- ▶  $E_2$  complement of  $E$
- ▶ partition tuples of  $\neq_V$  into  $E$  and  $\overline{E}$
- ▶ colour  $E$  edges black,  $\overline{E}$  red

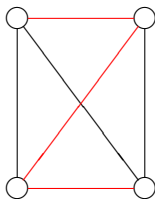


# Graph as relational structure



complemented representation:  $(V, (E_i)_{i \in \{1,2\}})$

- ▶  $E_1$  symmetric irreflexive relation over  $V$  (arity 2)
- ▶  $E_2$  complement of  $E$
- ▶ partition tuples of  $\neq_V$  into  $E$  and  $\overline{E}$
- ▶ colour  $E$  edges black,  $\overline{E}$  red

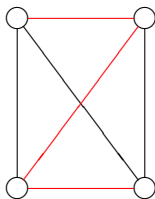


# Graph as relational structure



complemented representation:  $(V, (E, \overline{E}))$

- ▶  $E$  symmetric irreflexive relation over  $V$  (arity 2)
- ▶  $\overline{E}$  **complement** of  $E$
- ▶ partition tuples of  $\neq_V$  into  $E$  and  $\overline{E}$
- ▶ colour  $E$  edges black,  $\overline{E}$  red



# Graph as relational structure



complemented representation:  $(V, (E, \bar{E}))$

- ▶  $E$  symmetric irreflexive relation over  $V$  (arity 2)
- ▶  $\bar{E}$  complement of  $E$
- ▶ partition tuples of  $\neq_V$  into  $E$  and  $\bar{E}$
- ▶ colour  $E$  edges black,  $\bar{E}$  red
- ▶ generalise to **2-structures**: allow more colours

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ **source**:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ **source:**  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ **target:**  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  **similar**: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ **solution**: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow (f(v_1), \dots, f(v_{\rho(R)})) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$(v_1, \dots, v_{\rho(R)}) \in Q_i \Rightarrow f(v_1, \dots, v_{\rho(R)}) \in R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: function  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$f(Q_i) \subseteq R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: **function**  $f: V \rightarrow D$  such that  
 $\forall i \in I$

$$f(Q_i) \subseteq R_i$$

# Homomorphism representation



## Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: **homomorphism**  $f: S \rightarrow T$

# Homomorphism representation



Constraint satisfaction problem instance

- ▶ source:  $S = (V, (Q_i)_{i \in I})$ ,  $|V| = s$
- ▶ target:  $T = (D, (R_i)_{i \in I})$ ,  $|D| = t$
- ▶  $S$  and  $T$  similar: arities of  $Q_i$  and  $R_i$  match
- ▶ solution: homomorphism  $f: S \rightarrow T$

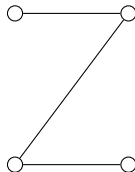
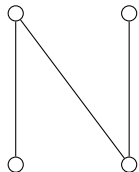
instance  $S \rightarrow T?$

# Graph isomorphism



Are graphs  $G = (V, Q)$ ,  $H = (D, R)$  isomorphic?

- ▶  $S = (V, (Q, \overline{Q}))$ ,  $T = (D, (R, \overline{R}))$
- ▶  $|V| = |D|$



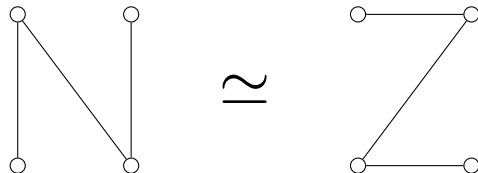
# Graph isomorphism



Are graphs  $G = (V, Q)$ ,  $H = (D, R)$  isomorphic?

- ▶  $S = (V, (Q, \overline{Q}))$ ,  $T = (D, (R, \overline{R}))$
- ▶  $|V| = |D|$

$G$  is isomorphic to  $H$  if and only if  $S \rightarrow T$



# Graph isomorphism

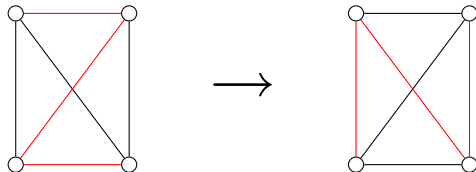


Are graphs  $G = (V, Q)$ ,  $H = (D, R)$  isomorphic?

▶  $S = (V, (Q, \overline{Q}))$ ,  $T = (D, (R, \overline{R}))$

▶  $|V| = |D|$

$G$  is isomorphic to  $H$  if and only if  $S \rightarrow T$



# Motivation



- ▶ **NP = SO $\exists$**  (Fagin 1973)
- ▶ MMSNP (Feder & Vardi 1993)
- ▶ MMSNP is large subset of NP:  
SATISFIABILITY, CLIQUE, GRAPH  
ISOMORPHISM, GRAPH COLOURING,  
LINEAR PROGRAMMING
- ▶ MMSNP  $\approx$  CSP (Kun 2006)
- ▶  $S \rightarrow *$
- ▶  $* \rightarrow T$

# Motivation



- ▶ NP = SO $\exists$  (Fagin 1973)
- ▶ MMSNP (Feder & Vardi 1993)
- ▶ MMSNP is large subset of NP:  
SATISFIABILITY, CLIQUE, GRAPH  
ISOMORPHISM, GRAPH COLOURING,  
LINEAR PROGRAMMING
- ▶ MMSNP  $\approx$  CSP (Kun 2006)
- ▶  $S \rightarrow *$
- ▶  $* \rightarrow T$

# Motivation



- ▶  $NP = SO\exists$  (Fagin 1973)
- ▶ MMSNP (Feder & Vardi 1993)
- ▶ MMSNP is large subset of NP:  
SATISFIABILITY, CLIQUE, GRAPH  
ISOMORPHISM, GRAPH COLOURING,  
LINEAR PROGRAMMING
- ▶  $MMSNP \approx CSP$  (Kun 2006)
- ▶  $S \rightarrow *$
- ▶  $* \rightarrow T$

# Motivation



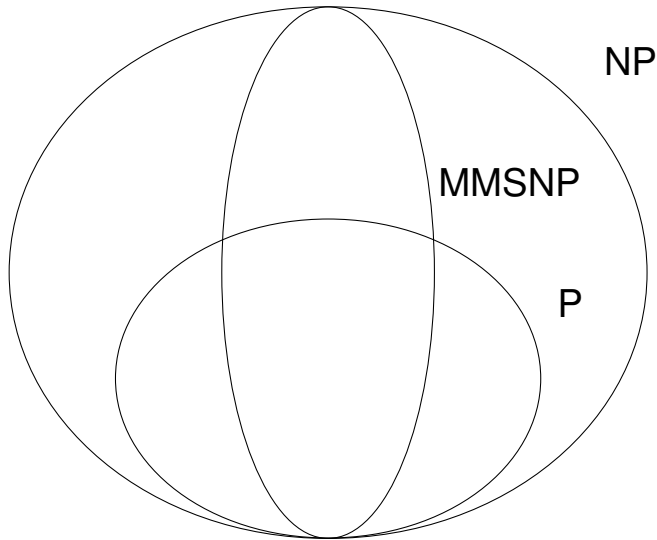
- ▶  $NP = SO\exists$  (Fagin 1973)
- ▶ MMSNP (Feder & Vardi 1993)
- ▶ MMSNP is large subset of NP:  
SATISFIABILITY, CLIQUE, GRAPH  
ISOMORPHISM, GRAPH COLOURING,  
LINEAR PROGRAMMING
- ▶  $MMSNP \approx CSP$  (Kun 2006)
- ▶  $S \rightarrow *$
- ▶  $* \rightarrow T$

# Motivation



- ▶  $NP = SO\exists$  (Fagin 1973)
- ▶ MMSNP (Feder & Vardi 1993)
- ▶ MMSNP is large subset of NP:  
SATISFIABILITY, CLIQUE, GRAPH  
ISOMORPHISM, GRAPH COLOURING,  
LINEAR PROGRAMMING
- ▶  $MMSNP \approx CSP$  (Kun 2006)
- ▶  $S \rightarrow *$
- ▶  $* \rightarrow T$

NP



NP

MMSNP

P

# Intuition



- ▶ graphs are 2-uniform
- ▶  $G = (V, E): E \subseteq V^2$

# Intuition



- ▶ adjacency matrix  $M$  is 2-dimensional
- ▶  $M_{ij} = 1$  iff  $(i, j) \in E$

# Intuition



- ▶  $r$ -uniform hypergraph

# Intuition



- ▶  $r$ -uniform hypergraph
- ▶  $r$ -dimensional adjacency structure

# Intuition



- ▶  $r$ -uniform hypergraph
- ▶  $r$ -dimensional adjacency structure
- ▶  $M_{v_1 v_2 \dots v_r} = 1$  iff  $(v_1, v_2, \dots, v_r) \in R$

# Intuition



- ▶ glitch: ordering within edges
- ▶ graphs: just use either half of matrix
- ▶ linear order on vertices
- ▶  $k$ -uniform: identify  $k!$  orderings
- ▶ “generalised” linear order

# Complete representation (CR)



- ▶ special case of uniform homomorphism representation
- ▶ all complete source structures isomorphic

# Complete representation (CR)



- ▶ special case of uniform homomorphism representation
- ▶ all complete source structures isomorphic

# Complete representation (CR)



- ▶ special case of uniform homomorphism representation
- ▶ use canonical complete source structure

# Complete representation (CR)



- ▶ special case of uniform homomorphism representation
- ▶ use canonical complete source structure
- ▶  $CS(r, s) = (V, (Q_i)_{i \in I})$  where
  - ▶  $V = \{0, \dots, s - 1\}$
  - ▶  $I = \{(v_1, \dots, v_r) \mid 0 \leq v_1 < \dots < v_r \leq s - 1\}$
  - ▶  $I$  ordered lexicographically
  - ▶  $Q_i = \{i\}$

# Complete representation (CR)



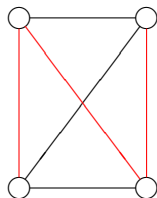
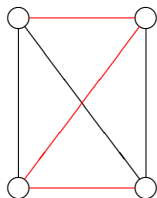
- ▶ special case of uniform homomorphism representation
- ▶ use canonical complete source structure
- ▶  $CS(r, s) = (V, (Q_i)_{i \in I})$  where
  - ▶  $V = \{0, \dots, s - 1\}$
  - ▶  $I = \{(v_1, \dots, v_r) \mid 0 \leq v_1 < \dots < v_r \leq s - 1\}$
  - ▶  $I$  ordered lexicographically
  - ▶  $Q_i = \{i\}$

# GRAPH ISOMORPHISM



homomorphism representation:

$$(V, (Q, \overline{Q})) \rightarrow (D, (R, \overline{R}))?$$



# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

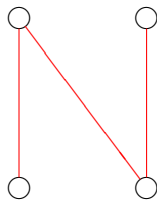
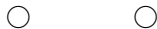
$$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$$

# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$

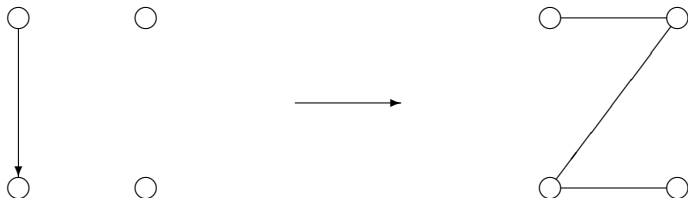


# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$

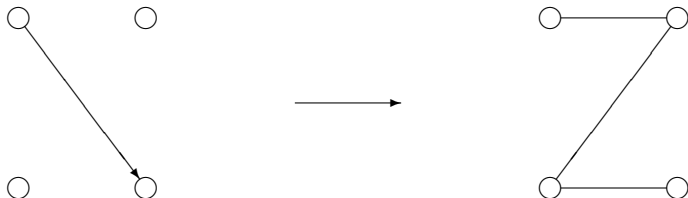


# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$

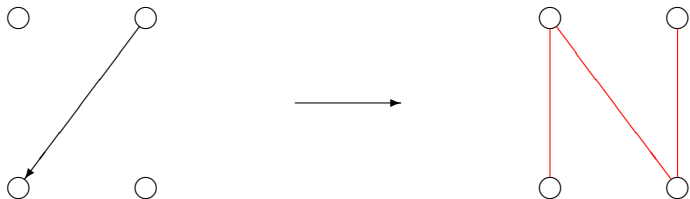


# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$

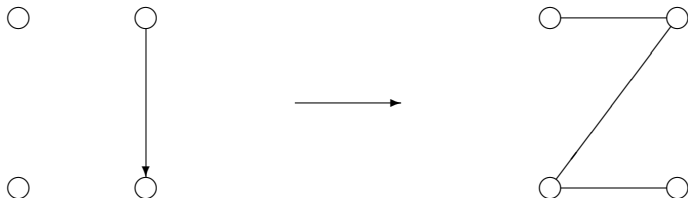


# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$

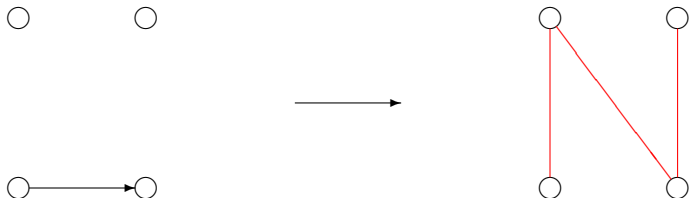


# GRAPH ISOMORPHISM



CR of GRAPH ISOMORPHISM:

$CS(2, s) \rightarrow (D, (R_i)_{i \in I})?$



# What is going on?



- ▶ source embedded in target
- ▶ pattern of relations
- ▶  $r$ -dimensional “adjacency matrix”

# Why do this?



- ▶ nice data structures when data is ordered (arrays, linked lists, mixed-radix systems)
- ▶ CR is often same size: no need to represent relations explicitly
- ▶ tune parameters  $r$ ,  $s$ ,  $t$  to obtain different problems

# Languages



- ▶  $\Gamma(P)$ : set of all relations in target structures
- ▶ expressibility:  $\langle \Gamma \rangle$  characterises tractability (Jeavons 1996)
- ▶ local language: tuples of relations in each instance

# Local languages



$CS(2, s) \rightarrow T$	$\{R_i\}$	$r, s, t$
<hr/>		
$s$ -CLIQUE	$E$	
$t$ -COLOURING	$*, \neq$	
GRAPH ISOMORPHISM	$E, \overline{E}$	$s = t$
SUBGRAPH ISOMORPHISM	$E, \overline{E}$	
$CS(r, s) \rightarrow T$		
<hr/>		
COMPLETE $(r, \_)$ -SUBSTRUCTURE	$R$	
HYPERGRAPH $t$ -COLOURING	$*, \neq$	

# Complexity



- ▶ instance size  $n$  usually  $\geq qt^r + \binom{s}{r} \log q$  bits
- ▶ exhaustive enumeration  $\leq t^s \binom{s}{r} r \log t$  steps
- ▶ upper bounds on runtime as  $r$  grows:
  - ▶ *very slowly*: exponential
  - ▶ *as quickly as  $s$* : polynomial (new tractable problems)
- ▶ intermediate problems likely not NP-complete, but not known to be tractable

# Complexity



- ▶ instance size  $n$  usually  $\geq qt^r + \binom{s}{r} \log q$  bits
- ▶ exhaustive enumeration  $\leq t^s \binom{s}{r} r \log t$  steps
- ▶ upper bounds on runtime as  $r$  grows:
  - ▶ *very slowly*: exponential
  - ▶ *as quickly as  $s$* : polynomial (new tractable problems)
- ▶ intermediate problems likely not NP-complete, but not known to be tractable

# Complexity



- ▶ instance size  $n$  usually  $\geq qt^r + \binom{s}{r} \log q$  bits
- ▶ exhaustive enumeration  $\leq t^s \binom{s}{r} r \log t$  steps
- ▶ upper bounds on runtime as  $r$  grows:
  - ▶ *very slowly*: exponential
  - ▶ *as quickly as  $s$* : polynomial (new tractable problems)
- ▶ intermediate problems likely not NP-complete, but not known to be tractable

# Complexity



- ▶ instance size  $n$  usually  $\geq qt^r + \binom{s}{r} \log q$  bits
- ▶ exhaustive enumeration  $\leq t^s \binom{s}{r} r \log t$  steps
- ▶ upper bounds on runtime as  $r$  grows:
  - ▶ *very slowly*: exponential
  - ▶ *as quickly as  $s$* : polynomial (new tractable problems)
- ▶ **intermediate problems** likely not NP-complete, but not known to be tractable

# Microstructure



- ▶ **direct product:**  $((u, v), (x, y)) \in Q \times R$  iff  $(u, x) \in Q$  and  $(v, y) \in R$
- ▶  $(V, (Q_i)_{i \in I}) + R = (V, (Q_i)_{i \in I}, R)$
- ▶ **microstructure complement:**  
 $MSC(S, T) = (S + =) \times \overline{T + =}$

# Application



- ▶ observation:  $S \rightarrow T$  iff  $MSC(S, T)$  contains independent set of size  $s = |V(S)|$
- ▶ INDEPENDENT SET is NP-complete
- ▶ <http://wwwteo.informatik.uni-rostock.de/isgci/>
- ▶ 1000 classes, many for which INDEPENDENT SET tractable
- ▶ perfect, claw-free, greedy, broken-triangle: new tractable classes of CSPs

# What next?



- ▶ relate CR to other intermediate approaches
- ▶ new tractable hybrid classes
- ▶ number problems

# How not to buy cakes



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

# How not to buy cakes



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes?

# How not to buy cakes



5.00

5.00

Can I bring exactly £10 of cakes?

# How not to buy cakes



1.49 1.49 1.09 1.25 0.99 0.85

0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes? **Yes!**

# How not to buy cakes



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes?

- ▶ SUBSET SUM is NP-complete

# How not to buy cakes



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes?

- ▶ SUBSET SUM is NP-complete
- ▶ as CSP: linear time search

# How not to buy cakes (not CSP)



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes?

- ▶ SUBSET SUM is NP-complete
- ▶ as CSP: linear time search
- ▶ ...of exponential size structure

# How not to buy cakes (CSP)



1.49 1.49 1.09 1.25 0.99 0.85  
0.85 1.09 1.09 1.69 1.55

Can I bring exactly £10 of cakes?

- ▶ SUBSET SUM is NP-complete
- ▶ as CSP: linear time search
- ▶ ...of exponential size structure

implicit representation (Green & Jefferson)

# Summary



- ▶ two new representations
- ▶ complete: intermediate complexity  $(r, s, t)$
- ▶ microstructure: new tractable classes