Learning Constraint Networks over Unknown Constraint Languages

Christian Bessiere | Clément Carbonnel | Areski Himeur

University of Montpellier, CNRS, LIRMM, Montpellier, France

{bessiere, clement.carbonnel, areski.himeur}@lirmm.fr

Background

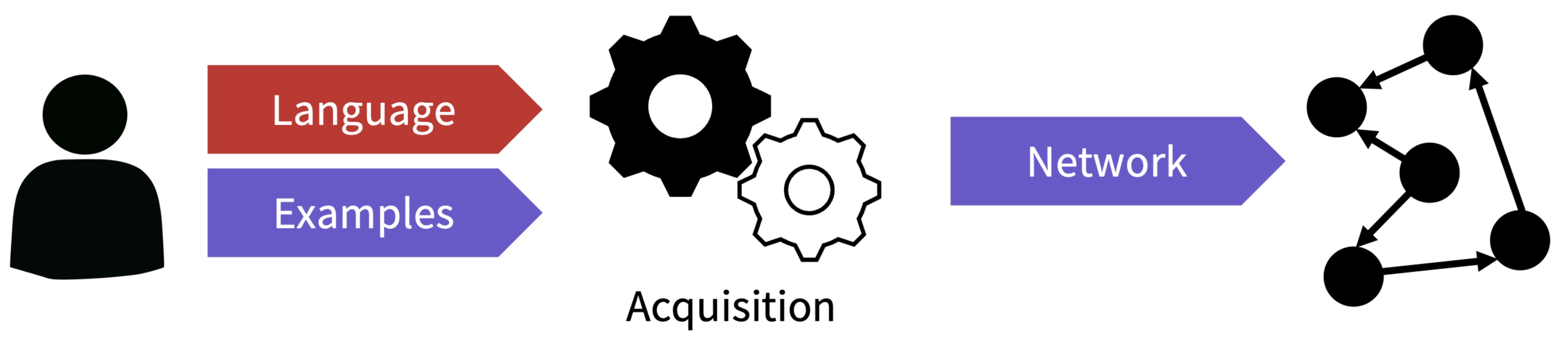
A **constraint network** consists in **variables** over a finite **domain** and **constraints**, i.e. relations between variables that must be satisfied in any **solution**. Relations in a network are its **language**.

Example Network representing Sudoku

Variables: 81 cells of domain $[1..9]$
Constraints: $x \neq y$ if on the same row, column or 3×3 block
Language: $\{\neq\}$

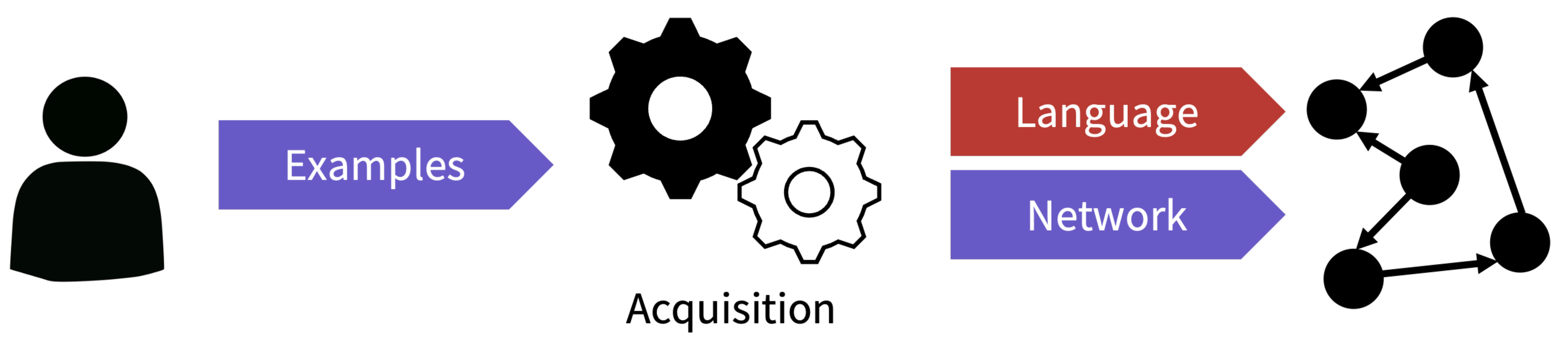
Constraint Acquisition

Learns a network automatically from examples of solutions and non-solutions. Current approaches require the language of the output network as input.



Language Acquisition

Our method **constructs a suitable constraint language as part of the learning process**.



Numerous languages can be used. Some are clearly unsatisfactory from a practical point of view (e.g. overfitting).

—

Intuition: the best language is the "simplest".
Approximation: minimizing the arity and number of relations.

Sub-problem: LANGUAGE-FREE ACQUISITION

Instance: Set of examples labelled as solutions and non-solutions, two integers k and r.
Question: Is there a network **over a language of size** $\leq k$ **and arity** $\leq r$ that correctly classifies the examples?

▶ NP-complete even for $k = r = 1$.

The Method

Goal **Compute a constraint network with minimum** (k, r) **that correctly classifies the examples.**

- **Strategy:** minimize $k + r^2$
- **Tie-breaking:** lower arity, more constraints, tighter constraints

General Method

Construct and solve a model for each (k, r) **following a bottom-up minimization:**

- Convert to an instance WEIGHTED PARTIAL MAX-SAT
- Compute an optimal network or prove that none exists
- Output the first constraint network found

Experimental results

Jigsaw Sudoku Sudoku with irregular shapes. $(\boldsymbol{k}, \boldsymbol{r}) = (1, 2)$.

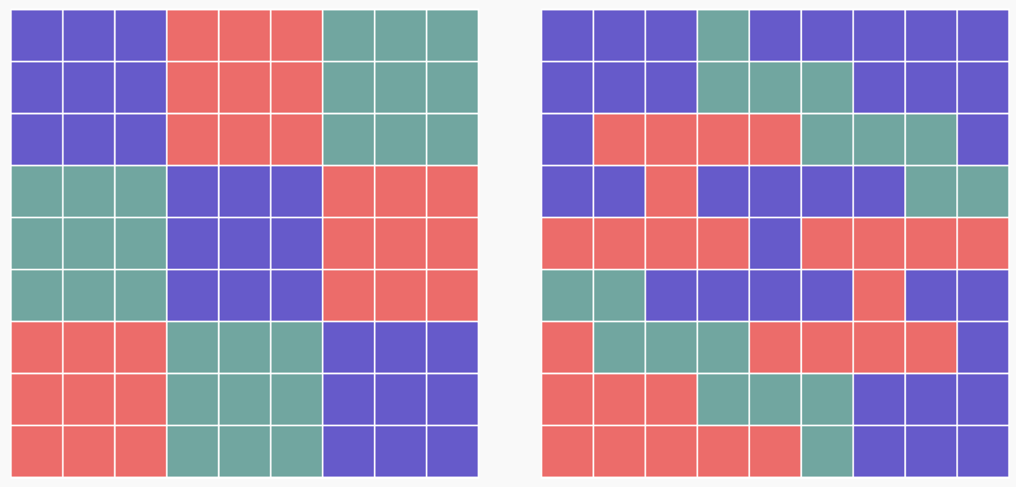


Figure Jigsaw sudoku (right) has irregular block shapes, in contrast to classical sudoku (left)

Schur's Lemma **Variables:** $x_1, \ldots, x_9$. **Constraints:** $NotAllEqual(x_i, x_j, x_k)$ if $i + j = k$. $(\boldsymbol{k}, \boldsymbol{r}) = (1, 3)$.

Subgraph Isomorphism Map C_5 in G having 20 vertices and 100 edges. **Variables:** $x_1, \ldots, x_5$. **Constraints:** for all (i, j), $x_i \neq x_j$ and $(i, j) \in C_5 \Rightarrow (x_i, x_j) \in G$. $(\boldsymbol{k}, \boldsymbol{r}) = (2, 2)$.

Golomb Ruler **Variables**: $x_1, \ldots, x_{10}$ on $[0..60]$. **Constraints:** $|x_i - x_j| \neq |x_k - x_l|$ for all i, j, k, l. $(\boldsymbol{k}, \boldsymbol{r}) = (1, 4)$.

8-Queens **Variables:** $x_1, \ldots x_8$. **Constraints:** $x_i \neq x_j$ and $|x_i - x_j| \neq |i - j|$ for all i, j. $(\boldsymbol{k}, \boldsymbol{r}) = (9, 2)$.

Problem	|E|	(k, r)	Lang	Eq	Target	Acc	Time
Sudoku	100	$(1, 2)$	✓	✗	✗	84%	$129s$
	200	$(1, 2)$	✓	✓	✓	100%	$35s$
Jigsaw Sudoku	200 to 1400	$(1, 2)$	✓	✓	✓	100%	$\simeq 30s$
Schur's Lemma	50	$(1, 3)$	✓	✗	✗	87%	$23s$
	800	$(1, 3)$	✓	✓	✓	100%	$2s$
Subgraph Isomorphism	400	$(2, 2)$	✗	✗	✗	98%	$1s$
	800	$(2, 2)$	✗	✓	✗	100%	$2s$
Golomb Ruler	1600	-	-	-	-	-	$> 12h$
	3200	$(1, 3)$	✗	✓	✗	100%	$7h$
8-Queens	184	$(3, 2)$	✗	✗	✗	99%	$17s$

Table $|E|$: number of examples; Lang: target language found; Eq: equivalent network found (i.e. same solutions); Target: target network found; Acc: accuracy measured on 2 000 unseen examples. Results for Jigsaw depend on shapes.

Future work **More sophisticated notions of simplicity** and **detecting topological information**.

TAILOR ANITI ISDM

Work supported by EU Horizon 2020 TAILOR (GA N° 952215), ANITI (GA N° ANR-19-PI3A-0004) and ANR AXIAUM (GA N° ANR-20-THIA-0005-01).
Experiments performed with the MESO@LR-Platform at University of Montpellier.