ECAI 2025

Learning Compact Representations of Constraint Networks

Christian Bessiere | Clément Carbonnel | **Areski Himeur***

University of Montpellier, CNRS, LIRMM, Montpellier, France
{bessiere, clement.carbonnel, areski.himeur}@lirmm.fr

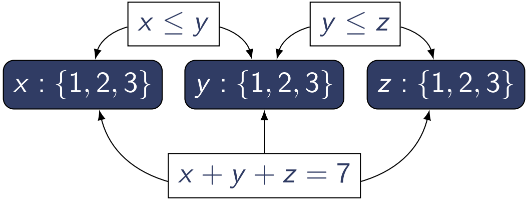
* *New affiliation: INSA Lyon, Inria, CITI lab. 69621 Villeurbanne, France*
areski.himeur@insa-lyon.fr

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)

Background

A **constraint network**:

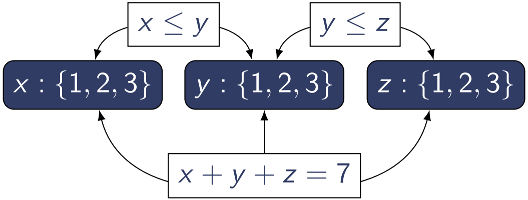
- a set of **variables** over a finite **domain**
- a set of **constraints**, i.e. relations between variables that must be satisfied in any **solution**



Background

A **constraint network**:

- a set of **variables** over a finite **domain**
- a set of **constraints**, i.e. relations between variables that must be satisfied in any **solution**

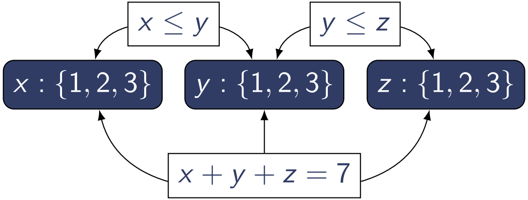


Constraint programming is an **expressive**, **flexible**, **efficient** paradigm for solving problems.

Background

A constraint network:

- a set of **variables** over a finite **domain**
- a set of **constraints**, i.e. relations between variables that must be satisfied in any **solution**



Constraint programming is an **expressive**, **flexible**, **efficient** paradigm for solving problems.

Challenge: The modeling process is a significant bottleneck

The Solution: Constraint Acquisition

Passive Constraint Acquisition

Instance: Set of examples (assignments labelled as solutions and non-solutions).

Goal: Find a constraint network consistent with the examples.

The Solution: Constraint Acquisition

Passive Constraint Acquisition

Instance: Set of examples (assignments labelled as solutions and non-solutions).

Goal: Find a constraint network consistent with the examples.

CONACQ.1 (Bessiere et al., 2006, 2017), MODELSEEKER (Beldiceanu and Simonis, 2012), BAYESACQ (Prestwich et al., 2021), COUNT-CP (Kumar et al., 2022), LANGUAGE-FREE ACQ (Bessiere et al., 2023), etc.

Which network?

Given a set of examples, there are often many constraint networks that are consistent with these examples.

Which network?

Given a set of examples, there are often many constraint networks that are consistent with these examples.

Which network will generalize best to new, unseen data?

Learning Compact Representations

Our claim

Learning compact representations of constraint networks is the key to better generalization.

Learning Compact Representations

Our claim

Learning compact representations of constraint networks is the key to better generalization.

Our Contribution

Learning Compact Representations

Our claim

Learning compact representations of constraint networks is the key to better generalization.

Our Contribution

1. A novel, compact representation for structured networks, which we call *template*.

Learning Compact Representations

Our claim

Learning compact representations of constraint networks is the key to better generalization.

Our Contribution

1. A novel, compact representation for structured networks, which we call *template*.
2. A new acquisition framework, TACQ, that learns these templates directly from examples.

Compact Representation

Example with the Sudoku

(a) Attribute 1 (Rows) (b) Attribute 2 (Columns) (c) Attribute 3 (Squares 3×3)

- For each variable (cell), we specify three *attributes* (ϕ_{row}, ϕ_{col}, and ϕ_{square}).
- We produce all the constraints of Sudoku using *rules* depending on these attributes.

What is a Template?

1. **Attributes:** Functions that assign numerical features to variables.

2. **Rules:** Mechanisms for producing many constraints based on attributes. It consists of:
   - A **Relation** for the constraint (e.g., $\neq$),
   - A **Selector** that specifies which attributes to check,
   - A **Trigger** function that determines if the constraint should be produced based on the selected attributes.

Template Example: Sudoku

A Concise Representation for Sudoku

Variables:

- 81 variables : $x_{i,j}$

Attributes:

- $\phi_{\text{row}}(x_{i,j}) = i$
- $\phi_{\text{col}}(x_{i,j}) = j$
- $\phi_{\text{square}}(x_{i,j}) = \left\lfloor \frac{i}{3} \right\rfloor \times 3 + \left\lfloor \frac{j}{3} \right\rfloor$

Rules :

1. **Row Rule:**
   Apply '$\neq$' to (x_u, x_v) *if* $\phi_{\text{row}}(x_u) = \phi_{\text{row}}(x_v)$.
2. **Column Rule:**
   Apply '$\neq$' to (x_u, x_v) *if* $\phi_{\text{col}}(x_u) = \phi_{\text{col}}(x_v)$.
3. **Square Rule:**
   Apply '$\neq$' to (x_u, x_v) *if* $\phi_{\text{square}}(x_u) = \phi_{\text{square}}(x_v)$.

Learning Algorithm

The TACQ Learning Framework

A Two-Step Process:

The TACQ Learning Framework

A Two-Step Process:

1. **Learn Initial Network:** Use a baseline method to learn an initial network N that is consistent with the training examples.

The TACQ Learning Framework

A Two-Step Process:

1. **Learn Initial Network:** Use a baseline method to learn an initial network N that is consistent with the training examples.
2. **Refine into a Template:** Learn a compact template T whose interpretation is a large subset of N.

The Template Learning Algorithm

Algorithm Sketch

Input: A training set E and an initial network N consistent with E.

Output: A template T consistent with E.

The Template Learning Algorithm

Algorithm Sketch

Input: A training set E and an initial network N consistent with E.

Output: A template T consistent with E.

---

1. Start with an empty template T.
2. **While** the template T is not consistent with the training set E:
   1. **Guess new attribute.**
   2. **Greedily add rules that produce many new constraints of N.**

The Template Learning Algorithm

Algorithm Sketch

Input: A training set E and an initial network N consistent with E.

Output: A template T consistent with E.

---

1. Start with an empty template T.
2. **While** the template T is not consistent with the training set E:
   1. **Guess new attribute.**
   2. **Greedily add rules that produce many new constraints of N.**

A key contribution is the heuristic which determines the width of new attributes.

Experimental Evaluation

Higher Accuracy / Fewer Examples

Problem	**Examples for 100% accuracy**		**Reduction**
	LFA	LFA+TACQ	
Sudoku	120	**80**	33%
Jigsaw [*3 instances*]	1490	**1130**	24%
Nurse Rostering [*3 instances*]	720	**590**	18%
Exam Timetabling [*3 instances*]	2536	**919**	64%
Schur's Lemma	560	560	0%
Subgraph Isomorphism	640	640	0%
Golomb Ruler (10 variables)	2100	2100	0%

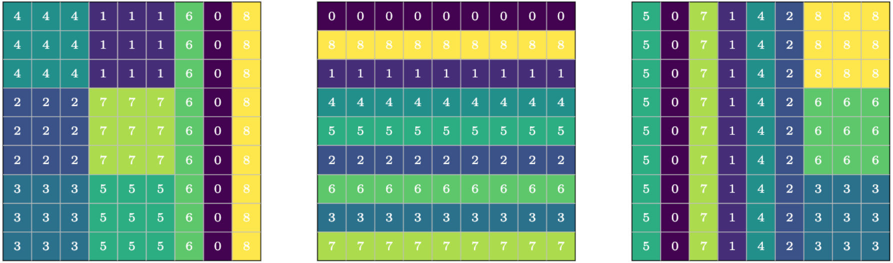
Table: Number of examples to reach 100% accuracy.

Results: Learning Interpretable Attributes

TACQ learns attributes corresponding to meaningful features.

Results: Learning Interpretable Attributes

TAcq learns attributes corresponding to meaningful features.



(a) Attribute 1 (ϕ_1) (b) Attribute 2 (ϕ_2) (c) Attribute 3 (ϕ_3)

Figure: Illustration of the three attributes learned by TAcq for Sudoku.

Conclusion

Key Takeaways

- Template is a representation of constraint networks in a compact and structured form,
- TACQ is a framework that learns Templates from examples,
- TACQ significantly **reduces the number of examples** needed to learn an accurate model for structured problems,
- The learned templates are often **interpretable**.

Future Work: Investigate learning parameterized models.

Thank you for your time and attention.

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)

References

Beldiceanu, N., & Simonis, H. (2012). A model seeker: Extracting global constraint models from positive examples. In M. Milano (Ed.), *Principles and practice of constraint programming - 18th international conference, CP 2012, québec city, qc, canada, october 8-12, 2012. proceedings* (pp. 141–157, Vol. 7514). Springer. https://doi.org/10.1007/978-3-642-33558-7_13 (cit. on pp. 5, 6).

Bessiere, C., Carbonnel, C., & Himeur, A. (2023). Learning constraint networks over unknown constraint languages. *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, 1876–1883. https://doi.org/10.24963/IJCAI.2023/208 (cit. on pp. 5, 6).

Bessiere, C., Coletta, R., Koriche, F., & O'Sullivan, B. (2006). Acquiring constraint networks using a sat-based version space algorithm. *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, 1565–1568. http://www.aaai.org/Library/AAAI/2006/aaai06-251.php

Bessiere, C., Koriche, F., Lazaar, N., & O'Sullivan, B. (2017). Constraint acquisition. *Artif. Intell.*, *244*, 315–342. https://doi.org/10.1016/j.artint.2015.08.001

Kumar, M., Kolb, S., & Guns, T. (2022). Learning constraint programming models from data using generate-and-aggregate. In C. Solnon (Ed.), *28th international conference on principles and practice of constraint programming, CP 2022, july 31 to august 8, 2022, haifa, israel* (29:1–29:16, Vol. 235). Schloss Dagstuhl - Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.CP.2022.29 (cit. on pp. 5, 6).

Prestwich, S. D., Freuder, E. C., O'Sullivan, B., & Browne, D. (2021). Classifier-based constraint acquisition. *Ann. Math. Artif. Intell.*, *89*(7), 655–674. https://doi.org/10.1007/s10472-021-09736-4 (cit. on pp. 5, 6).