Constraint Acquisition via Partial Queries

Christian Bessiere\textsuperscript{1} Remi Coletta\textsuperscript{1} Emmanuel Hebrard\textsuperscript{2} George Katsirelos\textsuperscript{3} Nadjib Lazaar\textsuperscript{1} Nina Narodytska\textsuperscript{4} Claude-Guy Quimper\textsuperscript{5} Toby Walsh\textsuperscript{4}

\textsuperscript{1}CNRS, U. Montpellier, France
\textsuperscript{2}LAAS-CNRS, Toulouse, France
\textsuperscript{3}INRA Toulouse, France
\textsuperscript{4}NICTA, UNSW, Sydney, Australia
\textsuperscript{5}U. Laval, Quebec City, Canada
Motivations

Problem CSP solution
Motivations

- **Question**: How does the user write down the constraints of a problem?
- **Limitations**: modelling constraint networks require a fair expertise
- **Need**: Simple way to build constraint model ➔ Modeller-assistant
Motivations

- **Question:** How does the user write down the constraints of a problem?
- **Limitations:** modelling constraint networks require a fair expertise

- **Need:** Simple way to build constraint model ➔ Modeller-assistant
- **How:** In a Machine Learning way (passive/active, offline/online, by reinforcement...)

---

Learning process

- Solutions
- Non-solutions

CSP

Problem

Solution
• **Question:** How does the user write down the constraints of a problem?
• **Limitations:** modelling constraint networks require a fair expertise

• **Need:** Simple way to build constraint model ➔ Modeller-assistant
• **How:** In a Machine Learning way (passive/active, offline/online, by reinforcement...)
• **Question:** How does the user write down the constraints of a problem?
• **Limitations:** modelling constraint networks require a fair expertise

• **Need:** Simple way to build constraint model ➔ Modeller-assistant
• **How:** In a Machine Learning way (passive/active, offline/online, by reinforcement...)
inputs:

- $(X, D)$: Vocabulary
- $B$: Bias (possible constraints)
- $C_T$: Target network
- $(E^+, E^-)$: positives and negatives

output:

- $C_L$: Learnt network s.t.,
  
  $C_L \subseteq B : C_L \equiv C_T$
Example

- $\Gamma = \{<, =\}$
- $B = \{x_i < x_j, x_i = x_j, \forall i, j\}$
- $C_T = \{x_1 = x_3, x_1 < x_2\}$
- $C_L = \{x_1 = x_3, x_3 < x_2\}$
CONACQ

- SAT-Based constraint acquisition
- Bidirectional search
- Conacq1.0 (passive learning) [Bessiere et al. ECML05]
- Conacq2.0 (active learning) [Bessiere et al. IJCAI07]
**CONACQ**

- **SAT-Based constraint acquisition**
- **Bidirectional search**
- **Conacq1.0 (passive learning)** [Bessiere et al. ECML05]
- **Conacq2.0 (active learning)** [Bessiere et al. IJCAI07]

**Diagram:**

\[ \mathcal{K} = (\neg x_1 \land \neg x_2 \land \neg x_3) \land (x_4 \lor x_5 \lor x_6 \lor x_7) \ldots \]
State of the art

**CONACQ**
- SAT-Based constraint acquisition
- Bidirectional search
- Conacq1.0 (passive learning) [Bessiere et al. ECML05]
- Conacq2.0 (active learning) [Bessiere et al. IJCAI07]

\[ \mathcal{K} = (\neg x_1 \land \neg x_2 \land \neg x_3) \land (x_4 \lor x_5 \lor x_6 \lor x_7) \ldots \]

**ModelSeeker** [Beldiceanu and Simonis, CP12]
- A passive learning
- Based on global constraint catalog (more than 400)
- Bottom-up search
QUACQ: Quick Acquisition

- QUACQ [Bessiere et al. IJCAI13]
  - Active learning approach
  - Bidirectional search
  - But it can be top-down search only if no positive example
  - Based on partial queries to elucidate the scope of the constraint to learn
Partial Queries

\[ \text{ask}(2, 8, 4, 2, 6, 5, 1, 6) \]
Partial Queries

ask(2, 8, 4, 2, 6, 5, 1, 6) = No
ask(2, 8, 4, 2, -, -, -, -) = No
Partial Queries

ask(2, 8, -, -, -, -, -, -) = Yes
Partial Queries

\[
\text{ask}(2, 8, 4, -, -, -, -, -, -) = \text{No}
\]
QUACQ: Quick Acquisition

- ask(e)
- yes
- reduce(B)
- Gen-query
QUACQ: Quick Acquisition

- **yes**
  - reduce(B)
- **ask(e)**
  - Gen-query
- **No**
  - partial-ask(e)
- **FindScope**
QUACQ: Quick Acquisition

- **Yes**: reduce(B)
- **No**: partial-ask(e)
- **ask(e)**: Gen-query
- **FindScope**: scope
- **FindC**: FindC

Diagram shows a decision process with nodes for reduce, ask, Gen-query, FindScope, and FindC connected by arrows indicating flow and decision points.
QUACQ: Quick Acquisition

- yes
  - reduce(B)
  - Gen-query
- ask(e)
- No
  - partial-ask(e)
  - FindScope
  - Update(C_L)
  - C
  - scope
  - FindC
QUACQ: Quick Acquisition

- **QUACQ**
  - Gen-query
  - B = \emptyset
  - Update(C_L)
  - FindScope
  - FindC

- **Reduction**
  - reduce(B)

- **Decision**
  - yes
    - ask(e)
  - no
    - partial-ask(e)

- **Output**
  - C_L
The number of queries required to find the target concept is in:

\[ O(\|C_T\| \cdot (\log |X| + |\Gamma|)) \]

The number of queries required to converge is in:

\[ O(|B|) \]
Some Results

- **Random**
  - Under-constrained instance \((X,D,C) = (50, 10, 12)\)
  - Phase transition instance \((X,D,C) = (50, 10, 122)\)
  - \(|B| = 7350\) built on \(\Gamma = \{=, \neq, <, \geq, >, \leq\}\)
Some Results

Random

- Under-constrained instance (X,D,C)=(50, 10, 12)
- Phase transition instance (X,D,C)=(50, 10, 122)
- |B|= 7350 built on $\Gamma = \{=, \neq, <, \geq, >, \leq\}$

|          | $|C_L|$ | #q  | #q_c | $\bar{q}$ | time |
|----------|--------|------|------|------------|------|
| rand_50_10_12 | 12     | 196  | 34   | 24.04      | 0.23 |
| rand_50_10_122| 86     | 1074 | 94   | 13.90      | 0.14 |

Intel Xeon E5462 @ 2.80GHz with 16 Gb of RAM.
Some Results

- Zebra puzzle
- QUACQ behavior on different bias sizes
Some Results

Sudoku

A target network on 81 variables with 810 constraints

| $|C_L|$ | #q | #q_c | $\bar{q}$ | time |
|---|---|---|---|---|
| Sudoku 9 × 9 | 810 | 8645 | 821 | 20.58 | 0.16 |
QUACQ: new constraint acquisition approach based on partial queries

- Active learning approach
- Learning a constraint in a log scale of #queries
- Queries are often much shorter than membership ones
- Can follow a top-down search to learn a constraint network
- QUACQ as a solver
  - QUACQ does not require positive examples
  - we can use it to solve an instance

- Ask more than yes/no questions
  - GENACQ for Generalization Acquisition [ECAI14] (next talk!)