Part 3: Interactive Constraint Learning

Luc De Raedt, Andrea Passerini, **Stefano Teso**

Why interactive learning?

- In settings where supervision is expensive, select instances to be labelled carefully
- Speed-up learning by asking smart queries to a sufficiently knowledgeable oracle
- Learn a model of an expert by interacting with her
- Elicit the preferences of a customer by asking her to judge alternative products

Note related to *Active Learning* and *Preference Elicitation* (later)

From offline to interactive

There is a hidden, **target theory** C^* over domain $\mathcal{X}$

- **Offline**

  Given instances x_i labelled by $y_i = \mathbb{1}\{x_i \models C^*\}$

  Find a theory C s.t. $y_i = 1 \Leftrightarrow \mathbb{1}\{x_i \models C\}$ for all $i = 1, \ldots, n$

$$\Downarrow$$

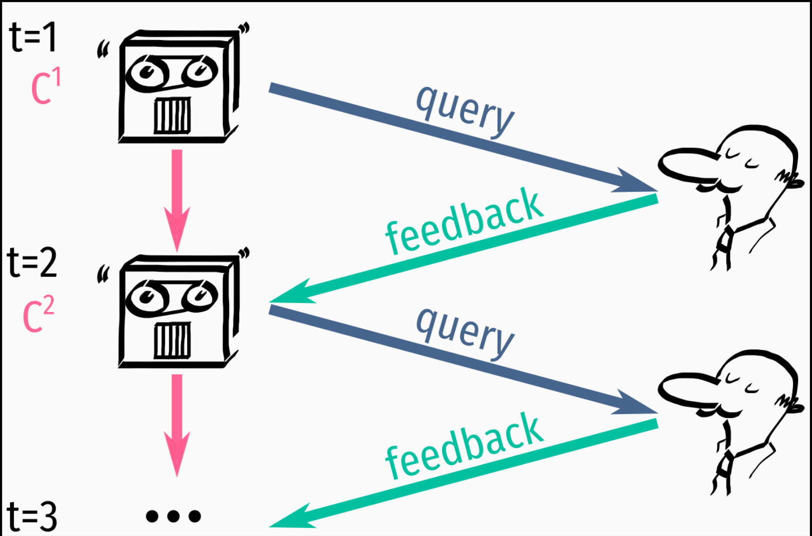
- **Interactive**

  Given **an oracle that answers queries by consulting** C^*

  Find a theory C consistent with all answers

(The soft constraints case changes analogously)

The generic learning loop



The generic learning loop

procedure LEARN (max iterations T)

$C^1 \leftarrow$ initial theory

for $t = 1, \ldots, T$ **do**

Choose a query q (e.g. an instance x)

Ask q to the oracle

Receive feedback (e.g. whether x is a model of C^*)

$C^{t+1} \leftarrow$ update C^t according to feedback

return C^T

Question what *kinds* of queries can be asked?

Question how to pick an *informative* query?

What kinds of questions?

Hard constraints

- Does x satisfy C^*? (membership)
- Does $x[V]$ satisfy C^*? (partial membership)
- Are C and C^* logically equivalent? If not, provide a counter-example (equivalence)

Soft constraints[1]

- What is the score $f(x)$ of x? (scoring)
- Is $f(x) \geq f(x')$? (ranking)
- Produce a configuration x' s.t. $f(x') > f(x)$ (improvement)

[1] E.g., for wCSP the score is $f(x) = \sum_i w_i \mathbb{1}\{x \models c_i\}$

Outline

- Hard theories — version spaces
  - Membership queries
  - Partial membership queries
  - Equivalence queries
- Soft theories, weight learning
  - Scoring queries (regression)
  - Ranking queries
  - Improvement queries (coactive learning)
- Soft theories, weight and constraint learning
  - Critiquing
- Links to related fields

Hard Constraints

Hard constraints

Assumption: hypothesis space $\mathcal{H}$ **contains** C^*

- C^* can be reconstructed perfectly

Assumption: the oracle is a **domain expert**

- Does *always* interpret/understand the queries
- Very dedicated, so *always* provides correct feedback

(Could also be a robot or a measurement apparatus)

$\Rightarrow$ *Realizable case*: enables **version space** approaches [Mit81]

Membership Queries: Monomials [BDH+16]

- Current hypothesis (conjunction)

$$C^t = \{\neg X_1, X_2, \neg X_3, X_4, \neg X_5\}$$

- To check whether $\neg X_1$ is really necessary, generate instance

$$x = \{X_1, X_2, \neg X_3, X_4, \neg X_5\}$$

- Ask membership query "$x \models C^*$?"
  - if positive, $\neg X_1$ is not necessary, delete it from C^t
  - if negative, $\neg X_1$ is necessary, keep it

Only $n + 1$ questions needed to recover C^*

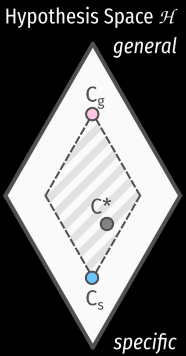
Membership Queries: ConAcq [BDH+16]

Bi-directional search

version space identified by

- most general candidate C_g
- most specific candidate C_s

C^* always within version space

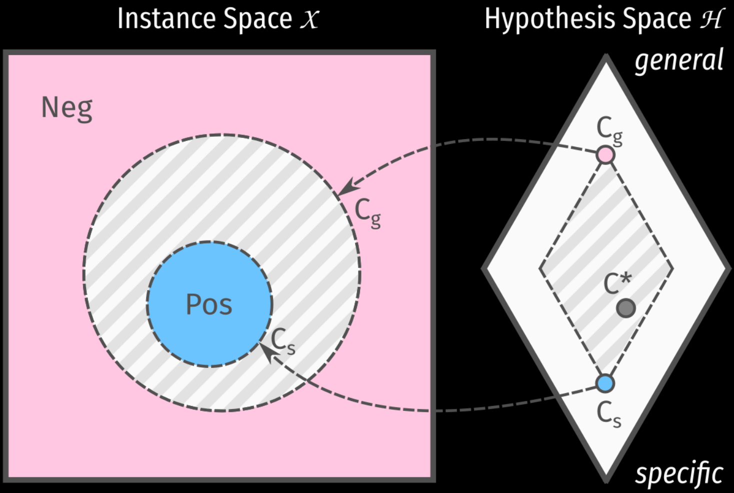


Idea pick instance $x \in \mathrm{Sol}(C_g) \setminus \mathrm{Sol}(C_s)$

- If x is positive, generalize most specific candidate
- If x is negative, specialize most generic candidate

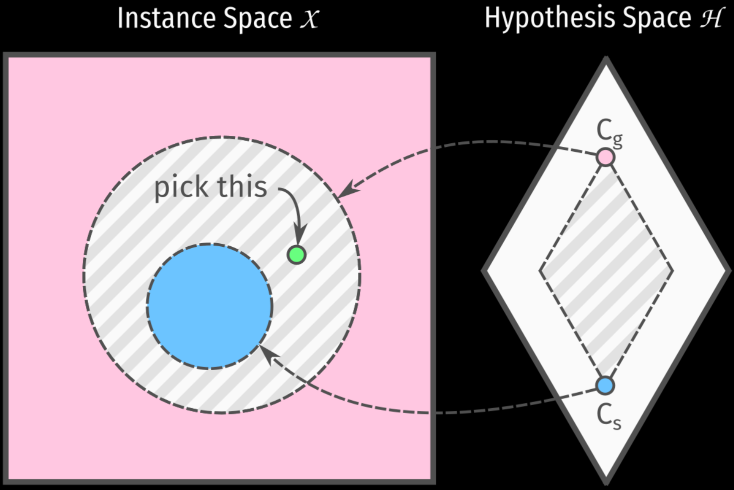
where $\mathrm{Sol}(C) = \{x : x \models C\}$

Version space and Instances



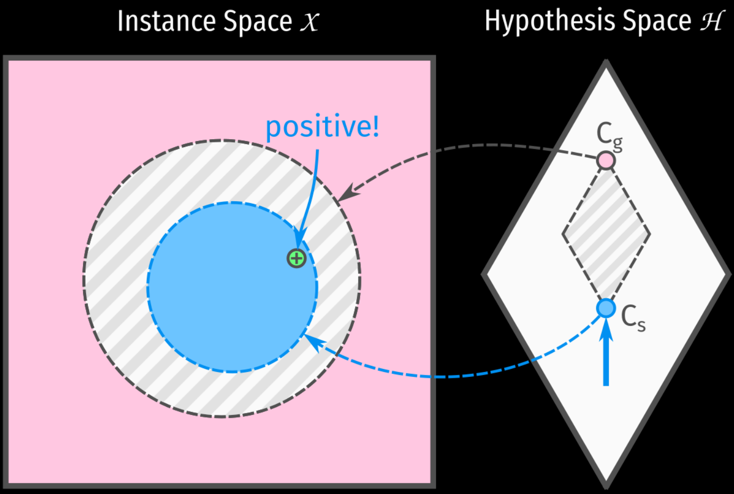
Left: Sol(C) is *inside* the circle

Query Selection



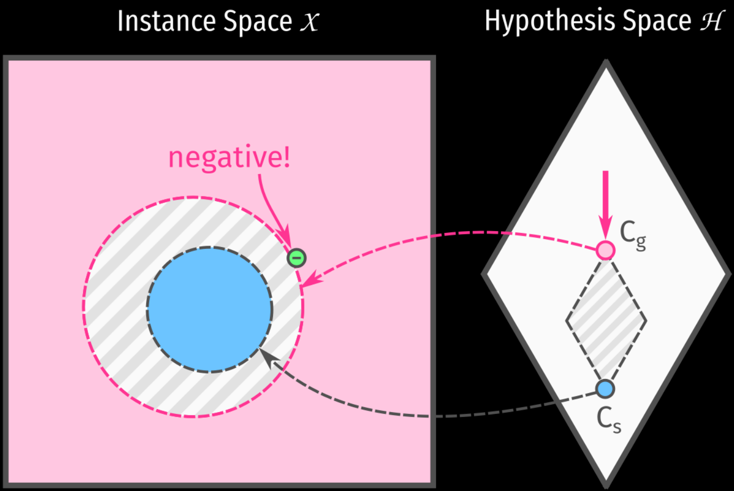
Select instance $x \in \mathrm{Sol}(C_g) \setminus \mathrm{Sol}(C_s)$

Positive $\Rightarrow$ generalize C_s



Generalizing C_s = removing constraints from it

Negative ⇒ specialize C_g



Specializing C_g = adding constraints to it

Partial Queries: Quacq [BDH+16]

Consider learning the 8-queens constraints

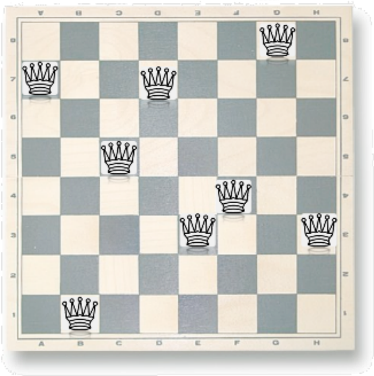
- Membership: does the board x satisfy all constraints, i.e., is it a solution?
- Partial membership: does a partial board (e.g. a 2×2 sub-board) violate at least one constraint?

The latter provides more information and more convergence: all completions of the partial configuration are also negative!

(It is also easier to answer from the oracle's perspective)

Example (C. Bessiere)

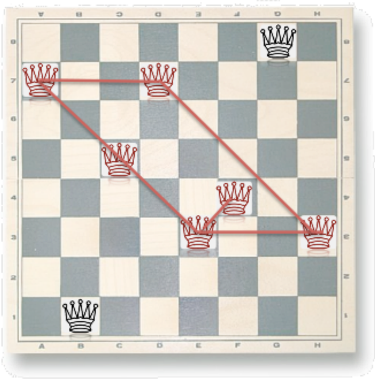
Partial Queries



ask(2, 8, 4, 2, 6, 5, 1, 6)

Example (C. Bessiere)

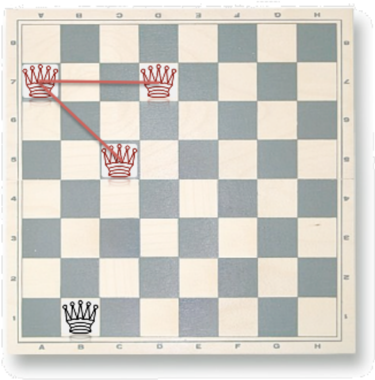
Partial Queries



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

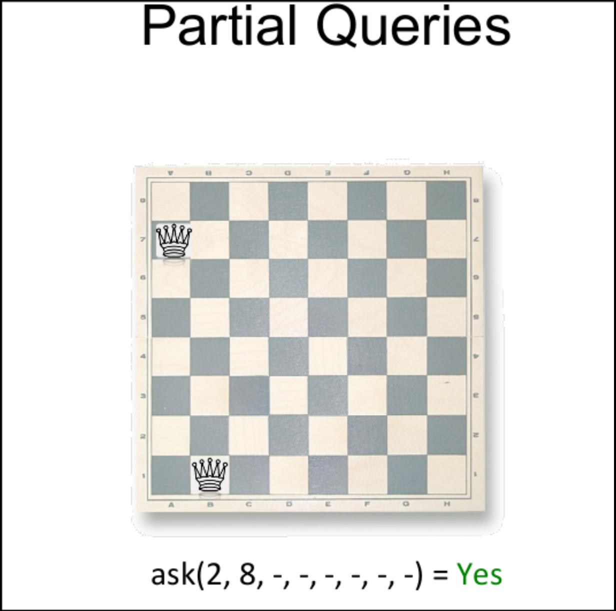
Example (C. Bessiere)

Partial Queries



ask(2, 8, 4, 2, -, -, -, -) = No

Example (C. Bessiere)



Equivalence Queries [Ang88]

Ask whether C^t is the same as C^*

If not, the oracle provides a counter-example x such that

$$x \models C^* \wedge x \not\models C$$

or vice-versa.

Equivalence queries can be simulated by poly-many membership queries.

More powerful than membership queries [BKLO17]:

All target concepts can be learned with eq. queries!

(Alas, impractical: even domain experts may have trouble answering them.)

Soft Constraints

Weighted Constraint Satisfaction Problems (wCSP)

Definition (same as before)

Given

- A set of pairs $\{(c_i, w_i)\}_{i=1}^{n}$ where:
  - c_i is a (soft) constraint
  - $w_i \in \mathbb{R}$ is a weight
- An indicator function $\mathbb{1}\{x \models c\}$ evaluating to one if c is satisfied by x, and zero otherwise

Find

$$x^* = \operatorname*{argmax}_{x \in \mathcal{X}} f(x) = \operatorname*{argmax}_{x \in \mathcal{X}} \sum_{i=1}^{n} w_i \cdot \mathbb{1}\{x \models c_i\}$$

Note hard constraints can be incorporated in $\mathcal{X}$

A user wishes to buy a custom PC. The PC is assembled from individual components: CPU, HDD, RAM, etc. Valid PC configurations must satisfy constraints, e.g. CPUs only work with compatible motherboards [TDP17]

Hard: "Intel CPUs are incompatible with AMD motherboards"

Soft: "The user prefers one CPU over another"

Interactive learning of soft theories

Assumption: hypothesis space $\mathcal{H}$ **contains** (C^*, w^*)

- For weight learning, we can reconstruct $\boldsymbol{w}^*$ perfectly

Depending on application: the oracle is not a **domain expert**

- May **not interpret/understand** the queries
- May provide **noisy** feedback

(For instance, a *customer* on an e-commerce website)

As a consequence, the **version space** may be empty!

The generic weight learning loop

procedure $\textsc{LearnWeights}$ (C, max iterations T)

$\boldsymbol{w}^1 \leftarrow$ initial weights

for $t = 1, \ldots, T$ **do**

Choose a query q (e.g. an instance x)

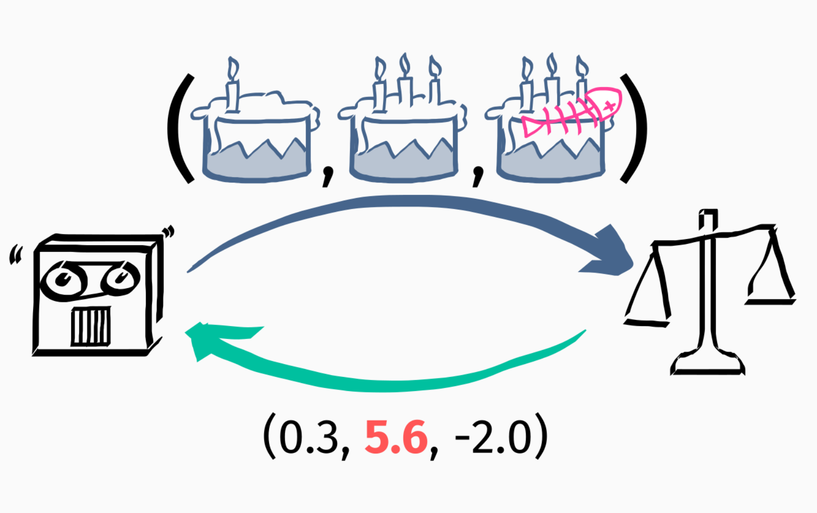
Ask q to the oracle

Receive feedback (e.g. what is the score of x w.r.t. f^*)

$\boldsymbol{w}^{t+1} \leftarrow$ update $\boldsymbol{w}^t$ according to feedback

return $\boldsymbol{w}^T$

Scoring Queries (for a wCSP about cakes!)



A pretty ideal setup — can observe $\{f^*(x_i)\}_i$ directly!

Weight learning of wCSP via regression [RS04]

Idea same as offline case, except $\mathcal{D}$ is built interactively

procedure LEARN (max iterations T, sample set size k)

$\mathcal{D} \leftarrow \varnothing$

$\boldsymbol{w} \leftarrow$ initial weights

for $t = 1, \ldots, T$ **do**

Sample $x_1, \ldots, x_k \in \text{argmax}_{x \in \mathcal{X}} f(x; \boldsymbol{w})$

Present $\{x_1, \ldots, x_k\}$ to the oracle

$\mathcal{D} \leftarrow \mathcal{D} \cup \{(x_j, y_j)\}_{j=1}^{k} \qquad (y_j = f^*(x_j) + \text{noise})$

$\boldsymbol{w}^{t+1} \leftarrow$ solve regression over $\mathcal{D}$

return $\boldsymbol{w}^T$

Regression amounts to solve, e.g.,

$$\boldsymbol{w}^{t+1} \leftarrow \underset{\boldsymbol{w}}{\text{argmin}} \sum_{(x,y) \in \mathcal{D}} (f(x; \boldsymbol{w}) - y)^2$$

A scoring oracle may not be available

$f^*($  $) = ?$

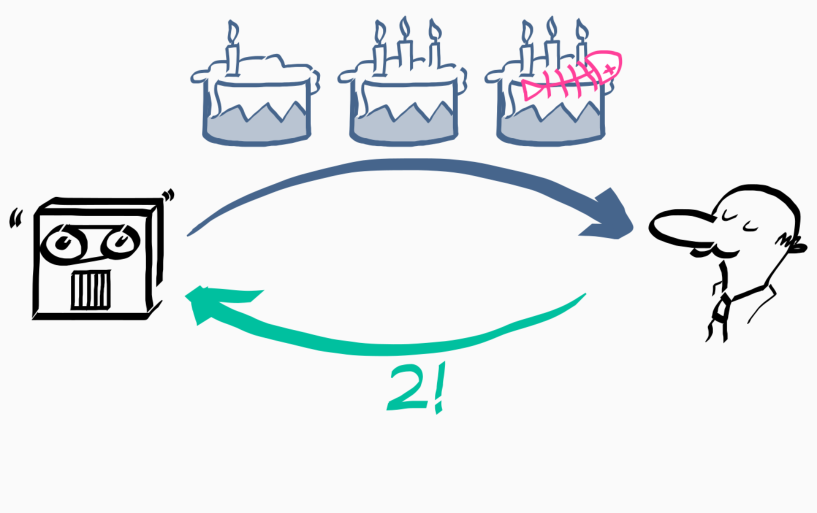
$f^*($

	7-8	17-19	19-20
Mon	running	--	swimming
Tue	--	--	gym
Wed	running	--	--
Thur	swimming	--	gym
Fri	--	soccer	soccer

$) = ?$

even experts may not be able to provide absolute scores reliably

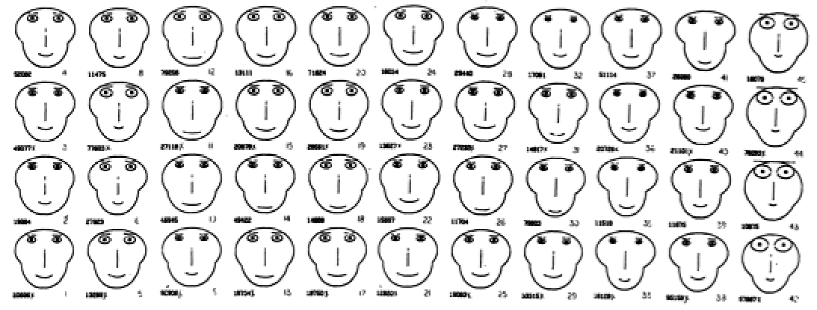
Ranking Queries



relative judgements only — no more pesky absolute scores

Ranking does not solve everything

What if it is hard to compare alternatives, e.g. they are too similar or too diverse? What if there are just too many?



From `https://eagereyes.org/criticism/chernoff-faces`

Preference learning for wCSP via ranking[2]

Offline case: SVM ranking

$$\begin{aligned} \min_{w} \quad & ||w||^2 + \lambda \textstyle\sum_{j=1}^{m} \xi_j \\ s.t \quad & f(x_j) - f(x'_j) \geq 1 - \xi_j \quad \forall j \in [1, n] \end{aligned}$$

where:

- $f(x) = \sum_{i=1}^{n} w_i \cdot \mathbb{1}\{x \models c_i\}$
- ξ_j is a penalty for not ranking x_j higher than x'_j with a large enough margin
- $||w||^2$ is a regularization term (margin is $2/||w|| \rightarrow$ large margin separation)
- $\lambda \in R^+$ trades off margin and correct rankings

[2]Slide: Andrea Passerini

Ranking for weight learning of wCSP [?]

Simple extension of offline ranking SVM

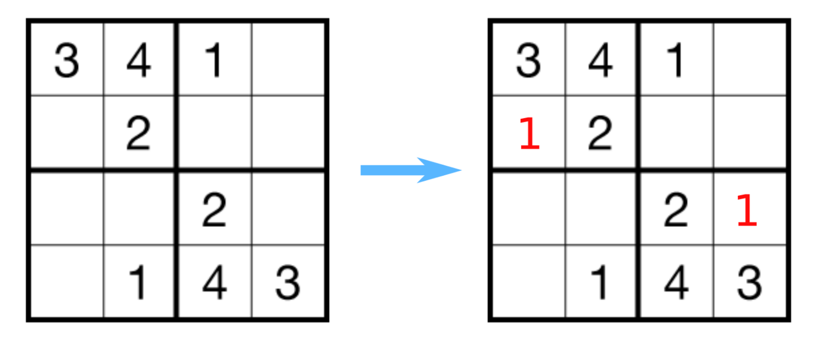
procedure LEARN (max iterations T)
  $C^1, \boldsymbol{w}^1 \leftarrow$ initial theory, initial weights
  for $t = 1, \ldots, T$ **do**
    Choose x, x' to be high scoring and reasonably diverse
    Present (x, x') to the oracle
    Add oracle ranking $x \succcurlyeq x'$ to $\mathcal{D}$
    $\boldsymbol{w}^{t+1} \leftarrow$ learn ranking from $\mathcal{D}$
  return $\boldsymbol{w}^T$

High score and reasonably diverse means solving, e.g.

$$\operatorname*{argmax}_{x,x'} f(x; w) + f(x'; w) + \alpha \cdot d(x, x')$$

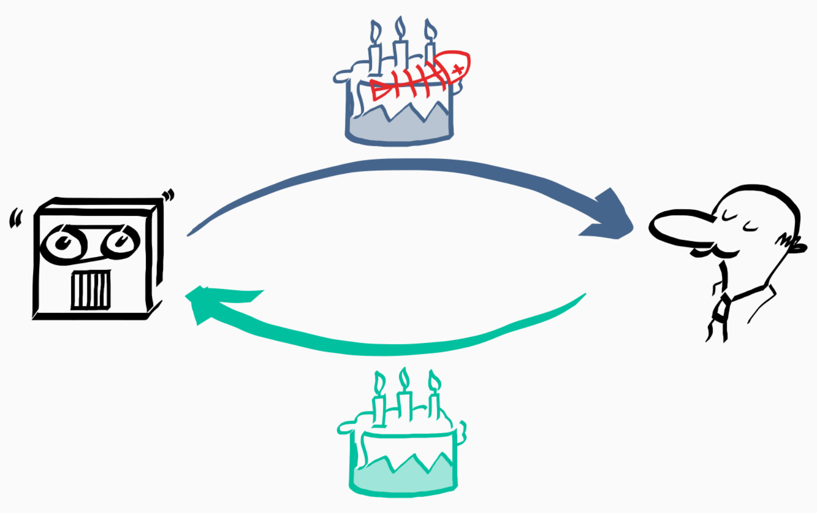
$$\text{s.t. } d(x, x') \leq d_{\max}$$

What if it is easy to manipulate the configurations?



It's possible to avoid "*k*-way inference" by asking the user to improve the current best configuration

Improvement Queries



boils down to a pairwise preference $f^*(\bar{x}^t) \geq f^*(x^t)$

Coactive Learning for weight learning of wCOP [SJ15]

Perceptron-based preference learning from improvement queries

procedure LEARN (max iterations T)

$C^1, \boldsymbol{w}^1 \leftarrow$ initial theory, initial weights

for $t = 1, \ldots, T$ **do**

$x^t \leftarrow \text{argmax}_{x \in \mathcal{X}} \sum_i w_i \phi_i(x)$

Present x^t to the oracle

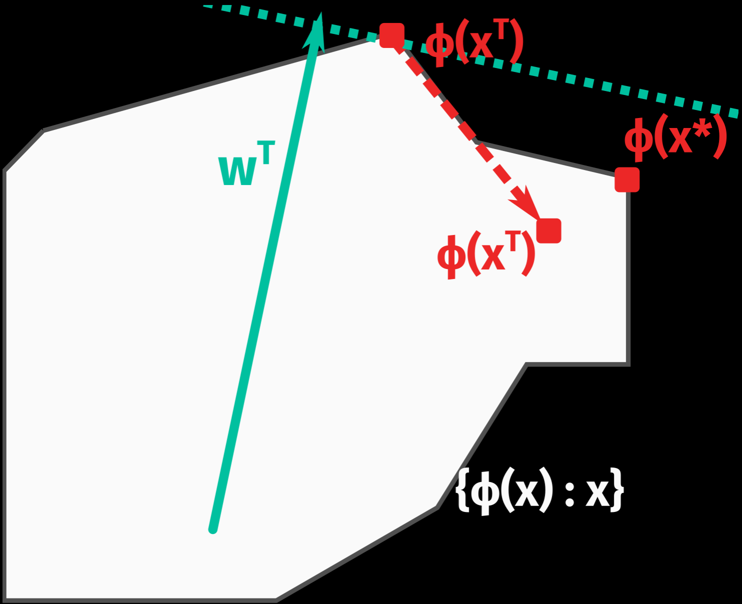
Obtain improved configuration $\bar{x}^t$

$\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \phi(\bar{x}^t) - \phi(x^t)$

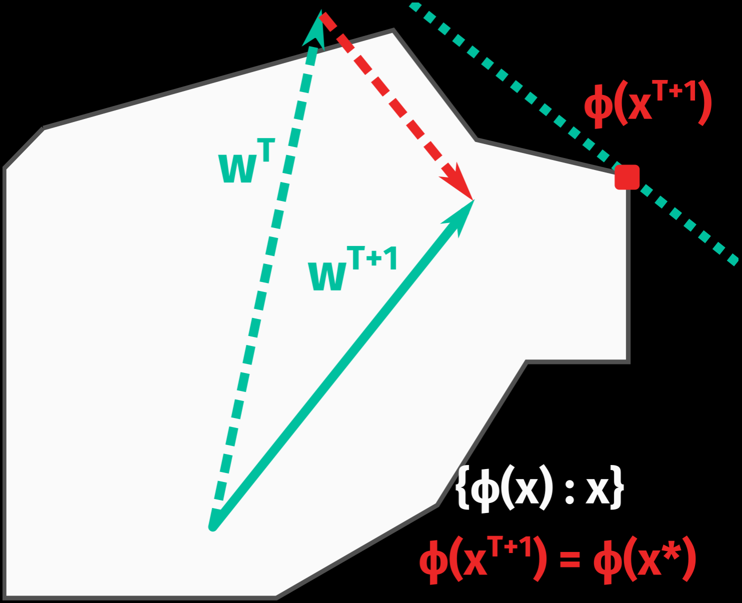
return C^T

Note quality of configurations approaches optimum as $O(1/\sqrt{T})$ under assumptions on the improvements

Coactive learning: iteration T



Coactive learning: iteration $T+1$



Coactive Critiquing [TDP17]

Recall that constraints $\approx$ features in wCSP

$$\phi_i(x) = \mathbb{1}\{x \models c_i\} \qquad \forall i = 1, \ldots, n$$

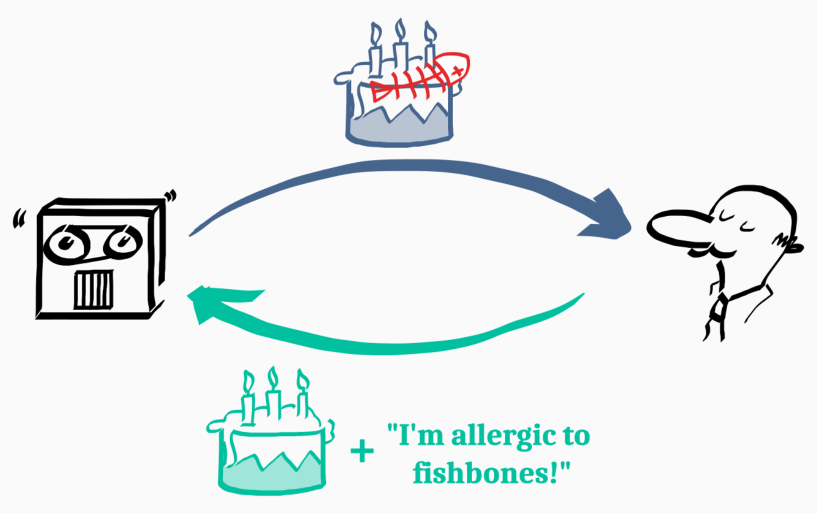
What if we don't have all of the constraints/features?

Idea

- If oracle improvement can't be explained by the learner (e.g. by linear spearability), a constraint is missing
- Ask for the missing constraint, acquire c_{n+1}

Add $\phi_{n+1} = \mathbb{1}\{x \models c_{n+1}\}$ to the pool of features, proceed as usual with Coactive Learning

Critiquing Queries



add the critique to the constraints, update the feature space

Critiquing Queries

A few remarks

- Can be proven to converge under assumptions, even if most constraints are acquired on-the-fly
- Not really "learning"
    - Critiquing queries provide the missing constraints

Once more: **powerful oracles make learning easier**

Still much work to do!

- How to combine learning of hard **and** soft constraints?
- What are the **“best” queries** in the soft setting?
- How to properly deal with **rationally bounded** oracles?
  e.g. how to combine technology and human interaction?
- How far can we **push** and/or **guide** the oracle?
  e.g. how to best exploit and control human abilities?

(and much more)

Related frameworks

Many topics related to interactive constraint learning

- Pool-based **Active Learning**
- **Preference Elicitation** (for interactive recommendation)
- Inverse Combinatorial Optimization [Heu04]
- Programming by Feedback [ASSS14]
- ...

We briefly overview the connection to CL; details in the references

Pool-based Active Learning [Set12, Han14]

Given hidden decision function $f^* : \mathcal{X} \to \{\pm 1\}$, instances $x_1, \ldots, x_n \in \mathcal{X}$, and an oracle that labels instances with f^*

Find a good estimate f of f^* with *as few queries as possible*

Remarks

- Like CP, focuses on quality of learned model $\text{loss}(f^*, f)$
- . . . but geometrical flavor: SVMs, Gaussian Processes
- **Many strategies in common** (e.g. version spaces)

Query types: labeling queries ($\approx$ membership), search queries ($\approx$ equivalence), rationales and explanations, . . .

Preference Elicitation [Bou02, PTV16]

Given products $x_1, \ldots, x_n \in \mathcal{X}$ and a user who ranks alternatives by relative preferrability

Find a good item $x \in \mathcal{X}$ with *the least cognitive effort*

If we knew the true user' scoring function $f(x)$, it would be easy!
But we don't, so we estimate it iteratively by asking queries

Remarks

- Must model **preferences**, similar to wCSP/wCOP
- wCSP/wCOP useful for recommending combinatorial items
- Unlike CP, **only quality of recommendation x matters**
- Learns approximation f of f^* only as byproduct

Methods: Bayesian, minimax regret, online learning

Wrapping up

Take-away

- Interactive acquisition can cut labeling cost and speed up learning
- Many classes of hard CSPs can be learned (also) interactively in the realizable setting via version space approaches
- Interactive weight learning of wCSP/wCOP can be cast as interactive regression/ranking + smart query selection
- Different kinds of queries affect both
  - theoretical learning efficiency [Ang88, BDH^{+}16]
  - ability to learn from human (non-)experts
- Plenty of applications, like extracting knowledge from experts and preferences from customers
- Plenty of interesting related areas

Thank you!

Dana Angluin.
Queries and concept learning.
Machine learning, 1988.

Riad Akrour, Marc Schoenauer, Michèle Sebag, and Jean-Christophe Souplet.
Programming by feedback.
In *International Conference on Machine Learning*, number 32, pages 1503–1511. JMLR. org, 2014.

Christian Bessiere, Abderrazak Daoudi, Emmanuel Hebrard, George Katsirelos, Nadjib Lazaar, Younes Mechqrane, Nina Narodytska, Claude-Guy Quimper, and Toby Walsh.
New approaches to constraint acquisition.
In *Data Mining and Constraint Programming*. 2016.

Christian Bessiere, Frédéric Koriche, Nadjib Lazaar, and Barry O'Sullivan.
Constraint acquisition.
Artificial Intelligence, 244:315–342, 2017.

Craig Boutilier.
A POMDP formulation of preference elicitation problems.
In *AAAI/IAAI*, 2002.

Steve Hanneke.
Theory of disagreement-based active learning.
Foundations and Trends® in Machine Learning, 2014.

Clemens Heuberger.
Inverse combinatorial optimization: A survey on problems, methods, and results.

Journal of combinatorial optimization, 2004.

- Tom M Mitchell.
  Generalization as search.
  In *Readings in artificial intelligence*, pages 517–542. Elsevier, 1981.

- Gabriella Pigozzi, Alexis Tsoukias, and Paolo Viappiani.
  Preferences in artificial intelligence.
  Annals of Mathematics and Artificial Intelligence, 2016.

- Francesca Rossi and Allesandro Sperduti.
  Acquiring both constraint and solution preferences in interactive constraint systems.
  Constraints, 9(4):311–332, 2004.

- Burr Settles.
  Active learning.

Synthesis Lectures on Artificial Intelligence and Machine Learning, 2012.

- Pannaga Shivaswamy and Thorsten Joachims.
Coactive learning.
JAIR, 2015.

- Stefano Teso, Paolo Dragone, and Andrea Passerini.
Coactive critiquing: Elicitation of preferences and features.
In *AAAI*, pages 2639–2645, 2017.