A NOVEL APPROACH TO
INDUCTIVE LOGIC PROGRAMMING

What is ILP?

positive examples

[b(N0), b(N1), b(N2), b(N3), r(N4), a(N4, N2), a(N1, N4),
a(No, N3), a(N1, N3), a(N1, No), a(N2, N1)]

hypothesis

negative examples

[b(N0), b(N1), b(N2), b(N3), r(N4), a(N4, N2), a(N1, N4),
a(No, N3), a(N1, N3), a(N1, No), a(N2, N1)]
ILP in detail

... , \text{arc}(a,b), \text{arc}(b,a), \text{arc}(a,c), \text{arc}(c,d), \text{red}(a), \text{blue}(c), \text{blue}(b) ...

\textbf{Does the hypothesis entail the example?}

\textit{\theta}\text{-subsumption}

\( H \subset E \)

\( \theta = \{ N1/a, N2/b \} \)

\textbf{Subsumption problem}

How to find an instantiation of variables in the hypothesis such that the hypothesis subsumes the example?

\( \text{arc}(N1,N2), \text{arc}(N2,N1) \)

\textbf{How to obtain the hypothesis from a given template?}

\( T \sigma = H \)

\( \sigma = \{ X4/X1, X3/X2 \} \)

\textbf{A template consistency problem}

How to find out which variables are unified in the template?

\textbf{Talk outline}

- **Subsumption checking**
  How to check whether the hypothesis subsumes an example?
  - Django algorithm

- **Template consistency**
  How to unify variables in the template to obtain a consistent hypothesis?
  - unification models with hints [AIMSA 2010]

- **Template generation**
  How to obtain a template?
  - stochastic history-driven generator [MICAI 2011]

- **Problem decomposition**
  How to improve overall efficiency?
  - a decompose-merge-refine approach [AIMSA 2012]
Subsumption problem

Does hypothesis \( H \) subsume an example \( E \)?

Recall:
- \textbf{example} is a set of grounded atoms
  \[ \ldots, \text{arc}(a,b), \text{arc}(b,a), \text{arc}(a,c), \text{arc}(c,d), \text{red}(a), \text{blue}(c), \text{blue}(b) \ldots \]
- \textbf{hypothesis} is a set of atoms with variables
  \[ \text{arc}(N1,N2), \text{arc}(N2,N1) \]

\( \theta \)-subsumption
- \( H \) subsumes \( E \) iff there exists a substitution \( \theta \) such that \( H\theta \subseteq E \)

Using Constraint Programming techniques to check \( \theta \)-subsumption.

What is CP?

\textbf{Constraint Programming} is a technology for solving combinatorial optimization problems modeled as constraint satisfaction problems:
- a finite set of decision \textbf{variables}
- each variable has a finite set of possible values (\textbf{domain})
- combinations of allowed values are restricted by \textbf{constraints} (relations between variables)

Mainstream \textbf{solving approach} combines
- \textbf{inference} (removing values violating constraints)
  with \textbf{search} (trying combinations of values)
Django algorithm

- The subsumption problem is formulated as a constraint satisfaction problem.

- Example defines the domains of constraints
  - atoms with the same name \( \rightarrow \) constraint domain
  - \( \ldots, \text{arc}(a,b), \text{arc}(b,a), \text{arc}(a,c), \text{arc}(c,d), \text{red}(a), \text{blue}(c), \text{blue}(b) \ldots \)
  - binary constraint \( \text{arc} = \{(a,b), (b,a), (a,c), (c,d)\} \)
  - unary constraint \( \text{blue} = \{(c), (b)\} \)

- Hypothesis formulates the CSP
  - atom with variables \( \rightarrow \) constraint
  - CSP: \( \text{arc}(N_1,N_2), \text{arc}(N_2,N_1) \)

Template consistency

How to obtain a hypothesis (from a given template) consistent with examples?

Recall:

- hypothesis is a set of atoms with shared variables \( \text{arc}(N_1,N_2), \text{arc}(N_2,N_1) \)
  - hypothesis H is consistent with examples iff H subsumes all positive examples and H does not subsume any negative example

- template is a set of atoms with unique variables \( \text{arc}(X_1,X_2), \text{arc}(X_3,X_4) \)
  - \( T\theta = H \), where \( \theta \) is a unification of variables

Using Constraint Programming techniques to solve the template consistency problem.
The concept

Subsumption model for positive example 1
Subsumption model for positive example 2
... 
Subsumption model for positive example N

A constraint model based on Django system describing whether the hypotheses subsumes an example.

Unification model keeps information about unified variables in hypothesis

Subsumption model for explored negative example

Index model (variables)

How to model which ILP variables are unified in the template?

template: arc(X₁,X₂), arc(X₃,X₄), blue(X₅)
hypothesis: arc(N₁,N₂), arc(N₂,N₁), blue(N₂)

- unification can be seen as a mapping
  - X₃ → X₂, X₄ → X₁, X₅ → X₂
  - always map the variable with the larger index to the variable with the smaller index
- mapping is modeled using index CP variables
  - Iᵢ with domain {1, ..., j}
    - ILP variable Xᵢ maps to ILP variable Xᵢᵢ
  - I₁=1, I₂=2, I₃=2, I₄=1, I₅=2
Index model (constraints)

- uniqueness: (X3 → X2, X5 → X2 vs. X3 → X2, X5 → X3)
- “symmetry breaking”: (blue(X4), blue(X5): X4 → X1, X5 → X2 vs. X4 → X2, X5 → X1)
- channeling: (from hypothesis to subsumption test)
- decisions (variables X2 and X3 are/aren’t unified)

Search framework

*How to decide about the unified variables?*

- Unification of variables in the template is necessary only to break subsumption of negative examples!

**Do** for all negative examples

**While** the example is subsumed by current hypothesis

- Find subsumption θ (solution to a corresponding CSP with variables X1,...,Xn)
- Select a pair Xi, Xj such that Xi θ ≠ Xj θ
- Post constraint Ii = Ij (alternatively Ii ≠ Ij)

Find subsumption for all positive examples
Boolean model

Problem: conflict $I_1 = I_2, I_1 \neq I_2$ is not discovered by local (arc) consistency!

We can strengthen inference by explicitly keeping information about unified/non-unified variables in a “Boolean matrix”.

$$
\begin{array}{|c|c|c|c|c|c|c|}
\hline
I_1 & I_2 & I_3 & I_4 & I_5 & I_6 \\
\hline
1 & 0 & A & B & C & D \\
0 & 1 & E & F & G & H \\
A & E & 1 & 0 & I & J \\
B & F & 0 & 1 & K & L \\
C & G & I & K & 1 & 0 \\
D & H & J & L & 0 & 1 \\
\hline
\end{array}
$$

$1_l = 1_j$

$1_l \neq 1_m, 1_m \neq 1_n, 1_n \neq 1_o, 1_o \neq 1_p$

Hints

- Assume that the example contains the following atoms for predicate arc:
  - $\text{arc}(a,b), \text{arc}(b,a), \text{arc}(a,c), \text{arc}(c,a)$
- Clearly $X_1$ can not unify to $X_2$ in $\text{arc}(X_1, X_2)$.
- But constraints $\text{arc}(X_1, X_2), X_1 = X_2$ are (arc) consistent!
- By exploring atoms with the same predicate symbol in positive examples, it is possible to deduce that some variables (of this atom in the hypothesis) cannot unify (hint).
  - $I_1 \neq I_2$
Experiments (inside)

- Comparison of runtimes (milliseconds) for identifying common structures in randomly generated structured graphs (Erdős-Rényi).

<table>
<thead>
<tr>
<th>#atoms</th>
<th>#vars</th>
<th>Index</th>
<th>Boolean</th>
<th>Combined</th>
<th>Decoupled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>full</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>0</td>
<td>15</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>0</td>
<td>16</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>31</td>
<td>281</td>
<td>31</td>
<td>21</td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>94</td>
<td>343</td>
<td>109</td>
<td>78</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>94</td>
<td>1046</td>
<td>125</td>
<td>93</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>328</td>
<td>1810</td>
<td>436</td>
<td>312</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>1232</td>
<td>9626</td>
<td>1606</td>
<td>1170</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>&gt;60000</td>
<td>86425</td>
<td>&gt;60000</td>
<td>236514</td>
</tr>
</tbody>
</table>

Experiments (outside)

- Comparison to widespread ILP system Aleph
Obtaining a template

How to obtain a template?

Recall:
- template consists of atoms with fresh variables (each variable appears exactly once)

The task:
- How many copies of each atom will appear in the template?

Searching the space of templates and learning the most promising atoms in the template.

Existing approach

Iterative deepening search
- generate all templates of given length
- if no consistent hypothesis found then increase the length

Features:
- guarantees finding the shortest hypothesis
- too slow (generate and test)
First idea

**Incremental probabilistic search**
- start with template containing one atom of each predicate symbol
- add new atoms randomly with uniform distribution
- if consistent hypothesis found then remove isolated atoms (do not share variables with other atoms)

**Features:**
- no guarantee of finding the shortest hypothesis
- faster convergence
- ready for tuning via the probability distribution for selecting added atoms

Second idea

**Stochastic history-driven tabu search**
- if the added atom is successful (increased the number of broken negative examples) then add it again
- if the added atom is not successful then put it to **tabu list** and select another atom (outside the tabu list) randomly
  - tabu list is emptied if all atoms are tabu, or added atom was successful
- stochastic version
  - probability of successful atom is set to a high value
  - probability of atoms returned from the tabu list is set to a low value
Experimental results

- looking for common sub-structure in random graphs (Barabási-Réka model)
- time limit 600 seconds
- 5 runs of stochastic algorithms

<table>
<thead>
<tr>
<th>IDS</th>
<th>time [s]</th>
<th>length</th>
<th>IPS</th>
<th>time [s]</th>
<th>length</th>
<th>SHDTS</th>
<th>time [s]</th>
<th>length</th>
<th>#unfinished</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.75</td>
<td>6</td>
<td>0.33</td>
<td>6</td>
<td>0.37</td>
<td>6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>13.97</td>
<td>7</td>
<td>2.34</td>
<td>7</td>
<td>1</td>
<td>1.85</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>13.92</td>
<td>7</td>
<td>1.31</td>
<td>7</td>
<td>0.62</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.27</td>
<td>5</td>
<td>0.11</td>
<td>5</td>
<td>1.17</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>455.43</td>
<td>8</td>
<td>334.51</td>
<td>8</td>
<td>-</td>
<td>273.75</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10.93</td>
<td>7</td>
<td>1.16</td>
<td>7</td>
<td>-</td>
<td>1.02</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>&gt;600</td>
<td>-</td>
<td>6.13</td>
<td>8</td>
<td>-</td>
<td>2.11</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>411.60</td>
<td>7</td>
<td>28.07</td>
<td>8</td>
<td>-</td>
<td>0.46</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>11.88</td>
<td>7</td>
<td>16.41</td>
<td>7</td>
<td>-</td>
<td>67.70</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>13.52</td>
<td>7</td>
<td>1.41</td>
<td>7</td>
<td>0.55</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Experimental results (2)

- looking for common sub-structure in random graphs (Erdös-Rényi model)
- time limit 1200 seconds

<table>
<thead>
<tr>
<th>nodes</th>
<th>IDS</th>
<th>time [s]</th>
<th>length</th>
<th>IPS</th>
<th>time [s]</th>
<th>length</th>
<th>SHDTS</th>
<th>time [s]</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.41</td>
<td>6</td>
<td>0.74</td>
<td>6</td>
<td>0.74</td>
<td>6</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>204.11</td>
<td>9</td>
<td>59.65</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>5.23</td>
<td>9</td>
<td>17.7</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>518.47</td>
<td>10</td>
<td>448.72</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>533.25</td>
<td>9</td>
<td>241.57</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>313.25</td>
<td>9</td>
<td>211.78</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>426.70</td>
<td>10</td>
<td>366.41</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>&gt;1200</td>
<td>-</td>
<td>181.28</td>
<td>9</td>
<td>215.01</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>&gt;1200</td>
<td>-</td>
<td>716.15</td>
<td>10</td>
<td>950.04</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>27.71</td>
<td>7</td>
<td>&gt;1200</td>
<td>-</td>
<td>3.98</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>&gt;1200</td>
<td>-</td>
<td>201.28</td>
<td>10</td>
<td>164.61</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Improving efficiency

How to improve efficiency of existing ILP algorithms?

Main idea:
- Problem decomposition (divide-and-conquer)

Our approach at glance:

- **Decompose**
  - split examples into several groups
  - find a consistent hypothesis for each group

- **Merge**
  - combine the hypotheses to obtain a single hypothesis consistent with all examples

- **Refine**
  - try to shorten further the hypothesis (Ockham’s razor)

Decomposition

- **How to split the examples?**
  - We will need to combine the obtained hypotheses!

- **Our proposal**
  - each group contains all positive examples and some negative examples (we only split the negative examples)

- **Claim**
  - union of obtained hypotheses is consistent
    - with all positive examples (subsumes them)
    - with all negative examples (does not subsume them)
Merging

- Can we do more sophisticated merging?
  - leading to a smaller hypothesis?
- Example:
  - $H_1 = \{\text{arc}_1(X_1,X_2), \text{arc}_2(X_3,X_1), \text{arc}_3(X_2,X_4), b_2(X_4)\}$
  - $H_2 = \{\text{arc}_2(Y_1,Y_3), \text{arc}_2(Y_3,Y_2), b_2(Y_2)\}$
  What if we unify $\text{arc}_1(X_2,X_4)$ and $\text{arc}_2(Y_1,Y_2)$?
  - $\{\text{arc}_1(X_1,X_2), \text{arc}_1(X_3,X_1), \text{arc}_1(X_2,X_4), b_1(X_4), \text{arc}_2(X_2,Y_3)\}$
- Claim
  - After unifying two predicates in hypotheses being merged
    - the final hypothesis is consistent with negative evidence
    - but may be no more consistent with positive evidence!
- In practice
  - we explore all possible pairs for unification
  - If consistency is violated after unification then the unification is excluded

Refinement

- Can we further shorten the hypothesis?
  - we can remove some predicate(s)
- Claim
  - After removing a predicate from the hypothesis
    - the final hypothesis is consistent with positive evidence
    - but may be no more consistent with negative evidence!
- In practice
  - we try to remove the largest subset of predicates
  - after each removal, we try to restore consistency by adding extra unifications of variables (the original template consistency algorithm)
  - if consistency is violated then the predicate is returned back
Experiments (setting)

- So, are we really better?
- Experimental setting
  - implemented in SICStus Prolog 4.1.2
  - 2.0 GHz Intel Xeon with 12 GB RAM
  - problems – looking for a common structure in graphs
    - random graphs generated using Barabási-Reka model (new nodes connected with three random arcs)
    - 20 nodes in graph, hidden structure of 5 nodes
    - 10 positive and 10 negative examples

Experiments (results)

<table>
<thead>
<tr>
<th>Alg. 1</th>
<th>Aleph</th>
<th>Decomposition</th>
<th>Merge</th>
<th>Refinement</th>
<th>t_total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>t len</td>
<td>t len</td>
<td>t len</td>
<td>t len</td>
</tr>
<tr>
<td>17.4</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>1.7 43</td>
<td>2.0 11</td>
</tr>
<tr>
<td>436</td>
<td>8</td>
<td>&gt;1200</td>
<td>-</td>
<td>7.2 54</td>
<td>6.3 20</td>
</tr>
<tr>
<td>460</td>
<td>8</td>
<td>&gt;1200</td>
<td>-</td>
<td>54.2 60</td>
<td>39.3 29</td>
</tr>
<tr>
<td>508</td>
<td>8</td>
<td>&gt;1200</td>
<td>-</td>
<td>23.9 53</td>
<td>10.9 43</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>10.2 51</td>
<td>6.8 15</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>23.4 55</td>
<td>15.4 33</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>90.9 60</td>
<td>31.3 31</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>88.8 61</td>
<td>41.8 36</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>103.1 53</td>
<td>41.5 28</td>
</tr>
<tr>
<td>&gt;1200</td>
<td>-</td>
<td>&gt;1200</td>
<td>-</td>
<td>99.9 57</td>
<td>48.3 34</td>
</tr>
</tbody>
</table>
Experiments (looking inside)

Summary

- **What has been done?**
  - checking subsumption using CP
  - finding consistent hypothesis from a template using CP
  - generating templates via search with learning
  - Improving efficiency via problem decomposition

- **What is next?**
  - a possible extension to noisy data
  - combination with other ILP algorithms