Linear Programming based Induction (LPI): a Constraint Logic Programming elaboration

George Potamias

Institute of Computer Science
Foundation for Research and Technology - Hellas (FORTH)

Science & Technology Parc of Crete
Vassilika Vouton
P.O. Box 1385
711 10 Heraklion, Crete
GREECE

1st Panhellenic Symposium on Logic
21-24 July, Cyprus
An example: logical \textit{and}

\textit{BINARY VARIABLE} representatives of attribute-values

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>AB</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

\begin{align*}
\text{A} = t &: A11 \\
\text{A} = f &: A12 \\
\text{B} = t &: A21 \\
\text{B} = f &: A22 \\
\text{AB} = t &: C1 \\
\text{AB} = f &: C2
\end{align*}

\text{Aij, Ck in \{0, 1\}}

\textit{CONTRAINTs based representation of examples}

\begin{align*}
\text{A} = t \ \text{and} \ \text{B} = t &: \Rightarrow \ AB = t \\
\text{A} = t \ \text{and} \ \text{B} = f &: \Rightarrow \ AB = f \\
\text{A} = f \ \text{and} \ \text{B} = t &: \Rightarrow \ AB = f \\
\text{A} = f \ \text{and} \ \text{B} = f &: \Rightarrow \ AB = f
\end{align*}

from clauses to constraints

\begin{align*}
\text{A11} \ast \text{A21} &: \leq \ C1 \\
\text{A11} \ast \text{A22} &: \leq \ C2 \\
\text{A12} \ast \text{A21} &: \leq \ C2 \\
\text{A12} \ast \text{A22} &: \leq \ C2
\end{align*}

The \text{PRODUCT} of the \text{PREMISE} binary variable values is \text{less_or_equal} of the \text{CLASS} binary variable value

\[ e1: \ 1 \ast 1 \leq 1 \Rightarrow A11 = 1 , A21 = 1 \Rightarrow C1 = 1 \]
An example: logical and

- From NON-LINEAR to LINEAR constraints .... more efficient solvers


\[
\begin{align*}
A_{11} \times A_{21} & \leq C_1 \\
A_{11} \times A_{22} & \leq C_2 \\
A_{12} \times A_{21} & \leq C_2 \\
A_{12} \times A_{22} & \leq C_2 \\
1 - \sum_{i=1}^{2} \sum_{j=1}^{2} (1 - A_{ij}) & \leq C
\end{align*}
\]

- Some EXTRA CONSTRAINTS

**MUTUAL-EXCLUSIVE Attribute values**

\[
\sum_{j=1}^{2} A_{ij} \leq 1, \text{ for all } 1 \leq i \leq 2
\]

**MUTUAL-EXCLUSIVE Class values**

\[
\sum_{k=1}^{2} C_k \leq 1
\]
Boolean Function Synthesis with LPI - continued

An example: logical and

.... BUT

\[
\begin{align*}
(\text{Ic}) & \iff \sum_{i=1}^{2} A_{ij} \leq 1 + \{0,1\} = 2 \\
& \iff \sum_{i=1}^{2} A_{ij} - 1 \leq 0 \iff \sum_{i=1}^{2} A_{ij} - 1 \leq 0
\end{align*}
\]

\#Attributes - 1

? .... which is ALWAYS true

all the combinations off att-, class- values

are valid ( = feasible solutions )

The idea ....

\[
R = (\#\text{Attributes} - 1 ) - 1 \Rightarrow \sum_{i=1}^{2} A_{ij} \leq (2 - 1) - 1 + 1 = 1 \iff \sum_{i=1}^{2} A_{ij} - 1 \leq 0
\]

GENERALIZATION

Find feasible solutions with just ONE binary attribute variable equals to 1

Find 1-IF place rules

... then proceed to 1-, 2-, ... \#Atts-IF place rules by increasing \( R \) .... ITERATE

General-to-Specific strategy of learning

PSL, Cyprus, 21-24 July 1997 George Potamias ICS/FORTH
Basics for the Theoretical Foundation of LPI

Basic Definitions

Propositional Clause (example):

\[ E(L) = \text{the set of all attribute and class values of a set of propositional examples} \]

\[ C \leftarrow A1_1, A1_2, ..., Av_v1, ..., Am_1, Am_2, ..., Am_vm \]

\[ C, Ai_j \text{ literals}, \text{positive and negative occurrences of elements in } E(L) \]

Binary Variable:

\[ X \text{ in } \{0,1\} \]

Binary Variable Representation of \( E(L) \):

\[ \{ Xc, XAi_j / C, Ai_j \text{ elements of } E(L) \} \]

Constraint Version of Propositional Examples:

\[ \text{cons}(e) : Xc \geq \bigwedge_{i=1}^{m} \bigvee_{j=1}^{V} Ai_j \]

\[ \text{conse}(E) = \{ \text{cons}(e) / e \text{ in } E \} \]

Theorem: [Bell, Nerode, Ng, Subrahmanian, 1996] \textbf{LINEARize} non-linear constraints

\[ X, X1, X2, ..., Nn \text{ binary variables} \]

\[ \bigwedge_{i=1}^{n} Xi \leq X \implies \sum_{i=1}^{n} Xi - X \leq n-1 = S_cons \]

e.g.

\[ e: H = \text{high}, C = \text{black} \implies C = \text{dangerous} \quad / \quad Hh \text{ and } Cb \rightarrow Cd \]

\[ \text{cons}(e) : XHh + XCh - Cd \leq 1 \]
More Definitions

Binary Variable Assignment: \[ S : Xi \rightarrow \{0,1\} \]
\[ Xi, \text{binary variables} \]

Binary Variable Assignment \( S_{I(E)} \) corresponding to \( I(E) \)

\( I(E) \) a model of set of propositional examples \( E \) (mathematical logic sense)

\[ S_{I}(Xi) = \begin{cases} 
1 & \text{if } X \text{ in } I \\
0 & \text{otherwise} 
\end{cases} \]

for all binary variables’ representative \( Xi \) of \( E \)

Card-minimal model:

\( I(E) \) is a card-minimal model of \( E \) \iff for all other models of \( E \), \( I'(E) \) \[ \text{card}(I) < \text{card}(I') \]

Minimal-model:

\( I(E) \) is a minimal of \( E \) \iff there exists no model, \( I'(E) \), of \( E' \subset I(E) \)

R-minimal-model:

\( I(E) \) is an R-minimal-model of \( E \) \iff its cardinality is \( R \) and it is a minimal model of \( E \)
Theoretical Basis of LPI

**Theorem:** [Bell, Nerode, Ng, Subrahmanian, 1996]

\[ S_{l(E)} \text{ is a solution of } \text{cons}(E) \iff l(E) \text{ is a model of } E \]

**Theorem:** [Bell, Nerode, Ng, Subrahmanian, 1996]

Xi: binary variable representatives of attribute values of E, \( 1 \leq i \leq M \)

... class values NOT included

\[ S_{l(E)} \text{ is an optimal solution of } \text{cons}(E) \text{ that minimize } \sum_{i=1}^{M} X_i \]

\[ \iff l(E) \text{ is a card-minimal model of } E \]

**Proposition:**

Fixing the value of \( \sum_{i=1}^{M} X_i \) to \( R \), then all the solutions of \( \text{cons}(E) \) are \( R \)-minimal-models of E and vice versa.

Given a set of propositional examples E, we may use a linear - INTEGER-{0,1} constraint solver in order to device ALL the \( R \)-card-minimal models of E, where \( R \) in \{1,2, ... \#Atts\}. 

PSL, Cyprus, 21-24 July 1997

George Potamias  ICS/FORTH  06
Given a set of examples E:

Device

--- all binary variable representatives of attribute- and class values of E: $V^n$

--- constraints for MUTUAL_EXCLUSIVE ATTRIBUTE VALUES: $\text{cons_atts}(Ai)$

--- constraint for MUTUAL_EXCLUSIVE ATTRIBUTE VALUES: $\text{cons_class}$

Set $R = 1$

while $R <= \#Atts$ repeat

--- Device set: $S_{\text{cons}}(E,R) = \{ S_{\text{cons}}(e,R) / e \in E \}$

--- $S_{\text{cons}}(e,R) = \sum X_i - X_c <= R - 1$

--- binary variable representatives for attribute values of e

--- binary variable for class value of e

--- Label(V)

repeat

label(V)

$X_1 = 1$, $X_2 = 1$, ..., $X_R = 1$, $X_c = 1$ \& $X_i \rightarrow X_c$

ADD constraint: $X_1 + X_2 ... + X_R <= R - 1$

Set $R = R + 1$

endwhile

ALL RULES THAT FIT THE GIVEN SET OF EXAMPLES ARE INDUCED
LPI: Some Preliminary Results

All data sets from UCI Machine Learning benchmark domains repository

<table>
<thead>
<tr>
<th></th>
<th>HAYES-ROTH</th>
<th>LENSES</th>
<th>SHUTTLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Cases / 4</td>
<td>238/26</td>
<td>17/7</td>
<td>11/5</td>
</tr>
<tr>
<td>#Att / 15</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>#Att-Values / 3</td>
<td>15</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>#CLASSES / 3</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>LPI Fittness /Accuracy</td>
<td>100/100/21</td>
<td>100/100/10</td>
<td>100/100/9</td>
</tr>
<tr>
<td>ID3</td>
<td>84/93/19</td>
<td>94/29/3</td>
<td>93/93/1</td>
</tr>
<tr>
<td>C45</td>
<td>85/93/9</td>
<td>94/29/3</td>
<td>93/93/1</td>
</tr>
</tbody>
</table>

- Better performance with MORE RULES .... overfitting

- When data are characterized by STRONG-RULE(s) then LPI is able to uncover EXACTLY this (these) rule(s)

On all data of MONKS_1 domain: \#LPI\_rules = \$C4.5\_rules
LPI and the CLP scheme

- **Current implementation in the CLP(FD) framework**
  - FD: Finite Domain
  - classical unification (Robinson, ...) replaced by CS processes
    --> Prolog + Constraint Solver
  - Porting in: clp(FD), Bin-Prolog, ECLiPSe

- **CLP vs. {C, Fortran}-LP-constraint-solvers framework:**
  - Unified/Compact representation for data, constraints, rules, IO operations
  - Control on the constraints satisfaction process: backtracking
  - Special CLP predicates: atmost, minimize(OF cost,..), #(Min, ConsList, Max),
  - time-performance
    - Prolog vs C, Fortran....
    - Parallel CLP implementation: ECLiPSe
LPI: Is it hard?

COMPLEXITY

- General-case FiniteCSP (FCSP) is NP-complete

- Tractable classes of FCSP: [Mackworth & Freuder, 1993, AI journal]
  - representation tradeoffs complexity: Restrictions on the topology-graph
  - tree-structured constraint graphs:

    In CLP, unification follows the standard DFSearch, so variables (delayed) are ordered -> tree-structure of the search space

  - \( a: \) attributes
  - \( n: \) variables
  - \( d: \) #values for each variable
  - \( e: \) # constraints

1977: \( O(na^3) \) [Mackworth, 1977], [Mackworth & Freuder, 1991], AI journal

1985: \( O(na^2) \) [Dechter & Pearl, 1988, AI journal]


Most CLP systems follow a CHIP-like constraints’ arrangement and satisfaction processes (ECLiPSe)

\( O(ae^2d) \)
LPI: The major advantages

The expressiveness spectrum of Machine Learning approaches:

```
<table>
<thead>
<tr>
<th>fully propositional</th>
<th>LPI</th>
<th>fully relational</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ID3, C4.5, CART, CN2, ...)</td>
<td></td>
<td>(MOBAL, KBG, ILP (FOIL, ...), ...)</td>
</tr>
</tbody>
</table>
```

- **Impoves the DECLARATIVE BIAS**

  - **Disjunctive information:**
    \[ A \text{ or } B :: A + B \geq 1 \]

  - **Negative information:**
    \[ \text{not}(A) :: A \leq 0 \]

  - **Attribute dependencies:**
    \[ A \rightarrow B :: A - B \leq 0 \]

  - **Model-based learning** (e.g. MOBAL)
    
    rule model: \[ P1 \& P2 \& P3 \rightarrow Pc :: \text{SUM}(Ai) \geq 3 \]

**BENCHMARKING SYSTEM:**

alternative generalizations could be examined