1. Motivation

2. Syntax and Semantics of LPDS

3. Examples

4. Computation

5. Qualitative Decision Making Using LPDS

6. Conclusions

Tommi Syrjänen, Helsinki University of Technology
Ilkka Niemelä, Helsinki University of Technology
Gerhard Brewka, University of Leipzig

With Ordered Distinction
Implementing Logic Programs
JELIA-07: Implementing LP$_R$ with Nienhuis and Strijen

AAAI-07: Logic programs with ordered disjunction (LP$_R$)

KR-07: Qualitative choice logic, prop. logic with ordered

blue, if this also doesn’t work then green.

Intuitive reading: X should be red, if this doesn’t work then

(\text{col}(X, \text{red}) \land \text{col}(X, \text{blue}) \land \text{col}(X, \text{green}) \land \text{col}(X, \text{red}) \times \text{col}(X, \text{blue}) \times \text{col}(X, \text{green}) \times \text{col}(X, \text{red}) \times \text{col}(X, \text{blue}) \times \text{col}(X, \text{green}))

... alternatives, options, choices, actions

Ordered disjunction to represent preferences among

Disjunction to represent alternatives, options, choices, ...
Use degrees to define preference relation on answer sets.

Answer sets satisfy rules to different degrees.

Define answer sets for LPD's.

\[ \text{if this is not possible } \mathcal{C}_1, \text{ but if this is not possible then } \mathcal{C}_2, \text{ if this is not possible } \mathcal{C}_3, \text{ etc.} \]

meaning: if body satisfied then some } \mathcal{C}_j \text{ must be true, preferably }

where the } \mathcal{C}_j \text{ and } \mathcal{B}_j \text{ are ground literals. }

\[ \mathcal{C}_1 \times \cdots \times \mathcal{C}_n \rightarrow \mathcal{A}_1', \cdots, \mathcal{A}_m', \not\mathcal{B}_1', \cdots, \not\mathcal{B}_l' \]

An LPD is a finite set of rules of the form:

Syntax of LPD's
But: split programs provide hint how to do it!

Thus: standard approaches to disjunctive LPS inadequate.

- Possible models reasonable

  Examples:

  \[
  \forall \exists B \times \mathcal{C} \times B \times \forall \exists \mathcal{C} \times B
  \]

  Answer sets of TPODs: using DLP semantics?
consistent answer set of a split program $P$ of $P$. 

Definition 3. A set of intervals $A$ is an answer set of $P$ if it is a set of intervals $A$ by exactly one of its options.

Definition 2. $P'$ is a split program of $P$ if it is obtained from $P$ by replacing each rule in $P$ by exactly one of its options.

Definition 1. Let $r = C_1 \times \cdots \times C_n \rightarrow \text{body}$ be a rule.

Answer sets of LPDs
\begin{align*}
\neg \neg P, \neg D & \rightarrow C \\
\neg \neg C, \neg \neg A & \rightarrow B \\
\neg \neg B & \rightarrow C \\
\neg C, \neg A & \rightarrow \neg B \\
A & \rightarrow B \\
A & \rightarrow \neg C \\
\neg A & \rightarrow B \times \neg C
\end{align*}

We obtain 4 split programs:

\begin{itemize}
\item[(1)] \( \neg A \rightarrow C \times B \)
\item[(2)] \( A \rightarrow C \times \neg B \)
\end{itemize}

Let \( P \) consist of the rules

Example
\[
d_{\lambda} < (\lambda)^{\varepsilon} d_{\lambda} < (\lambda)^{\varepsilon} d_{\lambda} < (\lambda)^{\varepsilon} d_{\lambda}
\]

for some \( \lambda > \ell \) and for no \( \lambda > \ell \) and \( (\lambda)^{\varepsilon} d_{\lambda} < (\lambda)^{\varepsilon} d_{\lambda} < (\lambda)^{\varepsilon} d_{\lambda} \)

\[
(d)_{\lambda}^{\varepsilon} S = (d)_{\lambda}^{\varepsilon} S
\]

for some \( \lambda > \ell \) and implies \( \lambda > \ell \)

\[
|d_{\lambda}^{\varepsilon} S| = |(d)_{\lambda}^{\varepsilon} S|
\]

for some \( \lambda > \ell \) and implies \( \lambda > \ell \)

\[
\{ \lambda = (\lambda)^{\varepsilon} d_{\lambda} < (\lambda) \} = (d)_{\lambda}^{\varepsilon} S
\]

\[
\{ \lambda \in \mathcal{C} \mid \lambda \in \mathcal{C} \}
\]

\[
\min \{ \lambda \mid \max \} = \ell
\]

and \( \lambda > \ell \), no \( \lambda > \ell \), all \( \lambda > \ell \)

\[
(\lambda \in \mathcal{C} \mid \lambda \in \mathcal{C} \}
\]

\[
(\lambda \in \mathcal{C} \mid \lambda \in \mathcal{C} \}
\]

\[
(\lambda \in \mathcal{C} \mid \lambda \in \mathcal{C} \}
\]

\[
C \times \cdots \times C
\]

Definition 4. \( S \) answer set, \( \ell \)

Preferences among answer sets.
as many nodes red in $W$ as in $\mathcal{W}$, and more blue in $W$.

\textit{4.} more nodes red in $W$, then in $\mathcal{W}$, or

\textit{5.} can\textit{d} there is no $\mathcal{W}$, such that

\textit{6.} a nodes blue in $\mathcal{W}$, super-set of nodes blue in $W$.

\textit{7.} and $\mathcal{W}$, and $W$, and

\textit{8.} same nodes red in $\mathcal{W}$, super-set of nodes red in $W$.

\textit{9.} and there is no $\mathcal{W}$, such that

\textit{10.} ind: there is no $\mathcal{W}$, such that at least 1 node has nicer and no

\textit{11.} par: there is no $\mathcal{W}$, such that

Model $W$ preferred if

\[ \lambda \neq X, (\lambda, X) \in \mathcal{E}, \text{edge}, (\lambda, \text{node}, C, \text{col}, \text{red}, X) \rightarrow (\lambda, \text{col}, X) \]

\[ \Rightarrow (\lambda, \text{col}, X, \text{node}, \text{green}, X) \rightarrow (\lambda, \text{col}, X, \text{red}) \times (\lambda, \text{col}, X, \text{blue}) \]

P tortured graph colouring
Creating a menu

Dessert
Coffee → tiramisu

Main

Starter

Beverages:

Needed components:

Soup → vegetarian

Red → alcohol

White → alcohol

Properties of components:

Vegetarian, espresso, cappuccino, red, white, and water.

Components: soup, salad, fish, beef, lasagne, ice—coffee,
determine menu which satisfies preferences as much as possible.

Drinks: alcohol or not, likes fish etc. (preferred answer sets

Given a description of the case at hand (visitor vegetarian or not,

<table>
<thead>
<tr>
<th>Preference</th>
<th>Menu, continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>not icecream, not coffee, dessert</td>
<td></td>
</tr>
<tr>
<td>icecream, not coffee</td>
<td></td>
</tr>
<tr>
<td>not icecream, dessert</td>
<td></td>
</tr>
<tr>
<td>espresso × cappuccino</td>
<td></td>
</tr>
<tr>
<td>white × red, water</td>
<td></td>
</tr>
<tr>
<td>not beef</td>
<td></td>
</tr>
<tr>
<td>red × white, water</td>
<td></td>
</tr>
<tr>
<td>beef</td>
<td></td>
</tr>
<tr>
<td>fish × beef, lasagne</td>
<td></td>
</tr>
<tr>
<td>main</td>
<td></td>
</tr>
<tr>
<td>soup × salad, starter</td>
<td></td>
</tr>
</tbody>
</table>
Modular representation of preference criteria.

not continue with one of tester's answer sets.

If tester fails $W$ is preferred, if tester tries to find better one.

Interleaved computation: generator produces answer set $W$'

$W \cup (d) (\forall \alpha ' \in W', P(\alpha '))$ such that $W$' answer set of $P$

$W \cup (d) (\forall \alpha ' \in W', P(\alpha '))$ such that $W$' answer set of $P$

$W \cup (d) (\forall \alpha ' \in W, P(\alpha '))$ such that $W$' answer set of $P$

Preferred answer sets can be computed on top of answer set $W$'.
(6) \( \{ d \in \mathcal{U} \mid (\mathcal{U}, d) \} \cap = (d) \mathcal{C} \)

The generator \((d) \mathcal{C}\) is:

\( (d) \mathcal{C} \cap \{ u \supset \gamma \mid (\gamma, u) \mathcal{C} \} \cap = \{ (u, \gamma) c, \cdots, (I, \gamma) c \} = (\gamma) \mathcal{C} \)

The translation \((\gamma) \mathcal{C}\) is:

\( \{ u \supset \gamma \supset I \mid (\gamma, u) c \rightarrow (\gamma, u) s \} \)

\( \cap \{ (u, \gamma) c, \cdots, (I, \gamma) c \} \rightarrow (\gamma, I) s \} = (\gamma) \mathcal{S} \)

The satisfaction translation \((\gamma) \mathcal{S}\) is:

\( \{ \text{body} \rightarrow \text{rule} \mid \gamma \rightarrow \text{rule}, \not \gamma \rightarrow \text{rule} \} = (\gamma) \mathcal{C} \)

The translation \((\gamma) \mathcal{C}\) of the \(k\)th option of \( r \) is denoted as:

\( \text{body} \rightarrow u \times \cdots \times \mathcal{C} \)

Let \( \mathcal{C}_r \) be an \( \mathcal{L}_{\mathcal{TPOD}} \). Then:

\( r = \mathcal{C}_1 \times \cdots \times \mathcal{C}_n \)
\[ L \cap \langle W, d \rangle \mathcal{C} = \langle W, d \rangle \mathcal{L} \]

The \( L \)-preference tester \((W, d) \mathcal{L}\) is defined as:

\[ \{d \not \rightarrow \} \cup \{s \not \rightarrow \} \]

\[ \{d \in \mathcal{L} \mid \exists \mathcal{L} \cup \{ (p, q) \} \subseteq \mathcal{L} \}
\]

\[ \mathcal{L} \subseteq \mathcal{L} \]

\[ \mathcal{L} \subseteq \mathcal{L} \]

\[ \mathcal{L} \subseteq \mathcal{L} \]

The tester preference independent part

Let \( p \) be an \( LpD\). Then, the core tester \((W, d) \mathcal{C}\) is defined as:

\[ \langle W, d \rangle \mathcal{C} = \langle W, d \rangle \mathcal{L} \]

\[ L \cap \langle W, d \rangle \mathcal{C} = \langle W, d \rangle \mathcal{L} \]
Figure 2: The inclusion preference tester $\mathcal{L}$

\[ (f) \not\in \text{better}'(f) \rightarrow (I)(\text{better} : (I)\text{worse}) \rightarrow \text{better} \]

\[ (f) \not\in \text{worse}'(f) \rightarrow (I)(\text{worse} : (I)\text{better}) \rightarrow \text{worse} \]

\[ (f) \not\in \text{better}'(f) \rightarrow (I)(\text{better} : (I)\text{worse}) \rightarrow \text{better} \]

\[ (f) \not\in \text{worse}'(f) \rightarrow (I)(\text{worse} : (I)\text{better}) \rightarrow \text{worse} \]

Figure 1: The Pareto-preference tester $\mathcal{L}$

\[ (f) \not\in \text{better}'(f) \rightarrow (I)(\text{better} : (I)\text{worse}) \rightarrow \text{better} \]

\[ (f) \not\in \text{worse}'(f) \rightarrow (I)(\text{worse} : (I)\text{better}) \rightarrow \text{worse} \]

\[ (f) \not\in \text{better}'(f) \rightarrow (I)(\text{better} : (I)\text{worse}) \rightarrow \text{better} \]

\[ (f) \not\in \text{worse}'(f) \rightarrow (I)(\text{worse} : (I)\text{better}) \rightarrow \text{worse} \]

The preference criteria
The preference criteria, etc.
in \( \forall \exists \).}

there exists a c-preferred answer set \( M \) such that \( M \in \mathcal{E} \).
4. Given an instance \( \exists \alpha \in \mathcal{E} \).

there exists a \( d \)-preferred answer set \( M \) such that \( M \in \mathcal{E} \).
3. Given an instance \( \exists \alpha \in \mathcal{E} \).

is \( k \)-preferred \( (\alpha \in \mathcal{E}) \) is \( \text{comp}-\text{complete}. \)
2. Let \( M \) be an answer set of \( \exists \alpha \in \mathcal{E} \).

\( \exists \alpha \in \mathcal{E} \).
1. Deciding whether \( \exists \alpha \in \mathcal{E} \).

Complexity.
1. Distinctly choose a set of decision literals, $C$ the agent can decide

2. Represent the different alternative decisions.

3. Represent alternative states of the world.

4. Represent relationships between and consequences of different alternatives.

5. Represent desired properties.

6. Use preference relation on answer sets to induce preference relation on possible decisions.

7. Pick one of the most preferred decisions.
possible decisions and states of the world:
\{ \text{in-omlette, in-cup, throw-away} \}
\text{C: set of intervals built from}

Savages' rotten egg example
\[ \text{Desires:} \]

Different omelettes mutually inconsistent (6 rules, omitted)

\[ \text{Effects of choices:} \]

Rotten eggs, et al.
The logic program has 6 answer sets:

$S^1 \{\text{o-melette, wash, fresh, in-omellette}\} = S^1$

$S^2 \{\text{o-melette, wash, fresh, in-omellette}\} = S^2$

$S^3 \{\text{o-melette, wash, fresh, in-cup}\} = S^4$

$S^5 \{\text{o-melette, wash, fresh, in-cup}\} = S^6$

$S^6 \{\text{o-melette, wash, fresh, in-omellette}\} = S^6$

$S^6 \{\text{o-melette, wash, fresh, in-omellette}\} = S^6$

Preference among answer sets:

Figure 1: Preference among answer sets.
Decision making strategies

- Cautious: compare answer sets statewise (here: ran or → run). C1

- Externally cautious: prefer decision C1 over C2 if least preferred answer set(s) containing C2 prefers to most preferred answer.

- Optimistic: in-omellette reasoning from preferred answer sets, assumes e&g will be fresh.

- Pessimistic: throw-away action with most tolerable worst outcome, assumes e&g rotten.

- Extremely cautious: throw-out preferred over C2 if for each state $L$ the least preferred answer set(s) containing C2 prefers to most preferred answer.
they contain: in—cup.

Decisions can be ordered based on average penalties of answer sets

\[
\begin{align*}
S_6 &: 5 \\
S_5 &: 5 \\
S_4 &: 6 \\
S_3 &: 5 \\
S_2 &: 5 \\
S_1 &: 0
\end{align*}
\]

otherwise, where \( f \) smallest integer such that \( f \in S' \).

0 if body not satisfied in \( S \) or \( c_1 \in S' \),

body \( \rightarrow (u(n) \cdots x(n)) x \cdots x (u(n)) y \).

Overall penalty for answer set \( S \): sum of all rule penalties

\[
\begin{align*}
& 6-\text{omelette} \times 5-\text{omelette} \times 0-\text{omelette} \times 5-\text{omelette} \\
&\text{wash—cup} \times \text{wash—cud} \times \text{wash—cup}
\end{align*}
\]
answer sets and strategies towards risk is necessary for general decision making problems considering all

- can be implemented on top of existing ASP provers
- certain conditions
- reasoning from most preferred answer sets adequate under
- interesting applications in different areas
- induces preference relation on answer sets
- preferences
- OD allows for convenient encoding of context dependent
- added OD to LPS

Conclusions