Learning Partial Lexicographic Preference Trees over Combinatorial Domains

Xudong Liu and Miroslaw Truszczynski

Department of Computer Science
University of Kentucky
Lexington, KY, USA

January 27, 2015

Motivation

Preferences over options are ubiquitous in real life.

1. People have preferences over dinners offered in a restaurant.
2. In political elections voters cast ballots over candidates.
3. Customers rate their purchased items on websites such as Bestbuy.com and Amazon.com.

 $\succ$  $\succ$  $\succ \cdots$

Figure: What do I like for dinner?

Motivation

 $\succ$  $\succ$  $\succ \cdots$

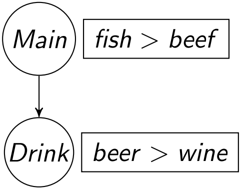
1. Preferences are **qualitative**.
2. Domains are **combinatorial**.
3. Can we model such preferences **compactly**?
4. If so, can we learn a compact model **efficiently**?

Motivation

1. What is the problem?
   – Qualitative preference modeling and learning over combinatorial domains.
2. Why is it important?
   - Predict preferences between unseen options.
   - Support preference reasoning tasks, such as dominance testing, optimization and aggregation.

Contributions

1. Propose *partial lexicographic preference trees* (PLP-trees):
   – lexicographic, compact, extending LP-trees[1]

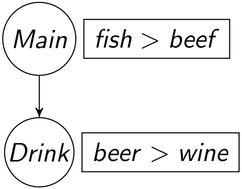


 $\succ$  $\succ$ 

[1] Booth et al., *Learning Conditionally Lexicographic Preference Relations*, 2010.

Contributions

1. Propose *partial lexicographic preference trees* (PLP-trees):
   – lexicographic, compact, extending LP-trees[1]



UI-UP PLP-tree

 $\succ$  $\succ$ 

[1] Booth et al., *Learning Conditionally Lexicographic Preference Relations*, 2010.

Contributions

1. Propose *partial lexicographic preference trees* (PLP-trees):
   – lexicographic, compact, extending LP-trees
2. Show results on *passive learning problems*, where all training examples ($o_i \succ o_i'$ or $o_k \approx o_k'$) are provided upfront, in the setting of PLP-trees:
   – ConsLearn, SmallLearn, MaxLearn

CONSLEARN UI-UP PLP-trees

1. Let $\mathcal{I} = \{X_1, \ldots, X_p\}$ be a set of binary issues, and $CD(\mathcal{I})$ be the combinatorial domain $\prod_{1 \leq i \leq p} \mathcal{D}_i$, where $\mathcal{D}_i = \{0_i, 1_i\}$. Given an example set $\mathcal{E} = \{e_1, \ldots, e_m\}$, where e_i is of form either $\alpha_i \succ \beta_i$ or $\alpha_i \approx \beta_i$ for $\alpha_i, \beta_i \in CD(\mathcal{I})$, the CONSLEARN problem is to learn a PLP-tree T (of a particular type) consistent with $\mathcal{E}$.
2. Denote by $\mathcal{E}^{\succ} = \{e_i \in \mathcal{E} : \alpha_i \succ \beta_i\}$ the set of *strict examples*, and $\mathcal{E}^{\approx} = \{e_i \in \mathcal{E} : \alpha_i \approx \beta_i\}$ the set of *equivalent examples*.
3. Denote by (X_i, x_i) a node label in a UI-UP PLP-tree, where X_i is an issue from $\mathcal{I}$ and x_i is the preferred value, either 0_i or 1_i, in $\mathcal{D}_i$.

4. $NEQ(\mathcal{E},\mathcal{I})$: set of issues in $\mathcal{I}$ that incorrectly handle at least one equivalent example in $\mathcal{E}$, i.e.,

$$NEQ(\mathcal{E},\mathcal{I}) = \{X_i \in \mathcal{I} : \exists \alpha \approx \beta \in \mathcal{E}, \alpha(X_i) \neq \beta(X_i)\}.$$

5. $EQ(\mathcal{E},\mathcal{I})$: set of issues in $\mathcal{I}$ that do not order any of the strict examples in $\mathcal{E}$, i.e.,

$$EQ(\mathcal{E},\mathcal{I}) = \{X_i \in \mathcal{I} : \forall \alpha \succ \beta \in \mathcal{E}, \alpha(X_i) = \beta(X_i)\}.$$

6. $AVI(\mathcal{E}, S)$: set of issues in S ($S \subseteq \mathcal{I} \setminus (NEQ(\mathcal{E},\mathcal{I}) \cup EQ(\mathcal{E},\mathcal{I}))$) available for selection as the next important issue, i.e.,

$$AVI(\mathcal{E}, S) = \{X_i \in S : \forall \alpha \succ \beta \in \mathcal{E}, \alpha(X_i) \geq \beta(X_i) \vee \\ \forall \alpha \succ \beta \in \mathcal{E}, \alpha(X_i) \leq \beta(X_i)\}.$$

ConsLearn UI-UP PLP-trees (cont'd)

1. $1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$
2. $0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$
5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

$NEQ = \{X_4\}, EQ = \{X_5\},$

$S = \{X_1, X_2, X_3\}$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E}, \mathcal{I}) \cup EQ(\mathcal{E}, \mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\mathcal{E}^{\succ} \neq \emptyset$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $AVI(\mathcal{E}, S) = \emptyset$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e \text{ is decided on } X_\ell\}$;
  $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

ConsLearn UI-UP PLP-trees (cont'd)

1. ~~$1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~
2. ~~$0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$
5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

1st: $S = \{X_1, X_2, X_3\}$,
$AVI = \{X_1, X_3\}, X_\ell = X_3$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist
$S = \mathcal{I} \setminus (NEQ(\mathcal{E}, \mathcal{I}) \cup EQ(\mathcal{E}, \mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\underline{\mathcal{E}^{\succ} \neq \emptyset}$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $\underline{AVI(\mathcal{E}, S) = \emptyset}$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e$ is decided on $X_\ell\}$;
  $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

1. $\cancel{1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5}$
2. $\cancel{0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5}$
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$
5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

1st: $S = \{X_1, X_2, X_3\}$,
$AVI = \{X_1, X_3\}, X_\ell = X_3$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist
$S = \mathcal{I} \setminus (NEQ(\mathcal{E}, \mathcal{I}) \cup EQ(\mathcal{E}, \mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\mathcal{E}^{\succ} \neq \emptyset$ **do**
    Construct $AVI(\mathcal{E}, S)$;
    if $AVI(\mathcal{E}, S) = \emptyset$ **then**
        return FAILURE;
    end
    $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
    Construct (X_ℓ, x_ℓ);
    $T \leftarrow T, (X_\ell, x_\ell)$;
    $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e$ is decided on $X_\ell\}$;
    $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

1. $\cancel{1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5}$
2. $\cancel{0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5}$
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$
5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

1st: $S = \{X_1, X_2, X_3\}$,
$AVI = \{X_1, X_3\}, X_\ell = X_3$

$(X_3, 0_3)$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

```
S = I \ (NEQ(E, I) ∪ EQ(E, I));
T ← empty sequence;
while E^≻ ≠ ∅ do
    Construct AVI(E, S);
    if AVI(E, S) = ∅ then
        return FAILURE;
    end
    X_ℓ ← an element from AVI(E, S);
    Construct (X_ℓ, x_ℓ);
    T ← T, (X_ℓ, x_ℓ);
    E ← E\{e ∈ E^≻ : e is decided on X_ℓ};
    S ← S\{X_ℓ};
end
return T;
```

ConsLearn UI-UP PLP-trees (cont'd)

1. ~~$1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~
2. ~~$0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. ~~$1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~
5. ~~$0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

1st: $S = \{X_1, X_2, X_3\}$,
$AVI = \{X_1, X_3\}, X_\ell = X_3$

$(X_3, 0_3)$

$S = \{X_1, X_2\}$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E}, \mathcal{I}) \cup EQ(\mathcal{E}, \mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\mathcal{E}^{\succ} \neq \emptyset$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $AVI(\mathcal{E}, S) = \emptyset$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \backslash \{e \in \mathcal{E}^{\succ}$: e is decided on $X_\ell\}$;
  $S \leftarrow S \backslash \{X_\ell\}$;
end
return T;

1. ~~$1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~
2. ~~$0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~
3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$
4. ~~$1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~
5. ~~$0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

2nd: $S = \{X_1, X_2\}$,
$AVI = \{X_1\}, X_\ell = X_1$

$(X_3, 0_3)$
$\downarrow$
$(X_1, 1_1)$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E},\mathcal{I}) \cup EQ(\mathcal{E},\mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\underline{\mathcal{E}^{\succ} \neq \emptyset}$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $\underline{AVI(\mathcal{E}, S) = \emptyset}$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e \text{ is decided on } X_\ell\}$;
  $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

ConsLearn UI-UP PLP-trees (cont'd)

1. ~~$1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~
2. ~~$0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~
3. ~~$1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$~~
4. ~~$1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~
5. ~~$0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~
6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

2nd: $S = \{X_1, X_2\}$,

$AVI = \{X_1\}, X_\ell = X_1$

$$\begin{array}{c} (X_3, 0_3) \\ \downarrow \\ (X_1, 1_1) \end{array}$$

$S = \{X_2\}$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$

Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E},\mathcal{I}) \cup EQ(\mathcal{E},\mathcal{I}))$;

$T \leftarrow$ empty sequence;

while $\mathcal{E}^{\succ} \neq \emptyset$ **do**

  Construct $AVI(\mathcal{E}, S)$;

  if $AVI(\mathcal{E}, S) = \emptyset$ **then**

    return FAILURE;

  end

  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;

  Construct (X_ℓ, x_ℓ);

  $T \leftarrow T, (X_\ell, x_\ell)$;

  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e \text{ is decided on } X_\ell\}$;

  $S \leftarrow S \setminus \{X_\ell\}$;

end

return T;

ConsLearn UI-UP PLP-trees (cont'd)

~~1. $1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~

~~2. $0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~

~~3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$~~

~~4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~

~~5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~

6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$

3rd: $S = \{X_2\}$,
$AVI = \{X_2\}, X_\ell = X_2$

$$(X_3, 0_3)$$
$$\downarrow$$
$$(X_1, 1_1)$$
$$\downarrow$$
$$(X_2, 1_2)$$

```
Input: A set 𝓔 of examples over 𝓘
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a
        UI-UP tree does not exist
S = 𝓘 \ (NEQ(𝓔, 𝓘) ∪ EQ(𝓔, 𝓘));
T ← empty sequence;
while 𝓔^≻ ≠ ∅ do
    Construct AVI(𝓔, S);
    if AVI(𝓔, S) = ∅ then
        return FAILURE;
    end
    X_ℓ ← an element from AVI(𝓔, S);
    Construct (X_ℓ, x_ℓ);
    T ← T, (X_ℓ, x_ℓ);
    𝓔 ← 𝓔\{e ∈ 𝓔^≻ : e is decided on X_ℓ};
    S ← S\{X_ℓ};
end
return T;
```

~~1. $1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~

~~2. $0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~

~~3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$~~

~~4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~

~~5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~

~~6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$~~

3rd: $S = \{X_2\}$,

$AVI = \{X_2\}, X_\ell = X_2$

$$(X_3, 0_3)$$
$$\downarrow$$
$$(X_1, 1_1)$$
$$\downarrow$$
$$(X_2, 1_2)$$

$S = \emptyset$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$

Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E},\mathcal{I}) \cup EQ(\mathcal{E},\mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\underline{\mathcal{E}^{\succ} \neq \emptyset}$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $\underline{AVI(\mathcal{E}, S) = \emptyset}$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e \text{ is decided on } X_\ell\}$;
  $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

ConsLearn UI-UP PLP-trees (cont'd)

~~1. $1_1 1_2 1_3 0_4 0_5 \approx 1_1 1_2 1_3 1_4 0_5$~~

~~2. $0_1 0_2 1_3 1_4 1_5 \approx 0_1 0_2 1_3 0_4 1_5$~~

~~3. $1_1 0_2 0_3 0_4 1_5 \succ 0_1 1_2 0_3 1_4 1_5$~~

~~4. $1_1 1_2 0_3 1_4 0_5 \succ 1_1 1_2 1_3 0_4 0_5$~~

~~5. $0_1 1_2 0_3 0_4 1_5 \succ 0_1 0_2 1_3 0_4 1_5$~~

~~6. $0_1 1_2 1_3 1_4 0_5 \succ 0_1 0_2 1_3 1_4 0_5$~~

$\mathcal{E}^{\succ} = \emptyset$,

Done!

$(X_3, 0_3)$

$\downarrow$

$(X_1, 1_1)$

$\downarrow$

$(X_2, 1_2)$

Input: A set $\mathcal{E}$ of examples over $\mathcal{I}$
Output: A sequence T of pairs (X_ℓ, x_ℓ), or FAILURE if a UI-UP tree does not exist

$S = \mathcal{I} \setminus (NEQ(\mathcal{E}, \mathcal{I}) \cup EQ(\mathcal{E}, \mathcal{I}))$;
$T \leftarrow$ empty sequence;
while $\underline{\mathcal{E}^{\succ} \neq \emptyset}$ **do**
  Construct $AVI(\mathcal{E}, S)$;
  if $\underline{AVI(\mathcal{E}, S) = \emptyset}$ **then**
    return FAILURE;
  end
  $X_\ell \leftarrow$ an element from $AVI(\mathcal{E}, S)$;
  Construct (X_ℓ, x_ℓ);
  $T \leftarrow T, (X_\ell, x_\ell)$;
  $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \in \mathcal{E}^{\succ} : e \text{ is decided on } X_\ell\}$;
  $S \leftarrow S \setminus \{X_\ell\}$;
end
return T;

Results

	FP	UP	CP
UI	P	P	*NP*
CI	P	NPC[2]	P

(a) ConsLearn

	FP	UP	CP
UI	NPC	NPC	NPC
CI	NPC	NPC	NPC

(b) SmallLearn

	FP	UP	CP
UI	NPC[3]	NPC	NPC
CI	NPC	NPC	NPC

(c) MaxLearn

Figure: Complexity results for passive learning problems

[2] Booth et al., *Learning Conditionally Lexicographic Preference Relations*, 2010.

[3] Schmitt and Martignon, *On the Complexity of Learning Lexicographic Strategies*, 2006.

Ongoing and Future Research

1. Settle the complexity for the ConsLearn problem for UI-CP.
2. Implement algorithms handling issues of multi-valued domains.
3. Compare PLP-tree empirically[4] with other models.

[4] Available datasets: UCI machine learning repository, preference-learning.org, and PrefLib.

Q&A

Thank you!

Related Work: Qual. Pref. Models

1. Graphical models: ceteris paribus[5] models (CP-nets[6], TCP-nets[7]), lexicographic models (lexicographic strategies[8], LP-trees[9], CLP-trees[10]).
2. Non-graphical models: ASO-theories[11], Possibilistic logic[12], CP-theories[13].

[5] Latin, it means "everything else being equal."

[6] Boutilier et al., *CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements*, 2004.

[7] Brafman and Domshlak, *Introducing Variable Importance Tradeoffs into CP-nets*, 2002.

[8] Schmitt and Martignon, *On the Complexity of Learning Lexicographic Strategies*, 2006.

[9] Booth et al., *Learning Conditionally Lexicographic Preference Relations*, 2010.

[10] Bräuning and Eyke, *Learning Conditional Lexicographic Preference Trees*, 2012

[11] Brewka, Niemelä and Truszczynski, *Answer Set Optimization*, 2003.

[12] Dubois, Lang and Prade, *A brief overview of possibilistic logic*, 1991.

[13] Wilson, *Extending CP-Nets with Stronger Conditional Preference Statements*, 2004.