Interactive Exploration of Multi-Dimensional Information Spaces with Preference Support

PhD Presentation

Panagiotis Papadakos
papadako@csd.uoc.gr

Supervisor: Yannis Tzitzikas

Computer Science Department
University of Crete, Greece
ICS-FORTH, Greece

Heraklion, 25 November 2013
This thesis proposes a novel interaction model

- for searching and exploring gradually information
- enriched with a language for expressing preferences
- over hierarchically organized and multi-valued attributes
- that is user friendly

Hippalus

- Implementation of Hippalus, a system that supports the aforementioned framework

The conducted user studies, show that this interaction model helps the users

- explore the information space
- easily express their preferences over the available choices in a gradual manner
- locate more easily/fast the desired information
Most Search Systems (eg. Web Search Engines) return a **ranked-list of results**

- Users have to explore the answer **linearly**
- In practice users tend to look **only** at the first page of results
  - Miss useful hits
  - Difficult to find **useful hits** if the query contains ambiguous words
  - Difficult to *guess the right words for expanding* the query

---

**Traditional Searching**

[Diagram showing a cycle of query, ranked set of hits, and linear presentation with a checkmark indicating user satisfaction or dissatisfaction]
Motivation

- Almost 60% of search tasks are of **exploratory** nature (Crawford, 2006)
  - As a result the user has to inspect more than one resources (hits) to be satisfied → which is typical in decision making (e.g. buy a car)
  - Exploratory tasks are **time consuming** and **session-based**
- Users rarely exploit the **available metadata**
  - e.g. Google Advanced Search or specific keywords (i.e. filetype:pdf)

Then narrow your results by...

- **language**: any language
- **region**: any region
- **last update**: anytime
- **site or domain**:
- **terms appearing**: anywhere in the page
- **SafeSearch**: Show most relevant results
- **reading level**: no reading level displayed
- **file type**: any format
- **usage rights**: not filtered by license
## Objective and Requirements

### General objective of thesis

- offer users a **flexible** and **effective** method for accessing **large amounts of data**
- supporting **recall-oriented** information needs and **decision making** (i.e. preferences)

### Requirements meeting the objective

- **Generality**
  - Capture a wide range of information spaces and user information needs
- **Expressiveness**
  - Interactively specify complex preference structures
- **Usability**
  - Users should be able to use and understand the interaction immediately
Faceted and Dynamic Taxonomies (FDT)

- is a **widely** used interaction scheme for **Exploratory Search**
- can be used over **structured** and **unstructured** information spaces
- supports **static** and **dynamically** mined metadata
- display data under different categorization schemes (**facets**) and categories (**terms**)
We propose the enrichment of the FDT with preference actions that:

- affect the ordering of **facets**, **terms**, **objects** (of the focus)
- support **hierarchically organized data**, automatically resolving conflicts
- support **multi-valued** attributes (commonly found in tags, etc.)
- offer composition of preference actions

**Exploratory Preference Based Searching**

*With Static and Dynamic Metadata*
Questions

1. How can we \textit{gradually} and \textit{flexibly} specify \textit{preferences} over information spaces that might be \textit{hierarchically} organized, resolving when possible \textit{conflicts}, and might support \textit{multi-valued} attributes, and which will be their \textit{semantics}?

2. How can we tackle the \textit{algorithmic} perspective so that the proposed interaction can be applied over \textit{large information bases}?

3. How does the proposed interaction affect the \textit{user effort} and other \textit{metrics} during exploratory tasks?
Outline of the Presentation

BG & RW

Proposed Preference Framework

Hippalus

Evaluation

Conclusion & FW
BG & RW

- Faceted and Dynamic Taxonomies
- Preferences in General
- Preferences in Databases
- Personalization in FDT
Faceted and Dynamic Taxonomies (FDT) is currently a widely used interaction scheme that provides exploration services (Sacco and Tzitzikas, 2009)

- They are used to **explore** and **browse** complex information bases in a **guided** and **gradual**, yet **unconstrained** way through a visual interface
- display of **current results** in multiple categorization schemes (**facets**)
- display categories (**terms**) leading to **non-empty results**
- display of the **count** of the indexed objects of each category
- the **focus** of the user can be extensionally defined (results of a query) or intentionally (a set of terms selected by the user)
Preferences in General

- FDT can be considered as hard constraints
- ... Preferences are not hard constraints, but wishes

General Limitations

- Users do not have stable preferences over time
- Preferences change according to available choices
- Without knowing the available choices expression of preferences
  - difficult and time-consuming
  - results to incomplete preferences
Preferences in Databases

- Preferences have been studied thoroughly in the DB world (Stefanidis et al., 2011)

Limitations

- Users have to formulate **complicated queries** or use **complex UIs**
- They must be **acquainted with the information space** and **available choices**
- Preferences are given in **one shot** (not **gradually**)
- Users cannot exploit **hierarchically organized** attribute values
- They only support **single-valued** attributes, (i.e. no **multi-valued** attributes)
Personalization in FDT

- Facets and terms ordered lexicographically or based on # indexed objects
- Personalization services over FDT have started to come up
- Automatically present the most preferred facets and terms based on:
  - set cover ranking (Dakka et al., 2005)
  - their "interestingness" (Dash et al., 2008)
  - collaborative filtering (Koren et al., 2008; Tvarožek et al., 2008)
  - less effort approaches (Roy et al., 2008; Kashyap et al., 2010)
  - semantically enriching tweets (Abel et al., 2011)
  - intuitions (small and uniform operations) (Wagner et al., 2011)
  - logs for modeling user behaviour (Pound et al., 2011)

Limitations

- The user cannot explicitly and gradually rank the facets, terms or objects
- Such approaches do not exploit inherent properties of FDTs like hierarchies
Proposed Preference Framework
We consider data forming a **table**, with **single-valued (SV)** and **multi-valued (MV)** attributes, where attribute values can be organized hierarchically in **taxonomies**.

In the context of the unstructured world, attributes can be **extracted from**:

- **static metadata** (e.g. filetype, date, etc.)
- **dynamic metadata** (e.g. clustering, entity mining, etc.) Papadakos et al. (2012a)

### Running Example

- Car(Id, Manuf., Model, Category, Price, Power, Volume, Year, Fuel, Location, Access., Comment)
- Hierarchically organized taxonomies like: Manufacturer, Fuel and Location
- Attribute Accessories is multi-valued

<table>
<thead>
<tr>
<th>Id</th>
<th>Manufacturer</th>
<th>Model</th>
<th>Category</th>
<th>Price</th>
<th>Power</th>
<th>Volume</th>
<th>Year</th>
<th>Fuel</th>
<th>Location</th>
<th>Accessories</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Porsche</td>
<td>Carrera 911</td>
<td>Cabrio</td>
<td>50000</td>
<td>350</td>
<td>3600</td>
<td>2005</td>
<td>Petrol</td>
<td>Cefalonia</td>
<td>{ABS, AT}</td>
<td>&quot;Uncrashed, like new&quot;</td>
</tr>
</tbody>
</table>

**MANUFACTURER**

- European
  - German
    - BMW
    - Audi
    - Porsche
  - Italian
    - Fiat
    - Lamborghini
    - Ferrari
  - French
- American
  - U.S.A.
    - Chrysler
  - Japanese
    - Toyota
    - Lexus
- Asian
  - Korean
    - Kia

References

Papadakos et al. (2012a)
Language for Expressing Preferences

Requirements:

- Easily enacted through a GUI
- Affecting facets, terms and objects
- Expressive enough, to support **a number of preference actions** (i.e. relative preferences, composition)
Language: BNF Language Syntax

\[
\langle \text{stmt} \rangle ::= \langle \text{scopeType} \rangle \langle \text{spec} \rangle \\
\langle \text{scopeType} \rangle ::= \text{facets order} : | \text{terms order} : | \text{object order} : \\
\langle \text{spec} \rangle ::= \langle \text{anchor} \rangle \langle \text{rankSpec} \rangle \\
\langle \text{anchor} \rangle ::= \text{facet} \langle F_i \rangle \\
| \quad \text{term} \langle t_j \rangle \\
| \quad \text{object} \langle o_k \rangle \\
| \quad \epsilon \quad \text{// the empty string} \\
\langle \text{rankSpec} \rangle ::= \{ \text{lexicographic} | \text{count} | \text{value} | \text{indexedBy} \} \{ \text{min} | \text{max} \} \\
| \quad \text{best} | \text{worst} \\
| \quad \text{use scoreFunction} \langle \text{score()} \rangle \{ \text{min} | \text{max} \}
\]

Examples

- facets order: facet Manufacturer best
- object order: term Location.Cefalonia best
- object order: facet Relevance value max
Language: Extending the language...

- To support *relative preferences* over facets and terms

  \( \langle \textit{stmt} \rangle \ | \ \text{facets order: prefer} \ \langle F_i \rangle \text{ to } \langle F_j \rangle \)

  \( \langle \textit{stmt} \rangle \ | \ \text{terms order: prefer} \ \langle t_i \rangle \text{ to } \langle t_j \rangle \)

  \( \langle \textit{stmt} \rangle \ | \ \text{object order: prefer} \ \langle t_i \rangle \text{ to } \langle t_j \rangle \)

- Extending syntax to allow *complex expressions*, synthesizing two or more actions
  e.g. *Pareto, prioritized synthesis*

  \( \langle \textit{stmt} \rangle \ | \ \text{objects order: Pareto} \ \langle \text{setOfFacets} \rangle \)

  \( \langle \textit{stmt} \rangle \ | \ \text{objects order: ParetoOptimal} \ \langle \text{setOfFacets} \rangle \)

  \( \langle \textit{stmt} \rangle \ | \ \text{objects order: Priority} \ \langle \text{orderedSetOfFacets} \rangle \)

  \( \langle \textit{stmt} \rangle \ | \ \text{objects order: Combinational} \ \langle \text{bucketOrderedSetOfFacets} \rangle \)
Preference Semantics

- Let $B$ be the set of preference statements the user has given.
- Then $B$ can be partitioned to $k + 2$ subsets ($k$ is the number of facets)
  - $B_F$ holds the user actions for facets
  - $B_{T_i}$ holds the user actions for the terms of each facet $F_i$ ($k$ such subsets)
  - $B_{Obj}$ holds the user actions regarding the objects preferences
  - So $B = B_F \cup (\bigcup_i B_{T_i}) \cup B_{Obj}$

- A preference over a set of elements $E$ can be expressed as a **binary relation** $(E, \succ)$ over $E$.
- Independently of the number of actions we have $k + 2$ preference relations
  - 1 over the facets: $\left(\{F_1, \ldots, F_k\}, \prec_F\right)$
  - $k$ preferences relations defined over the terms $T_i$ of each facet $F_i$ (having the form $(T_i, \prec_i)$)
  - 1 preference relation for the objects $(A, \prec_{Obj})$

**Definition (Valid Preference)**

A preference relation $R$ over a set of elements $E$ is valid, if it is acyclic.

**Definition ($L$ Respects $R$)**

We say that a linear or bucket order $L$ over $E$ respects a preference relation $R$, if $R \subseteq L$. 
Cases

- Flat Single Valued (SV)
- Hierarchical non-Relative Single Valued
- Hierarchical Relative Single Valued
- Flat Multi Valued (MV)
- Hierarchical Relative Multi Valued
Case Flat SV: Input & Output

- **Input:**
  - Let $E = \{e_1, \ldots, e_n\}$ be a set of elements
  - Let $B$ be the elements of $E$ on which a **best** action has been defined
  - Let $W$ be the elements of $E$ on which a **worst** action has been defined
  - Let $R_\succ$ be binary relation holding the **relative** preferences (of the form $e_i \succ e_j$)
  - Let $Policy$ determine the ordering of **inactive elements** (e.g. minimal, maximal)

- **Output:**
  - A bucket order $L$ of $E$
Case Flat SV: Algorithm

**Algorithm 1** \(\text{Apply}(E, B, W, R_{\succ}, Policy)\)

**Input:** the set of elements \(E\), the set of best elements \(B\), the set of worst elements \(W\), a set of relative relationships \(R_{\succ}\), and \(Policy\) for inactive elements

**Output:** a bucket order over \(E\) that respects \(R\)

1: \(R_{bw} \leftarrow \{(b, w) \mid b \in B, w \in W\}\) // each best is preferred than each worst
2: \(R \leftarrow R_{bw} \cup R_{\succ}\) //add relative prefs
3: \(L \leftarrow \text{SourceRemoval}(R)\) //produce blocks with boundaries
4: \(I \leftarrow E \setminus (B \cup W \cup \text{dom}(R_{\succ}))\) // \(I\) contains inactive elements
5: \(L' \leftarrow \text{addInactiveElements}(L, I, Policy)\)
6: return \(L'\)

**Algorithm 2** \(\text{SourceRemoval}(R)\), a topological sorting algorithm

**Input:** a binary relation \(R\) over \(E\)

**Output:** a bucket order over \(E\)

1: \(L \leftarrow \langle \rangle\)
2: repeat
3: \(S \leftarrow \text{maximal}_{\succ}(R)\)
4: \(R \leftarrow R \setminus \{(x, y) \in R \mid x \in S\}\) // Remove maximal
5: \(L \leftarrow L.\text{append}(S)\) // Append a bucket to \(L\)
6: until \(S \neq \emptyset\)
7: return \(L\)
Case Flat SV: Example

- let $B = \{Ferrari\}$ and $W = \{Fiat, Lancia\}$
- let $R_\succ = \{Porsche \succ Ferrari, Porsche \succ Fiat\}$

After running topological sorting, $L = \langle Porsche, Ferrari, \{Fiat, Lancia\}\rangle$
Exploit Hierarchies

Question

- What if a user wants to say that all German manufactures are considered best, while all Italian as worst?

- Express a preference action **for each** available German and Italian manufacturer
  - Best(BMW)
  - Best(Mercedes)
  - Worst(Fiat)
  - Worst(Lancia)
  - etc.

- FDT **inherently** support hierarchies
- **Exploit hierarchically** organized taxonomies
  - through **preference inheritance**
  - **simplify user interaction** (less number of preference actions)
Case Hierarchical SV Non-Relative: Scope & Inheritance

- Let $b$ be an action in $B$, either \textbf{best} or \textbf{worst}
- We denote with $\text{scope}(b)$ the \textit{scope} of the action $b$ (i.e. either the facets, terms, or objects that it affects)
- To capture \textit{inheritance} we redefine the scope of inherited actions, as:

\textbf{Definition (Scope and Inheritance)}

Let $b$ be an inherited action $b = \langle e, rs \rangle$ where $e$ is its anchor and $rs$ the other part of the action. The scope of $b$ is defined as:

$$\text{scope}(b) = \bigcup_{e' \in N^*(e)} \text{scope}(\langle e', rs \rangle)$$

where $N^*(e)$ stands for $e$ and the narrower elements of $e$, formally

$N^*(e) = \{e\} \cup N^+(e) = \{e' \mid e' \leq e\}$. 


Case Hierarchical SV Non-Relative: Dominance Rule

- Unfolding each $b \in B$ based on its scope might lead to cycles (i.e. non valid preferences) e.g. Worst(Italian), Best(Ferrari)
- Automatically resolve such conflicts through action refining
- Define a preorder relation over $B$, denoted by $(B, \sqsubseteq)$ according to definition:

**Definition (Refinement Checking)**

We say that an action $b$ is equally or more refined than an action $b'$, denoted by $b \sqsubseteq b'$, if $\text{scope}(b) \subseteq \text{scope}(b')$.

- We can use $(B, \sqsubseteq)$ to resolve conflicts due to inheritance, based on the following rule:

**Definition (Scope-based Dominance Rule)**

If $A \subseteq \text{scope}(b) \subseteq \text{scope}(b')$ then $b'$ is dominated by $b$ on $A$, thus action $b$ should determine the ordering of $A$. 
Hierarchical SV Non-Relative: Active Scope & Example

- Define the *active scope* of each action by **excluding** from its scope the scopes of its direct children with respect to $\sqsubseteq$

**Definition (Active Scope)**

*If $C(b)$ denotes the direct children of $b$ with respect to $\sqsubseteq$, then the active scope of $b$, denoted by $a\text{Scope}(b)$, is defined as: $a\text{Scope}(b) = \text{scope}(b) \setminus (\bigcup_{b' \in C(b)} \text{scope}(b'))$*

- Assuming **term scoped** actions

<table>
<thead>
<tr>
<th>action</th>
<th>scope</th>
<th>active scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_1$: Best(European)</td>
<td>{European, German, Audi, BMW, Porsche, French, Citroen, Peugeot, Italian, Alfa Romeo, Ferrari, Fiat, Lamborghini}</td>
<td>{European, German, Audi, BMW, Porsche, French, Citroen, Peugeot}</td>
</tr>
<tr>
<td>$b_2$: Worst(Italian)</td>
<td>{Italian, Alfa Romeo, Ferrari, Fiat, Lamborghini}</td>
<td>{Italian, Alfa Romeo, Ferrari, Fiat, Lamborghini}</td>
</tr>
<tr>
<td>$b_3$: Best(Ferrari)</td>
<td>{Ferrari}</td>
<td>{Ferrari}</td>
</tr>
</tbody>
</table>
Hierarchical SV Non-Relative: Active Scope & Example

- Define the *active scope* of each action by excluding from its scope the scopes of its direct children with respect to $\sqsubseteq$.

**Definition (Active Scope)**

*If $C(b)$ denotes the direct children of $b$ with respect to $\sqsubseteq$, then the active scope of $b$, denoted by $aScope(b)$, is defined as:*

$$aScope(b) = \text{scope}(b) \setminus \left( \bigcup_{b' \in C(b)} \text{scope}(b') \right)$$

- Assuming **term scoped** actions

  $b_1 : \text{Best(European)}$
  
  $b_2 : \text{Worst(Italian)}$
  
  $b_3 : \text{Best(Ferrari)}$

- Assuming **object scoped** actions

<table>
<thead>
<tr>
<th>Id</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>Porsche</td>
</tr>
<tr>
<td>$L$</td>
<td>Lancia</td>
</tr>
<tr>
<td>$A_1$</td>
<td>Alfa Romeo</td>
</tr>
<tr>
<td>$A_2$</td>
<td>Alfa Romeo</td>
</tr>
<tr>
<td>$F$</td>
<td>Ferrari</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>action</th>
<th>scope</th>
<th>active scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_1 : \text{Best(Europe)}$</td>
<td>${P, L, A_1, A_2, F}$</td>
<td>${P}$</td>
</tr>
<tr>
<td>$b_2 : \text{Worst(Italian)}$</td>
<td>${L, A_1, A_2, F}$</td>
<td>${L, A_1, A_2}$</td>
</tr>
<tr>
<td>$b_3 : \text{Best(Ferrari)}$</td>
<td>${F}$</td>
<td>${F}$</td>
</tr>
</tbody>
</table>
Algorithm 3 PrefOrder\((E, \mathcal{B}, Policy)\)

**Input:** the set of elements \(E\), the set of actions \(\mathcal{B}\), and \(Policy\) for inactive elements

**Output:** a bucket order over \(E\)

1: /** Part (1): Computation of \((\mathcal{B}, \sqsubseteq)\) */
2: Compute the scopes of the actions in \(\mathcal{B}\)
3: Form \((\mathcal{B}, \sqsubseteq)\)
5: Use \((\mathcal{B}, \sqsubseteq)\) to compute the active scopes of the actions in \(\mathcal{B}\)
6: Use the active scopes to expand the set \(\mathcal{B}\) to a set \(\mathcal{B}'\)
7: /** Part (3): Derivation of the final bucket order */
8: \((\mathcal{B}, W, R_{\succ}) \leftarrow \text{Parse}(\mathcal{B}', \text{ActiveScope})\)
9: return Apply\((E, \mathcal{B}, W, R_{\succ}, Policy)\) // call to Alg. 1
Question

- What about relative preference actions over hierarchical organized single-valued attributes?
We have to redefine *scope* and *active scope*

**Definition (Scope of Relative Preferences)**

The scope of a preference relationship $e_i \succ e_j$, denoted by $\text{scope}(e_i \succ e_j)$, is defined as:

$$\text{scope}(e_i \succ e_j) = (N^*(e_i) \times N^*(e_j)) \cup (N^*(e_j) \times N^*(e_i)).$$

**Definition (Expansion of Relative Preferences)**

The expansion of a preference relationship $e_i \succ e_j$, denoted by $\text{expansion}(e_i \succ e_j)$, is defined as:

$$\text{expansion}(e_i \succ e_j) = \{ e'_i \succ e'_j \mid e'_i \in N^*(e_i), e'_j \in N^*(e_j) \}$$

- $\text{expansion}(e_i \succ e_j)$ actually “unfolds” the preference relationship $e_i \succ e_j$ on the basis of the subsumption relationships.
- The scope-based ordering of such actions is defined as before, i.e. $b \sqsubseteq b'$ iff $\text{scope}(b) \subseteq \text{scope}(b')$. 

Case Hierarchical SV Relative: Active Scope

- We can define the *active scope* of each action by excluding from its expansion the scopes of its direct children, wrt to $\sqsubseteq$

**Definition (Active Scope)**

*The active scope of a preference relationship $b$, in the context of a set of preference actions $B$ is defined as:*

$$\text{AScope}(b) = \{e_i > e_j \in \text{expansion}(b) \mid \nexists b' \in B \text{ s.t. } b' \sqsubseteq b, \text{ and } (e_i, e_j) \in \text{scope}(b')\},$$

*and this is equivalent to:*

$$\text{AScope}(b) = \text{expansion}(b) \setminus \bigcup_{b' \sqsubseteq b} \{e_i > e_j \mid (e_i, e_j) \in \text{scope}(b')\}$$

- We can use Alg. PrefOrder with *scope* and *active scope* as defined here
## Case Hierarchical SV Relative: Example

**MANUFACTURER**

- European
- Asian
  - Fiat
  - BMW
  - Kia
  - Toyota
  - Lexus

### Action

<table>
<thead>
<tr>
<th>Action</th>
<th>Expansion</th>
<th>Active scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_1 : \text{Asian} \succ \text{European}$</td>
<td>$\text{Asian} \succ \text{European}$, $\text{Asian} \succ \text{BMW}$, $\text{Asian} \succ \text{Fiat}$, $\text{Kia} \succ \text{European}$, $\text{Kia} \succ \text{BMW}$, $\text{Kia} \succ \text{Fiat}$, $\text{Toyota} \succ \text{European}$, $\text{Toyota} \succ \text{BMW}$, $\text{Toyota} \succ \text{Fiat}$, $\text{Lexus} \succ \text{European}$, $\text{Lexus} \succ \text{BMW}$, $\text{Lexus} \succ \text{Fiat}$</td>
<td>$\text{Asian} \succ \text{European}$, $\text{Asian} \succ \text{Fiat}$, $\text{Toyota} \succ \text{European}$, $\text{Toyota} \succ \text{Fiat}$, $\text{Lexus} \succ \text{European}$, $\text{Lexus} \succ \text{Fiat}$</td>
</tr>
<tr>
<td>$b_2 : \text{European} \succ \text{Kia}$</td>
<td>$\text{European} \succ \text{Kia}$, $\text{BMW} \succ \text{Kia}$, $\text{Fiat} \succ \text{Kia}$</td>
<td>$\text{European} \succ \text{Kia}$, $\text{BMW} \succ \text{Kia}$</td>
</tr>
<tr>
<td>$b_3 : \text{BMW} \succ \text{Asian}$</td>
<td>$\text{BMW} \succ \text{Asian}$, $\text{BMW} \succ \text{Kia}$, $\text{BMW} \succ \text{Toyota}$, $\text{BMW} \succ \text{Lexus}$</td>
<td>$\text{BMW} \succ \text{Asian}$, $\text{BMW} \succ \text{Kia}$, $\text{BMW} \succ \text{Toyota}$, $\text{BMW} \succ \text{Lexus}$</td>
</tr>
<tr>
<td>$b_4 : \text{Kia} \succ \text{Fiat}$</td>
<td>$\text{Kia} \succ \text{Fiat}$</td>
<td>$\text{Kia} \succ \text{Fiat}$</td>
</tr>
<tr>
<td>$b_5 : \text{Toyota} \succ \text{Kia}$</td>
<td>$\text{Toyota} \succ \text{Kia}$</td>
<td>$\text{Toyota} \succ \text{Kia}$</td>
</tr>
</tbody>
</table>

- By applying Alg. PrefOrder we get (only for the leaves):
  - $\langle \{ \text{BMW} \}, \{ \text{Asian, Toyota, Lexus} \}, \{ \text{European} \}, \{ \text{Kia} \}, \{ \text{Fiat} \} \rangle$
Case Flat MV

Question

- How can we order multi-valued attributes based on preferences defined over single terms?
Case Flat MV: MoreWins-Rule

Definition (Induced Preference over Sets: MoreWins-Rule)

If $s, s'$ are two subsets of $E$, with $\text{wins}(s, s')$ we will denote the number of "times" $s$ beats $s'$ according to $\succ$. Formally:

$$\text{wins}(s, s') = |\{(e, e') \mid e \in s, e' \in s', e \succ e'\}|$$

Any subset $S$ of the powerset of $E$ (i.e. $S \subseteq P(E)$), can be ordered according to a preference relation denoted by $\succ \{\}$, defined by the following rule:

$$s \succ \{\} s' \iff \text{wins}(s, s') > \text{wins}(s', s)$$
Case Flat MV: Example

- Let $E = \{ABS, ESP, AT, DVD\}$
- Let $ABS$ be best, $ESP$ be worst and $ABS \succ AT$
- Let $S = \{\{ABS\}, \{ESP\}, \{ABS, ESP\}, \{AT, ABS\}, \{AT, ESP\}, \{DVD, ESP\}\}$

<table>
<thead>
<tr>
<th>${s, s'}/w(s', s)$</th>
<th>${ABS}$</th>
<th>${ESP}$</th>
<th>${ABS, ESP}$</th>
<th>${AT, ABS}$</th>
<th>${AT, ESP}$</th>
<th>${DVD, ESP}$</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>${ABS}$</td>
<td>0/0</td>
<td>1/0</td>
<td>1/0</td>
<td>1/0</td>
<td>2/0</td>
<td>2/0</td>
<td>5/0</td>
</tr>
<tr>
<td>${ESP}$</td>
<td>0/1</td>
<td>0/0</td>
<td>0/1</td>
<td>0/2</td>
<td>0/1</td>
<td>0/1</td>
<td>0/5</td>
</tr>
<tr>
<td>${ABS, ESP}$</td>
<td>0/1</td>
<td>1/0</td>
<td>1/1</td>
<td>1/2</td>
<td>2/1</td>
<td>2/1</td>
<td>3/2</td>
</tr>
<tr>
<td>${AT, ABS}$</td>
<td>0/1</td>
<td>2/0</td>
<td>2/1</td>
<td>1/1</td>
<td>3/0</td>
<td>3/0</td>
<td>4/1</td>
</tr>
<tr>
<td>${AT, ESP}$</td>
<td>0/2</td>
<td>1/0</td>
<td>1/2</td>
<td>0/3</td>
<td>1/1</td>
<td>1/1</td>
<td>1/3</td>
</tr>
<tr>
<td>${DVD, ESP}$</td>
<td>0/2</td>
<td>1/0</td>
<td>1/2</td>
<td>0/3</td>
<td>1/1</td>
<td>1/1</td>
<td>1/3</td>
</tr>
</tbody>
</table>

$L = \langle \{ABS\}, \{AT, ABS\}, \{ABS, ESP\}, \{\{AT, ESP\}, \{DVD, ESP\}\}, \{ESP\} \rangle$
Case Flat MV: Breaking ties

Question

- But how to break ties when we compare sets with only **best** or only **worst** elements?
Case Flat MV: MoreGoodLessBad-rule

Definition (Breaking ties: MoreGoodLessBad-rule)

- If \(\text{wins}(s, s') = \text{wins}(s', s) = 0\) and \(\text{Support}(s) > \text{Support}(s')\) then \(s \succ \{\} \ s'\), where \(\text{Support}(s) = \sum_{e \in s} \text{sup}(e)\) and \(\text{sup}(e) = |\{e' \in E \mid e \succ e'\}| - 1\)

- Let both \(ABS\) and \(ESP\) be \textbf{best} and \(AT\) and \(DVD\) be \textbf{worst}

\[
\text{wins}(\{ABS\}, \{ABS, ESP\}) = \text{wins}(\{ABS, ESP\}, \{ABS\}) = 0
\]
\[
\text{wins}(\{AT\}, \{AT, DVD\}) = \text{wins}(\{AT, DVD\}, \{AT\}) = 0
\]

- Then, \(\text{Support}(\{ABS, ESP\}) = 2 > \text{Support}(\{ABS\}) = 1 > \text{Support}(\{AT\}) = -1 > \text{Support}(\{AT, DVD\}) = -2\)

- So, \(L = \langle\{ABS, ESP\}, \{ABS\}, \{AT\}, \{AT, DVD\}\rangle\)
**Algorithm 4** ApplyOverSets($E, B, W, R_\succ, Policy$)

**Input:** the set of elements $E$ (here each element of $E$ is a set), the set of best elements $B$, the set of worst elements $W$, a set of relative relationships $R_\succ$, and $Policy$ for inactive elements

**Output:** a bucket order over $E$

1: $R_{bw} \leftarrow \{(b, w) \mid b \in B, w \in W\}$
2: $R \leftarrow R_{bw} \cup R_\succ$
3: $R \leftarrow Closure_{transitivity}(R)$ // Addition of the transitively induced links

4: **for** each $e, e' \in E$, s.t. $e \neq e'$ **do**
5:  **if** $wins(e, e') > wins(e', e)$ **then**
6:    set $e \succ \{\} e'$
7:  **else if** $wins(e, e') < wins(e', e)$ **then**
8:    set $e' \succ \{\} e$
9:  **else if** $wins(e, e') = wins(e', e) = 0$ **then**
10:    **resolve the tie by computing the support(e) and support(e')**

11: $L \leftarrow SourceRemoval(\succ \{\})$
12: $I \leftarrow E \setminus \text{dom}(\succ \{\})$ // $I$ is the set of inactive elements
13: $L' \leftarrow \text{addInactiveElements}(L, I, Policy)$
14: **return** $L'$
Case Hierarchical MV

Question

- What about relative preference actions over hierarchical organized multi-valued attributes?
Ordering → Hierarchical MV (Relative)

- Exploit **inheritance** by combining PrefOrder and ApplyOverSets

### Algorithm 5 PrefOrderSetValued($E, B, Policy$)

**Input:** the set of elements $E$ ($E$ is a family of sets), the set of actions $B$, and $Policy$ for inactive elements

**Output:** a bucket order over $E$

1. // As in Alg. PrefOrder
2. Compute the *scopes* of the actions in $B$ and form $(B, \sqsubseteq)$
3. Use $(B, \sqsubseteq)$ to compute the *active scopes* of the actions in $B$
4. Use the active scopes to expand the set $B$ to a set $B'$
5. $(B, W, R_{\succ}) \leftarrow Parse(B')$
6. // As in Alg. ApplyOverSets
7. $R_{bw} \leftarrow \{(b, w) \mid b \in B, w \in W\}$
8. $R \leftarrow R_{bw} \cup R_{\succ}$
9. $R \leftarrow Closure_{transitivity}(R)$ // Addition of the transitively induced links
10. Compute $\succ \{\}$ based on *wins* and *support* as in Alg. ApplyOverSets
11. $L \leftarrow SourceRemoval(\succ \{\})$
12. $I \leftarrow E \setminus dom(\succ \{\})$ // $I$ is the set of inactive elements
13. $L' \leftarrow addInactiveElements(L, I, Policy)$
14. return $L'$
Algorithms: Composition of Preference Actions

Composition of object scoped actions over different facets

- Prioritized Composition
- Pareto Composition
- Combination of the above
Algorithmic Complexity & Optimizations
Algorithms: Complexities

For Ordering Facets & Terms
- The number of facets and terms is usually small

For Ordering Objects
- Number of Obj can be huge
- Cost of computing scopes is very high (i.e. based on objects)

By exploiting FDT
- A few clicks can reduce objects to a focus A
- Preferences can be applied when |A| is relatively small (less than a threshold)
Algorithms: Complexities & Optimizations (Exploiting FDT)

- Compute $b_1 \sqsubseteq b_2$ without computing scopes based on objects
  ... but by using anchors and the subsumption relation $\leq$
  ... this can be done very fast using labeling
- Derivation of objects ordering in $A$ by using the descriptions of objects in $A$

<table>
<thead>
<tr>
<th>Alg.</th>
<th>Object-based</th>
<th>Focus-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceRemoval&amp; Apply</td>
<td>(O(</td>
<td>\text{Obj}</td>
</tr>
<tr>
<td>PrefOrder</td>
<td>(O(</td>
<td>\text{Obj}</td>
</tr>
<tr>
<td>ApplyOverSets</td>
<td>(O(</td>
<td>T_i</td>
</tr>
<tr>
<td></td>
<td>(</td>
<td>E</td>
</tr>
<tr>
<td>MFPriority</td>
<td>(O(</td>
<td>\text{Obj}</td>
</tr>
<tr>
<td>MFPareto</td>
<td>(O(</td>
<td>\text{Obj}</td>
</tr>
</tbody>
</table>

Usually $|B| < |T_i| < |\text{Obj}|$ and $|A| \ll |\text{Obj}|$.
In the worst case $|B| = \frac{|T_i|(|T_i|-1)}{2}$
Executive Summary of the Preference Framework

Key aspects of our approach:

- Support of **hierarchically organized attribute values**
- Support of **multi-valued attributes**
- Support **"aggregated" semantics of several preference actions** (needed for the **gradual** method that we want to support at the interaction level) with a **scope-based method for resolving conflicts**
- Prioritized and **Pareto** multi-facet composition
- Algorithms for **producing the preference-based ordering** without having to **compute the scopes** of the actions
Hippalus
**Hippalus offers:**

- **Browsing** and **exploration** of RDF sources using the FDT paradigm
- Enriched with the **proposed preference framework**

**Hippalus uses:**

- **Jena** framework
- Preference actions provided through **HTML5 context menus** (best, worst, lexicographic, count, value, relative, around)
- **HTML5 drag & drop for composition** of preference actions over different facets

- A demo of Hippalus is available at http://www.youtube.com/watch?v=Cah-z7KmlXc
Demo
Evaluation

- 1. Study of FDT convergence and preferences
- 2. Evaluation of DiFEPreKO hypothesis
- 3. Evaluating Exploratory Interaction Schemes
- 4. Evaluation of Hippalus system
1. Choices and Effort: FDT Convergence + Preferences

FDT converges fast

- i.e. the user can greatly restrict the focus with a few clicks
- Optimizations are based on the assumption that focus $A$ can be reduced very fast

Enrichment of FDT with preference actions

- Assuming that the most preferred option for each facet is prompted first
  ... ideally reduces the number of choices to the number of required clicks

Examples

- Assume $k$ complete and balanced tree taxonomies of depth $d$ and degree $b$. Each leaf indexes 10 objects
- Optimally the number of choices is $k \times b \times d$ while number of clicks is $k \times d$
- Selection of desired 10 objects from a peta-sized collection needs only 30 clicks
- Dynamic taxonomies will further reduce the number of choices

<table>
<thead>
<tr>
<th>$n/10$</th>
<th>$k$</th>
<th>$b$</th>
<th>$d$</th>
<th>Num. of Choices</th>
<th>Num. of Clicks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$531.441 \sim 10^6$</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
<td>12</td>
</tr>
<tr>
<td>$3.486.784.401 \sim 10^{11}$</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>60</td>
<td>20</td>
</tr>
<tr>
<td>$\sim 10^{15}$</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>90</td>
<td>30</td>
</tr>
</tbody>
</table>
2. User Based: DiFEPreKO Hypothesis

DiFEPreKO Hypothesis: Difficulty of Formulating Effective Preferences without Knowing the Options

Without the ability to view and explore the existing choices, the expression of preferences is time-consuming and in most cases results to incomplete preferences (i.e. preferences that are not sufficient for selecting the most desired option from a particular set of choices).

- **30 Participants**: 18 male and 12 female from 12 countries
- Initially participants were asked to **intentionally** express preferences regarding the ideal car they can buy
- Afterwards they were asked to select the car they would buy from a list of 50 cars

**Results:**

- On average users expressed in **10 minutes 9.73 intentionally** preferences
- They spent **4 minutes** on average to select the ideal car from the list
- Only **20%** selected the ideal car according to their **intentionally defined preferences**
- Users spent a lot of time expressing preferences that are **not applicable** to the characteristics provided in the list of cars or are not **consistent** with their final decision
2. User Based: DiFEPreKO Hypothesis (Statistical Significance)

**Null Hypothesis**

*More than half* of the users expressed their preferences without exploring available cars, and were returned the ideal car for them from a car collection.

**Alternate Hypothesis**

*Less than half* of the users expressed their preferences without exploring available cars, and were returned the ideal car for them from a car collection.

**Statistical Significance Results:**

- We have a *very strong* evidence against the *Null Hypothesis* ($\alpha$ value of 0.01)
3. User Based: Exploration Approaches

- We evaluated **3 different exploratory UIs** over Mitos (Clustering, FDT, Clustering + FDT)
- **2 users groups**: 10 *Plain* users and 3 *Expert* ones
- **Four exploratory tasks** based on tasks refinement steps described in (Kules and Capra, 2008)

**Results (described in detail in (Papadakos et al., 2012a)):**

- FDT-based approaches were **the most preferred**
- **Higher degree of task completeness** with FDT-based approaches (20% for advanced users)
- **Least number of queries** with FDT-based approaches (50% for advanced users)
- More than 50% of both plain and expert users were **highly satisfied** from FDT-based UIs
- The clustering only UI was **the least preferred** for 65% of plain users
- Clustering was **the least satisfactory UI** (16.6% and 12.5% for advanced and plain users resp.)
- With 95% confidence degree, only 5% of the regular users were not satisfied by the FDT only interface
4. User Based: Hippalus 🌼 (Evaluation Setup)

- We evaluated 2 different UIs over Hippalus (plain FDT and FDT + preferences)
- 2 users groups, 20 participants for plain and 6 expert users
- Each user completed four tasks, 2 for both UIs, using both Pareto and Priority composition
  - Tasks were designed over the most commonly user expressed attributes of the DiFEPreKO hypothesis (i.e. Price, Manufacturer, Engine Volume, Body Type, Fuel Type)
    - Plain users tasks used criteria over 3 attributes
    - Expert users tasks used criteria over 6 attributes
- Graeco-Latin Square Design rotating both the order of tasks and the order in which participants experience interfaces (Kelly, 2009)
  - Control order effects and increase the chance that results can be attributed to the experimental treatments and conditions
- Information base of only 50 cars
4. User Based: Hippalus ☺ (Results)

- **All users preferred** the preference UI over the plain FDT
  - Plain users: 75% preferred it **very strongly**, 20% **strongly**, 5% **strong enough**
  - Expert users: 50% preferred it **very strongly** and 50% **strongly**

- **None of the users** was able to successfully complete both tasks with the plain FDT
  - Only 10% of plain and 16.6% of expert users completed successfully one of the two tasks

- All users completed **successfully all** of the tasks using the preference UI
- On average with the preference UI they used a **third of the time** to complete the tasks
- ... using a **third of user interactions** compared to the plain FDT

**Time - Actions**

- **Time (sec)**: Plain (600), Expert (500)
- **Actions**: Plain (100), Expert (120)

**IR Metrics**

- **Recall**, **Precision**, **AP**: U1 (0.8), U2 (0.7)
Conclusion & FW
We introduced a preference framework especially suited for the interaction scheme of FDT.

The framework supports:
- **Incremental** preference specification mode
- **Hierarchically organized** attribute values
- **Scope-based** method for resolving conflicts
- **Multi-valued** attributes
- Session-based **Prioritized** and **Pareto** composition

Algorithmic perspective:
- Studied the **complexities** of the algorithms supporting the above framework
- Provided logical optimizations so that algorithms are **independent of the size of the information base** (depend only on the focus A)
Results show that:

- The hypothesis that users cannot express complete and effective preferences without knowing the available choices is **supported**.
- FDT allows **restricting the information space very fast** and is the **most preferred UI for exploratory tasks**.
- Evaluation over Hippalus showed that the **preference-based UI helped** the users to:
  - **successfully** find the **desired results**
  - **in less time** (1/3 of the time in our experiments)
  - **and with less interactions** (1/3 of the interactions in our experiments)
## Future Research

### Applicability
- Develop wrappers that can be used for feeding Hippalus with results of queries over different sources like:
  - OpenSearch compatible search engines
  - Database sources
  - SPARQL queries over different SPARQL endpoints

### Algorithms
- It is still interesting to investigate optimizations for the case where the current answer is very big

### Structure of the information space
- Investigate required advancements of the interaction model and preference framework
  - ... for information spaces with more complex structures like RDF/S
  - ... or objects accompanied with numbers expressing different quality aspects (i.e. trust, authority, popularity, accuracy, specificity)
Publications

Related to the application of FDT over a WSE


3) Papadakos, P., Kopidaki, S., Armenatzoglou, N., and Tzitzikas, Y. “Exploratory Web Searching with Dynamic Taxonomies and Results Clustering”. In HDMS ’09: Proceedings of the 8th Hellenic Data Management Symposium. 2009b


Http://www.ieee-tcdl.org/Bulletin/v6n1/Papadakos/papadakos.html


Extension of FDT with Preferences


Under Review


Related to IR indexing and Querying


References


Kules, B. and Capra, R.  **Creating Exploratory Tasks for a Faceted Search Interface.** In *Procs of the 2nd Workshop on Human-Computer Interaction (HCIR’08)*. 2008.


References


Interactive Exploration of Multi-Dimensional Information Spaces with Preference Support

Thank you!

Questions?

Panagiotis Papadakos
papadako@csd.uoc.gr

Supervisor: Yannis Tzitzikas

Computer Science Department
University of Crete, Greece

ICS-FORTH, Greece

*Image and modified XeLaTeX template available from https://github.com/drbunsen/drbunsen-beamer