Comparative Preferences Induction Methods for Conversational Recommenders

Walid Trabelsi[1,2] and Nic Wilson[1] and Derek Bridge[2]

[1] Cork Constraint Computation Centre
w.trabelsi@4c.ucc.ie, n.wilson@4c.ucc.ie
[2] Department of Computer Science,
University College Cork, Ireland
d.bridge@cs.ucc.ie

Abstract. In an era of overwhelming choices, recommender systems aim at recommending the most suitable items to the user. Preference handling is one of the core issues in the design of recommender systems and so it is important for them to catch and model the user's preferences as accurately as possible. In previous work, comparative preferences-based patterns were developed to handle preferences deduced by the system. These patterns assume there are only two values for each feature. However, real-world features can be multi-valued. In this paper, we develop preference induction methods which aim at capturing several preference nuances from the user feedback when features have more than two values. We prove the efficiency of the proposed methods through an experimental study.

1 Introduction

Choosing the right or the best option is often a demanding and challenging task for the user when there are many available alternatives (e.g., a customer in an online retailer). Recommender systems aim at recommending the most suitable items to the user. However, the recommended items proposed by the system may not match the users' needs as recommender systems might miss on the users' preferences (see, e.g., [1]). One approach which ensures that the system is kept aware of the user needs is to establish a conversation between the user and the system by means of conversational recommender systems.

Preference handling is one of the core issues in the design of recommender systems and so it is important for them to catch and model the user's preferences as accurately as possible. In fact, preferences aim at offering the user the ability to express her relative or absolute satisfaction when faced with a choice between different options. One of the major approaches in today's recommender systems is utility functions which assign a numerical score to each data item [2]. A second major approach is the relational preference structures [3] as the user may wish to state simple comparisons. She may want to make no explicit quantification of preference or utility, leaving the preference purely qualitative. This could be the case for example in a travel problem, where there is a large number

of possible attributes involving times, transportation means and locations that vary from one user to another. In such a case, the user may want to say that she likes to travel to a country during the summer in that country, with all other attributes being equal. The user will then avoid having to communicate an accurate numerical model. It has also been claimed that the qualitative specification of preferences is more general than the quantitative one, as not all preference relations can be expressed by scoring functions [4].

In this paper, we look for elaborate and generic comparative preferences-based induction methods that we can prove to be efficient in practice with conversational recommender systems that suggest multi-valued feature products to the user.

The rest of the paper is organized as follows. In Section 2, we give an overview of conversational recommender systems and how preferences are handled in these systems. Then, Section 3 describes the framework of preference dominance that will be used in this paper. The conversational recommender system we are using is detailed in Section 4. A fundamental step in the process of recommending for conversational recommender system is preferences induction. We introduce a number of preference induction methods in Section 5. We performed experimentations which allowed us to assess the efficiency of these methods with regards to the obtained results which we discussed in Section 6. In Section 7, we conclude the paper with possible extensions of the proposed approaches.

2 Related Work

2.1 Conversational recommender systems

Generally speaking, people do not state their preferences up-front because initially they only have a vague idea of the product they would like to have [5]. Usually, criteria about the product the customer would like to purchase are specified during the dialogue with the seller. This is still the case even for knowledgable customers in the domains where expert users need to be assisted because available products dynamically change. A distinctive example is the list of special offers (e.g., flight tickets) which change frequently.

Conversational recommender systems [5] recognise that their users may be willing and able to provide more information on their constraints and preferences, over a dialogue. The main difference with the single-shot recommendation scenario is that in the case where the user is not satisfied she can revise her request.

2.2 Preference Handling in Conversational Recommender Systems

The acquisition of preferences is a central challenge in interactive systems like recommender systems [6]. There are two major approaches in today's recommender systems: utility functions [2] and relational preference structures [3]. A utility function assigns a numerical score to each item. Relational preference

structures link pairs of items through the notions of "is preferred to" and "is equally preferable as" thus leading to qualitative preference orderings. Typically, the task of the recent online conversational recommenders is to elicit the customer requirements, while interacting with her, in a personalized way.

Critiquing [7] is an interaction model that allows users to build their preferences by examining or reviewing examples shown to her by the system. The user feedback employed in conversational recommender systems was also studied in [8] through two comparison-based recommendation approaches: *More Like This* (MLT) and *Partial More Like This* (PMLT). Their role is to induce preferences when the user reacts to the recommended items. They both generate preference statements stating the preference of features that mark the selected item over those that characterize the rejected items during an interaction stage. Information Recommendation [9] is a recommendation approach that aims at suggesting to the user how to reformulate her queries to a product catalogue in order to find the products that maximize her utility. In [9], the authors showed that, by observing the queries selected by the user among those suggested, the system can make inferences on the true user utility function and eliminate from the set of suggested queries those with an inferior utility. Authors in [10] proposed a novel use of the formalism of preference elicitation in [9]. They invoked comparative preferences-based patterns to handle the preferences deduced by the system. These patterns assume there are only two values for each feature. However, real-world features can be multiple-valued. In this paper, we investigate preference induction methods which can handle the user preferences in a conversational recommender for products with multiple-valued features.

3 CP-Tree-Based Dominance

Products in online databases need to be compared by pairs, through dominance testing, to find out which options are dominated and to eliminate them consequently . In this paper, dominance testing is based on some structure called cp-trees which were introduced in [11].

3.1 Description of a CP-Tree

A cp-tree is a directed rooted tree. Associated with each node N in the tree is a set of variables Y_N. Let γ be the maximum number of variables in Y_N. The cp-tree represents a form of lexicographic order where the importance ordering on nodes and their assignments depends on more important nodes and their assignments.

Example 1. Let $V = \{X, Y, Z\}$ be a set of variables whose domains are as follows. $\underline{X} = \{x1, x2\}$, $\underline{Y} = \{y1, y2\}$ and $\underline{Z} = \{z1, z2\}$ respectively. Figure 1 represents an example of a cp-tree with $\gamma = 1$. Each node in the cp-tree depicted in Figure 1 is labeled with a variable. The root is labeled by X as the most important variable. Each node is also associated with a preference ordering of the values of the variable. We can see the total pre-order of the outcomes below the cp-tree.

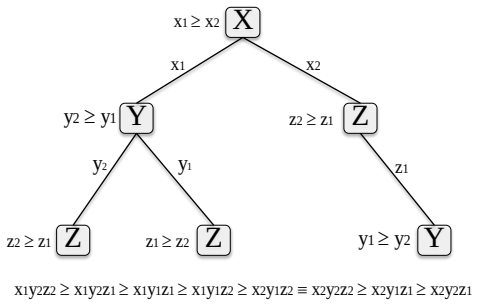

Fig. 1. A cp-tree σ, along with its associated ordering $\succcurlyeq_\sigma$ on outcomes, with $\gamma = 1$ (i.e., with at most one variable associated with a node)

3.2 CP-Tree-Based Dominance

Let Γ be a set of comparative preference statements. Let $\succcurlyeq_\Gamma$ be the associated preference relation of Γ on outcomes. Let α and β be two outcomes. The following definition is based on [11].

Definition 1. *α dominates β if and only if all possible cp-trees (every cp-tree represents a total pre-order) that satisfy Γ, prefer α over β. In other words, $\alpha \succcurlyeq_\Gamma \beta$ holds if every cp-tree σ that extends all preferences in Γ has α come before β.*

In this paper, we are using a dominance testing as stated in Definition 1 and which can check, in polynomial time, whether $\alpha \succcurlyeq_\Gamma \beta$.

4 The Case Study: A "Select and Get More Products" Conversational Recommender System

The product search, which needs a filter-based retrieval, can take place in tandem with preference elicitation. This motivates our present work which suggests a conversational recommender system that guides the user towards her target through a simple conversation during which the system can deduce different forms of comparative preferences from the user feedback.

4.1 The Advisor

The advisor helps the user identify a suitable product to purchase among the relatively large set of available products. The proposed system in this paper can be regarded as an instance of a kind of system in which the user is repeatedly shown products and criticises one or more of the shown products, until she finds a product that she is keen to have. During the interaction with the user, the

advisor infers preference relations from the user's selections at each step. The selected potential products that are shown to the user are meant to best match these preference relations.

The user's preferences are stored in a kind of user model which is progressively updated by the system as the dialogue continues. Having that user model, the system starts to be able to determine whether certain products dominate others and to suggest to the user those that are not dominated. Dominance is computed as described in Section 3.2. The advisor asks little from the user who will only select one of the products shown to her. This will steer the user smoothly towards her target.

4.2 Products

We assume that the products are modeled with a collection of n multi-valued features $V = \{F_1, \ldots, F_n\}$. The features are intended to relate to a set of products that the user is interested in choosing between. For example, if the product is a hotel, one feature might be the size of a swimming pool in the hotel (e.g., the swimming pool can be small, medium or large).

4.3 Dialogue

Let Ω be the global set of products. Let $\mathcal{K}$ (i.e., 9) be the maximum number of products shown to the user in each step of the dialogue.

Initially, the user is shown the first $\mathcal{K}$ non-dominated products retrieved from Ω and selects a product $\mathcal{P}$. The interaction between the user and the recommender system proceeds as follows:

- The recommender system analyzes the current product $\mathcal{P}$ and induces some constraints on the user's preferences with particular regard to differences between $\mathcal{P}$ and the products the user might have selected. The nature of the induced preferences depends on the induction method used.
- From the second step of the dialogue, the system computes $\mathcal{K}$ non-dominated products among the remaining products. In fact, the system keeps retrieving products from those remaining in the database and not yet checked by the system then pruning the dominated ones until finding a set of $\mathcal{K}$ non-dominated products or there are no more products remaining in Ω. The system adds the product that the user selected in the previous step of the dialogue to the set of non-dominated products already computed. Since $\mathcal{K}$ is set to 9 in these experiments, the user is shown 10 products. If the number of products remaining in the database is less than $\mathcal{K}$ then we select the non-dominated among the remaining products and we show them to the user with the product chosen in the previous step.
- The user selects a product which becomes the new current product $\mathcal{P}$.

The sequence of steps stated above is repeated until the user is satisfied with $\mathcal{P}$ (by either choosing the same product a number of times (set to 3) or she gets the most preferred product with regards to her true preferences) or the set of remaining products in Ω is empty.

5 Induction of Constraints on Preferences Within the System

Each time the user reviews a product, the recommender system, described in Section 4 induces some of the user's preferences. There are several induction methods that specify patterns of preference statements that can be induced from the user's selection. In this section, we discuss a number of these methods.

5.1 Constraint Language

Let V be a set of variables. Wilson [11] presented comparative preference theories which involve preference statements φ of the form $p > q||T$ where p and q are the respective assignments to sets of variables P and Q, and T is a set of variables ($P \subseteq V$, $Q \subseteq V$ and $T \subseteq V$). Such a statement expresses a preference for an assignment p over another assignment q with variables T held constant. We are specifically using Wilson's preference language to allow the system to handle the preferences induced from the user selection.

5.2 Preferences Deduction Within the System

This section explains what the system induces when it observes the product that the user selects and the remaining products that were shown but not selected by the user. We adopt approaches which are based on comparative preference and partially inspired from *MLT* and *PMLT* approaches briefly described in Section 2.2. We have derived three patterns of preference statements that the system can induce when the user makes her selection. Let $V = \{F_1, \ldots, F_n\}$ be a set of n variables that represent features. Let C and R be two products. The combinations of feature values in the two products are denoted as follows. C=$\{f_1^C \ldots f_n^C\}$ and R=$\{f_1^R \ldots f_n^R\}$ where f_i^C and f_i^R are the two respective values that C and R have for the feature F_i (i=$\{1, \ldots, n\}$).

When the user chooses a product C and rejects another product R among a set of $\mathcal{K}$ non-dominated products that are shown to her, the system induces preference statements whose form depends on the following inference methods.

- **Basic:** A straightforward kind of preference to be induced is to express the preference of the features values combination included in C over the combination of values included in R. Thus, we model the following preference statement $f_1^C \ldots f_n^C \geq f_1^R \ldots f_n^R \; ||\emptyset$.

- **Lex-Basic:** Lexicographic preference models are regarded as simple and reasonably intuitive preference representations, and so lexicographic ordering can be well-understood by humans that use it to make preference decisions [12]. This is why we adopt a lexicographic model of the **Basic** format described above. This pattern allows the system to induce unconditional preference statements with regards to **Basic**.

Let $U \subseteq V$ be the set of features for which C and R have the same values (i.e., C and R agree on U). Let $S \subseteq V$ be the set of features for which C and R have different values. The combinations of features values of C and R can be represented by assignments us and us' respectively, with u is the assignment to U (i.e., $u \in \underline{U}$), and s and s' are the respective assignments to S of C and R which differ on each feature: $s(F) \neq s'(F)$ for all $F \in S$.
Instead of stating the preference of us over us' as in **Basic** (i.e., $us > us'||\emptyset$), the idea is to induce a preference statement saying partial assignment s is preferred over partial assignment s' with all remaining features in V (i.e., features not in S) being equal. Then, the induced preference statement can be written as $s > s'||U$, the corresponding unconditional statement.

- **Every-Selected-Value:** We induce the preference statements $f_i^C \geq f_i^R||V \setminus \{F_i\}$, for every value f_i^C assigned to feature F_i in the chosen product C and for any value f_i^R assigned to feature F_i in a rejected product R.
  This states the superiority of every feature value taken by the chosen product C (i.e., f_i^C) over any other value (of the same feature) that appears at least in one rejected product R (i.e., f_i^R).

- **Cond-Selected-Value:** We induce the preference statements $\hat{f_i^C} \geq f_i^R||V \setminus \{F_i\}$. $\hat{f_i^C}$ represents every value assigned to feature F_i in the chosen product C without being present in any rejected product R. f_i^R represents every value assigned to feature F_i in a rejected product R.
  This states the superiority of every feature value $\hat{f_i^C}$ taken by the chosen product C, and which does not appear in any rejected product R, over any other possible value f_i^R (of the same feature) that appears at least in one rejected product R. The difference with *Every-Selected-Value* form is that the preferred feature value $\hat{f_i^C}$ needs to be present in the chosen product C and it should not appear in any rejected product R while *Every-Selected-Value* involves all feature values in C.

Let Φ_B, Φ_{LB}, Φ_{ESV} and Φ_{CSV} be the sets of preference statements induced by the system with **Basic**, **Lex-Basic**, **Every-Selected-Value** and **Cond-Selected-Value** respectively. We shall notice that any cp-tree that satisfies Φ_{LB} will also satisfy Φ_B as preference statements in Φ_{LB} imply statements in Φ_B. We also notice that the set of preference statements in Φ_{CSV} is included in Φ_{ESV}. Thus, all cp-trees that agree with statements in Φ_{ESV} also agree with statements in Φ_{CSV}. The set of cp-trees S_{Φ_B} that satisfy Φ_B is likely to be larger than the set of models $S_{\Phi_{LB}}$. This can explain a weaker inference for **Basic** with regards to **Lex-Basic**. The set of models that agree with statements in Φ_{ESV} will necessarily be smaller than the set of models satisfying Φ_{CSV} as $\Phi_{ESV} \subseteq \Phi_{CSV}$. Thus, this will probably make the dominance relation based on Φ_{ESV} stronger than the dominance relation based on Φ_{CSV}.

Example 2. Let $V = \{F_1, F_2, F_3\}$ be a set of features whose domains are as follows. $\underline{F_1} = \{f_1^1, f_1^2, f_1^3\}$, $\underline{F_2} = \{f_2^1, f_2^2\}$, $\underline{F_3} = \{f_3^1, f_3^2, f_3^3, f_3^4\}$. Let us assume

that the user is initially shown the three following products: $f_1^1 f_2^1 f_3^2$, $f_1^2 f_2^1 f_3^3$ and $f_1^2 f_2^1 f_3^1$. Then, the user chooses $f_1^2 f_2^1 f_3^3$. The system will induce preferences in a format that depends on which among the methods introduced above is used. For **Basic**, the system induces the set of statements $\Phi_1 = f_1^2 f_2^1 f_3^3 \geq f_1^1 f_2^1 f_3^2 \ ||\emptyset$, $f_1^2 f_2^1 f_3^3 \geq f_1^2 f_2^1 f_3^1 \ ||\emptyset$. For **Lex-Basic**, the system induces the set of statements $\Phi_2 = \{f_1^2 f_3^3 \geq f_1^1 f_3^2 || \ \{F_2\}$, $f_3^3 \geq f_3^1 || \{F_1, F_2\}$. For **Every-Selected-Value**, the system induces the set of statements $\Phi_3 = \{f_1^2 \geq f_1^1 || \{F_2, F_3\}, f_3^3 \geq f_3^2 || \{F_1, F_2\}, f_3^3 \geq f_3^1 || \{F_1, F_2\}$. For **Cond-Selected-Value**, the system induces the set of statements $\Phi_4 = f_3^3 \geq f_3^2 || \{F_1, F_2\}, f_3^3 \geq f_3^1 || \{F_1, F_2\}$.

Let us suppose now that the system wants to show other undominated products to the user who is not yet satisfied with her selection. The system has three other products in the database and it will check whether they are dominated. Let $\alpha = f_1^2 f_2^2 f_3^3$ and $\beta = f_1^1 f_2^2 f_3^1$ be two of them.

For **Basic**, $\alpha \not\succcurlyeq_{\Phi_1} \beta$ as we can identify cp-trees which satisfy preference statements in Φ_1 and prefer β over α. An example illustrating this is a cp-tree σ with root node associated with variable F_2 (and value ordering e.g., such that $f_2^1 \succ f_2^2$), and associated with value f_2^2 is a child node with variable F_1 and local ordering such that $f_1^1 \succ f_1^2$.

For **Lex-Basic**, $\alpha \succcurlyeq_{\Phi_2} \beta$ as all cp-trees that satisfy preference statements in Φ_2 have nodes F_1 with a local ordering such that $f_1^2 \succeq_{F_1} f_1^1$, and nodes F_3 with a local ordering such that $f_3^3 \succeq_{F_3} f_3^1$ and $f_3^3 \succeq_{F_3} f_3^2$. All these cp-trees prefer α over β as any product that has f_1^2 and f_3^3 as values for F_1 and F_3 respectively will be preferred over any product that has f_1^1 and f_3^1 for F_1 and F_3 respectively. For **Every-Selected-Value**, with a similar justification as $\alpha \succcurlyeq_{\Phi_2} \beta$, $\alpha \succcurlyeq_{\Phi_3} \beta$.

For **Cond-Selected-Value**, $\alpha \not\succcurlyeq_{\Phi_4} \beta$ as there exist cp-trees which satisfy Φ_4 but prefer β over α. The cp-tree σ described above is an illustrative example.

6 Experimentation and Results

This section describes experiments to assess the inference methods presented in Section 5.2. By these experiments we aim at showing the applicability and efficiency of these methods since this is the first time it is applied in the context of recommender system with multi-valued features. These experiments illustrate how a recommender system can exploit the expressiveness of comparative preferences and their relatively fast preference dominance engine.

6.1 Experiment Design

We report experiments with simulated users. The ultimate evaluation and validation of the preference dominance approaches for conversational recommender systems should be performed online. However, experiments with real users cannot be used to extensively test alternative newly-deployed interaction control algorithms. Indeed,a number of researchers pointed out the limitations of offline experiments and their evaluation mechanisms, whereas others argued that

off-line experiments are attractive because they allow comparing a wide range of approaches at an affordable cost [13].

We make assumptions concerning the behaviour of users. For a simulated user to make choices about which among the recommended products is the best one for her to have, she must be assigned a set of *true preferences*. The user's true preferences are represented either in the weights vector model by randomly generating weights vectors over product features or in the cp-tree model by randomly generating cp-trees over product features. The weights are related to product features; they are randomly selected real numbers in the interval [0,1]. The cp-trees representing the user's true preferences have the same structure as the cp-tree described in Section 3.1.

We have generated random products with n (e.g., 10) variables having three values each. Four recommenders use the four induction methods while other four recommenders consider four combinations of these approaches. For each pairing of a user with a recommender system, we ran 1,000 simulated dialogues. In total then, we are reporting results for 8 ways of inducing the user's preferences $\times$ 2 ways of representing the user's true preferences $\times$ 1,000 dialogues, which is 16,000 runs of the system. Experiments were run as a single thread on Dual Quad Core Xeon CPU, running Linux 2.6.25 x64, with overall 11.76 GB of RAM, and processor speed 2.66 GHz.

6.2 Pruning

The recommender system considered in this work will keep only those products which are not dominated regarding the user's preferences collected so far during the dialogue between the user and the system. In the experiments, we compare the pruning rates achieved by the eight recommender systems. As mentioned in Section 4.3, in each step of the dialogue, the goal of the system is to show a (predefined) number of non-dominated products to the user. Thus, the system selects $\mathcal{K}$ non-dominated products from a subset of $\mathcal{L}$ products among those remaining in the global set of products and not yet retrieved by the system. The pruning rate is defined as the proportion of $\mathcal{K}$ in $\mathcal{L}$.

6.3 Discussion of Results

The capability of pruning dominated combinations of features is an important success key of a conversational recommender system. But, the pruning capacity is not sufficient to make a conversational recommender system prevail over another. For instance, when the system prunes a large number of products, the user-system dialogue could be longer and the user might take more time to meet her target. Therefore, several factors might determine how good a conversational recommender system is. These factors include the pruning rate (Pruning), the running time (Time), the dialogue length (Steps) and the shortfall (Fall). The running time records, in milliseconds (ms), the time spent in checking the dominance of the products. The shortfall expresses how far is the preference of the product the user ended up with from the best product (in the database) the

user could have obtained (in percentage). Table 1 and Table 2 give the results of the experiments with the true preferences of the simulated users represented as weights vectors and cp-trees respectively. The measures shown are averaged over 1, 000 dialogues.

Table 1. Averages (over 1,000) of the pruning rates, the computation time, the number of steps per dialogue and the shortfalls for each induction method and each combination of induction methods (users as weights vectors)

Induction methods	*Pruning (%)*	*Time (ms)*	*Steps*	*Fall (%)*
Basic	3.03	0.017	6.06	0.068
Lex-Basic	27.73	0.03	5.59	0.064
Every-Selected-Value	55.09	0.027	4.94	0.052
Cond-Selected-Value	0.42	0.008	6.1	0.069
Basic + Every-Selected-Value	55.09	0.036	4.94	0.052
Basic + Cond-Selected-Value	5.17	0.018	6	0.069
Lex-Basic + Every-Selected-Value	55.09	0.049	4.94	0.052
Lex-Basic + Cond-Selected-Value	30.42	0.033	5.52	0.065

Table 1 shows that, the amount of pruning increases as the preference statements induced become less conservative (from *Basic* to *Every-Selected-Value*). For example, pruning goes from 3.03% *Basic* to 55.09% *Every-Selected-Value*. *Lex-Basic* has also significantly improved its pruning rate with regards to *Basic* (27.73% versus 3.03%) after unconditioning the preference statements that were conditional in *Basic*. The exception to this is *Cond-Selected-Value* case (0.42%) which is probably due to the fact that the system induces much less preference information about the user. In fact, experiments have shown that one feature value that is seen in the chosen product is likely to be in at least one of the rejected products which makes the system refrain from inducing a preference statement that involves that feature value when *Cond-Selected-Value* is adopted. Thus, *Cond-Selected-Value* implies preference statements that would be satisfied by a quite large set of models which makes the inference weaker. *Cond-Selected-Value* has the smallest pruning rate with no positive effect on the shortfall. *Every-Selected-Value* distinguishes itself by having the best pruning capability (55.09%) and the shortest dialogue (4.94) that did not prevent it from having the best shortfall (0.052%).

When combined with a more conservative method as *Basic* or *Lex-Basic*, *Every-Selected-Value* takes longer period of time (0.036ms and 0.049ms versus 0.027ms) even though the pruning and the dialogue length are still the same. We can also see the running time is increasing with the pruning rate. In fact, this conversational recommender system keeps retrieving products from the database and trying to gather a predefined number of undominated products. A high pruning rate usually indicates that the number of products retrieved is quite large. This involves more pairwise comparisons between products and so takes more time.

Table 2 gives the results of the experiments with the true preferences of the simulated users represented as cp-trees. The measures shown are averaged over 1,000 dialogues. A look at Table 2 shows that we can infer similar conclusions to the deductions made from results in Table 1. We can see that all the pruning rates are higher than the pruning percentages in Table 1 as well as the shortfall percentages. These differences can be explained by the nature of the user's true preferences and the way the user satisfaction is computed for both preference models (i.e., weights vectors and cp-trees). The shortfalls are all very small. It may be that *Basic* has the smallest shortfall (i.e., 0.187) in the second setting because it is the most cautious, i.e., it makes the weakest assumptions on the preferences.

Table 2. Averages (over 1,000) of the pruning rates, the computation time, the number of steps per dialogue and the shortfalls for each induction method and each combination of induction methods (users as cp-trees)

Induction methods	*Pruning (%)*	*Time (ms)*	*Steps*	*Fall (%)*
Basic	18.68	0.049	9.558	0.187
Lex-Basic	64.83	0.045	5.482	0.642
Every-Selected-Value	74.74	0.039	4.383	0.646
Cond-Selected-Value	1.28	0.015	11.211	0.640
Basic + Every-Selected-Value	74.74	0.048	4.383	0.646
Basic + Cond-Selected-Value	21.18	0.052	9.52	0.642
Lex-Basic + Every-Selected-Value	74.74	0.064	4.383	0.646
Lex-Basic + Cond-Selected-Value	65.19	0.046	5.435	0.641

7 Conclusions and Perspectives

Recommender systems are gaining momentum in the e-commerce applications market to face the "information overload" problem. This progressively reveals an increasing need to enable those recommender systems with suitable preference formalisms and dominance engines that can efficiently handle and reason with the user preferences while conversing with her (see, e.g., [10]). We specify new preference induction methods based on a recently developed preference language (i.e., comparative preference theories). We implemented these methods for a conversational recommender system to handle the user's preferences when recommending multi-valued feature products. We showed that these methods allow the system to capture preference nuances and various forms of preferences without giving up the attractive computational properties of the preference dominance relation.

As a continuation of this work, we will consider similar preference induction methods to be integrated with different critiquing-based recommender systems. Extending the conclusion to a more general scope, in the future, we intend to look for more elaborate and intuitive preference elicitation formalisms that we can prove

to be efficient in practice with conversational recommenders. These formalisms will adapt with the different dialogue strategies the conversational recommenders go through. They can be part of an intelligent query selection strategy to drive the elicitation process in the recommenders.

Acknowledgements

This material is partly supported by the Science Foundation Ireland under Grant No. 08/PI/I1912.

References

1. Zaslow, J.: If tivo thinks you are gay, here's how to set it straight. The Wall Street Journal (2002)
2. Fishburn, P.C.: Lexicographic orders, utilities, and decision rules: A survey. Management Science **20**(11) (1974) 1442–1471
3. Öztürk, M., Tsoukiàs, A., Vincke, P.: Preference modelling. In Bosi, G., Brafman, R.I., Chomicki, J., Kießling, W., eds.: Preferences. Volume 04271 of Dagstuhl Seminar Proceedings., IBFI, Schloss Dagstuhl, Germany (2004)
4. Stefanidis, K., Koutrika, G., Pitoura, E.: A survey on representation, composition and application of preferences in database systems. ACM Transactions on Database Systems **36**(3) (2011) 19
5. Bridge, D.G., Göker, M.H., McGinty, L., Smyth, B.: Case-based recommender systems. The Knowledge Engineering Review **20**(3) (2005) 315–320
6. Chen, L., Pu, P.: Survey of preference elicitation methods. In: Technical Report IC/200467. (2004)
7. McGinty, L., Reilly, J.: On the evolution of critiquing recommenders. In Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., eds.: Recommender Systems Handbook. Springer (2011) 419–453
8. McGinty, L., Smyth, B.: Comparison-based recommendation. In Craw, S., Preece, A.D., eds.: ECCBR. Volume 2416 of Lecture Notes in Computer Science., Springer (2002) 575–589
9. Bridge, D.G., Ricci, F.: Supporting product selection with query editing recommendations. In Konstan, J.A., Riedl, J., Smyth, B., eds.: RecSys, ACM (2007) 65–72
10. Trabelsi, W., Wilson, N., Bridge, D.G., Ricci, F.: Preference dominance reasoning for conversational recommender systems: a comparison between a comparative preferences and a sum of weights approach. International Journal on Artificial Intelligence Tools **20**(4) (2011) 591–616
11. Wilson, N.: Efficient inference for expressive comparative preference languages. In Boutilier, C., ed.: IJCAI. (2009) 961–966
12. Yaman, F., Walsh, T.J., Littman, M.L., desJardins, M.: Democratic approximation of lexicographic preference models. Artificial Intelligence **175**(7-8) (2011) 1290–1307
13. Shani, G., Gunawardana, A.: Evaluating recommendation systems. In: Recommender Systems Handbook. (2011) 257–297