Empowering Recommendation Technologies Through Argumentation

Carlos Iván Chesñevar, Ana Gabriela Maguitman and María Paula González

1 Introduction and motivations

User support systems have evolved in the last years as specialized tools to assist users in a plethora of computer-mediated tasks by providing guidelines or hints [19]. Recommender systems are a special class of user support tools that act in cooperation with users, complementing their abilities and augmenting their performance by offering proactive or on-demand, context-sensitive support. Recommender systems are mostly based on machine learning and information retrieval algorithms, providing typically suggestions based on *quantitative* evidence (i.e. measures of similarity between objects or users). The inference process which led to such suggestions is mostly unknown (i.e. 'black-box' metaphor). Although the effectiveness of existing recommenders is remarkable, they still have some serious limitations. On the one hand, they are incapable of dealing formally with the defeasible nature of users' preferences in complex environments. Decisions about user preferences are mostly based on heuristics which rely on ranking previous user choices or gathering information from other users with similar interests. On the other hand, they are not equipped with explicit inference capabilities. This is a hindrance for providing explanation facilities which could help the user to assess the analysis underlying the

Carlos Iván Chesñevar
CONICET (National Council of Technical and Scientific Research) – Department of Computer Science and Eng. – Universidad Nacional del Sur – Bahía Blanca, Argentina, e-mail: cic@cs.uns.edu.ar

Ana Gabriela Maguitman
CONICET (National Council of Technical and Scientific Research) – Department of Computer Science and Eng. – Universidad Nacional del Sur – Bahía Blanca, Argentina, e-mail: agm@cs.uns.edu.ar

María Paula González
CONICET (National Council of Technical and Scientific Research) – Department of Computer Science and Eng. – Universidad Nacional del Sur – Bahía Blanca, Argentina
GRIHO Research Group – University of Lleida – Lleida, Spain. e-mail: mpg@cs.uns.edu.ar

recommendations provided (i.e., which elements were taken into account in order to come up with a specific suggestion).

In fact, quantitative approaches in AI, as opposed to qualitative approaches, have often been criticized for their inability to obtain conclusions supported by a rationally justified procedure. The quantitative techniques adopted by most existing user support systems suffer also from this limitation. The absence of an underlying formal model makes it hard to provide users with a clear explanation of the factors and procedures that led the system to come up with some particular recommendations. As a result, serious trustworthiness issues may arise, especially in those cases when business interests are involved, or when external manipulation is possible. Logic-based approaches could help to overcome these issues, enhancing recommendation technology by providing a means to formally express constrains and to draw inferences. In this context, frameworks for defeasible argumentation [10, 25] constitute an interesting alternative for empowering recommendation technologies by providing appropriate inference mechanisms for qualitative reasoning. In fact, the argumentation paradigm has proven to be successful in a growing number of real-world applications such as multiagent systems [3, 6], legal reasoning [24], intelligent web-based forms [15], and semantic web [26, 9], among many others.

This chapter presents a generic approach to characterize *argument-based recommender systems*, *i.e.* user support tools in which recommendations are provided on the basis of arguments. The proposed approach is based on modelling user preference criteria by means of facts, strict rules and defeasible rules encoded in an argumentation formalism. These preference criteria are combined with additional background information and used by the argumentation framework to prioritize potential suggestions, thus enhancing the final results provided to the user. The rest of the chapter is structured as follows. Section 2 presents an overview of current recommender system technologies. Section 3 summarizes the main elements of Defeasible Logic Programming, a general-purpose argumentation formalism based on logic programming used for our proposal. Section 4 discusses our approach to empowering recommendation technologies through argumentation. Section 5 describes a particular application which emerged as an instance of this approach, oriented towards providing suitable decision support in the context of content-based web search. Finally, Section 6 discusses related work and Section 7 presents some conclusions and future research directions.

2 Recommendation Technologies: an overview

Recommendation systems are aimed at helping users to deal with the problem of information overload by facilitating access to relevant items. They attempt to generate a model of the user or user's task and apply diverse heuristics to anticipate what information may be of interest to the user. User support systems operate in association with the user to effectively accomplish a range of tasks. Some of these systems serve the purpose of expanding the user's natural capabilities, for example by acting

as intelligence or memory augmentation mechanisms [13]. Some of these systems reduce the user's work by carrying out the routinizable tasks on the user's behalf. Others offer tips on how to refine or complete human generated products (such as electronic documents) by highlighting potential inaccuracies and proposing alternative solutions, thus minimizing the user's cognitive effort. Some aides "think ahead" to anticipate the next steps in a user's task providing the capability for the user to confirm the prediction and ask the system to complete the steps automatically.

Recommender systems are a special class of user support tools that act in cooperation with users, complementing their abilities and augmenting their performance by offering proactive or on demand context-sensitive support. They usually operate by creating a model of the user's preferences or the user's task with the purpose of facilitating access to items (e.g., news, web pages, books, etc.) that the user may find useful. While in many situations the user explicitly posts a request for recommendations in the form of a query, many recommender systems attempt to anticipate the user's needs and are capable of proactively providing assistance. In order to come up with recommendations for user queries, conventional recommender systems rely on *similarity measures* between users or contents, computed on the basis of methods coming either from the information retrieval or the machine learning communities. Recommender systems adopt mainly two different views to help predict information needs. The first approach is known as *user modeling* and relies on the use of a profile or model of the users, which can be created by observing users' behavior (e.g., [20]). The second approach is based on *task modeling*, and recommendations are based on the context in which the user is immersed (e.g., [7]). The context may consist of an electronic document the user is editing, web pages the user has recently visited, etc.

Two main techniques have been used to compute recommendations: *content-based* and *collaborative filtering*. Content-based recommenders [23] are driven by the premise that user's preferences tend to persist through time. These recommenders frequently use machine-learning techniques to generate a profile of the active user. Typically, a model of the active user is stored as a list of rated items. In order to determine if a new item is a potentially good recommendation, content-based recommender systems rely on similarity measures between the new items and the rated items stored as part of the user model. On the other hand, recommender systems based on collaborative filtering [28] are based on the assumption that users' preferences are correlated. These systems maintain a pool of users' profiles associated with items that the users rated in the past. For a given active user, collaborative recommender systems find other similar users whose ratings strongly correlate with the current user. New items not rated by the active user can be presented as suggestions if similar users have rated them highly.

A combination of collaborative-filtering and content-based recommendation gives rise to *hybrid recommender systems* (e.g., [4]). Other combinations can be made resulting on other kinds of hybrid technologies. A survey of hybrid recommender systems can be found in [8]. Additional dimensions of analysis for recommender systems are the content of the suggestion (e.g., news, URLs, people, articles, text, products), the purpose of the suggestion (e.g., sales or information), the event that

triggers the search for suggestions (by user's demand or proactively), and the level of intrusiveness (none, low, moderate or high).

2.1 Limitations of current recommendation technologies

Although large amounts of qualitative data is available on the Web in the form of rankings, opinions, and other facts, this data is hardly used by existing recommenders to perform inference. Even quantitative data available on the Web could give rise to highly reliable and traceable suggestions if used by a system with the ability to perform qualitative inference on this data. An important deficiency of current recommendations technologies, therefore, is due to their inability to qualitatively exploit these data. This gives rise to a number of research opportunities for the development of a new generation of recommenders.

- **Exposing underlying assumptions.** Because the evidence used to provide suggestions is not traced or tracked, existing recommenders are unable to expose the underlying assumptions to careful scrutiny by the user. While recommendations in the form of simple pointers or hints may be useful in many situations, it is easy to come up with scenarios in which the user may need further evidence before taking a course of action based on a recommendation.
- **Dealing with the defeasible nature of users' preferences.** Users preferences are dynamic and typically change as time evolves or as new material becomes available for analysis. Because quantitative approaches are not equipped with mechanism to revise previous conclusions, the changing nature of users preferences is poorly dealt with. Modeling the dynamics of users preferences can help to keep the system up-to-date, without disregarding selections and decisions made by the user in the past.
- **Approaching trust and trustworthiness.** Recommendation technologies are increasingly gaining importance in commercial applications. However, most existing systems simply focus on tracking a customer's interests and make suggestions for the future without a contextualized justification. As a result the user is unable to evaluate the reasons that led the system to present certain recommendations. In certain domains (e.g., e-commerce), this is not sufficient, as this lack of justification can be associated with ulterior motives on the recommendation provider's side, leading to lack of confidence or reliability [22]. This emerging area requires a careful investigation of the notion of trust and trustworthiness.
- **Proving rationally compelling arguments.** The absence of a formal model underlying quantitative approaches makes it hard to provide users with a justification of why certain recommendations should be trusted or preferred. In many situations, more than one potential course of action could be proposed by a recommendation tool. However, the convenience of using these support tools is limited if no rational arguments for these suggestions are provided to help the user make a final decision.

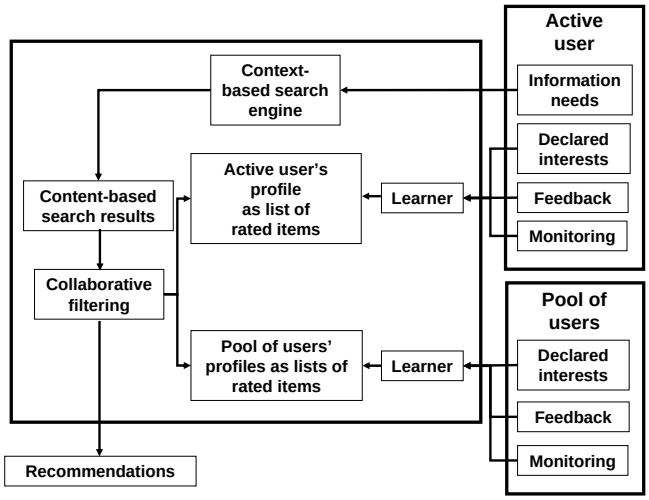

Fig. 1 A schematic view of a hybrid recommender system

- **Going beyond the basic collaborative model.** Because trust is to a great degree subjective, quantifying trust by combining measures coming from a pool of credibility assessments may not be entirely realistic. Although the "wisdom of the crowds" [29] is typically useful, it should be combined in a rational manner with information coming from individual users or communities.

A solution to some of these problems can be provided by integrating existing user support technologies with appropriate inferential mechanisms for qualitative reasoning. As we will see in the next sections, the use of argumentation will allow to enhance recommender systems with inference abilities to present reasoned suggestions, which the user will be able to further investigate and accept only if a convincing case can be made by the recommendation tool.

3 Defeasible Logic Programming in a Nutshell[1]

Defeasible logic programming (DeLP) [14] is a general-purpose defeasible argumentation formalism based on logic programming, intended to model inconsistent and potentially contradictory knowledge. A defeasible logic program has the form $\mathscr{P} = (\Pi, \Delta)$, where Π and Δ stand for *strict* and *defeasible* knowledge, respectively. The set Π involves *strict rules* of the form $P \leftarrow Q_1, \ldots, Q_k$ and *facts* (strict rules with empty body), and it is assumed to be *non-contradictory* (i.e., no complementary literals P and $\sim P$ can be inferred, where $\sim P$ denotes the contrary of P). The set Δ involves *defeasible rules* of the form $P \prec Q_1, \ldots, Q_k$, which stand for "$Q_1, \ldots Q_k$ *provide a tentative reason to believe* P." Rules in DeLP are defined in terms of *literals*. A literal is an atom A or the strict negation ($\sim A$) of an atom. Default negation (denoted not A) is also allowed in the body of defeasible rules (see [14] for details).

Deriving literals in DeLP results in the construction of *arguments*. An argument $\mathscr{A}$ for a literal Q (denoted $\langle \mathscr{A}, Q \rangle$) is a (possibly empty) set of ground defeasible rules that together with the set Π provide a proof for a given literal Q, satisfying the additional constraints of *non-contradiction* (*i.e.*, an argument should not allow the derivation of contradictory literals) and *minimality* (*i.e.*, the set of defeasible information used to derive Q should be minimal).[2] Note that arguments are obtained by a mechanism similar to the usual query-driven SLD derivation from logic programming, performed by backward chaining on *both* strict and defeasible rules; in this context a negated literal $\sim P$ is treated just as a new predicate name *no_P*. In DeLP, arguments provide tentative support for claims (literals). Clearly, as a program $\mathscr{P}$ represents incomplete and tentative information, an argument $\langle \mathscr{A}, Q \rangle$ may be *attacked* by other arguments also derivable from $\mathscr{P}$. An argument $\langle \mathscr{B}, R \rangle$ is a *counter-argument* for $\langle \mathscr{A}, Q \rangle$ whenever a subargument $\langle \mathscr{A}', Q' \rangle$ (with $\mathscr{A}' \subseteq \mathscr{A}$) in $\langle \mathscr{A}, Q \rangle$ can be identified, such that $\langle \mathscr{B}, R \rangle$ and $\langle \mathscr{A}', Q' \rangle$ cannot be simultaneously accepted since their joint acceptance would allow contradictory conclusions to be inferred from $\Pi \cup \mathscr{A}' \cup \mathscr{B}$. If the attacking argument $\langle \mathscr{B}, R \rangle$ is preferred over $\langle \mathscr{A}', Q' \rangle$, then $\langle \mathscr{B}, R \rangle$ is called a *defeater* for $\langle \mathscr{A}, Q \rangle$. The preference criterion commonly used is *specificity* [14], preferring those arguments which are more direct or more informed, although other criteria could be adopted.

In DeLP the search for defeaters for a given argument $\langle \mathscr{A}, Q \rangle$ prompts a recursive process, resulting in the generation of a *dialectical tree*: the root node of this tree is the original argument at issue, and every children node in the tree is a defeater for its parent. Additional restrictions help to avoid circular situations when computing branches in a dialectical tree, guaranteeing that every dialectical tree is finite (see [14] for details). Nodes in the tree can be marked either as *defeated* (*D*-nodes) or as *undefeated* (*U*-nodes). The marking of the dialectical tree is performed as in an AND-OR trees: leaves are always marked as undefeated nodes (as they have no defeaters); inner nodes can be be marked either as undefeated (if and only if every

[1] For an in-depth description of Defeasible Logic Programming the reader is referred to Chapter **??** in this book.

[2] This definition of argument was originally introduced by Simari & Loui [30].

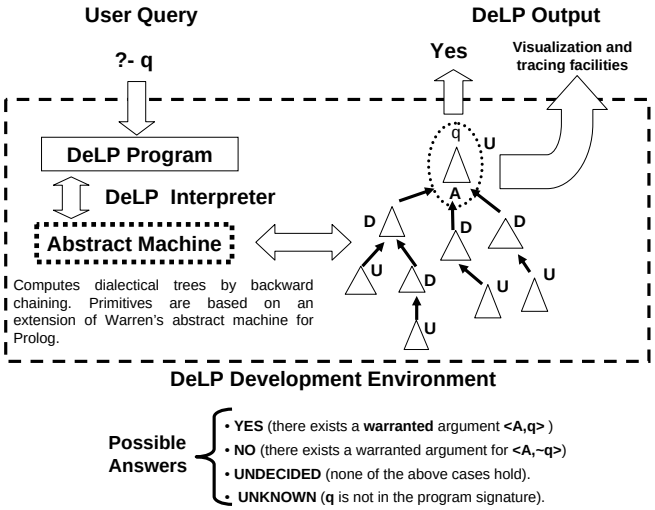

Fig. 2 A schematic view of the DeLP development environment

of its children nodes is marked as defeated) or as defeated (whenever at least one of its children has been marked as undefeated). The original argument $\langle \mathscr{A}, Q \rangle$ (the root of tree) is deemed as ultimately acceptable or *warranted* whenever it turns out to be marked as undefeated after applying the above process.

Given a DeLP program $\mathscr{P}$, solving a query Q with respect to $\mathscr{P}$ may result in four possible answers:

- YES (there is at least one warranted argument $\mathscr{A}$ for Q);
- NO (there is at least one warranted argument $\mathscr{A}$ for $\sim Q$);
- UNDECIDED (none of the previous cases hold); and
- UNKNOWN (Q is not present in the program signature).

The emerging semantics is skeptical, computed by DeLP on the basis of the goal-directed construction and marking of dialectical trees, which is performed in a depth-first fashion. Additional facilities (such as visualization of dialectical trees, zoom-in/zoom-out view of arguments, etc.) are integrated in the DeLP environment to facilitate user interaction when solving queries. The DeLP environment is available online at `http://lidia.cs.uns.edu.ar/delp_client`.

4 Argument-based Recommendation Technologies

We contend that argument-based reasoning can be integrated into recommender systems in order to provide a qualitative perspective in decision making. This can be achieved by integrating inference abilities to offer reasoned suggestions modelled in terms of arguments in favor and against a particular decision. This approach complements existing qualitative techniques by enriching the user's mental model of such computer systems in a natural way: suggestions are statements which are backed by arguments supporting them. Clearly, conflicting suggestions may arise, and it will be necessary to determine which suggestions can be considered as valid according to some rationally justified procedure. The role of argumentation is to provide a sound formal framework as a basis for such analysis.

In this context, our proposal is based on modelling users' preference criteria in terms of a DeLP program built on top of a traditional content-based search engine. Figure 4 depicts the basic architecture of a generic argument-based user support system based on DeLP. In such a setting users' preferences and background knowledge can be codified as facts, strict rules and defeasible rules in a DeLP program. These facts and rules can come from different sources. For example, user's preferences could be entered explicitly by the user or could be inferred by the system (e.g., by monitoring the user's behavior). Additional facts and rules could be obtained from other repositories of structured (e.g., databases) and semistructured data (e.g., the web.).

We will distinguish particular subsets in a DeLP program, representing different elements in a user support system. For example, a DeLP program could take the form $\mathscr{P} = \mathscr{P}_{user} \cup \mathscr{P}_{pool} \cup \mathscr{P}_{domain}$, where sets $\mathscr{P}_{user}$ and $\mathscr{P}_{pool}$ represent preferences and behavior of the active user and the pool of users, respectively. In the case of the active user, his/her profile can be encoded as facts and rules in DeLP. In the case of the pool of users, rule induction techniques are in order[3] resulting in defeasible rules characterizing trends and general preference criteria (e.g., *normally if a given user likes X then she also likes Y*). The set $\mathscr{P}_{domain}$ represents the domain (background) knowledge, encoded using facts and rules in DeLP. Either proactively or upon a user's request, an argument-based user support system triggers the search for suggestions. If needed, the collected results could be codified as facts and added to the DeLP program. Finally, a DeLP interpreter is in charge of performing the qualitative analysis on the program and to provide the final suggestions to the user.

Given the program $\mathscr{P}$, a user's request is transformed into suitable DeLP queries, from which different *suggestions* are obtained. For the sake of simplicity, we will assume in our analysis that user suggestions will be DeLP terms associated with a distinguished predicate name *rel* (which stands for *relevant* or *acceptable as a valid suggestion*). Using this formalization, suggestions will be classified into three sets, namely:

- S^w (warranted suggestions): those suggestions s_i for which there exists at least one warranted argument supporting $rel(s_i)$ based on $\mathscr{P}$;

[3] An approach for inducing defeasible rules from association rules can be found in [17].

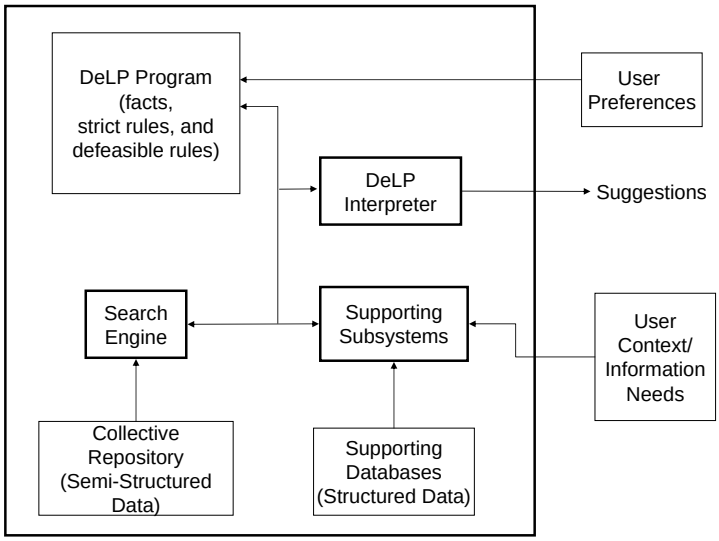

Fig. 3 A Generic Argument-Based User Support System based on DeLP

- S^u (undecided suggestions): those suggestions s_i for which there is no warranted argument for $rel(s_i)$, neither there is a warranted argument for $\sim rel(s_i)$ on the basis of $\mathscr{P}$, and
- S^d (defeated suggestions): those suggestions s_i such that there is a warranted argument supporting $\sim rel(s_i)$ on the basis of $\mathscr{P}$.

Given a potential suggestion s_i, the existence of a warranted argument $\langle \mathscr{A}_1, rel(s_i) \rangle$ built on the basis of the DeLP program $\mathscr{P}$ will allow to conclude that s_i should be presented as a final suggestion to the user. If results are presented as a ranked list of suggestions, then warranted suggestions will be more relevant than those which are undecided or defeated. Note that the above classification has a direct correspondence with the doxastic attitudes associated with answers to DeLP queries.

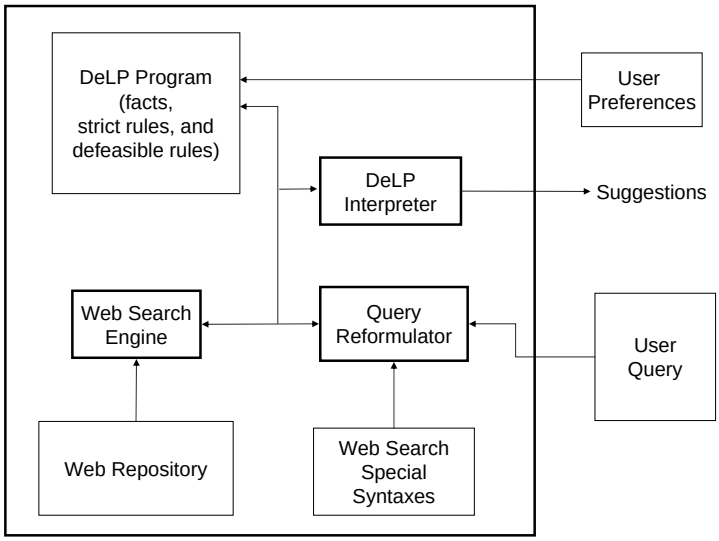

Fig. 4 A particular instance of the Generic Argument-Based User Support System for Content-based Search

5 Providing Argument-based User Support for Content-Based Web Search

We will present next a particular instantiation of our approach: an argument-based recommendation tool for content-based search queries [12]. In this context, the intended user support system aims at providing an enriched (content-based) web search engine which categorizes results, where the user's needs correspond to strings to be searched on the web. The underlying search engine may be a conventional search engine (e.g., GOOGLE), or a specialized content-based search engine. Final recommendation results for a query q are prioritized according to domain background knowledge and the user's declared preferences. Figure 4 illustrates the architecture of our argument-based recommender system.

Given a user query q, it will be given as an input to a traditional content-based web search engine, returning a list of search results L. If required, the original query q could be suitably re-formulated in order to improve the quality of the search results to be obtained. In the list L we can assume that s_i is a unique name characterizing a piece of information $info(s_i)$, in which a number of associated features (meta-

ALGORITHM Recommend_on_Query
INPUT: Query q, DeLP program $\mathscr{P} = \mathscr{P}_{user} \cup \mathscr{P}_{pool} \cup \mathscr{P}_{domain}$
OUTPUT: List L_{new} {*recommendation results wrt* $\mathscr{P}'$}
Let $L = [s_1, s_2, \ldots s_k]$ be the output of solving q
wrt content-based search engine *SE*
{L *is the list of (the first k) results obtained from query q via SE* }
$\mathscr{P}_{search}$ = {facts encoding $info(s_1)$, $info(s_2) \ldots info(s_k)$}
{$info(s_i)$ *stands for features associated with result* s_i }
$\mathscr{P}'$:= **Revise** ($\mathscr{P} \cup \mathscr{P}_{search}$).
{**Revise** *stands for a belief revision operator to ensure consistency in* $\mathscr{P}'$ }
Initialize S^w, S^u, and S^d as empty sets.
{S^w, S^u, *and* S^d *stand for the set of results* s_i*'s which are warranted as relevant, undecided and warranted as non-relevant, respectively* }
FOR EVERY $s_i \in L$
DO
Solve query $rel(s_i)$ using DeLP program $\mathscr{P}'$
IF $rel(s_i)$ is warranted **THEN** add s_i to S^w
ELSE
IF $\sim rel(s_i)$ is warranted **THEN** add s_i to S^d
ELSE add s_i to S^d
Return Recommendation $L_{new} = [s^w_1, s^w_2, \ldots, s^w_{j1}, s^u_1, s^u_2, \ldots, s^u_{j2}, s^d_1, \ldots, s^d_{j3}]$

Fig. 5 High-level algorithm for solving queries with argumentation support in content-based search

tags, filename, URL, etc.) can be identified. We assume that such features can be identified and extracted from $info(s_i)$ by some specialized tool, as suggested by Hunter [18] in his approach to dealing with structured news reports. Such features will be encoded as a set $\mathscr{P}_{search}$ of new DeLP facts, extending thus the original program $\mathscr{P}$ into a new program $\mathscr{P}'$. A special operator **Revise** deals with possible inconsistencies found in $\mathscr{P}_{search}$ with respect to $\mathscr{P}'$, ensuring $\mathscr{P} \cup \mathscr{P}_{search}$ is not contradictory.[4] Following the algorithm shown in Figure 5 we can now analyze L in the context of a new DeLP program $\mathscr{P}'$=$\mathscr{P} \cup Facts$, where $Facts$ denotes the set corresponding to the collection discussed above and $\mathscr{P}$ corresponds to domain knowledge and the user's preferences about the search domain.[5] For each s_i, the query $rel(s_i)$ will be analyzed in light of the new program $\mathscr{P}'$. Elements in the original list L of content-based search results will be classified into three sets of warranted, undecided, and defeated results. The final output presented to the user will be a sorted list L' in which the elements of L are ordered according to their epistemic status with respect to $\mathscr{P}'$. Figure 5 outlines a high level algorithm, which will be exemplified in the case study shown next.

[4] For example, contradictory facts may be found on the web. A simple belief revision criterion is to prefer the facts with a newer timestamp over the older ones.

[5] In this particular context, note that $\mathscr{P} = \mathscr{P}_{domain} \cup \mathscr{P}_{user}$.

5.1 A worked example

Example 1. Consider a tourist who wants to search for books about London. A content-based query *q* containing the terms *books*, *about*, *London* will return thousands of search results. The tourist performing the query may have some implicit knowledge to guide the search, such as:

1. She usually considers as relevant those books about London which are not outdated and are written by authors with good reputation.
2. She trusts a ranked list provided by the website `www.rankings.com` to assess the quality of a given author. Authors ranked less than 20 are considered trustworthy.[6]
3. According to the tourist's opinion, there is a particular author (John Doyle) which has a good reputation in books about London (independently of web-based rankings).
4. Usually all search results which include the keyword `'London'` are of interest, but not those involving `'Jack'` and `'London'` (as they are assumed to belong to the American writer Jack London, and not the city of London).
5. she usually likes books which are not expensive (not exceeding $ 300), except for those books published by Acme Inc. (in that case, she is willing to pay a higher price).
6. she usually considers non-relevant those books published by Boring Books Ltd.

Such rules and facts can be modelled in terms of a DeLP program $\mathscr{P}$ as shown in Figure 7. Note that some rules in $\mathscr{P}$ rely on "built in" predicates computed elsewhere and not provided by the user.

For the sake of example, let us suppose that the original query "*books about London*" returns a list of content-based search results L=[s_1, s_2, s_3, s_4]. Note that a traditional content-based search engine would present these results exactly in this order to the user, independently of the user's preferences or background knowledge. However, as discussed before, most of such results will be associated with XML or HTML pages, in which a number of attributes can be identified (e.g. author, date, URL, etc.). Such attributes can be encoded in a collection of DeLP facts as shown in Figure 6. We can now analyze s_1, s_2, s_3 and s_4 in the context of the user's preference theory about the search domain by considering the DeLP program $\mathscr{P}'$=$\mathscr{P}\cup$*Facts*, where *Facts* denotes the set corresponding to the collection of facts in Figure 6. For each s_i, the query $rel(s_i)$ will be analyzed wrt this new program $\mathscr{P}'$.

Consider the case for s_1. The search for an argument for $rel(s_1)$ returns the argument $\langle \mathscr{A}_1, rel(s_1)\rangle$: s_1 should be considered as a relevant item since it corresponds to a book about London written by a good author (John Doyle), and at a reasonable price. In this case we have the argument[7]

[6] It must be remarked that the ranked list mentioned in this example is fictitious, although websites such as `all-rankings.com` (a trademark of Lists & Ranks, S.L., Barcelona, Spain) allow to create such lists, making them available to others users. In this example the number 20 is an arbitrary threshold value, just for illustrative purposes.

[7] For the sake of clarity, semicolons to separate elements in an argument $\mathscr{A}$ = $\{e_1 ; e_2 ; \ldots ; e_k \}$.

$\mathscr{A}_1$= { $rel(s_1) \multimap goodauthor(s_1)$, $aboutlondon(s_1)$, $goodprice(s_1)$;
$goodauthor(s_1) \multimap author(s_1, \texttt{'john doyle'})$, $trust(\texttt{'john doyle'})$;
$aboutlondon(s_1) \multimap keywords(s_1, [\texttt{'london'}, \texttt{'history'}])$,
$member(\texttt{'london'}, [\texttt{'london'}, \texttt{'history'}])$;
$goodprice(s_1) \multimap \texttt{not}\ expensive(s_1)$ }.

Search for defeaters for argument $\langle \mathscr{A}_1, rel(s_1) \rangle$ will result in a proper defeater $\langle \mathscr{A}_2, \sim rel(s_1) \rangle$: s_1 is not relevant as it corresponds to an outdated book (more than 20 years old). In this case we have the argument

$\mathscr{A}_2$= { $\sim rel(s_1) \multimap goodauthor(s_1)$, $aboutlondon(s_1)$, $goodprice(s_1)$, $outdated(s_1)$;
$goodauthor(s_1) \multimap author(s_1, \texttt{'john doyle'})$, $trust(\texttt{'john doyle'})$;
$aboutlondon(s_1) \multimap keywords(s_1, [\texttt{'london'}, \texttt{'history'}])$,
$member(\texttt{'london'}, [\texttt{'london'}, \texttt{'history'}])$;
$goodprice(s_1) \multimap \texttt{not}\ expensive(s_1)$ }.

There are no other arguments to consider in the dialectical analysis. The dialectical trees for $\langle \mathscr{A}_1, rel(s_1) \rangle$ and $\langle \mathscr{A}_2, \sim rel(s_1) \rangle$ are shown in Figure 8a. As $\sim rel(s_1)$ is warranted, the item s_1 will be considered non-relevant.

Now consider the case for s_2. Let us assume that the author Joe Foo is ranked number 10 in `'www.rankings.com'`. Following a similar analysis as above, the search for an argument for $rel(s_2)$ returns the argument $\langle \mathscr{B}_1, rel(s_2) \rangle$, where

$\mathscr{B}_1$= { $rel(s_2) \multimap goodauthor(s_2)$, $aboutlondon(s_2)$, $goodprice(s_2)$;
$goodauthor(s_2) \multimap author(s_2, \texttt{'joe foo'})$, $trust(\texttt{'joe foo'})$;
$aboutlondon(s_2) \multimap keywords(s_2, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$,
$member(\texttt{'london'}, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$;
$goodprice(s_2) \multimap \texttt{not}\ expensive(s_2)$ }.

However, an argument $\langle \mathscr{B}_2, \sim rel(s_2) \rangle$ can be found defeating the previous argument: s_2 should not be considered as a relevant item since it seems to correspond to the writer Jack London, and not the city of London:

$\mathscr{B}_2$= { $rel(s_2) \multimap \sim aboutlondon(s_2)$;
$\sim aboutlondon(s_2) \multimap keywords(s_2, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$,
$member(\texttt{'london'}, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$,
$member(\texttt{'jack'}, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$;

Note that there exists a third argument $\langle \mathscr{B}_3, aboutlondon(s_2) \rangle$, where

$\mathscr{B}_3$= { $aboutlondon(s_2) \multimap keywords(s_2, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$,
$member(\texttt{'london'}, [\texttt{'london'}, \texttt{'jack'}, \texttt{'stories'}])$}

which would also attack $\langle \mathscr{B}_2, \sim rel(s_2) \rangle$. However, such argument is discarded in the dialectical analysis, since it is strictly *less specific* than $\langle \mathscr{B}_2, \sim rel(s_2) \rangle$ and hence cannot be considered as a defeater. There are no more arguments to consider, and consequently (as in the previous case), since $\sim rel(s_2)$ is warranted, the item s_2 can be deemed as non-relevant (see Figure 8b).

Now consider the case for s_3. Once again, let us assume that Jena Bar is an author ranked number 5 in `'www.rankings.com'`. There is an argument $\langle \mathscr{C}_1, rel(s_3)\rangle$ supporting the conclusion $rel(s_3)$, namely:

$$\begin{aligned}\mathscr{C}_1 = \{\ & rel(s_3) \prec goodauthor(s_3),\ aboutlondon(s_3),\ goodprice(s_3)\ ;\\ & goodauthor(s_3) \prec author(s_3, \texttt{'jena bar'}),\ trust(\texttt{'jena bar'});\\ & aboutlondon(s_3) \prec keywords(s_3, [\texttt{'london'}, \texttt{'tourism'}]),\\ & member(\texttt{'london'}, [\texttt{'london'}, \texttt{'tourism'}]);\\ & goodprice(s_3) \prec \texttt{not}\ expensive(s_3)\ \}.\end{aligned}$$

However, this argument is defeated by argument $\langle \mathscr{C}_2, expensive(s_3)\rangle$, since the price of this particular book is above $ 300.

$$\mathscr{C}_2 = \{\ expensive(s_3) \prec price(s_3, 350), 350 > 300.\ \}$$

Nevertheless, there is a third argument $\langle \mathscr{C}_3, \sim expensive(s_3)\rangle$ which attack the previous one, supporting the claim that this particular book is considered as relevant for the user, as it is published by ACME, which is an exceptional publisher.

$$\mathscr{C}_3 = \{\ \sim expensive(s_3) \prec price(s_3, 350), 350 > 300, publisher(s_3, acme).\ \}$$

There are no more arguments to consider, and consequently we have computed the dialectical tree for $\langle \mathscr{C}_1, rel(s_3)\rangle$ (see Figure 8c). According to the marking criterion introduced in Section 3, the root of the tree turns out to be an undefeated node, and hence $\langle \mathscr{C}_1, rel(s_3)\rangle$ is warranted and considered as a relevant information item.

Finally, let us consider the case for s_4. Let us assume that Tim Burton is an author ranked number 5 in `'www.rankings.com'`. As in the case of s_1, there is an argument $\langle \mathscr{D}_1, rel(s_4)\rangle$ supporting the fact that s_4 is a relevant item, since it corresponds to a book about London written by a good author and at a reasonable price.

$$\begin{aligned}\mathscr{D}_1 = \{\ & rel(s_4) \prec goodauthor(s_4),\ aboutlondon(s_4),\ goodprice(s_4)\ ;\\ & goodauthor(s_4) \prec author(s_4, \texttt{'tim burton'}),\ trust(\texttt{'tim burton'});\\ & aboutlondon(s_4) \prec keywords(s_4, [\texttt{'london'}, \texttt{'history'}]),\\ & member(\texttt{'london'}, [\texttt{'london'}, \texttt{'history'}]);\\ & goodprice(s_4) \prec \texttt{not}\ expensive(s_4)\ \}.\end{aligned}$$

However, it turns out that this book was edited by Boring Books Ltd., so that it should not be considered as relevant. Indeed, there is an argument $\langle \mathscr{D}_2, \sim rel(s_4)\rangle$ defeating the previous one

$$\mathscr{D}_2 = \{\ \sim rel(s_4) \prec publisher(s_4, \texttt{'boring books ltd'}).\ \}$$

It must be noted that $\langle \mathscr{D}_2, \sim rel(s_4)\rangle$ is a blocking defeater for $\langle \mathscr{D}_1, rel(s_4)\rangle$, and $\langle \mathscr{D}_1, rel(s_4)\rangle$ is a blocking defeater for $\langle \mathscr{D}_2, \sim rel(s_4)\rangle$, as both arguments cannot be compared by specificity. According to the marking criterion presented before, the conclusion $rel(s_4)$ is deemed as undecided, as neither $\langle \mathscr{D}_1, rel(s_4)\rangle$ nor $\langle \mathscr{D}_2, \sim rel(s_4)\rangle$ are warranted arguments (see Figure 8d).

From the previous analysis, and according to the high-level algorithm in Figure 5, we get the sets $S^w = \{s_3\}$, $S^d = \{s_1, s_2\}$, and $S^u = \{s_4\}$. Consequently, the system

author(s_1, 'John Doyle').
title(s_1, 'Everything about London').
publisher(s_1, 'peterbooks').
pubyear(s_1, 1960).
price(s_1, 20).
keywords(s_1, ['london', 'history']).
author(s_2, 'Joe Foo').
title(s_2, 'American Stories').
publisher(s_2, 'inkhouse').
pubyear(s_2, 2007).
price(s_2, 20).
keywords(s_2, ['london', 'jack', 'stories']).
author(s_3, 'Jena Bar').
title(s_3, 'London for tourists').
publisher(s_3, 'acme').
pubyear(s_3, 2007).
price(s_3, 250).
keywords(s_3, ['london', 'tourism']).
author(s_4, 'Tim Burton').
title(s_4, 'London History').
publisher(s_4, 'boring books ltd').
pubyear(s_4, 2007).
price(s_4, 20).
keywords(s_4, ['london', 'history']).

Fig. 6 Facts encoded from original content-based search results

will return the list L_{new}=[s_3, s_4, s_1, s_2] of final recommendation results, where users' preferences and background knowledge were taken into account, instead of just suggesting the original list of content-based search results L [s_1, s_2, s_3, s_4].

6 Related Work

As explained before, our approach for integrating argumentation in recommendation systems is based on modelling users' preference criteria in terms of a DeLP program built on top of a content-based search engine. Part of our recent research involved a variation of this schema in order to provide assessment on natural language usage using the web as a linguistic corpus [11]. The user preferences consist of a number of (possibly defeasible) rules and facts which encode different aspects of adequate language usage, defining the acceptability of different terms on the basis of so-called "usage indices", which are good indicators of the suitability of an expression on the

$$
\begin{aligned}
rel(B) &\prec goodauthor(B),\ aboutlondon(B),\ goodprice(B).\\
\sim rel(B) &\prec goodauthor(B),\ aboutlondon(B),\\
&\quad goodprice(B),\ outdated(B).\\
\sim rel(B) &\prec \sim aboutlondon(B).\\
\sim rel(B) &\prec publisher(B, \texttt{'boring books ltd'}).\\
goodauthor(B) &\prec author(B,A), trust(A).\\
trust(A) &\leftarrow queryranking(A, \texttt{'www.rankings.com'}, Rank),\\
&\quad Rank < 20.\\
trust(A) &\leftarrow A = \texttt{'John Doyle'}.\\
aboutlondon(B) &\prec keywords(B,K), member(\texttt{'london'}, K).\\
\sim aboutlondon(B) &\prec keywords(B,K), member(\texttt{'jack'}, K),\\
&\quad member(\texttt{'london'}, K).\\
goodprice(B) &\prec \texttt{not } expensive(B).\\
expensive(B) &\prec price(B,P), P > 300.\\
\sim expensive(B) &\prec price(B,P), P > 300, publisher(B, acme).\\
oudated(B) &\leftarrow pubyear(B,Y), getcurrentyear(CY),\\
&\quad (CY - Y) > 20.
\end{aligned}
$$

$$
\begin{aligned}
getcurrentyear(T) &\leftarrow [\text{Computed elsewhere}]\\
member(String, List) &\leftarrow [\text{Computed elsewhere}]\\
keywords(Book, List) &\leftarrow [\text{Computed elsewhere}]
\end{aligned}
$$

Fig. 7 DeLP program modelling user preferences about books

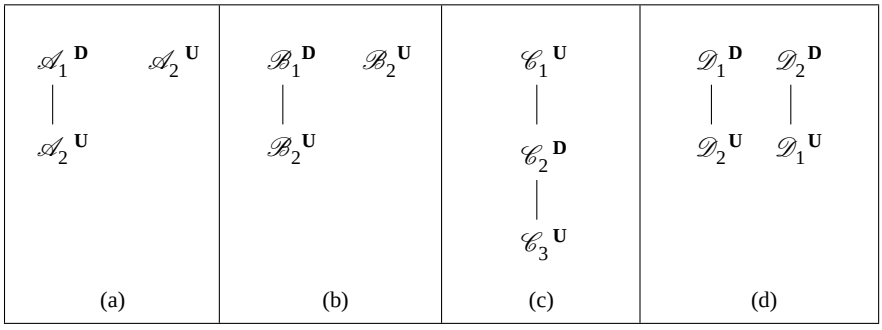

Fig. 8 Dialectical trees associated with (a) $\langle \mathscr{A}_1, rel(s_1)\rangle$ and $\langle \mathscr{A}_2, \sim rel(s_1)\rangle$; (b) $\langle \mathscr{B}_1, rel(s_2)\rangle$ and $\langle \mathscr{B}_2, \sim rel(s_2)\rangle$; (c) $\langle \mathscr{C}_1, rel(s_3)\rangle$ and (d) $\langle \mathscr{D}_1, rel(s_4)\rangle$ and $\langle \mathscr{D}_2, \sim rel(s_4)\rangle$

basis of the Web corpus. Argumentation is used to determine if a given expression is ultimately recommendable on the basis of a DeLP program which encodes the user's preferences.

Recommender systems can be seen as a particular instance of decision making systems oriented to assist users in solving computer-mediated tasks. In the last years there have been several efforts towards integrating argumentation in generic decision making systems. In [16] argumentation was applied in the context of modelling Shared Knowledge and Shared Knowledge Awareness when solving tasks collaboratively in a computer-supported collaborative learning (CSCL) environment. While Shared Knowledge (SK) refers to the common knowledge students acquire when they work in a collaborative activity, Shared Knowledge Awareness (SKA) is associated with the consciousness on the Shared Knowledge that a particular student has. In that context, DeLP was used to formalize an automated argumentation system as a support tool for characterizing SK and SKA in CSCL scenarios. Warranted arguments could be seen as suggestions provided by the argumentation system, providing a part of the SK among students, whereas visualization and explanation facilities provided by the argumentation system will help to make explicit the associated SKA.

In [34], an argument-based approach to modelling group decision making is presented, in which argumentation is used to support group decision task generation and identification. In contrast with our approach, the argumentation process is not automated, and the authors use argumentation for agreement among multiple users in a team, whereas we focus on argumentation for eliciting conclusions for a particular user on the basis of available information. In [31] a number of interesting *argument assistance tools* are presented. Even though there is a sound logical framework underlying this approach, the focus is rather restricted to legal reasoning, viewing the application of law as dialectical theory construction and evaluating alternative ways of representing argumentative data. In contrast, our analysis is oriented towards characterizing more generic argument-based user support systems.

Recent research has led to some interesting proposals to combine argumentation and machine learning techniques for rule induction. As discussed in Section 4, such combination lends itself applicable for extending our current approach towards a collaborative filtering setting, where defeasible rules would characterize trends and general preference criteria. Two recent research works are particularly relevant in this direction. In [21], a generic argumentation-based machine learning model (ABML) is proposed. This approach combines machine learning from examples with concepts from the field of argumentation. The idea is to provide expert's arguments, or reasons, for some of the learning examples. An improved argument-based rule learning was developed, which could be naturally integrated in the context of obtaining and justifying new rules which capture the knowledge associated with a pool of users. A different direction is adopted in the PADUA protocol [33], where a novel combination of argumentation and datamining is introduced in order to classify objects in a domain. Classification is the topic of a dialogue game between two agents, based on an argument scheme and critical questions designed to be used by agents whose knowledge of the domain comes from data mining. Each agent has its own set of examples which it can mine to find arguments based on association rules for and against a classification of a new instance. As in the case of ABML, this approach can help improve recommendation technologies in a collaborative setting,

allowing agents to perform argument-based reasoning from the database of cases corresponding to knowledge provided by users in the past.

During the last years there has been a growing attention to the development of the Semantic Web [5]. In particular, the integration of Semantic Web languages (such as OWL and RDF Schema) with argumentation has been recently explored with promising results by Iyad Rahwan *et al.*(*e.g.* [27, 26]). Their proposal involves an argumentation ontology (based on Walton's argumentation schemes [32]) which enables the representation of networks of arguments on the Semantic Web. Using a Semantic Web-based system called ARGDF, users can create arguments using different argumentation schemes and can query arguments using a suitable Semantic Web query language. Clearly, such integration can provide a powerful platform for the development of more evolved argument-based recommendation technologies, where arguments presented by others users can be taken into account for making decisions or presenting recommendations.

7 Conclusions. Future Directions

We have introduced a novel approach for enhancing recommendation technologies through the use of qualitative, argument-based analysis. As we have shown in this chapter, the argumentation formalism provided by Defeasible Logic Programming constitutes a powerful tool for carrying out this analysis when dealing with users' complex information needs. We performed some preliminary experiments on the integration of argumentation and recommendation technologies, which only served as a "proof of concept" prototype. We are currently carrying out more through evaluations in order to assess the full applicability of our proposal. Part of our current research in this context is related to combining quantitative and qualitative features in the argumentative analysis through the use of P-DeLP [1, 2], an extension of Defeasible Logic Programming which incorporates the treatment of possibilistic uncertainty at the object language level. In P-DeLP, in contrast to DeLP, arguments are attached with numerical values which determine their strength. Such values (necessity degrees) are assigned to the facts and rules present in the program, and propagated when performing inference via generalized modus ponens. In that respect, P-DeLP would be a natural tool for enhancing our current approach to argument-based recommendation, as such necessity degrees can be associated with rankings or user-assigned values.

We contend that the evolution of recommender systems will result in efficient and reliable content-based search environments, where both quantitative and qualitative analysis will play important roles. We believe our proposal is a realistic and doable approach to help fulfill this long-term goal.

Acknowledgements This research was funded by Agencia Nacional de Promoción Científica y Tecnológica (PICT 2005 - 32373), by CONICET (Argentina), by Projects TIN2006-15662-C02-01 and TIN2008-06596-C02-01 (MEC, Spain), and Project 24/ZN10 (SGCyT, UNS).

References

1. T. Alsinet, C. Chesñevar, L. Godo, S. Sandri, and G. Simari. Formalizing argumentative reasoning in a possibilistic logic programming setting with fuzzy unification. *Int. J. Approx. Reasoning*, 48(3):711–729, 2008.
2. T. Alsinet, C. Chesñevar, L. Godo, and G. Simari. A logic programming framework for possibilistic argumentation: Formalization and logical properties. *Fuzzy Sets and Systems*, 159(10):1208–1228, 2008.
3. L. Amgoud and M. Serrurier. Agents that argue and explain classifications. *Autonomous Agents and Multi-Agent Systems*, 16(2):187–209, 2008.
4. M. Balabanovic and Y. Shoham. Content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.
5. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.
6. R. Brena, J. Aguirre, C. Chesñevar, E. Ramírez, and L. Garrido. Knowledge and information distribution leveraged by intelligent agents. *Knowl. Inf. Syst.*, 12(2):203–227, 2007.
7. J. Budzik, K. Hammond, and L. Birnbaum. Information access in context. *Knowledge based systems*, 14(1–2):37–53, 2001.
8. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
9. C. Chesñevar, J. McGinnis, S. Modgil, I. Rahwan, C. Reed, G. Simari, M. South, G. Vreeswijk, and S. Wilmott. Towards an argument interchange format. *The Knowledge Engineering Review*, 21(4):293–316, 2006.
10. C. Chesñevar, A. Maguitman, and R. Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, Dec. 2000.
11. C. Chesñevar, A. Maguitman, and M. Sabaté. An argument-based decision support system for assessing natural language usage on the basis of the web corpus. *International Journal of Intelligent Systems (IJIS)*, 21(11):1151–1180, 2006.
12. C. Chesñevar, A. Maguitman, and G. Simari. Argument-Based Critics and Recommenders: A Qualitative Perspective on User Support Systems. *Journal of Data and Knowledge Engineering*, 59(2):293–319, 2006.
13. D. Engelbart. Augmenting human intellect: A conceptual framework. Summary report, Stanford Research Institute, on Contract AF 49(638)-1024, October 1962.
14. A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.
15. S. Gómez, C. Chesñevar, and G. Simari. Defeasible reasoning in web-based forms through argumentation. *International Journal of Information Technology and Decision Making (IJITDM)*, 7(1):71–101, Mar. 2008.
16. M. González, C. Chesñevar, C. Collazos, and G. Simari. Modelling shared knowledge and shared knowledge awareness in cscl scenarios through automated argumentation systems. In J. M. Haake, S. F. Ochoa, and A. Cechich, editors, *CRIWG*, volume 4715 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2007.
17. G. Governatori and A. Stranieri. Towards the application of association rules for defeasible rules discovery. In *Legal Know. & Inf. Sys.*, pages 63–75. JURIX, IOS Press, 2001.
18. A. Hunter. Hybrid argumentation systems for structured news reports. *Knowledge Engineering Review*, pages 295–329, 2001.
19. J. Konstan. Introduction to recommender systems: Algorithms and evaluation. *ACM Trans. Inf. Syst.*, 22(1):1–4, 2004.
20. F. Linton, D. Joy, and H. Schaefer. Building user and expert models by long-term observation of application usage. In *Proceedings of the seventh international conference on User modeling*, pages 129–138. Springer-Verlag New York, Inc., 1999.
21. M. Mozina, J. Zabkar, and I. Bratko. Argument based machine learning. *Artif. Intell.*, 171(10-15):922–937, 2007.
22. J. O'Donovan and B. Smyth. Trust in recommender systems. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174, New York, NY, USA, 2005. ACM.

23. M. Pazzani and D. Billsus. Content-based recommendation systems. pages 325–341. 2007.
24. H. Prakken and G. Sartor. The role of logic in computational models of legal argument: A critical survey. In A. C. Kakas and F. Sadri, editors, *Computational Logic: Logic Programming and Beyond*, volume 2408 of *Lecture Notes in Computer Science*, pages 342–381. Springer, 2002.
25. H. Prakken and G. Vreeswijk. Logical Systems for Defeasible Argumentation. In D. Gabbay and F.Guenther, editors, *Handbook of Phil. Logic*, pages 219–318. Kluwer, 2002.
26. I. Rahwan. Mass argumentation and the semantic web. *J. Web Sem.*, 6(1):29–37, 2008.
27. I. Rahwan, F. Zablith, and C. Reed. Laying the foundations for a world wide argument web. *Artif. Intell.*, 171(10-15):897–921, 2007.
28. J. Sandvig, B. Mobasher, and R. Burke. A survey of collaborative recommendation and the robustness of model-based algorithms. *IEEE Data Eng. Bull.*, 31(2):3–13, 2008.
29. R. Schenkel, T. Crecelius, M. Kacimi, T. Neumann, J. Parreira, M. Spaniol, and G. Weikum. Social wisdom for search and recommendation. *IEEE Data Eng. Bull.*, 31(2):40–49, 2008.
30. G. Simari and R. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence*, 53:125–157, 1992.
31. B. Verheij. Artificial argument assistants for defeasible argumentation. *Artif. Intell.*, 150(1-2):291–324, 2003.
32. D. Walton. *Argumentation Schemes for Presumptive Reasoning*. Erlbaum, Mahwah, NJ, 1996.
33. M. Wardeh, T. Bench-Capon, and F. Coenen. Arguments from experience: The padua protocol. In P. Besnard, S. Doutre, and A. Hunter, editors, *COMMA*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 405–416. IOS Press, 2008.
34. P. Zhang, J. Sun, and H. Chen. Frame-based argumentation for group decision task generation and identification. *Decision Support Systems*, 39:643–659, 2005.