# Argument-based User Support Systems using Defeasible Logic Programming

Carlos I. Chesñevar[1], Ana G. Maguitman[2], and Guillermo R. Simari[3]

[1] Artificial Intelligence Research Group – Department of Computer Science
Universitat de Lleida – C/Jaume II, 69 – E-25001 Lleida, SPAIN – Email: cic@eps.udl.es
[2] Computer Science Department – Indiana University
Bloomington, IN 47405-7104, USA – Email: anmaguit@cs.indiana.edu
[3] Department of Computer Science and Engineering – Universidad Nacional del Sur
Alem 1253, (8000) Bahía Blanca, ARGENTINA – Email: grs@cs.uns.edu.ar

**Abstract.** Over the last few years, argumentation has been gaining increasing importance in several AI-related areas, mainly as a vehicle for facilitating rationally justifiable decision making when handling incomplete and potentially inconsistent information. In this setting, user support systems can rely on argumentation techniques to automatize reasoning and decision making in several situations such as the handling of complex policies or managing change in dynamic environments. This paper presents a generic argument-based approach to characterize user support systems, in which knowledge representation and inference are captured in terms of Defeasible Logic Programming, a general-purpose defeasible argumentation formalism based on logic programming. We discuss a particular application which has emerged as an instance of this approach oriented towards providing user decision support for web search.

Keywords: argumentation, logic programming, user support systems, knowledge engineering

## 1 Introduction and motivations

Critics and recommender systems (commonly known under the general term *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 [8]. Most critics and recommenders are based on machine learning and information retrieval algorithms. The resulting systems typically provide suggestions based on *quantitative* evidence (i.e. measures of similarity between objects or users), whereas the inference process which led to these suggestions is commonly unknown (i.e. 'black-box' metaphor). Although the effectiveness of existing critics and recommenders is remarkable, they still have serious limitations as they are unable to perform qualitative inference on the suggestions they offer and are incapable of dealing with the defeasible nature of users' preferences. A solution for this problem can be provided by integrating existing user support technologies with appropriate inferential mechanisms for qualitative reasoning.

Please use the following format when citing this chapter:
Chesñevar, Carlos, Maguitman, Ana, Simari, Guillermo, 2006, in IFIP International Federation for Information Processing, Volume 204, Artificial Intelligence Applications and Innovations, eds. Maglogiannis, I., Karpouzis, K., Bramer, M., (Boston: Springer), pp. 61–69

In this context, *defeasible argumentation* frameworks [1, 10] constitute an interesting alternative, as they have matured in the last decade to become a sound setting to formalize commonsense, qualitative reasoning. In the last few years, particular attention has been given to extensions of logic programming as a suitable framework for formalizing argumentation in a computationally attractive way. One of such approaches that has been considerably successful is *Defeasible Logic Programming* (DeLP) [5], a general-purpose argumentation formalism based on logic programming.

This paper presents a generic approach to characterize *argument-based user support systems, i.e.* user support systems in which recommendations are provided on the basis of arguments. We describe a particular real-world application which emerged as an instance of this approach oriented towards providing suitable decision support in the context of web search.

## 2 Defeasible Logic Programming: overview

Defeasible logic programming (DeLP) [5] is a general-purpose defeasible argumentation formalism based on logic programming, intended to model inconsistent and potentially contradictory knowledge.[1] A defeasible logic program is a set $\mathcal{P} = (\Pi, \Delta)$ of Horn-like clauses, where $\Pi$ and $\Delta$ stand for sets of *strict* and *defeasible* knowledge, resp. The set $\Pi$ of strict knowledge 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*.[2] The set $\Delta$ of defeasible knowledge involves *defeasible rules* of the form $P \prec Q_1, \ldots, Q_k$, which stands for "$Q_1, \ldots Q_k$ *provide a tentative reason to believe* $P$." Strict and defeasible rules in DeLP are defined in terms of *literals* $P$, $Q_1$, $Q_2$, .... A literal is an atom or the strict negation ($\sim$) of an atom.

Deriving literals in DeLP results in the construction of *arguments*. An argument $\mathcal{A}$ for a literal $Q$ (denoted $\langle \mathcal{A}, Q \rangle$) is a (possibly empty) set of ground defeasible rules that together with the set $\Pi$ provide a SLD-like proof for a given literal $Q$, satisfying the additional requirements of *non-contradiction* (*i.e.*, an argument should not involve contradictory information) and *minimality* (*i.e.*, the set of defeasible information used should be minimal). 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*. As a program $\mathcal{P}$ represents incomplete and tentative information, *conflicting* arguments may arise. An argument $\langle \mathcal{B}, R \rangle$ is a *counterargument* for another argument $\langle \mathcal{A}, Q \rangle$ if if there exists a sub-argument $\langle \mathcal{C}, L \rangle$ of $\langle \mathcal{A}, Q \rangle$ (*i.e.*, $\mathcal{C} \subseteq \mathcal{A}$) such that there exists a literal $P \in \mathcal{P}$ verifying both $\Pi \cup \{L, R\} \vdash P$ and $\Pi \cup \{L, R\} \vdash \neg P$. Intuitively, this means that both arguments cannot be accepted simultaneously as they their joint acceptance leads to contradictory conclusions. A preference criterion among arguments " $\succeq$ " is used to determine when

[1] For space reasons, we will restrict ourselves to a basic set of definitions and concepts which make this paper self-contained. For more details, see [5, 1].

[2] Contradiction stands for deriving two complementary literals wrt strict negation ($P$ and $\sim P$) or default negation ($P$ and not $P$).

an argument is a ***defeater*** for another argument. An argument $\langle \mathcal{B}, R \rangle$ ***defeats*** another argument $\langle \mathcal{A}, Q \rangle$ if $\langle \mathcal{B}, R \rangle$ is a counterargument for $\langle \mathcal{A}, Q \rangle$ and $\langle \mathcal{B}, R \rangle \succeq \langle \mathcal{A}, Q \rangle$.

However, as defeaters are arguments, they may on its turn be defeated by other arguments, which could on their turn be defeated by other arguments, and so on. This prompts a recursive ***dialectical*** process rooted in a given argument $\langle \mathcal{A}_0, Q_0 \rangle$, considering all their defeaters, defeaters for such defeaters, and so on. The process can be characterized in a tree-like structure called ***dialectical tree*** $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$, in which nodes are arguments, the root node is the original argument at issue, and every children node defeats its parent node. Every path in a dialectical tree is a sequence $[\langle \mathcal{A}_0, Q_0 \rangle, \langle \mathcal{A}_1, Q_1 \rangle, \langle \mathcal{A}_2, Q_2 \rangle, \ldots, \langle \mathcal{A}_n, Q_n \rangle]$ that can be thought of as an exchange of arguments between two parties, a ***proponent*** (evenly-indexed arguments) and an ***opponent*** (oddly-indexed arguments).[3] Each $\langle \mathcal{A}_i, Q_i \rangle$ is a defeater for the previous argument $\langle \mathcal{A}_{i-1}, Q_{i-1} \rangle$ in the sequence, $i > 0$. A path is ***won*** by the proponent if its length is odd (***i.e.***, the last argument in the path was given by the proponent, and no defeater followed it); otherwise the path is ***lost***. An argument $\langle \mathcal{A}_0, Q_0 \rangle$ is ***warranted*** iff every path in $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$ is won. Given a DeLP program $\mathcal{P} = (\Pi, \Delta)$, a query $Q_0$ wrt $\mathcal{P}$ is solved by computing the preceding tree-like structure. Three answers are distinguished: YES (there is at least one warranted argument $\mathcal{A}_0$ for $Q_0$); NO (there is at least one warranted argument $\mathcal{A}_0$ for $\sim Q_0$); UNDECIDED (none of the previous cases hold).

## 3 Argument-based User Support Systems using DeLP

Our proposal is to model users' preference criteria in terms of a DeLP program built on top of a traditional content-based search engine. Figure 1(left) presents the basic architecture of a generic argument-based user support system based on DeLP. In this setting users preferences and background knowledge can be codified as facts and 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 $\mathcal{P} = \mathcal{P}_{user} \cup \mathcal{P}_{pool} \cup \mathcal{P}_{domain}$, where sets $\mathcal{P}_{user}$ and $\mathcal{P}_{pool}$ 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[4] 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 $\mathcal{P}_{domain}$ 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,

[3] Under certain constraints (e.g. avoiding cycles), all paths in a dialectical tree can be guaranteed to be finite. For details see [5].

[4] An approach for inducing defeasible rules from association rules can be found in [6].

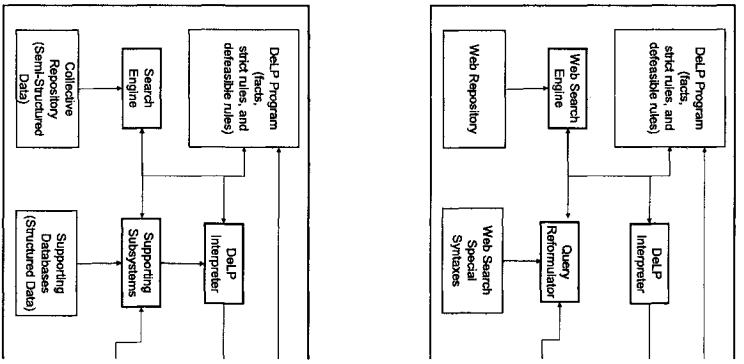


**Fig. 1.** A Generic Argument-Based User Support System based on DeLP (left); The ARGUENET Framework as a particular instance for argument-based web search (right)

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 $\mathcal{P}$, 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: (a) $S^w$ (warranted suggestions): those suggestions $s_i$ for which there exists at least one warranted argument supporting $rel(s_i)$ based on $\mathcal{P}$; (b) $S^u$ (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 $\mathcal{P}$, and (c) $S^d$ (defeated suggestions): those suggestions $s_i$ such that there is a warranted argument supporting $\sim rel(s_i)$ on the basis of $\mathcal{P}$. Given a potential suggestion $s_i$, the existence of a warranted argument $\langle \mathcal{A}_1, rel(s_i) \rangle$ built on the basis of the DeLP program $\mathcal{P}$ 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.

## 4 ARGUENET: Argument-based User Support for Web Search

Next, we will present a concrete instantiation of an argument-based user support system: a recommendation tool for web search queries called ARGUENET [2]. In this context, the intended user support aims at providing an enriched web search engine which categorizes results, and where the user's needs correspond to strings to be searched

**ALGORITHM** Recommend_on_Query
**INPUT:** Query $q$, DeLP program $\mathcal{P} = \mathcal{P}_{user} \cup \mathcal{P}_{pool} \cup \mathcal{P}_{domain}$
**OUTPUT:** List $L_{new}$ {*recommendation results wrt* $\mathcal{P}'$}
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* $SE$ }
$\mathcal{P}_{search}$ = {facts encoding $info(s_1)$, $info(s_2) \ldots info(s_k)$}
{$info(s_i)$ *stands for features associated with result* $s_i$ }
$\mathcal{P}'$ := **Revise** $(\mathcal{P} \cup \mathcal{P}_{search})$.
{**Revise** *stands for a belief revision operator to ensure consistency in* $\mathcal{P}'$ }
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 $\mathcal{P}'$
**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 $L_{new} = [s_1^w, s_2^w, \ldots, s_{j1}^w, s_1^u, s_2^u, \ldots, s_{j2}^u, s_1^d, \ldots, s_{j3}^d]$

**Fig. 2.** Algorithm for solving queries ARGUENET

on the web. The search engine is a conventional search engine (e.g., GOOGLE). Final recommendation results for a query $q$ are prioritized according to domain background knowledge and the user's declared preferences. Figure 1(right) illustrates the architecture of an argument-based news 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-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 [7] in his approach to dealing with structured news reports. Such features will be encoded as a set $\mathcal{P}_{search}$ of new DeLP facts, extending thus the original program $\mathcal{P}$ into a new program $\mathcal{P}'$. A special operator **Revise** deals with possible inconsistencies found in $\mathcal{P}_{search}$ with respect to $\mathcal{P}'$, ensuring $\mathcal{P} \cup \mathcal{P}_{search}$ is not contradictory.[5] Following the algorithm shown in Fig. 2 we can now analyze $L$ in the context of a new DeLP program $\mathcal{P}'$=$\mathcal{P} \cup Facts$, where $Facts$ denotes the set corresponding to the collection discussed above and $\mathcal{P}$ corresponds to domain knowledge and the user's preferences about the search domain.[6] For each $s_i$, the query $rel(s_i)$ will be analyzed in light of the new program $\mathcal{P}'$. 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

[5] 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.

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

$$
\begin{aligned}
rel(X) &\prec author(X,A), trust(A).\\
\sim rel(X) &\prec author(X,A), trust(A), outdated(X).\\
trust(A) &\prec \text{not } faked\_news(A).\\
\sim rel(X) &\prec address(X,Url), biased(Url).\\
biased(Url) &\prec thailandian(Url).\\
biased(Url) &\prec japanese(Url).\\
\sim biased(Url) &\prec domain(Url,D), D = \text{"}jpt.jp\text{"}.\\
rel(X) &\leftarrow author(X, bob\_beak).\\
oudated(X) &\leftarrow date(X,D), getdate(Today),\\
&\quad (Today - D) > 100.\\
thailandian(X) &\leftarrow \text{[Computed elsewhere]}\\
japanese(X) &\leftarrow \text{[Computed elsewhere]}\\
domain(Url,D) &\leftarrow \text{[Computed elsewhere]}\\
getdate(T) &\leftarrow \text{[Computed elsewhere]}\\
faked\_news(chin\_yao\_lin) &\leftarrow
\end{aligned}
$$

| |
|---|
| $author(s_1, chin\_yao\_lin)$. |
| $address(s_1, \text{"}jpt.jp/...\text{"})$. |
| $date(s_1, 20031003)$. |
| $author(s_2, jen\_doe)$. |
| $address(s_2, \text{"}news.co.uk/...\text{"})$. |
| $date(s_1, 20001003)$. |
| $author(s_3, jane\_truth)$. |
| $address(s_3, \text{"}jpt.jp/...\text{"})$. |
| $date(s_3, 20031003)$. |
| $author(s_4, bob\_beak)$. |
| $address(s_4, \text{"}mynews.com/...\text{"})$. |
| $date(s_4, 20031003)$. |

**Fig. 3.** (a) DeLP program modeling preferences of a journalist; (b) Facts encoded from original web search results

ordered according to their epistemic status with respect to $\mathcal{P}'$. Fig. 2 outlines a high level algorithm, which will be exemplified in the case study shown next.

*Example 1.* Consider a journalist who wants to search for news articles about recent outbreaks of bird flu. A query $q$ containing the terms *news*, *bird*, and *flu* will return thousands of search results. Our journalist may have some implicit knowledge to guide the search, such as: (1) she always considers relevant the newspaper reports written by Bob Beak; (2) she usually considers relevant the reports written by trustworthy journalists; (3) Reports written by trustworthy journalists which are out of date are usually not relevant; (4) Knowing that a journalist has not faked reports provides a tentative reason to believe he or she is trustworthy. By default, every journalist is assumed to be trustworthy. (5) Japanese and Thailandian newspapers usually offer a biased viewpoint on bird flu outbreaks; (6) The *"Japanese Times"* (http://jpt.jp) is a Japanese newspaper which she usually considers non biased; (7) Chin Yao Lin is known to have faked a report. Such rules and facts can be modelled in terms of a DeLP program $\mathcal{P}$ shown in Fig. 3(a). Note that some rules in $\mathcal{P}$ rely on "built in" predicates computed elsewhere and not provided by the user.[7]

For the sake of example, suppose that the above query returns a list of search results $L$=[$s_1$, $s_2$, $s_3$, $s_4$]. Most of these results will be associated with XML or HTML pages, containing a number of features (e.g. author, date, URL, etc.). Such features can be encoded as discussed before in a collection of DeLP facts as shown in Fig. 3(b). 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 $\mathcal{P}'$=$\mathcal{P}\cup Facts$, where $Facts$ denotes the set corresponding to the collection of facts in Fig. 3(b). For each $s_i$, the query $rel(s_i)$ will be analyzed wrt this new program $\mathcal{P}'$.

Consider the case for $s_1$. The search for an argument for $rel(s_1)$ returns the argument $\langle \mathcal{A}_1, rel(s_1)\rangle$: $s_1$ should be considered relevant since it corresponds to a newspaper article written by Chin Yao Lin who is considered a trustworthy author (note that every journalist is considered to be trustworthy by default.) In this case we have the argument[8] $\mathcal{A}_1$={ $rel(s_1) \prec author(c_1, chin\_yao\_lin)$, $trust(chin\_yao\_lin)$ ; $trust(chin\_yao\_lin) \prec$ not $faked\_news(chin\_yao\_lin)$ }. Search for defeaters for argument $\langle \mathcal{A}_1, rel(s_1)\rangle$ will result in a defeater $\langle \mathcal{A}_2, \sim rel(s_1)\rangle$: $s_1$ is not relevant as it comes from a Japanese newspaper,

[7] E.g., determining the country of origin corresponding to a specific web domain can be found querying Internet directory services such as WHOIS.

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

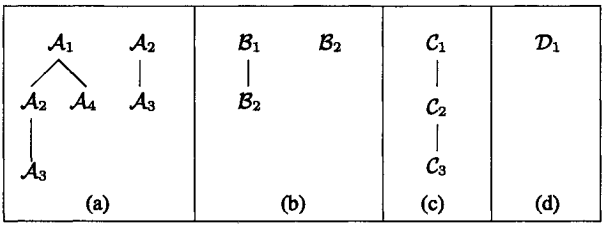


**Fig. 4.** Dialectical trees associated with (a) $\langle \mathcal{A}_1, rel(s_1) \rangle$ and $\langle \mathcal{A}_2, \sim rel(s_1) \rangle$; (b) $\langle \mathcal{B}_1, rel(s_2) \rangle$ and $\langle \mathcal{B}_2, \sim rel(s_2) \rangle$; (c) $\langle \mathcal{C}_1, rel(s_3) \rangle$ and (d) $\langle \mathcal{D}_1, rel(s_4) \rangle$

which is assumed to be biased about bird flu. In this case we have the argument $\mathcal{A}_2$={ $\sim rel(c_1) \prec address(c_1, \text{"jpt.jp..."})$, $biased(\text{"jpt.jp..."})$ ; $biased(\text{"jpt.jp..."}) \prec japanese(\text{"jpt.jp..."})$ }. Note that we also have an argument $\langle \mathcal{A}_3, \sim biased(\text{"jpt.jp..."}) \rangle$ which defeats $\langle \mathcal{A}_2, \sim rel(s_1) \rangle$: Usually articles from the "Japanese Times" are not biased. In this case we have $\mathcal{A}_3$={ $\sim biased(\text{"jpt.jp..."}) \prec domain(\text{"jpt.jp..."}, \text{"jpt.jp"}), (\text{"jpt.jp"} = \text{"jpt.jp"})$ }. Finally, another defeater for $\langle \mathcal{A}_1, rel(s_1) \rangle$ is found, namely $\langle \mathcal{A}_4, faked\_news(chin\_yao\_lin) \rangle$, with $\mathcal{A}_4 = \emptyset$. No other arguments need to be considered. The resulting dialectical tree rooted $\langle \mathcal{A}_1, rel(s_1) \rangle$ is shown in Fig 4a (left). Not all paths have odd length, and hence $\langle \mathcal{A}_1, rel(s_1) \rangle$ is not warranted. Carrying out a similar analysis for $\sim rel(s_1)$ results in the dialectical tree shown in Figure 4a (right). A similar situation results. There are no other candidate arguments to consider; hence $s_1$ is deemed as *undecided*.

The case of $s_2$ is analogous. The argument $\langle \mathcal{B}_1, rel(s_2) \rangle$ can be built, with $\mathcal{B}_1$={ $rel(s_2) \prec author(s_2,)$, $trust(jen\_doe)$ ; $trust(jen\_oldie) \prec$ not $faked\_news(jen\_doe)$ }. This argument is defeated by $\langle \mathcal{B}_2, \sim rel(s_2) \rangle$, with $\mathcal{B}_2$={$\sim rel(s_2) \prec author(s_2, jen\_doe)$, $trust(jen\_doe)$, $outdated(s_2)$ ; $trust(jen\_doe) \prec$ not $faked\_news(jen\_doe)$}. There are no more arguments to consider, and $\langle \mathcal{B}_1, rel(s_2) \rangle$ is deemed as non warranted ((Fig. 4b (left)). The analysis of $\sim rel(s_2)$ results in a single argument. Thus, its associated dialectical tree has a single node $\langle \mathcal{B}_2, \sim rel(s_2) \rangle$, the only possible path has an odd length, and it is *warranted*.

Following the same line of reasoning used in the case of $s_1$ we can analyze the case of $s_3$. An argument $\langle \mathcal{C}_1, rel(s_3) \rangle$ can be built supporting the conclusion $rel(s_3)$ (a newspaper article written by Jane Truth is relevant as she can be assumed to be a trustworthy author). A defeater $\langle \mathcal{C}_2, \sim rel(s_3) \rangle$ will be found: $s_1$ is not relevant as it comes from a Japanese newspaper, which by default is assumed to be biased about bird flu. But this defeater in its turn is defeated by a third argument $\langle \mathcal{C}_3, biased(s_3) \rangle$. The resulting dialectical tree for $\langle \mathcal{C}_1, rel(s_3) \rangle$ is shown in Fig. 4c (left)). The original argument $\langle \mathcal{C}_1, rel(s_3) \rangle$ can be thus deemed as *warranted*. Finally let us consider the case of $s_4$. There is an argument $\langle \mathcal{D}_1, rel(s_4) \rangle$ with $\mathcal{D}_1 = \emptyset$, as $rel(s_4)$ follows directly from the strict knowledge in $\mathcal{P}$. Clearly, there is no defeater for an empty argument (as no defeasible knowledge is involved). Hence $rel(s_4)$ is *warranted* (see dialectical tree in Fig. 4d).

Applying the criterion given in the algorithm shown in Fig. 2, the initial list of search results [$s_1$, $s_2$, $s_3$, $s_4$] will be shown as [$s_3$, $s_4$, $s_1$, $s_2$] (as $\langle \mathcal{C}_1, rel(s_3) \rangle$ and $\langle \mathcal{D}_1, rel(s_4) \rangle$ are warranted, $\langle \mathcal{A}_1, rel(s_3) \rangle$ is undecided and $\langle \mathcal{B}_2, \sim rel(s_2) \rangle$ is warranted (i.e., $s_2$ is warranted to be a non-relevant result).

## 5 Related work. Conclusions

Several kinds of user support systems that operate on top of Internet services have been proposed over the past years. In the case of web-based recommender systems (e.g. SurfLen [4], and Quickstep [9], among others) the usual approach involves taking into account the user's interests –either declared by the user or conjectured by the system– to rank or filter web pages. However such approaches differ from our proposal in that they do not attempt to perform a *qualitative* analysis to warrant recommendations. In [12] 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.

In this paper we have presented a novel approach towards the development of user support systems by enhancing recommendation technologies through the use of qualitative, argument-based analysis. In particular, we have shown that DeLP is a suitable computational tool for carrying on such analysis in a real-world application for intelligent web search, providing thus a tool for higher abstraction when dealing with users' information needs. Preliminary experiments on the use of ARGUENET were performed on the basis of a prototype. However, it must be remarked that these initial experiments only serve as a "proof of concept" prototype, as thorough evaluations are still being carried out. As performing defeasible argumentation is a computationally complex task, an abstract machine called JAM (Justification Abstract Machine) has been specially developed for an efficient implementation of DeLP [5], allowing to solve queries and computing dialectical trees very efficiently. The JAM provides an argument-based extension of the traditional WAM (Warren's Abstract Machine) for PROLOG. A full-fledged implementation of DeLP is available online,[9] including facilities for visualizing arguments and dialectical trees. Several other features leading efficient DeLP implementations have also been recently studied, in particular those related to comparing conflicting arguments by *specificity* [11] as a syntax-based preference criterion and pruning dialectical trees to speed up the argumentative inference procedure [3].

Current trends in user support system technologies show clearly that the combination of quantitative and qualitative analysis of user preferences will play a major role in the future. In this context, we think that defeasible argumentation techniques will constitute a powerful tool to make inference in user support systems more reliable and and user-friendly. Our approach intends to be a first step to reach this long-term goal.

**Acknowledgements**: This research work was supported by Projects TIC2003-00950, TIN 2004-07933-C03-03, by Ramón y Cajal Program (MCyT, Spain) by CONICET (Argentina), and by Agencia Nacional de Promoción Científica y Tecnológica (PICT 2002 No. 13.096).

[9] See `http://lidia.cs.uns.edu.ar/DeLP`

## References


1. C. Chesñevar, A. Maguitman, and R. Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, December 2000.
2. C. Chesñevar, A. Maguitman, and G. Simari. Argument-Based Critics and Recommenders: A Qualitative Perspective on User Support Systems. *Data and Knowledge Engineering (to appear)*, 2005.
3. C. Chesñevar, G. Simari, and L. Godo. Computing dialectical trees efficiently in possibilistic defeasible logic programming. *LNAI Springer Series Vol. 3662 (Proc. of the 8th Intl. Conf. on Logic Programming and Nonmonotonic Reasoning LPNMR 2005)*, pages 158–171, September 2005.
4. Xiaobin Fu, Jay Budzik, and Kristian J. Hammond. Mining navigation history for recommendation. In *Intelligent User Interfaces*, pages 106–112, 2000.
5. A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.
6. 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.
7. Anthony Hunter. Hybrid argumentation systems for structured news reports. *Knowledge Engineering Review*, pages 295–329, 2001.
8. Joseph A. Konstan. Introduction to recommender systems: Algorithms and evaluation. *ACM Trans. Inf. Syst.*, 22(1):1–4, 2004.
9. S. Middleton, D. DeRoure, and N. Shadbolt. Capturing knowledge of user preferences: Ontologies in recommender systems. In *Proc. ACM K-CAP'01*, Canada, 2001. ACM Press.
10. 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.
11. F. Stolzenburg, A. García, C. Chesñevar, and G. Simari. Computing Generalized Specificity. *JANCL*, 13(1):87–113, 2003.
12. Bart Verheij. Artificial argument assistants for defeasible argumentation. *Artif. Intell.*, 150(1-2):291–324, 2003.