Recommender System based on Argumentation by Analogy

Paola D. Budán[1,2], Federico Rosenzvaig[1,2], Maximiliano C. D. Budán[1,2,3], and Guillermo R. Simari[2]

[1] Universidad Nacional de Santiago del Estero, Santiago del Estero, Argentina.
[2] Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA), Universidad Nacional del Sur, Bahía Blanca, Argentina.
[3] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET).

Abstract. Argumentation has contributed to the formalization of a reasoning model, similar to the human reasoning. In general, argumentation can be associated with the interaction of reasons in favour and against certain conclusions, so as to determine what conclusions are acceptable. A way of arguing in which the way in which the arguments are constructed, is Defeasible Logic Programming (DeLP); this is a formalism that combines logic programming and defeasible argumentation. This work focuses on the strengthening of the reasoning process, identifying partial connections or determinations between knowledge pieces. Through these relations, it is possible to increase the justifications and foundations that support a particular recommendation, by an analogy process.

Keywords: Argumentation, Defeasible Logic Programming, Determination, Analogy.

1 Introduction

Argumentation constitutes a study area of special interest in the field of Artificial Intelligence, mainly because it allows reasoning in environments in which it is possible to access to the information in a partial or potentially contradictory way. These characteristics make them particularly appropriate for their use in the implementation of the superior cognitive component of an autonomous agent [9]. Therefore, this type of reasoning becomes particularly attractive to be used in decision-making or specific recommenders [4, 2].

Argumentative systems based on rules (SABR) are those that consider the way in which the arguments are built. In these systems, there exists a set of inference rules with which, from a certain knowledge (antecedent or set of premises) new information (conclusions) can be inferred in a tentative way. In this type of systems, the rules are stored in a knowledge base, together with other information in the form of facts or presuppositions, that represent the evidence that the agent obtains from its environment. From this evidence, the agent can use the inference rules to construct arguments in favour or against a statement.

Once this has been done all the constructed arguments are evaluated and it is determined which ones of them are accepted, seeking to conclude if, from the agent's knowledge base, this statement can be accepted or not. These formalisms are non-monotonic since the introduction of new information to the system can generate new arguments that turn out to be contradictory with the ones already existing. Within the SABR, we will focus on the *Defeasible Logic Programming (DeLP)*. This is a formalism that combines Logic Programming and Defeasible Logic. This formalism permits the identification of pieces of information that are in contradiction, and through a dialectical process, to decide which of them prevail.

One of the fields of application with the greatest attention over the last years has been that of the *Recommender Systems (RS)*. The problem of justifying the recommendations that the system provides has been studied from Artificial Intelligence [16, 14] being present the necessity of attaining procedures that allow the system to be able to explain to the user why a particular recommendation is made.

In this work, a *RS* based on *DeLP* is proposed, on which a previous analysis to the knowledge base that will feed the *RS* so as to find different relationships of determination among the components will be carried out, to be then introduced into *DeLP* programme in the form of defeasible rules. Thus, it is possible to obtain a more complex reasoning process, taking information coming from past experiences to do an action in the present [3, 15, 6].

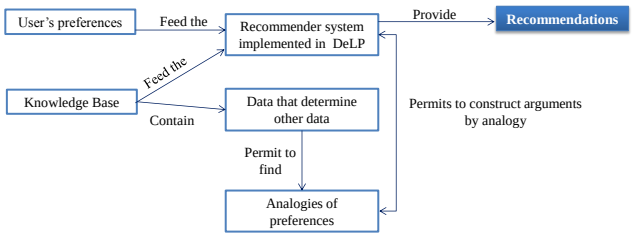

Fig. 1: Conceptual Relationships

This work is organized as follows: Section 2 presents *SR*; in Section 3, a brief introduction to the *Defeasible Logic Programming* is presented. Reasoning by analogy and the necessary conceptual elements to support it are introduced in Section 4; the incorporation of determination rules in *DeLP* are presented in Section 5. Finally, conclusions, future related works are outlined in Section 6.

2 Recommender Systems

Recommender systems (RS) have called a strong attention over the last years due to the fact they facilitate users the access to those relevant elements within enormous universe of possibilities available in these days. These systems are based on the users' preferences to determine particular elements, for example the sites they visit, information about films or music they like, among others. In other words, they make recommendations to the users from the information

about their preferences making a screening and facilitating them the search for objects of their interest.

The information required by this type of systems can be acquired in either an explicit or implicit way. In the first case, the users' assessments or audience ratings are used; and in the second, it is obtained from the monitoring of the users' behaviours, for instance the applications that they download or the web sites they visit. In the *RS*, the collaborative methods of screening gain importance, especially those that use the information coming from the social networks [1, 5].

In all *RS*, the transparency with which recommendations [16, 14], are provided is important so as to prevent the user from spending time on understanding why the system produces such an output. To meet such a need, argumentation supplies one of the alternatives by which *RS* provide explanations in their answers, presenting the arguments in favour of such answer [4, 2].

In this work, the combination of argumentation with the reasoning by analogy is proposed, analyzing in a deeper way the knowledge base of the *RS*, achieving a more complex reasoning process, and providing more refined results. Next, in the following sections, the main notions of *DeLP* and the reasoning by analogy are presented.

3 Defeasible Logic Programming (DeLP)

Defeasible Logic Programming (DeLP) is a formalism that combines Logic Programming and Defeasible Argumentation. *DeLP* arises as an extension of the programming in conventional logic, using concepts of defeasible argumentation, with the goal of capturing aspects of the common sense reasoning which is very difficult to express through the conventional logic programming. The defeasible logic programmes permit to express potentially inconsistent or incomplete information, being able to decide among contradictory goals by a particular preference criterion [8].

In *DeLP*, a defeasible logic programme is made up of: (1) a set of defeasible rules that represent incomplete or tentative information, they are represented as clauses of the form $L \prec P_1, ..., P_n$ with $n \geq 0$, where L is a literal and P_i is a literal (in the particular case that n = 0, it will be denoted as $L \prec True$, and in this opportunity L is referred to as a presupposition) and (2) a set of strict rules that represent strict information, they are represented as clauses of the form $L \leftarrow P_1, ..., P_n$ with $n \geq 0$, where L is a literal and P_i is a literal (in the particular case that n = 0, it will be denoted as $L \leftarrow True$, and in this opportunity L is referred to as fact). It is important to take into consideration that the set of strict rules should be consistent, i.e. it should not contain contradictory rules since these represent definite information (indisputable). On the contrary, the set of defeasible rules can be or not consistent, as these rules represent tentative information.

Definition 1 (*Argument*) *Let h be a literal and $\mathcal{P} = (\Pi, \Delta)$ a defeasible logic programme. An argument for h is a pair $\langle A, h \rangle$, where A is a set of defeasible rules of Δ, such that:*

1. *There exists a defeasible derivation for h from $\Pi \cup A$.*
2. *$\Pi \cup A$ is non-contradictory, and*
3. *A is minimal, i.e there is no proper subset A' de A such that A' satisfies conditions (1) and (2).*

An argument $\langle B, q\rangle$ is a sub-structure of $\langle A, h\rangle$ if and only if, $B \subseteq A$.

DeLP incorporates the argumentation formalism to deal with contradictory knowledge. This formalism permits to identify knowledge pieces that are in contradiction and through a dialectical process, decide which of them is the one that prevails. Argument structures can interact in different ways through the conflict and defeat relationships.

Two arguments in conflict can be compared by different criteria that establish a preference order between them.

The presence of multiple defeaters for an argument produces a splitting up of argumentation lines, giving rise to a defeaters' tree that is called a dialectical tree. In this tree, each path from the root to a leaf corresponds to an argumentation line. Once the dialectal tree has been built, a process of marking a dialectical tree to determine the acceptability of a specific literal h is carried out.

3.1 DBI-DeLP framework

In certain real life applications, for instance the *RS*, a formalism capable of managing big quantities of data related to the object requiring to be recommended is useful, in addition to the information of the system users. Thus, instead of including such information directly in the *DeLP* program as facts, relational databases are used. The *DeLP* version, called *Database Integration for Defeasible Logic Programming* (*DBI-DeLP*) [7], makes an integration of *DeLP* with relational databases.

As the information stored in the database can be contradictory, it can not be included in set Π of the programme in *DeLP*; thus, the notion of presupposition is useful [8]. The tuples in the database are represented as a particular type of these presuppositions, called operative presupposition, which are literals of the form $pred(q_1, ...q_m) \prec true$. Finally, the programme *DeLP* is extended to include information in the form of operative presuppositions obtained from the databases. A *DBI-DeLP* programme adds to the basic elements of *DeLP* programme, a set Σ of operative pressupositions, associated to the registers of the data set used to create the arguments that support the recommendation. Formally:

Definition 2 (*DBI-DeLP Programme*) *Let $\boldsymbol{D}$ be $\boldsymbol{D} = \{D_1, ...D_n\}$ a set of databases, $\mathcal{P} = (\Pi, \Delta)$ a DeLP programme, X a set of all the predicates in the rules of $\mathcal{P}$. A DBI-DeLP programme $\mathcal{P}'$ is a triple (Π, Δ, Σ) where $\Sigma = OPset_x, D$ is the set of operative presumptions for $(X, \boldsymbol{D})$.*

4 Reasoning by Analogy

Reasoning by analogy solves a new problem by focusing on the solutions given to analogous or similar problems [3, 6]. This type of reasoning is non-deductive but plausible, and it implies considering the following elements: T is a target situation, (*target*), P a property or set of properties shared with other object or known situation S (*source*) and Q is a conclusion projecting from S over T. there exists a set of shared properties that permits projecting the conclusion, which is not derived systematically from the premises, but it constitutes a plausible recommendation, in the following way:

$$(\mathsf{P}(\mathsf{S}) \wedge \mathsf{Q}(\mathsf{S})) \wedge (\mathsf{P}(\mathsf{T}) \Rightarrow \mathsf{Q}(\mathsf{T}))$$

The main problem in this type of reasoning consists in how to warrant the conclusion, since this derives from the information that is not provided in the premises; an approximation is taking into account the similitude that exists between S and T. In order to find such similarities, it is useful to use the determination rules, that can be understood as the connection points between S and T.

Example 1 *Let's assume that the user likes the film The Lord of the Rings because he likes the director (past experience); furthermore, this director determines the genre (determination rule). Then, it is possible to recommend some other films that correspond to the same genre; in this case, the property in common that films seen and to be seen have is the genre.*

In conclusion, the main objective by reasoning through analogies is to find a correspondence between T and S. Thus, it is necessary to analize the databases containing information that refers to the users' preferences (data known) to achieve recommendations that consider reasoning by analogy using the determination rules.

4.1 Determination Rules and Reasoning by Analogy

Establishing an analogy relationship implies comparing two objects or situations in a particular domain, considering certain properties or aspects. We can say that to be considered analogous, all the objects with a property P also have a property Q, or none of them has it:

$$(\forall(x)\mathsf{P}(x) \Rightarrow \mathsf{Q}(x)) \vee (\forall(x)\mathsf{P}(x) \Rightarrow \sim\mathsf{Q}(x))$$

This presumption is enough to warrant the conclusion. In other words, P decides if Q is true for any situation $\mathsf{Q}(x)$. This type of dependences is called *total dependences* and they are difficult to verify. In other words, the set of properties that are shared between the known situation and the new situation, is the one that determines if the conclusion that is applied on the first is plausible of being projected to the second. This establishes the dependence relationship, that is defined in the following way [6]:

$(\forall x \forall y, F(x) = F(y) \Rightarrow G(x) = G(y))$, where F and G are functions.

F determines functionally the value of G,because the value assigned by F is associated with an only value assigned by G. It is possible to know if this is true without knowing exactly what value is associated with each instance of G, according to a particular value of F.

Example 2 *Let A and B be two films where:*

$$\frac{\begin{array}{c}\texttt{Rating}(A, High) \wedge \texttt{Rating}(B, High) \\ \texttt{Genre}(A, Comedy) \wedge \texttt{Genre}(B, Comedy) \\ \texttt{Tags}(A, Excellent) \wedge \texttt{Tags}(B, Excellent) \\ \texttt{Recommendable}(A)\end{array}}{\texttt{Recommendable}(B)}$$

Knowing the values of `Rating`*,* `Genre`*, and* `Tags` *of films A and B,and if the first is recommendable, it is possible to determine if film B is recommendable. Then P (set of shared properties) is said to determine Q (the conclusion). In symbols: $\mathsf{P} \succ \mathsf{Q}$*

If P implies Q then P determines Q, but it is not necessarily true the other way round. For example, `user` $\succ$ `rating` but it is not true that `rating` $\succ$ `user`. The first is true because a user assigns one and only one rating to a given film. Thus, to find warrants for the conclusions, it is possible to use determination rules. To this effect, from the point of view of logic, it is necessary to find, on one hand, the determination of the true or polarity value of an expression of the form "$\mathsf{P}(x)$ decide if $\mathsf{Q}(x)$", and on the other hand, the rules of functional determination. Being total determinations difficult to verify, it is convenient to introduce the concept of *partial determinations*, which are generalizations of the functional dependences and refer to the *probability or factor of determination* f in which two tuples chosen at random have the same values of determined attributes [13]. In symbols, $\mathsf{P} \succ_f \mathsf{Q}$.

According to [6], the proposal of using determination rules to reinforce the reasoning process is based mainly on the fact that:

- In some domains there is not a solid implicational theory. Therefore, the determination rules can generate expertise, from the training with appropriate examples to carry out a reasoning by analogy.
- Even though this theory is not available, it can be easier to deduce knowledge wondering what the factors intervening in the decision-making about Q, using determination rules.

The reasoning based on determinations can be added to a *DeLP* system. The programmer could add determination rules arising from a previous analysis of the knowledge base in the form of defeasible rules, helping in this way to form a grounded and complex defeasible logic programme.

5 Construction of a DeLP Programme from the study of Determination Rules

In this work we will focus on the construction of *RS* of movies. Starting from a knowledge base that is stored in a database movielens with 10.000.000 registers

[2]. Generally speaking, we propose the inclusion of a *Module* called *Partial Determinations* (from now *Module-D*) that studies the knowledge base with the objective of obtaining information to find determination rules between the elements of that knowledge base is proposed. As it is possible to see in figure 2,

Fig. 2: Module Partial Determinations and DeLP

the objective of *Module-D* is to find relationships between the data that permit to introduce determination rules in *DeLP*, in the form of defeasible rules that will express the partial determination between the data coming from the knowledge base. Thus, it is possible to make recommendations of movies through a reasoning by analogy.

The main advantage that this proposal offers is that of optimizing and supporting the obtaining of defeasible rules for a particular *DeLP* programme. The disadvantage lies in the difficulty in finding the partial determinations which are dependent on the domain, and which require a semantic interpretation.

5.1 Functioning of *Module–D*

The problem of finding multi-valued dependencies (*MVD*) among data in a database is constantly being studied, so as to propose efficient algorithm solutions [10–12]. It is important to take into account that there exists a multi-valued dependence when an attribute X implies multiple values of other attribute Y. *Module-D* functioning entails the following steps:

1. Analizing the Knowledge Data Base looking for existing semantic relationships;
2. Finding attributes X that imply more than a value for the attribute Y (MVD);
3. Determining what the probability of each determined attribute Y is.

In other words, analyzing the data from the Knowledge Database, *Module-D* finds that, for example, a director may direct more than a genre. This *MVD* is relevant for this RS to recommend movies given an user's preference for a director. This means that, if the user likes the movie for its director, he could like other movies of the genre that that director most frequently directs. Then, *Module-D* calculates the probabilities that a director directs each genre in the following way:

– It consults, for each director, the genre of each movie he directed:

```
{ VIEW ‘director_by_genre_view‘ AS
    select
        ‘md‘.‘director_id‘ AS ‘director_id‘,
        ‘mg‘.‘genre_id‘ AS ‘genre_id‘,
        count(‘mg‘.‘genre_id‘) AS ‘Cant‘
    from
        (‘movies_directors‘ ‘md‘
        join ‘movies_genres‘ ‘mg‘ ON ((‘md‘.‘movie_id‘ = ‘mg‘.‘movie_id‘)))
    group by ‘md‘.‘director_id‘ , ‘mg‘.‘genre_id‘
    order by ‘md‘.‘director_id‘ , ‘Cant‘ desc;}
```

– It obtains, for each director, the genre that he directs with greater probability:

```
{ insert into x(director_id,genre_id,Prob)
    select director_id, genre_id,Cant/total*100 as Prob
        from director_por_genero_view dpgv join
        (select director_id, sum(Cant) as total
            from director_por_genero_view
            group by director_id) c1 using (director_id);}
```

– It considers those genres directed by the director with a probability greater than $Prob$, where $Prob$ is a parameter that can be adjusted depending on the domain requirements.
– It generates a table to contain this information, where *dir* represents an id of director, and $Prob$ is the probability that a director directs that genre.

```
select max(prob) into pr from x where director_id = dir;
insert into _director_genre_probability(director_id,genre_id,Prob)
   select distinct director_id,genre_id,Prob from x where director_id = dir and prob = pr;
```

Module-D carries out this enquiry for each of the directors. The table generated (*director-genre-probability*) is used as a base to be able to make the determinations by director or by genre. On the basis of the information provided by *Module-D*, the determination rule director determines genre is obtained and will be incorporated to the set Δ, in the form of defeasible rule:

$$genreDirector(G, D) -\!\prec directorMovie(D, M).$$

According to this rule, the most probable genre that the director of the movie X directs is determined. In this way and carrying out different interactions through *Module-D* we can obtain the following *DeLP* programme that is shown below, made up of sets Δ of defeasible rules, $\Delta' \subseteq \Delta$ of determination rules, and the set Π of strict rules and facts.

$$\Pi = \left\{ \begin{array}{ll} goodRating(pulpf) & likesDirector(u, pulpf, tarantino) \\ genreDirector(crime, tarantino) & \sim goodRating(keystulsa) \end{array} \right\}$$

$$\Delta = \left\{ \begin{array}{ll} rec(M_1, U) -\!\prec likesGenre(U, M_1, G) & \sim rec(M_1, U) -\!\prec \sim likesGenre(U, M_1, G) \\ rec(M_1, U) -\!\prec likesDirector(U, M_1, D) & \sim rec(M_1, U) -\!\prec \sim likesDirector(U, M_1, D) \\ rec(M_1, U) -\!\prec likesActor(U, M_1) & \sim rec(M_1, U) -\!\prec \sim likesActor(U, M_1) \\ rec(M_1, U) -\!\prec goodRaiting(M_1) & \sim rec(M_1, U) -\!\prec \sim goodRaiting(M_1) \\ rec(M_1, U) -\!\prec goodScript(M_1) & \sim goodScript(M_1) -\!\prec \sim faithfullOriginal(M_1) \\ goodScript(M_1) -\!\prec faithfullOriginal(M_1) & \sim goodScript(M_1) -\!\prec \sim goodStory(M_1) \\ goodScript(M_1) -\!\prec goodStory(M_1) & likesGenre(U, M_1, G) -\!\prec genreDirector(G, D) \\ rec(M_2, U) -\!\prec likesGenre(U, M_1, G), & rec(M_2, U) -\!\prec likesDirector(U, M_1, D), \\ \quad likesGenre(U, M_2, G) & \quad likesDirector(U, M_2, D) \end{array} \right\}$$

$$\Delta' = \{\ genreDirector(G, D) -\!\prec likesDirector(U, M_1, D)\ \}$$

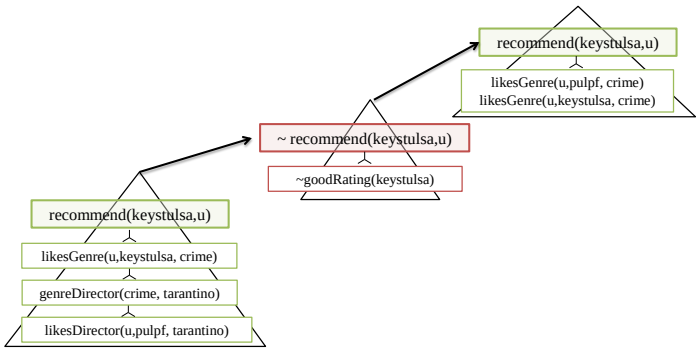

Fig. 3: Dialectical tree to recommend the movie Keys to Tulsa

When the DeLP reasoner is consulted about a particular movie for a specific user, the reasoner constructs the arguments in favour and against the mentioned recommendation from the *DeLP* programme and a dialectical tree is created as it is shown in figure 3.

On the basis of the dialectical tree presented, it is concluded that "*Keys to Tulsa* ″ will be recommended to the user since there are enough reasons that support such a conclusion.

6 Conclusions, Related and Future Works

Given a knowledge base, it is possible to find determinations among the data to build analogies. The concept of analogy, at the level of data manipulation, permits to make recommendations based on the information provided about the users' preferences. Therefore, it is necessary to find certain types of semantic relationships among the data like determinations. Once the determinations that allow for the construction of analogies are found, a reasoning designed finds arguments in favour of a determined recommendation.

In conclusion, obtaining determination rules that permit to model semantic connections within a knowledge base provides a reinforcement to the arguments used to support a determined recommendation.

The *RS* have been widely studied over the last years. Such is the case of *ArgueNet* [4], a *RS* based on arguments to solve search consults on the web, classifying the results according to the preference criteria specified in a declarative way by the user. This work proposes an integrated framework to recommend results, based on defeasible argumentation, preserving the simplicity of the traditional web search engines.

In this work an initial proposal is presented. Therefore, it is expected that it will continue with a formalization and a more detailed scheme of *Module-D* to be developed and implemented as an associated module with *DeLP*. Such formalisms will be applied in different domains, and the results obtained will be analyzed.

References

1. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowledge-Based Systems 46, 109–132 (2013)
2. Briguez, C.E., Budán, M.C., Deagustini, C.A., Maguitman, A.G., Capobianco, M., Simari, G.R.: Argument-based mixed recommenders and their application to movie suggestion. Expert Systems with Applications 41(14), 6467–6482 (2014)
3. Carbonell, J.: Learning by analogy: Formulating and generalizing plans from past experience. Springer (1983)
4. Chesnevar, C.I., Maguitman, A.G.: Arguenet: An argument-based recommender system for solving web search queries. In: Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference. vol. 1, pp. 282–287. IEEE (2004)
5. Chesnevar, C.I., Maguitman, A.G., Simari, G.R.: A first approach to argument-based recommender systems based on defeasible logic programming. In: NMR. pp. 109–117 (2004)
6. Davies, T.R., Russell, S.J.: A logical approach to reasoning by analogy. Tech. rep., DTIC Document (1987)
7. Deagustini, C.A., Fulladoza Dalibón, S.E., Gottifredi, S., Falappa, M.A., Chesñevar, C.I., Simari, G.R.: Relational databases as a massive information source for defeasible argumentation. Knowledge-Based Systems 51, 93–109 (2013)
8. García, A., Simari, G.: Defeasible logic programming: An argumentative approach. Theory and practice of logic programming 4(1+ 2), 95–138 (2004)
9. Garcıa, A.J., Gollapally, D., Tarau, P., Simari, G.R.: Deliberative stock market agents using jinni and defeasible logic programming. In: Proceedings of esaw00 engineering societies in the agents world, workshop of ecai 2000 (2000)
10. Lakshmanan, V., Veni Madhavan, C.: An algebraic theory of functional and multi-valued dependencies in relational databases. Theoretical Computer Science 54(1), 103–128 (1987)
11. Link, S.: Characterisations of multivalued dependency implication over undetermined universes. Journal of Computer and System Sciences 78(4), 1026–1044 (2012)
12. Lopes, S., Petit, J.M., Lakhal, L.: Efficient discovery of functional dependencies and armstrong relations. In: Advances in Database TechnologyEDBT 2000, pp. 350–364. Springer (2000)
13. Pfahringer, B., Kramer, S.: Compression-based evaluation of partial determinations. In: KDD. pp. 234–239 (1995)
14. Sinha, R., Swearingen, K.: The role of transparency in recommender systems. In: CHI'02 extended abstracts on Human factors in computing systems. pp. 830–831. ACM (2002)
15. Sowa, J., Majumdar, A.: Analogical reasoning. In: Conceptual Structures for Knowledge Creation and Communication, pp. 16–36. Springer (2003)
16. Tintarev, N., Masthoff, J.: Designing and evaluating explanations for recommender systems. In: Recommender Systems Handbook, pp. 479–510. Springer (2011)