A Group Recommendation System for Movies based on MAS

Christian Villavicencio, Silvia Schiaffino, J. Andrés Díaz-Pace, Ariel Monteserin

ISISTAN (CONICET - UNCPBA), Campus Universitario, Tandil, Argentina

{christian.villavicencio, silvia.schiaffino, andres.diazpace, ariel.monteserin}@isistan.unicen.edu.ar

KEYWORD

Multi-Agent Systems; Recommender Systems; Group Recommendation

ABSTRACT

Providing recommendations to groups of users has become popular in many applications today. Although several group recommendation techniques exist, the generation of items that satisfy all group members in an even way still remains a challenge. To this end, we have developed a multi-agent approach called PUMAS-GR that relies on negotiation techniques to improve group recommendations. We applied PUMAS-GR to the movies domain, and used the monotonic concession protocol to reach a consensus on the movies proposed to a group.

1. Introduction

Recommender systems provide assistance to users by identifying items that match a user's needs, preferences, and goals from a usually long list of potentially interesting items. Several recommendation techniques have been proposed in the literature (Ricci, et al., 2010). The aim of a group recommender system is to make item recommendations that are "good" for a group of users as a whole, i.e., the items satisfy, as much as possible, the individual preferences of each group member (Jameson & Smyth, 2007).
Group recommendation brings new challenges, since users might have competing interests within a group, and thus issues beyond individual recommendation have to be considered. In the literature we can see that most approaches developed to produce group recommendations usually rely on aggregation techniques for: (i) the generation of a group profile combining individual profiles (Christensen & Schiaffino, 2014); (ii) the integration of recommendations obtained for each member separately, such as in ranking aggregation (Baltrunas, et al., 2010); or (iii) the aggregation of individual ratings using, for example, approaches such as minimizing misery or maximizing average satisfaction. The problem with this kind of approaches is that the aggregation techniques often fail to satisfy the whole group in an even way and there is still no agreement regarding how to assess the utility of recommendations (Baltrunas, et al., 2010; Masthoff, 2011).

Other authors have applied MAS to recommendation systems both for individuals and groups. Some examples are the systems proposed in (Blanco-Fernandez, et al., 2004), (Skocir, et al., 2012), (Bekkerman, et al., 2006), (Garcia, et al., 2009), among others. However, particularly for group recommendation, there are not many systems and from those which do use MAS for generating group

©Ediciones Universidad de Salamanca / cc by-nc-nd

recommendations only one of them (Garcia, et al., 2009) avoids the use of aggregation techniques in the recommendation process.

In this work, we present a multi-agent approach, called PUMAS-GR, for group recommendation. The novelty of our approach is that it leverages on negotiation techniques in order to integrate recommendations (previously) obtained for each group member into a list of recommendations for the group. Each user is represented by a personal agent that works on her behalf. The agents carry out a cooperative negotiation process based on the multilateral Monotonic Concession Protocol (MCP) (Endriss, 2006). We argue that this negotiation process can generate recommendations that satisfy the different group members more evenly than traditional group recommendation approaches, since it mirrors the way in which human negotiation seems to work (Wooldridge, 2009). We have applied PUMAS-GR to the movies domain (MovieLens), but the approach is applicable to other domains as well.

The rest of the article is organized as follows. In Section 2 we present the details of PUMAS-GR. Then, in Section 3 we explain the negotiation process and depict the functionality of the application with an example. In Section 4 we describe some related works. Finally, in Section 5 we give the conclusions and outline some future work.

2. Proposed Approach

Our approach conceives the multi-agent system (MAS) as the group recommender system, according to the client-server architecture of *Figure 1*. The user interacts with a Web-based client, which can make different functional requests to a server, such as: log into a session, rate sequences of movies presented by the system, or ask for a group recommendation. The latter is what actually triggers the agent negotiation. On the server side, the *Group Recommender* hosts a collection of *Agent* instances along with a *Moderator* component. This *Moderator* is responsible for coordinating the agents according to the MCP rules the MCP. Information about user credentials, membership to different groups, and movies watched by users are stored in the *User Profiles* repository. Information about available movies for recommendation are kept in a separate repository. The Movies Dataset contains data from MovieLens[1].

Each *Agent* is a process that implements a number of negotiation commands, which are enacted by the *Moderator*. The negotiation commands refer to three aspects:

(i) computation of the agent utility function, which is used for determining agreements;
(ii) computation of the agent "willingness" to risk a conflict, and
(iii) the concession strategy (e.g., Nash, egocentric), in case the Moderator decides that the agent must concede.

Furthermore, each agent is able to generate a ranking of movies of interest for its associated user. This ranking only contains movies that the user has not watched before. Internally, each agent relies on a basic (single-user) recommender system that generates the rankings (the instance of the recommender is shared between the agents). To do so, we relied on the Duine framework[2], as it provides predefined prediction techniques for estimating movie scores. These techniques use item and user similarity models to feed predictors, which are then able to estimate the rating a user would have given to a movie, using

[1] http://grouplens.org/datasets/movielens/
[2] http://www.duineframework.org/

©Ediciones Universidad de Salamanca / cc by-nc-nd

information from the user profile (e.g., looking for similar users and assessing the ratings they have given to the movie) and information about movies she rated in the past (e.g., assessing the similarity between those movies and the target movie).

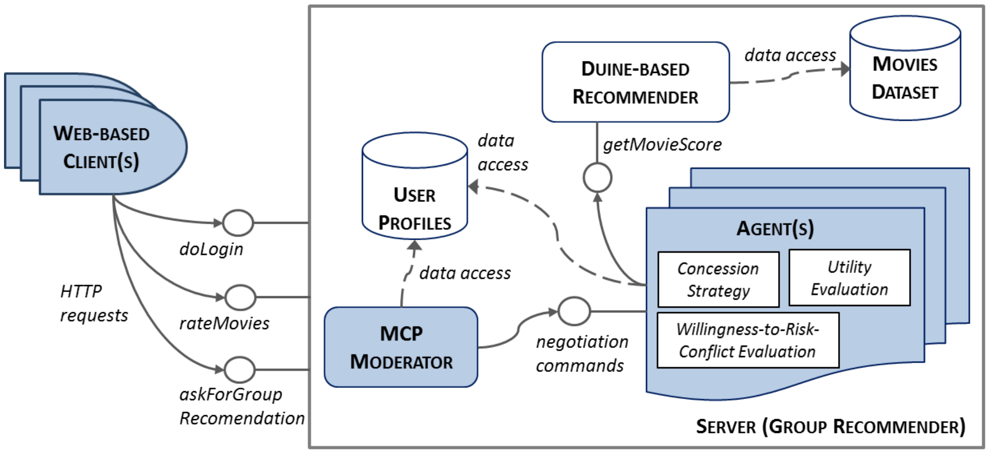

Figure 1: Architecture of PUMAS-GR.

3. PUMAS-GR application at work

In this section we firstly explain the negotiation process carried out by the agents when PUMAS-GR is asked to produce a group recommendation, and then we propose a usage example of the prototype of the tool.

3.1. Negotiation process

At the beginning, each agent makes an initial proposal with its favorite (top-ranked) movie, which is the movie with the highest score (step 1 of Figure 2). Then, proposals are interchanged among the agents in order to determine if an agreement can be reached. The notion of agreement is defined in terms of the utility of a given proposal for the agents. To do so, each agent computes a utility function that maps agreements to non-negative values. If the user already watched a given movie, then she probably assigned a score (utility) to it. If a user did not rate (or watched) a movie, it is possible to compute an estimated utility via Duine. Specifically, the utility is the product of the prediction score for the movie and the certainty of that prediction. There is an agreement if one agent makes a proposal that is at least as good (regarding utility) for any other agent as their own current proposals. If so, the proposal that satisfies all the agents is chosen (if several proposals meet this criterion, the Moderator simply picks one of them randomly).

If an initial agreement is not possible, the agents engage in rounds of negotiation, each one making movie proposals that need to be assessed by the other agents, until an agreement is reached or the negotiation finishes with a conflict (step 2 of Figure 2). The agents abide by a set of predefined MCP

©Ediciones Universidad de Salamanca / cc by-nc-nd

rules, which specify the range of legal moves available at each agent at any stage of the negotiation process. These rules correspond to the negotiation commands discussed for Figure 1. In case a round of negotiation ends up in a conflict, one of the agents must make a concession (step 3). A concession means that an agent seeks an inferior proposal with the hope of reaching an agreement. If none of the agents can concede, the process finishes with no-agreement. Several concession strategies are possible (Endriss, 2006).

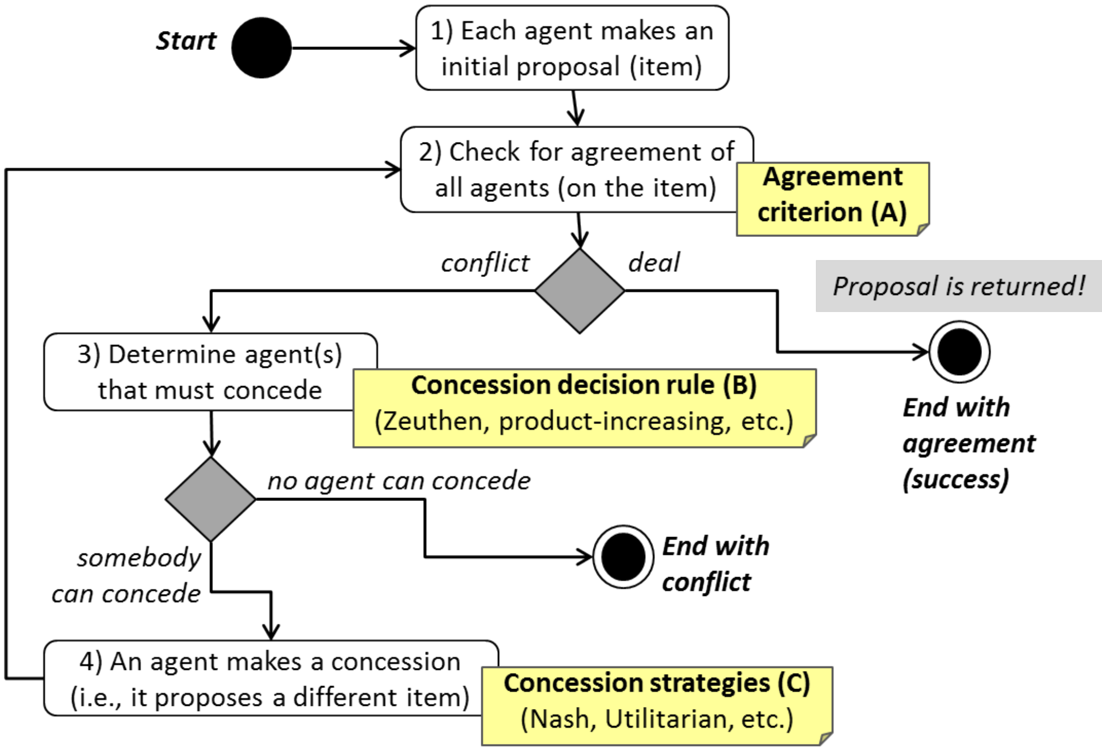

Figure 2: Negotiation Steps.

Selecting the agent(s) that must concede is determined by applying the Zeuthen strategy (Zeuthen, 1930) around the concept of willingness to risk conflict (WRC). In the bilateral MCP (i.e., two agents), both agents evaluate their WRC value and the agent with the lowest value makes the next concession. The strategy can be generalized to a multilateral setting (i.e., more than two agents), in which Zeuthen evaluates the loss in utility in case of concession assuming the worst possible outcome for the agent. As for the concession itself (i.e., the new proposal made by the agent/s determined by the Zeuthen generalization), various strategies are discussed in the literature (Endriss, 2006). For our work, we selected the so-called Nash concession, because it guarantees termination and deadlock-freedom. In this kind of concession, an agent makes a proposal such that the product of utilities of the other agents increases (Nash product).

3.2. Usage example

When using the tool for the very first time, the users should register in order to be able to log in.

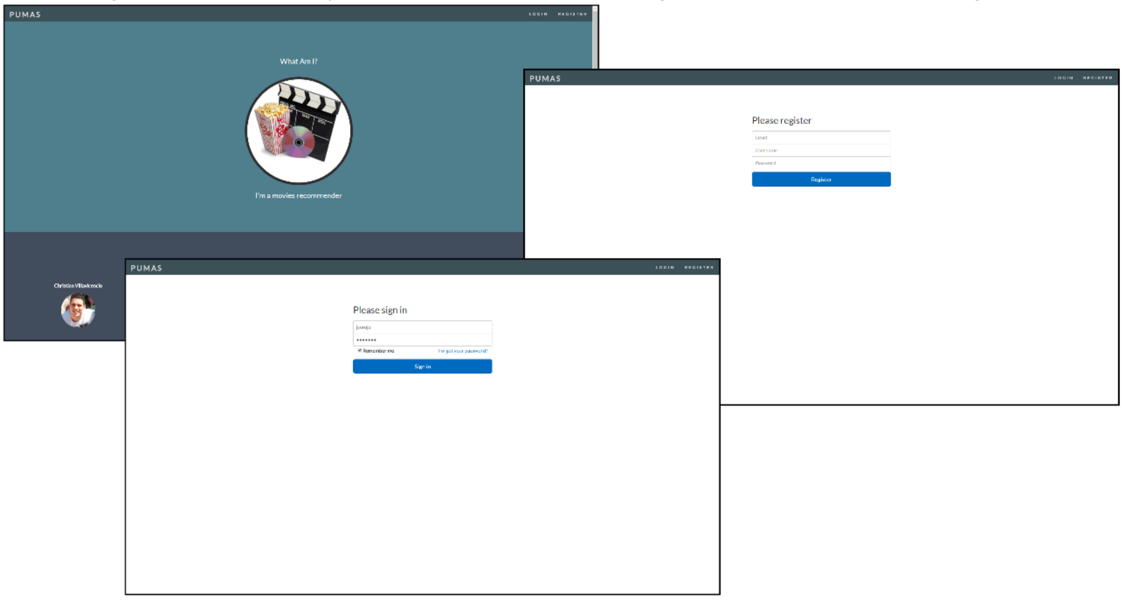

Figure 3: Welcome, Register and Login views.

The registration process only requests an email, a username and a password (Figure 3) (the users can later add some additional information to their profile, using the User Profile menu, accessible through the dropdown menu placed in the top-right corner, within the navigation bar). After a user registers himself, he is automatically logged in and:

- He can complete his user profile (adding more information like: name, surname, etc.)
- He is able to assign rating to movies he has watched in the past (Figure 4)
- He can revise which movies he has already rated, which were the ratings given to those movies, and also remove any of those ratings (so as to be able to rate the movies again) (Figure 4)
- He is able to revise the list of groups they belong to
- He, as a member of a group, can ask for a group recommendation (Figure 6)

In the following paragraphs we present a guideline that contains the basic steps that the users should follow if they want to generate a group recommendation using our tool.

Step 1: Create user's preferences models

When seeking to get a group recommendation the members of the group must build their preference model first. This can be achieved by rating at least 15 movies (Figure 4), including movies from different genres if possible so as to add variety to the preference model and allow the recommender system to produce recommendations that are closer to the user preferences. The number 15 is an empiricallydetermined parameter, but we consider that if the users rate less than 15 movies, the estimation of the preferences will not be good enough to produce acceptable recommendations.

Step 2: Create the group

The next step consists in creating a group using the group creator (Figure 5) which is accessible through the *Group Recommendation* view (Figure 6). There are two restrictions that the users must respect when creating groups: (i) every group must have a name, and (ii) every group must be composed of at least 1 member.

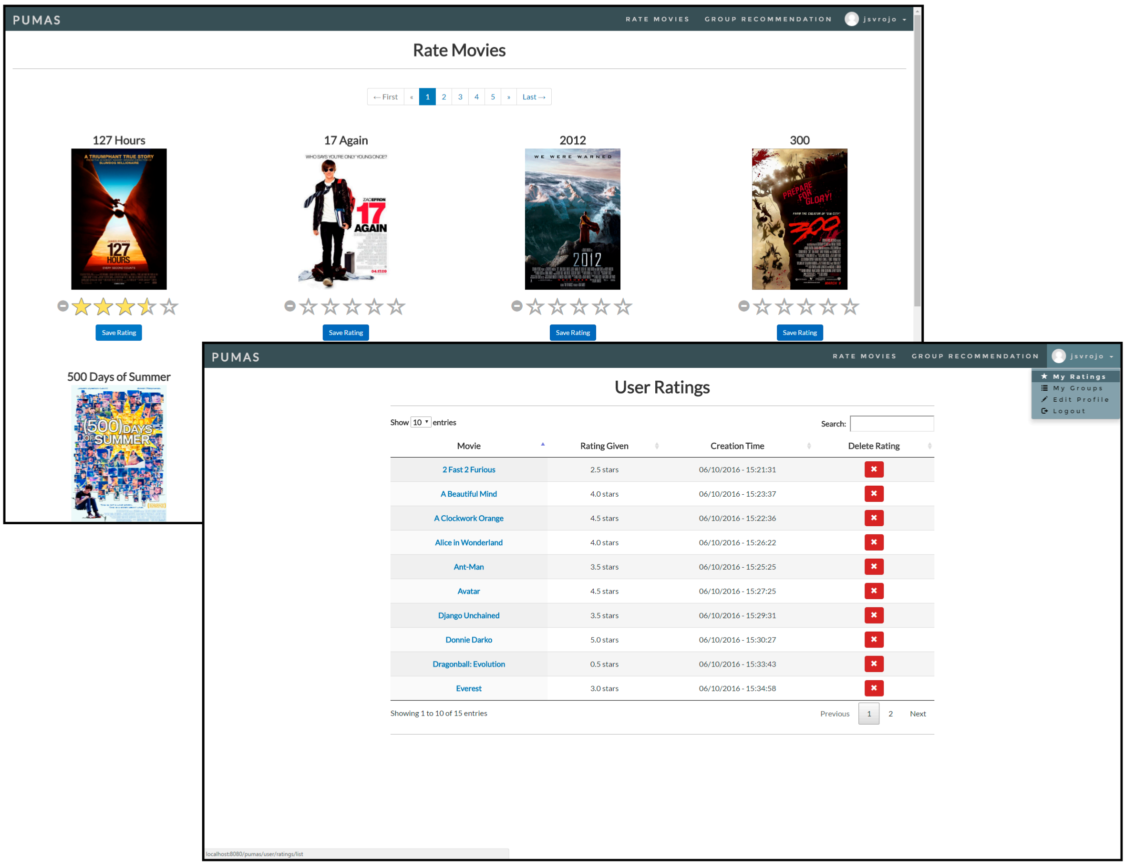

Figure 4: Rate movies and User Ratings views.

Step 3: Define the recommendation process parameters

Once the group was created, it is displayed in the *Group Recommendation* view and the user can use it to ask for a group recommendation. Additionally, the active user (the one who is going to ask for the group recommendation) must select the desired amount of recommendations (k) and the recommendation approach the application should use. Currently, the tool only allows the users to select between 2

©Ediciones Universidad de Salamanca / cc by-nc-nd

approches: the MAS-based approach (denoted by "PUMAS" in the recommender type selector of Figure 6) or the one based on aggregation techniques ("TRADGREC"). In the example of Figure 6 we can see that the active user is already part of 2 groups and he selected the first of them for the recommendation process, and he wants the application to produce 10 recommendations ($k = 10$) using the MAS-based approach.

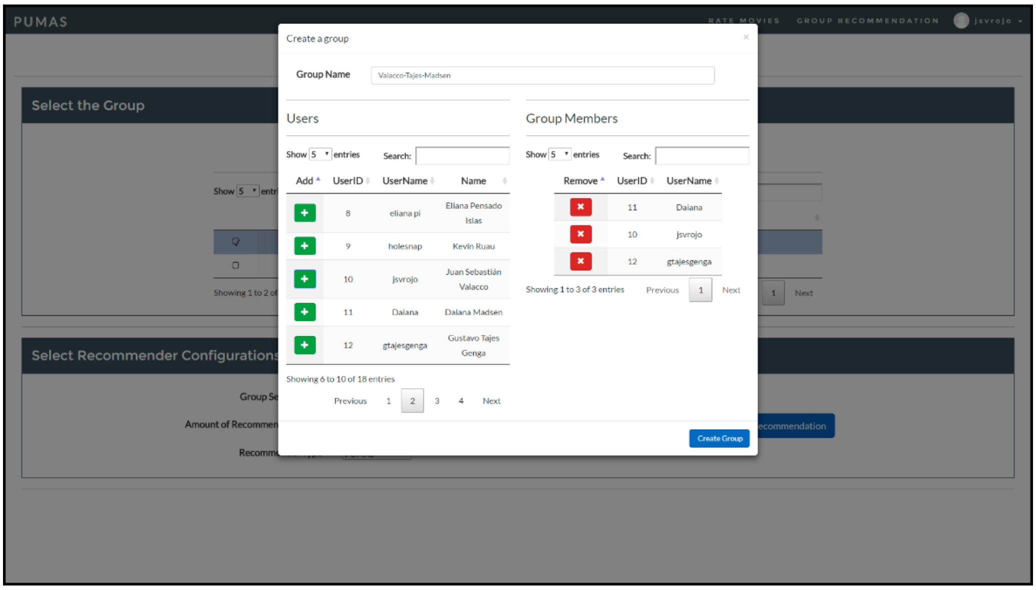

Figure 5: Group Creation (Example).

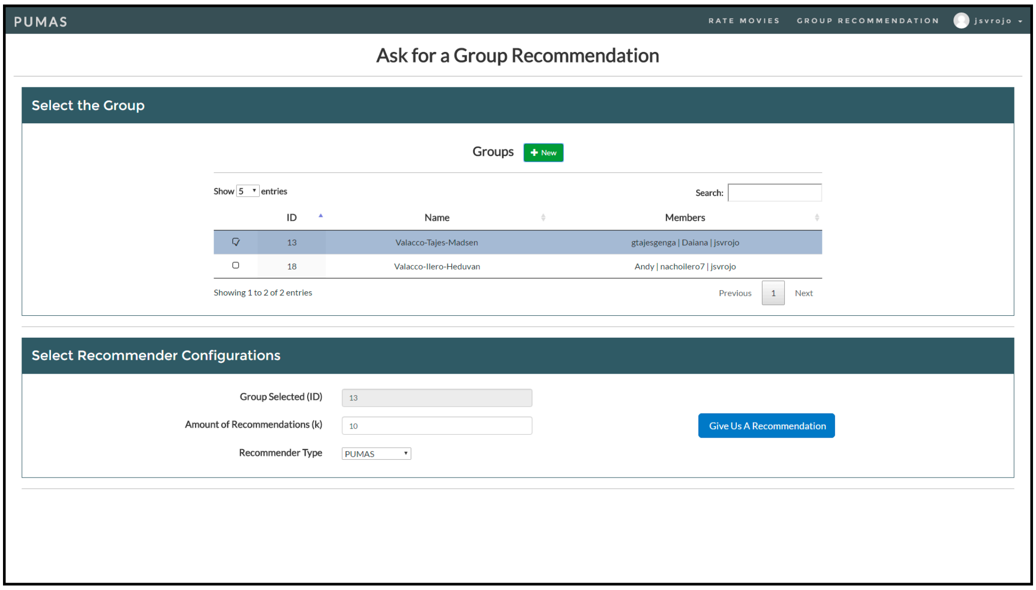

Figure 6: Group Recommendation view.

Step 4: Ask for a group recommendation

When all the parameters of the group recommendation process were defined (group, *k* and the recommendation approach), the active user only needs to click on the *"Give Us a Recommendation"* button and wait until the *Recommendation Results* view is showed by the application. The response time of the recommender system depends on the technique selected, the group size, the group member's preference models (for groups in which the users don't have enough preferences loaded in their profiles, the recommendations takes more time regardless the approach used), among other minor factors.

Additionally, as explained in (Villavicencio, et al., 2016), even though the recommendation process when using PUMAS approach can take a bit longer than when using the TRADGREC one, the quality of the recommendation tends to be better when using the former approach and also the recommendation time is in most of the cases within an acceptable time window (between 1 and 15 seconds).

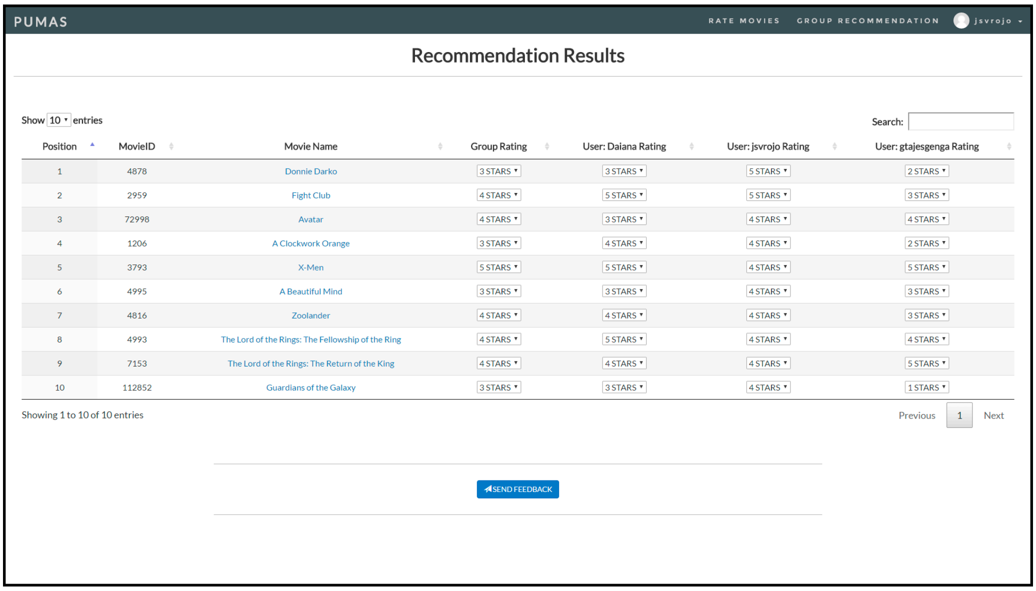

Figure 7: Recommendation Result and Feedback view.

Step 5: Get the recommendation and give feedback about it

After the recommendation is produced, the application presents to the user the list of recommendations in the "Recommendation Results" view (Figure 7). In the mentioned view the users receive also a form to place their feedback on the recommendations. This form was created only for evaluation purposes only, to assess the quality of the recommendation from the group members point of view, and to assess the estimation errors of both of the recommenders. This feedback mechanism allows us to compare the rating the recommender thought the user will give to a certain movie against the rating given by the user. When filling this form, the users must rate every one of the recommended movies both individually and as a group (in the latter case, the group members should discuss among each other about the group rating they would give to the movie). The ratings are, at the time given in terms of "stars"[3], and they depict the interest of the group member/group in the movie.

[3] This will be changed in the future as we do consider that stars cannot capture the real essence of the feedback.

©Ediciones Universidad de Salamanca / cc by-nc-nd

4. Related works

The problem of generating recommendations to groups began to be investigated in the last decade (Cantador & Castells, 2012), and most of the proposed solutions for this problem share one trait: they seek to reuse the technology used for producing recommendations for individual users by using aggregation techniques. Where and when they use those techniques varies from one recommender system to another, but it is possible to classify all the systems in three main categories:

i. Those that merge individual recommendations. These systems generate individual recommendations for every one of the group members and then aggregate those recommendations using some technique so as to produce the group recommendation (Baltrunas, et al., 2010).

ii. Those that aggregate the individuals' profiles. These systems generate an artificial profile that contains the aggregated information of the profiles of the individuals that form part of the group. This way, the group is treated as any other user and, therefore, the recommendation techniques for individuals can be applied to produce group recommendations (Christensen & Schiaffino, 2014).

iii. Those that perform an aggregation of individuals' preferences (ratings). Similarly to what the systems of the second category do, these systems also attempt to create a virtual user that represents the group but the preferences of the users are aggregated instead of their profiles. The process to create the recommendations is the same: once the group user (virtual user) is created, it is added along with its preferences (computed using the aggregation technique) to a single user recommender, which treats the group user as any other user, and therefore can produce recommendations for him.

The aggregation technique to be used depends on the category in which the system falls. This is because not all the techniques are suitable to be applied to every type of data and every situation, for example, a technique that is useful for merging individual recommendations probably will not be useful for computing the aggregated rating of one item.

Multi-agent systems (MAS) have been applied in various domains. When it comes specifically to recommendation systems, some approaches have proposed multi-agent techniques to generate recommendations to both individual users and groups in different domains, like adaptive customization of websites (Morais, et al., 2012), e-commerce (Lee, 2004), games on mobile phones (Skocir, et al., 2012), semantic knowledge extraction (Lopes, et al., 2009), tourism (Bedi, et al., 2014), among others. One thing to notice is that most of those systems can produce recommendations targeted only to individual users.

In (Blanco-Fernandez, et al., 2004), the authors present AVATAR, a modular multi-agent architecture for a personalized recommender system on the TV shows domain, whose main novelty is the semantic reasoning about user preferences and historical logs, using an OWL ontology. The system presented in (Bedi, et al., 2014), MARST, uses a Reputation based Collaborative Filtering (RbCF) algorithm for generating relevant recommendations to a user. Finally, in (Marivate, et al., 2008) the authors present a Multi-Agent approach to the problem of recommending training courses to engineering professionals.

To the best of our knowledge, only a few works have targeted group recommendations with MAS. In (Bekkerman, et al., 2006) a group recommender system relying on the application of cooperative negotiation is presented. The authors propose a process in which agents, acting on behalf of group members, participate in a direct (alternating offers) or mediated (merging rankings) negotiation. This negotation produces group recommendations, based on individual recommendations and user preference models. The approach has only been tested with simulations involving two agents while we will test our approach on bigger groups of users. In (Garcia, et al., 2009) an agent-based negotiation schema that uses alternating offers is developed, in which agents negotiate the preferences of the whole group. The authors of (Sebastiá, et al., 2011) propose a system named *e-Tourism* that is able to produce recommendations for both individuals and groups, but the downside of this system is that for producing the latter it makes use of aggregation techniques. Finally, in (Garcia & Sebastia, 2014) the authors propose a MAS where user agents negotiate with the aim of building a group profile that satisfies the users' requirements. A mediator governs the negotiation in order to facilitate the agreements. Our work differs from the ones of Garcia in that they negotiate user preferences while we negotiate recommendations.

5. Conclusions

PUMAS-GR is a MAS approach for group recommendation based on negotiation techniques. Preliminary experiments with our prototype in the movies domain have shown promising results in terms of satisfaction of group members, when compared to traditional rank aggregation techniques. A limitation of our prototype is the high reliance on movie scores predicted by Duine as the main source of rankings for individual users. In addition, Duine sometimes presents performance problems when recommender system is used constantly by several users. However, our architecture is flexible to admit other scoring strategies or (single-user) recommender systems. Currently, we are in the process of substituting Duine by Mahout[4], in order to improve the performance of the prototype. Finally, we plan to evaluate our approach in other domains involving groups (e.g., tourism, software architecture decision making), and to compare it with other standard techniques for group recommendation.

6. Acknowledgements.

This work has been partially supported by projects ANPCyT-PICT2011-0366 and CONICET-PIP112-201101-00078 (Argentina), and also by “PUMAS” CONICET-CNRS bilateral cooperation project.

[4] http://mahout.apache.org/

7. References

Baltrunas, L., Makcinskas, T. & Ricci, F., 2010. *Group recommendations with rank aggregation and collaborative filtering*. s.l., ACM, pp. 119-126.

Bedi, P., Agarwal, S. K., Jindal, V. & Richa, 2014. *MARST: Multi-Agent Recommender System for e-Tourism Using Reputation Based Collaborative Filtering*. New York, NY, USA, Springer-Verlag New York, Inc., pp. 189-201.

Bekkerman, P., Sarit, K. & Ricci, F., 2006. *Applying cooperative negotiation methodology to group recommendation problem*. s.l., s.n.

Blanco-Fernandez, Y. et al., 2004. *AVATAR: An Advanced Multi-agent Recommender System of Personalized TV Contents by Semantic Reasoning*. s.l., s.n., pp. 415-421.

Cantador, I. & Castells, P., 2012. Group Recommender Systems: New Perspectives in the Social Web. In: *Recommender Systems for the Social Web*. s.l.:Springer Berlin Heidelberg, pp. 139-157.

Christensen, I. & Schiaffino, S., 2014. A hybrid approach for group profiling in recommender systems. *J. of Universal Computer Science,* 20(4), pp. 507-533.

Endriss, U., 2006. *Monotonic Concession Protocols for Multilateral Negotiation*. New York, NY, USA, ACM, pp. 392-399.

Garcia, I. & Sebastia, L., 2014. A negotiation framework for heterogeneous group recommendation. *Expert Systems with Applications ,* 41(4,1), pp. 1245-1261.

Garcia, I., Sebastia, L. & Onaindia, E., 2009. *A Negotiation Approach for Group Recommendation*. s.l., s.n., pp. 919-925.

Jameson, A. & Smyth, B., 2007. The Adaptive Web. In: P. Brusilovsky, A. Kobsa & W. Nejdl, eds. s.l.:Springer-Verlag, pp. 596-627.

Lee, W.-P., 2004. *Towards agent-based decision making in the electronic marketplace: interactive recommendation and automated negotiation*. s.l.:Elsevier {BV}.

Lopes, J. S., Alvarez-Napagao, S., Confalonieri, R. & Vázquez-Salceda, J., 2009. USE: a Multi-Agent User-Driven Recommendation System for Semantic Knowledge Extraction.

Marivate, V. N., Ssali, G. & Marwala, T., 2008. *An Intelligent Multi-Agent Recommender System for Human Capacity Building*. s.l., IEEE, pp. 909-915.

Masthoff, J., 2011. Recommender Systems Handbook. In: F. Ricci, L. Rokach, B. Shapira & P. Kantor, eds. s.l.:Springer Science+Business Media, pp. 677-702.

Morais, J., Oliveira, E. & Jorge, A., 2012. Distributed Computing and Artificial Intelligence. In: s.l.:Springer, pp. 281-288.

Ricci, F., Rokach, L., Shapira, B. & Kantor, P., 2010. *Recommender Systems Handbook*. s.l.:Springer.

Sebastiá, L., Giret, A. & García, I., 2011. A Multi Agent Architecture for Single User and Group Recommendation in the Tourism Domain.

Skocir, P., Marusic, L., Marusic, M. & Petric, A., 2012. Agent and Multi-Agent Systems. Technologies and Applications. In: s.l.:Springer, pp. 104-113.

©Ediciones Universidad de Salamanca / cc by-nc-nd

Villavicencio, C., Schiaffino, S., Diaz-Pace, J. A. & Monteserin, A., 2016. PUMAS-GR: A Negotiation-Based Group Recommendation System for Movies. In: Y. Demazeau, T. Ito, J. Bajo & M. J. Escalona, eds. *Advances in Practical Applications of Scalable Multi-agent Systems. The PAAMS Collection: 14th International Conference, PAAMS 2016, Sevilla, Spain, June 1-3, 2016, Proceedings*. Cham: Springer International Publishing, pp. 294-298.

Wooldridge, M., 2009. *An Introduction to MultiAgent Systems*. Second Edition ed. s.l.:John Wiley & Sons.

Zeuthen, F. L. B., 1930. *Problems of Monopoly and Economic Warfare*. London, UK: Routledge and Sons.

©Ediciones Universidad de Salamanca / cc by-nc-nd