CATS: A Synchronous Approach to Collaborative Group Recommendation

Kevin McCarthy, Maria Salamó, Lorcan Coyle, Lorraine McGinty, Barry Smyth & Paddy Nixon
Adaptive Information Cluster, School of Computer Science & Informatics
University College Dublin Belfield, Dublin 4, Ireland.
{kevin.mccarthy, maria, lorcan.coyle, lorraine.mcginty, barry.smyth, paddy.nixon}@ucd.ie

Abstract

Group recommender systems introduce a whole set of new challenges for recommender systems research. The notion of generating a set of recommendations that will satisfy a group of users with potentially competing interests is challenging in itself. In addition to this we must consider how to record and combine the preferences of many different users as they engage in simultaneous recommendation dialogs. In this paper we introduce a group recommender system, called CATS, that is designed to provide assistance to a group of friends trying the plan a skiing vacation. The system uses the DiamondTouch interactive tabletop to allow up to 4 users to simultaneously engage in parallel recommendation sessions and we describe how personal and shared profiles and interaction spaces can be managed to generate sets of recommendations for the individual and the group.

Introduction

The paper describes a novel conversational, collaborative group recommender system called CATS (Collaborative Advisory Travel System), designed to help a group of up to 4 friends plan and arrange their skiing vacation. This system is designed around the DiamondTouch interactive tabletop. The CATS system is based around the notion of a shared collaborative space for a group of users who also can access their own personal spaces. Individual user feedback in used to update explicit user models, on a per user basis, as well as a global user model. In addition, recommendations for the individual are generated in response to direct user feedback while at the same time group recommendations are generated proactively through the shared interaction space.

Before describing the details of the CATS framework, we begin with a background review that looks at related work in recommender systems research as well as providing an overview of the DiamondTouch device. Following this we detail the CATS framework focusing on the core interface and recommendation components and providing an example walk through of a particular recommendation session.

Research Background

In this section we discuss related research to this project in the areas of collaborative/co-operative recommendation and multi-user interfacing. We are especially interested in conversational content-based recommender systems where critiquing is appropriate as a user-feedback strategy. We also introduce the DiamondTouch interface device used by the CATS framework to showcase our simultaneous collaborative group critiquing recommender system.

Group Recommendation

Group decision making has long been a topic of research in distributed AI and multi-agent systems. The notion of an agent or a multi-agent system interacting, collaborating or negotiating in an environment has been previously addressed by several researchers in the agent community. Within the area, the applications range from virtual environments (Prada & Paiva 2005) to sales by action (Faratin, Sierra, & Jennnings 2002). The vast majority of these systems assume an automated negotiation that is based on some static preferences previously defined in the system. However, in our case, we have human users with individual (often different) initial preferences, and usually these preferences change as the recommendation session progresses. Therefore, a static model of preferences is inappropriate and an alternative method for modelling these preferences (and preference conflicts) is required. Moreover, preference modelling needs to not only capture individual user preferences but also an effective way of modelling the groups evolving preferences as a whole is required.

Other research in the area of *group* recommendation includes the MUSICFX System (McCarthy & Anagnost 1998). MUSICFX is a group preference arbitration system that adjusts the selection of music playing in a fitness center to best accommodate the musical preferences of the people working out at any given time. The preferences have been previously specified by the members who are currently working out. POLYLENS (O'Connor *et al.* 2001) is a generalization of the MOVIELENS system that recommends movies to group of users. In that case, the recommender is based on collaborative filtering which uses the history of preferences of past users in similar situations. Another example, is the TRAVEL DECISION FORUM (Jameson 2004) prototype which helps a group of users to agree on the de-

sired attributes of a vacation that they are planning to take together. Special attention is given to support for users who are not collocated and who can therefore not engage in face-to-face discussions. Plua and Jameson previously worked on a group travel recommender system (Plua & Jameson 2002) where users can get help from others in their group about preferences when their domain knowledge may be incomplete. However this system was intended for use by a group that interact asynchronously rather than simultaneously.

Critiquing Based Recommenders

We are especially interested in a form of user feedback called *critiquing* (McGinty & Smyth 2003), where a user indicates a directional feature preference in relation to a presented recommendation. For example, in a travel vacation recommender, a user might indicate that they are interested in a vacation that is *longer* than the currently recommended option; in this instance, *longer* is a critique over the *duration* feature. Within recommender systems literature the basic idea of critiquing can be traced back to the seminal work of Burke et al. (Burke, Hammond, & Kozlovsky 1995; Burke, Hammond, & Young 1997). Entrée is a restaurant recommender system that employs critiquing to allow users to refine restaurant features such as *price, style, atmosphere,* etc. The advantage of critiquing is that it is a very low-cost form of feedback, in the sense that the user does not need to provide specific feature values, while at the same time helping the recommender to narrow its search focus quite significantly (McGinty & Smyth 2003). Recently, there has been renewed interest in critiquing, as recommender systems become more commonplace, and a number of enhancements have been proposed to the basic critiquing approach. For instance, an improved approach (Reilly *et al.* 2004) is to consider a user's critiquing history, as well as the current critique, when making new recommendations. Given that this approach has been shown to deliver significant improvements in recommendation efficiency (McCarthy *et al.* 2005), it is the assumed method of feedback by the CATS framework.

However, the conventional implementation of critiquing can pose problems in a group recommender setting because each user represents two roles within the system at the same time: their individual role, and implicitly their membership to the group role. Some of the individual user preferences may conflict with the group preferences. For example, our user might have received a recommendation for a luxury 2-week package in Spain for 2000. She might be interested in something around the 1500 mark and so may indicate that she wants a *cheaper* recommendation. However, the group preference is for a more expensive recommendation because they want a luxury package holiday. There is little to be gained from conflicting interactions between individual and group preferences, so the coordination of both roles becomes one of the most important challenges for a collaborative recommender. Another key challenge is how best to make interacting users aware of other user's preferences. In this paper we propose a simple approximation to average the interaction of the individual and the groups roles within a group recommender. We also describe how we communicate the evolving preferences of the group to each of the participants in order to facilitate convergence on a single recommendation through dynamic visualization, and a combination of proactive and reactive recommendation techniques. We also discuss some lessons we have learned from our initial design, and discuss avenues for future research.

The DiamondTouch

The majority of collaborative applications involve separate (often distributed) workspaces, however, with technologies such as the DiamondTouch and Mimio (http://www.mimio.com) it is possible to allow users to work together co-operatively around a common workspace. The DiamondTouch (see Figure 1) supports small group collaboration by providing a display interface that allows users to discuss decisions openly whilst interacting with the display simultaneously (i.e., without having to take turns) (Dietz & Leigh 2001). It consists of a touch sensitive table connected to a computer whose display is projected onto the table. The table can detect and distinguish between simultaneous touch events, allowing the development of innovative and intuitive collaborative and cooperative applications. It enables up to four users interface with the same touch-surface in a very simple fashion, and is capable of discriminating between multiple simultaneous interactions.

Figure 1: The DiamondTouch interactive table-top device.

Research has shown that the DiamondTouch is a more effective interface for solving certain collaborative problems than the two-mouse and one-monitor alternative (Kobourov *et al.* 2005). It is a natural interface for the collaborative task where friends need to book a skiing holiday together. Its 'coffee table' form factor, and intuitive flat orientation, allow users to easily and co-operatively search the space of options and at the same time understand the preferences of the other participants. We propose to use the DiamondTouch tabletop device to showcase our new synchronous collaborative group recommender system.

Collaborative Recommendation

Our approach to group recommendation is based on a collaborative recommender framework that, at the interface layer, assumes the availability of individual and group interaction spaces and at the recommendation layer, assumes a recommendation engine that is able to record and manage personal

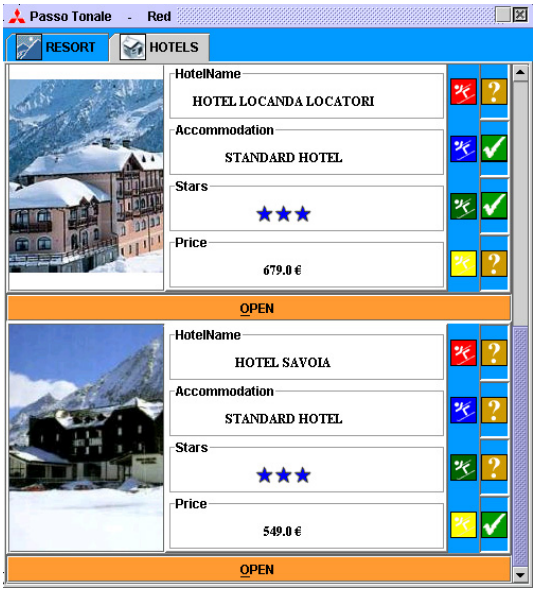

Figure 2: The hotel overview panel.

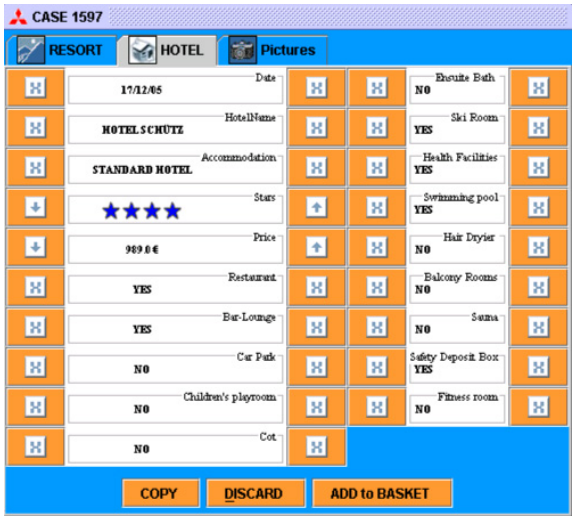

Figure 3: The case panel, where critiquing takes place.

as well as group profiles as the basis for its recommendations. In fact, the group profile is the combination of individual personal preferences and recommendations are made for an individual based on their personal preferences and the preferences of the group. At the same time recommendations can be proactively made for the group with reference to the group preferences.

The CATS (Collaborative Advisory Travel System) will help groups of users find appropriate ski holidays. Our ski holiday dataset consists of 5738 vacation cases. Each case is described in terms of 43 features; of which 25 belong to the resort description (i.e., *beginner*, *cannons*, *transfer time*, *drag lifts*, etc.) and 18 belong to the hotel description (i.e, *stars*, *price*, *balcony room*, *ensuite bath*, etc). The dataset contains nominal and ordinal features.

Illustrated Walk Through

In this section, for simplicity, we walk through the interaction of an individual user with the CATS system (Please note, the DiamondTouch can take input from up to four users). The initial screen presented on the Diamond Touch shows a map of Europe with ski resorts depicted by specialised icons. We think that this simple representation of the search space based on one of the most important features considered by the user, the *location*, helps the user to focus and to understand better the distribution of the information. The map represents the shared space for the group of users. The resorts on this screen can be accessed by any user and the map will reflect individual user activity and the group preferences, through change in icons, icon size, etc. (explained later).

The user can access her own individual personal space by touching one of the resort icons on the shared map. The user is then presented with the resort panel; detailing information about that particular resort. The resort panel shows information about the particular resort such as the number of colour coded ski runs, number of ski lifts, etc. The user can choose to look at the available hotels in this resort by tapping the hotel tab.

The hotel overview panel, Figure 2, shows a list of basic hotel information for the hotels available in a particular resort. The user can then open a hotel and is presented with a new panel containing details of the resort and the hotel. This panel details an instance, also known as case, of our skiing holidays dataset. In Figure 3 you can see the appearance of a case, which is detailed in terms of its features. At this point critiques may be made on the features of the hotel or indeed the resort. These critiques may be used to reject a feature value (when dealing with nominal features) or increase or decrease a feature value (when dealing with ordinal features). In summary, we allow the user to critique each one of the available features by specifying a directional preferences on the feature. A critique over the current case means that the case does not completely satisfy the user and the recommender should propose a new case that is similar to the previous case but, at the same time, has a more desired value to the feature critiqued, and thus better satisfies the users expectations. After a critique has been made by the user, the new case proposed by the recommender is shown. The user can continue critiquing and try to find the best case that satisfies all her ski holiday preferences. Otherwise, the user can just *discard* the case and continue navigation though the map. If the proposed case completely satisfies the users expectations, she can finish her session by tapping the *add to the basket* button.

Interaction Component

Figure 4 shows the interface as presented to a group of four users. We can see that there are four resort/hotel panels each oriented in the direction of each user. These panels represent the individual personal spaces for the users. These panels

appear on top of the map - the group shared space.

Individual Interaction The individual personal space has some interesting characteristics that are novel in interfacing terms and occur in this interface to further aid the group of users, as a whole, converge on a specific ski holiday product. In the first of these mechanisms, the hotel overview panel (see Figure 2) lists the different hotel information depending on the preferred cases of the other users. The basic idea behind ordering the hotel information is to focus the users on the cases where a collaboration or discussion on preferences can be performed by the members of the group. Using the individual personal model of each user, the system counts the number of users interested in that particular hotel and orders the hotels in decreasing order of preferences. At the same time, the hotel overview panel also shows an icon for each user showing their preference for that particular hotel with a check mark or her lack of interest with question mark. With this information the user can see which of the other users in her group have shown a preference for this particular hotel. In addition to these two mechanisms, the user can make a *copy* of the case she is currently critiquing (see Figure 3). The user can then pass the case to another user in the group. Once the receiving user touches the case, they obtain possession of it and the case is added to their individual personal space. In such a situation the first user is giving awareness of her preferences to another user in order to try and reach a mutual agreement and converge the search for a ski holiday.

Group Interaction In the group space (the map), there are also some interesting mechanisms that help aid the group recommendation process. Firstly, the icon that marks the resort that the user is currently critiquing a case from, has a colour coded snowflake associated with it, see Figure 5. Each user has a different coloured snowflake. This allows all users of the system to see what resorts other users are currently viewing. In addition, the size of the snowflake changes according to the preferences of the user as stored in the user's individual personal model. This allows all users to gauge the *level* of interest of other users in a particular resort. Secondly, the size of the icon that represents each resort grows or shrinks in accordance with the preferences of the whole group. This allows all users to see which resorts best suit the needs of the group as a whole, see Figure 5. Also if users are concentrating on just some resorts, they may not realise that there are other resorts available which fit the preferences of the group as whole. With the changing size of the resort icons, the system can direct the users towards resorts (parts of the product space) which they may not have considered before. This helps the group as a whole find a ski holiday which fits their needs.

Discussion Apart from the simultaneous collaborative aspects of the whole group interfacing with the system on a single DiamondTouch, this situation also allows other interesting multi-user interaction. The users are all positioned around the DiamondTouch device. They can copy and pass cases of interest to other users, but they can also confer on a face-to-face basis about their preferences. This type of face-to-face user interaction is not often found in collaborative recommender systems and should help users, along with recommendation and interfacing aids, converge on a suitable ski holiday more quickly.

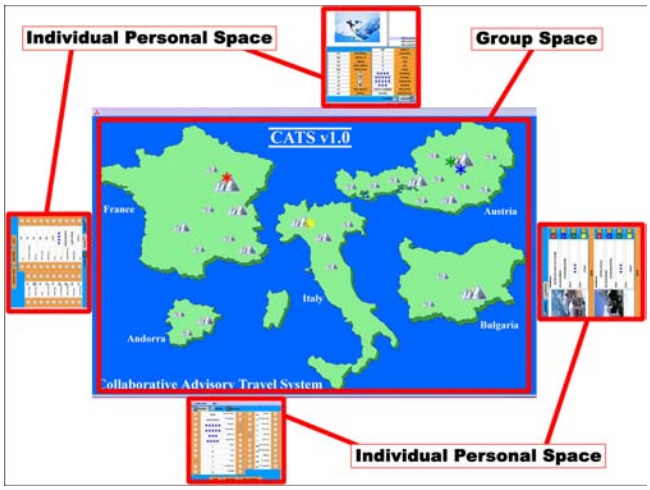

Figure 4: The CATS system as presented on the DiamondTouch.

Figure 5: Section of interface showing "snowflake" indicator and changing resort icon size.

Recommendation Component

Figure 6 outlines a high level overview of the recommendation architecture. There are two parts to the recommendation component of the system; 1) individual recommendation, where the system reactively recommends cases to the user and 2) group recommendation, where the system proactively pushes recommendations to the group of users through the group space. The system maintains a session-based *individual personal model*, that is made up of those critiques chosen by the user so far and those cases that are still opened. At the same time a *group user model* is also maintained which contains all the information of each of the users individual personal models. In this section we describe the user models and how they influence both individual and group recommendations.

Generating Individual Recommendations The user models are based on the user interaction with the system.

Each user interacts with the recommender using critiques. These critiques are stored in the personal individual model (IM) of preferences, $IM = \{I_1, ..., I_n\}$, where I_i is a single unit critique. At the same time these critiques update the group model. The group user model (GUM) is given by $GUM = \{G_1, ..., G_n\}$, where G_i is a record of a single *unit critique* with its corresponding *user name*. Each record is stored in the group user model. However, the recommender should interact in a fair way with the individuals preferences and the group preferences. This interaction is not easy to perform because some of the preferences of one user may be inconsistent with the preferences of another in the group. So, to improve the process, we take into account both preferences: the individual's and the group's preferences.

In this sense, we propose a basic user modelling strategy that averages the preferences of the individual and the preferences of the remaining members of the group. To put it differently, the cases presented by the recommender include both preferences, the individual and the remaining members of the group. This behaviour is established in order to reduce the number of cycles needed to arrive at a compromise of all the members. Thus, during recommendation to a user, the GUM model is broken down into the remaining members' preferences model (MM), $MM = \{M_1, ..., M_n\}$, where M_i is a single unit critique and an associated user identifier.

Both the individual model and the member model are used during recommendation to influence the choice of a new product case, along with the current critique. Maintaining an accurate user model, however, is not quite as simple as storing a list of previously selected critiques for the individual in IM or for the remaining members of the group in MM. Users may not always provide consistent feedback, sometimes they make mistakes or change their mind.

To eliminate preference inconsistencies, the *critiquing strategy* should consistently update the model. In this paper, both models are updated in the same way. The model is updated by adding the latest critique only after pruning previous critiques. Specifically, prior to adding a new critique all existing critiques that are inconsistent with it are removed, as are all existing critiques for which the new critique is a refinement. The basic idea behind the user model is that it should be used to influence the recommendation process, prioritising those product cases that are compatible with its critiques.

The standard approach to recommendation, when using critiquing, is a two step procedure. Firstly, the remaining cases are filtered by eliminating those cases that fail to satisfy the current critique. Next, these filtered cases are rank ordered according to their similarity to the current recommendation. Incremental critiquing (Reilly *et al.* 2004) makes one important modification to this procedure. Instead of ordering the filtered cases on the basis of their similarity to the recommended case, it also computes a *compatibility* score for each candidate case. The compatibility score is essentially the percentage of critiques in the user model (IM or MM) that this case satisfies, see Equation 1.

$$IComp(c', U) = \frac{\sum_{\forall i} satisfies(U_i, c')}{|U|} \quad (1)$$

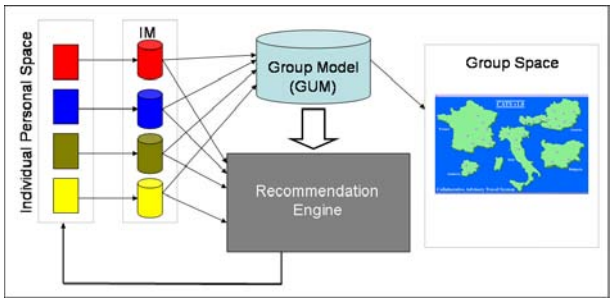

Figure 6: Recommendation architecture.

It is important to note that $satisfies(U_i, c')$ returns a score of 1 when the critique, U_i, satisfies the filtered case, c', and returns 0 otherwise. Thus a case that satisfies 3 out of the 5 critiques in a user model obtains a compatibility score of 0.6.

We maintain the basis of the compatibility score but adapt it to our group recommender system, see Equation 2. We compute separately the compatibility for the individual model and the remaining members of the group model. For the purpose of this paper we set up the α=0.5, so we average equally the individual preferences and the preferences of the remaining members model. This is only one way of combining the individual and group preferences, however, a discussion of other strategies for combining user models is beyond the scope of this paper.

$$GComp(c', GUM) = \alpha * IComp(c', IM) + (1 - \alpha) * IComp(c', MM) \quad (2)$$

$$Qual(c', c, GUM) = \beta * GComp(c', GUM) + (1 - \beta) * Sim(c', c) \quad (3)$$

Once the compatibility of the group is computed, the $GComp$ score and the candidate case's, c', similarity to the current recommend case, c, are combined in order to obtain an overall *quality* score, see Equation 3, where β=0.75. The quality score is used to rank the filtered cases prior to the next recommendation cycle; of course, the case with the highest quality is then chosen as the new recommendation. The above formulation allows us to prioritise those candidate cases that: (1) satisfy the current critique; (2) are similar to the previous recommended case; and (3) satisfy many individual and members previous critiques. In so doing we are implicitly treating the past critiques in the user model as *soft constraints* for future recommendation cycles; it is not essential for future recommendations to satisfy all of the previous critiques, but the more they satisfy, the better they are regarded as recommendation candidates. Moreover, given two candidates that are equally similar to the previously recommended case, the algorithm prefers the one that satisfies the greater number of critiques.

Generating Group Recommendations In addition to this reactive recommendation of cases, on the basis of explicit user feedback, the group user model (GUM) is also used to

bring new cases to the attention of the group in a variety of ways. For example, when a user is viewing the available hotels of a particular resort, the group user model is queried and the hotels are reordered based on critiques contained in the GUM. In this way those cases that are most consistent with the overall preferences of the group are presented first to the user in question.

The group user model (GUM) is also responsible for proactive recommendations that are made via the group space. The objective is to bring potential new sets of cases to the attention of a user, cases that might not be recommended according to the users current critiquing session, but cases that are consistent with group preferences in general. Essentially these new cases are highlighted through the shared interaction space by increasing the size of the resort icons associated with those resorts that best match the group preferences learned so far. This occurs each time the group model is updated; that is each time one of the individual users registers a new critique as part of their normal feedback. In this way the central resorts map is continuously being updated to reflect the current group preferences, thereby providing user with an opportunity to 'break out' of a given recommendation session in order to evaluate a new resort that has been highlighted as a current group preference.

Conclusion

In this paper, we have introduced CATS, our Collaborative Advisory Travel System, which allows a group of users to simultaneously collaborate on choosing a skiing holiday package which satisfies the group as a whole. This system has been developed around the DiamondTouch interactive tabletop, which makes it possible to develop a group recommender that can be physically shared between up to 4 users. In the paper we have focused on the core interface, profiling and recommendation issues that have arisen during the development of the system. We have, for example, described how users manipulate their own personal interaction spaces to received personalized recommendations that address their particular needs and emerging group preferences. We have also explained how users interact with a shared space so that they can be alerted to proactive recommendations that are automatically generated by the recommender system based on the developing group model.

While the research continues to be a work in progress, at this point the system has been developed and tested in small-scale recommendation scenarios. Future work will focus on a large-scale online evaluation of the system with live users to analyse the effectiveness of the group recommendation model. In addition we also anticipate a variety of different approaches to constructing group models from the feedback of the separate individual users.

Acknowledgement

This material is based on works supported by Science Foundation Ireland under Grant No. 03/IN.3/I361.

References

Burke, R.; Hammond, K.; and Kozlovsky, J. 1995. Knowledge-based Information Retrieval for Semi-Structured Text. In *Proceedings of the AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, 19–24. AAAI Press.

Burke, R.; Hammond, K.; and Young, B. 1997. The FindMe Approach to Assisted Browsing. *Journal of IEEE Expert* 12(4):32–40.

Dietz, P., and Leigh, D. 2001. Diamondtouch: a multi-user touch technology. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, 219–226. New York, NY, USA: ACM Press.

Faratin, P.; Sierra, C.; and Jennnings, N. 2002. Using similarity criteria to make issue trade-offs in automated negotiations. volume 142, 205–237.

http://www.mimio.com. The mimio electronic white board.

Jameson, A. 2004. More than the sum of its members: Challenges for group recommender systems. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, 48–54.

Kobourov, S. G.; Pavlou, K.; Cappos, J.; Stepp, M.; Miles, M.; and Wixted, A. 2005. Collaboration with diamondtouch. In *10th International Conference on Human-Computer Interaction (INTERACT)*.

McCarthy, J., and Anagnost, T. 1998. Musicfx: An arbiter of group preferences for computer aupported collaborative workouts. In *Proc. of Conference on Computer Supported Cooperative Work*, 363–372.

McCarthy, K.; Reilly, J.; McGinty, L.; and Smyth, B. 2005. Experiments in Dynamic Critiquing. In Riedl, J.; Jameson, A.; Billsus, D.; and Lau, T., eds., *Proceedings of the International Conference on Intelligent User Interfaces (IUI'05)*, 175–182. ACM Press. San Diego, CA., USA.

McGinty, L., and Smyth, B. 2003. Tweaking Critiquing. In *Proceedings of the Workshop on Personalization and Web Techniques at the International Joint Conference on Artificial Intelligence*. Morgan-Kaufmann.

O'Connor, M.; Cosley, D.; Konstan, J.; and Riedl, J. 2001. PolyLens: A Recommender System for Groups of Users. In *Proc. of European Conference on Computer-Supported Cooperative Work*, 199–218.

Plua, C., and Jameson, A. 2002. Collaborative preference elicitation in a group travel recommender system. In *Proceedings of the AH 2002 Workshop on Recommendation and Personalization in eCommerce*, 148–154.

Prada, R., and Paiva, A. 2005. Believable groups of synthetic characters. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 37–43.

Reilly, J.; McCarthy, K.; McGinty, L.; and Smyth, B. 2004. Incremental Critiquing. In Bramer, M.; Coenen, F.; and Allen, T., eds., *Research and Development in Intelligent Systems XXI. Proceedings of AI-2004*, 101–114. Springer. Cambridge, UK.