Chapter 13
On the Evolution of Critiquing Recommenders

Lorraine McGinty and James Reilly

Abstract Over the past decade a significant amount of recommender systems research has demonstrated the benefits of conversational architectures that employ critique-based interfacing (e.g., *Show me more like item A, but cheaper*). The critiquing phenomenon has attracted great interest in line with the growing need for more sophisticated decision/recommendation support systems to assist online users who are overwhelmed by multiple product alternatives. Originally proposed as a powerful yet practical solution to the preference elicitation problem central to many conversational recommenders, critiquing has proved to be a popular topic in a variety of related areas (e.g., group recommendation, mixed-initiative recommendation, adaptive user interfacing, recommendation explanation). This chapter aims to provide a comprehensive, yet concise, source of reference for researchers and practitioners starting out in this area. Specifically, we present a deliberately non-technical overview of the critiquing research which has been covered in recent years.

13.1 Introduction

The evolution of the critiquing research landscape has largely been influenced by the changing needs of online users and the increasingly complex product domains that they have turned to explore. Over the past decade, a variety of critique-based recommendation methodologies have been proposed along with demonstrated evidence of their potential to improve recommendation performance. It could be argued that the significance of a piece of research can be often measured not just in terms of the

Lorraine McGinty
UCD School of Computer Science and Informatics, University College Dublin, Dublin 4, Ireland.
e-mail: lorraine.mcginty@ucd.ie

James Reilly
Google Inc., 5 Cambridge Center, Cambridge, MA 02142, United States.
e-mail: jamesreilly@google.com

F. Ricci et al. (eds.), *Recommender Systems Handbook*,
DOI 10.1007/978-0-387-85820-3_13,

number of questions answered (e.g., methodology proposed to solve a given problem), but also in terms of the number of new issues raised; whose answers present as goals for future research.

This chapter is directed towards early-stage researchers starting out in this area. It aims to provide a useful and deliberately non-technical overview of the promises and pitfalls of critiquing that have been brought into focus in recent years. In Section 2 we start off by describing the benefits of the approach as recognised by early critiquing systems. Next, in Section 3, we outline some of the key issues and challenges that have been identified as well as the approaches that have been taken to improve: (1) critique presentation in view of optimising preference acquisition, and (2) the retrieval performance of critique-based recommenders. Section 4 provides an overview of the various design considerations that have influenced the integration of the critiquing mode of feedback across alternate platforms and user environments (e.g., mobile space, individual and collaborative platforms, etc.). Section 5 summarises the resources, methodologies and evaluation criteria that have been typically adopted by practitioners in the area to date. Finally, by way of conclusion we outline some of the open challenges and opportunities that exist for critique-based recommenders.

13.2 The Early Days: Critiquing Systems/Recognised Benefits

Interactive recommender systems typically engage users in a conversational dialogue to learn their preferences and use this feedback to improve the system's recommendation accuracy. Many conversational recommenders drive the preference elicitation task through the use of examples (for further details see Chapter 21). For instance, a user may be presented with one or more examples/recommendations (e.g., a movie, book, camera) and asked to provide feedback (e.g., provide ratings, indicate a preference). The *critiquing* mode of feedback has become a popular topic amongst those conducting recommender systems research and those developing example-based conversational architectures. The primary reason why it has become so popular is that it strikes an acceptable balance between the effort that a user must expend when providing feedback and the information value it provides. In comparison to the standard value elicitation approach it is a very *low-cost* form of feedback (i.e., in terms of user effort) that provides a relatively unambiguous indication of the user's current requirement (e.g., "Show me more like item A, but different in terms of feature X") . Critiquing is also well-suited to even the most basic interfaces and to users with only a rudimentary understanding of certain recommendation domains [7, 8, 65].

In many domains, it cannot be assumed that users are able to completely articulate their preferences from the outset [48], and/or have a clear understanding of the feature trade-offs/compromises that exist (for related discussion see Chapter 11). For example, a user hoping to buy a new laptop may not initially think about the trade-off that often exists between the products weight and it having an integrated

CD-ROM. If they were to formulate a query for a laptop that was less than 2kg in weight with a CD-rom facility they may not find any products that satisfy *both* of these preferences. Instead, as users become more familiar with the domain and the product options available, their preferences often change, becoming more rigid [48]. Users often tend to lack the motivation to completely specify their preferences up front without any perceived benefits [67]. In fact, this may be unnecessary, as sometimes only a partial set of preferences may be required to make good recommendations [17].

Critique-based conversational recommenders offer flexible support to users as they navigate product catalogues and help them to better understand their preference requirements. Instead of requiring users to specify their preferences from the outset, user preferences are built up over a series of *recommendation cycles*. In each cycle of a *recommendation session* the system makes one or more recommendations to the user and invites them to critique one of the examples. Feature critiques typically take the form of *directional* or *replacement* (a term that was initially defined by [37]). *Directional* critiques effect an increase or decrease over numerical feature values (e.g., *cheaper* implies [$<$ *price*]). *Replacement* critiques allow for the substitution of any value (i.e., aside from critiqued value) for a non-numeric feature (e.g., *different manufacturer* implies [$!=$ *manfacturer*]). A recommendation satisfying the applied critique is returned and the user is invited to critique this in line with their requirements. This process continues until the user: (1) accepts a recommendation, (2) exhausts all potential possibilities, or (3) terminates the session prematurely.

Early work in this area dates back to the early 1980's when the RABBIT systems [70, 71] introduced the critiquing as a new interface paradigm for formulating database queries. Users could critique fields of example records with options like *prohibit* or *specialise*. Based on user feedback the system would reformulate the query and present another example record. The FindMe Systems [7, 8, 19] developed by Burke *et al.*, were the first to employ critiquing in web-based recommenders recognising the need to focus on educating the user about the options space[1]. Originally, FindMe recommenders were developed as *browsing assistants*, that helped users browse through large information-spaces by providing critiques on presented examples, such as restaurants (*Entrée*), automobiles (*Car Navigator*), apartments (*RentMe*), and movie rentals (*Video Navigator*)[2]. Other examples of early example-based critiquing systems include: *Apt Decision* [64], *SmartClient* [53, 54, 55], and the *Automated Travel Assistant* (ATA) [22]. In the next section we outline key challenges in critique-based recommendation that have been addressed by these systems and other subsequent research.

[1] More recent research has highlighted that there is a tension between the need for a user to explore the space of items to understand the options and desire for short recommendation dialog.

[2] The Wasabi Personal Shopper [4] was developed as a general-purpose domain-independent version of the FindMe systems

13.3 Representation & Retrieval Challenges for Critiquing Systems

Early systems and more recent critiquing research can be reviewed along a number of dimensions. In this section we focus our review on two: critique *representation* and recommendation *retrieval*. In the first instance, we distinguish between the alternate critique representation/formulation approaches that have been developed to optimise preference acquisition. We describe how the nature of the critiques that are presented to the user as feedback options dictate very different limitations and benefits. In the second instance, we outline some of the key issues that have presented retrieval challenges in critiquing systems, and discuss (where appropriate) how these have influenced the development of more sophisticated approaches to preference modeling and revision.

13.3.1 Approaches to Critique Representation

Over the past decade a variety of alternate approaches to critique representation have been proposed. The Entrée [8] restaurant recommender, is one of the earliest and most well-known member of the FindMe family of critique-based recommenders. Entrée offers two ways for users to specify their initial preferences. One is to specify a known restaurant that is similar to what the user is looking for. The alternative requires that users start the recommendation session by specifying their dining interests according to a number of high-level features. Along with the returned recommendation are seven pre-defined *unit* critique options which can be applied over features; *cheaper* critiques the *price* feature, and *more formal* critiques the *style* feature, for example. The critiques serve as temporary filters over the product space, eliminating incompatible restaurants from consideration for the next recommendation cycle. In this section we discuss a number of challenges that motivated further research and advancement in this area.

13.3.1.1 Over-critiquing & protracted recommendation dialogues

Unit critiques only allow users to express preferences over a single feature in each recommendation cycle. This ultimately limits the ability of the recommender to narrow its focus, which can result in slow unnecessarily long recommendation dialogues. Furthermore, a user may not understand the feature trade-offs that exist within a particular domain and hence might be inclined to continue to critique a specific feature (e.g., *price*) until a recommendation with an acceptable value has been reached. However, they may later realise that the value of another important feature has since changed and is no longer acceptable [39]. An alternative strategy is to consider the use of *compound* critiques (the term itself was first introduced by

[26]). These are critiques that operate over multiple features and have the potential to improve recommendation efficiency because they allow the recommender system to gather more preference information in a single recommendation cycle. The promise is that the application of compound critiques enables users to take larger steps though the recommendation space towards preferred options, thus reducing session lengths/interaction times.

The idea of compound critiques is not new. In fact, the early FindMe Systems [7] introduced the *Car Navigator* System (see Fig. 13.1) which uses compound critiquing. Automobiles are described in terms of features such as *horsepower*, *price*, *sportier*, or *gas mileage* that can be directly manipulated by users. Compound critiques are also presented alongside individual feature-level unit critiques providing the user with two alternate ways to refine recommendations. When a user applies the *sportier* compound critique, for example, this has the effect of filtering the remaining options in terms of a number of features; that is, *engine size, acceleration and price* are all increased. Similarly, in the context of a PC recommender a *high performance* compound critique might simultaneously increase *processor speed, RAM, hard-disk capacity* and *price* features.

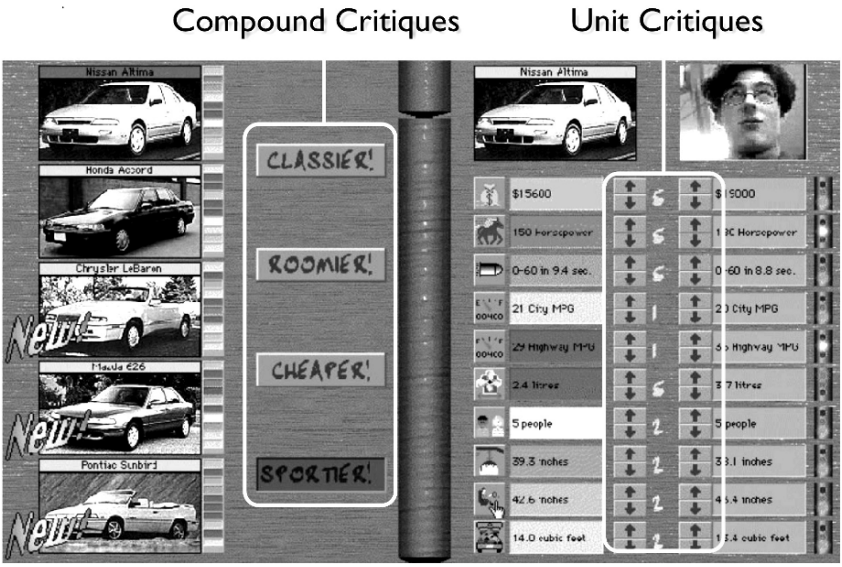

Fig. 13.1: Illustrating how *Car Navigator* presents both static unit and compound critiques.

13.3.1.2 Critique redundancy & hidden feature-dependency

The notion of automated critique generation was first proposed by McCarthy *et. al.* [26], motivated by the observation that certain *static* critiques may not always be relevant. They identified that *session dead-ends* could result whereby a user may apply a critique filter (i.e., unit or compound) and find that there are no subsequent matching recommendations (related work refers to these as *retrieval failures* [18, 37, 39]). For instance, in the *Car Navigator* system the *sportier* critique would continue to be presented as an option even when there are no recommendation candidates remaining that match that criteria. Static critiques (both *unit* and *compound*) do not afford the user any understanding of what recommendation options are available beyond the current example. In fact, they could lead to unnecessary user confusion and interface redundancy. For instance, users may not all share the same understanding of what these static *labels* actually mean (i.e., they are uncertain about what hidden feature filters might be applied). Inevitably, this means that some users may find these static compound critiques to be of little use.

McCarthy *et. al.* [26] argue the need for a more flexible *dynamic* approach to critique generation whereby compound critiques are composed on-the-fly, on each recommendation cycle, in view of presenting applicable critiques that (1) better focus the recommender, (2) eliminate interface redundancy, and (3) remove user apprehension. They make the point that only compound critiques that actually *cover* available recommendation candidates should be presented to the user. Of course, it is reasonable to apply the same rationale to the generation of *unit* critiques, as is implemented by the *Tweak* system [39] to ensure that the user is only presented with unit critique options that lead to at least one product option. Three approaches to the *dynamic* generation of compound critiques have been proposed by (see [26] and [73], & [12]). All of the approaches provide the user with multi-feature critique options that expose the user to the feature changes that will result from an application of that critique (i.e., the recommendation consequence). For example, a user could apply the critique (*more memory, less price, different manufacturer*) if they want to see an alternate PC from that is similar to the current recommendation but cheaper, with more memory, and from a different manufacturer. All approaches have been demonstrated to be effective in terms of the benefits they offer to varying degrees with respect to recommendation efficiency, accuracy, applicability, and usability. However, the key distinguishing factor between the approaches is the knowledge they use to influence critique generation (i.e., domain vs user preference knowledge). Further differences and the motivations for each methodology are described in the next subsections.

13.3.1.3 Limited product-space vision

Reilly *et. al.* [59] demonstrate how to use only domain-knowledge about *available* recommendation candidates to automatically generate compound critiques before every recommendation cycle in their *QwikShop* system (see Fig. 13.2). They con-

centrate on increasing the users depth of understanding about characteristics governing the remaining recommendation candidates, through raising awareness about the feature trade-offs and dependencies that exist beyond the current recommendation. Their so-called *dynamic compound critiquing algorithm* uses the Apriori association data-mining algorithm [1] in order to uncover any frequently occurring feature relationships amongst the remaining recommendation candidates. Typically, large numbers of compound critiques of the form (e.g., *more resolution, more memory, and different manufacturer* [than the current PC recommendation]) are generated by this methodology. It is not practical to present the large lists of compound critiquing options from a user interfacing perspective. Various strategies could be employed for ranking compound critique options (e.g., support, confidence, leverage, lift, and conviction[3]) in order to present the user with k compound critiques alongside the standard unit critique options. McCarthy *et. al.* [26] and Reilly *et. al.* [56] investigate how best to rank compound critiques by their support values (where the *support* refers to the proportion of the products that satisfy the critique). They demonstrate that presenting users with low-support compound critique options provides the best balance with respect to their likely applicability and their ability to focus the search.

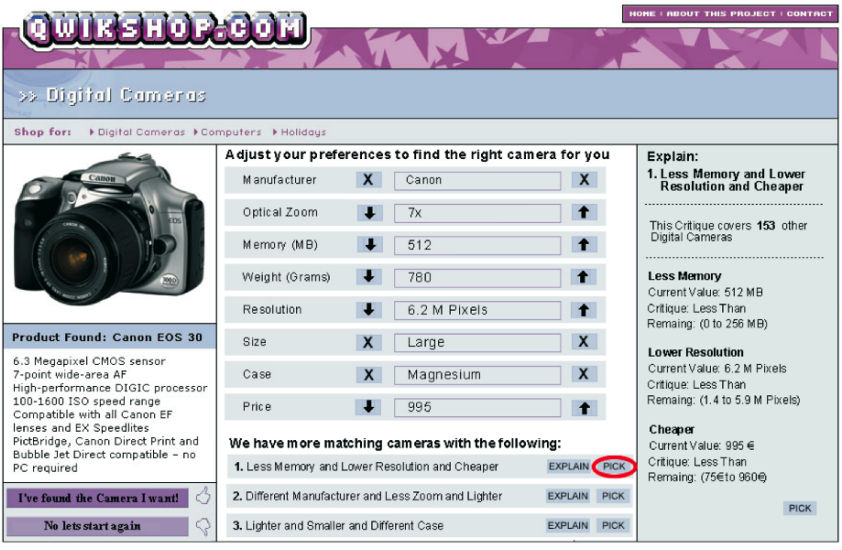

Fig. 13.2: Screen-shot of the QwikShop critiquing system.

It is possible that the compound critique options that are ultimately selected show *limited feature diversity* (i.e., their constituent individual features can overlap to a high degree) due to the high number of remaining options that exhibit the same feature differences to the current recommendation. The related issue of selecting rec-

[3] These are commonly used interest measures for association rules [1].

ommendation candidates for a user, that are relevant to their preferences but different from each other, is a familiar problem to those working in the recommender systems area. In response, a number of diversity-enhancing solutions have been proposed (see Section 13.3.2). Approaches for dynamically presenting diverse compound critiques have been investigated [28] and shown to generate feedback options that are up to 32% more applicable to users. A further potential limitation of this approach is that it does not take into account a users' preferences. In the *QwikShop* system user preferences influence only how recommendations are selected, and not the feedback options presented to the user. However, being less tightly bound to the users preference model, this approach has more potential for the discovery of serendipitous recommendations.

13.3.1.4 Weak relevance of presented feedback options

As mentioned above there is no guarantee that the compound critiques generated by Reilly *et. al.*'s methodology will be relevant to the user as it is influenced only by product domain knowledge. Zhang and Pu [73] propose an alternative methodology that relies on user preference knowledge. Their Multi-Attribute Utility Theory (MAUT), approach that is similar to that implemented by the SmartClient system [53, 54, 55], as the means for dynamically generating compound critiques. Their *EasyShop* recommender, shown in Fig. 13.3, maintains user preference models based on their critiquing feedback and these are used to calculate product *utility* values for each recommendation candidates.

The product with the highest utility is selected as the next recommendation and the k products with the next highest utility values are represented by compound critiques and presented as feedback options. Comparative off-line evaluations [60] show how these preference oriented compound critiques tend to be more more aligned with the user's intended critiquing criteria than the approach described McCarthy *et. al.* [26]. In addition, they tend to result in shorter sessions with higher recommendation accuracy. However, the MAUT approach does suffer from some drawbacks. First, each MAUT-generated compound critique describes only one product, and so these critiques offer users limited exposure to the remaining recommendation opportunities. Secondly, this approach assumes an accurate user preference model. However, user preferences can be very inconsistent and are often subject to rapid change (see Section 13.3.2). Finally, it does not promote diversity across the critiques, and the compound critiques tend to contain a high number of features.

13.3.1.5 Limitations of domain & preference driven approaches

Chen & Pu [12] propose a methodology for dynamically generating compound critiques that aims to preserve the advantages (while minimising the limitations) of the previously described alternatives. Their *preference-based organisation* approach

Fig. 13.3: Screen-shot of the EasyShop critiquing system (from [73]).

aims to dynamically generate diverse critique options that are adaptive to user preferences *and* representative of the remaining recommendation candidates.

User preferences are represented using a multi-attribute utility model. A data-mining algorithm generates the set of compound critiques. These compound critiques are *categories* of available recommendation candidates that best match the user's preference model. From each compound critique, a selection of products are extracted for presentation such that they exhibit high *trade-off utilities* against the current recommendation and are diverse from each other. The *trade-off utility* for a category indicates how well it adapts to the user model. The intuition is that a higher tradeoff utility category covers products that potentially offer more *pros* than *cons* over the current best candidate (see Fig. 13.4).

13.3.1.6 Restricted user control

In more recent work, Chen and Pu [9], make the distinction between *system-proposed* and *user-motivated* critiquing approaches (see Fig. 13.5). Up to this point we have concentrated our discussion on what they refer to as *system-proposed* compound critiques whereby the presented critiques are not defined by the user. Chen and Pu present the example of a user looking for a digital camera with higher resolution and more optical zoom relative to the current recommendation. Suppose

The top candidate according to your preferences

Manufacturer	Price	MegaPixels	Optical zoom	Memory type	Flash memory	LCD screen size	Depth	Weight	
Canon	$242.00	5.0 MP	3x	CompactFlash Card	32 MB	1.8 in	1.37 in	8.3 oz	choose

We have more products with the following

they are cheaper and lighter, but have fewer megapixels

Nikon	$167.95	4 MP	3x	SD Memory Card	14 MB	1.8 in	1.4 in	4.6 oz	choose
Canon	$230.00	4.1 MP	3x	CompactFlash Card	32 MB	1.5 in	1.09 in	6.53 oz	choose
Canon	$180.00	3.3 MP	3x	SD Memory Card	16 MB	2 in	0.83 in	4.06 oz	choose
Canon	$219.18	4.2 MP	4x	MultiMedia Card	16 MB	1.8 in	1.51 in	6.35 oz	choose
Canon	$163.50	3.2 MP	4x	MultiMedia Card	16 MB	1.8 in	1.5 in	6.3 oz	choose
Canon	$199.40	3.2 MP	2.2x	SD Memory Card	16 MB	1.5 in	1.4 in	5.8 oz	choose

they have more megapixels and bigger screens, but are more expensive

Sony	$365.00	7.2 MP	3x	Internal Memory	32 MB	2.5 in	1.5 in	6.9 oz	choose
Canon	$439.99	7.1 MP	3x	SD Memory Card	32 MB	2 in	1.04 in	6 oz	choose
Fuji	$253.00	6.3 MP	4x	XD-Picture Card	16 MB	2 in	1.4 in	7.1 oz	choose
Sony	$336.00	7.2 MP	3x	Internal Memory	32 MB	2 in	1 in	5 oz	choose
Nikon	$304.18	7.1 MP	3x	Internal Memory	13.5 MB	2 in	1.4 in	5.3 oz	choose
Olympus	$334.00	7.4 MP	5x	XD-Picture Card	32 MB	2.0 in	1.7 in	7.1 oz	choose

they are lighter and thinner, but have less flash memory

Pentax	$238.99	5.3 MP	3x	Internal Memory	10 MB	1.8 in	0.8 in	3.7 oz	choose
Canon	$273.18	4.0 MP	3x	SD Memory Card	16 MB	2 in	0.82 in	4.59 oz	choose
Nikon	$329.95	5.1 MP	3x	Internal Memory	12 MB	2.5 in	0.8 in	4.2 oz	choose
Canon	$316.18	5.3 MP	3x	SD Memory Card	16 MB	2 in	0.81 in	4.59 oz	choose
Casio	$386.00	7.2 MP	3x	Internal Memory	8.3 MB	2.5 in	0.88 in	4.48 oz	choose
Fuji	$309.18	6.3 MP	3x	XD-Picture Card	16 MB	2.5 in	1.1 in	5.5 oz	choose

they have more optical zoom with different memory type, but are thicker and heavier

Panasonic	$386.00	5.0 MP	12x	SD Memory Card	16 MB	1.8 in	3.34 in	11.52 oz	choose
Konica Minolta	$349.99	5.0 MP	12x	SD Memory Card	16 MB	2 in	3.3 in	12 oz	choose
Fuji	$259.18	4.23 MP	10x	XD-Picture Card	16 MB	1.5 in	3.1 in	11.9 oz	choose
Olympus	$253.00	4.0 MP	10x	XD-Picture Card	16 MB	1.8 in	2.7 in	9.9 oz	choose
Olympus	$284.99	4.0 MP	10x	XD-Picture Card	16 MB	1.8 in	2.7 in	10.6 oz	choose
Nikon	$259.18	4.2 MP	8.3x	Internal Memory	13.5 MB	1.8 in	2.2 in	9 oz	choose

Fig. 13.4: Illustrating the *preference-based organisation* approach to critique presentation (from [12]).

there is no suggested compound critique matching the user's current requirements, even though the proposed critiques can give them some information about subsequent recommendation options (e.g. *greater screen size & more memory*). At this point, they can only apply unit critiques to one feature at a time; at the risk of being involved in longer interaction cycles and lower levels of decision accuracy. They suggest that this limitation could be addressed by allowing users the flexibility to define their *own* compound critiques. The assumption is that only unit critique options that cover subsequent recommendation options are presented to the user (i.e., dynamic unit critiques). The *user-motivated* critiquing approach invites the user to make one or more critique selections over any combination of these category (i.e., compound critique) options.

Apt Decision, proposed by Shearin & Lieberman [64] is an example of an early critiquing system that also implements a user-motivated approach. Users in the apartment rental market are free to critique multiple features (from a set of 21 possibilities) relating to recommended apartment descriptions. Chen and Pu take a different approach, affording the user greater control over what preference constraints influence recommendation. They describe how users have the freedom to specify their tradeoff criteria in terms of improvement and compromise regarding the rec-

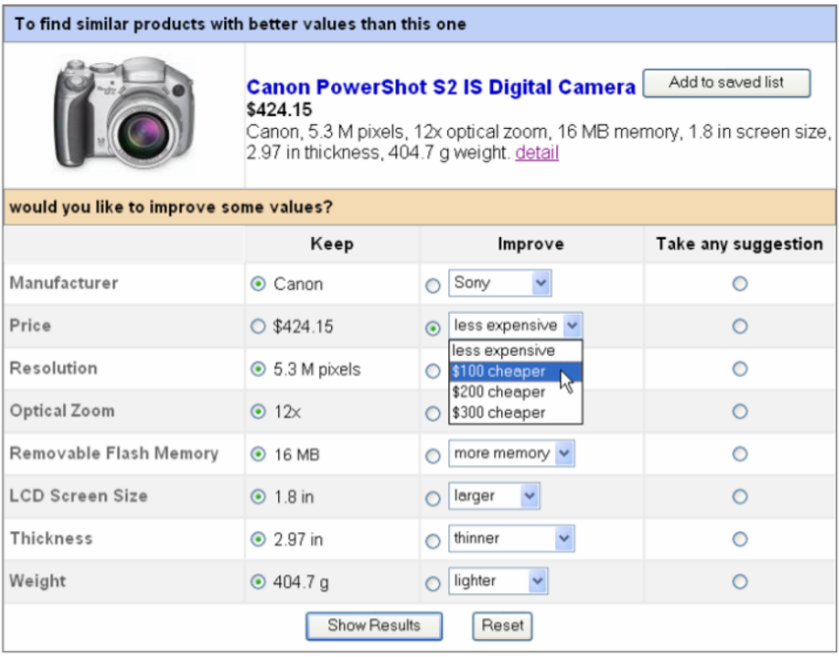

Fig. 13.5: Illustration of user-motivated critiquing interface taken from [9].

ommendation features. For example, users can indicate that they would prefer to see a recommendation that is (e.g., *X amount cheaper* [than the current recommendation]) as shown in Fig. 13.5.

Fig. 13.6 shows how Apt Decision, by contrast, provides direct access to the preference profile where the user can easily indicate preference revisions (both positive and negative). Aside from allowing users to indicate the importance of their feature preferences, Apt Decision does not support users to define any more-specific boundary constraints in terms of the extent of the compromises they are willing to accept. Chen & Pu show that users achieved higher confidence in choice and decision accuracy through user-motivated critiquing. However, some users still preferred the system-proposed critiques, reporting that they found them intuitive to use and quickly led to suitable products when relevant options were presented. *SmartClient* [53, 54, 55] also affords greater user-control over preference elicitation. Designed to help users find airline flights, it has also been used to recommend vacation packages, insurance policies and apartment rentals. Again, product selection is represented as a decision theory problem where the user is trying to select a product that optimally satisfies their preferences. Choosing the best product is often a trade-off problem, where the user must decide how and if he should compromise on some product features in order to optimise others. SmartClient employs critiques as soft constraints. The interface allows the user to directly construct a value function (by specifying weights) for each attribute and the recommender combines this with the constraints

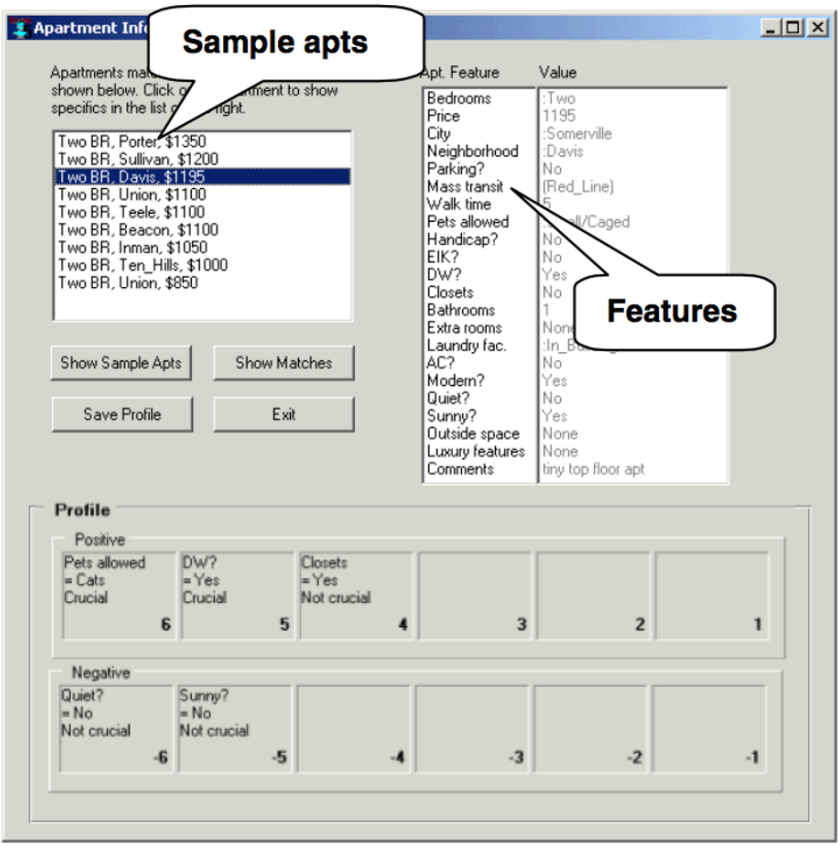

Fig. 13.6: Screen-shot of the Apt Decision user-motivated critiquing interface adapted from [64].

to compute a *utility* for each attribute and product. The system presents a ranked list of the best products with the highest utility. When the user revises his preferences, either through the addition of another critique or by changing its weight, the products are re-ranked to reflect the updated utilities.

13.3.2 Retrieval Challenges in Critique-Based Recommenders

Irrespective of the type of critiques that are used (e.g., unit/compound), or the manner in which the critiques have been generated (e.g., static versus dynamic, system-proposed vs user-motivated), recommendation success is heavily influenced by user critiquing behaviour. As an example, we mentioned earlier that users may *over-critique* a feature (e.g., *price*), and find that they are presented with recommenda-

tions where none of their other feature preferences are satisfied [39]. While little can be done to control how the user chooses to apply critiques, there are some retrieval challenges that *can* be avoided in view of improved recommendation performance. In this section we outline a number of these and briefly discuss how some of them have already been addressed.

13.3.2.1 Preference inconsistency and longevity

The preference model behind the *FindMe* system is a feature vector obtained from the entry example along with the user's initial high-level feature constraints. The application of a critique will update the model with the most recent feature critique and temporarily remove those restaurants that are incompatible with it from the recommendation candidates. The remaining restaurants are then sorted using a hierarchical sort on their similarities to the preference model.

As mentioned earlier in Section 13.2, users cannot be relied upon to provide consistent feedback throughout the course of a recommendation session. Traditional critique-based recommenders (e.g., *RentMe*, *Entrée*) focus on the current critique and the current recommendation, without considering the critiques that have been applied in the past. This can also lead to retrieval failures. Many users are unlikely to have a clear understanding of their requirements at the beginning of a recommendation session. As a result users may select apparently incompatible critiques during a session as they explore different areas of the product space in order to build up a clearer picture of what is available. For example, a given cycle may find a prospective digital camera owner looking for a camera that is *cheaper* than the current €500 recommendation, but later may ask for a camera that is *more expensive* than another €500 recommendation.

Apt Decision [64] maintains more flexible user preference models of user critiques. A key differentiator between it and others is that it has separate positive and negative preference models which are visible and accessible to users. It also supports preference-based comparisons of apartments from which further implicit preference information is gathered and added to the model. Shearin & Lieberman argue that learning an accurate profile is more beneficial than trying to constrain the search-space based on user preferences [64]. If the search-space is over-constrained, users will be unaware of the other potential options that exist containing features that they might be willing to compromise on. Instead, relaxing constraints allows users to explore the product space more, giving the recommender the opportunity to learn more preferences and make better recommendations.

The *SmartClient* system also treats applied critiques as explicit representations of user preferences. Both employ constraint solvers to obtain optimal solutions. User preferences in the form of critiques are modelled as constraints in the CSP formalism. An agent constantly observes the users modifications to the expressed preferences and refines the elicited model to improve solution accuracy. Zhang & Pu [73] describe a MAUT-preference model that adjusts the *utility* of feature preferences based on a user 's critiquing feedback. Recommendations that maximize

overall preference utilities are subsequently retrieved. In similar work, Chen & Pu [12] describe a MAUT-preference modeling approach whereby a user can specify their tradeoffs by directly manipulating criteria.

Reilly *et. al.*'s *incremental critiquing* [58] approach maintains a user preference model which is made up the actual critiques that have been applied by the user so far. The intuition is that the critiques applied so far provides a representation of the user's evolving requirements. To maintain accuracy of the user model *inconsistent* critiques are eliminated, as are all existing critiques for which the most recent critique is a refinement. When retrieving recommendations priority is given to those candidates that: (1) satisfy the current critique; (2) are similar to the previous recommendation; and (3) satisfy a large proportion of previous critiques. [4] Given two candidates that are equally similar to the previously recommended case, their algorithm will prefer the one that satisfies the greater number of recently applied critiques (i.e., that which returns the higher *compatibility* score). Consequentially, they report session-length reductions of up to 70% [58].

Finally, Nguyen & Ricci have more recently motivated the need to maintain alternate preference models that distinguish between long-term and session-specific user preferences. They propose a methodology for integrating both kinds of preference information in order to generate personalised recommendations [41]. Session-specific preferences include both contextual preferences (e.g., a restaurant open at the time in question or within proximity of the user) and product feature preferences (e.g., a cheap chinese restaurant). Long-term user preferences [44] refer to information (about the user) which persists along a relatively long timespan (i.e., through many consecutive recommendation sessions). These preferences are typically elicited at registration time, and revised later through continued system use.

13.3.2.2 Diminishing choices & unreachability

As described earlier, early critique-based recommenders applied critiques act as temporary filters over the remaining available product options and all *critiqued* recommendations are eliminated from further consideration. McSherry & Aha uncover a potential drawback - a problem they refer to as *the diminishing choices problem* [40]. They argue that by preventing users from navigating back to products they critiqued earlier can result in retrieval failures when the *only* acceptable product option has been eliminated. They present the example shown by Fig. 13.7 where a user is presented with Case 1 (i.e., a 3 bedroom detached property in location A). Suppose that the user would *prefer* a 4 bedroom detached property in that same location. Assuming that features are equally weighted and the similarity between two cases is the number of matching features, a critique over the bedroom feature (i.e., $>$ Beds) would see the user presented with Case 2. Realising that there is no case that will satisfy all of their requirements the user may then which to re-evaluate Case 1 how-

[4] In so doing they are implicitly treating the past critiques in the user model as *soft constraints* for future recommendation cycles; it is not essential for recommendations to satisfy all of the previous critiques, but the more they satisfy, the better they are regarded as recommendation candidates.

ever this has been removed they have no choice but to consider Case 3 instead (or re-start their recommendation session).

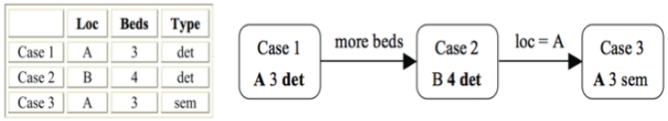

Fig. 13.7: Illustrating the diminishing choices problem that can result as a consequence of eliminating critiqued recommendations [from [40]].

On the contrary, the *unreachability problem* [40, 18] refers to the consequence of *not* eliminating previous recommendations. In [40] McSherry & Aha also demonstrate that this can potentially lead to the situation whereby acceptable products that satisfy the user's requirements, if any, will never be retrieved. This problem is well illustrated by Fig. 13.8 where a user is sequentially presented a recommendation from the catalogue of options shown, and on the fourth critique application is brought back to the initial recommendation without ever being presented with Case 5. An important point here is that recommendation retrieval in this instance is influenced only by the most recently applied critique; that is, the initial query and any previous critiques are not considered.

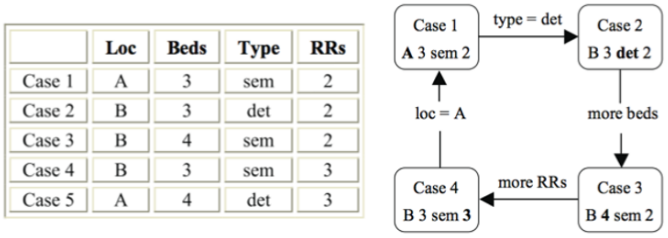

Fig. 13.8: Illustrating the unreachability problem that can result as a consequence of *not* eliminating critiqued recommendations [from [40]].

The *Tweak* system described by McSherry & Aha [39] implements the *progressive critiquing* approach, which like incremental critiquing [58] involves maintaining a history of a user's previous critiques. The recommendation retrieved in response to a user's critique must also satisfy any previous critiques and constraints specified by the initial query. Fig. 13.9 shows how a recommendation option satisfies a users set of *current constraints* which consists of their initial query (i.e., *make=Dell* and *price <= 1000*) and their previous critiques (i.e., *type=laptop* and *< screen size*). Note that *type=laptop* is the only constraint provided by the user at this point that is not satisfied by the current recommendation. The progressive critiquing approach retrieves the most similar product to the current recommendation

and critique, prioritising those preferences contained in the preferences model. This model records *explicit* (E), *assumed* (A), and *predicted* (P) preferences for feature values. Here explicit preferences refer to those set out by the initial query and/or as the result of subsequent replacement critique applications on nominal features (e.g., *type=laptop*). Assumed preferences refer to preferences decisions that are implicitly made in relation to features such as *price* (i.e., where is may be reasonable to assume that *less-is-better* (LIB)) and *memory* (i.e., *more-is-better (MIB))*. And, predicted preferences refer to those features where users tend to have an *ideal* value and would prefer values that are close to this (i.e., *nearer-is-better* (NIB))[37, 35].

Recommended Case	Current Constraints		Preferences	
make = Dell	make = Dell	Y	make = Dell	E
price = 719	price ≤ 1000	Y	price = 658	A
type = desktop	type = laptop	N	type = laptop	E
screen = 15	screen > 12	Y	screen = 13.3	P
speed = 1.6			speed = 2.2	A
memory = 512			memory = 2000	A
hard disk = 20			hard disk = 120	A
chip = Intel Pentium				

Fig. 13.9: Illustrating the knowledge available in a typical progressive critiquing session; current recommendation, current query, and user preferences [from [39]].

Unlike incremental critiquing, revisions to the preference model within the *Tweak* system are made on the basis of the most recent critiquing event without reference to the history of critique applications on that feature. That is, the application of a *make=Sony* critique replaces the value *Dell* within the preference model with *Sony*. For the NIB features, the preferred value is predicted to be that *nearest* to the critiqued recommendation (e.g., a $<$ *screen* critique would result in setting the screen value to 14). Incremental critiquing, by contrast, accounts for the fact that a user may have been presented with a number of *Dell* options over a series of prior recommendation cycles but was content to accept it at that point (i.e, they prioritise other critique changes). The *compatibility* measure used by incremental critiquing takes into account each candidate product's similarity to the current and the number of previous critiques it satisfies. However, a potential problem here is that the existence of a highly similar product may play more influence and be retrieved over alternatives that satisfy more critiques. Similar to Nguyen & Ricci [41], Salamo *et. al.* [63] addresses the issue by using similarity to the critiqued recommendation as a secondary retrieval criterion.

In agreement with Shearin & Lieberman [64], McSherry suggests the system's current understanding of user preferences should be accessible to the user. He feels this is especially important in the interest of avoiding what he terms *progression failures* (i.e., the non-existence of a product that satisfies all of the user's requirements). In Shearin & Lieberman's *Apt Decision* system, the user has constant and

direct access to their preference model. While this provides the user with some depth of understanding as to why certain recommendations are retrieved, it is still open to progression failures. McSherry takes an interesting explanation-based approach where in the event that all of the preference constraints are not met by any recommendation candidate the user is provided with an explanation of what compromises might be appropriate. A more recent version of the progressive critiquing algorithm [40] concentrates on addressing the unreachability problem. The key difference in this implementation of progressive critiquing is that previously recommended (i.e., critiqued) items are not eliminated, as is the case with the incremental approach.

13.3.2.3 Refining recommendation retrievals

Although the majority of critique recommenders, such as *QwikShop*, *EasyShop*, *CritiqueShop*[5], *Entrée*, and *RentMe* concentrate on retrieving only one candidate in each recommendation cycle, others like *Apt Decision* and *MobyRek* present the user with a set of recommendations to choose from. Refining recommendation candidates can be challenging. Early work in this area introduced the *EntréeC* hybrid recommender system [5, 6] which adds a collaborative filtering (see Chapter 5) component to *Entrée* [7], creating a knowledge-based/collaborative cascade hybrid. Like *Entrée*, it uses its knowledge of restaurants to make recommendations based on the user's stated preferences. The collaborative recommender is called upon to refine the competing recommendations returned by the knowledge-based recommender. Key benefits for using the cascading approach here are that it is more efficient than, for example, a weighted hybrid that applies all of its techniques to all items, since the cascade's second step focuses only on those items for which additional discrimination is needed. Furthermore, the cascade is, by nature, tolerant of noise in the operation of a lesser-priority technique, since results returned by the high-priority recommender can only be refined, not overturned. Experiments with *EntréeC* indicate that collaborative filtering does improve the performance over the knowledge-based *Entrée* system acting alone. In [6] Burke discusses the trade-offs that exist between alternate recommendation techniques and reviews some of the combination methods that have been used in other conventional *hybrid* recommender systems, that also would be relevant to critique-based recommenders (see also Chapters 22, 2, and 24).

Faltings *et. al.*[15] argue that a key design question in critiquing systems is *what* recommendations to present users in order to best help them locate their most preferred solution. They propose two key requirements: (1) that the recommendations must stimulate the user to express further preferences (i.e., by showing the range of alternatives available), (2) that presented recommendations must contain the solution that the user would consider optimal (if the currently expressed preference model was complete) so that they could select it as a final solution. The *dynamic critiquing* approach attempts to address these requirements through the provision of

[5] http://www.critiqueshop.com/

compound critiques that describe where the user can navigate to. However, systems that rely on *unit* critiques may find the low feature-level feedback it provides can be insufficiently detailed to sharply focus the next recommendation cycle [33]. For example, by specifying that they are interested in a digital camera with a *greater resolution* than one of the presented recommendations the user is helping the recommender to narrow its search but this may still lead to a large number of available candidates to chose from. Instead a combination of critiquing and value elicitation feedback modes (e.g., Show me a *5 megapixel* camera) is likely to reduce the number of product options much more effectively in this instance.

McGinty & Smyth [32, 33] demonstrate how critique-based recommenders can suffer from protracted recommendation sessions, when compared to value elicitation approaches. As a potential solution they describe a novel *switching* strategy whereby the mode of retrieval is adapted in accordance with user critiquing behaviour. They demonstrate how their *Adaptive Selection* strategy, offers potential dialogue reductions of 60% over standard similarity-based and diversity-enhanced retrieval approaches.

In more recent work, Chen and Pu [9] describe their *example critiquing* approach, recognising that user preferences are often context dependent. Here, multiple recommendations are presented and the user chooses one to critique. They take a *preference-based organisation* [12] approach to recommending the highest utility products and also allow users to freely compose compound critiques over multiple features which indicate the simple and complex trade-offs they are prepared to make.

13.3.2.4 Multi-user preference handing

The task of recommending items/products to a group of users presents a number of challenges (see for example, Chapters 22, 18, 21 and 16 in addition to the works of [3], [47], and [24]). Early work in this area by Jameson [20] highlights a number of these (summarised by Fig. 13.10) and considers how they could be dealt with within the context of a prototype group recommender: *The Travel Decision Forum*. While these systems are susceptible to the same problems as single-user recommenders (e.g., preference inconsistency and volatility) other key distinguishing characteristics include the need to promote *mutual awareness* of individual preferences amongst group members, and *consensus negotiation* (see Section 13.4 for an overview of how the interface plays a critical role here). While the notion of generating a set of recommendations to satisfy a group of *distributed* users with potentially competing interests is challenging in itself, a further challenge is how to record and combine the preferences (and resolve conflicts) for multiple users as they engage in *live* synchronous recommendation dialogs [29]. Key objective questions here include: (1) how can multi-user interaction be managed to facilitate the harvesting of feedback and preferences from multiple simultaneous users?, and (2) how to dynamically maintain models of individual and group preferences with a

view to influencing recommendation such that the resulting suggestions are likely to satisfy *both* the individual and the group?

Recent work by McCarthy *et al.* [29, 31] has concentrated on preference aggregation and consensus negotiation within a critique-based, group recommendation architecture. They introduce the *Collaborative Advisory Travel System* (CATS) designed to provide assistance to a group of friends trying to arrive at a consensus in relation to planning a skiing vacation together. The system supports both individual and multi-user feedback modes through the use of both dynamic unit and compound critiques, and individual and group preference models. It is reasonable to assume that an individual user may need to revisit a previously seen recommendation in this kind of a system in the light of subsequent feedback from other group members. For example, a user may later be willing to compromise on the *price* of a holiday in the knowledge that other members are also willing to do so. *CATS* supports this by facilitating the generation of both *proactive* as well as *reactive* recommendations. *Reactive* recommendations refer to those suggestions presented to the individual user, in response to their own critiques based on their personal user model. *Proactive* recommendations, by contrast, are automatically generated by the system to the group when a recommendation candidate satisfies an unusually high proportion of group preferences; irrespective that a member may have previously rejected this candidate. In addition, individual group members can also identify to the system what they feel are potential recommendation candidates that may be of interest to the whole group. Preference inconsistencies within individual user preference models are managed through incremental critiquing. The group model may contain conflicting preferences and the objective when generating recommendations is that these inconsistencies are minimized by preferring candidates that are maximally compatible multi-user preferences.

	Phase of the recommendation process	Difference from recommendation to individuals	Novel issue
1.	Members specify their preferences.	It may be desirable for members to examine each other's preference specifications.	What benefits and drawbacks can such examination have, and how can it be supported by the system?
2.	The system generates recommendations.	Some procedure for aggregating preferences must be applied.	How can the aggregation procedure effectively discourage manipulative preference specification?
3.	The system presents recommendations to the members.	The (possibly different) suitability of a solution for the individual members becomes an important aspect of a solution.	How can relevant information about suitability for individual members be presented effectively?
4.	Members decide which recommendation (if any) to accept.	The final decision is not necessarily made by a single person; negotiation may be required.	How can the system support the process of arriving at a final decision when members cannot engage in face–to–face discussion?

Fig. 13.10: Challenges facing group recommendation architectures as summarised by Jameson [20].

To the best of our knowledge (at the time of writing) there is no other work in the area of critique-based recommendation that has concentrated on modeling multi-user preferences. As such, in the interest of completeness, we draw some compar-

isons with typical (non-critique-based) group recommenders. Existing multi-user negotiation/collaborative applications range from virtual environments [50] to sales by action [16]. For the most part of these systems assume an automated negotiation that is based on existing static individual preference models in the system. The *live* interactive nature of the *CATS* system renders the use of static preference models inappropriate. In *CATS* individual preference models tend to be especially volatile as a consequence of the influence of multi-user feedback. Other research in the general area of *group* recommendation includes the *MusicFX* System [25]. *MusicFX* is a group preference arbitration system automatically adjusts the selection of music playing in a fitness center to best satisfy the musical tastes of a group in the same environment. *PolyLens* [46] is a generalization of the *MovieLens* system that recommends movies to groups of users though preference modelling. In contrast, it uses collaborative filtering which draws on the historical music preferences of other users in similar contexts. Like *CATS*, the *Travel Decision Forum* [20] helps a group of users to agree on a vacation that they are planning to take together. However, it concentrates on supporting users who are not co-located. In other related work, Plua & Jameson [49] propose a group recommendation approach where users can get help from others in their group about preferences when their domain knowledge may be incomplete. Unlike CATS, this system was intended for use by a group that interact asynchronously rather than simultaneously.

13.4 Interfacing Considerations Across Critiquing Platforms

Different domain and platform characteristics present recommender interface designers with very different technical and usability challenges. In this section we concentrate on design decisions that have been implemented within existing critique-based recommenders (see also Chapter 16). Unsurprisingly, a common theme is how best to manage transparency and control, while also ensuring the level of cognitive and interaction effort required of the user is kept to a minimum.

13.4.1 Scaling to Alternate Critiquing Platforms

While the majority of critique-based recommenders assume desktop web-based platforms (e.g., *The FindMe Systems*, *QuickShop*, *EasyShop*, *CritiqueShop*, *AptDecision*, *SmartClient*), critiquing has also been demonstrated to be an effective interfacing methodology across other platform settings. For instance, Ricci & Nguyen [61, 43, 62] concentrate on interface design and evaluation for critique-based recommenders targeted at mobile devices. They point out that very few web-based systems are designed for mobile users; none of them being conversational. Persistent direct manipulation of the interface is not practical here. Key challenges include: that these devices have much smaller screen-sizes, less keypad functionality than traditional

PDA's (e.g., Palm or Pocket PC), and limited computing power. Their *MobyRek* mobile portal interfaces with a web-based recommender that helps users make travel plans in advance. Essentially, *MobyRek* offers the *on-tour* support required when a user is traveling to or has arrived at their destination.

Staying in the mobile space, Fig. 13.11 illustrates how the *CritiqueShop* interface has been scaled to the iPhone. The key considerations influencing design were the limited screen area and increased opportunity for direct user manipulation (bearing in mind that the iPhone has a touch-sensitive interface). Another design decision made here was to move away from text-based compound critique representation, preferring instead to present more visual alternatives (this is discussed later).

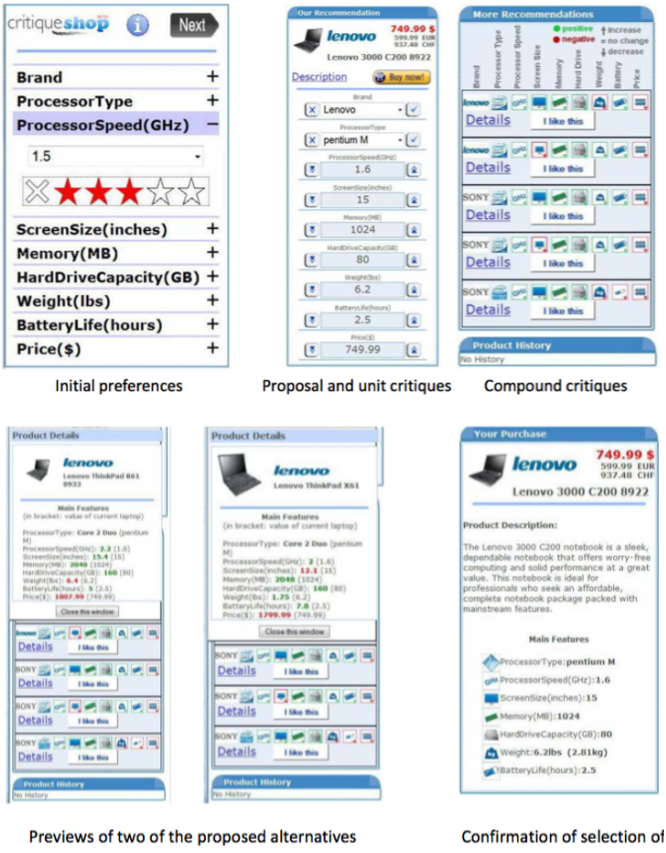

Fig. 13.11: Screen-shot of the CritqueShop visual interface for the iPhone [www.critiqueshop.com] .

In other work, McCarthy et. al. [30] describe how the *CATS* group recommender operates on the interactive, multi-user MERL *DiamondTouch*[6] table-top device. The DiamondTouch table is a multi-user, touch-and-gesture-activated screen for supporting small group collaboration. Users interact with the display simultaneously (i.e., without having to take turns), browsing through the potential ski holiday options and critiquing recommendations returned to them (i.e., recommendations that are influenced by the evolving individual and group member critiquing patterns, as discussed earlier). Users can also copy and paste potential options of interest to other users, as well as confer on a face-to-face basis about their preferences as they interact in this unusual manner.

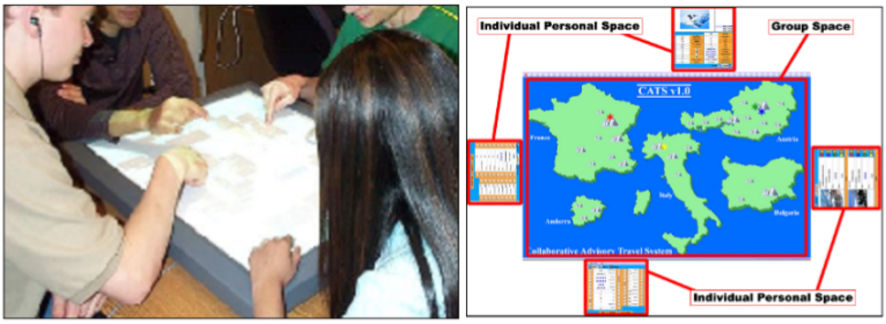

Fig. 13.12: Illustrating the illustrating the CATS interaction with the Diamond Touch.

13.4.2 Direct Manipulation Interfaces vs Restricted User Control

A fundamental consideration that has influenced the design of critiquing interfaces is the importance of finding the right balance between eliciting precise preference information and the cost associated with acquiring it. Early critique-based recommenders offered very restricted user control (i.e., *static* critiques were presented by the system and the user could only select *one* to apply). Later systems, such as *QwikShop, EasyShop* and *CritiqueShop*, offer the user a little more interfacing control through the provision of alternate (i.e., unit/compound) critiquing modes to choose from, but do not allow the user to directly revise their preference model (e.g., *Apt Decision*), or set explicit constraint boundaries on certain features (e.g., as is the case in *SmartClient*).

Chen and Pu [11] demonstrate how their hybrid interface supports both *user-motivated* and *system-proposed critiquing* and enables users to achieve a higher

[6] The DiamondTouch product line has moved operations from the MERL research lab and into a separate company called Circle Twelve Inc.

level of decision accuracy and interfacing satisfaction, while consuming less cognitive effort, than that of a standalone system-proposed critiquing interface. Instead of suggesting pre-computed critiques for users to choose, the self-motivated critiquing approach focuses on showing examples and stimulating users to make unit or compound critique selections over any combination of features as they wish. Importantly, users also have the freedom to specify their tradeoff criteria in terms of improvement and compromise regarding the individual features, and see a new set of products better approaching their ideal choice (see previous Fig. 13.5). McSherry [38] seeks a middle-ground by introducing the notion of direct, user-motivated *relaxation* critiques in addition to system-generated relaxation mechanisms in a progressive critiquing recommender. This mixed-initiative approach implemented by the *Tweak 2* system allows the user to request an item *like* the current recommendation but with no restriction on the value of a particular (previously critiqued) feature of their choosing.

Mobile platforms present the user with a very limited control and interfacing license [42]. In addition, telecommunication service costs tend to discourage users from engaging in lengthy interaction sessions. In this space it is usual to measure efficiency by the number of clicks, scrolls, and keypad actions, the user needs to perform. User-control is often sacrificed in view of keeping costs low (i.e, time and money). Two key goal requirements that influenced the adoption of critiquing within MobyRek as a suitable direct feedback mechanism were: (1) useful preference information needed to be captured within a very small screen area, and (2) the user-system interaction had to be low-cost; that is, requiring minimal time to obtain a useful recommendation. Evaluations demonstrate that users are typically recommended acceptable recommendation options[7] within 2-3 cycles/clicks [42, 61].

13.4.3 Supporting Explanation, Confidence & Trust

There are a number of Chapters in this handbook that are relevant to this heading (see for example, Chapters 20, 14, 15, 16, and 25). A recommender system's ability to establish trust with users and convince them of its recommendations, such as which camera or PC to purchase, is highlighted as a crucial design factor [51, 52]. If users perceive it to be capable and efficient at assisting them to make decisions, they are more likely to return to the interface. A number of researchers have developed design principles and strategies for building trust in critique-based recommenders through the explanation of recommendations to users (e.g., see [52, 57, 66]). Generally a product recommender may use explanations to explain: (1) the reasons *why* a particular product was (or *was not*) recommended [36], and/or (2) *what* opportunities remain: that is, *"where can I get to from here"*, when presented with an unsuitable recommendation [57]. In the first instance, explaining why a product was recommended can be simply managed by showing the user the information con-

[7] For the purpose of the evaluation a mobile restaurant application is presented whereby location information is collected using the GPS receiver connected via Bluetooth

Fig. 13.13: The MobyRek Mobile Restaurant Recommender [61].

tained in the user model, as is the case in with *Apt Decision* (see Fig.13.6) by way of *justification* [66]. McSherry [36, 37] highlights the greater importance of explaining the cause of any retrieval failure when a recommendation does not satisfy a user's requests. He describes a mixed-initiative approach to recovery from retrieval failure by highlighting subsets of query features that cannot be satisfied such that the user might revise their constraint boundaries (e.g. *"there are no cameras with price less than €300 and resolution greater than 4 mega-pixels"*).

Reilly *et. al.* [57] argue that dynamic compound critiques help the user to better understand the *recommendation opportunities* that exist beyond the current cycle by helping them to appreciate common interactions between features (i.e., explanation through *increased system transparency* [66]). In many recommender domains, where the user is likely to have incomplete knowledge about the finer details of the feature-space, compound critiques will help to effectively map out this space and minimise decision error. For instance, with standard critiquing in the digital camera domain a user might naively select the $[Price, <]$ unit critique in the mistaken belief that this may deliver a cheaper camera that satisfies all of their other requirements. However, reducing price in this way may lead to a reduction in resolution that the user might not find acceptable and, as a result, they will have to backtrack. Hadzic & O'Sullivan [18] highlight a further potential problem here - that is, critiques suffer from a lack of symmetry that may prove to be counter-intuitive to a user. So, attempting to undo a critique by applying its opposite may not work like using the *back* button on a web-browser. This problem is less likely to occur if the compound critique $\{[Price, <], [Resolution, <]\}$ is presented because the user will come to understand the implications of a price-drop prior to selecting any critique. In addition, *QwikShop* reserves an area of the interface which provides further explanation support. The user is given information about the number of products that

relate to each presented critique option and the feature value ranges for each that it covers as illustrated by Fig. 13.2.

Providing system transparency through explanation allows users to more confidently assess the reliability of a system (i.e. increased user-confidence). Complementary to the concept of explanation is the concept of *system confidence* in a recommendation. In order to fully maximise user confidence in a recommendation, the system itself should be capable of assessing its own confidence, or lack thereof, in the recommendation. Reilly *et. al* [59] propose a methodology for modeling confidence at the system-level designed to work with critique-based recommenders. By informing the user of how confident the system itself is in a recommendation, the user can better judge how much trust to place in the recommendation. They propose a feedback influenced model that calculates measures of system confidence at both the feature and product-levels. A low system confidence score for the *price* feature, for example, would translate that the system is uncertain if the recommendation focus is concentrating on the correct price range for that user. Once users have a clearer understanding of those features needing clarification, they might be more inclined to refocus their feedback (i.e, refine and improve their preference model). The authors also demonstrate how product-level confidence scores supplement existing similarity knowledge, in order to guide the recommender towards more confident suggestions.

13.4.4 Visualisation, Adaptivity, and Partitioned Dynamicity

While Chapter 17 highlights the benefits of using visual interfaces in product recommenders here we concentrate on critique-based product recommenders. Zhang *et. al.* [72] introduce a visual interface where compound critiques are represented by meaningful icons as opposed to through plain text. Fig. 13.14 shows an example of how a single compound critique is represented by both approaches. Their studies demonstrate that users are more likely to apply visual compound critiques over the textual form (i.e., recording application frequency improvements of nearly 50%), and subsequently benefit from reduced interaction times (e.g., reductions in session length of up to 53%)[8]. Fig. 13.11 illustrates how the visual interface proposed by Zhang *et. al* operates on the iPhone.

Other research has demonstrated the power of highly adaptive visual interfaces that support the dynamic change of interface icons in response to user critiquing feedback. Specifically, Averjanova et. al. [2] demonstrate how recommendation effectiveness (e.g., average session length reductions of 17%) and user satisfaction can be improved in the *MobyRek* system [61] through the integration of a map-based visualisation interface. Similarly, the usability studies of *CATS* group recommender [30] show that the dynamic adaptive interface promotes the mutual awareness of changing multi-user preferences. A key design consideration common to both sys-

[8] Results refer to data gathered using a version of the system that operated over a laptop dataset.

Fig. 13.14: Illustrating textual and visual representations of a single compound critique (from [72]).

tems was how best to address the problem of *limited critique influence on recommendations* whereby it was hard for the user to see the effect of a critique. This problem was identified in both cases through real-user usability evaluations. Both systems address this in a very similar way though the the use of dynamic interfacing components (i.e., icon resizing and colour coding recommendations). For example, in CATS if the level of interest in a particular holiday resort is high amongst the group then this resort is resized to be larger than those that are of lesser interest. Another concern reported by users of the early MobyRek interface was *limited distance perception* whereby it was difficult for users to compare distances to restaurants from their current position as they were on the move. A further extension in the revised *MapMobyRek System* included a map-based interface to address this problem. Other extensions included colour-coding to represent the degree of suitability of recommendation and the functionality to support side-by-side item comparisons.

Fig. 13.15: Illustrating the visualisation interface of the MapMobyRek Mobile Recommender [2].

Like *MobyRek*, *CATS* implements a map-based interface, however an additional design consideration was how best to communicate progress towards consensus

agreement to all group members. This is effectively achieved through the use of highly visual color-coded consensus barometers that summarize evolving user opinions on competing candidate vacation options. A further concern when designing adaptive interfaces that change rapidly *in-session* is the risk of user confusion. *QwikShop, EasyShop, CritiqueShop* partition the dynamic and static elements of their interfaces [69]. This was also a key design consideration in the *CATS* recommender where it was necessary to keep members aware of each other's preferences and motivational orientations, without confusion. This is managed through the use of intuitive and distinct, *shared* and *individual* spaces and careful design of visual cues. In addition, a range of dynamic interfacing components are introduced to communicate information about evolving group preferences and monitor progress towards reaching a group consensus.

13.4.5 Respecting Multi-cultural Usability Differences

It is well documented that aspects of a user interface that are appropriate for one culture may not be suitable for another (see for example, [21], [23], [45]). In recent related work, Chen and Pu present a comprehensive cross-cultural evaluation of web-based critiquing interfaces [13]. Specifically, they compare user responses to two strategies for displaying e-commerce (i.e, laptop) recommendations: (1) as a ranked ordered list of items where each item has an explanation as to why it was retrieved, and (2) as a *preference-based organization*, whereby groups of recommendations are categorised and summarized in terms of their collective differences/trade-offs relative to the top ranked product. Very briefly, subjective evaluations over 120 participants (60 western culture/60 oriental culture) have shown that organisational view had the most impact on all users, in terms of how they perceived recommendation quality and their overall satisfaction. For a more comprehensive breakdown of comparisons regarding subjective perceptions please refer to [13].

13.5 Evaluating Critiquing: Resources, Methodologies and Criteria

In this section we first point to some of the resources that have been commonly used for evaluation purposes in this area, and provide details of where they can be accessed. Next, we summarise the typical evaluation methodology and outline some of the key evaluation criteria that are commonly used.

13.5.1 Resources & Methodologies

Those starting out in this area should be aware that a large number of the datasets that have been used for evaluation purposes in the published research are freely available for download[9]. Examples include the *Travel* (e.g., as used by [26, 58, 34]), *PC* (e.g., as used by [58, 39, 40, 34]), *Whiskey* (e.g. as used by [34]), and *Digital Camera* (used by [27]) datasets. The *Entrée* data is also freely accessible[10]. In other work, apartment [55, 73] and ski-holiday [29] datasets have also been used but these are not publicly available.

In general, most evaluation methodologies that are common to recommender systems evaluation can be applied here (see Chapters 8 and 15). In the ideal case, *live* usability studies and performance evaluations should be ultimately carried with real users when evaluating the performance (both subjective and objective) of critique-base recommenders. However, it can be difficult to recruit sufficient numbers of volunteers to participate in multiple trials. As a solution here, it is common for simulated studies to be conducted (the results of which are expected to be later validated in a real-user setting) across a range of different datasets (such as those mentioned earlier), and artificial user profiles. If sufficiently well designed, these off-line simulations can be reliable indicators of real-world performance. A common methodology that has been widely adopted in the literature on critique-based research when conducting performance evaluations is the *leave-one-out methodology* [33]. Very briefly, by this methodology, each product of a dataset is set as a recommendation *target* and is temporary removed from the dataset. A subset of its features is chosen as the initial query. In each recommendation cycle the critique applications are applied such that they concur with the features of the *target*. The simulated recommendation session ends when the most similar product to the ideal product is recommended. Importantly, experimental setups should always have a corresponding *control* setup (e.g., using an alternate retrieval strategy, or feedback approach) to allow for the appropriate assessment of the significance of results. Unfortunately, the only acceptable way to evaluate usability is through real-user interaction trials.

13.5.2 Evaluation Criteria

Common objective performance measures that have been used by evaluations of critiquing research include:

Efficiency: Positively demonstrated by reductions in session length (i.e., the number of critiquing cycles/interactions) that a user engages in before they find their target product (see [39, 40, 73, 38, 72, 13, 61, 58, 60, 2, 27]) , the amount of time taken to reach a *target* recommendation [10, 13, 2, 68].

[9] URL - http://cbrwiki.fdi.ucm.es/wiki/index.php/Case$_{\text{B}}$ases

[10] URL - http://kdd.ics.uci.edu/databases/entree.data.html

Accuracy: There are many ways to measure correctness. Some of these include:
Critique Prediction Accuracy: Refers to the number of times a presented critique *matches* that which was selected by a real user [12, 60, 6].
Critique Application Frequency: Refers to the proportion of the time critiques are applied by the user (i.e., their relevance to the preference model) [73, 72, 60, 11, 27].
Recommendation Accuracy: Measured by Chen & Pu in [12] as the % of cycles where following the application of an applied critique the *target* recommendation was *reachable* (i.e., located in the recommended products), and by [61, 40, 37, 68] as the % of successful sessions.
Decision Accuracy: Refers to the proportion of the time where the recommendation that was ultimately accepted by the user was actually the best solution. Measured in [9, 11] as % of cycles where the user changed their mind once they were shown all the alternatives. Measured by [44] as the degree of position displacement.

Interaction Effort: Usually refers to average session *click-distance* or the number of interactions (i.e., critiques) to arrive at a given target. Assuming one critique is applied in each cycle will be equivalent to measuring session length [12, 10, 13, 11].

Real-user usability performance evaluations of critiquing research has commonly surveyed participants and sought their subjective feedback on a *Likert* scale over criteria such as: (see for example, [9, 10, 62, 61, 13, 60, 26, 11, 2]).

System Design:

System Transparency:	Does the user understand why recommendations were made?
User Control:	Did the user feel that they had control over the specification over their preferences throughout the interaction?

Competence

Perceived Ease of Use:	Did the user find the system easy to use?
Perceived Usefulness:	Did the user feel that the system was useful (e.g., relevant recommendations, good explanations, etc.).
Decision Confidence:	How confident was the user that they found the *best* product for them?
Perceived Effort:	How easily did they find the information they were looking for?
Perceived Accuracy:	Where the suggestions accurate?

Trust

Satisfaction:	How satisfied was the user with the interaction?
Recommendation trust:	Did the user feel that the recommender consistently provided suggestions that were suited to their preferences?

User Intention

Purchase Intention:	Would the user purchase this product if given the opportunity?
Return Intention:	Would the user return to use the system (over another)?

13.6 Conclusion / Open Challenges & Opportunities

In this chapter we have presented, a non-technical overview of the evolution of critiquing research over the past decade. We believe that this could serve as a useful reference for researchers starting out in this area. We have described the benefits of the approach as they apply to a number of existing critiquing systems. Key issues and challenges brought into focus by the community have been discussed in terms of how advances have been made towards: (1) automated critique presentation in view of optimising preference acquisition, and (2) the improved retrieval performance of critique-based recommenders. In addition, we have presented an overview of the various design considerations that have influenced the integration of the critiquing mode of feedback across alternate platforms and user environments. Finally, we summarised some of the popular evaluation criteria that tend to be used when evaluating critique-based systems.

Although there has been a considerable level of research activity in the area of critiquing there are still many open challenges and opportunities. There are far too many to cover here but by way of concluding the chapter we will highlight a few examples. First, the majority of existing systems have assumed that users will execute only *positive critiques* that in the direction of *preferred* recommendation items (e.g., *Show me more like item A, but cheaper*). Sometimes it may make sense for a user to apply *negative critiques*, such as *Do not show me any more like X that are in location Y* as these are arguably just as important for the system when understanding the user's needs and preferences.

There has not been a lot of work on the topic of *entry-point* decision making, whereby the focus is on what recommendations to present first (see [15, 41]). Recent work by Hadzic & O'Sullivan introduced the notion of a *critique graph* as a formal basis for reasoning about the set of products that can be *reached* using critiquing from a given recommendation. They propose that a useful basis for calculating the best *entry recommendations* to select is to use the concept of *minimum catalogue cover* (i.e., identify the set of recommendation candidates from which every other recommendation can be reached if critiqued optimally). It is important to note that not all recommendations are reachable from a given recommendation candidate. In a similar vein, the authors describe how certain recommendations can exist that are not covered by *any* other product (i.e., that are not reachable through any series of critique applications). In this situation a useful principle might be to prioritise inclusion of these in the entry set. This idea has not been implemented or explored further by existing research as yet, although it has very good promise.

The challenge of modeling user preferences gathered from critiquing feedback is still a topic that a lot of current research continues to explore. Other approaches to modeling user preferences and comparative evaluations of existing approaches would be very interesting. Also, while recent work has looked at the idea of maintaining separate long-term and short-term preference models [44], there has been no work that has concentrated on how information might be transferred and maintained between them, and what the benefits/consequences of doing this might be. The investigation of further approaches for the aggregation of multi-user preferences is

another open challenge, as very little work has been carried out in this area to date. While early work in group critique-based recommenders [29] has demonstrated the benefits of one such approach in the context of a collaborative environment, assuming synchronous feedback, other opportunities remain. Examples include, the extension of critiquing to other asynchronous, and/or non-collaborative environments (perhaps where multiple users may interact in a non co-operative fashion). Such research could lead to the adoption of critiquing as an interaction mode in application areas currently unexplored by critiquing research such as game-play, for example.

References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules in large databases. Advances in Knowledge Discovery and Data Mining pp. 307–328 (1996)
2. Averjanova, O., Ricci, F., Nguyen, Q.N.: Map-based interaction with a conversational mobile recommender system. In: UBICOMM '08: Proceedings of the 2008 The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, pp. 212–218. IEEE Computer Society, Washington, DC, USA (2008)
3. Baatarjav, E.A., Phithakkitnukoon, S., Dantu, R.: Group recommendation system for facebook. In: OTM '08: Proceedings of the OTM Confederated International Workshops and Posters on On the Move to Meaningful Internet Systems, pp. 211–219. Springer-Verlag, Berlin, Heidelberg (2008)
4. Burke, R.: The wasabi personal shopper: A case-based recommender system. In: Proceedings of the 11th National Conference on Innovative Applications of Artificial Intelligence, pp. 844–849. AAAI Press (1999). Menlo Park, CA, USA
5. Burke, R.: A case-based reasoning approach to collaborative filtering. In: E. Blanzieri, L. Portinale (eds.) Proceedings of the Fifth European Conference on Case-Based Reasoning, EWCBR '00, pp. 370–379. Springer (2000). Trento, Italy
6. Burke, R.: Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction **12**(4), 331–370 (2002)
7. Burke, R., Hammond, K., Young, B.: Knowledge-based navigation of complex information spaces. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, pp. 462–468. AAAI Press/MIT Press (1996). Portland, OR
8. Burke, R.D., Hammond, K.J., Young, B.C.: The findme approach to assisted browsing. IEEE Expert: Intelligent Systems and Their Applications **12**(4), 32–40 (1997)
9. Chen, L., Pu, P.: Evaluating critiquing-based recommender agents. In: In Proc. AAAI 2006, pp. 157–162 (2006)
10. Chen, L., Pu, P.: The evaluation of a hybrid critiquing system with preference-based recommendations organization. In: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, pp. 169–172. ACM, New York, NY, USA (2007)
11. Chen, L., Pu, P.: Hybrid critiquing-based recommender systems. In: IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces, pp. 22–31. ACM, New York, NY, USA (2007)
12. Chen, L., Pu, P.: Preference-Based Organization Interfaces: Aiding User Critiques in Recommender Systems, pp. 77–86. Springer-Verlag, Berlin, Heidelberg (2007)
13. Chen, L., Pu, P.: A cross-cultural user evaluation of product recommender interfaces. In: RecSys '08: Proceedings of the 2008 ACM Conference on Recommender systems, pp. 75–82. ACM, New York, NY, USA (2008)

14. Cohen S., Rokach L., Maimon O., Decision Tree Instance Space Decomposition with Grouped Gain-Ratio, Information Science, Volume 177, Issue 17, pp. 3592-3612 (2007)
15. Faltings, B., Pu, P., Torrens, M., Viappiani, P.: Designing example-critiquing interaction. In: IUI '04: Proceedings of the 9th international conference on Intelligent user interfaces, pp. 22–29. ACM, New York, NY, USA (2004)
16. Faratin, P., Sierra, C., Jennnings, N.: Using similarity criteria to make issue trade-offs in automated negotiations. Artificial Intelligence **142**(2), 205–237 (2002)
17. Ha, V., Haddawy, P.: Problem-focused incremental elicitation of multi-attribute utility models. In: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, (UAI-97), pp. 215–222. Morgan Kaufmann (1997). URL `citeseer.ist.psu.edu/ha97problemfocused.html`. San Francisco
18. Hadzic, T., O'Sullivan, B.: Critique graphs for catalogue navigation. In: RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems, pp. 115–122. ACM, New York, NY, USA (2008)
19. Hammond, K., Burke, R., Schmitt, K.: A case-based approach to knowledge navigation. In: D. Leake (ed.) Case-Based Reasoning Experiences, Lessons and Future Directions., pp. 125–136. AAAI Press (1996)
20. Jameson, A.: More than the sum of its members: Challenges for group recommender systems. In: AVI '04: Proceedings of the working conference on Advanced visual interfaces, pp. 48–54. ACM, New York, NY, USA (2004)
21. Lee, K., Joshi, K., McIvor, R.: Understanding multicultural differences in online satisfaction. In: SIGMIS-CPR '07: Proceedings of the 2007 ACM SIGMIS CPR conference on Computer personnel research, pp. 209–212. ACM, New York, NY, USA (2007)
22. Linden, G., Hanks, S., Lesh, N.: Interactive assessment of user preference models: The automated travel assistant. In: C.P.A. Jameson, C. Tasso (eds.) User Modeling: Proceedings of the Sixth International Conference, pp. 67–78. Springer Wien (1997)
23. Lodge, C.: The impact of culture on usability: Designing usable products for the international user. In: N.M. Aykin (ed.) Usability and Internationalization. HCI and Culture, Second International Conference on Usability and Internationalization, UI-HCII 2007, Held as Part of HCI International 2007, Beijing, China, July 22-27, 2007, Proceedings, Part I, pp. 365–368. Springer (2007)
24. Masthoff, J., Gatt, A.: In pursuit of satisfaction and the prevention of embarrassment: Affective state in group recommender systems. User Modeling and User-Adapted Interaction **16**(3-4), 281–319 (2006)
25. McCarthy, J., Anagnost, T.: Musicfx: An arbiter of group preferences for computer aupported collaborative workouts. In: Proc. of Conference on Computer Supported Cooperative Work, pp. 363–372 (1998)
26. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: On the dynamic generation of compound critiques in conversational recommender systems. In: Adaptive Hypermedia and Adaptive Web-Based Systems, Third International Conference, AH 2004, Eindhoven, The Netherlands, August 23-26, 2004, Proceedings, *Lecture Notes in Computer Science*, vol. 3137, pp. 176–184. Springer (2004)
27. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: Experiments in dynamic critiquing. In: IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces, pp. 175–182. ACM, New York, NY, USA (2005)
28. McCarthy, K., Reilly, J., Smyth, B., McGinty, L.: Generating diverse compound critiques. Artif. Intell. Rev. **24**(3-4), 339–357 (2005)
29. McCarthy, K., Salamó, M., Coyle, L., McGinty, L., Smyth, B., Nixon, P.: Cats: A synchronous approach to collaborative group recommendation. In: G. Sutcliffe, R. Goebel (eds.) Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference, Melbourne Beach, Florida, USA, May 11-13, 2006, pp. 86–91. AAAI Press (2006)
30. McCarthy, K., Salamó, M., Coyle, L., McGinty, L., Smyth, B., Nixon, P.: Group recommender systems: A critiquing based approach. In: IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces, pp. 267–269. ACM, New York, NY, USA (2006)

31. McCarthy, K., Salamó, M., McGinty, L., Smyth, B.: The needs of the many: A case-based group recommender system. In: ICCBR '06: Proceedings of the 11th international conference on Intelligent user interfaces, pp. 196–210. Springer LNCS (2006)
32. McGinty, L., Smyth, B.: The role of diversity in conversational systems. In: D. Bridge, K. Ashley (eds.) Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03). Springer (2003). Troindheim, Norway.
33. McGinty, L., Smyth, B.: Tweaking critiquing. In: Proceedings of the Workshop on Personalization and Web Techniques at the International Joint Conference on Artificial Intelligence (IJCAI-03). Morgan-Kaufmann (2003). Acapulco, Mexico
34. McGinty, L., Smyth, B.: Adaptive selection: An analysis of critiquing and preference-based feedback in conversational recommender systems. Int. J. Electron. Commerce **11**(2), 35–57 (2006)
35. McSherry, D.: Similarity and compromise. In: K. Ashley, D. Bridge (eds.) Case-Based Reasoning Research and Development. LNAI Vol.2689, pp. 291–305. Springer (2003)
36. McSherry, D.: Explaining the pros and cons of conclusions in cbr. In: P. Funk, P.A. González-Calero (eds.) Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings, pp. 317–330. Springer (2004)
37. McSherry, D.: Retrieval failure and recovery in recommender systems. Artif. Intell. Rev. **24**(3-4), 319–338 (2005)
38. McSherry, D., Aha, D.: Mixed-initiative relaxation of constraints in critiquing dialogues. In: ICCBR '07: Proceedings of the 7th international conference on Case-Based Reasoning, pp. 107–121. Springer-Verlag, Berlin, Heidelberg (2007)
39. McSherry, D., Aha, D.W.: Avoiding long and fruitless dialogues in critiquing. In: M. Bramer, F. Coenen, A. Tuson (eds.) Research and Development in Intelligent Systems XXIII. BCS Conference Series, pp. 173–186. Springer, London (2006)
40. McSherry, D., Aha, D.W.: The ins and outs of critiquing. In: M.M. Veloso (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India, January 6-12, 2007, pp. 962–967 (2007)
41. Nguyen, Q., Ricci, F.: User preferences initialization and integration in critique-based mobile recommender systems. In: In Proc. 5th Int'l Workshop Artificial Intelligence in Mobile Systems (AIMS 04),, p. pp. 7178 (2004)
42. Nguyen, Q., Ricci, F., Cavada, D.: Critique-based recommendations for mobile users: Gui design and evaluation. In: In Proceedings of the 3rd International Workshop on HCI in Mobile Guides, in conjunction with the 6th International Conference on Mobile Human-Computer Interaction (2004). Glasgow, Scotland.
43. Nguyen, Q.N., Ricci, F.: Replaying live-user interactions in the off-line evaluation of critique-based mobile recommendations. In: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, pp. 81–88. ACM, New York, NY, USA (2007)
44. Nguyen, Q.N., Ricci, F.: Long-term and session-specific user preferences in a mobile recommender system. In: IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces, pp. 381–384. ACM, New York, NY, USA (2008)
45. Noiwan, J., Norcio, A.F.: Cultural differences on attention and perceived usability: Investigating color combinations of animated graphics. Int. J. Hum.-Comput. Stud. **64**(2), 103–122 (2006)
46. O'Connor, M., Cosley, D., Konstan, J., Riedl, J.: Polylens: A recommender system for groups of users. In: Proc. of European Conference on Computer-Supported Cooperative Work, pp. 199–218 (2001)
47. Park, Y.J., Chang, K.N.: Individual and group behavior-based customer profile model for personalized product recommendation. Expert Syst. Appl. **36**(2), 1932–1939 (2009)
48. Payne, J., Bettman, J., Johnson, E.: The Adaptive Decision Maker. Cambridge University Press (1993)
49. Plua, C., Jameson, A.: Collaborative preference elicitation in a group travel recommender system. In: Proceedings of the AH 2002 Workshop on Recommendation and Personalization in eCommerce, pp. 148–154. Malaga, Spain (2002)

50. Prada, R., Paiva, A.: Believable groups of synthetic characters. In: Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 37–43. The Netherlands (2005)
51. Pu, P., Chen, L.: Trust building with explanation interfaces. In: IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces, pp. 93–100. ACM, New York, NY, USA (2006)
52. Pu, P., Chen, L.: Trust-inspiring explanation interfaces for recommender systems. Know.-Based Syst. **20**(6), 542–556 (2007)
53. Pu, P., Faltings, B.: Personalized navigation of heterogeneous product spaces using smart-client. In: IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces, pp. 212–213. ACM, New York, NY, USA (2002)
54. Pu, P., Faltings, B.: Decision tradeoff using example-critiquing and constraint programming. Constraints **9**(4), 289–310 (2004)
55. Pu, P., Z., H., Kumar, P.: Evaluating example-based search tools. In: EC '04: Proceedings of the 5th ACM conference on Electronic commerce, pp. 208–217. ACM, New York, NY, USA (2004)
56. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic critiquing. In: P. Funk, P.A. González-Calero (eds.) Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings, *Lecture Notes in Computer Science*, vol. 3155, pp. 763–777. Springer (2004)
57. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Explaining compound critiques. Artif. Intell. Rev. **24**(2), 199–220 (2005)
58. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Incremental critiquing. Knowl.-Based Syst. **18**(4-5), 143–151 (2005)
59. Reilly, J., Smyth, B., McGinty, L., McCarthy, K.: Critiquing with confidence. In: H. Muñoz-Avila, F. Ricci (eds.) Case-Based Reasoning, Research and Development, 6th International Conference, on Case-Based Reasoning, ICCBR 2005, Chicago, IL, USA, August 23-26, 2005, Proceedings, *Lecture Notes in Computer Science*, vol. 3620, pp. 436–450. Springer (2005)
60. Reilly, J., Zhang, J., McGinty, L., Pu, P., Smyth, B.: A comparison of two compound critiquing systems. In: IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces, pp. 317–320. ACM, New York, NY, USA (2007)
61. Ricci, F., Nguyen, Q.N.: Critique-based mobile recommender systems,. GAI Journal, GAI Press **Volume 24, Number 4** (2004)
62. Ricci, F., Nguyen, Q.N.: Acquiring and revising preferences in a critique-based mobile recommender system. IEEE Intelligent Systems **22**(3), 22–29 (2007)
63. Salamo, M., Reilly, J., McGinty, L., Smyth, B.: Improving incremental critiquing. In: In Procedings of the 16th Conference on Artificial Intelligence and Cognitive Science (AICS05), pp. 379–388 (2005)
64. Shearin, S., Lieberman, H.: Intelligent profiling by example. In: IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces, pp. 145–151. ACM, New York, NY, USA (2001)
65. Smyth, B., McGinty, L.: Improving the Performance of Recommender Systems that Use Critiquing, chap. Intelligent Techniques for Web Personalization, pp. 114–132. 978-3-540-29846-5. Springer (2005)
66. Sørmo, F., Cassens, J., Aamodt, A.: Explanation in case-based reasoning-perspectives and goals. Artificial Intelligence Review **24**(2), 109–143 (2005)
67. Spiekermann, S., Paraschiv, C.: Motivating Human-Agent Interaction: Transferring Insights from Behavioral Marketing to Agent Design, pp. 255–285. Kluwer Academic Publishers (2002)
68. Viappiani, P., Faltings, B., Pu, P.: Preference-based search using example-critiquing with suggestions. Journal of Artificial Intelligence Research **27**, 2006 (2006)
69. Weld, D., Anderson, C., Domingos, P., Etzioni, O., Lau, T., Gajos, K., Wolfman, S.: Automatically personalizing user interfaces. In: Proceedings of the 18th International Joint Confer-

ence Artificial Intelligence (IJCAI-03), pp. 1613–1619. Morgan Kaufman (2003). Acapulco, Mexico
70. Williams, M.: What makes rabbit run? International Journal of Man-Machine Studies **21**, 333–352 (1984)
71. Williams, M., Tou, F.: Rabbit: An interface for database access. In: In Proceedings of the ACM'82 Conference, pp. 83–87. ACM, New York, USA (1982)
72. Zhang, J., Jones, N., Pu, P.: A visual interface for critiquing-based recommender systems. In: EC '08: Proceedings of the 9th ACM conference on Electronic commerce, pp. 230–239. ACM, New York, NY, USA (2008)
73. Zhang, J., Pu, P.: A comparative study of compound critique generation in conversational recommender systems. In: V.P. Wade, H. Ashman, B. Smyth (eds.) Adaptive Hypermedia and Adaptive Web-Based Systems, 4th International Conference, AH 2006, Dublin, Ireland, June 21-23, 2006, Proceedings, *Lecture Notes in Computer Science*, vol. 4018, pp. 234–243. Springer (2006)