Chapter 2
Algorithms for Group Recommendation

Alexander Felfernig, Müslüm Atas, Denis Helic,
Thi Ngoc Trang Tran, Martin Stettinger, and Ralph Samer[ab]

[a] Citation: Alexander Felfernig, Müslüm Atas, Denis Helic, Thi Ngoc Trang Tran, Martin Stettinger, and Ralph Samer. Algorithms for Group Recommendation, in: Group Recommender Systems – An Introduction, Alexander Felfernig, Ludovico Boratto, Martin Stettinger, and Marko Tkalčič (eds.), Springer, pp. 27–58, ISBN: 978-3-319-75066-8, 2018.

[b] This is a pre-print version of the chapter published in the book "Group Recommender Systems: An Introduction": http://www.springer.com/us/book/9783319750668.

Abstract In this chapter, our aim is to show how group recommendation can be implemented on the basis of recommendation paradigms for individual users. Specifically, we focus on collaborative filtering, content-based filtering, constraint-based, critiquing-based, and hybrid recommendation. Throughout this chapter, we differentiate between (1) *aggregated predictions* and (2) *aggregated models* as basic strategies for aggregating the preferences of individual group members.

2.1 Introduction

As discussed in Chapter 1, there are many real-world scenarios where recommendations have to be made to groups. The main task in these scenarios is to generate relevant recommendations from the preferences (evaluations) of individual group members. As illustrated in Table 2.1, group recommendation approaches can be differentiated with regard to the following characteristics [45, 46].

Preference Aggregation Strategy. In group recommender systems, there are two basic aggregation strategies [35]. First, recommendations are determined for individual group members and then aggregated into a group recommendation.[1] Second, the preferences of individual users are aggregated into a *group profile* which is then used to determine a group recommendation. In this chapter, we show how both strategies can be applied with different recommendation algorithms.

Recommendation Algorithm. The recommendation logic of group recommenders is in many cases based on single user recommenders (collaborative filtering, content-based filtering, constraint-based, critiquing-based, and hybrid recommendation) [22] combined with selected *aggregation functions* from *social choice theory*

[1] One can also distinguish between the *aggregation of items* and the *aggregation of evaluations* (e.g., *ratings* in collaborative filtering) [6, 35] – in this chapter we will provide examples of both.

characteristic	description
Preference Aggregation Strategy	(1) determination of *items/ratings* for individual group members, thereafter aggregation of these items/ratings to a group recommendation, or (2) aggregation of the preferences of group members into a *group profile*, thereafter determination of a recommendation for the group.
Recommendation Algorithm	One of the recommendation algorithms (i.e., collaborative, content-based, constraint-based, critiquing-based, and hybrid).
Preferences Known Beforehand?	For example, in collaborative filtering, ratings are known beforehand [2]. In conversational approaches, preferences are constructed over time [36].
Immediate Item Consumption?	Group recommenders can recommend (1) items that will be *consumed in the future* (e.g., holiday destinations as a basis for a final decision taken by (a) a responsible person or (b) a group on the basis of a discussion [34]), or (2) items *consumed immediately* (e.g., songs [45]).
Active or Passive Group?	A group is *passive* if it does not actively influence the construction of a group profile [2]. *Active* groups negotiate the group profile [33, 50].
Number of Recommended Items	A group recommender can focus on the recommendation of (1) a single item as is the case with travel destinations [33] or (2) multiple items represented, for example, as a sequence (e.g., television items [45]).
Type of Preference Acquisition	Preferences can be acquired by interpreting, for example, the ratings of items or by engaging users in a preference construction process [34].

Table 2.1: Characteristics to classify group recommenders [45, 46].

[45, 52]. These functions will be discussed on the basis of examples from the travel domain introduced in Chapter 1.

Preferences Known Beforehand. Consider the example of single-shot recommendations determined on the basis of collaborative filtering. Some user preferences are already known from previous recommendation sessions, and so do not need to be determined in an iterative process. In contrast, conversational recommender systems [9, 12, 20, 42, 47, 49] engage users in a dialog to elicit user preferences.

Immediate Item Consumption. On the one hand, a pragmatics of a recommendation can be that a group directly experiences the recommended items. For example, consider songs consumed by members of a fitness studio or commercials shown on public screens. On the other hand, recommendations are often interpreted as proposals without the items being experienced immediately.

Active or Passive Group. On the one hand, group profiles can be generated automatically if the preferences of the group members are known. On the other hand, especially when using constraint-based or critiquing-based recommenders, preferences are constructed (i.e., not known beforehand) and thus are adapted and extended within the scope of negotiation processes. The more intensively group models are discussed and negotiated, the higher the degree of group activity.

Number of Recommended Items. The output of a group recommender can be a single item (e.g., restaurant for a dinner or a movie), but also packages (e.g., travel packages), sequences (e.g., songs or travel plans), and even configurations (e.g., software release plans and cars).[2]

[2] See Chapter 7.

Type of Preference Acquisition. Preferences can be collected *implicitly* (through observation, for example, of user's item consumption patterns) or *explicitly* by engaging users in a preference construction process. The latter is the case especially in conversational recommendation [9, 12, 20, 42, 47].

2.2 Preference Aggregation Strategies

Independent of the way preferences are acquired from individual group members (see Chapter 5), a group recommendation is determined by aggregating these preferences in one way or another [35]. In group recommender systems, the determination of recommendations depends on the chosen *preference aggregation strategy* [2, 6, 27, 35, 37, 43, 62].

There are two aggregation strategies (see Figure 2.1): (1) *aggregating recommended items* (or *evaluations*) that were generated separately for each user profile $up(u_i)$ and (2) *aggregating individual user profiles* $up(u_i)$ into a group profile gp. In the first case, *the recommendation step precedes the aggregation step* – item evaluations or items recommended to individual group members are *aggregated* into a corresponding group recommendation. In the second case, *the aggregation step precedes the recommendation step* – group profiles aggregated from individual user profiles are the basis for determining a group recommendation. Following the discussions in [3, 35], we denote the first aggregation strategy as *aggregated predictions* and the second one as *aggregated models* (see Figure 2.1).

Aggregated Predictions. There are two basic approaches to aggregate predictions. *First*, recommendations (items) determined for individual group members can be merged. This approach can be used if a *a set of candidate solutions* should be presented and the group members are in charge of selecting one out of the candidate items. In this context, specific items which are not very appealing for some group members are not filtered out. Group members play an important role in the decision making process, since *no ranking* of the individual candidate items is provided. *Second*, group-member-specific predictions for candidate items are aggregated. The outcome of this approach is a *ranking of candidate items*.

Aggregated Models. Instead of aggregating recommendations for individual users, this approach constructs a *group preference model* (*group profile*) that is then used for determining recommendations. This is especially useful in scenarios where group members should have the opportunity to *analyze, negotiate, and adapt the preferences of the group* [35]. Another advantage of applying group preference models is that the *privacy concerns* of users can be alleviated, since there is no specific need to record and maintain individual user profiles.

Although studies exist that compare the predictive quality of the two basic aggregation approaches (*aggregated predictions* and *aggregated models*) [2, 3, 5, 15], more in-depth comparisons are needed that also focus on specific group properties such as size, homogeneity (e.g., similarity between group members can have a negative impact on the decision quality), the item domain (e.g., high-involvement vs.

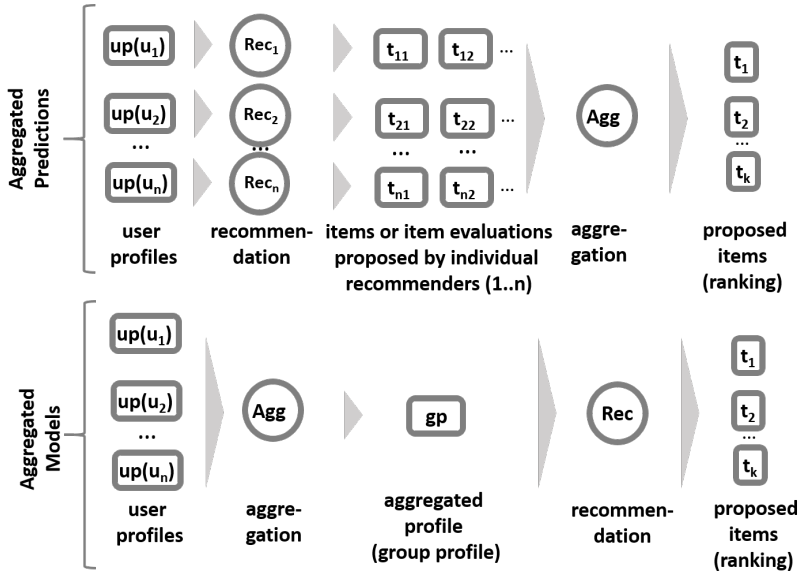

Fig. 2.1: Two basic aggregation strategies in group recommendation: (1) recommendation based on *single user profiles* with a downstream aggregation of items (or evaluations/ratings) recommended to group members/users (*aggregated predictions*) and (2) recommendation based on *aggregated models* (*group profiles*).

low-involvement items [19]), and also the ways in which individual and group rating behavior differs [56]. After introducing a couple of *social choice based preference aggregation functions* that help to implement aggregated predictions and aggregated models, we show how preference aggregation can be implemented in the context of *collaborative-* and *content-based filtering* as well as *constraint-based*, *critiquing-based*, and *hybrid recommendation*.

2.3 Social Choice based Preference Aggregation Functions

A major issue in all of the mentioned group recommendation scenarios is how to adapt to the group as a whole, given information about the individual preferences of group members [1, 45]. As there is *no optimal way* to aggregate recommendation lists [1], corresponding approximations (in the following denoted as *aggregation functions*) have to be used to come up with a recommendation that takes into account 'as far as possible' the individual preferences of group members. As mentioned in [46, 57], the aggregation functions can be categorized into *majority-based* (*M*), *consensus-based* (*C*), and *borderline* (*B*). Table 2.2 provides an overview of different

kinds of *aggregation functions* taken from *social choice theory*[3] [11, 44, 45, 58] and their categorization into one of the three mentioned categories (M, C, and B).

aggregation strategy	description	recommendation
Additive Utilitarian (ADD) [C]	sum of item-specific evaluations	$\underset{(t \in I)}{argmax}(\Sigma_{u \in G} eval(u,t))$
Approval Voting (APP) [M]	number of item-specific evaluations above an approval threshold	$\underset{(t \in I)}{argmax}(\lvert\{u \in G : eval(u,t) \geq threshold\}\rvert)$
Average (AVG) [C]	average of item-specific evaluations	$\underset{(t \in I)}{argmax}(\frac{\Sigma_{u \in G} eval(u,t)}{\lvert G \rvert})$
Average without Misery (AVM) [C]	average of item-specific evaluations (if all evaluations are above a defined threshold)	$\underset{(t \in I : \nexists u \in G \lvert eval(u,t) \leq threshold)}{argmax}(\frac{\Sigma_{u \in G} rating(u,t)}{\lvert G \rvert})$
Borda Count (BRC) [M]	sum of item-specific scores derived from item ranking	$\underset{(t \in I)}{argmax}(\Sigma_{u \in G} score(u,t))$
Copeland Rule (COP) [M]	number wins (w) - number losses (l) in pair-wise evaluation comparison	$\underset{(t \in I)}{argmax}(\lvert w(t, I - \{t\})\rvert - \lvert l(t, I - \{t\})\rvert)$
Fairness (FAI) [C]	item ranking as if individuals ($u \in G$) choose them one after the other	$\underset{(t \in I)}{argmax}(eval(u,t))$ [in each iteration]
Least Misery (LMS) [B]	minimum item-specific evaluation	$\underset{(t \in I)}{argmax}(mineval(t))$
Majority Voting (MAJ) [B]	majority of evaluation values per item	$\underset{(t \in I)}{argmax}(majorityeval(t))$
Most Pleasure (MPL) [B]	maximum item-specific evaluation	$\underset{(t \in I)}{argmax}(maxeval(t))$
Most Respected Person (MRP) [B]	item-evaluations of most respected user	$\underset{(t \in I)}{argmax}(eval(u_{mrp}, t))$
Multiplicative (MUL) [C]	multiplication of item-specific evaluations	$\underset{(t \in I)}{argmax}(\Pi_{u \in G} eval(u,t))$
Plurality Voting (PLU) [M]	item with the highest #votes from $u \in G$	$\underset{(t \in I)}{argmax}(votings(t))$ [in each iteration]

Table 2.2: Basic *aggregation functions* for group recommender systems [11, 40, 45, 46, 57] where *argmax* is assumed to return a recommended item. Tie breaking rules such as *random selection* can be applied. M, C, and B denote the aggregation categories *majority-based*, *consensus-based*, and *borderline*; u represents a *user* (*group member*), G a *group*, t an *item*, and I a set of *items*.

Majority-based aggregation functions (M) represent aggregation mechanisms that focus on those items which are the *most popular* [44, 58]. Examples of majority-based functions are *Plurality Voting* (PLU) (winner is the item with the highest number of *votes*), *Borda Count* (BRC) (winner is the item with the best *total rank-*

[3] Also denoted as *group decision making*.

ing score where each item rank[4] is associated with a score 0 .. $\#items - 1$), and *Copeland Rule* (COP) (winner is the item that outperforms other items in terms of pairwise *evaluation*[5] comparison) (see Table 2.3). Equal evaluations in BRC are handled as follows: in the example of Table 2.3, user u_2 provided the rating 2.5 for t_2 and t_3; both items receive the same *score* which is $\frac{0+1}{2} = 0.5$.

When comparing the items t_1 and t_2 in Table 2.3, t_1 outperforms t_2 two times and looses once in terms of user evaluations ($u_1 : 5.0$ vs. $u_2 : 3.0$, $u_1 : 4.5$ vs. $u_2 : 2.5$, and $u_1 : 3.5$ vs. $u_2 : 4.0$) which results in a win ("+") 2:1. Comparing items t_2 and t_3 results in a tie 1:1 which is indicated by "0" in Table 2.3. Such an evaluation has to be performed for each item in order to determine a winner on the basis of COP (see the *rhs* of Table 2.3). A further majority-based aggregation function is *Approval Voting* (APP) that recommends items with the highest number of supporting users. In this context, support is measured in terms of the number of item evaluations above a defined threshold.

item	votes			*PLU*	evaluations (scores)			*BRC*	evaluations			COP index			*COP*
	u_1	u_2	u_3		u_1	u_2	u_3		u_1	u_2	u_3	t_1	t_2	t_3	
t_1	1	1	0	2 $\surd$	5.0(2)	4.5(2)	3.5(1)	5 $\surd$	5.0	4.5	3.5	0	+	+	2$\surd$
t_2	0	0	1	1	3.0(0)	2.5(0.5)	4.0(2)	2.5	3.0	2.5	4.0	-	0	0	-1
t_3	0	0	0	0	3.5(1)	2.5(0.5)	1.5(0)	1.5	3.5	2.5	1.5	-	0	0	-1

Table 2.3: Examples of *majority-based* aggregation: Plurality Voting (PLU), Borda Count (BRC), and Copeland Rule (COP, "+" indicates a win, "-" a loss, and "0" a tie). $\surd$ denotes the item t_i with the best evaluation, i.e., the *recommendation*.

Consensus-based functions (C)[6] represent aggregation mechanisms that take into account the preferences of *all group members* [58]. Examples are *Additive Utilitarian* (ADD) (winner is the item with the maximum sum of user-individual evaluations), *Average* (AVG) (winner is the item with the maximum average of the user-individual evaluations – in the line of ADD[7], the function causes problems in the context of larger groups since the opinions of individuals count less), and *Multiplicative* (MUL) (winner is the item with the maximum product of the user-individual evaluations) (see Table 2.4). Further majority-based aggregation functions are *Average without Misery* (AVM) that recommends the average evaluation for items that do not have individual ratings below a defined threshold and *Fairness* (FAI) which ranks items as if individuals are choosing them in turn [45].

Borderline functions (B) represent aggregation mechanisms that take into account only *a subset of the user preferences* [58]. Examples of borderline functions are *Least Misery* (LMS) (winner is the item with the highest of all lowest evaluations

[4] The highest rank is assumed to be 1. For example, in collaborative filtering it is associated with the highest rating. The highest rank is associated with the score *#items-1*.

[5] For example, when using collaborative filtering, evaluations are denoted as *ratings*.

[6] Also denoted as *democratic functions*.

[7] ADD and AVG result in the same rankings.

item	evaluations			*ADD*	*AVG*	*MUL*
	u_1	u_2	u_3			
t_1	5	2	2	9	3	20
t_2	3	3	4	10$\surd$	3.3 $\surd$	36 $\surd$
t_3	2	3	2	7	2.3	12

Table 2.4: Examples of *consensus-based* aggregation: Additive Utilitarian (ADD), Average (AVG), and Multiplicative (MUL).

given to items – when using this function, items may be selected that nobody hates but also nobody really likes; furthermore, there is the danger that a minority dictates the group (especially in settings involving larger groups) [44]), *Most Pleasure* (MPL) (winner is the item with the highest of all individual evaluations – items may be selected that only a few persons really like)[8], and *Majority Voting* (MAJ) (item with the highest number of all evaluations representing the majority of item-specific evaluations) (see Table 2.5). A further borderline aggregation function is *Most Respected Person* (*MRP*) that recommends a rating (evaluation) proposed by the most respected individual.

item	evaluations			*LMS*	*MPL*	*MAJ*
	u_1	u_2	u_3			
t_1	5	2	2	2	5 $\surd$	2
t_2	3	3	4	3$\surd$	4	3 $\surd$
t_3	2	3	2	2	3	2

Table 2.5: Examples of *Borderline* aggregation: Least Misery (LMS), Most Pleasure (MPL), and Majority Voting (MAJ).

The following discussions of group recommendation approaches will be based on a set of example items from the travel domain (see Chapter 1). Using these items, we will show how different recommendation approaches can determine group recommendations with *aggregated models* and *aggregated predictions*.

2.4 Collaborative Filtering for Groups

Collaborative filtering (CF) [38, 41] is based on the idea of recommending items that are derived from the preferences of *nearest neighbors*, i.e., users with preferences similar to those of the current user. In the following, we show how *aggregated predictions* and *aggregated models* can be applied to CF for groups.

[8] Variants thereof can be considered [44], for example, *most pleasure without misery* where only items are considered that do not have evaluations below a predefined threshold.

Aggregated Predictions. When applying the *aggregated predictions* strategy in combination with collaborative filtering, ratings are determined for individual users and then aggregated into a recommendation for the group (see Figure 2.2).

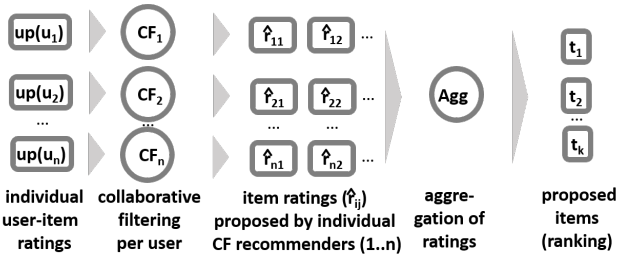

Fig. 2.2: Collaborative filtering for groups based on *aggregated predictions* (*ratings*). $\hat{r}_{ij}$ is the rating prediction for item j proposed by recommender i ($i = 1..n$).

Following this approach, for each group member (and corresponding recommender) i and each item j not rated by this group member, a rating prediction $\hat{r}_{ij}$ is determined [4]. For simplicity, we assume that the items $\{t_1,..,t_{10}\}$ in Table 2.6 have not been previously consumed by the group members, i.e., the rating has been proposed by a collaborative filtering algorithm.[9] Thereafter, these predictions are aggregated on the basis of different aggregation functions (see Table 2.2). In the following example, we assume that some variant of collaborative filtering [17, 28, 53] has already been applied to predict ratings (e.g., a matrix factorization approach [51, 56] can be applied to infer *user* $\times$ *item* rating tables as shown in Table 2.6).

item	name	rating predictions $\hat{r}_{ij}$ (scores)					aggregation		
		u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	Vienna	5.0(9)	3.5(2)	1.0(0)	4.5(7)	5.0(9)	3.8	27	1.0
t_2	Yellowstone	2.5(0)	4.0(4)	3.0(3)	2.0(0)	1.1(0)	2.5	7	1.1
t_3	New York	4.9(8)	3.8(3)	4.0(7)	3.3(4)	4.0(5)	4.0	27	3.3$\surd$
t_4	Blue Mountains	3.1(2)	5.0(9)	4.2(8)	2.4(1)	4.4(8)	3.8	28	2.4
t_5	London	4.0(4)	4.3(7)	3.3(5)	4.1(6)	2.9(3)	3.7	25	2.9
t_6	Beijing	4.5(6)	4.1(5)	5.0(9)	3.2(3)	4.2(6)	4.2$\surd$	29$\surd$	3.2
t_7	Cape Town	4.2(5)	4.2(6)	3.4(6)	3.1(2)	3.8(4)	3.7	23	3.1
t_8	Yosemity	3.4(3)	2.6(0)	1.6(1)	5.0(9)	2.4(2)	3.0	15	1.6
t_9	Paris	4.7(7)	3.1(1)	2.7(2)	3.6(5)	2.2(1)	3.3	16	2.2
t_{10}	Pittsburgh	2.6(1)	4.5(8)	3.1(4)	4.6(8)	4.3(7)	3.8	28	2.6

Table 2.6: Rating predictions and corresponding *scores* (scores are used by *BRC*). Recommendations are derived on the basis of aggregation functions (*AVG*, *BRC*, *LMS*). The $\surd$ symbol indicates the item with the best evaluation.

[9] Item predictions for individual users can be based on collaborative recommendation approaches as introduced in Chapter 1.

The result of the aggregation step is a *ranking of candidate items*. In our example, the majority of aggregation functions recommends the item t_6.

An alternative to the aggregation of ratings is to *aggregate predicted items* where *items* determined by individual recommenders are aggregated into a group recommendation (see Figure 2.3).

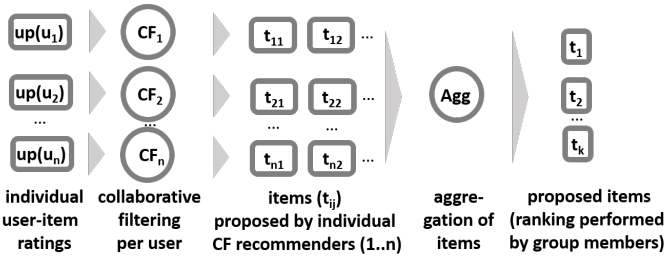

Fig. 2.3: Collaborative filtering for groups based on *aggregated predictions* (*items*).

Following this approach, items with the highest predicted rating for a specific user are considered as part of the recommendation. If we want to generate a recommendation consisting of, for example, at most 10 items, the two top-rated items (upper bound) in each group member specific recommendation can be included in the group recommendation. In the example shown in Table 2.6, $\{t_1, t_3\}$ are the two top-rated items of user u_1, $\{t_4, t_{10}\}$ are chosen for user u_2, $\{t_4, t_6\}$ for user u_3, $\{t_8, t_{10}\}$ for user u_4, and $\{t_1, t_4\}$ for user u_5. The union of these group member individual recommendations is $\{t_1, t_3, t_4, t_6, t_8, t_{10}\}$ which represents the group recommendation – in this context, group members are in charge of item ranking. This way of constructing a group recommendation is similar to the idea of the *Fairness* (FAI) aggregation function (see Table 2.2).

Aggregated Models. When using this aggregation approach, ratings of individual users are aggregated into a group profile gp (see Figure 2.4). Based on the group profile (gp), collaborative filtering determines a ranking for each candidate item.

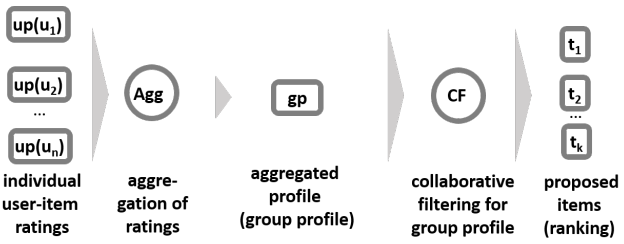

Fig. 2.4: Collaborative filtering for groups based on *aggregated models*.

In the aggregated models approach, the group is represented by a *group profile* (gp) that includes item-specific evaluations (ratings) derived through aggregation functions applied to the item ratings of individual group members. Often, the aggregation is based on a weighted average function (see, e.g., [4]), however, the aggregation functions mentioned in Table 2.2 can be considered alternatives.

Following the aggregated models strategy, collaborative filtering is applied to individual group profiles, i.e., for a given group profile (gp), similar group profiles (k nearest neighbors k-NN)[10] are retrieved and used for determining a recommendation. In our example, the item t_2 (*Yellowstone*) is *not* known to the current group gp but received the highest ratings from the nearest neighbor groups gx and gy (see Table 2.7) which makes it a recommendation candidate for gp.

item	name	gp	$gx \in NN$	$gy \in NN$	recommended ratings
t_1	Vienna	5.0	5.0	4	-
t_2	Yellowstone	-	4.0	4.5	$4.49\surd$
t_3	New York	4.0	3.0	3.5	-
t_4	Blue Mountains	-	4.5	4	4.44
t_5	London	4.0	3.9	3.5	-
t_6	Beijing	-	3.5	3	3.44
t_7	Cape Town	-	4.7	3	3.99
t_8	Yosemity	3.0	3.8	3.2	-
t_9	Paris	4.0	3.9	2.9	-
t_{10}	Pittsburgh	-	5.0	3.3	4.28
average		4.0	4.13	3.5	-

Table 2.7: Applying collaborative filtering (CF) to a group profile gp (gp-ratings have no relationships to earlier examples). The $\surd$ symbol indicates the item with the best CF-based evaluation.

The *similarity* between the group profile gp and another group profile gx (the nearest neighbor) can be determined, for example, using the Pearson correlation coefficient (see Chapter 1). Formula 2.1 is an adapted version that determines the similarity between a group profile and the profiles of other groups. In this context, TD_c represents the set of items that have been rated by both groups (gp and gx), r_{gx,t_i} is the rating of group gx for item t_i, and $\overline{r_{gx}}$ is the average rating of group gx.

$$similarity(gp,gx) = \frac{\sum_{t_i \in TD_c}(r_{gp,t_i} - \overline{r_{gp}}) \times (r_{gx,t_i} - \overline{r_{gx}})}{\sqrt{\sum_{t_i \in TD_c}(r_{gp,t_i} - \overline{r_{gp}})^2} \times \sqrt{\sum_{t_i \in TD_c}(r_{gx,t_i} - \overline{r_{gx}})^2}} \tag{2.1}$$

The information about groups with a similar rating behavior (i.e., nearest neighbors NN) compared to the current group gp is the basis for predicting the rating of gp for an *item* t that has not been rated by members of gp (see Formula 2.2).

[10] In our example, we assume $k = 2$.

$$prediction(gp,t) = \hat{r}(gp,t) = \overline{r_{gp}} + \frac{\sum_{gj \in NN} similarity(gp,gj) \times (r_{gj,t} - \overline{r_{gj}})}{\sum_{gj \in NN} similarity(gp,gj)} \quad (2.2)$$

Recommendations can also be determined on the basis of *ensemble voting* [59]: each aggregation function can represent a vote. The more such votes an item receives, the higher is it's relevance for the group. In our running example, item t_6 is regarded as favorite item since it received the best evaluation by the majority of the used aggregation functions (see the *aggregated predictions* example in Table 2.6).

2.5 Content-based Filtering for Groups

Content-based filtering (CBF) is based on the idea of recommending new items with *categories*[11] similar to those preferred by the current user. Categories preferred by a user (group member) are stored in a user profile; these categories are derived from descriptions of items already consumed by the user.

Aggregated Predictions. When using this aggregation strategy, group member individual content-based recommenders determine the similarity between (a) items not consumed by him/her and (b) his/her user profile.[12] The identified item similarities (or items) are then aggregated and thus form the basis of a group recommendation (see Figure 2.5).

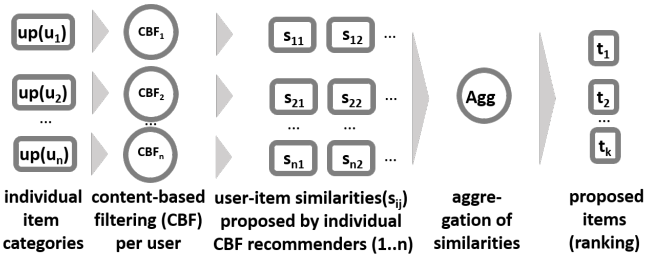

Fig. 2.5: Content-based filtering for groups based on *aggregated predictions*. Similarity s_{ij} denotes the similarity between user i and item j determined by recommender i ($i = 1..n$).

Table 2.8 depicts example profiles of group members $u_1..u_5$. For each of these profiles, the similarity to the items included in Table 1.8 is determined (we assume that these items have not been consumed/evaluated by the group members). These similarity values are the basis for a group recommendation (see Table 2.9).

[11] Alternatively, *keywords* extracted from item descriptions.

[12] The determination of *user* $\times$ *item similarities* can be based on content-based recommendation approaches as discussed in Chapter 1.

category	individual item categories				
	u_1	u_2	u_3	u_4	u_5
beach	x	x	x	x	x
city tours	-	x	-	x	-
nature	x	-	x	-	x
entertainment	-	-	-	-	-

Table 2.8: Example profiles of group members (preferences regarding travel destinations). If a group member u_i likes a category, this is denoted with 'x'.

The user-item similarities of Table 2.9 are calculated by a content-based recommender (*similarity* metric 1.3 in Chapter 1). The calculation is based on the item categories included in Table 2.8, i.e., *beach*, *city tours*, *nature*, and *entertainment*. For example, $similarity(u_1,t_2) = \frac{2*|categories(u_1) \cap categories(t_2)|}{|categories(u_1)|+|categories(t_2)|} = \frac{2}{3} = 0.66$.

item	name	user-item similarities (scores)					aggregation		
		u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	Vienna	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_2	Yellowstone	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_3	New York	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_4	Blue Mountains	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_5	London	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_6	Beijing	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_7	Cape Town	0.66(7.5)	0.66(8.5)	0.66(7.5)	0.66(8.5)	0.66(7.5)	$0.66\surd$	$39.5\surd$	$0.66\surd$
t_8	Yosemity	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_9	Paris	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_{10}	Pittsburgh	0(2.5)	0.66(8.5)	0(2.5)	0.66(8.5)	0(2.5)	0.26	24.5	0

Table 2.9: User $\times$ item similarities (and corresponding *scores* used by *BRC*) as input for *AVG*, *BRC*, *LMS* to derive a group recommendation. The $\surd$ symbol indicates the item with the best evaluation.

On the basis of a user $\times$ item similarity matrix, aggregation functions can determine a group recommendation. An alternative to the aggregation of similarities is to *aggregate items* proposed by individual content-based recommenders. If we want to generate a recommendation consisting of, for example, at most 5 items (upper bound), the highest rated item of each group member can be included in the group recommendation. In our example depicted in Table 2.9, $\{t_2\}$ is among the highest rated items of user u_1 (the other three are excluded due to the user-specific limit of one item), t_7 can be selected for user u_2, t_4 for user u_3, t_{10} for user u_4, and t_2 for user u_5. The group recommendation includes all of these items: $\{t_2,t_4,t_7,t_{10}\}$.

Aggregated Models. When using this strategy, preferred categories of individual users are integrated into a group profile gp. Thereafter, content-based filtering de-

termines recommendations by calculating the similarities between gp and candidate items (items not consumed by the group – see Figure 2.6).

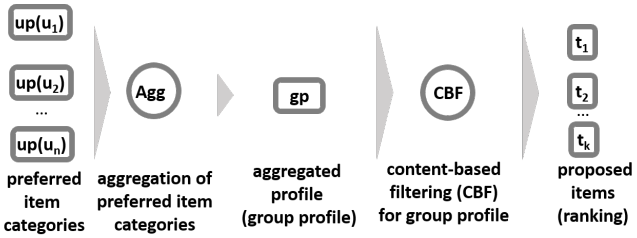

Fig. 2.6: Content-based filtering for groups based on *aggregated models*.

In our example (see Table 2.10), the derived group profile is represented by the union of the categories stored in the individual user profiles. Items are recommended that are similar to the categories in the group profile and have not been consumed by group members. In our example, the derived group profile gp entails the categories *Beach*, *City Tours*, and *Nature*.

category	individual item categories					gp
	u_1	u_2	u_3	u_4	u_5	
beach	x	x	x	x	x	x
city tours	-	x	-	x	-	x
nature	x	-	x	-	x	x
entertainment	-	-	-	-	-	-

Table 2.10: Aggregation of preferences (categories) of group members into a group profile gp.

The similarity between the group profile gp and candidate items can be determined using Formula 2.3 which is an adaption of Formula 1.3 to group settings. The similarities between gp and items t_i (taken from our example itemset shown in Table 1.8) are determined by comparing the categories *beach*, *city-tours*, *nature*, and *entertainment* (see Table 2.11). For example, $similarity(gp,t_1) = \frac{2*|categories(gp) \cap categories(t_1)|}{|categories(gp)|+|categories(t_1)|} = \frac{2}{5} = 0.4$. In this context, we assume that the items of Table 2.11 have not been consumed by the group.

$$similarity(gp,item) = \frac{2*|categories(gp) \cap categories(item)|}{|categories(gp)|+|categories(item)|} \tag{2.3}$$

item	name	similarity(gp, t_i)
t_1	Vienna	$\frac{2}{5} = 0.4$
t_2	Yellowstone	$\frac{2}{4} = 0.5$
t_3	New York	$\frac{2}{5} = 0.4$
t_4	Blue Mountains	$\frac{2}{4} = 0.5$
t_5	London	$\frac{2}{5} = 0.4$
t_6	Beijing	$\frac{2}{5} = 0.4$
t_7	Cape Town	$\frac{6}{7} = 0.86$ $\surd$
t_8	Yosemity	$\frac{2}{4} = 0.5$
t_9	Paris	$\frac{2}{5} = 0.4$
t_{10}	Pittsburgh	$\frac{2}{4} = 0.5$

Table 2.11: Applying content-based filtering (CBF) to a group profile gp (see Table 2.10). The $\surd$ symbol indicates the item with the best evaluation determined by CBF.

2.6 Constraint-based Recommendation for Groups

Taking into account groups in constraint-based recommendation [20] requires the extension of our definition of a recommendation task, as given in Chapter 1.

Definition (Recommendation Task for Groups). A recommendation task for groups can be defined by the tuple (G, $R = R_1 \cup .. \cup R_m$, I) where $G = \{u_1, u_2, .., u_m\}$ represents a group of users, $R_j = \{r_{1j}, r_{2j}, ..., r_{nj}\}$ represents a set of requirements (r_{ij} denotes the requirement i of group member j), and $I = \{t_1, .., t_k\}$ represents a set of items. The goal is to identify items in I which fulfill all requirements in R. A solution for a recommendation task can be defined as follows.

Definition (Recommendation Task for Groups – Solution). A solution for a recommendation task for groups (G,R,I) is a set $S \subseteq I$ such that $\forall t_i \in S : t_i \in \sigma_{[R]} I$ where σ is the selection operator of a conjunctive query, R represents requirements defined by group members, and I represents a collection of items.

In group recommendation settings, each group member should specify his/her *requirements* (in our example, these are hard constraints related to season and topics) and *preferences* (*weights* or *soft constraints*) with regard to a set of *interest dimensions* (in our example, *security*, *attractiveness*, and *crowdedness*) – see Table 2.12. Requirements are constraints that are used to pre-select items, preferences specify weights that are used to rank the pre-selected items.

user	requirements		preferences (weights)		
	season	*topics*	*security*	*attractiveness*	*crowdedness*
u_1	r_{11} :spring	-	0.5	0.4	0.1
u_2	r_{12} :spring	r_{22} :citytours	0.2	0.7	0.1
u_3	-	r_{13} :entertainment	0.3	0.3	0.4
u_4	r_{14} :spring	-	0.6	0.2	0.2
u_5	-	r_{15} :citytours	0.1	0.8	0.1

Table 2.12: User-specific *requirements* and *preferences* (*weights*).

In both scenarios, i.e., *aggregated predictions* and *aggregated models*, group members have to define their requirements and preferences.

Aggregated Predictions. We will first show how to handle aggregated predictions in constraint-based recommendation for groups (see Figure 2.7).

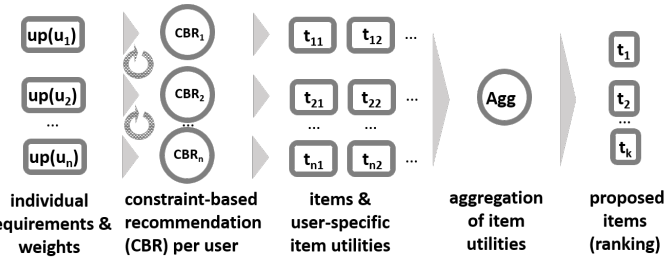

Fig. 2.7: Constraint-based recommendation for groups based on *aggregated predictions*. User preferences are constructed iteratively (conversational recommendation approach). Item t_{ij} represents item j (including corresponding item utilities) determined by recommender i.

A constraint-based recommender derives user-specific recommendations (items and user-specific item utilities) on the basis of a set of requirements and preferences. Item utilities for specific group members can be determined with multi-attribute utility theory (MAUT) [60, 62] (see Formula 1.4). For example, on the basis of the user requirements defined in Table 2.12 and the example itemset of Table 1.8, the utility of item t_1 for user u_1 can be determined as follows: $utility(u_1,t_1) = \sum_{d \in Dimensions} contribution(t_1,d) \times weight(u_1,d) = contribution(t_1,security) \times weight(u_1,security) + contribution(t_1,attractiveness) \times weight(u_1,attractiveness) + contribution(t_1,crowdedness) \times weight(u_1,crowdedness)$ = 5.0 × 0.5 + 5.0 × 0.4 + 2.0 × 0.1 = 2.5 + 2.0 + 0.2 = 4.7. These user-specific item utilities are aggregated into a group recommendation (see Table 2.13).

If an entry of item t_i in *user-specific item utilities* in Table 2.13 > 0, this indicates that the item t_i fulfills all requirements of the corresponding group member. In contrast, table entries $= 0$ are used to indicate that an item does not completely fulfill the requirements of a group member. For example, the requirements of u_2 ($\{r_{12}, r_{22}\}$) are not completely fulfilled by t_2 ($r_{22} : topics = citytours$ is not supported). Even if an item does not completely fulfill the requirements of some users, it could be recommended. The lower the number of users with completely fulfilled requirements with regard to a specific item t_i, the lower the probability that t_i will be recommended. A set of individual user requirements can also be inconsistent with the effect that no fitting item can be identified. In such a case, diagnosis methods can help to guide the user out of the *no solution could be found dilemma* [23].[13]

[13] Issues related to conflict resolution will be discussed at the end of this section.

item	item contribution			user-specific item utilities (scores)					aggregation		
	secur.	*attr.*	*crowd.*	u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	5.0	5.0	2.0	4.7(9)	4.7(9)	3.8(9)	4.4(9)	4.7(9)	4.46$\surd$	45.0$\surd$	3.8$\surd$
t_2	4.0	4.0	4.0	4(7.5)	0.0	0.0	4(7)	0.0	1.6	14.5	0.0
t_3	3.0	5.0	1.0	3.6(4.5)	4.2(7.5)	2.8(7.5)	3(4)	4.4(7.5)	3.6	31.0	2.8
t_4	4.0	3.0	5.0	3.7(6)	0.0	0.0	4(7)	0.0	1.54	13.0	0.0
t_5	3.0	4.0	1.0	3.2(3)	3.5(6)	2.5(6)	2.8(2)	3.6(6)	3.12	23.0	2.5
t_6	3.0	3.0	1.0	2.8(1)	2.8(3.5)	2.2(5)	2.6(1)	2.8(3)	2.64	13.5	2.2
t_7	2.0	3.0	3.0	2.5(0)	2.8(3.5)	0.0	2.4(0)	2.9(4)	2.12	7.5	0.0
t_8	4.0	4.0	4.0	4(7.5)	0.0	0.0	4(7)	0.0	1.6	14.5	0.0
t_9	3.0	5.0	1.0	3.6(4.5)	4.2(7.5)	2.8(7.5)	3(4)	4.4(7.5)	3.6	31.0	2.8
t_{10}	3.0	3.0	3.0	3(2)	3(5)	0.0	3(4)	3(5)	2.4	16.0	3.0

Table 2.13: User-specific item utilities (and corresponding *scores* used by *BRC*) with regard to *security*, *attractiveness*, and *crowdedness* determined by utility analysis (see Chapter 1). The $\surd$ symbol indicates the item with the best evaluation.

Also in constraint-based recommendation, an alternative to the aggregation of user $\times$ item utilities (Table 2.13) is to *aggregate items* proposed by individual recommenders. If we want to generate a recommendation based on the *Fairness* (FAI) aggregation strategy and 5 is the upper bound for the number of proposed items, each group member would choose his/her favorite item (not already selected by another group member). In the example shown in Table 2.13, t_1 has the highest utility for user u_1, it also has the highest utility for user u_2, however, since u_1 already selected t_1, u_2 has to identify a different one, which is now t_3. Furthermore, we assume that u_3 selects t_9, u_4 selects t_8, and user u_5 selects t_5. The group recommendation resulting from this aggregation step is $\{t_1, t_3, t_5, t_8, t_9\}$.

Aggregated Models. Another possibility of determining recommendations for groups in constraint-based recommendation scenarios is to first aggregate individual user preferences [33] (requirements and weights related to interest dimensions) into a *group profile* gp and then to determine recommendations (see Figure 2.8).

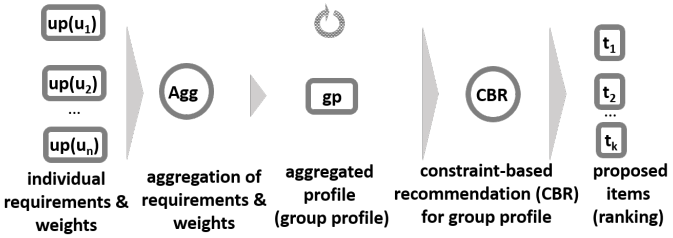

Fig. 2.8: Constraint-based recommendation for groups based on *aggregated models*. Group preferences are constructed iteratively (conversational recommendation).

The construction of a group profile gp is sketched in Table 2.14. Beside aggregating the user requirements $R = \{r_{11}, r_{12}, r_{22}, r_{13}, r_{14}, r_{15}\}$, we also have to aggregate user preferences specified in terms of weights related to the interest dimensions *security*, *attractiveness*, and *crowdedness*.

weights & requirements	u_1	u_2	u_3	u_4	u_5	gp
security	0.5	0.2	0.3	0.6	0.1	0.34 (AVG)
attractiveness	0.4	0.7	0.3	0.2	0.8	0.48 (AVG)
crowdedness	0.1	0.1	0.4	0.2	0.1	0.18 (AVG)
season	r_{11}: spring	r_{12}: spring	-	r_{14}: spring	-	r_{11}, r_{12}, r_{14}
topics	-	r_{22}: citytours	r_{13}: entertainment	-	r_{15}: citytours	r_{22}, r_{13}, r_{15}

Table 2.14: Construction of a group profile (gp). User-specific weights regarding the interest dimensions *security*, *attractiveness*, and *crowdedness* are aggregated into gp using AVG. Furthermore, user requirements r_{ij} are combined into $R = \{r_{11}, r_{12}, r_{22}, r_{13}, r_{14}, r_{15}\}$.

On the basis of the requirements defined in gp and the item definitions in Table 1.8, a conjunctive query $\sigma_{[r_{11}, r_{12}, r_{22}, r_{13}, r_{14}, r_{15}]} I$ results in: $\{t_1, t_3, t_5, t_6, t_7, t_9\}$, i.e., these items are consistent with the requirements defined in gp. Formula 2.4 can be used then to determine item-specific utilities on the basis of the group profile gp. For example, $utility(gp, t_1) = \sum_{d \in Dimensions} contribution(t_1, d) \times weight(gp, d) = contribution(t_1, security) \times weight(gp, security) + contribution(t_1, attractiveness) \times weight(gp, attractiveness) + contribution(t_1, crowdedness) \times weight(gp, crowdedness) = 5 \times 0.34 + 5 \times 0.48 + 2 \times 0.18 = 1.7 + 2.4 + .36 = 4.46$. The resulting utilities are shown in Table 2.15.

$$utility(gp, item) = \sum_{d \in Dimensions} contribution(item, d) \times weight(gp, d) \qquad (2.4)$$

It can be the case that a set of user requirements is *inconsistent with all items* of an itemset. In such a situation, users of a constraint-based recommender have to adapt their requirements such that at least one solution can be identified. Related techniques will be discussed in the following section.

2.7 Handling Inconsistencies

Since item retrieval in constraint-based recommendation is based on semantic queries (e.g., conjunctive queries), situations can occur where no solution can be identified for the given set of requirements [25], i.e., $\sigma_{[R]} I = \emptyset$ (R represents the union of requirements specified by individual group members and I represents the

item	item contribution			utility(gp, t_i)
	secur.	*attr.*	*crowd.*	
t_1	5	5	2	4.46 $\surd$
t_2	4	4	4	4.0
t_3	3	5	1	3.6
t_4	4	3	5	3.7
t_5	3	4	1	3.12
t_6	3	3	1	2.64
t_7	2	3	3	2.66
t_8	4	4	4	4.0
t_9	3	5	1	3.6
t_{10}	3	3	3	3

Table 2.15: Item utilities determined on the basis of the weights defined in gp (see Table 2.14). Only items t_i are taken into account that are consistent with the requirements in gp (others are shown greyed out). The $\surd$ symbol indicates the item with the highest utility.

example itemset shown in Table 1.8). An example of such a situation is the following (adapted version of the examples introduced in the previous sections): R = $\{r_{11} : season = summer, r_{21} : eval = 5.0, r_{12} : season = summer, r_{13} : topics = entertainment, r_{14} : topics = entertainment, r_{15} : eval = 5.0\}$ where $\sigma_{[R]}I = \emptyset$. Also in the context of group recommendation scenarios, we are interested in how to change the requirements defined by group members in order to be able to come up with a recommendation consistent with the requirements of all group members.

In the *aggregated predictions* scenario, inconsistencies induced by requirements occur on the 'single user' level: a user specifies his/her requirements but no recommendation can be identified (see Chapter 1). In this context, diagnosis algorithms help to identify possible changes to the user requirements such that a recommendation can be identified. This way, it can be guaranteed that no user-specific inconsistent requirements are passed to the group level.

In the *aggregated models* scenario, the task of resolving inconsistent situations is a similar one: in the case of inconsistencies between requirements defined by a specific group member, diagnosis (see Chapter 1) can actively support him/her in restoring consistency.[14] However, even if the requirements of a user profile are consistent, integrating the requirements of individual users into a group profile gp can induce inconsistencies on the group level [18]. In the aggregated models scenario, diagnosis also supports the achievement of *global consistency*, i.e., all joint preferences defined by individual group members allow the derivation of at least one solution. Table 2.16 shows the user requirements specified in our example.

The conflict sets induced by our example requirements (R) are: $CS_1 : \{r_{11}, r_{21}\}$, $CS_2 : \{r_{11}, r_{15}\}$, $CS_3 : \{r_{12}, r_{21}\}$, $CS_4 : \{r_{12}, r_{15}\}$, $CS_5 : \{r_{13}, r_{21}\}$, $CS_6 : \{r_{13}, r_{15}\}$, $CS_7 : \{r_{14}, r_{21}\}$, and $CS_8 : \{r_{14}, r_{15}\}$. If we resolve the conflicts by deleting the re-

[14] A discussion of algorithms for diagnosis determination can be found in [21, 23, 26, 55].

requirement	Δ_i		
	Δ_1	Δ_2	Δ_3
$r_{11(season=0100)}$		•	•
$r_{21(eval=5.0)}$	•		•
$r_{12(season=0100)}$		•	
$r_{13(topic=entertainment)}$		•	
$r_{14(topic=entertainment)}$		•	
$r_{15(eval=5.0)}$	•		•

Table 2.16: Example user requirements and related diagnoses in the *aggregated models* scenario (r_{ij} = requirement i of user j): $\Delta_1 = \{r_{21}, r_{15}\}$, $\Delta_2 = \{r_{11}, r_{12}, r_{13}, r_{14}\}$, and $\Delta_3 = \{r_{11}, r_{21}, r_{15}\}$. Δ_3 is a non-minimal diagnosis included to show that aggregation functions prefer minimal diagnoses.

quirements r_{21} and r_{15}, a corresponding diagnosis (hitting set) $\Delta_1 = \{r_{21}, r_{15}\}$ can be identified. The second diagnosis is $\Delta_2 = \{r_{11}, r_{12}, r_{13}, r_{14}\}$. The determination of the diagnoses Δ_1 and Δ_2 is shown on the basis of the HSDAG approach (Hitting Set Directed Acyclic Graph) [55] (see Figure 2.9). Table 2.16 includes a third diagnosis ($\Delta_3 = \{r_{11}, r_{21}, r_{15}\}$) which has been included to show that non-minimal diagnoses $\Delta_{\neg min}$ are not preferred by aggregation functions (see Tables 2.17 – 2.18). A corresponding subset ($\Delta \subset \Delta_{\neg min}$) exists that already fulfills the diagnosis properties. In our example, $\Delta_1 \subset \Delta_3$ holds, i.e., Δ_3 is a non-minimal diagnosis.

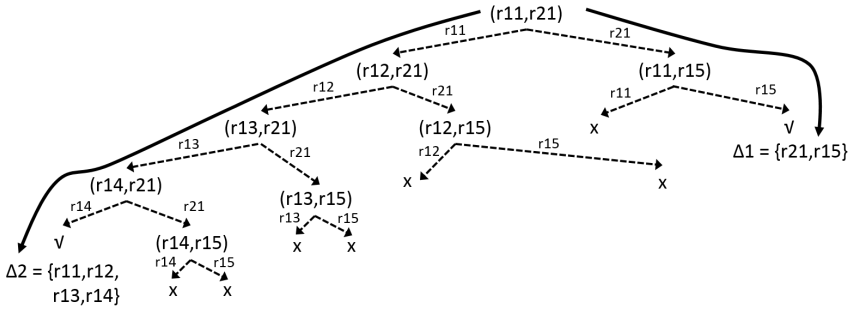

Fig. 2.9: Determination of the minimal diagnoses Δ_1 and Δ_2 using the HSDAG approach [55] (paths to minimal diagnoses are denoted with $\surd$).

As different diagnosis candidates exist ($\Delta_1, \Delta_2, \Delta_3$), we have to figure out which one should be recommended to the group. Similar to the determination of recommendations, diagnosis candidates can be ranked on the basis of different aggregation functions. In Table 2.17 we sketch an approach to rank diagnoses depending on the number of requirements that have to be deleted/adapted by individual group members. Diagnosis Δ_1 has the *lowest number of needed changes* (*ADD*); consequently it can be recommended. *Least Misery* (LMS) recommends one out of $\{\Delta_1, \Delta_2\}$. As

mentioned, we will not discuss diagnosis algorithms in this chapter; for a detailed discussion of diagnosis search and selection in group contexts we refer to [18].

diagnosis	changes per user					aggregation	
	u_1	u_2	u_3	u_4	u_5	*ADD*	*LMS*
Δ_1	1	0	0	0	1	2 $\surd$	1$\surd$
Δ_2	1	1	1	1	0	4	1$\surd$
Δ_3	2	0	0	0	1	3	2

Table 2.17: Diagnosis recommendation in the *aggregated models* scenario based on (1) counting the needed changes per user and (2) *LMS*. The $\surd$ symbol indicates recommended diagnosis candidates.

Diagnosis ranking can be better personalized, if we assume that requirements have importance weights learned, for example, on the basis of previous group decisions [23, 29]. Table 2.18 depicts an example of the determination of diagnosis utilities on the basis of weighted requirements – the utility of a diagnosis can be determined on the basis of Formula 2.5. That implements an additive aggregation strategy: the higher the sum of the individual weights $w(r_{ij})$, the higher the importance of the related requirements for the group members. Consequently, the lower the total importance of the included requirements, the higher the utility of the corresponding diagnosis (see Formula 2.5). In this setting, diagnosis Δ_2 outperforms Δ_1 (also Δ_3) since Δ_2 includes requirements less relevant for the individual group members. *Least Misery* (LMS) in this context analyzes (user-wise) attribute-specific estimated negative impacts of requirement deletions.

Δ_i	weighted requirements						aggregation	
	$w(r_{11}) = 0.1$	$w(r_{21}) = 0.3$	$w(r_{12}) = 0.1$	$w(r_{13}) = 0.1$	q(r_{14})0.1	w(r_{15})=0.3	*utility*	*LMS*
Δ_1	0	0.3	0	0	0	0.3	1.67	0.3
Δ_2	0.1	0.0	0.1	0.1	0.1	0	2.5$\surd$	0.1$\surd$
Δ_3	0.1	0.3	0	0	0	0.3	1.42	0.3

Table 2.18: Utility-based diagnosis recommendation in the *aggregated models* scenario. The $\surd$ symbol indicates the highest rated diagnosis.

$$utility(\Delta) = \frac{1}{\sum_{r_{ij} \in \Delta} w(r_{ij})} \tag{2.5}$$

Remark. An issue for future work in this context is to analyze the possibility of combining the group profile (gp) with local user profiles. This could serve to assure consensus in the group earlier, and avoid efforts related to conflict resolution on the group level. If parts of the group profile are integrated into individual user profiles, this could also help to take into account the requirements of other group members at the very beginning of the decision making process. Further details on

how to determine personalized diagnoses on the basis of search heuristics can also be found in [23, 24].

2.8 Critiquing-based Recommendation for Groups

Critiquing-based recommendation [10, 30] is based on the idea of showing *reference items* to users and allowing users to give feedback in terms of *critiques*. Critiques trigger a new critiquing cycle where *candidate items* (items that fulfill the critiques defined by the user[15]) are compared with regard to their utility as a new *reference item*. This utility is evaluated on the basis of (a) *similarity metrics* (see Chapter 1) that estimate the *similarity* between a reference item and a candidate item and (b) the degree of *support* of the critiques already defined by a user.[16] Intuitively, the more similar a candidate item is with regard to the reference item and the more critiques it supports, the higher its utility. In the following, we assume that the determination of candidate items for a specific group member takes into account his/her previous critiques and the similarity between reference and candidate item. The utility of a candidate item as the next reference item can be determined on the basis of Formulae 2.6 – 2.8. In this context, $utility(c,r,u)$ denotes the utility of a candidate item c to act as a reference item for user u taking into account the current reference item r. Furthermore, $sim(c,r)$ determines the similarity between r and c. Finally, $support(c, critiques(u))$ evaluates the support candidate item c provides for the critiques defined by user u. In this context, *support* is measured in terms of (a) *consistency* between candidate item and critiques and (b) the *weight* of individual critiques (for example, older critiques could have a lower weight).

$$utility(c,r,u) = sim(c,r) \times support(c, critiques(u)) \tag{2.6}$$

$$support(c, critiques) = \Sigma_{crit \in critiques} consistent(c, crit) \times weight(crit) \tag{2.7}$$

$$consistent(c, crit) = \begin{cases} 1 & \text{if } \sigma_{[crit]}\{c\} \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

Let us assume that the first reference item (item r that is the first one shown to start a critiquing session) shown to each group member is t_1. Table 2.19 depicts example critiques defined thereafter on t_1 by the group members u_1, u_2, and u_3. We also assume that items used in the example correspond to the travel destinations itemset shown in Table 1.8. Finally, we assume *equal weights* for critiques.

---

[15] Different variants thereof exist in critiquing-based systems ranging from *taking into account only the most recent critique* to *all critiques in the critiquing history* (see Chapter 1).

[16] Also denoted as *compatibility score* [47].

user	1^{st} *critique*	2^{nd} *critique*
u_1	t_1:winter $\in$ season (cr_{11})	t_3:eval $>$ 3.3 (cr_{12})
u_2	t_1:nature $\in$ topics (cr_{21})	t_2:winter $\in$ season (cr_{22})
u_3	t_1:eval $>$ 4.5 (cr_{31})	t_4:citytours $\in$ topics (cr_{32})

Table 2.19: A group-based critiquing scenario: each group member already specified two critiques (denoted as *critiquing history*). The reference item for the 1^{st} critiquing cycle is assumed to be $t_1(u_1, u_2, u_3)$, the reference items for the 2^{nd} critiquing cycle are $t_3(u_1)$, $t_2(u_2)$, and $t_4(u_3)$.

The similarities between potential combinations of reference items (r) and candidate items (c) are depicted in Table 2.20. The attributes *season* (EIB), *topics* (EIB), and *eval* (NIB) are taken into account.[17] For example, sim(t_1,t_2) = s($t_1.season.spring$, $t_2.season.spring$)$\times\frac{1}{9}$ + s($t_1.season.summer$, $t_2.season.summer$)$\times\frac{1}{9}$ + s($t_1.season.autumn$, $t_2.season.autumn$)$\times\frac{1}{9}$ + s($t_1.season.winter$, $t_2.season.winter$)$\times\frac{1}{9}$ + s($t_1.topics.citytours$, $t_2.topics.citytours$)$\times\frac{1}{9}$ + s($t_1.topics.entertainment$, $t_2.topics.entertainment$)$\times\frac{1}{9}$ + s($t_1.topics.nature$, $t_2.topics.nature$)$\times\frac{1}{9}$ + s($t_1.topics.beach$, $t_2.topics.beach$)$\times\frac{1}{9}$ + s($t_1.eval$, $t_2.eval$)$\times\frac{1}{9}$ = 0.66.

item	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
t_1	1.0	.66	.75	.32	.86	.88	.54	.61	.74	.77
t_2	-	1.0	.43	.64	.53	.54	.67	.96	.42	.64
t_3	-	-	1.0	.52	.88	.86	.54	.42	.99	.74
t_4	-	-	-	1.0	.4	.44	.53	.6	0.51	.56
t_5	-	-	-	-	1.0	.96	.42	.53	.89	.84
t_6	-	-	-	-	-	1.0	.43	.5	0.85	.88
t_7	-	-	-	-	-	-	1.0	.62	.53	.53
t_8	-	-	-	-	-	-	-	1.0	0.42	.6
t_9	-	-	-	-	-	-	-	-	1.0	.73
t_{10}	-	-	-	-	-	-	-	-	-	1.0

Table 2.20: Items of Table 1.8 (similarity with regard to *season*, *topics*, and *eval*).

The selection of a new reference item in the critiquing scenario shown in Table 2.19 is depicted in Table 2.21. In this context, reference items are not considered potential candidate items, since the same item should not be presented in follow-up critiquing cycles. Each table entry represents the utility of a specific candidate item (from Table 1.8) with regard to a reference item. For example, $utility(c : t_2, r : t_3, u : u_1) = sim(t_2, t_3) \times support(t_2, critiques(u_1)) = 0.43 \times (0 \times 0.5 + 1 \times 0.5) = 0.21$ (two critiques, i.e., equal weights = 0.5).

If a user interacts with a critiquing-based recommender in *standalone* mode (critiques of other users are not taken into account), he/she receives recommendations

[17] Similarity metrics introduced in Chapter 1 – we assume, *minval*=0 and *maxval*=5.

u	r	$utility(t_i, r, u)$									
		t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
u_1	t_3	-	.21	-	.52	0	.43	.54$\surd$	0	.49	.37
u_2	t_2	-	-	.21	.64	0	0	.67$\surd$	.48	.21	0
u_3	t_4	-	0	.26	-	.2	.44	.26	0	.25	.56$\surd$

Table 2.21: Selection of new reference items based on the *utility* of candidate items t_i (calculation is based on Formula 2.6). We assume that previous reference items are not reference item candidates anymore (represented by '-' entries). The $\surd$ symbol denotes the selected new reference items.

related to his/her preferences [47]. In parallel, critiques from individual group members can be forwarded to a group recommender. Different variants thereof are possible. For example, recommendations determined for a single user can also take into account the preferences of the whole group by simply taking into account some or all of the critiques stored in the group profile [47]. In this context, weights regarding the trade-offs between the importance of user-individual critiques and critiques on the group level have to be specified.

Aggregated Predictions. The process of critiquing-based group recommendation using aggregated predictions is sketched in Figure 2.10.

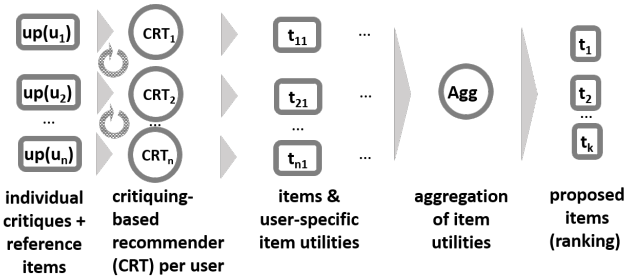

Fig. 2.10: Critiquing-based recommendation for groups with *aggregated predictions*. User preferences are constructed iteratively (conversational recommendation).

On the basis of an initial reference item, individual critiquing-based recommenders start the first critiquing cycle and – depending on user feedback – determine follow-up reference items. In other words, several interaction cycles preceed a decision. After individual group members have completed their selection process, the corresponding results (see, e.g., Table 2.19) can be used to determine a group recommendation. Table 2.22 depicts user-specific utilities of new items (the similarity values are taken from Table 2.21).

An alternative to the aggregation of item utilities (Table 2.22) is to *aggregate items* proposed by individual critiquing-based recommenders. A group recommendation can be determined, for example, by taking the item with the highest utility

c	utility(c,r,u) (score)			aggregation		
	$u_1(r:t_3)$	$u_2(r:t_2)$	$u_3(r:t_4)$	AVG	BRC	LMS
t_1	0 (1.5)	0 (2)	0 (1.5)	0	5	0
t_2	0.21 (4)	0 (2)	0 (1.5)	0.07	7.5	0
t_3	0 (1.5)	0.21 (5.5)	0.26 (6.5)	0.16	13.5	0
t_4	0.52 (8)	0.64 (8)	0 (1.5)	0.39	17.5	0
t_5	0 (1.5)	0 (2)	0.2 (4)	0.07	7.5	0
t_6	0.43 (6)	0 (2)	0.44 (8)	0.29	16	0
t_7	0.54 (9)	0.67 (9)	0.26 (6.5)	0.49 $\surd$	24.5$\surd$	0.26 $\surd$
t_8	0 (1.5)	0.48 (7)	0 (1.5)	0.16	10	0
t_9	0.49 (7)	0.21 (5.5)	0.25 (5)	0.32	17.5	0.21
t_{10}	0.37 (5)	0 (2)	0.56 (9)	0.31	16	0

Table 2.22: User-specific utilities of new items (see Formula 2.6). $\surd$ indicates the item with the best evaluation determined by the corresponding aggregation function.

value per group member (Formula 2.6). The group recommendation is $\{t_7, t_{10}\}$. As discussed in [30], items can be proposed by group members and group members can provide counter-proposals that – with some likelihood – are acceptable to other group members.

Aggregated Models. Following this strategy, a group model (critiquing history on the group level) has to be generated (see Figure 2.11).

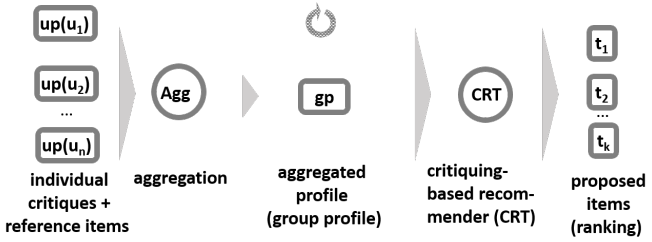

Fig. 2.11: Critiquing-based recommendation for groups with *aggregated models*. Group preferences are constructed iteratively (conversational recommendation).

On the basis of a group model (group profile - gp), a corresponding group recommendation can be determined. In order to build a group profile (gp), critiques defined by group members have to be aggregated. Table 2.23 depicts an example of the aggregation of group member specific critiquing histories into a group profile (gp). In this scenario, the aggregation of individual critiques can lead to a situation where none of the items completely fulfills the defined critiques (see the example group profile in Table 2.23). As a consequence, we have to identify recommendations which support as many critiques as possible. In order to determine a ranking for the different items, Formula 2.9 can be applied where $utility(t, gp)$ denotes the utility of item t with regard to the critiques part of the group profile gp, and *weight*

represents the weight of a critique. In our example, we assume equal weights, however, weights can also be used to reduce the impact of less up-to-date critiques.

group G	group profile (defined by critiques) of G
$\{u_1, u_2, u_3\}$	winter $\in$ season, nature $\in$ topics, eval $>$ 4.5, citytours $\in$ topics

Table 2.23: Set of critiques (=group profile gp) defined by the group $G = \{u_1, u_2, u_3\}$.

$$utility(t, gp) = \Sigma_{crit \in critiques(gp)} consistent(t, crit) \times weight(crit) \qquad (2.9)$$

Table 2.24 represents a list of items (determined on the basis of Formula 2.9) and corresponding utilities with regard to the critiques contained in the group profile gp. For example, $utility(t_1, gp) = 0 \times 0.25 + 0 \times 0.25 + 0 \times 0.25 + 1 \times 0.25 = 0.25$.

item	utility
t_1	0.25
t_2	0.25
t_3	0.5
t_4	0.75 $\surd$
t_5	0.25
t_6	0.5
t_7	0.75 $\surd$
t_8	0.25
t_9	0.5
t_{10}	0.5

Table 2.24: Group-specific utilities of new items determined on the basis of Formula 2.9. The $\surd$ symbol indicates items with the highest utility values.

2.9 Hybrid Recommendation for Groups

As already mentioned, hybrid recommendation helps to compensate specific limitations of one recommendation approach with the strengths of another one [8, 13]. In Chapter 1, we already took a look at different basic hybrid recommendation approaches. We will now sketch hybridization in the context of group recommender systems [3, 14, 16].

Weighted. The idea of weighted hybrid recommendation is to combine the results received from individual recommenders into a corresponding group recommendation. Table 2.25 shows a simple example of applying weighted hybridization in the context of group recommendation. A collaborative recommender for groups (CF) based on the *aggregated models* (AM) strategy and a content-based filtering recommender (CBF) for groups based on the *aggregated predictions* (AP) strategy return

the item rankings shown in Table 2.25. The *Borda Count* (BRC) strategy (see Table 2.2) can now be applied to aggregate the corresponding scores.

item	recommender-specific evaluations (scores)		aggregation
	CF ratings (AM,AVG)	CBF similarities (AP,LMS)	BRC
t_1	4.9 (8)	0.81 (9)	17 $\surd$
t_2	2.2 (1)	0.32 (1)	2
t_3	5.0 (9)	0.66 (7)	16
t_4	4.3 (7)	0.61 (6)	13
t_5	1.5 (0)	0.2 (0)	0
t_6	3.8 (3)	0.55 (5)	8
t_7	3.4 (2)	0.49 (4)	6
t_8	4.1 (4)	0.45 (3)	7
t_9	4.2 (5.5)	0.33 (2)	7.5
t_{10}	4.2 (5.5)	0.79 (8)	13.5

Table 2.25: Recommendation results of two group recommenders (CF based on *aggregated models* (AM) and CBF based on *aggregated predictions* (AP)) as list of ranked items are aggregated on the basis of *Borda Count* (BRC). The $\surd$ symbol indicates the item with the best evaluation.

Mixed. Hybrid recommendation based on the mixed strategy combines the recommended items returned by the individual recommenders (see Table 2.26).

item	recommender-specific rankings		aggregation
	CF (AM,AVG)	CBF (AP,LMS)	FAI (ranking)
t_1	10	9	10
t_2	2	1	1 $\surd$
t_3	7	6	7
t_4	6	5	6
t_5	1	4	2
t_6	3	7	4
t_7	5	3	5
t_8	9	8	9
t_9	4	2	3
t_{10}	8	10	8

Table 2.26: Recommendation results of two group recommenders (CF and CBF) as a list of ranked items aggregated on the basis of *Fairness* (FAI) that implements the zipper principle (alternate inclusion of best ranked items – the item ranked highest by CBF is integrated first). $\surd$ indicates the item with the highest ranking.

In our example, the rankings returned by two group recommenders are aggregated using the fairness (FAI) function where items are included in the final recommendation following the zipper principle, i.e., the item ranked highest by the

CBF recommender is integrated first, then the item ranked highest by the CF recommender is integrated into the recommendation result, and so on.

2.10 Matrix Factorization for Groups

Up to now, we have discussed ways to apply the recommendation approaches of collaborative filtering, content-based filtering, constraint-based, critiquing-based, and hybrid recommendation in group contexts. *Matrix factorization* is a popular approach to collaborative filtering based recommendations [39] . The underlying idea is to explain ratings by characterizing items and users on the basis of a set of factors. The original user $\times$ item matrix is separated into two lower-dimensional ones that explain user item interactions on the basis of the mentioned factors (see Table 2.27).

In this context, each item t is associated with a vector q_t that describes to which extent t represents the factors. Furthermore, each user u is associated with a vector p_u that describes to which extent the factors are important for the user. Finally $\hat{r}_{ui} = q_i^T p_u$ represents an approximation of a user's u rating of t (r_{ui} denotes a user's real rating). More formally, we factorize the rating matrix $\boldsymbol{R} \in \mathbb{R}^{n \times m}$ containing known ratings for n users and m items into matrices $\boldsymbol{P} \in \mathbb{R}^{n \times k}$ and $\boldsymbol{Q} \in \mathbb{R}^{m \times k}$ such that $\boldsymbol{P}\boldsymbol{Q}^T$ closely approximates $\boldsymbol{R}$. In literature and practice, there are several possibilities to measure and minimize the approximation error of the factorization. A popular choice for the approximation error is the sum of the squared errors combined with a simple regularization term, e.g. $\sum_{r_{u,i} \neq \bullet} (r_{u,i} - \mu - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda (||\mathbf{p}_u||^2 + ||\mathbf{q}_i||^2)$, where μ is the global rating average and $\bullet$ represents an unknown rating. Minimization of the error is typically computed with a variant of the gradient descent method.

	i1	i2	i3	i4	i5	i6	i7	i8
u1	5	1		2	2			2
u2	1	4	2		5	1		4
u3			3	5		1	2	4
u4	4		5	3		5	3	
u5	4	1		1		4	3	
u6	4		1	1		5		1
u7		2	2		2		4	1
u8	4		3			4	3	

(a) Rating matrix $\boldsymbol{R}$

	$p_{u,1}$	$p_{u,2}$	$p_{u,3}$
u1	0.27984223	1.33194126	-0.25748666
u2	-0.13639896	-1.00299326	1.09421098
u3	0.29145967	-1.08249042	0.85824434
u4	1.29398513	0.94631031	0.77574863
u5	0.25258519	0.72222304	-1.20079938
u6	-1.26795635	1.16829146	-0.11806872
u7	-0.82074846	0.92881098	-0.20635514
u8	0.15691092	0.64969789	0.53725284

(b) User factors $\boldsymbol{P}$

	$q_{i,1}$	$q_{i,2}$	$q_{i,3}$
i1	0.23698436	1.38151089	-0.23953783
i2	-0.19159424	-1.01718827	0.59249957
i3	1.60559024	0.21567611	-0.26351522
i4	0.3814163	-0.94038814	0.9485212
i5	0.43533279	-0.3900103	1.52692825
i6	0.1258419	1.88675619	0.13922563
i7	-0.47012841	0.65365324	0.03900384
i8	0.98047664	-0.63261944	0.51537004

(c) Item factors $\boldsymbol{Q}$

Table 2.27: We factorize the rating matrix $\boldsymbol{R}$ containing known ratings for $n = 8$ users and $m = 8$ items into matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ such that $\boldsymbol{P}\boldsymbol{Q}^T$ closely approximates $\boldsymbol{R}$. For illustration purposes, we minimize the sum of the squared errors of the approximation together with a simple squared L2-norm regularization term. We set $k = 3$ factors and regularization parameter to $\lambda = 0.02$. We initialize $\boldsymbol{P}$ and $\boldsymbol{Q}$ randomly and optimize with the gradient descent algorithm. Note that factorization brings similar users close to each other in the factor space (c.f. factors of users $u2$ and $u3$ in $\boldsymbol{P}$), whereas dissimilar users are projected further apart (c.f. factors of users $u1$ and $u2$ in $\boldsymbol{P}$).

An approach to the application of matrix factorization in the context of group recommendation scenarios is presented in [51]. The authors introduce two basic strategies denoted as *After Factorization* (AF) and *Before Factorization* (BF). When using AF (see Table 2.28), user-individual matrix factorization is performed in order to identify user-specific factors which are thereafter aggregated (e.g., by determining the average (AVG) of the user-individual factor values). When using BF (see Table 2.29), first user-individual item ratings are aggregated into a group profile, followed by a matrix factorization approach. These two basic variants follow the idea of *aggregated predictions* (AF) and *aggregated models* (BF).

	$p_{G,1}$	$p_{G,2}$	$p_{G,3}$
G	0.144968	-0.25118	0.56499

(a) AF: Group factors

	i1	i2	i3	i4	i5	i6	i7	i8
G	2.40	3.41	2.88	3.68	3.87	2.47	2.64	3.44

(b) AF: Predicted ratings

Table 2.28: In the *After Factorization* (AF) approach the group of users is factorized by merging factors of users (e.g., by calculating averages) in a given group. In our example, we group three users from $G = \{u1, u2, u3\}$. Note that users $u2$ and $u3$ are highly similar to each other but are highly dissimilar to user $u1$. Thus, we expect the group ratings to be biased towards the ratings of users $u2$ and $u3$ as group ratings for items $i1$ (lower because of a low rating from user $u2$) and $i2$ (higher because of a high rating of user $u2$) show.

	$p_{G,1}$	$p_{G,2}$	$p_{G,3}$
G	0.12873	-0.56466	-0.03111

(a) BF: Group factors

	i1	i2	i3	i4	i5	i6	i7	i8
G	2.11	3.38	2.94	3.40	3.08	1.80	2.42	3.32

(b) BF: Predicted ratings

Table 2.29: In the *Before Factorization* (BF) approach a virtual group user is created from the rating matrix by e.g. calculating the average ratings (AVG) for the users from a given group. In the next step, the group factors are calculated from the given factorization by calculating the (Ridge) regression coefficients on the ratings of the virtual user. Finally, the group factors allow us to predict group ratings. The intuition behind BF approach is that the virtual user is a better representation of the users group than a simple aggregation of users factors. In our example, BF predicts a significantly lower rating than AF for item $i6$ because there is much stronger evidence in the data for a low rating (two 1-star ratings).

Due to its simplicity, AF is efficiently calculated and provides a solid baseline for group recommendation approaches based on matrix factorization. However, in practice BF gives significantly better prediction results on larger datasets and for larger groups. For more details on matrix factorization based recommendation approaches

we refer to [39]. Approaches to apply matrix factorization in the context of group recommendation scenarios are discussed in [32, 51].

2.11 Conclusions and Research Issues

In this chapter, we have introduced different group recommendation techniques which are based on the recommendation approaches for individual users introduced in Chapter 1. We showed how related group recommendation scenarios can be designed for collaborative filtering, content-based filtering, constraint-based including utility-based recommendation, critiquing-based, and hybrid recommendation. In this context, we focused on a discussion of the two aggregation strategies: (1) *aggregated predictions* (*items*) and (2) *aggregated models*. In (1), recommendations are determined for individual group members and then aggregated. In (2), the preferences of group members are aggregated, and recommendations are then determined on the basis of information contained in the integrated group profile. An issue already solved in a couple of *person-2-person* recommendation environments is which algorithms can be used to find a person that fits another person with regard to a set of predefined criteria. An online dating application is reported, for example, in [61]. Another application is the identification of experts to support the answering of specific questions [48]. A related issue, especially relevant in the context of group decision making, is *group synthesis*, i.e., the identification of a group that is able to solve a specific problem or to make a decision. Initial work on group synthesis in the context of open innovation scenarios can be found in [7, 31]. A major criteria is to identify a group that is able to solve a given (decision) task, taking into account availability aspects such as engagement in other projects. This scenario can become even more complex if we want to configure a set of groups to solve a specific task. Consider the following university-based task: Given that there are 300 students registered in a software engineering course, divide the population into groups of 6, such that each group is best suited to complete a specific project. A related issue is the analysis of inter-group influences, for example, in which way *influential groups* influence *susceptible groups* [54]. Further research issues are related to the topics of *evaluating group recommenders*, *explaining group recommendations*, *taking into account group dynamics*, and *counteracting biases* that trigger suboptimal decisions. These issues will be discussed in the following chapters of this book.

References

1. K. Arrow. The Difficulty in the Concept of Social Welfare. *Journal of Political Economy*, 58(4):328–346, 1950.
2. L. Baltrunas, T. Makcinskas, and F. Ricci. Group Recommendations with Rank Aggregation and Collaborative Filtering. In *4th ACM Conference on Recommender Systems*, pages 119–126, Barcelona, Spain, 2010.

3. S. Berkovsky and J. Freyne. Group-based Recipe Recommendations: Analysis of Data Aggregation Strategies. In *4th ACM Conference on Recommender Systems*, pages 111–118, Barcelona, Spain, 2010.
4. S. Berkovsky, J. Freyne, M. Coombe, and D. Bhandari. Recommender Algorithms in Activity Motivating Games. *ACM Conference on Recommender Systems (RecSys'10)*, pages 175–182, 2010.
5. L. Boratto and S. Carta. The Rating Prediction Task in a Group Recommender System that Automatically Detects Groups: Architectures, Algorithms, and Performance Evaluation. *Journal of Intelligent Information Systems*, 45(2):221–245, 2015.
6. L. Boratto, S. Carta, and G. Fenu. Investigating the Role of the Rating Prediction Task in Granularity-based Group Recommender Systems and Big Data Scenarios. *Information Sciences*, 378:424–443, 2017.
7. M. Brocco and G. Groh. Team Recommendation in Open Innovation Networks. In *ACM Conference on Recommender Systems (RecSys'09)*, pages 365–368, NY, USA, 2009.
8. R. Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction (UMUAI)*, 12(4):331–370, 2002.
9. L. Chen, G. Chen, and F. Wang. Recommender Systems Based on User Reviews: the State of the Art. *User Modeling and User-Adapted Interaction*, 25(2):99–154, 2015.
10. L. Chen and P. Pu. Critiquing-based Recommenders: Survey and Emerging Trends. *User Modeling and User-Adapted Interaction (UMUAI)*, 22(1–2):125–150, 2012.
11. Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A Short Introduction to Computational Social Choice. In *33rd conference on Current Trends in Theory and Practice of Computer Science*, pages 51–69, Harrachov, Czech Republic, 2007.
12. K. Christakopoulou, F. Radlinski, and K. Hofmann. Towards Conversational Recommender Systems. In *International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 815–824, San Francisco, CA, USA, 2016.
13. T. DePessemier, J. Dhondt, and L. Martens. Hybrid Group Recommendations for a Travel Service. *Multimedia Tools and Applications*, 76(2):2787–2811, 2017.
14. T. DePessemier, J. Dhondt, K. Vanhecke, and L. Martens. TravelWithFriends: A Hybrid Group Recommender System for Travel Destinations. In *9th ACM Conference on Recommender Systems, Workshop on Tourism Recommender Systems*, pages 51–60, 2015.
15. T. DePessemier, S. Dooms, and L. Martens. An Improved Data Aggregation Strategy for Group Recommenders. In *3rd Workshop on Human Decision Making and Recommender Systems (held in conjunction with the 7th ACM Conference on Recommender Systems)*, pages 36–39, Hong Kong, China, 2013.
16. T. DePessemier, S. Dooms, and L. Martens. Comparison of Group Recommendation Algorithms. *Multimedia Tools and Applications*, 72(3):2497–2541, 2014.
17. M. Ekstrand, J. Riedl, and J. Konstan. Collaborative Filtering Recommender Systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):81–173, 2011.
18. A. Felfernig, M. Atas, T.N. Trang Tran, and M. Stettinger. Towards Group-based Configuration. In *International Workshop on Configuration 2016 (ConfWS'16)*, pages 69–72, 2016.
19. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, and S. Polat-Erdeniz. An Analysis of Group Recommendation Heuristics for High- and Low-Involvement Items. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*, pages 335–344, Arras, France, 2017.
20. A. Felfernig and R. Burke. Constraint-based Recommender Systems: Technologies and Research Issues. In *ACM International Conference on Electronic Commerce (ICEC08)*, pages 17–26, Innsbruck, Austria, 2008.
21. A. Felfernig, G. Friedrich, D. Jannach, and M. Stumptner. Consistency-based Diagnosis of Configuration Knowledge Bases. *Artificial Intelligence*, 152(2):213–234, 2004.
22. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, and S. Reiterer. Toward the Next Generation of Recommender Systems. In *Multimedia Services in Intelligent Environments: Recommendation Services*, pages 81–98. Springer, 2013.

23. A. Felfernig, M. Schubert, G. Friedrich, M. Mandl, M. Mairitsch, and E. Teppan. Plausible Repairs for Inconsistent Requirements. In *21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 791–796, Pasadena, CA, 2009.
24. A. Felfernig, M. Schubert, and S. Reiterer. Personalized Diagnosis for Over-Constrained Problems. In *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, pages 1990–1996, Peking, China, 2013.
25. A. Felfernig, M. Schubert, and C. Zehentner. An Efficient Diagnosis Algorithm for Inconsistent Constraint Sets. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AIEDAM)*, 26(1):53–62, 2012.
26. A. Felfernig, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger, and F. Reinfrank. Group Decision Support for Requirements Negotiation. *Springer Lecture Notes in Computer Science*, 7138:105–116, 2011.
27. H. Garcia-Molina, G. Koutrika, and A. Parameswaran. Information Seeking: Convergence of Search, Recommendations, and Advertising. *Communications of the ACM*, 54(11):121–130, 2011.
28. S. Ghazarian and M. Nematbakhsh. Enhancing Memory-based Collaborative Filtering for Group Recommender Systems. *Expert Systems with Applications*, 42(7):3801–3812, 2015.
29. J. Guo, L. Sun, W. Li, and T. Yu. Applying Uncertainty Theory to Group Recommender Systems Taking Account of Experts Preferences. *Multimedia Tools and Applications*, pages 1–18, 2017.
30. F. Guzzi, F. Ricci, and R. Burke. Interactive Multi-party Critiquing for Group Recommendation. In *5th ACM Conference on Recommender Systems*, pages 265–268, Chicago, IL, USA, 2011.
31. S. Hong, C. Mao, Z. Yang, and H. Lai. A New Team Recommendation Model with Applications in Social Network. In *18th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 644–648, NY, USA, 2014.
32. X. Hu, X. Meng, and L. Wang. SVD-based Group Recommendation Approaches: An Experimental Study of Moviepilot. In *ACM Recommender Systems 2011 Challenge on Context-aware Movie Recommendation*, pages 23–28, 2011.
33. A. Jameson. More than the Sum of its Members: Challenges for Group Recommender Systems. In *International Working Conference on Advanced Visual Interfaces*, pages 48–54, 2004.
34. A. Jameson, S. Baldes, and T. Kleinbauer. Two Methods for Enhancing Mutual Awareness in a Group Recommender System. In *ACM Intl. Working Conference on Advanced Visual Interfaces*, pages 447–449, Gallipoli, Italy, 2004.
35. A. Jameson and B. Smyth. Recommendation to Groups. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 596–627. 2007.
36. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, and L. Chen. Human Decision Making and Recommender Systems. In F. Ricci, L. Rokach, and B. Shapira, editors, *Recommender Systems Handbook, 2nd Edition*, pages 611–648. Springer, 2015.
37. M. Kompan and M. Bielikova. Group Recommendations: Survey and Perspectives. *Computing and Informatics*, 33(2):446–476, 2014.
38. J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM*, 40(3):77–87, 1997.
39. Y. Koren, R. Bell, and C. Volinsky. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*, 42(8):30–37, 2009.
40. J. Levin and B. Nalebuff. An Introduction to Vote-Counting Schemes. *Journal of Economic Perspectives*, 9(1):3–26, 1995.
41. G. Linden, B. Smith, and J. York. Amazon.com Recommendations – Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
42. T. Mahmood and F. Ricci. Improving Recommender Systems with Adaptive Conversational Strategies. In *20th ACM Conference on Hypertext and Hypermedia*, pages 73–82, Torino, Italy, 2009.
43. J. Marquez and J. Ziegler. Preference Elicitation and Negotiation in a Group Recommender Systems. In *Interact 2015*, volume 9297 of *LNCS*, pages 20–37. Springer, 2015.

44. J. Masthoff. Group Modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers. *User Modeling and User-Adapted Interaction (UMUAI)*, 14(1):37–85, 2004.
45. J. Masthoff. Group Recommender Systems: Combining Individual Models. *Recommender Systems Handbook*, pages 677–702, 2011.
46. J. Masthoff. Group Recommender Systems: Aggregation, Satisfaction and Group Attributes. *Recommender Systems Handbook*, pages 743–776, 2015.
47. K. McCarthy, L. McGinty, B. Smyth, and M. Salamó. Social Interaction in the CATS Group Recommender. In *Workshop on the Social Navigation and Community based Adaptation Technologies*, 2006.
48. D. McDonald and M. Ackerman. Expertise Recommender: A Flexible Recommendation System and Architecture. In *Conference on Computer Support Cooperative Work*, pages 231–240, Philadelphia, PA, USA, 2000.
49. T. Nguyen. Conversational Group Recommender Systems. In *International Conference on User Modelling, Adaptation and Personalization (UMAP'17)*, pages 331–334. ACM, 2017.
50. T. Nguyen and F. Ricci. A Chat-Based Group Recommender System for Tourism. In R. Schegg and B. Stangl, editor, *Information and Comm. Tech. in Tourism*, pages 17–30. 2017.
51. F. Ortega, A. Hernando, J. Bobadilla, and J. H. Kang. Recommending Items to Group of Users using Matrix Factorization based Collaborative Filtering. *Information Sciences*, 345(C):313–324, 2016.
52. D. Pennock, E. Horvitz, and C. Giles. Social Choice Theory and Recommender Systems: Analysis of the axiomatic foundations of collaborative filtering. In *17th National Conference on Artificial Intelligence (AAAI)*, pages 729–734, Austin, TX, USA, 2000.
53. L. Quijano-Sánchez, D. Bridge, B. Díaz-Agudo, and J. Recio-García. A Case-Based Solution to the Cold-Start Problem in Group Recommenders. In *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, pages 3042–3046, 2013.
54. L. Recalde. A Social Framework for Set Recommendation in Group Recommender Systems. In *European Conference on Information Retrieval*, pages 735–743. Springer, 2017.
55. R. Reiter. A Theory of Diagnosis from First Principles. *AI Journal*, 32(1):57–95, 1987.
56. D. Sacharidis. Group Recommendations by Learning Rating Behavior. In *International Conference on User Modelling, Adaptation and Personalization (UMAP'17)*, pages 174–182. ACM, 2017.
57. C. Senot, D. Kostadinov, M. Bouzid, Gerome Picault, A. Aghasaryan, and C. Bernier. Analysis of Strategies for Building Group Profiles. In *Conference on User Modeling, Adaptation, and Personalization (UMAP 2010)*, volume 6075 of *LNCS*, pages 40–51, Big Island, Hawaii, USA, 2010.
58. C. Senot, D. Kostadinov, M. Bouzid, J. Picault, and A. Aghasaryan. Evaluation of Group Profiling Strategies. In *IJCAI 2011*, pages 2728–2733, 2011.
59. M. Stettinger and A. Felfernig. CHOICLA: Intelligent Decision Support for Groups of Users in Context of Personnel Decisions. In *ACM RecSys'2014 IntRS Workshop*, pages 28–32, Foster City, CA, USA, 2014.
60. D. Winterfeldt and W. Edwards. *Decision Analysis and Behavioral Research*. Cambridge University Press, 1986.
61. W. Wobcke, A. Krzywicki, Y. Kim, X. Cai, M. Bain, P. Compton, and A. Mahidadia. A Deployed People-to-People Recommender System in Online Dating. *AI Magazine*, 36(3):5–18, 2015.
62. Z. Yu, X. Zhou, Y. Hao, and J. Gu. TV Program Recommendation for Multiple Viewers based on User Profile Merging. *User Modeling and User-Adapted Interaction*, 16(1):63–82, 2006.