METRIC BASED RECOMMENDER SYSTEMS

Ali Akhtarzada, Cristian S. Calude, John Hosking

Abstract: *Information overload and an abundance of choices create situations where selecting one option becomes extremely difficult or even worse, a guessing game. Collaborative ranking systems address this problem by creating intelligent rankings of items based on user opinions aggregation. This paper presents a metric-based multi-criteria recommender system that can be used on non-rigid sets of criteria. These systems fare well with respect to accuracy, transparency and flexibility.*

Keywords: *Multi-dimensional recommender system, metric method.*

ACM Classification Keywords*: H.4: Information systems applications*

MSC*: 68N30: Mathematical aspects of software engineering*

Introduction

This paper presents metric-based multi-criteria recommender systems that can be used on non-rigid sets of criteria (that may be defined by the users of a system). The approach uses a novel concept of an ideal candidate, which is an aggregation of users' digital belief systems: the algorithm first calculates a hypothetical ideal candidate, which is then used as the pivot point.

The paper is structured as follows. First, we briefly present the motivation and related results. Then, an overview of the algorithm is informally described, an example scenario in which the algorithm is applied is discussed, and a detailed description of the algorithm itself is given. We then describe a proof of concept implementation of the algorithm and a user study evaluating the perceived accuracy of the algorithm and usability of the system. We finish with conclusions and a brief discussion of future work.

Motivation and related facts

The biggest motivating factor for recommendation systems in general is that of information overload. Our society produces more information than it produces anything else [13; 27; 28][1]. Information overload leads to situations where the inputs to one's decision making process exceed the "capacity to assimilate and act on the information as well as the ability to evaluate every alternative" [25]. Information overload has also been linked with negative psychological impacts created by the illusion that more choices lead to better results [19]. Recommendation systems generally use single criteria ratings that define how good an entity is. For example [15] uses a single 10 star rating for each movie for their recommendations. More recently, multi-criteria recommendation systems have become popular, as evidenced by Yahoo! Movies' recent movie recommender system. Various surveys and papers [1; 2] have shown the increase in accuracy multi-criteria rating systems can achieve and have indicted the need for research activity in this area. Because of the amount of research out there, other papers [31] present methods on how to efficiently evaluate a recommender system. Transparency, "the opposite of secrecy" [17], is important because it goes hand-in-hand with trust and accountability. Transparency increases trust, hence the acceptance of a recommendation [11]. Transparency increases accountability too, as seen in numerous situations presented in Wikileaks (http://www.wikileaks.org). Fundamentally, a multi-criteria system allows for more transparency because one sees how each rating is broken down to create the overall rating. Flexibility is also paramount: it allows users to participate with their own preferences and knowledge. There has also been extensive work done

[1] The amount of digitally stored information in 2009 was estimated to be 500 exabytes [29].

on content, collaborative and hybrid based recommender systems and social information filtering, see for example [5; 22; 7; 2; 24]. Demographic, utility and knowledge based systems have been proposed by [7]. Recently, matrix factorisation methods have been used in [16].

Multi-criteria approaches to recommendation and ranking systems have been considered in [1; 12]. The authors crawled Yahoo! Movies and extracted a number of movie ratings decomposed into 4 criteria; they found that using multi-criteria ratings allows for more accurate ratings than single ratings. One approach they used was to simply divide the multi-criteria problem into M single criteria recommendation problems, thus treating each criteria as a single rating. Another approach taken by [21] treats tags as multiple dimensions and first infers users' preferences for tags and then resultantly for items (movies).

Multi-criteria approaches to recommendation and ranking systems are subject to limitations which were first proved for voting systems. The most famous result, Arrow's Impossibility Theorem [4] (also known as Arrow's Paradox), states that no voting system can turn individual preferences into a global (community) ranking if there are three or more options to choose from and a few "innocent-looking" conditions (such as, non-dictatorship, Pareto efficiency, and independence of irrelevant alternatives) are satisfied. Another limitation may appear because of the lack of independence of preferences.

Metric-Based Algorithms

Overview

Our main algorithm applies on a system composed of 5 parts: the users, the entities, the value dimensions, the belief system and the ideal candidate. *Entities* are anything that the system is recommending or ranking. For example in a movie recommender system, the movies would be the entities. *Value dimensions* are a set of factors that influence the ratings of an entity. For example, taste and price are value dimensions that influence the ratings of menu items in a restaurant. All entities are defined over a set of value dimensions. *Users* collaborate within the system by rating an entity over the set of value dimensions. For example a user may rate price high and taste poor, or price low and taste excellent.

The *belief system* is personal to each user. Each user is allowed to tell the system what ideal value they want a value dimension to have, and how important that value dimension is to them. For example, most people's belief system would have a value of 'low' for the dimension 'price' but depending on your level of income, the importance of price may vary. Finally, the *ideal candidate* is the set of ideal value dimensions. The system determines the ideal value dimensions by aggregating all users' belief systems into an average. That is, if there were 2 people in the system, and user one's belief system had the ideal value for price set to high, and user two's belief system had the ideal value for price set to low, then the ideal candidate will have it's ideal value for price set to 'in between low and high'. The ideal candidate can be thought of as the belief system of a hypothetical user that takes everyone's opinions into account.

Value dimensions can be either hard or soft. Hard value dimensions are factual, such as the price of an item, or the location of a building. Soft value dimensions are subjective, i.e. an opinion. The major difference between a soft and hard value dimension is that a hard dimension cannot be rated, while a soft one can. While a hard value dimension cannot be rated, it's belief weight can still be set. The price of an entity is a factual piece of information (bar bargaining practices), so it is an example of a hard value dimension. On the other hand, quality is a subjective (hence soft) value dimension as there are no standard measurements to quality which is subject to individual perspectives.

The ideal candidate discussed above is the global ideal candidate (i.e. all users' belief systems are aggregated into one). The system also uses a local ideal candidate, which is simply equivalent to a single user's belief system. The distinction between a local and global ideal candidate results in two different types of rankings and two different types of recommendations—a global and local ranking and a global and local recommendation. A global ranking

Dimension	Low value	High value
Humour (d_1)	not funny	hilarious
Complexity (d_2)	no brainer	very complex
Action (d_3)	no action	action packed
Acting (d_4)	bad	excellent

Table 1: Ranges for each dimension.

	Inception				Rush Hour				Dumb and Dumber			
	d_1	d_2	d_3	d_4	d_1	d_2	d_3	d_4	d_1	d_2	d_3	d_4
Frodo	2	5	4	5	4	1	4	4	3	1	1	4
Gimly	2	4	4	5	5	1	3	4	4	1	1	5
Bilbo	1	5	5	4	4	2	4	4	5	2	2	5
Sam	2	5	5	4	5	3	5	5	5	1	2	4
Pippin	2	5	4	4	3	2	5	4	4	1	1	5
Average	1.8	4.8	4.4	4.4	4.2	1.8	4.2	3.2	4.2	1.2	1.4	4.6

Table 2: Ratings given to each dimension per user and an average of all the ratings per entity.

of entities is one which calculates distances between an entity and the global ideal candidate, and a local ranking calculates distances between an entity and a local ideal candidate. Likewise, a global recommendation uses the global ideal candidate, which represents a community at large, and a local recommendation uses the local ideal candidate (personal to each individual). The ideal candidate refers to the global ideal candidate unless explicitly stated otherwise.

The algorithm that has been developed for the multi-criteria recommendation process is based on a distance metric that calculates the distance between an entity and the ideal candidate. The distances are then weighted to take into account importance levels. Two types of recommendations can be performed. One recommends similar items by finding similar entities to a pivot entity. The second recommends any items that match each user's belief system. That is, the second method is the same as the ranking algorithm, except instead of using the ideal candidate, it uses the specific user's belief system.

An example

We present an example to illustrate the developed algorithms, formally defined in the next section. In the scenario we have defined 5 users, 3 entities and 4 criteria. The entities we are rating are movies and the value dimensions that they are rated over are d_1 = humour, d_2 = complexity, d_3 = action and d_4 = acting. The five different users have all rated each movie along a set of 4 value dimensions.

The data we use is shown in Table 1, Table 2 and Table 3. All the data is presented as ratings between 1 and 5. This means that dimensions are rated on a 5-point scale, as are all the weights. What the scales represent per dimension is described in Table 1.

From Table 3 we can loosely categorize each user as follows: Frodo likes a well balanced movie and the quality of acting is very important. Gimly likes complicated, preferably action movies with some comedy, where complexity and acting matters most to him. Bilbo likes simple funny movies but no one particular dimension is overly important to him. Sam likes funny and dumb movies, and would prefer no action, but is not that fussed about action movies; the dumb part is important to him. Pippin likes action movies with no complexity (important). In short: Frodo is well balanced, Gimly likes complicated action with comic relief, Bilbo likes simple action comedies, Sam likes comedy sans action and Pippin likes action sans comedy.

	Value dimensions				Weights			
	d_1	d_2	d_3	d_4	w_1	w_2	w_3	w_4
Frodo	3	3	3	5	2	2	2	5
Gimly	3	4	5	5	2	4	3	4
Bilbo	5	2	3	5	3	3	3	3
Sam	5	1	1	5	4	5	3	1
Pippin	1	1	5	5	5	5	5	1

Table 3: Users' belief system.

Dimension	Ideal value	Ideal weight
d_1	3.4	3.2
d_2	2.2	3.8
d_3	3.4	3.2
d_4	5.0	2.8

Table 4: The ideal candidate calculated from all the user belief systems.

The three entities the algorithms will be applied to are the movies 'Inception', 'Rush Hour' and 'Dumb and Dumber', the ratings of which can be seen in Table 2.

The first step is to calculate the ideal candidate, which is an average of all the belief systems. The ideal candidate can be seen in Table 4. From this table we can see that the most important dimension is d_2 with a height of 3.8. The dimension that matters least in the system is d_4. Next we can determine the rankings of the entities by calculating the additive distance between each entity's average dimensional ratings, and the ideal candidate. We also apply the ideal weights to the calculations. Table 5 shows the global rankings of the system because the distances are calculated from the *global ideal candidate*. There is also the concept of the *local ideal candidate* which calculates the distance between an entity and a belief system which is local to the user in question. So we may also have a local ranking for each user, which is shown in Table 6. The local rank list is also the recommendation list.

From Table 5 and Table 6 we see that, according to the data provided, the rankings make sense. In Table 6, the person who likes complicated movies gets Inception ranked highest, the person who likes fun and dumb movies gets Dumb and Dumber. Table 7 shows the rankings for Sam, after we take out all of his rating data. So the system now has no information on Sam's ratings, but retains his belief system. We can see that, while the distances have changed, the order of recommended movies is perfect for him.

In Table 8 we see the second recommendation method, which finds the similarity between entities. In that table Inception is more similar to Dumb and Dumber than it is to the Rush Hour. This is a result of using the global ideal candidate as the pivot point. Alternatively, we can use specific belief systems as the pivot point for computing similarities. This is shows in Table 9, which is specific to the user Gimly, who likes complicated action movies. In that table, it shows that Inception is more similar to Rush Hour than to Dumb and Dumber (which would hold true for a recommendation personal to Gimly). A system using this recommendation process would recommend Rush Hour to Gimly if he was looking at the Inception page.

Entity	Rank	Distance
Rush Hour	1	0.1506
Dumb and Dumber	2	0.1857
Inception	3	0.1983

Table 5: Rankings of each entity.

	Rank	Entity	Distance		Rank	Entity	Distance
Frodo	1^{st}	Inception	0.2402	Gimly	1^{st}	Inception	0.1541
	2^{nd}	Rush Hour	0.242		2^{nd}	Rush Hour	0.173
	3^{rd}	Dumb and Dumber	0.272		3^{rd}	Dumb and Dumber	0.2545
Bilbo	1^{st}	Rush Hour	0.14	Sam	1^{st}	Dumb and Dumber	0.0934
	2^{nd}	Dumb and Dumber	0.155		2^{nd}	Rush Hour	0.195
	3^{rd}	Inception	0.22		3^{rd}	Inception	0.2721
Pippin	1^{st}	Inception	0.163				
	2^{nd}	Rush Hour	0.1761				
	3^{rd}	Dumb and Dumber	0.2053				

Table 6: Local rankings of each entity for each user.

Sam		
Rank	Entity	Distance
1^{st}	Dumb and Dumber	0.2882
2^{nd}	Rush Hour	0.3647
3^{rd}	Inception	0.4514

Table 7: Recommended movies for Sam created from Sam's belief system and ignoring all his ratings in Table 2.

	Inception	Rush Hour	Dumb and Dumber
Inception	1.0	0.9523	0.9868
Rush Hour	0.9523	1.0	0.9655
Dumb and Dumber	0.9868	0.9655	1.0

Table 8: Similarities between each movie, with 1 representing full similarity.

Gimly			
	Inception	Rush Hour	Dumb and Dumber
Inception	1.0	0.9811	0.8997
Rush Hour	0.9811	1.0	0.9186
Dumb and Dumber	0.8997	0.9186	1.0

Table 9: Similarities between each movie, personalized to Gimly's belief system.

Main Algorithm

The algorithm developed for the multi-criteria recommender system in [3] uses a weighted sum approach which is defined in multi-objective optimization literature [14]. The goal of a recommendation system is to construct the $Users \times Items$ 2-dimensional matrix by predicting the missing values in the matrix. The approach we take involves the reconstruction of a 3-dimensional matrix, with the third dimension being the set of criteria defined over the items, i.e. the value dimensions. Additionally, we use a weighted approach so that more important value dimensions make more of a difference to the final calculations.

For the rest of this section we use the following notation. The set of users U has $n = |U|$ elements. The set of entities denoted by E has $m = |E|$ elements. The set of value dimensions is denoted by V; let $l = |V|$ be the number of dimensions for each entity. Finally, let W be the set of weights such that $|W| = |V|$.

Our goal is to predict values in the $n \times m \times l$ matrix. There are three concepts used by the proposed algorithm: 1) value dimensions (i.e. criteria), 2) the belief system, and 3) the ideal candidate. *Value dimensions* determine the ratings of each entity in E. Each entity is defined by l value dimensions, which are collaboratively rated and then normalized to the range $[0, 1]$ before being used as the input to the algorithm. Therefore, each entity is a vector $e = (v_1, v_2, \cdots, v_l)$ for all $v \in V$.

A *belief system* (see Table 3) allows each user to define their beliefs using two components: 1) the values for each criteria and 2) the weights attached to each criteria. The weights are normalized to the range $[0, 1]$ with 0 representing no importance and 1 indicating utmost importance. Formally, each user u has a belief system B, which is the ordered pair $B_u = (v_u, w_u)$ where $v_u = (v_1, v_2, \cdots, v_l), w_u = (w_1, w_2, \cdots, w_l)$ are vectors and v_i, w_i represent the user's preferred value for value dimension i and weight i, respectively.

The *ideal candidate* is used as the pivot point for all distance calculations. Instead of calculating the distance of entities from the origin or from other entities, the algorithm makes use of a hypothetical ideal entity that is an aggregation of each users' belief system (see Table 4). Formally, the ideal candidate is an ordered pair $I = (v_I, w_I)$, where $v_I = (v_1, v_2, \cdots, v_l), w_I = (w_1, w_2, \cdots, w_l)$ and are calculated as follows:

$$I = \frac{1}{N}\sum_{i=1}^{N} B_i = \left(\frac{1}{N}\sum_{i=1}^{N} v_i, \frac{1}{N}\sum_{i=1}^{N} w_i\right). \tag{1}$$

Equation (1) is referred to as the global ideal candidate I_G, which takes into account every user's belief system. The local ideal candidate I_L is specific to each user and is simply equal to B_u. The ideal candidate is an entity as well, hence any algorithm that can calculate the distance or similarity between two entities can operate similarly with the ideal candidate.

Discrete Metrics and Similarities

Let $\mathbb{M}$ be a nonempty set of nonnegative real numbers with the greatest element $a = 1$. Then $d : \mathbb{M} \times \mathbb{M} \to \mathbb{R}$ is a metric on M and the ordered pair $(\mathbb{M}, d)$ is a metric space [9]:

$$d(x, y) = \begin{cases} \frac{1}{2}(1 + |x - y| - |1 - x - y|), \text{if } x \neq y, \\ 0, \text{if } x = y, \end{cases} \tag{2}$$

For our multi-criteria problem we can naturally extend the metric to $\mathbb{M}^l$

$$d_l(x, y) = \frac{1}{l}\sum_{i=1}^{l} d(x_i, y_i), x, y \in \mathbb{M}^l, \tag{3}$$

and to a weighted metric $d_w : \mathbb{M}^l \times \mathbb{M}^l \times \mathbb{M}^l \to \mathbb{R}$:

$$d_l(x,y,w) = \frac{1}{l}\sum_{i=1}^{l}(1-w_i)d(x_i,y_i), x,y,w \in \mathbb{M}^l \tag{4}$$

The weights have to be normalized in the range $0 \leq w \leq 1$. From Table 3 we can see that the higher the weight w_i is, the more important it is. The more important a weight is the less it increases the distance results. Therefore, as a weight approaches 0, it's importance is reduced and the distance is increased. See more details in [3].

From equation (4) we can define a multi-criteria rating function in terms of $e \in E$, $u \in U$ and I. We define two rating functions, the global rating function $r_G : E \times I \to \mathbb{R}$:

$$r_G(e, I_G) = d_w(e, v_I, w_I), \tag{5}$$

and the local rating function $r_L : E \times I \to \mathbb{R}$:

$$r_L(e, I_L) = d_w(e, v_u, w_u). \tag{6}$$

The difference between r_G and r_L is that the global rating function calculates the distance between an entity and the global ideal candidate I_G while the local one calculates the distance between the local ideal candidate I_L.

There are two ways in which recommendations can be made using the algorithms. The first is a ranking of items obtained by using function (6). The results of this can be seen in Table 7. The second method uses a similarity metric to recommend entities that are similar to other entities. The similarity method can also be divided into two functions, one which uses the global ideal candidate I_G to obtain global recommendations, and one which uses a local ideal candidate I_L to obtain personalized recommendations.

From [10], any normalized distance metric d can be converted into a similarity metric s defined as follows: $s = 1 - d : 0 \leq d \leq 1$. Equations (2), (3) and (4) are normalized distance metrics that return a value in the range $[0,1]$. If d is a normalized distance between two entities e_1 and e_2, then we can define a global similarity metric s_G as:

$$s_G(e_1, e_2, I_G) = 1 - |d_w(e_1, v_I, w_I) - d_w(e_2, v_I, w_I)|, \tag{7}$$

and the local similarity metric as:

$$s_L(e_1, e_2, I_L) = 1 - |d_w(e_1, v_u, w_u) - d_w(e_2, v_u, w_u)|. \tag{8}$$

Since s_G and s_L are both normalized similarity metrics, because d is normalized, they both satisfy the following "coherence" properties [10] for all x, y, z, I: $s_{G|L}(x,x,I) \geq 0$, $s_{G|L}(x,x,I) \geq s_{G|L}(x,y,I)$, $s_{G|L}(x,y,I) = s_{G|L}(y,x,I)$, $s_{G|L}(x,y,I) + s_{G|L}(x,z,I) \leq s_{G|L}(x,z,I) + s_{G|L}(y,y,I)$, $s_{G|L}(x,x,I) = s_{G|L}(y,y,I) = s_{G|L}(x,y,I) \iff x = y$.

The following functions showcase part of the pseudo code used for the developed prototype system. The function *distance* represents equation (4), the function *rating* represents equations (6), (5), and the function *similarity* represents equations (7,) (8).

```
Function distance( rating, ideal, weight )
  min = 1
  max = 5
  x = (min - rating) / (min - max)
  y = (min - ideal) / (min - max)
  if x == y, return 0
  return (1 - weight) * (0.5 * (1 + |x - y| - |1 - x - y|))
End

Function rating( entity )
```

```
  num_dims_used = 0
  total_distance = 0
  For each dimension dim in entity.category
    rating = average_rating(dim)
    total_distance += distance(rating, dim.ideal, dim.weight)
    num_dims_used += 1
  End For
  // Return a rating in between 0 and 1 (1 means perfect)
  return 1 - (num_dims_used / total_distance)
End

Function similarity( entity1, entity2 )
  return 1 - |rating(entity1) - rating(entity2)|
End
```

Implementation

A proof of concept system was developed for a local restaurant in Auckland, New Zealand. Users are required to sign up to obtain personalized recommendations. Signing up also requests demographic information so it can be used for extracting intelligence from the data (segmenting user preferences in to region, for example). Signed in users can add or edit a value dimension, set their beliefs, rate a menu item or view details about a menu item.

Adding a value dimension and editing a value dimension use the same input screen. End users can only add soft value dimensions. Only administrators can add hard value dimensions, such as Price, via a different input screen. Once a hard value dimension is added it becomes part of an entity's profile. For example, having added Price, each priced entity has attached to it a new property, called price, the value of which can be changed on the entity's edit page. Furthermore, the type of scale that is used can be chosen by the user as well. The prototype implementation supports the addition of ordinal and nominal scales. The user must also specify which category the value dimension applies over.

Once value dimensions are provided, users can set their beliefs. Figure 1 shows the belief entry screen. Two columns are shown, one for the ideal value for a belief and the other for the weight of the value dimension. There are two scale types: one is an ordinal scale with a low and high range and the other a nominal scale. These ideal values are used to calculate the global ideal value and importance of a dimension.

Users can also obtain valuable statistics at this point. For example clicking on the value dimension 'healthiness', you will be informed that 44 people in the system care about this dimension and that most people think that the healthiness of mains, desserts and starters should be, ideally, almost as healthy as you can get and that while healthiness matters, it is not vital nor is it very important in determining the rating of food. You will also be told that while healthiness matters somewhat in the United States, it's more important in Germany, and less important in Canada. This is of course reflective of the beliefs of the system's users.

The prototype allows users to rate menu items over all soft value dimensions entered into the system; hard value dimensions are facts and can only be changed by the administrators of the system. The rating screen can be seen in Figure 2. Users are free to ignore certain value dimensions. The more a value dimension is ignored, the more obvious it becomes that this value dimension is not worth having in the system. A straightforward extension would be for the system to provide a confidence level to each value dimension's importance level.

The system can perform two types of recommendations, one based on entity-to-entity similarity and one showing users predicted ratings of all the entities in the system (that have been rated by anyone). The second type of recommendation does not depend on entity-to-entity similarity, nor on user similarity, only on the similarity between a user's belief system and the entity's profile (which is composed of value dimensions and their aggregated values). The first type of recommendation (entity-to-entity) is shown in Figure 3. It uses the user's belief system as the pivot point if the user is logged in.

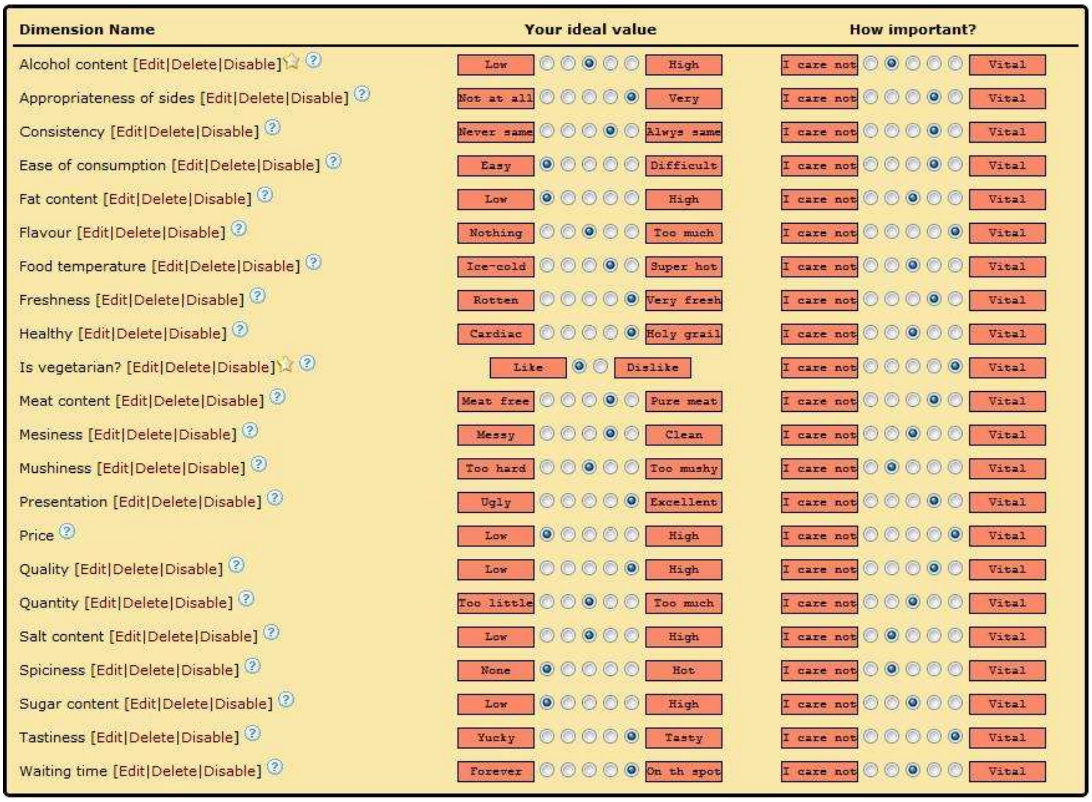

Figure 1: Setting your beliefs involves setting an ideal value and a weight.

The system will list all the details about the entity in question. This includes how many users have rated it, what the global rating is, what the predicted rating for the user is and any hard values. It will then show the user the rating screen in the case that a user wants to rate it, and this will be followed by a number of item-to-item recommendations based on the similarity calculations between this entity and every other entity, with the user's belief system used as the pivot point.

Evaluation

A user survey was carried out on a random sample of 20 patrons of the restaurant that the prototype was developed for. Each user was asked to use the system and rate items that they had tried. They were then given a list of 5 recommendations and asked to rank the algorithms in order from most to least accurate. We obtained 85 unfiltered orderings of the algorithms. After filtering them out for incomplete orderings we had 78 orderings. In addition to ranking the algorithms that were employed, the participants of the survey were also asked a number of questions regarding the usability of the system. The following five algorithms were rated:

1. Algorithm A: Rating based on global ideal candidate and average ratings–equation 5.
2. Algorithm B: The predicted rating based on user's belief system and average ratings–equation 6.
3. Algorithm C: Weighted sum [2].
4. Algorithm D: Adjusted weighted sum [2].
5. Algorithm E: Rating based on global ideal candidate and user's personal ratings.

The results of the algorithm rankings are presented in Table 10. The first 5 columns of the table represent how many times the algorithm was ranked at that position. So, the Algorithm B was ranked first 29 times, second 14 times, etc.

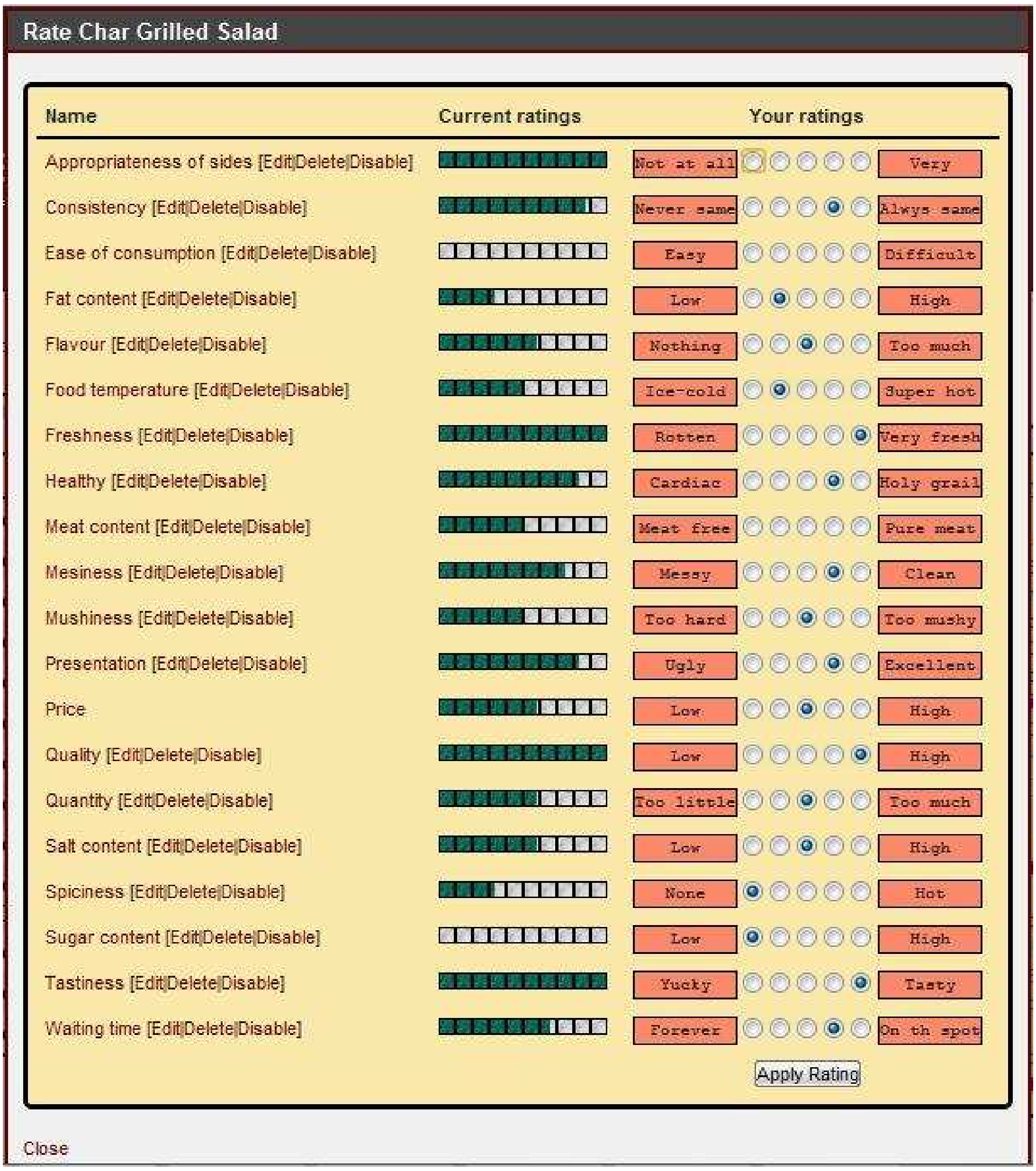

Figure 2: Rating a menu item.

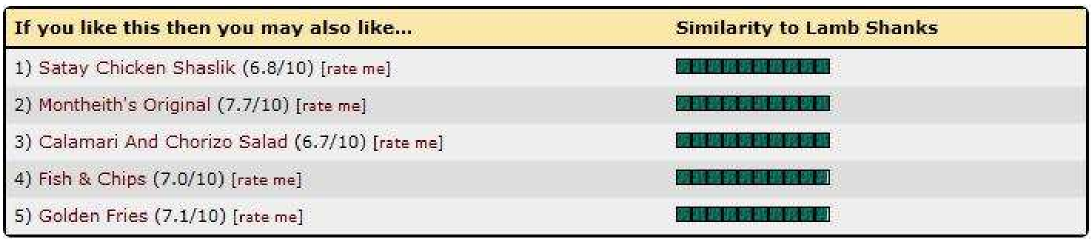

Figure 3: Recommended items when viewing the Lamb Shank page.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	$\%1^{st}$	Linear rating	Exp rating	% exp	% lin
Algorithm A	13	21	32	8	4	16.7	265	1048	21.4	22.4
Algorithm B	26	14	12	24	3	33.3	273	1250	25.6	23.1
Algorithm C	12	21	18	21	7	15.4	247	962	19.7	20.9
Algorithm D	7	4	4	6	58	9	133	460	9.4	11.2
Algorithm E	21	19	13	19	7	27	265	1170	24	22.4

Table 10: The rankings of each of the algorithms and their additive scores.

The algorithm that has been proposed in this paper is Algorithm B. It can be seen from the results that it does better than all the others. Algorithm A is also used by the system, not for personal results, but for global rankings of items. The only difference between A and B is that B uses the user's personal belief system to calculate the ratings and A uses the global ideal candidate. As it turns out, the personalized algorithm, based on a digitized belief system, is the most accurate of them all.

A number of other statistics were also calculated. The $\%1^{st}$ column represents the percentage that the algorithm came first. Algorithm B dominates this by being the most accurate 33 percent of the time out of all 5 algorithms and algorithm E comes in second with 27 percent. The 2 algorithms together, both of which add in a personal aspect to the rating process, are the most accurate 60 percent of the time.

The next 4 columns are an additive ranking that was calculated for each algorithm by taking into account the number of times the algorithms came in each position rather than in just first spot. Each position was given a weighting and then a total rating was calculated. The algorithm with the highest rating wins, and again it shows that B came in on top with A and E a close second. One may note here that even though algorithm E came in first place almost 2 times more frequently than algorithm A, their linear rating is exactly the same. This happens because algorithm A came in third place almost 3 times more frequently than algorithm E, which linearly made a significant difference.

The linear rating for algorithm L was calculated using the formula $rating_L = \sum_{i=1}^{M}(M-i) * L_i$ and the exponential rating was calculated with $rating_L = \sum_{i=1}^{M} 2^{M-i} * L_i$, where L_i represents the number of times algorithm L came on the i^{th} place. The linear rating gives a consistent weight to each of the positions, which assumes that the most accurate algorithm is worth as much more than the second most accurate algorithm is worth over the third most. The exponential model seems to be more accurate; it gives more value to the difference in weight between the first and second place than the difference in weight between the second and third place.

Methods on evaluating recommender systems are presented in [31]. The evaluation method we used was a mix of a user study and online study. While the users were given specific tasks to do, those tasks were prerequisites for allowing/encouraging them to roam free within the system. This allowed us to also collect data that was not relevant to the algorithm presented in this paper, but on the dynamics of the user interface, the experienced speed of the calculations and the perceived accuracy of the results (described above).

A number of properties are considered desirable in a recommender system [31]. *Prediction accuracy* was shown to be better than for some of the other algorithms. *Ranking measures* was explicit in the system because the recommendations use the user's own belief system (i.e. ideal candidate) as the pivot point to determine the similarity between items. Measuring the accuracy of this ranking would require the user to have tried all the dishes in the list (given the nature of the prototype) and so was unfeasible to perform.

Item coverage is an important property for a recommendation system. While some algorithms may be limited to recommending only items that have a lot of associated data, our algorithm has 100 percent coverage. The accuracy of cases where one item has a large number of ratings and another item has only one was not tested, but adding a confidence level to ratings is a technique that may be used to alleviate that situation. A similar property is user coverage (i.e. providing recommendations based on a given number of users within the system). The prototype we developed provides full coverage, and may also be enhanced with the addition of a confidence level.

One flaw of this algorithm is that it is extremely susceptible to the cold start problem, in which items that have not been given data would be quite useless. If value dimensions have not been given importance levels, or belief systems have not been defined, then the system would be unusable.

Conclusion

The system that has been implemented is a proof of concept for the envisioned final system. The current implementation provides a ranking based on any number of value dimensions, with weights and ideal values definable by the users. There are many directions the work on this framework can take.

First, the ontology system, which provides the ability for users to arrange the entities under categories and link categories if need be, can be made much more robust with the addition of the ability to categorize. It is still not clear if a universal ontology would be best for the framework, or if other methods should be used to categorize entities into classes of value dimensions. One such method would be collaborative tagging as described in [26]. That is, instead of arranging the structure in a directory-like manner, the users can tag a category instead, and the entities within that group will implicitly have the same tags, creating a folksonomy. The value dimensions could be attached to tags instead of the category as well. This may work better to enable the ability to share value dimensions across categories as well as impressive recommendations.

Secondly, the governance mechanisms have not been implemented in a way that is scalable. The governance mechanisms determine how the system governs itself at the micro and macro levels. Concepts such as policy citations on ‘talk-pages’ enhance governance at the micro-level on website such as Wikipedia [6]. The macro-level governance mechanisms (namely the aggregation of the belief systems and rankings) are in place, but there is a lot of work that needs to be done at the micro-level.

Thirdly, work is needed on trust and reputation support. Incorporating trust into users’ profiles would be one direction. Perhaps the framework should take a user’s reputation into account when applying the value dimension weightings in the algorithms, maybe just for the global rankings. These extra calculations could, however, give rise to performance. The current system implementation can be slow at times because it is running five different algorithms at the same time for the purpose of allowing users to rank various algorithms (as discussed in Section **Evaluation**).

And finally, the system does not take temporal dynamics in to account. As time goes by, items become less valuable and no longer appealing in the same way. One approach that tackles this problem keeps degradation information along with the rating data and creates a temporal model out of the rating patterns [33].

Giving users less work to do in the system, i.e. by automating certain processes, is also desirable. One example may be the weighting system for value dimensions. The trust system [30] developed for Wikipedia could be used to create implicit weightings over the value dimensions by determining collaborative importance through analyzing a value dimension’s revision history.

Knowledge extraction mechanisms are needed for the framework to be useful. This is the component that will provide the most value to users in the long run. The ranking algorithm is a central part of the entire knowledge extraction system. This part can answer questions such as: What is a positive contribution to the world? What is a negative contribution to this world, and why? What is the most important value-dimension for universities, for countries, or for various businesses?

A problem faced by this recommendation system and many others is the cold-start problem. How do we get initial input form users so that the system gets in to a usable state. Wikipedia addressed this by appealing to the expert community [6]. Other systems, which are not that appealing to "experts" have adopted a hybrid approach [32] that assigns ratings to items based on their similarity to other items that have already been rated, e.g. in a trial run of the system.

Another area of future work is method to enable iterative aggregation, that is, how to enable the inclusion of nominal value dimensions.

The evaluation section showed that user’s ranked the proposed algorithm at a higher *subjective* accuracy than the other algorithms that were implemented. Another area of research in this regard would be to implement various other algorithms and carry out another evaluation study against the proposed system. Furthermore, [31] pointed out a number of evaluation criteria for recommendation systems that were not included in this evaluation, such as the user's intent (which in our case was to test the system since they were hand picked—but this is not the intent of a real user of a recommendation system), how much they trust the recommendations, the novelty of a recommendation (i.e. whether the user knew about the recommended item or not), serendipity, diversity, utility, adaptability and scalability.

Acknowledgements

We thank M. Stay (Google, Mountainview) and the anonymous referee for comments which improved the paper and Professor Michel Petitjean for generous support. The paper is published with financial support by the project ITHEA XXI of the Institute of Information Theories and Applications FOI ITHEA (www.ithea.org) and the Association of Developers and Users of Intelligent Systems ADUIS Ukraine (www.aduis.com.ua).

Bibliography

[1] Adomavicius, G. and Kwon, Y. (2007). New recommendation techniques for multi-criteria rating systems. *Intelligent Systems*, 22(3), pp.48-55.

[2] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), pp.734–749.

[3] Akhtarzada, A., Calude, C. S., Hosking, J. (2011). A multi-criteria metric algorithm for recommender systems, *Fundamenta Informaticae* 110, 1–11.

[4] Arrow, K. J. (1950). A difficulty in the concept of Social Welfare. *Journal of Political Economy*, 58(4), pp.328–346.

[5] Belkin, N. J. and Croft, W. B. (1992) Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12), pp.29–38.

[6] Beschastnikh, I., Kriplean, T. and W. McDonald, D. (2008). Wikipedian Self-Governance in Action: Motivating the Policy Lens. *International AAAI Conference on Weblogs and Social Media*, Menlo Park, CA, AAAI Press.

[7] Burke, R. (2002). Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4), pp.331–370.

[8] Calude, C. and Calude, E. (1982). A metrical method for multicriteria decision making. *St. Cerc. Mat.* 34, pp.223–234. (in Romanian)

[9] Calude, C. and Calude, E. (1983). On some discrete metrics. *Bull. Math. Soc. Sci. Math. R. S. Roumanie (N. S.)* 27(75), pp.213–216.

[10] Chen, S., Ma, B. and Zhang, K. (2009). On the similarity metric and the distance metric. *Theoretical Computer Science*, 410(24-25), pp.2365–2376.

[11] Cramer, H., Evers, V., Ramlal, S., Someren, M. v., Rutledge, L., Stash, N., Aroyo, L. and Wielinga, B. (2008). The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User Adapted Interaction*, 18(5), pp.455–496.

[12] Dinu, L. P., Popescu, M. (2008). Multi-criteria decision method based on rank distance, *Fundamenta Informaticae* 86(1-2), pp. 79–91.

[13] Edmunds, A. and Morris, A. (2000). The problem of information overload in business organizations: a review of the literature. *International Journal of Information Management*, 20(1), pp.17–28.

[14] Fishburn, P. C. (1967). Additive utilities with incomplete product sets: application to priorities and assignments. *Operations Research*, 15(3), (May - Jun., 1967), pp. 537–542.

[15] IMdB ([Online]) http://www.imdb.com. Visited at 10, May, 2009.

[16] Koren, Y., Bell, R. and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), pp.30–7.

[17] Meyer, P. D. (2003). The truth about transparency. *ASAE and The Center for Association Leadership*, http://www.asaecenter.org/PublicationsResources/EUArticle.cfm?ItemNumber=11786. Visited at Sept 30, 2010.

[18] Schafer, J. B. , Konstan, J. and Riedl, J. (2002). Meta-recommendation systems: User-controlled integration of diverse recommendations. *Proceedings of the 2002 International ACM CIKM Conference on Information and Knowledge Management*, pp.43–51. McLean, VA, USA.

[19] Schwartz, B. (2003). *The Paradox of Choice. Why More Is Less*. Ecco Publishing, New York.

[20] Sen, A. (1979). Personal utilities and public judgements: or what's wrong with welfare economics. *The Economic Journal*, 89(355), pp. 537–55.

[21] Sen, S., Vig, J. and Riedl, J. (2009). Tagommenders: connecting users to items through tags. *Proceedings of the 18th international conference on World wide web*, pp.671–680, ACM, New York, NY, USA.

[22] Shardanand, U., Maes, P. (1995) Social information filtering: algorithms for automating "word of mouth". *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM/Addison-Wesley, Denver, Colorado, USA, pp.210–217.

[23] Stuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation and Application*. Wiley and Sons, Inc.

[24] Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009 (Jan. 2009), 2-2. Hindawi Publishing Corporation.

[25] Walker, J. P. (1971). Decision-making under conditions of information overload: alternative response modes and their consequences. *American Educational Research Association Annual Meeting*, New York, USA.

[26] Xu, Z., Fu, Y., Mao, J. and Su, D. (2006). Towards the semantic web: collaborative tag suggestions. *Workshop on Collaborative Web Tagging*, Edinburgh, Scotland.

[27] Kelly, K. (2006). The speed of information. *The Technicum*, http://www.kk.org/thetechnium/archives/2006/02/the_speed_of_in.php. Visited Oct 30, 2010.

[28] Lyman, P and Varian, H. R. (2003). How much information? *School of Information Management and Systems, University of California, Berkley*, http://www2.sims.berkeley.edu/research/projects/how-much-info-2003. Visited Oct 30, 2010.

[29] Wray, R. (2009). Internet data heads for 500bn gigabytes. *The Guardian*, http://www.guardian.co.uk/business/2009/may/18/digital-content-expansion. Visited May 14, 2010.

[30] Adler, B., T., Chatterjee, K., de Alfaro, L., Faella, M., Pye, I. Raman, V. (2008). Assigning trust to Wikipedia content. *WikiSym 2008: International Symposium on Wikis*, Porto, Portugal.

[31] Shani, G., Gunawardana, A. (2011). Evaluating recommendation systems. *Recommender Systems Handbook*, pp.257–297. Springer US, 2011.

[32] Schein, A. I., Popescu, A. Ungar, L. H., Pennock, A. M. (2002). Methods and metrics for cold-start recommendation systems. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pp.253–260. ACM 2002.

[33] Koren, Y. (2009). Collaborative filtering with temporal dynamics. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, pp.447–456. ACM 2009.

Authors' Information

Ali Akhtarzada *- Department of Computer Science, University of Auckland, Auckland, New Zealand; e-mail: ali.akhtarzada@gmail.com*
Major Fields of Scientific Research: Software engineering, multi-dimensional information systems

Cristian S. Calude *- Department of Computer Science, University of Auckland, Auckland, New Zealand; e-mail: cristian@@cs.auckland.ac.nz*
Major Fields of Scientific Research: Algorithmic information theory and quantum theory

John Hosking *- ANU College of Engineering and Computer Science Australian National University, Canberra, Australia; e-mail: john.hosking@anu.edu.au*
Major Fields of Scientific Research: Software engineering, software tools and meta-tools, visual languages