International Journal of Physical Sciences Vol. 7(15), pp. 2327 - 2338, 9 April, 2012
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.1599
ISSN 1992 - 1950 © 2012 Academic Journals

Full Length Research Paper

An interactive evolutionary approach to designing novel recommender systems

Hyun-Tae Kim and Chang Wook Ahn*

Department of Computer Engineering, Sungkyunkwan University (SKKU) Cheoncheon-dong, Suwon 440-746, Republic of Korea.

Accepted 20 January, 2012

Recently, recommender systems have been widely used in e-commerce websites to help customers discover items they want. Since a recommender system should be able to provide users with helpful information regarding items that might interest them, the ability to immediately respond to changes in a user's preference is a valuable asset of the systems. Thus, this work presents a novel recommender system that effectively adapts and immediately responds to any changes in the system by utilizing content-based filtering within the framework of interactive evolutionary computation. In addition, a data grouping technique is employed to enhance the computational time efficiency. The proposed system is then tested with music data. Experimental results confirm that the proposed system is able to offer users suitable music items with assured quality in a timely manner. Furthermore, the experimental results suggest that this system provides more reliable music recommendations than other content-based filtering systems even in the working environment changes.

Key words: Recommender system, information filtering, interactive evolutionary computation.

INTRODUCTION

In daily life, people often face decision-making situations without having sufficient information on the various alternatives. Consequently, people may have a difficult time making the correct decision since the number of alternatives may be very large. As a result, people may be more likely to ask someone for help, such as a friend having extensive knowledge on the subject matter or an article related to the decision. Yet, while the recommendations received by consulting friends or other materials may be beneficial in making the appropriate decision, they may not align well with the person's preferences. Recommender systems can be useful in this regard: they can provide the person or user with a refined recommendation based on alternatives that are well tailored to the user's preferences (Resnick and Varian, 1997; Terveen and Hill, 2001). Recommender systems are developed out of the desire (or need) to efficiently process the vast amount of information available to today's consumers.

*Corresponding author. E-mail: cwan@skku.edu. Tel: +82 31 299 4588. Fax: +82 31 299 4664.

Since the 1990s, technology has undergone impressive changes in regards to information and entertainment. With these changes, numerous resources, such as TV channels, books, music and interactive documents on the World Wide Web, have been made available to a wide audience of consumers. The web, in particular, provides an environment in which people can communicate with one another and can easily access a broad array of resources (Schafer et al., 1999).

Yet, because of this asset, many consumers are overloaded with information (that is, information overload problem). They must sort through multitude of alternatives before choosing items suitable to their preferences or needs. In response to this dilemma, many e-commerce sites, such as Amazon, eBay and LastFM have employed recommender systems. These systems suggest items to their customers that are likely to fit the customers' demands (Linden et al., 2003; Sarwar et al., 2000; Schafer et al., 1999).

In general, there are two main recommendation methods: content-based filtering and collaborative filtering. The first method, content-based filtering, pro-vides a recommendation by classifying items in a particular way.

This method computes the similarity between the profile, which is composed of ratings or evaluations that represent a user's preference. On the other hand, in the collaborative filtering, the recommender system builds a database of profiles, and then recommends items based on how well other users' profiles match the database. Similarly, other approaches have attempted to combine the two methods in an effective way, suggesting that the current recommender systems can be further improved. Consequently, this topic is still being investigated with great interest (Adomavicius and Tuzhilin, 2005; Burke, 2007).

Interactive evolutionary computation (IEC) has been successfully applied to real-world problems in diverse areas (Kim et al., 2010; Marques et al., 2010; Takagi, 2001). Constantly evolving with user's evaluation, IEC is apt to trace the user's uncertain, time-varying preferences. Yet, owing to the use of evolutionary operations, IEC may run a long while in order to discover an (near) optimal solution. Since IEC tries to make the best decision under the given condition, it can rapidly return a solution workable for a given problem. Initially, the quality of solution may be poor, but it would gradually improve as more interactions happen. Note that most real-world applications of IEC operate in a steady state. Recommender systems, robotics, music and graphics belong to representative examples (Kim et al., 2010; Marques et al., 2010; Takagi, 2001). It denotes that IEC can be used with no concern about the time complexity. In addition, to improve the performance of IEC (that is, alleviate user's fatigue), all the unevaluated date can be objectively assessed via the user's subjective information that has already been evaluated (Hao et al., 2009; Kim et al., 2010). As a result, IEC is suitable to the design of recommender systems.

In this paper, a novel recommender system is proposed that combines the content-based filtering technique with IEC and then applies data grouping to enhance the recommendation speed. This proposed system is then evaluated with music data and their extracted features.

RELATED WORK

Recommender system

The purpose of a recommender system is to provide relevant information to a user. This information should fulfill the user's needs by being pertinent and interesting. Information provided by an information retrieval system, on the other hand, should respond to a user's inquiry. In both systems, the user expects to obtain helpful information from the search results; however, some differences between the two systems exist. In particular, the definition of information is different in each system. In the information retrieval system, the information is defined by the degree of matches between the user's query and the system's alternatives. In that system, any method (for example, relevance feedback) can be employed that help to refine the query and make simple recommendations (Salton and Buckley, 1990). Hence, a recommender system can be regarded as an extension of an information retrieval system (Balabanovi'c and Shoham, 1997; Burke, 2007; Resnick et al., 1994).

Collaborative filtering versus content-based filtering

In recommender systems, the usefulness of an item is generally represented by a rating, which declares how a user likes the item. The rating for each item can then be used as a criterion for making future decisions regarding that item. A recommender system collects each user's rating and stores it as a profile. After gathering users' profiles, the system applies recommendation methods to the profiles to make suitable suggestions.

In the collaborative filtering method, the recommender system discovers relationships between the profiles stored in the system. After computing the correlations for entire profiles, the system groups similar profiles together. The system then provides recommendations derived from other users' profiles with similar ratings in the past (Cohen and Fan, 2000; Goldberg et al., 1992; Pazzani, 1999; Resnick et al., 1994). In the content-based filtering method, the recommender system extracts sets of features from the items, which are then assessed by the users. After that, the system analyzes the similarities between the items in a user's profile and the other remaining items. Finally, the system recommends items that possess similar features to the items the user preferred (Logan, 2002; Pazzani, 1999). Figure 1 shows the overall procedures of each method.

Nevertheless, these existing methods have some defining limitations. For example, in both the collaborative and content-based filtering methods, when a set of new items is added to the recommender system, they lack a user rating. Due to the absence of the user's preference, the recommender system using these methods cannot make any proper recommendations for new items. The content-based filtering method, however, also suffers from an overspecialization problem whereby it only suggests items that are directly related to items rated highly by the user. Therefore, the recommender system employing the content-based filtering is limited by only suggesting items similar to the specific items that the use rated favorably in the past.

In this study, we propose a new recommender system that takes into account the limitations of the existing recommendation methods.

Feature extraction

In content-based filtering, feature extraction is an important technique that acquires the unique properties of a given item, such as a document, music track or photo. Thus, the technique is suitable for data classification. As mentioned earlier, the proposed recommender system will make use of the content-based filtering method. Thus, feature extraction is also essential to our recommender system.

Interactive evolutionary computation (IEC)

Evolutionary computation (EC) is the general term for several computational techniques that are based on the evolution of biological life in the natural world. The most widely used application of EC is in genetic algorithms (GAs). GAs are stochastic search methods inspired by the mechanisms of natural evolution and genetic inheritance. GAs work on a population of candidates solutions. Each solution has a fitness value indicating its proximity to the optimal solution of the problem. The solutions with higher fitness values are selected and survive to the next generation. Thus, GAs produce better solutions (that is, offspring) via a combination of selected solutions. The candidate solutions

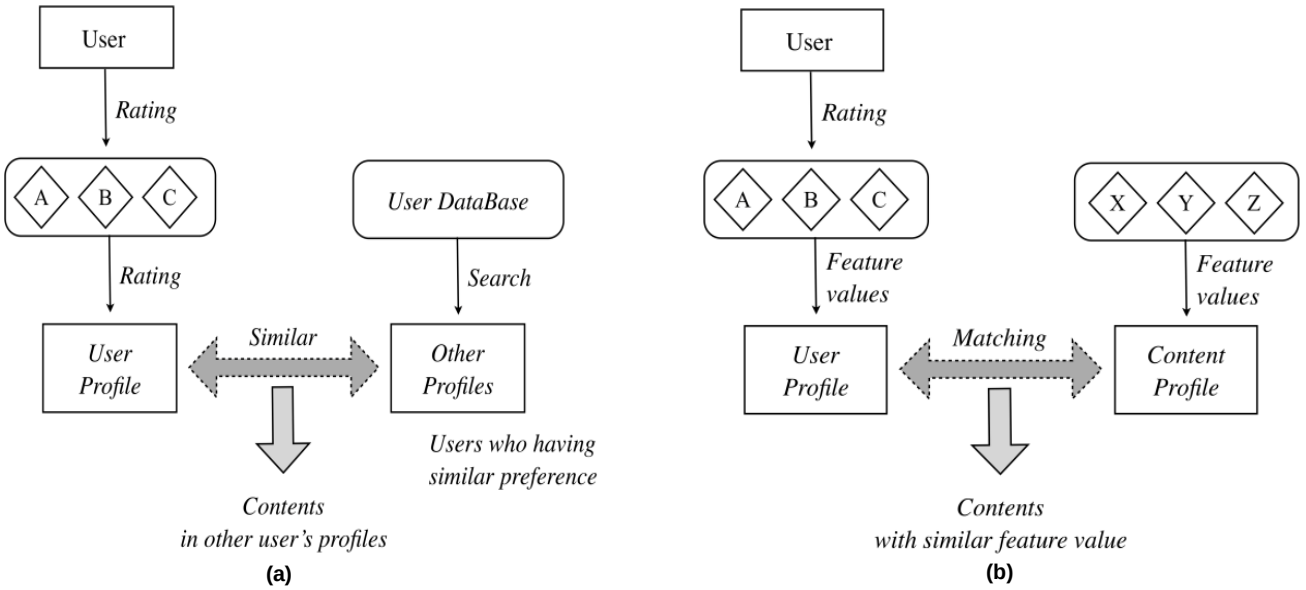

Figure 1. Procedures of methods. a, Collaborative filtering; b, content-based filtering.

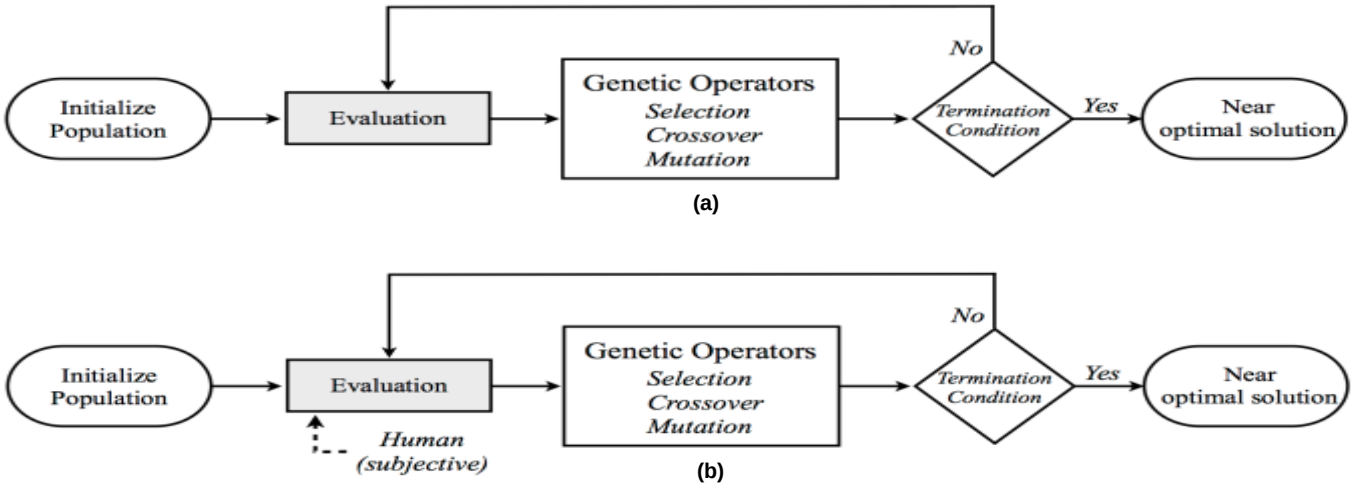

Figure 2. Structures of EC and IEC. a, Evolutionary computation (EC); b, interactive evolutionary computation (IEC).

gradually evolve towards the optimum by discovering, preserving and propagating the promising sub-solutions of each generation (Adomavicius and Tuzhilin, 2005; Goldberg and Holland, 1988).

Accordingly, IEC handles optimization problems in an interactive fashion (Takagi, 2001). Similar to GAs, IEC also uses genetic operators (that is, selection, crossover and mutation) to find the optimal solution. Unlike the generic EC, however, IEC employs the user to evaluate the fitness values of candidate solutions. In other words, the user's (subjective) evaluation replaces the objective function that determines the fitness values in the existing EC (Hao et al., 2009; Takagi, 2001).

Figure 2 shows the structures of both the EC and IEC. As shown in Figure 2, a user evaluates the fitness of various items. As a result, IEC can be applied to the proposed recommender system since it operates well with the user's evaluation and naturally utilizes the genetic operators to reach the optimal solution. Likewise, these IEC properties are ideal for the proposed recommender system since a user's preference may be traced even though his or her preference may change with a certain degree of uncertainty. Thus, in the proposed system, IEC is used to recognize a user's favorite items and to trace the user's preference by continual interactions between the user and the system.

Data grouping: *k*-means algorithm

In the proposed system, data grouping is employed to enhance the speed and the quality of the recommendations. For instance, the designed system creates groups out of all the alternative items according to the similarities of their structural features. After creating the groups, the system provides a recommendation based on the user's evaluation by looking for items within a particular

_k_-means clustering algorithm

Clustering (or grouping) N data points into k disjoint subset.

1. Randomly choose k points from N data points (to divide N data into k groups)

2. Set the k points to the centroid of each group

3. Apply pseudo-code below to data points of each group
While *no more changes in each group* do

For each item (I_k) in i-th group (C_i)
1. Calculate a distance (D_i) from I_k to C_i
$D_i = ||I_k – C_i||$

2. Calculate distances from the item (I_k) to the other centroids (C_j)
$D_j = ||I_k – C_j||$, for all $j(\neq i)$

3. Find a target group whose distance is the minimum
$D_t = \min\{D_j \mid \text{for all } j\ (\neq i)\}$

4. If $D_t < D_i$
Change the I_k's group from the original i-th group to the target t-th group

Figure 3. Procedure of the k-means clustering algorithm.

group. Thus, the system can rapidly offer new items to a user in a time-efficient manner. The system can save computational time in searching for proper items because only the candidate items within a corresponding group are considered.

In the proposed recommender system, the k-means clustering algorithm is employed owing to its simplicity and efficiency. The k-means clustering algorithm is commonly used to separate a data set into k groups, based on the features of each set.

It starts by choosing the centroids for k initial clusters, and then iteratively refines the given data set by checking the distances between an item and the centroids of other groups. The algorithm subsequently reassigns each item from the original group to another (new) group with the minimal distance between the item and its centroid. Note that if the distance between an item and its current centroid is the minimum distance possible, then no change occurs. These operations run repeatedly until the termination condition (that is, no more changes occur) is satisfied for each item and group. Figure 3 shows the procedure of the k-means clustering algorithm (David, 2003; Wagstaff et al., 2001).

PROPOSED APPROACH

The proposed recommender system is composed of three phases: preprocessing, user evaluation, and IEC. The system employs IEC to effectively recognize the user's preferences and promptly adapt to the user's behavior. In addition, it applies the content-based filtering method to generate the initial population for IEC and find similar items relevant to the user's evaluations. Note that since the proposed system was verified with music data, all the operational details will be explained here using music data examples.

Preprocessing phase

Prior to recommending items, a given data set should be initially

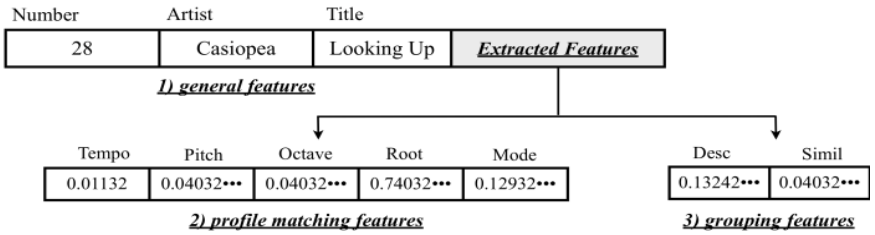

Figure 4. Example: Individual's composition.

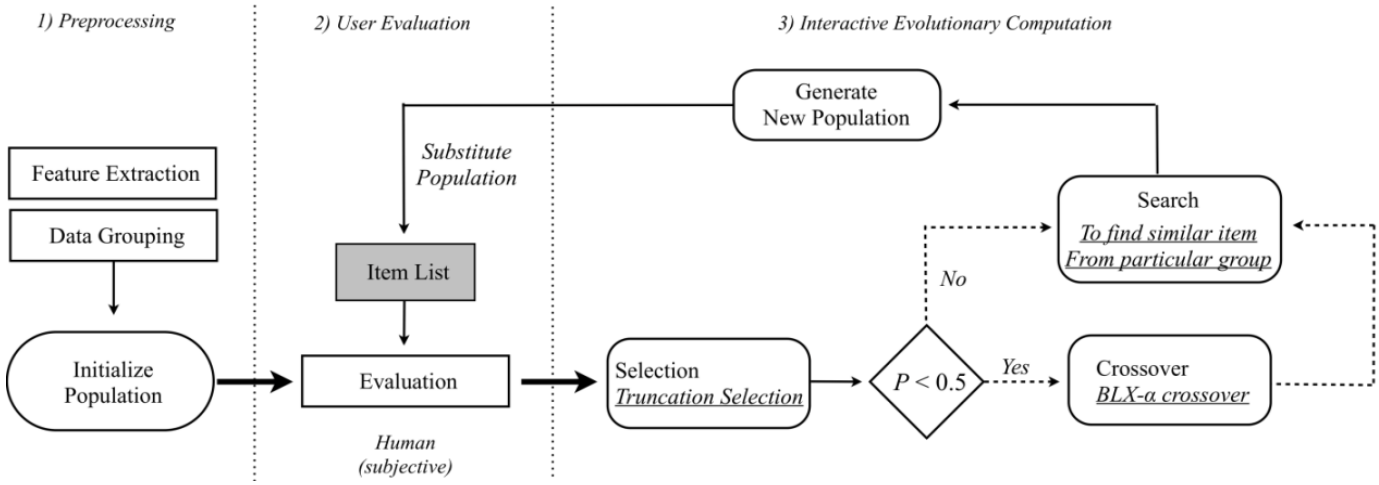

Figure 5. Structure of the proposed recommender system.

processed via feature extraction and data grouping. This step is essential to properly prepare the working environment of the proposed system. Then, the system operates on the items (for example, the music tracks) composed of the unique features extracted from the data set. Note that all of the music tracks were formatted as MP3 (Mpeg-1 Audio Layer 3) files.

First, each music track undergoes feature extraction using the software framework, CLAM (Clam, 2011; Amatriain et al., 2005), which outputs XML-formatted files containing the extracted features of each music data set. Next, the proposed system parses 20% of the output data to create the initial individuals for the IEC.

In the proposed system, seven types of the extracted features are considered: tempo, pitch, octave, root, mode, description (Desc), similarity group (Simil). Each feature consists of a set of real numbers that are normalized between 0 and 1. The features can be divided into three classes pertaining to their usage: general, profile matching and grouping. The first one, general feature, has several pieces of information for each music track (for example, number, artist name and song title). The profile matching feature is used to create each data set's profile and to find similar music tracks in the IEC phase. The last one, the grouping feature, provides the structural information of each music track, which can be useful for creating clusters or groups. Figure 4 shows an example of individual's composition with respect to the extracted features.

Once the feature extraction is completed, the proposed system creates a set of groups for the given data set using the *k*-means clustering algorithm. Based on the classified features of each individual, the system stores each group's information regarding a list of items within the same group. In addition, the system uses the stored group information to find similar items when making a recommendation in the IEC phase.

User evaluation phase

As mentioned earlier, the proposed recommender system allows users to evaluate the fitness value of each music track. A user can assign ratings to the items provided by the system, which reliably represents his or her subjective preference. After rating the given items, the evaluated data are stored in the system. Then, by referring to the user evaluation data, IEC is applied to the data set so that a population may evolve.

Interactive evolutionary computation (IEC) phase

The IEC phase is a very important component of making proper recommendations. As described earlier, IEC follows a procedure similar to the generic EC to reach an optimal solution. It also utilizes the genetic inheritance mechanisms. In this system, we consider two genetic operators - the selection and crossover operators - but we do not employ the mutation operator. By considering the individual's structure, the mutation operator has the potential to destroy the common pattern of the candidate solutions discovered by the evolutionary process thus far, which would be detrimental to the proposed system.

Figure 5 shows how IEC operates within the preprocessing and the user evaluation phases. IEC applies the selection and crossover

BLX-α crossover

1. Select two items $X(t)$ and $Y(t)$ at t-th generation
2. Create two offspring $X(t+1)$ and $Y(t+1)$ as follows:
   For i = 1 to n(*length of chromosome*) do
   (1) Calculate distance between $X_i(t)$ and $Y_i(t)$
   $D_i = | X_i(t) - Y_i(t) |$
   (2) Randomly generate *two real numbers* u and v from the following interval:
   $[\min(X_i(t), Y_i(t)) - (\alpha * D_i), \max(X_i(t), Y_i(t)) + (\alpha * D_i)]$
   $X_i(t+1) = u$
   $Y_i(t+1) = v$
   End do

Figure 6. Details of BLX-α crossover.

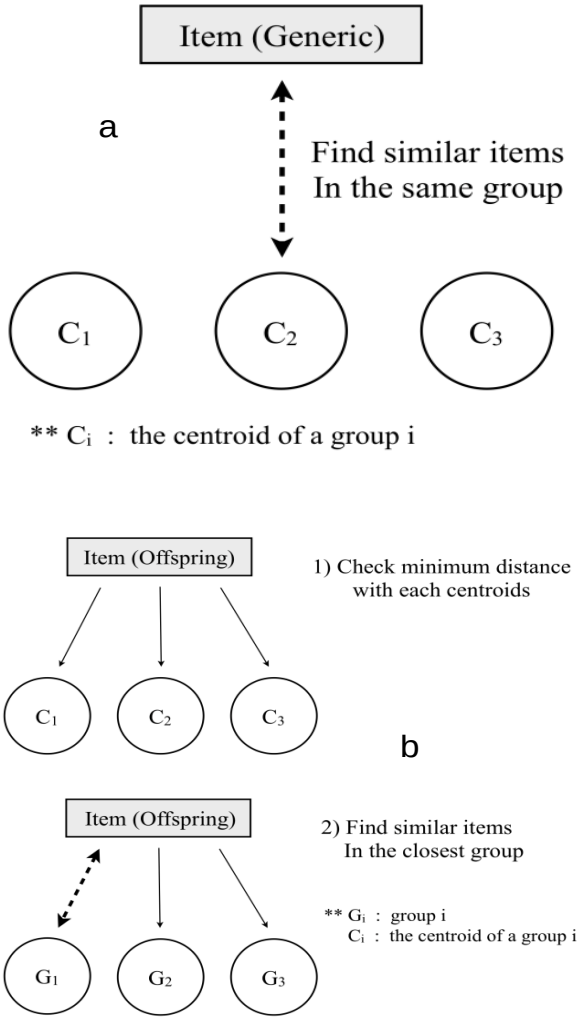

Figure 7. Different search steps for item types. a, Search step for generic case; b, search step for offspring case.

operators to the user evaluation data, performs a search to find similar items (using distance metrics and grouping methods) and generates a new population of the possible recommendation items.

Selection

In IEC, the genetic operators (selection and crossover) are fundamental components. Thus, the performance of our system is influenced by the choice of appropriate genetic operators, and such operators must be carefully chosen in order to design a better system. Although proportionate selection is typically used by many researchers, it is sensitive to selection noise. In this context, ordinal selection is preferable. Moreover, an item receiving a low rating by the user should be excluded from the recommendations; the system must recognize the user's preferences and respond. Consequently, truncation selection (a typical ordinal selection) is employed in the proposed system. This method has the strength of elitism, which imposes a high selection pressure to favor the top τ% candidate solutions (those with higher fitness values). The remaining items rated low by the user are subsequently discarded (Crow and Kimura, 1979; Thierens and Goldberg, 1994).

During this selection process, two pools for storing items are prepared: the selection pool and the remainder pool. The selected items are stored in the selection pool, while the other items are placed in the remainder pool. If the number of items in the selection pool through the truncation selection is odd, an item from the remainder pool is randomly chosen and added to the selection pool because the next step (crossover) requires an even number of items. Once the selection process is complete, half of items in the selection pool are applied to the crossover in a probabilistic manner.

Crossover

In this step, the proposed system applies the BLX-α crossover method (Fogel, 2005) to the selected items in the previous step. The crossover produces new offspring, which have properties inherited from their parents. In general, many different types of crossover operators exist for each individual's structure. For instance, 1-point and 2-point crossover operators, which are usually used in GAs, work well individuals that have a binary representation. In the proposed system, however, the extracted features of each individual are composed of a set of real numbers. Thus, the BLX-α crossover is employed as a suitable replacement. Through this step, the system produces two new individuals from each pair of processed items. Figure 6 shows the procedure of the BLX-α crossover method.

Search

The aim of this step is to find appropriate items for recommendation. To this end, the content-based filtering method is applied to search for similar items. As mentioned previously, each individual in the proposed system consists of its own music features, which are extracted by CLAM. The features play an important role in classifying the numerous resources and then finding similar items. To search for suitable candidates, the system calculates the similarities between the music features of the items resulting from previous steps and those of the remaining items.

As shown in Figure 7, the search step considers two cases - the generic case and offspring case. In the generic case, the system deals with the items chosen by the selection process. The input items or the source items have their own group information by which other items in the source item's group are considered as recommendation candidate items or target items. The system then computes the similarities between the source items and the target items.

Alternatively, in the offspring case, the system handles all the items in a different manner. Since the input items are produced by combining the two parent items via crossover, they do not have any

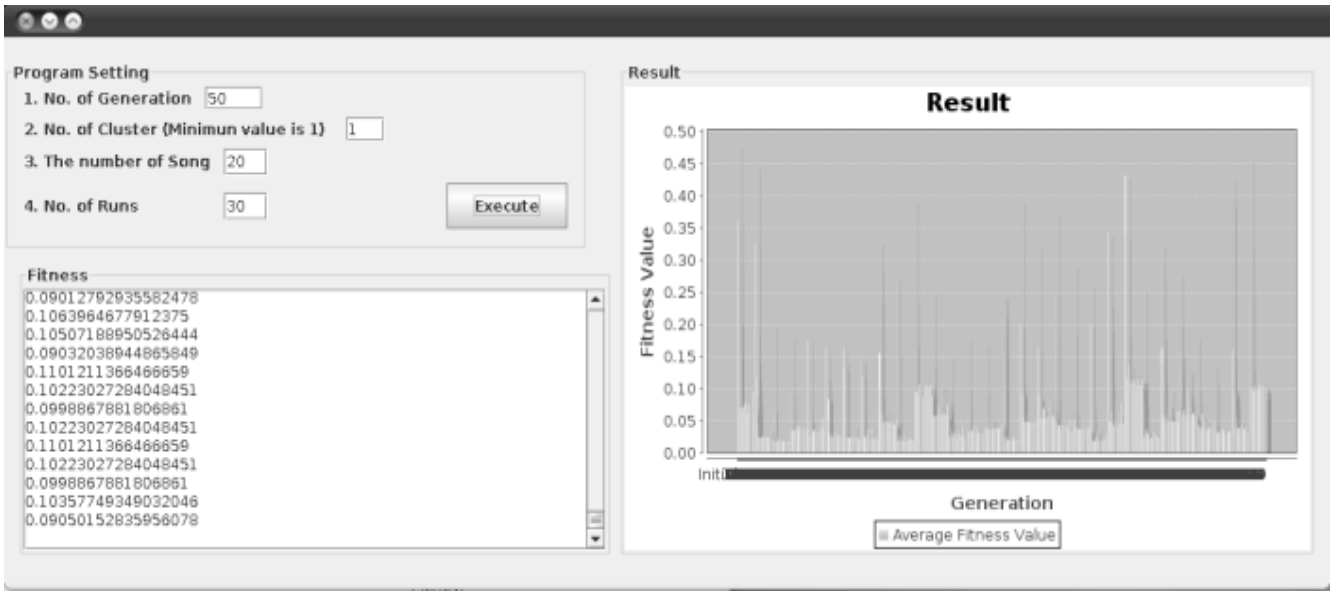

Figure 8. Test agent application.

group information. Accordingly, the proposed system first checks the distances between an item and the centroids of all the groups in order to find the closest group (that is, the group with the minimum distance). After finding a suitable group, the system performs the same procedure as in the generic case with regard to the found group.

In this study, the Euclidean distance is computed using a similarity metric:

$$distance(s,t) = \sqrt{\sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} (s_j - t_j)^2} \quad (1)$$

Where *distance (s,t)* is the Euclidean distance between two items, *s* is the source item the system already knows, and *t* is a candidate item for recommendation. Moreover, *n* is the number of unique features each items possesses and *m* indicates the length of each individual feature.

Note that before calculating the distance between the two items, the values of music features are normalized to lie within the range of 0 and 1.

EXPERIMENTAL RESULTS

In this section, we describe the experimental setup and present its results. In order to assess our proposed system, we used music tracks as the data set, which were composed of a total of 400 MP3 music files from five genres (pop, rock, hip-hop, jazz and Korean pop). Additionally, we extracted the unique features of each data set using CLAM, which were stored in our designed system. Note that we employed a test agent to perform our experiments automatically.

In this experiment, we investigated the quality and speed of making a recommendation in the proposed system. We also compared our designed system with an existing system that utilized the content-based filtering method.

Test framework

The goal of the proposed system was to produce a proper recommendation properly tailored to the user preferences. To examine the effectiveness of the system in achieving this goal, we designed a test agent to simulate the user in the experiment. It performs experiments and collects experimental results in an automatic way. In previous work (Kim et al., 2010), a website was built - whereby a user could evaluate items - as an attempt to establish a test framework. However, it was difficult to collect enough valuable data, even though the website had a better accessibility than test agents. Gathering the data from the real world is time consuming since users may not continuously participate in the experiment.

Therefore, we employed a test agent to evaluate the proposed system because it saves time in terms of collecting and analyzing the results. For instance, the test agent application selects a reference item randomly from the given dataset and calculates an average similarity value (that is, the distance) between a list produced by the proposed system and the reference item. Clearly, a smaller similarity value denotes a better recommendation. Figure 8 shows the test agent application developed in this study.

Test cases

In order to measure the performance of the proposed

Table 1. Description of test cases.

	Objective	Measure
Test case 1	Analyzing the proposed system's performance with data grouping	To minimize the following criteria: - average similarity value - execution time (*ms*)
Test case 2	Analyzing the performances between the proposed system and the existing system with the content-based filtering	- To minimize the average similarity value
Test case 3	Analyzing the performances between the proposed system and the existing system with the content-based filtering when new data sets are introduced	- Minimize the average similarity value - Keep the trend of the average similarity value after the specific epochs at which new data sets are added

system, we designed three test cases as shown in Table 1. Each experiment was conducted to verify whether or not its own objective was achieved by the proposed system. In the second and third test cases, we performed comparative experiments with another system, which used the existing recommendation method (that is, content-based filtering). In all of the test cases, we were interested in the average similarity values. The test agent application traced the changes of the average similarity values with the given evaluation measures for each test case.

Experimental result

Test case 1

This study proposes a new type of recommender system using a new method, which combines IEC and content-based filtering. In addition, this novel system employs data grouping to enhance the time efficiency. In this sense, this test case was designed to assess the recommendation quality (that is, the average similarity value) and time consumption (that is, the execution time). The first test case had the following conditions:

1) 200 runs
2) Generate 50 lists for each run
3) Each list contains 10 items
4) Number of groups: 1 to 10

Under the given conditions, when the system set up the number of groups as 1, the system did not apply any data grouping. In other words, the system regarded all of the items as target items for the search step, as mentioned earlier. When an average similarity value for each run was measured, the last 5 out of 50 lists were chosen to calculate the similarity value. This process allows the system to gather reliable results with respect to the reference item (that is, user's preferences). Furthermore, by analyzing the changes in the last 5 lists, we were able to measure whether or not the designed system had responded to the given conditions.

Figures 9 and 10 show the average similarity values and the average execution times when changing the number of groups. As shown in Figure 9, the average similarity value increased with the number of groups. On the other hand, the execution time decreased significantly when two or three groups were used. Table 2 shows the results for the first test case

As shown in Table 2, the difference between the average similarity values from the first row and the other rows was sufficiently small. Moreover, the proposed system decreased the execution time by 62% (on average) when the data grouping was applied (see the second and third rows). Consequently, the results show that the proposed system can guarantee an acceptable recommendation quality and better time efficiency by incorporating data grouping.

Test case 2

In this test case, we compared the performance between our proposed system and an existing system that employs a memory-based content-based filtering method (Yoshii et al., 2008). For the second test case, we assigned the following experimental conditions:

1) 200 runs
2) Generate 50 lists for each run
3) Each list contains 10 items
4) Number of groups (in the proposed system): 2

We used two groups for this experimental case since better results were obtained in the first test case when two groups were used.

Figure 11 compares the average similarity values of the two systems after each system had processed 50 lists (500 items). As shown in Figure 11, both systems initially had high similarity values since each system creates an initial list by choosing items randomly. However, the

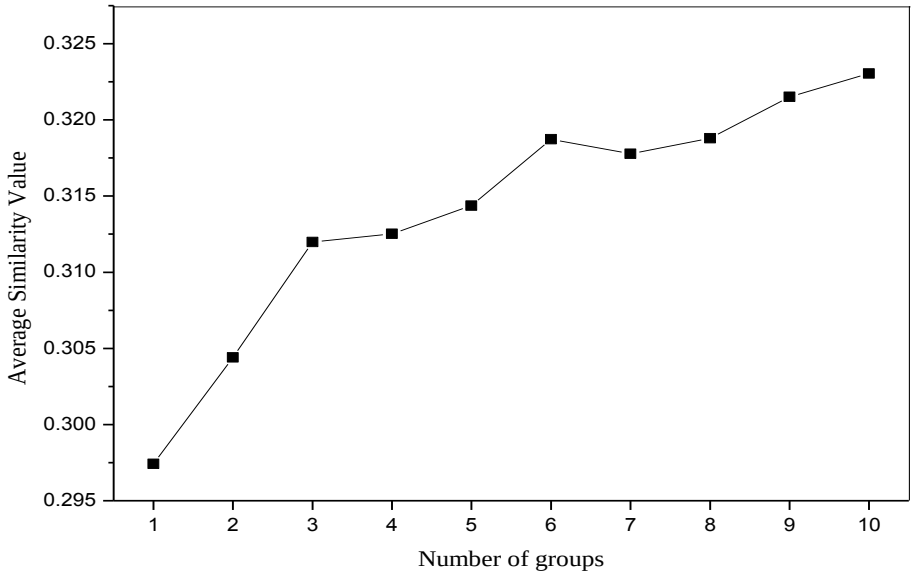

Figure 9. Test case 1: Average similarity values with respect to the number of groups.

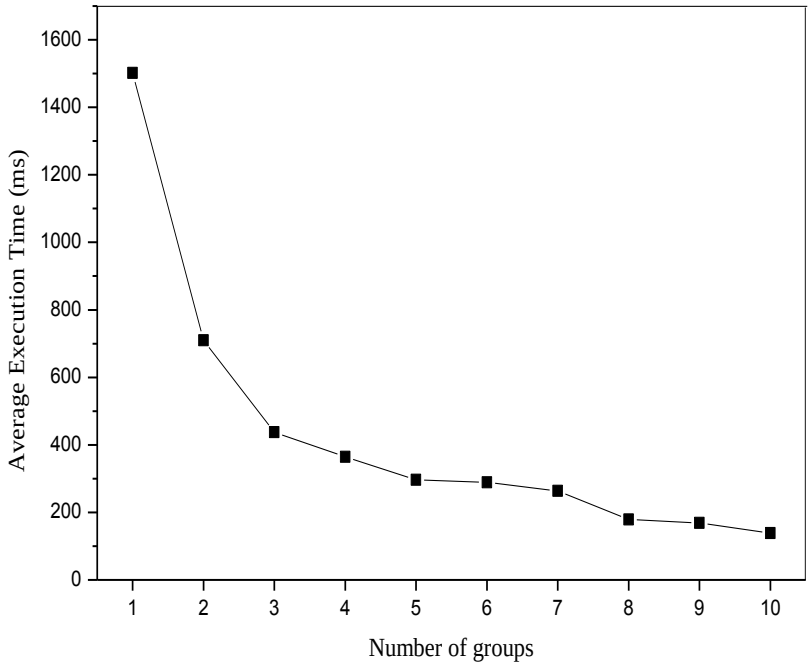

Figure 10. Test case 1: Average of execution times with respect to the number of groups.

Table 2. Results of Test case 1.

Number of group	Average similarity value	Average execution time
1	0.297422813	4739
2	0.30440591 (2%)	2725 (53%)
3	0.311981003 (5%)	1677 (71%)

The digit in parenthesis indicates the amount of improvement compared to the reference that runs with one group.

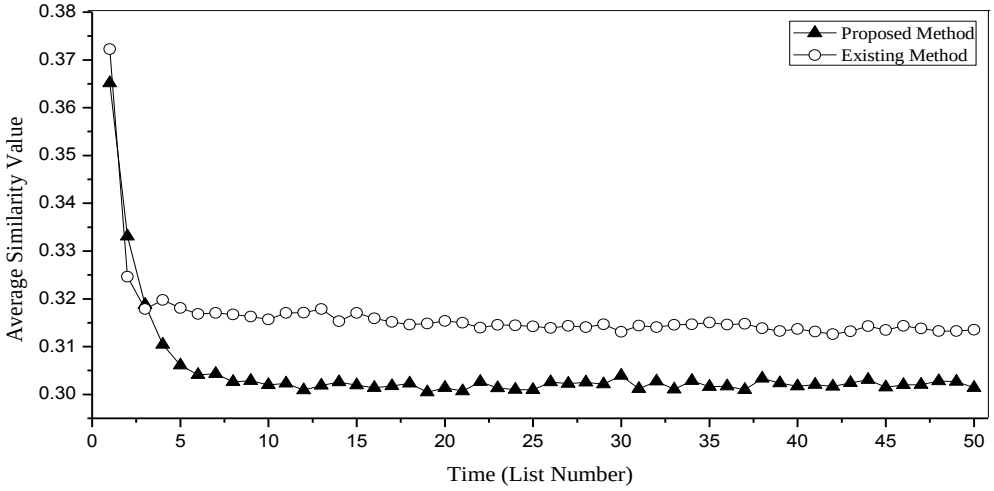

Figure 11. Test case 2: Average similarity values of our proposed system and the existing system.

Table 3. The changing epochs and the number of items at each epoch.

Changing epoch	Number of Item
0	200 (Initial items)
50	300 (+100 new items)
100	400 (+100 new items)

designed system produced a list with similarity values that were lower than those from the existing method. In addition, the proposed system returned promising solutions after the fifth generation; its fast convergence speed (that is, a matter of time complexity) was verified.

Test case 3

In the third test case, we examined how the system responded when a new data set was inserted. In other words, we tested how efficiently the proposed system dealt with newly added items, which is the most critical limitation (that is, sparse problem) of the existing recommendation systems as mentioned earlier. The experimental conditions for the third test case are follows

1) 200 runs
2) Generate 50 lists for each run
3) Each list contains 10 items
4) Number of groups (in the proposed system): 2

For this test case, the same experimental conditions were used as in the second test case because we intended to investigate the changes or trends of the average similarity value by introducing new items.

Table 3 shows the specific epochs at which the new items are added and the total number of items. As shown in Table 3, in the third test case, 100 new items were added at the 50th and 100th epochs, respectively.

Figure 12 shows the average similarity values after adding new data sets to the system. The results support that after adding new items, the change in the average similarity values of the proposed system tended to be similar to the past values (that is, before adding the new data). Moreover, the proposed system produced better recommendations than the existing system. Consequently, the proposed system was able to quickly respond to environmental changes, such as the introduction of new items; thus, it may help resolve the sparse problem and become a promising alternative to the existing recommendation methods.

CONCLUSION

In this paper, we have presented a new recommender system that was able to accurately recognize the trend of a user's preference and adaptively provide an appropriate recommendation in a time efficient manner. To this end, the proposed system combined the strengths of content-based filtering and IEC. In addition, the system employed a data grouping scheme (that is, the *k*-means algorithm).

The proposed system initially extracts the unique features of each item using the content-based filtering. As inputs to IEC, the individuals are composed of the extracted features of the data set. The system then requests the user to evaluate the fitness of each item. Next, IEC operates on the evaluated items to discover the most appropriate items for recommendation while the data grouping technique is applied to search for the candidate items more rapidly.

Experiments conducted with music data showed that the average similarity value increased as the number of

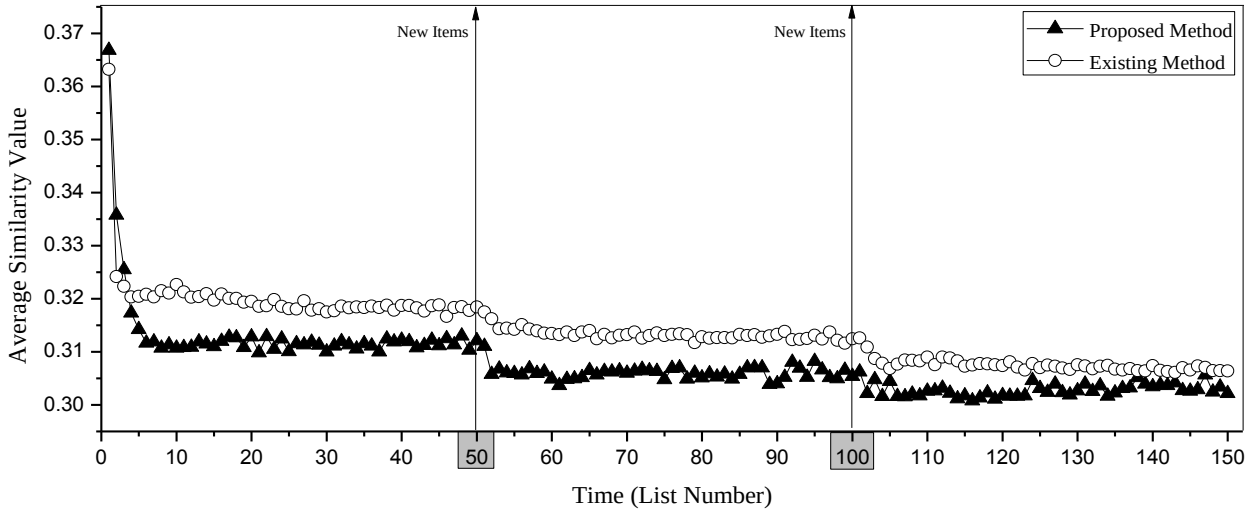

Figure 12. Test case 3: Average similarity value of the two systems for various numbers of data sets.

groups increased; however, the improvement became minor when more than three groups were used. The execution time, on the other hand, exponentially improved when three groups were used, suggesting that the proposed system can recommend appropriate items in a time efficient manner. Moreover, we performed comparative experiments with the existing system that used the content-based filtering. The experimental results verified that the proposed system is able to make better recommendations than the content-based filtering system. Finally, we considered an additional situation whereby new data sets were inserted into the system at specific times. The results from this test confirmed the effectiveness of employing the proposed system to resolve the sparse problem.

We believe that the proposed framework is a vital step to designing a new recommender system that determines a user's preferences and responds to his or her behavior.

ACKNOWLEDGMENT

This research was supported by MKE, Korea under ITRC NIPA-2012-(C-1090-1221-0008).

REFERENCES

Adomavicius G, Tuzhilin A (2005).Toward the next generation of recommendersystems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowledge Data Eng., 17: 734-749.

Amatriain X, Massaguer J, Garcia D, Mosquera I (2005).The clam annotator: A cross-platform audio descriptors editing tool. In: International Symposium on Music Information Retrieval, pp. 426-427.

Balabanovi'c M, ShohamY (1997). Fab: Content-based, collaborative recommendation. Commun. ACM, 40(3): 66-72.

Burke R (2007) Hybrid web recommender systems. Lecture Notes in Computer Science, 4321: 377-408.

Clam (2011). C++ library for audio and music.URL http://clam-project.org.

Cohen WW, Fan W (2000). Web-collaborative filtering: recommending music by crawlingthe web. Comput. Networks, 33(1): 685-698.

Crow JF, Kimura M (1979).Efficiency of truncation selection. In: Proceedings of National Academy of Sciences of the United States of America, pp. 396-399.

David M (2003). An Example Inference Task:Clustering, Cambridge University Press. Information Theory, Inference and Learning Algorithm. pp. 284-292.

Fogel DB (2005). Evolutionary Computation:Toward a New Philosophy of Machine Intelligence, 3rd Edition. Wiley-IEEE Press, pp. 142-158.

Goldberg DE, Holland JH (1988).Genetic algorithms and machine learning. Machine Learning, 3: 95-99.

Goldberg DE, Nichols D, Oki BM, Terry D (1992).Using collaborative filtering to weave an information tapestry. Commun. ACM, 35: 61-70.

Hao GS, Gong DW, Yuan J, Yan YR, Yan JR (2009).User's attention knowledge learning in interactive evolutionary computation. In: Proceedings of Control and Decision Conference, pp. 4270-4275.

Kim HT, Kim E, Lee JH, Ahn CW (2010). A recommender system based on genetic algorithm for music data. In: Proceedings Comput. Eng. Technol., 6: 414-417.

Linden G, Smith B, York J (2003).Amazon.com recommendations item-to-item collaborative filtering. IEEE Internet Computing, 7(1): 76-80

Logan B (2002). Content-based playlist generation: Exploratory experiments. In: Proceedings of International Society for Music Information Retrieval, pp. 295-296.

Marques V, Reis C, Tenreiro Machado J (2010). Interactive Evolutionary Computation in music. In: Proceedings of Systems Man and Cybernetics (SMC). pp. 3501-3507.

Pazzani MJ (1999).A framework for collaborative, content-based and demographic filtering. Artificial Intell. Rev., 13(5-6): 394-408.

Resnick P, Varian HR (1997).Recommender system. Commun. ACM, 40(3): 56-58.

Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994). Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of Computer Supported Cooperative Work, pp. 175-186.

Salton G, Buckley C (1990). Improving Retrieval Performance by Relevance Feedback. J. Am. Soc. for Inf. Sci., 41(4): 288-297.

Sarwar B, Karypis G, Konstan J, Riedl J (2000).Analysis of recommendation algorithms for e-commerce. In: Proceedings of the 2nd ACM Conference on Electronic Commerce. pp. 158-167.

Schafer J, Konstan J, Riedl J (1999).Recommender systems in e-commerce. In: Proceedings of the 1st ACM Conference on Electronic Commerce, pp. 158-166.

Takagi H (2001). Interactive evolutionary computation: fusion of the capabilities of ec optimization and human evaluation. Proc. IEEE, 89(9): 1275-1296.
Terveen L, Hill W (2001). Beyond Recommender Systems: Helping People Help Each Other. HCI in the New Millennium, Addison-Wesley.
Thierens D, Goldberg DE (1994). Elitist recombination: an integrated selection recombination GA. In: Proceedings of the First IEEE Conference on Evolutionary Computation, 1: 508-512.
Wagstaff K, Cardie C, Rogers S, Schroedl S (2001).Constrained k-means clustering with background knowledge. In: Proceedings of International Conference on Machine Learning, pp. 577-584.
Yoshii K, Goto M, Komatani K, Ogata T, Okuno H (2008).An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model. IEEE Trans. Audio, Speech, and Language Processing, 16(2): 435-447.

.