# Privacy Aware Obfuscation Middleware for Mobile Jukebox Recommender Services

Ahmed M. Elmisery, Dmitri Botvich

Waterford Institute of Technology – WIT,
Telecommunications Software and Systems Group – TSSG, Co. Waterford, Ireland
{ael-misery, dbotvich}@tssg.org

***Abstract.*** Mobile Jukebox is a service offered by mobile operators to their clients, such that subscribers can buy or download anywhere, anytime full-length music tracks over the 3G Mobile networks. Unlike some music download services, the subscribers can reuse the selected tracks on their music players or computers. As the amount of online music grows rapidly, Jukebox providers employ automatic recommender service as an important tool for music listeners to find music that they will appreciate. On one hand, Jukebox recommender service recommend music based on users' musical tastes and listening habits which reduces the browsing time for searching new songs and album releases. On the other hand, users care about the privacy of their preferences and individuals' behaviors regarding the usage of recommender service. This work presents our efforts to design an agent based middle-ware that enables the end-user to use Jukebox recommender services without revealing his sensitive profile information to that service or any third party involved in this process. Our solution relies on a distributed multi-agent architecture involving local agents running on the end-user mobile phone and two stage obfuscation process used to conceal the local profiles of end-users with similar preferences. The first stage is done locally at the end user side but the second stage is done at remote nodes that can be donated by multiple non-colluding end users that requested the recommendations or third parties mash-up service. All the communications between participants are done through anonymised network to hide their network identity. In this paper, we also provide a mobile jukebox network scenario and experimentation results.

**Keywords:** privacy, clustering, mobile jukebox, recommender service, multi-agent.

## 1 Introduction

Being music a very important thing in people's life, music applications are present in every computer and over many mobile devices. For such reason, mobile operators offer Mobile Jukebox as a moderate price service to their clients, such that subscribers can buy or download full-length music tracks over 3G Mobile networks. Unlike some music download services, the user can reuse the selected tracks on their music players or portable music devices.

According to [1] the more services appear in future, the more demand for personalization services will be to fight against information overload and find information relevant to each user. Recommender services can be seen as a suitable solution to these problems as they customize the offered services according to unique and individual needs of each user. The Jukebox providers employ recommender services to reduce browsing time for music listeners as the amount of online music grows rapidly. Jukebox recommender service becomes an increasingly important tool for music listeners to easily find new songs or playlists that they will appreciate. Examples of available Jukebox services are Apple iTunes® and Last.fm®. Apple iTunes automatically generates a playlist of songs from the user's library which is similar to the selected songs, While Last.fm builds a detailed profile for each user's musical taste by recording details of the songs the user listens to either from internet radio stations or user's computer or portable music devices. This information is transferred to Last.fm's database via music player and the user's profile data can be displayed on his profile page.

Jukebox recommenders commonly use collaborative filtering (CF) techniques to recommend music based on the listening behaviors of other music listeners. The Jukebox recommender harness the "wisdom of the crowds" to recommend music. Even though they generate good recommendations there are still some problems like the cold-start problem, as a recommender needs a significant amount of data before it can generate appropriate recommendations. However the acquisition, storage and application of sensitive personal information cause privacy concerns for users. There are many things having an impact on the perception of privacy for users like what kind of information is collected, how the information is used and the degree of accessibility of the information by others.

The authors in [1] have done an empirical research concerning privacy preferences and individuals' behaviors regarding personalization in music recommender systems. They found out that information about the purpose of the disclosure, recipients of the information, and the degree of the information involved and the benefits users expect to gain from disclosing personal information are the main factors influencing disclosure behavior. Based on their questionnaire in [2], participants were more willing to disclose music preferences than their personality. Participants considered information about personality traits more personal and more sensitive information than preferences for music genres. Participants expressed worries about not knowing how their information will be used in the system and who gets access to their personal information. The sensitivity of information affects on the disclosure decision. The questionnaire also shows that some participants even consider what benefits they will gain from disclosing the information. Participants can be divided into two groups based on their disclosure behavior, depending on whether they want to disclose anonymously or including identity information. One important factor have an impact on people's disclosure behavior is the security and privacy standards taken by the Jukebox providers.

In this work, we proceed with our approach presented in [3-7] to build AMPR (i.e. acronym for agent based middleware for private recommendations) that allows end-

users to receive useful recommendations without disclosing their real preferences to the service. In the following section we will describe some properties for AMPR:

1. AMPR is running as a multi-agent based middleware to support different types of clients either thin or thick. Moreover, this architecture enables smooth integration with wide range of existing recommender services
2. AMPR preserves the aggregates in the obfuscated profiles to maximize their utility in order to attain acceptable recommendations accuracy, which facilitate AMPR to work with different state-of art filtering algorithms. Extra overhead in computation and communication to be added in the recommendation process due to the two stage obfuscation process
3. AMPR employs two stage obfuscation process to conceal the user's preferences in his profile. The real user profile doesn't leave his mobile phone until it is properly desensitized and it is maintained encrypted with private password that is known only to the user. If the user doesn't accept to be tracked by the recommender service using his network identity, AMPR hides his identity by routing the submission of his locally obfuscated profile through relaying nodes in an anonymous communication network before sending it to the recommender service.

In the rest of this paper we will generically refer to songs and playlists as Items. In section 2, we describe some related work. Section 3 introduces recommender system for mobile jukebox service scenario landing AMPR. Section 4 introduces our proposed solution. Section 5 describes the recommendation strategy used in our PCRS. Section 6 presents some experiments and results based on our proposed solution. Section 7 includes conclusions and future work.

## 2 Related Work

The majority of the literature addresses the problem of privacy for recommender services based on collaborative filtering technique, Due to it is a potential source of leakage of private information shared by the users as shown in [8]. In [9] it is proposed a theoretical framework to preserve the privacy of customers and the commercial interests of merchants. Their system is a hybrid recommender that uses secure two party protocols and public key infrastructure to achieve the desired goals. In [10, 11] it is proposed a privacy preserving approach based on peer to peer techniques using users' communities, where the community will have a aggregate user profile representing the group as whole and not individual users. Personal information will be encrypted and the communication will be between individual users and not servers. Thus, the recommendations will be generated at client side. In [12, 13] it is suggest another method for privacy preserving on centralized recommender systems by adding uncertainty to the data by using a randomized perturbation technique while attempting to make sure that necessary statistical aggregates such as mean don't get disturbed much. Hence, the server has no knowledge about true values of individual rating profiles for each user. They demonstrate that this method does not decrease essentially the obtained accuracy of the results. Recent research work [14, 15] pointed

out that these techniques don't provide levels of privacy as it was previously thought. In [15] it is pointed out that arbitrary randomization is not safe because it is easy to breach the privacy protection it offers. They proposed a random matrix based spectral filtering techniques to recover the original data from perturbed data. Their experiments revealed that in many cases random perturbation techniques preserve very little privacy. Similar limitations were detailed in [14].

## 3 Recommender System for Mobile Jukebox Service - Scenario

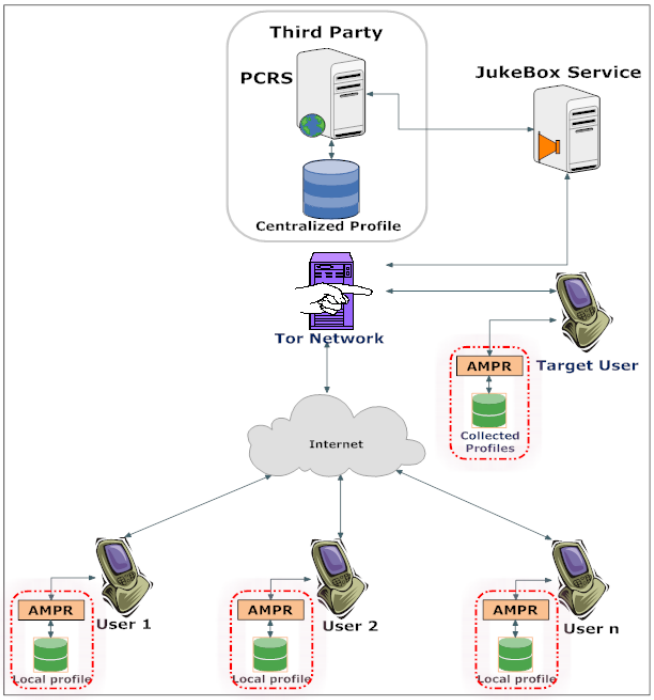


**Fig. 1.** Jukebox Service for Mobile Users with Third Party Private Recommender Service

We consider the scenario where a private centralized recommender service (PCRS) is implemented on an external third party server and end-users give information about their preferences to that server in order to receive music recommendations. The user preferences stored in his profile in the form of ratings or votes for different item, such that items are rated explicitly or implicitly on a scale from 1 to 5. An item with rating of 1 indicates that the user dislikes it while a rating of 5 means that the user likes it. PCRS collects and stores different users' preferences in order to generate useful recommendations.

In this scenario there are two possible ways for user's discloser: through his personal preferences included in his profile [16] or through the user's network address (IP). AMPR employs two principles to eliminate these two disclosure channels, respectively. The obfuscation agents perturb user's preferences for different items in his profile and the synchronize agent hides the user's network identity by routing the communication with other participants through relaying nodes in Tor [17] anonymous network. The main challenge for synchronize agent is to tune up Tor and optimize its performance while maintaining the user anonymity.

We don't assume the server to be completely malicious. This is a realistic assumption because the service provider needs to accomplish some business goals and increase its revenues. In our framework, we will use the mobile phone storage to store the user profile. On the other hand, the Jukebox service maintains a centralized rating database that is used by the PCRS; Figure (1) shows the architecture of our approach. Additionally, we alleviate the user's identity problems stated above by using anonymous pseudonyms identities for users.

## 4 Proposed Solution

In the next sub-sections, we will present our proposed middleware for protecting the privacy of users' profiles

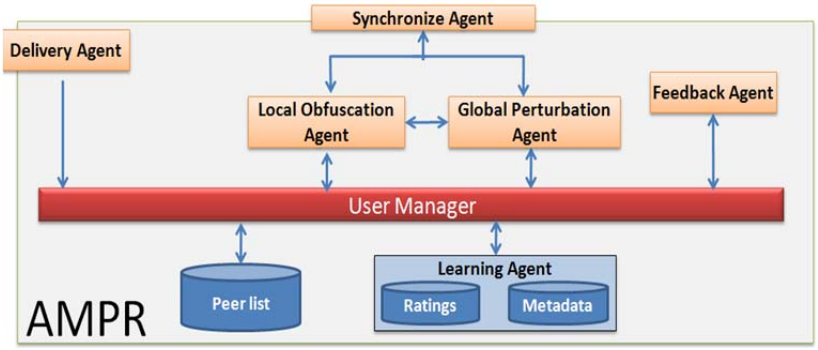


**Fig. 2.** AMPR Components

Figure (2) demonstrates AMPR components that are running in the mobile phone at the user side. As shown, AMPR consists of different co-operative agents, A Learning agent captures user preferences about items explicitly or implicitly to build a rating table and meta-data table. The local obfuscation agent implements *CTA* obfuscation algorithm to achieve user privacy while sharing the data with other users or the system. The global perturbation agent is only invoked if the user is acting as a target user in recommendation process; it executes *EVS*-algorithm on the collected profiles finally. The synchronize agent is responsible for selecting the best suitable routing paths in the anonymised network to enhance its performance.

The recommendation process based on the two stage obfuscation process in our framework can be summarized as following:

1. The learning agent collects user's preferences about different items which represent a local profile. The local profile is stored in two databases, the first one is the rating database that contains (item_id, rating) and the other one is the metadata database that contains the feature vector for each item (item_id, feature1, feature2, feature3). The feature vector can include genre, author, album, decade, vocalness, singer, instruments, number of reproductions and so on.
2. The target user broadcast a message to other users near him to request recommendations for specific genre or category of items. Individual users who decide to respond to that request use their local obfuscation agent to obfuscate a part of their local profiles that match query. The group members submit their locally obfuscated profiles to the requester using an anonymised network like TOR to hide their network identities. Enhancing the performance of communication through Tor is discussed in the next sub-section. If the size of group formation less than a specific value, the target user contacts the PCRS directly to gets recommendation from the centralized profiles stored in it.
3. In order to hide items identifiers and meta-data from the requester and the PCRS. The manger agent at each participant side use locality-sensitive hashing (LSH) [18] to hash these values. One interesting property for LSH is that similar items will be hashed to the same value with high probability. PCRS is still be able to perform computations on the hashed items using appropriate distance metrics like hamming distance or dice coefficient.
4. After the target user receives all the participants' profiles (group profile), he/she incites his global perturbation agent to perturb the collected profiles. Then he can interact with PCRS by acting as an end-user has group profile as his own profile. The target user submits his/her group profile through an anonymised network to PCRS in order to attain recommendations.
5. PCRS performs its filtering techniques on the group profile which in turn return a list of items that are correlated with that profile. This list is encrypted with a private key provided by target-user and it is sent back on the reverse path to the target user that in turn decrypts and publishes it anonymously to the other users that participated in the recommendation process.

### 4.1 Enhancing The Anonymized Network (Tor) Performance

Tor [17] is an anonymity network based on the original onion routing design but with several modifications in terms of security, efficiency, and deployability. The Tor network includes a small set of trusted authoritative directory servers responsible for aggregating and distributing signed information about known routers in the network. Tor clients periodically fetch the directory information from directory mirrors in order to learn information about other servers in the network.

Tor network suffers from serious performance degradation because of its random path selection algorithms that use self-reported bandwidth values only, which might

select with high probability a router with low bandwidth because it is sensitive to loads and changing network conditions. The synchronize agent seeks to enhance the performance by partitioning the Tor network into classes of high or low bandwidth Tor routers to better understand the relationships between different classes of routers /potential paths. Paths drawn from the class of high-bandwidth routers can provide better performance. Paths can be reserved for participating in specific recommendation requests based on a user's priorities or preferences. Therefore, inspired from the work in [19] we have implemented a simple parameterized path selection algorithm (*PPS*) that allow the synchronize agent to enhance the path selection in the Tor network with two priorities and it can be easily extended to support priorities larger than two. The synchronize agent can create circuits in advance to reduce the waiting time then measure the path throughput $P_T$ before using each path. *PPS* consists of the following steps:

1. The user input minimum path throughput $P_T$, and circuit throughput $C_T$ to the synchronize agent.
2. Based on Tor authoritative directory servers, the algorithm start partitioning the class of high-bandwidth routers into a set of overlapping clusters (based on geographical location, platform, bandwidth, uptime, last update, number of connections and self-reported bandwidth estimate) using the algorithm on [20].
3. It builds a pool of Tor nodes whose bandwidth $\geq P_T$ from each cluster. In order to decrease the delay in circuit creation, the synchronize agent can select overlapping routers between clusters
4. Then, it randomly builds circuits passing through these clusters. Then measure each circuit throughput, and select the first circuit that achieve bandwidth $\geq C_T$.
5. The synchronize agent then negotiates session keys with each router in the circuit. The exit router is responsible for establishing the connection from the Tor network to the client's intended destination
6. The synchronize agent records the previously used Tor nodes and exclude them from future circuit building clusters.

### 4.2 Proposed obfuscation Algorithms

In the next subsections, we present two different algorithms used by the obfuscation agents in AMPR to obfuscate the user profile in a way that secure user's preferences in PCRS with minimum loss of accuracy.

**Local Obfuscation using Clustering Transformation Algorithm (*CTA*).**
We proposed a novel algorithm for obfuscating the user profile before sharing it with other users. *CTA* designed especially for the sparse data problem we have here. *CTA* partitions the user profile into smaller clusters and then pre-process each cluster such that the distances inside the same cluster will maintained in its obfuscated version. We use local learning analysis (*LLA*) clustering method proposed in [21] to partition the dataset. After complete the partitioning, we embed each cluster into a random dimension space so the sensitive ratings will be protected. Then the resulting cluster

will be rotated randomly. In such a way, *CTA* obfuscates the data inside user profile while preserving the distances between the data points to provide high accurate results when performing recommendations. The algorithm consists of the following steps:

1. The user ratings is stored in his mobile phone as dataset $D$ consists of $c$ rows, where each row is a sequence of $X$ attributes where $X = x_1\ x_2\ x_3 \dots \dots \dots\ x_n$.
2. The dataset $D$ is portioned vertically into $D_1\ D_2\ D_2 \dots \dots D_m$ subsets of length $L$, if $n/L$ is not perfectly divisible then *CTA* randomly selects attributes already assigned to any subset and joins them to the attributes of the incomplete subsets.
3. Cluster each subset $\forall_{j=1}^{m} D_j$ Using *LLA* algorithm, that result in $K$ clusters $D_j = C_{j1}, C_{j2}, C_{j3} \dots . C_{jk}$for each subset. So every point in the original dataset $D$ falls exactly in one cluster. The aim of this step is to increase the privacy level of the transformation process and make reconstruction attacks difficult.
4. *CTA* generates two sets for each cluster in the subset $D_j$ these are $H_{ji}$and $O_{ji}$.Where $H_{ji}$is the set of points with highest values for field function and $O_{ji}$is the rest of points in $C_{ji}$.For each point $x_{1i} \in H_{ji}$construct a weighted graph $\Gamma_i$that contains its k-nearest neighbours in $O_{ji}$, each edge $e \in \Gamma_i$ has a weight equals to the influence function of that point $f_{Gauss}^{b_{ji}}(x_{1i})$.
5. Estimate the geodesic distances by Computing the shortest distance between each two points in graph $\Gamma_i$using Dijkstra or Floyd algorithm and then build a distance matrix $D_{\Gamma_i} = \{ f_{Gauss}^{b_i}(x_i)\}$.
6. Based on $D_{\Gamma_i}$,we find a *d*-dim embedding space $C'_{ji}$ using classical *MDS* [22] as follows
    - Calculate the matrix of squared distances $S = D_{\Gamma_i}^2$ and the centering matrix $H = 1 - 1/N\ ee^T$
    - The characteristic vectors are chosen to minimize $E = \left\|\tau(D_{\Gamma_i}) - \tau(D_d)\right\|_{L^2}$, where $\tau(D_d)$ is the distance matrix for the *d-dim* embedding space, and converts distances to inner products $\tau = -HSH/2$.
7. For each cluster $\forall_{j=1}^{m}\forall_{i=1}^{k} C'_{ji}$, *CTA* randomly select two attributes $x_a$and $x_b$ to perform rotation perturbation on selected attributes $R(x_a, x_b)$ using transformation matrix $M_j^{\theta}$ setup by the user for each cluster using range of angles defined in advance by the user.
8. Repeating steps 4-7 for all clusters in $\forall_{j=1}^{m} D_j$ to get the obfuscated portion $D'_j$. Finally, the obfuscated dataset is obtained by $D' = \cup_{j=1}^{n} D'_j$.

**Global Perturbation using Enhanced Value-Substitution (EVS) Algorithm.**

After executing the local obfuscation process the global perturbation phase starts. The key idea for *EVS* is based on the work in [23] that uses Hilbert curve to maintain the association between different dimensions. In this subsection, we extend this idea as following, we also use Hilbert curve to map m-dimensional profile to 1-dimensional profile then *EVS* discovers the distribution of that1-dimensional profile. Finally, we perform perturbation based on that distribution in such a way to preserve the profile range. The steps for *EVS* algorithm consists of the following steps:

1. We denote the collected m-dimensional user profiles as dataset *D* of *c* rows, where each row is a sequence of *m* dimensions $\mathrm{A} = \mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4 \ldots \ldots, \mathrm{A}_\mathrm{m}$.
2. *EVS* divides the *m*-dimensional profile into grids of order *k* (where *k* is user defined value) as shown in [23, 24]. For order *k*, the range for each dimension divided into $2^k$ intervals.
3. For each dimension $\forall_{i=1}^{m} A_i$ of the collected profile :
    - Compute the *k*-order Hilbert value for each data point $\forall_{\mathrm{x}=1}^{\mathrm{c}} \mathrm{a}_{\mathrm{ix}}$. This value represents the index of the corresponding interval where it falls in.
    - *EVS* sort the Hilbert values from smallest to biggest, then use the step length (a user defined parameter) to measure whether any two values are near from each other or not. If these values are near, they are placed in the same partition $\forall_{v=1}^{k} k_{iv}$.

    These two steps iterates for all *m*-dimensions. The final result from these steps is partitions for each dimension denoted as $\forall_{i=1}^{m} \forall_{v=1}^{k} C_{iv}$
4. EVS constructs a *N* shared nearest neighbour sets $\mathrm{S}_\mathrm{r}$ where $\mathrm{r} = 1 \ldots . \mathrm{N}$ as in [25] from different partitions with a new modified similarity function as following, two partitions in different dimensions $C_{iv}, C_{i+1v}$ form a shared nearest neighbour set $\mathrm{S}_\mathrm{r}$ if they share *k*-number of common elements such that $S_r = C_{iv} \cup C_{i+1v}$
5. For each newly created set $\mathrm{S}_\mathrm{r}$ , EVS calculates the interquartile range. Then, for each point $\mathrm{a}_\mathrm{i} \in \mathrm{S}_\mathrm{r}$ generate a uniform distributed random point $n$ in that range that can substitutes $\mathrm{a}_\mathrm{i}$.
6. Finally, the new set $D' = \cup_{r=1}^{N} \mathrm{S}_\mathrm{r}$ is sent to PCRS

## 5 Recommendation Strategy

PCRS employ online mode filtering algorithms to make predictions on the ratings of a particular user by collecting preference information from other users. The collected profiles (group profile) represented as $m * n$ user item matrix which contains a collection of numerical obfuscated ratings of *M* users on *N* items. After that, the neighbourhood formation at PCRS is done by calculating the similarity between users in the user-item matrix. Users similar to the target user using some proximity metric will form a proximity based neighbourhood with him [12]. This neighbourhood will utilize later for predication step. The prediction on rating of user i for item *K* is given by a weighted average [26] of users whose ratings are similar to the target user.

$$\mathrm{P}_{\mathrm{ik}} = \overline{\mathrm{v}_\mathrm{i}} + \frac{\sum_{\mathrm{j} \in \mathrm{U}_\mathrm{k}} \mathrm{s}(\mathrm{u}_\mathrm{i}, \mathrm{u}_\mathrm{j})(\mathrm{v}_{\mathrm{jk}} - \overline{\mathrm{v}_\mathrm{j}})}{\sum_{\mathrm{j} \in \mathrm{U}_\mathrm{k}} |\mathrm{s}(\mathrm{u}_\mathrm{i}, \mathrm{u}_\mathrm{j})|}$$

Where $\mathrm{U}_\mathrm{k} = \{\mathrm{i} \in \mathrm{U} | \mathrm{v}_{\mathrm{ik}} \neq \emptyset\}$ is the set of users who have rated the $\mathrm{k} - \mathrm{th}$ item. $\overline{\mathrm{v}_\mathrm{j}}$ is the mean of all ratings made by user i. The weights of average $\mathrm{s}(\mathrm{u}_\mathrm{i}, \mathrm{u}_\mathrm{j})$ are the similarity between user $\mathrm{u}_\mathrm{i}$ and $\mathrm{u}_\mathrm{j}$ such as the Pearson correlation coefficient or Euclidean distance. We represent the user as a vector consists of *n* features slots, one for each item. These slots contain user's ratings for different items or $\emptyset$. The similarity

between users' vectors is calculated as the cosine of the angle formed between them as following:

$$s(u_i, u_j) = \frac{\sum_{k=1}^{n} v_{ik}\, v_{jk}}{\sqrt{v_{i1}^2 + \cdots + v_{in}^2}\sqrt{v_{j1}^2 + \cdots + v_{jn}^2}}$$

Finally, the recommendation process is to produce a predicted rating based on the neighbourhood for a list of items that have not been rated by the user, these items have a high potential of interest (predicated with high positive rating) to the user. The resulting items list can be further constrained based on marketing or Qos rules.

## 6 Experiments

The proposed algorithms are implemented in C++. We used message passing interface (MPI) for a distributed memory implementation of *EVS* algorithm to mimic a distributed reliable network of peers. We evaluated the proposed algorithms from two different aspects: privacy achieved and accuracy of results. The experiments presented here were conducted using the Movielens dataset provided by Grouplens [27]. The dataset contains users' ratings on movies using discrete value between 1 and 5. The data in our experiments consists of 100.000 ratings for 1.682 items by 943 users. The experiments involve dividing the data set into a training set and testing set. The training set is obfuscated then used as a database for the PCRS. Each rating record in the testing set is divided into rated items and unrated items. The rated items are presented to the PCRS for making predication for the unrated items. To evaluate the accuracy of generated predications, we used the mean absolute error (*MAE*) metric proposed in [28]. *MAE* measures the predication verity between the predicated ratings and the real ratings, so smaller *MAE* means better recommendations provided by PCRS. To measure the privacy or distortion level achieved using our algorithms, we use variation of information metric *VI* [29] to estimate data error. Where, the higher *VI* means the larger distortion between the obfuscated and original dataset, which means higher privacy level.

To evaluate the accuracy of *CTA* algorithm with respect to different number of dimensions in user profile, we control *d-dim* parameters of *CTA* to vary number of dimensions during the evaluation. Figure (3) shows the performance of recommendations of locally obfuscated data, as shown the accuracy of recommendations based on obfuscated data is little bit low when *d-dim* is low. But at a certain number of dimensions (500), the accuracy of recommendations of obfuscated data is nearly equal to the accuracy obtained using original data.

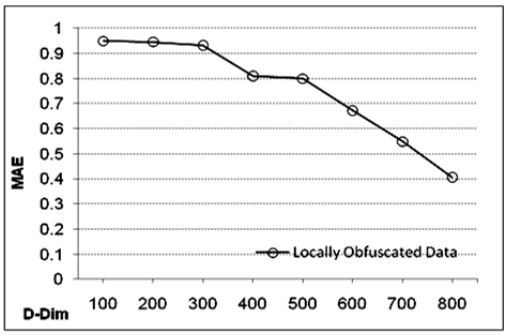


**Fig. 3.** Accuracy of recommendations for obfuscated dataset using *CTA*

In the second experiment performed on *CTA* algorithm, we examine the effect of *d-dim* on *VI* values. As shown in Figure (4), *VI* values decrease with respect to the increase in *d-dim* values in user profile. *d-dim* is the key element for privacy level where smaller *d-dim* value, the higher *VI* values (privacy level) of *CTA*. However, clearly the highest privacy is at *d-dim*=100. There is a noticeable drop of *VI* values when we change *d-dim* from 300 to 600.*d-dim* value 400 is considered as a critical point for the privacy.Note that rotation transformation adds extra privacy layer to the data and in the same time maintains the distance between data points to enable PCRS to build accurate recommendation models.

In the first experiment performed on *EVS* algorithm, we measured the relation between different Hilbert curve parameters (order and step length) on the accuracy and privacy levels attained. We map the locally obfuscated dataset to Hilbert values using order 3, 6 and 9. We gradually increased the step length from 10 to 80. Figure (5) shows the accuracy of recommendations based on different step length and curve order. We can see that as the order increases, the obfuscated data can offer better predictions for the ratings. This is because as the order has higher value, the granularity of the Hilbert curve becomes finer. So, the mapped values can preserve the data distribution of the original dataset. On the other hand, selecting larger step length increases *MAE* values as large partitions are formed with higher range to generate random values from it, such that these random values substitute real values in the dataset.

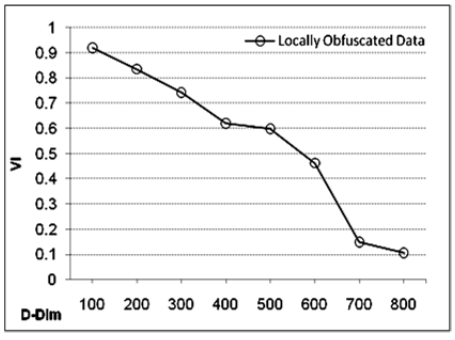


**Fig. 4.** Privacy levels for the obfucated dataset using *CTA*

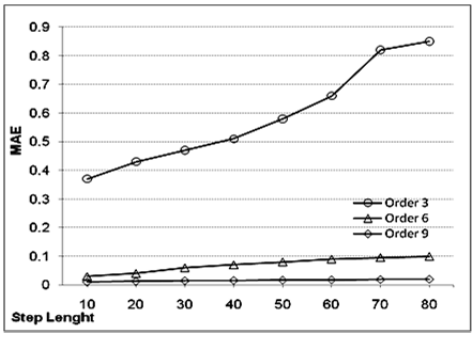


**Fig. 5.** Accuracy level for different step length and orders for *EVS*

As shown in Figure (6), when the order increases a smaller range is calculated within each partition which introduces less substituted values compared with lower orders that attain higher *VI* values. The reason for this is larger order divides the *m*-dimensional profile into more grids, which makes Hilbert curve better reflects the data distribution. Also, we can see that for the same Hilbert curve order the *VI* values are generally the same for different step length except for order 3, in which *VI* values has a sharp increase when step length grows from 50 to 60.The effect of increasing step length on *VI* values is more sensible in lower curve orders as fewer girds are formed and the increase of step length covers more portions of them, which will introduce a higher range to generate random values from it. So the target user should select *EVS* parameters in such a way to achieve a trade off between privacy and accuracy.

Finally, we measured the overall performance of *PPS* algorithm in terms of the enhancement achieved in uploading time for the collected profiles. Figure (7) illustrates the cumulative distribution function (*CDF*) of time uploading the collected profiles of 331,25 bytes from the target user under the proposed *PPS* algorithm. The term *PPS* = $C_T \leq$34KB/s refers to executing *PPS* with a circuit throughput equal to 34KB/s. Table (1) gives the mean, median for execution of *PPS* with different circuit throughput values. Our analysis of Figure (7) and Table (1) lead us to the following observations; the performance of Tor's default path selection algorithm is unacceptable for responsive recommender services. The largest uploading time for the profiles is 182.61s; also our *PPS* algorithm significantly improves path selection performance.

**Table 1.** Uploading Time for Different $C_T$ values

| | default | PPS= $\mathbf{C_T} \leq$10KB/s | PPS= $\mathbf{C_T} \leq$20KB/s | PPS= $\mathbf{C_T} \leq$30KB/s | PPS= $\mathbf{C_T} \leq$34KB/s |
|---|---|---|---|---|---|
| Mean | 30.38 | 25.34 | 19.64 | 12.54 | 8.41 |
| Median | 33.56 | 17.35 | 15.18 | 10.73 | 9.59 |

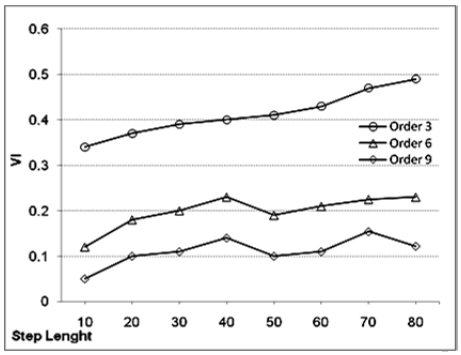


**Fig. 6.** Privacy level for different step length and orders for *EVS*

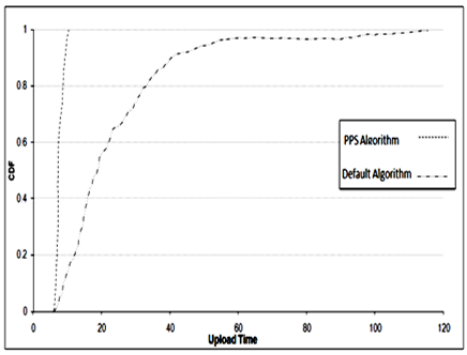


**Fig. 7.** Uploading time using PPS Algorithm

## 7 Conclusions and Future Work

In this paper, we presented our ongoing work on building an agent based middleware for private recommendation services. We gave a brief overview of the recommendations process with application to Jukebox music recommendations. Also we presented the novel algorithms that provide the users with complete control of the privacy of their profiles using two stage obfuscation process. We tested The performance of these proposed algorithms on real dataset. The experiential results show that preserving users' privacy for Jukebox recommender service is possible. In particular mean average error can be reduced with proper tuning of the algorithms' parameters for large number of users. We realized that there are many challenges in building an agent based middleware scenario. This allow us to move forward in building an integrated system while studying issues such as a dynamic data release at a later stage and deferring certain issues such as virtualized schema and auditing to future research agenda. We need to perform extensive experiments in other real data set from the UCI repository and compare the performance with other techniques. Also we need to consider different data partitioning techniques as well as identify potential threats and add some protocols to ensure the privacy of the data against those threats.

**Acknowledgments :** This work has received support from the Higher Education Authority in Ireland under the PRTLI Cycle 4 programme, in the FutureComm project (Serving Society: Management of Future Communications Networks and Services).

## References

1. Perik, E., de Ruyter, B., Markopoulos, P., Eggen, B.: The Sensitivities of User Profile Information in Music Recommender Systems. Proceedings of Private, Security, Trust (2004)
2. Perik, E., de Ruyter, B., Markopoulos, P.: Privacy & Personalization: Preliminary Results of an Empirical Study of Disclosure Behavior. Proceedings of PEP, Edinburgh, UK. (2005)
3. Elmisery, A., Botvich, D.: An Agent Based Middleware for Privacy Aware Recommender Systems in IPTV Networks. 3rd International Conference on Intelligent Decision Technologies Springer Verlag, University of Piraeus, Greece (2011)
4. Elmisery, A., Botvich, D.: Agent Based Middleware for Private Data Mashup in IPTV Recommender Services. 16th IEEE International Workshop on Computer Aided Modeling, Analysis and Design of Communication Links and Networks. IEEE, Kyoto, Japan (2011)
5. Elmisery, A., Botvich, D.: Agent Based Middleware for Maintaining User Privacy in IPTV Recommender Services. 3rd International ICST Conference on Security and Privacy in Mobile Information and Communication Systems. ICST, Aalborg, Denmark (2011)
6. Elmisery, A., Botvich, D.: Privacy Aware Recommender Service for IPTV Networks. 5th FTRA/IEEE International Conference on Multimedia and Ubiquitous Engineering. IEEE, Crete, Greece (2011)
7. Elmisery, A., Botvich, D.: Private Recommendation Service For IPTV System. 12th IFIP/IEEE International Symposium on Integrated Network Management. IEEE, Dublin, Ireland (2011)

8. McSherry, F., Mironov, I.: Differentially private recommender systems: building privacy into the net. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, Paris, France (2009) 627-636
9. Esma, A.: Experimental Demonstration of a Hybrid Privacy-Preserving Recommender System. In: Gilles, B., Jose, M.F., Flavien Serge Mani, O., Zbigniew, R. (eds.), Vol. 0 (2008) 161-170
10. Canny, J.: Collaborative filtering with privacy via factor analysis. Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, Tampere, Finland (2002) 238-245
11. Canny, J.: Collaborative Filtering with Privacy. Proceedings of the 2002 IEEE Symposium on Security and Privacy. IEEE Computer Society (2002) 45
12. Polat, H., Du, W.: Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques. Proceedings of the Third IEEE International Conference on Data Mining. IEEE Computer Society (2003) 625
13. Polat, H., Du, W.: SVD-based collaborative filtering with privacy. Proceedings of the 2005 ACM symposium on Applied computing. ACM, Santa Fe, New Mexico (2005) 791-795
14. Huang, Z., Du, W., Chen, B.: Deriving private information from randomized data. Proceedings of the 2005 ACM SIGMOD international conference on Management of data. ACM, Baltimore, Maryland (2005) 37-48
15. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the Privacy Preserving Properties of Random Data Perturbation Techniques. Proceedings of the Third IEEE International Conference on Data Mining. IEEE Computer Society (2003) 99
16. Parameswaran, R., Blough, D.M.: Privacy preserving data obfuscation for inherently clustered data. Int. J. Inf. Comput. Secur. **2** (2008) 4-26
17. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. Proceedings of the 13th conference on USENIX Security Symposium - Volume 13. USENIX Association, San Diego, CA (2004) 21-21
18. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM, Dallas, Texas, United States (1998) 604-613
19. Pingley, A., Yu, W., Zhang, N., Fu, X., Zhao, W.: CAP: A Context-Aware Privacy Protection System for Location-Based Services. Proceedings of the 2009 29th IEEE International Conference on Distributed Computing Systems. IEEE Computer Society (2009) 49-57
20. Fellows, M.R., Guo, J., Komusiewicz, C., Niedermeier, R., Uhlmann, J.: Graph-Based Data Clustering with Overlaps. Proceedings of the 15th Annual International Conference on Computing and Combinatorics. Springer-Verlag, Niagara Falls, NY (2009) 516-526
21. Elmisery, A., Huaiguo, F.: Privacy Preserving Distributed Learning Clustering Of HealthCare Data Using Cryptography Protocols. 34th IEEE Annual International Computer Software and Applications Workshops, Seoul, South Korea (2010)
22. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications (Springer Series in Statistics). Springer (2005)
23. Ghinita, G., Kalnis, P., Skiadopoulos, S.: PRIVE: anonymous location-based queries in distributed mobile systems. Proceedings of the 16th international conference on World Wide Web. ACM, Banff, Alberta, Canada (2007) 371-380
24. Reaz, A., Raouf, B.: A Scalable Peer-to-peer Protocol Enabling Efficient and Flexible Search. (2010)
25. Jarvis, R.A., Patrick, E.A.: Clustering Using a Similarity Measure Based on Shared Near Neighbors. IEEE Trans. Comput. **22** (1973) 1025-1034

26. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: GroupLens: Applying Collaborative Filtering to {Usenet} News. Communications of the ACM **40** (1997) 77-87
27. Lam, S., Herlocker, J.: MovieLens Data Sets. Department of Computer Science and Engineering at the University of Minnesota. (2006)
28. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22** (2004) 5-53
29. Kingsford, C.: Information Theory Notes. (2009)