A RECOMMENDATION SYSTEM COMBINING CONTEXT-AWARENESS AND USER PROFILING IN MOBILE ENVIRONMENT

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SERKAN ULUCAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2005

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. İsmet Erkmen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. İsmet Erkmen
Co-Supervisor

Prof. Dr. Aydan Erkmen
Supervisor

Examining Committee Members

Assoc. Prof. Dr. Gözde Bozdağı Akar (METU, EE) _____

Prof. Dr. Aydan Erkmen (METU, EE) _____

Prof. Dr. İsmet Erkmen (METU, EE) _____

Assoc. Prof. Dr. Veysi İşler (METU, CENG) _____

Asst. Prof. Dr. Afşar Saranlı (METU, EE) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Serkan ULUCAN :

Signature :

ABSTRACT

A RECOMMENDATION SYSTEM COMBINING
CONTEXT-AWARENESS AND USER PROFILING
IN MOBILE ENVIRONMENT

Ulucan, Serkan

M.S., Department of Electrical and Electronics Engineering

Supervisor : Prof. Dr. Aydan Erkmen

Co-Supervisor: Prof. Dr. İsmet Erkmen

December 2005, 122 pages

Up to now various recommendation systems have been proposed for web based applications such as e-commerce and information retrieval where a large amount of product or information is available. Basically, the task of the recommendation systems in those applications, for example the e-commerce, is to find and recommend the most relevant items to users/customers. In this domain, the most prominent approaches are "collaborative filtering" and "content-based filtering". Sometimes these approaches are called as "user profiling" as well.

In this work, a context-aware recommendation system is proposed for mobile environment, which also can be considered as an extension of those recommendation systems proposed for web-based information retrieval and e-commerce applications. In the web-based information retrieval and e-commerce applications, for example in an online book store (e-commerce), the users' actions are independent of their instant context (location, time…etc). But as for mobile environment, the users' actions are

strictly dependent on their instant context. These dependencies give raise to need of filtering items/actions with respect to the users' instant context.

In this thesis, an approach coupling approaches from two different domains, one is the "mobile environment" and other is the "web", is proposed. Hence, it will be possible to separate whole approach into two phases: "context-aware prediction" and "user profiling". In the first phase, combination of two methods called "fuzzy c-means clustering" and "learning automata" will be used to predict the mobile user's motions in context space beforehand. This provides elimination of a large amount of items placed in the context space. In the second phase, hierarchical fuzzy clustering for users profiling will be used to determine the best recommendation among the remaining items.

Keywords: Context-Awareness, User Profiling, Learning Automata, Fuzzy C-means Clustering, Hierarchical Fuzzy Clustering

ÖZ

MOBİL ORTAMDA
KULLANICI PROFİLLEME VE DURUM MUHAKEME TABANLI
AKILLI ÖNERİ SİSTEMİ

Ulucan, Serkan
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Aydan Erkmen
Ortak Tez Yöneticisi: Prof. Dr. İsmet Erkmen

Aralık 2005, 122 sayfa

Şimdiye kadar çok fazla bilginin ve ürünün bulunduğu web tabanlı bilgi edinme ve e-ticaret gibi uygulamalar için çeşitli yönlendirme sistemleri önerildi. Temel olarak bu uygulamalardaki, örneğin e-ticarette, yönlendirme işi kullanıcıyla/müşteriyle en çok ilgili olan nesnelerin bulunması ve önerilmesi işidir. Bu alanda en çok göze çarpan yaklaşımlar "içerik tabanlı filtreleme" ve "iş birliği ile yapılan filtrelemedir". Bazen bu yaklaşımlar "kullanıcı profillerini" çıkarma olarak ta adlandırılır.

Bu çalışmada mobil ortam için, web tabanlı bilgi edinme ve e-ticaret uygulamalarında kullanılan yönlendirme sistemlerinin genişletilmesi olarak ta düşünülebilecek bir durum algılayan yönlendirme sistemi önerilmiştir. Web tabanlı bilgi edinme ve e-ticaret uygulamalarında, örneğin kitap satışı yapılan bir sitede (e-ticaret), kullanıcıların hareketleri anlık durumlarından (yer, zaman… vs.) bağımsızdır. Fakat mobil ortam için kullanıcı hareketleri anlık durumlarına sıkı olarak bağımlıdır. Bu bağlılıklar

nesnelerin/hareketlerin, kullanıcıların anlık durumlarına göre filtrelenme gereğini ortaya çıkarır.

Bu tezde, biri “mobil ortam” diğeri “web” olan iki farklı alandan gelen yaklaşımları birleştiren bir yaklaşım önerdik. Bu şekilde, bütün yaklaşımı iki faz olarak düşünmek mümkün: “durum-algılayan tahminleme” ve “kullanıcı profili oluşturma”. İlk fazda kullanıcı hareketlerinin durum uzayı içinde önceden tahmini için “fuzzy c-means” ve “learning automaton” olarak adlandırılan iki metodun kombinasyonu kullanılıyor. Bu durum uzayı içinde yer alan birçok nesnenin filtrelenmesini sağlar. İkinci fazda ise kalan nesneler arasından en iyi yönlendirmeyi belirlemek için “hierarchical fuzzy clustering” metodu kullanılacaktır.

Anahtar Kelimeler: Durum Algılama, Kullanıcı Profili Oluşturma, Learning Automaton, Fuzzy C-means Clustering, Hierarchical Fuzzy Clustering

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Prof. Dr. Aydan ERKMEN and Prof. Dr. İsmet ERKMEN for their supervision, valuable guidance and helpful suggestions.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Problem Statement and Motivation

Recommendation as a social process plays an important role in many applications for consumers, because it is overly expensive for every consumer to learn about all possible alternatives independently. Depending on the specific application setting, a consumer might be a buyer (e.g., in online shopping), or an information seeker (e.g., in information retrieval). In these scenarios, an assistant knowing the interest and goals of the consumers can help them by providing personalized suggestions. Therefore, these sorts of assisting consumers give rise to emerging technologies as automated software applications that act as an assistant, that we will refer from here on as "recommendation systems".

Generally, recommendation systems have been used in two domain; e-commerce and information retrieval. Both of them are web based and interact with user via internet and personal computers. The e-commerce domain generally refers to the act of conducting business online that includes buying and selling products with digital cash and via Electronic Data Interchange. Here the task of the recommendation can be defined as finding and offering products that the buyer is interested in. Information retrieval (IR) or document retrieval is the semantic manipulation of the textual information so that it can be easily found even repeatedly. In other words, it is the process of determining the relevant documents from a collection of documents, based on a query presented by the user. The task of the recommendation includes classification, selection and suggestion of documents to the user in question depending on different user queries or user interests.

Digital libraries and search engines on the World Wide Web are prominent examples of such information retrieval.

The first requirement to develop a recommendation system for a specific domain is tractability of user behaviour, since recommendation systems use historical data about the user behaviour to generate their suggestions. Consequently, almost all recommendation systems generated so far have been based on web technologies, since the medium is highly suitable to tracking users and there exist various ways of tracking user behaviours whereby interacting with the user via personal computers. For example, in Amazon.com, which is a commercial book store, customers/users search for books through a standard electronic shop web-based interface [43]. As the user shops, by navigating through the site's web pages, Amozon.com presents opportunistic recommendations of books that the user might be interested in. These recommendations are based on the previous navigation pattern and buying habits of the user. Another one is Citeseer digital library of which content consist of academic publications [44]. When the user clicks on hyperlinks to papers then glances at papers' abstract, the system presents other papers that the user might be interested in, also gives them a score representing relevance degree. Above the first example refers to the e-commerce and the second one refers to the information retrieval. And there exist many others similar to these ones.

Nowadays, mobile devices allow users to access mobile internet services and run applications at any time and any place. Advance in wireless network technology and continuously increasing number of users of mobile terminal makes this a possible channel for offering personalised services to mobile users and gives space to the rapid development of the mobile services. As a result, we think about possibility of usage of the recommendation system which helps the mobile users in their usual activities (such as shopping, eating, entertainment…etc) in mobile environment.

1.2 Objective and Goals

In this thesis work, we propose a recommendation system which helps mobile users in their usual activities such as shopping, eating, entertainment…etc. in the mobile domain. The most distinctive features of the recommendation system proposed for the mobile domain are "context-awareness", "prediction of user's motion" and "user profiling". Context-awareness means user's context, which are contextual information such location and time, is used in generation of recommendation. In the mobile environment, since user's context conveys implicit information related to user behaviour, this implicit information should be exploited through generation of recommendations. Accordingly, our first aim is to design a method for context-awareness. The second feature of the recommendation system is prediction of the user's motion. We consider all users move in a context space. If it is possible to predict the user's motion in the context space, the recommendation system can generate the recommendations previously, thus, the recommendations become more useful for the user. The second aim of this work is to design a method which predicts the user's motion. The third feature of the recommendation system is user profiling. There exist heterogonous items (actions) in the mobile domain. Therefore, probably each user will have interest profile on those items. Since user interest profile is information showing the interest degrees of the user to each item, such information should be exploited through generation of recommendations also. Then, the third aim is to design a method which extracts the user profiles. Finally, we will combine these three methods in a single recommendation system which uses features of these methods in recommendation.

As for mobile domain, the recommendation system is not popular as in web applications since tractability of users is a big problem and development cost is very expensive. And any database is not readily available for our case. In this thesis work, we will develop a simulator to test the proposed recommendation system. In simulation, synthetic data produced by simulator will be used. A detailed discussion on usage of simulated or synthesized dataset is given in [41]. The author in that reference work reports that

synthesized data sets may be required in some limited cases, but only as early steps while gathering data sets or constructing complete systems.

1.3 Problem Characteristic and Methodology

The most distinctive features of the mobile environment are mobility and ubiquity. These features add two new dimensions on top of similar system in the web domain which are “location” and “time”. From now on, these two dimensions will be called as context. Thus, the question is that “How the context can be extracted through in the filtering process of the recommendation system”. In the literature, authors refer this problem as “Context Awareness”.

1.3.1 Context Awareness

The term context means that the general condition in which an event or action takes places. In [1] context is defined as any information (such as location, time, day… etc) that can be used to characterize the situation of an event.

The ability to reason from context is crucial to human communication. It allows a large amount of implicit information to be conveyed with a small amount of explicit description. If the same kind of ability can be provided to software agents, then agents, even with a small amount of built-in knowledge, can adjust their behaviours according to the available implicit information in the environment. From the design point of view, a context-aware software agent is simply a way to model a context aware application based on the agent approaches. Similar to the traditional context aware application, the context aware agent is designed to make use of context providing task-relevant information and services to users.

As seen from works in the literature, the context aware agents have not been used in the web based recommendation systems. In other words, the web based recommendation

agents are not designed to handle context. Following example will clarify this case. On an online book store, consider a user searching a book and a recommender system helps the user to make his decision. Here, prominent attributes of context are as follows: he is searching a book “at home” “at 10:00 pm”. Let the same user search another book on the same online book store “at work” “at 12:00 am”. In both case there exist two different contexts. Now the question is that: “does the context have any influence on the user’s behaviour?” Obviously, the context does not influence the user. Consequently, most of the recommendation systems designed for the web domain do not take care of the user’s context. But in the mobile domain (environment) context is a determinative factor on the users’ action. Consider another user having habits such that he usually goes out at Friday night and prefers listening live music, and he usually goes to brunch in Sunday mornings. It is possible to say that he has interests such as live music and brunch, and any recommendation agent working as the web based recommendation agents can infer these user’s interests. But, it will not be useful to recommend him a new place performing live music in Sunday morning and a good place for brunch at Friday night. This illustrative scenario clarifies the importance of the context awareness for mobile domain.

In our work, the generated recommendation system is a context-aware process. Each user is defined by their instant context and moves in the context spaces. Context is regarded as a three dimensional vector defined by the x and y coordinates of location and time. The users’ instant contexts have determinative influence on the users when choosing an action. Namely, it is possible to say that each action has a context profile which represents points of occurrence in the context space, and the users moving in the context space always choose an action such that the context profile matches the users’ instant context. For example, the context profile of a museum can be described semantically as following: “The museum A placed at coordinate X and Y can be visited at weekday at 8:30am-4:30pm”. This data can be stored manually in database with a proper format. In a context aware recommendation process such an approach may be proper. For each user, context profile of proper action set fitting the user’s instant

context can be queried from the database, and this set can be used in the recommendation process. But this approach may have some shortcomings. Firstly, it will not be practically applicable since computation process becomes a huge burden where number of actions and users reaches millions in the database. Secondly, it is not convenient to assign approximate context profile to each action manually, since it is not known to what extent this approximate data is reliable. Also how this approximate data will be obtained is a live problem of what this approximation job will be based upon.

To alleviate the shortcomings mentioned above we propose a method based on "Fuzzy C-Means Clustering". Fuzzy C-Means (FCM) is a data clustering technique wherein each data point belongs to a cluster to some degree that is specified by a membership grade. It provides a method that shows how to group data points that populate some multidimensional space into a specific number of different clusters, and it allows one piece of data to belong to two or more clusters. The data derived from the users' context based action logs is provided as an input vector set to the FCM. Remember that each element of input set consists of three dimensional data called x, y and t. The output of the FCM is cluster centres together with their memberships in the form of a matrix. The coordinates for each output cluster centre are in the rows of the cluster centres matrix, each row representing a point (x, y, t) in the context space, and each of them representing the context profile for all members of that cluster. Furthermore, these cluster centres refer to action density points in the context space. These FCM results can be used in recommendation in such way that when a user enters into neighbourhood of a cluster centre, the actions being members of the cluster will be possible actions to be recommended.

1.3.2 Prediction in Context Space

In the mobile domain, behaviours of the users are more ordered than user behaviours in the web domain. What we mean is that generally while surfing on the web the users do not plan their actions beforehand since instantaneous and random actions do not

introduce additional cost in the web domain. But in real world unplanned decision regarding actions will not satisfy the users' needs and introduce additional cost. Consequently, any user does not think about too much before clicking a link to a web page, but consider attentively where to go for dinner beforehand. So, it will not be useful to recommend an action related to the users' instant context. Since the users usually make their decision about actions previously, as a result it will be difficult to change their prearranged decisions by recommending some alternatives. From this view point, prediction according previously planed the users' movement in the context space also becomes a critical issue. Or putting this into different words, it will be more useful to recommend actions related to the users' next status in the context space or recommend actions placed in any cluster way before the users move into that cluster.

To predict the users' movement in the context space, we use a method based on stochastic learning automata, which is introduced in [36]. They use this approach to predict the users' transitions between base stations of a mobile network. A similar approach is accommodated to our case. A learning automaton can be defined as a decision maker which operates in a random environment, updating its strategy for choosing actions on the basis of the environment's response. The automaton has a finite number of actions and the response of the environment to each action can be either favourable or unfavourable. In our system, as a result of FCM processing, the entire context space is now represented by fixed points called the cluster centres. There, we can represent movements of any user by transitions between cluster centres, considered as a finite number of context dependent actions. To this extent, we apply the output of the FCM to a stochastic learning automaton. The term stochastic emphasizes the adaptive nature of the automaton. This adaptation is the result of the learning process. Hence, the learning process can be described as deduction of routines in transition between the cluster centres. It accepts previous transition of a user as an input and tries to predict his/her next transition, which is, the user's next cluster. And obviously, the actions covered by predicted cluster with highest membership degree constitute possible recommendations. By selecting an action from recommendation, the user provides a

feedback, which is the user's next cluster, to stochastic learning automaton. According to feedback coming from the user, it applies penalty or reward process and updates related probabilities.

1.3.3 User Profiling and Generation of Recommendation

Using FCM and stochastic learning automaton we are at a stage that instant context of a user has been found, and the next context has been predicted. Now, the remaining problem is that of recommendation problem by choosing actions that matches the user's profile. Here, two factors effect the user's profile: similarity of the user to other users and interests of the user. Namely, the remaining problem is to complete the design of users profiling mechanism.

Pure collaborative filtering algorithms work well where the items are comprised by single category. For example, in an online movie portal all contents/items consist of movies, and then obviously, all users of the portal are interested in movie. In other words, it is assumed that each user of the portal can be respected as an expert on movies to some extent. Consequently, it is possible to look for direct collaboration between those users. But as for mobile domain contents/actions have heterogeneous miscellaneous structure, and they are divided into many ontological categories (like restaurant, cinema, historical place … etc). Each user may be interested in only one, two or more categories. In this case, it is improper to establish collaboration between all users. Rather than establishing direct collaboration, we propose to construct interest profiles for the users.

The approach proposed in [19] will be adopted to build users' profiles. The proposed approach is called as Unsupervised Fuzzy Divisive Hierarchical Clustering (UFDHC). They apply this method on a web site consisting of many ontological categories. Each user is represented by the number of access to each category. Access log data are input to UFDHC. The UFDHC method is a fuzzy divisive hierarchical method, which

determines a structure in the data in the form of a binary tree by firstly detecting large clusters and then splitting them in necessary numbers to generate a final objective classification. The output of UFDHC is a binary tree of user groups characterized by a set of common interests and represented by a prototype, which would correspond to the profile of the users of the group in our case. We therefore use the same approach for profiling. In our case, action set consists of subsets represented by ontological category. Similarly, the access logs to each category are stored for each user, and they are used as input. The output is the binary tree of the users' profiles (or user clusters) representing the interest degree to each category in the mobile domain. Each cluster is identified by a prototype which summarizes the navigation preference of the users strongly belonging to that cluster, thus identifying the profile of its typical members. We utilize the hierarchy of clusters in the recommendation process as follows: only the actions relating to a specific cluster (profile) are recommended to that cluster's members. Furthermore, the members of each cluster have similar interest on ontological category, and the collaboration can be established between the same cluster members.

The recommendations are generated based on two data: next cluster id predicted by stochastic learning automaton and the user's profile retrieved from binary tree produced by UFDHC. Among the categories comprising the user's profile, three categories are determined by Roulette Wheel. The higher percentage of contribution degree the categories have, the more chances to be selected they have. Therefore, the recommendation set presented to the user consists of those actions, which are included in one of three categories and are member of the predicted next cluster.

Finally, the problem of designing a recommendation for mobile domain is defined as an architecture combining context awareness, prediction and user profiling in the recommendation process.

1.4 Contributions

Here, a context-aware recommendation is proposed for mobile domain. The recommendation problem is divided into three sections: Context-awareness, Prediction in context space and user profiling. Through this work our contributions are as follows:

- The FCM clustering is proposed for context awareness. By means of the FCM, the density points (cluster centres) of action occurrences are obtained. These density points refer to context profiles of the members, which are actions, belonging to the cluster. Thus, the context-awareness problem is solved in an intelligent manner.

- To predict users' movement in context space, we use an approach based on stochastic learning automata, which is proposed in [36]. They use this approach to predict the users' transitions between base stations of a mobile network. This approach is accommodated to our case. Instead of base station, cluster centres are produced by FCM, and user's motions between cluster centres are predicted. In other words, the problems of context-awareness and prediction in context space are solved in single architecture by means of combining FCM and stochastic learning automaton.

- For user profiling, the approach proposed in [19] is used. They apply this method on a web site consisting of many ontological categories. Each user is represented by the number of access to each category. Access log data are input to UFDHC. Similarly, the actions are represented by ontological categories in the mobile domain. Using the same approach, the interest profiles of the users of the mobile domain are extracted. The access logs to each category are stored for each mobile user, and they are used as input to UFDHC. The output is the binary tree of the users' profiles (or users' clusters) representing the interest degree to each category in the mobile domain.

- Finally, two data, the user's profile data from UFDHC and the predicted point of the users in the context space from FCN and stochastic learning automaton, are used to produce recommendations.

1.5 Organization of the Chapters

Previous works on related issues were surveyed and could be grouped into three categories: recommendation system, context-awareness and path prediction. In CHAPTER 2 the detailed description of these three categories and prior works related to them are given.

In CHAPTER 3 our methodology, which is based on FCM, stochastic learning automaton, UFDHC and generation of recommendation respectively, is introduced in a detailed manner.

After proposing the method, the design pattern and implementation details of the simulator are given in CHAPTER 4.

The generation of simulated data is explained and test results with this simulated data are presented in CHAPTER 5.

Finally, the general inferences drawn from this work and possible future works are presented in CHAPTER 6. This is the conclusion part of the thesis.

CHAPTER 2

BACKGROUND AND RELATED WORKS

2.1 Recommendation Systems

Firstly, let describe what the meaning of recommendation is. It means to present to the user relevant information that suits the user profile in terms of his/her general interests, location, current activities and also planed ones for the near feature filtered things to the user. According to this description, the task of a recommender system is to provide personalized suggestions about items that the user might find interesting. In other words, it can be defined as an adaptive function mapping users to items set under specific user based criteria. It is difficult to give an exact and general definition of recommendation system because its details vary as much as the application environment varies. Instead of trying to formulate a general definition, it is better to address the general features of recommender systems. Up to now, many features have been defined by researchers worldwide. As a result, their definitions bear somewhat domain specific features. After a detailed investigation on works [2, 4, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 21] has been done so far, we think that following features are common for all the recommendation systems.

- There exists a user profile representation format, which is a structured representation of the user's habits, interests and needs, being particular to that domain.
- A knowledge acquisition technique must be employed to gather feedback (positive and negative sample, some times called training samples) from users.
- Every recommendation system uses a technique to mine training samples from which the system infers the user preferences (machine learning).

2.2 User Profile Representation

Roughly, a user profile is a structured representation of the user's habits, interests and needs. User profile representation is always domain specific and depends on the filtering method used for training the system. The profiles are inputs of the filtering method.

Generally speaking, profile representation falls into three classes [2]: rating-based representation, content-based representation and knowledge-based representation. Namely, rating based representation usually refers to profiling in collaborative filtering where it is difficult to differentiate items as to its content, and content-based representation usually refers to profiling in content-based filtering where items' content consist of descriptive attributes.

2.2.1 Ratings-based Representations

When users receive recommendations it is common to elicit feedback on how interesting the recommendations are to the needs of the user. This type of feedback is called relevance feedback. Relevance feedback is elicited by offering the user a rating scale for each recommendation; the choice is commonly either "interesting" and "not interesting" or a 3 to 5-point scale of interest [4, 8, 10, 11, 12, 13]. The representation of relevance feedback is thus a set of recommended items and the associated interest values provided by each user.

Relevance feedback can be acquired implicitly, allowing inference from observed user behaviour [2]. The problem with implicit feedback is that the assumptions made to allow inference often introduce errors. For example, a user may read an initially interesting looking document, only to find it was actually not interesting after all when its details are known; if all documents that are read are inferred to be interesting this situation would clearly introduce an error into the relevance feedback acquired.

A balance must be made between interrupting the user to acquire high quality explicit feedback and unobtrusive methods to obtain lower quality implicit feedback. Exactly how much interruption the users will tolerate will depend upon the specific application domain.

2.2.2 Content-based Representations

Most content-based analysis is performed on textual documents such as web pages, newspaper articles or document abstracts. The reason for this is that textual documents easily break down into individual words, whereas video and audio sources require sophisticated analysis to decompose into useful sub-components. Almost all content-based recommender systems work with textual content.

Term-frequency vector representation: The most common abstraction of a textual document in the machine-learning context is a term-frequency vector. To create a term-frequency vector the terms within a document are counted and the frequency values stored in an n-dimensional vector [14, 15, 16]. The number of dimensions of the vector is the number of unique terms within a document.

Binary class profile representation: The most common profile representation for the content-based recommender systems is the binary class profile, representing user interests as a set of positive and negative examples [45]. The positive, or “interesting”, examples are represented as a collection of term-frequency vectors of documents that the user has rated as “interesting”. The negative, or “not interesting”, examples are likewise represented. This binary class representation is very suitable for a great many machine-learning techniques.

Since relevance feedback is required to obtain the sets of positive and negative examples, a ratings-based profile is often additionally implemented to create a hybrid recommender system.

Multi-class profile representation using an ontology: The alternative to the binary class representation is a multi-class representation. Rather than simply having positive and negative classes, an ontology of classes can be created that map to domain concepts such as newspaper topics like “sport”. A user’s profile is thus represented in terms of which classes they are most interested in, abstracting away from the specific examples of interest [19]. When relevance feedback is acquired, examples of interest are classified according to the classes within the ontology, and the user’s interest in that class recorded.

Multi-class classification is considerably more complex than binary class classification. Having more than two classes reduces the number of examples available for each class, thus reducing the accuracy of the machine-learning technique employed. In addition, since classes are shared between users, there will be a loss of information about individual user interests when compared to a binary representation where each user has their own set of examples; sharing examples does allow for a larger training set, however. These factors are the reason why very few recommender systems adopt this approach.

Most ontology’s are created manually by a knowledge engineer and domain experts. They thus capture the relevant classes within a domain and relationships between them. It is possible to create classes automatically using clustering machine-learning algorithms. Clustering finds similar term frequency vectors and groups them together to make a class. Classes created by clustering, however, have no domain knowledge associated with them, making useful inference from them difficult.

2.2.3 Knowledge-based Profile Representation

Knowledge-based profile representations appear in the user modelling literature. Typically these approaches require questionnaires and interviews with users to acquire information about their requirements before a profile can be built [22]. Profiles consist

of asserted facts about a user in a knowledge-base, from which inferences can be drawn about user stereotypes and interests. Knowledge-based profiles are often used in the related fields of agent and intelligent tutoring systems.

The literature proposes the user profile representation in three different classes as we have just overviewed. However we want to mention about another classification of profile representation. [3] exists in the literature classifying the profile representation problem into four topics: static content profiling, dynamic content profiling, static collaborative profiling, and dynamic collaborative profiling. Static content profiling refers to the gathering of static information regarding the user usually upon registration. Typically, systems allow users to enter a simple profile when they first register with the system. It is static as the registration is done only once. Knowledge based representation discussed above can be considered as a static profiling. For dynamic content profiling, the system gathers information based on the dynamic changes in the behaviour of the user. This means that the system should keep track of the user's behaviour when interacting with them. Term frequency vector representation discussed within Content-Based representation is a dynamic content profiling as well. The concept of static collaborative profiling refers to explicitly clustering users with similar behaviours through user's explicit request. Every time a new user is added into the system, the system will take a period of time to collect information about the user and to construct the user's profile with information that will aid the system in serving the user's needs. We can reduce the learning curve of the system by reusing a current user's profile by matching the new user's profile with other current user's profile. The categories or terms listed in the user's profile are matched across other users' profiles. If the term or keyword in the user's profile is found in another user's profile, the similarity measure for these two users is increased accordingly. Dynamic collaborative profiling refers to clustering users with similar behaviour into peer groups based on the user's profile and filtering information pertaining to group's interest.

In static collaborative filtering and dynamic collaborative filtering the term ‘profiling’ also contains collaborative filtering. In the literature, sometimes the term profiling means the output of the collaborative filtering method.

2.3 Knowledge Acquisition

The knowledge acquisition technique is also a domain specific feature. It is determined by taking into account all users’ behaviours. Knowledge can either be implicitly or explicitly acquired from the user [2, 14]. Implicit knowledge acquisition is often the preferred mechanism since it has little or no impact on the user’s normal work activity. Unobtrusive user monitoring of the user discovers behavioural data about the user’s normal work activity over a period of time; this data can be used to infer preferences for frequently occurring items. Heuristics can also be employed to infer facts from existing data. Implicitly acquired knowledge requires some degree of interpretation to understand the user’s real goals; this is an inherently error prone process, reducing overall confidence in any resulting user profiles.

Explicit knowledge acquisition requires the user to interrupt his/her normal work to provide feedback or conduct some sort of programming of the system. Explicit knowledge is generally high confidence information, since it is provided by the users themselves and not acquired from indirect inference. Feedback types include item relevance, interest and quality. User programming occurs when the user is asked to create filter rules, either visually or via a programming language, or to tell the system about groups or categories of items that exist in the domain.

Both implicit and explicit knowledge acquisition have advantages and disadvantages. In some application [14] they are hybridized to alleviate each others shortcomings.

2.4 Recommendation Technique

In the past years, many recommendation techniques have been developed. That can simply be categorized into two classes: content-based filtering and collaborative filtering. In addition, the hybrid filtering may be considered as third class [4]. It seems proper to categorize recommendation techniques as in Figure 2.1.

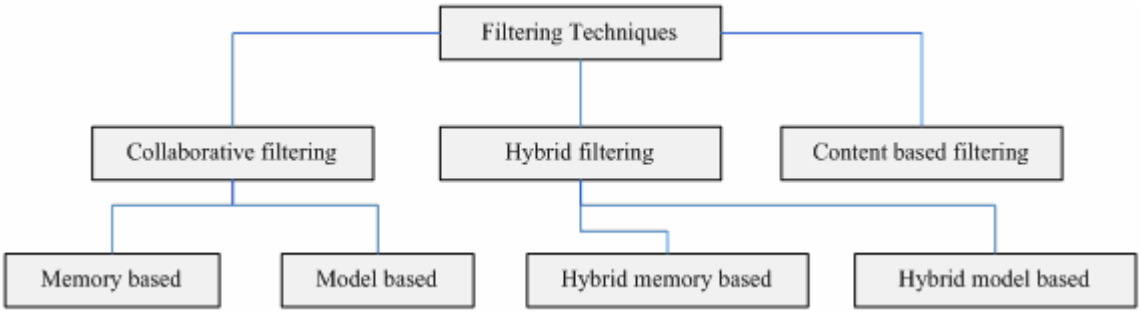

Figure 2.1 Categorization of filtering techniques.

2.4.1 Content-Based Filtering

Generally, content-based filtering has been applied to information retrieval. Namely, this technique is an alternative paradigm that has been used mainly in the context of recommending items such as books, web pages, news, etc. content-based methods make recommendations by analyzing the description of the items that have been rated by the user and the description of the items to be recommended. A variety of algorithms have been proposed for analysing the content of text documents and finding regularities in this context that can serve as the basis for making recommendations. Many approaches are specialized versions of classification learners, in which the goal is to learn a function that predicts which class a document belongs to (i.e., either liked or not liked). Other algorithms would treat this as a regression problem in which the goal is to learn a function that predicts a numeric value (i.e., the rating of the document). There are two important sub problems in designing a content based filtering system. The first is finding

a representation of documents. The second is to create a profile that allows for unseen documents to be recommended [5].

A personalized information filtering method that learns user's interests by observing his or her behaviours during the interaction with the system is proposed in [14]. The web documents are presented with term frequency vectors and users' profiles consist of weights each of them represents the interest degree of user to a specific topic. To construct the users' profile, they implement reinforcement algorithm. Filtering is viewed as an interactive process which involves a generate-and-test method, whereby the agent tries actions, observes the outcomes, and selectively retains those that are the most effective. Most of the time WAIR chooses the highest-ranked documents, but with probability ε, it chooses lower-ranked documents too. The rationale behind this policy is that it combines exploitation and exploration of search behaviour.

A genetic algorithm is used to build user profiles from a collection of documents previously retrieved by the user [15]. A gene in the chromosome of the genetic algorithm is defined by a term and a fuzzy number of occurrences of the term in documents belonging to the class of documents that satisfy the user's information need. In this way, the terms that allow the system to discern between good and bad documents are selected and stored as a part of the user's profile to be used in future queries to the system.

Users' profiles can also be represented as an interest hierarchy. Firstly, the documents are clustered then each document is assigned to a node in hierarchy [16]. By using document access patterns of the user, which can also be regarded as user's access patterns to nodes, a user interest hierarchy is built. A user might possess interests at different abstraction levels. The higher-level interests are more general, while the lower-level ones are more specific.

As an improvement on existing content-based filtering, an adaptive algorithm for learning the changes in user interests is proposed in [21]. They extend the model for the purpose of information filtering by taking into account the user current interests and their decay in time. Queries are represented as TF-IDF feature vectors with an additional temporal dimension (current interest weight) set to a preset positive initial value that decays in time. This fact implies that some specific user interests could decrease as time goes on. However, user interest for a category can be maintained/increased if the user is searching for elements belonging to an already existing category in his profile. The scheme proposed in this work keeps track of both the user's Recent and Long-Term Profiles.

2.4.2 Collaborative Filtering

Note that content based filtering works well where the contents have well defined vector representation. But in many application areas it is not possible to define appropriate item descriptors as well as text based items. Collaborative filtering is proposed to alleviate shortness of the content based filtering where analysis of contents is difficult, that is, it is not possible to represent the contents by means of well defined descriptor. Thus, performance of collaborative filtering surpasses that of any content based filtering in many domains.

Collaborative filtering predicts the utilities of items for a particular user based on the rating information for the same set of items given by many other users. Two approaches have been developed for collaborative filtering [6]. The first approach, referred to as user-based, relies on the fact that each persons belongs in a larger group of similarly behaving individuals. As a result, items frequently liked by the various members of the group can be used to form a basis for recommended items. In many research papers user-based approach is called memory based approach [7, 8, 9]. As to [7] another description of this approach, nearest neighbour based collaborative filtering algorithms are categorized as being memory-based. Nearest-neighbour methods use some notation

of similarity between the user for whom predictions are being generated and other users in the database. Because of the simplicity and robustness, the memory based approaches have been widely used in many real world applications.

Despite its success in many application settings, the memory-based filtering approach has been reported to have nevertheless several major limitations including the scalability, sparsity and real-time performance problems. The scalability problem is that the computational complexity of these methods grows linearly with the growing number of customers and items, which in typical commercial applications can grow to be several millions. The use of dimension reduction techniques have been proposed to address this issue. The sparsity problem occurs when processing or feedback data is sparse and insufficient for identifying neighbours and it is a major issue limiting the quality of recommendations. Finally, even though the throughput of algorithms can be increased by increasing the number of servers running the recommendation algorithm, they can not decrease the latency of each top-N recommendation, which is critical for near real time performance. One way of reducing the complexity of the nearest-neighbour computations is to cluster the users and then to either limit the nearest-neighbour search among the users that belong to the nearest cluster, or use the cluster centres to derive the recommendations. These approaches, though they can significantly speed up the recommendation algorithm, tent to decrease the quality of the recommendations.

In [20], the author proposes an approach to deal with sparsity problem by applying an associative retrieval framework and related spreading activation algorithms to explore transitive associations among consumers through their past transactions and feedback. By exploring the transitive interactions between consumers and items, the consumer–product interaction matrix is augmented and becomes meaningfully “dense” for recommendation purposes.

For many memory-based approaches, items are treated with equal importance. Apparently, this is undesirable because the discrepancies in different items have not

been taken into account. To address this problem, a new weighting scheme based on the leave-one-out (LOO) method is proposed [9]. The work is built upon the intuition that the rating behaviour of an individual user should be similar to the rating behaviours of some but not all other users. Therefore, a good weighting scheme for items should bring users of similar interests closer and meanwhile separate users of different interests further apart. They formalize this idea into a probabilistic optimization problem, in which the appropriate weights of items are found by maximizing the likelihood for each user to be similar to at least one of other users.

Another improvement on memory-based filtering is reported in [17]. The nearest-neighbour (NN) algorithm fails because the predictions it generates fail to take into account the quality and quantity of the information used to generate each prediction. They create a new algorithm based on belief distribution that enabled a system to predict not only rating values, but also to explicitly represent the uncertainty in each predicted rating.

The second approach is known as model-based which groups together different users in the training database into a small number of classes based on their rating patterns. In order to predict the rating from a user on a particular item, these approaches first categorize the test user into one of the predefined user classes and use the rating of the predicted class on the targeted item as the prediction.

Compared to the memory-based approaches, the model-based approaches have an advantage that only the profiles of models need to be stored. However, the memory-based approaches are usually much simpler than the model-based approaches and require little offline computation whereas model-based approaches usually have to spend many computation cycles on creating model profiles. Furthermore, the model-based approaches tend to assume that a small number of user classes are sufficient for modelling the rating patterns of many different users, and thus may lose the diversity of users. Finally, model-based approaches tend to perform worse than the memory-based

approaches when the number of training users is small. This is because ratings by only a small number of users are usually insufficient to create reliable clusters of users.

Sparsity and cold-start problems exist in both memory-based and model-based approaches. According to the definition given in [22], the latter means that it is very difficult to give a recommendation for new products totally without rating or little rating at the beginning for the recommendation system which use a pure collaborative filtering approach. To alleviate the cold-start problem, the integration of content-based rating to collaborative filtering is a remedy.

An adaptive algorithm improves the recommendation quality of a recommendation system by an optimization algorithm which optimizes the users' ratings based on two assumptions [22]. For items rated on a numerical scale, as the number of ratings increases, the average rating approaches the actual "value" of the item. The good reviewers are those who consistently predict the average rating of items.

A model-based approach is implemented in [6]. The models are built by analysing similarities between items and then these models are used to identify the set of items to be recommended.

In [12] author describes the collaborative ensemble learning, a novel statistical learning approach to this task. It at first applies probabilistic support vector machines (SVMs) to model each individual user's profile based on given examples, i.e. liked or disliked paintings. Due to the high complexity of profile modelling, the SVMs can be rather *weak* in predicting preferences for new paintings. To overcome this problem, we combine a society of users' profiles, represented by their respective SVM models, to predict a given user's preferences for painting images.

For typical web sites user, a model-based approach which mines the users' access patterns to built users' profile is proposed in [19]. They assume that the pages of the web

site have been prearranged in a number of different categories. Each user is, therefore, represented by the number of access to each category. Access log data are input to a slightly modified version of the Unsupervised Fuzzy Divisive Hierarchical Clustering (UFDHC) method. The UFDHC method is a fuzzy divisive hierarchical method, that is, it determines a structure in the data in the form of a binary tree by firstly detecting large clusters and then splitting them as much as it is necessary to generate a final objective classification. The output of UFDHC is a binary tree of user groups characterized by a set of common interests and represented by a prototype, which corresponds to the profile of the users of the group.

2.4.3 Hybrid Filtering

Pure collaborative filtering approaches haves some shortcoming as mentioned above. Many researches have been done to over come these shortcomings. Hybrid methods that combines collaborative filtering and content based filtering are one of the approaches being proposed to alleviate collaborative filtering short coming. The works [10, 11, 4, 12, 13, 8] are examples of hybrid approaches.

In [10] the author suggests that content-based and collaborative filtering can be combined under a nonparametric hierarchical Bayesian framework.

A hybrid model is implemented for information retrieval as an extension of pure memory-based approach in [4]. In general item are represented by users rating in collaborative filtering. But, in this work firstly text based items are represented by key word and then ratings are assigned to that keyword. Lastly, memory based algorithm is applied, and the result of memory based methods is an matrix showing that whether a user thinks another user's evaluation of any keyword is reliable or not.

In [13] a novel, unified approach that systematically integrates all available training information such as past user-item ratings as well as attributes of items or users to learn

a prediction function is proposed. The key ingredient of this method is the design of a suitable kernel or similarity function between user-item pairs that allows simultaneous generalization across the user and item dimensions. Also they propose an on-line algorithm (JRank) that generalizes perceptron learning.

A recommender system which combines collaborative filtering and content based filtering is applied as for the Web-based shopping mall [18]. The proposed algorithm is based on two ideas: agents insert artificial ratings into the system using collected users' behaviour and taste similarities between users and similarity in features between items previously rated by the user are both used to recommend items for a user. In this work, both approaches (content-based and collaborative filtering) are implemented separately. For each user item pair, two score are produced by content based and collaborative filtering then combined with a proper weight.

2.5 Context Awareness

In sections 2.1, 2.2, 2.3 and 2.4 we have investigated in detail existing leading recommendation systems. Note that most of them regard users' interaction with the system as a "desktop experience". But in this thesis our aim is to design a recommendation system for mobile environment where the users' interaction is not restricted to solely live a "desktop experience". Computation is now packaged in a variety of devices. Smaller and lighter laptop/notebooks, as powerful as conventional personal computers, free us from the confines of the single desk. Specialized devices such as handheld personal organizers are portable enough to be with us all the time. Wireless technology allows devices to be fully interconnected with the electronic world. Also advances in software technology and the increasing volume of digital knowledge offer the opportunity for more sophisticated and user-friendly digital services. Mobile software agents represent one of recently emerged paradigms related to pervasive computing that carry a tremendous promise in solving some of the real world problems

and enabling “anytime, anywhere, with any device, within any context” access to digital information and services.

The mobile environment can be considered as a domain consisting of sub domains such as restaurants, cinemas, concerts…etc. And contents of the mobile environment can be defined as actions under those sub domains. Because of the high granularity of the mobile environment, finding a solution to the recommendation problem is not as standard as for the web domain. Beyond the granularity, other factors make recommendation problem more complex such as mobility and ubiquity. These factors cause dynamic changes on the user’s context being strictly determinative on the user’s behaviour.

The term context covers a large amount of information which is perceived by the mobile user and affects his/her behaviour in the mobile environment. [23] defines context as physical and social situation. A more detailed definition of context is given in [25], such that context is what surrounds, and in mobile and ubiquitous computing the term is primarily used in reference to the physical world that surrounds the use of a mobile device. This has also been referred to as physical context, to distinguish it from other kinds of context in mobile computing, such as conditions in the surrounding systems and network infrastructure. As to these definitions, the context may consist of information such that location, time, temperature, brightness… etc. If such contextual information about a mobile user becomes available, a recommendation system can adapt itself to the user context and produce more valuable recommendations. In the literature, exploitation of contextual information by software agent for any purpose is called as “context-awareness”.

In [24] the term context-awareness is defined as intelligence that enables an application to discover reason and predict a situation and adapt to it in a dynamically changing environment. The use of context-awareness in mobile domain is receiving considerable attention in various fields of research including mobile computing, wearable computing,

augmented reality, ubiquitous computing and human-computer interaction. The actual utility of context-awareness in mobile systems has been demonstrated in a wide range of application examples. In [25] author classifies utilization of context into three levels. At the first level, context-sensitive resource and power management are exploited. At the second level, context-awareness enables both adaptive applications and explicitly context-based services. At the third level, the use of context facilitates a shift from explicit to implicit human-computer interaction, towards less visible if not invisible user interface. The aim of our work covers both second and third level utilization.

Location based services hosted in mobile network are sample of context-aware application. In [26] all attributes of data about a user is separated into two classes: predictive attributes and contextual attributes. Predictive attributes refer to personal data used in web based recommendation system. Contextual attributes refers to user's current location, time, weather, current user's mood … etc. To handle both class of information, they propose a multilevel probabilistic model (the Bayesian Metanetwork), which is the extension of traditional Bayesian networks and is also considered as a useful tool to predict a mobile user's preferences.

GeoNotes is another location based information system [27]. In this work authors present general aspects of designing a location based information system for public spaces. Also they implement proposed architecture in a campus Wireless LAN network.

A context-aware personal assistant which helps users in daily activities is proposed in [28]. The approach consists of context-awareness and user modelling parts. The first part implies a persistent database and context management software. The context is modelled hierarchically and expressed using XML technologies. In the second part, a user modelling agent which can predict a user's behaviour is proposed.

A mobile context-based hypermedia retrieval system is proposed for cultural heritage in [29]. They define context as the user's location in virtual space and the particular mobile

device being modelled together with user preferences or profile. The purpose is to automatically push relevant data from the database server to the client based on this comprehensive definition of the user's context.

Ubicomp[31] is a context-aware recommendation system for mobile environment. It extends the collaborative filtering algorithm to handle contextual data as well as collaborative data.

PILGRIM[33] is a location-based recommendation system. The aim of this work is to develop a location-aware recommendation system which should be able to produce a top-k items list for a given user whose location is known with a precision ranging from a few meters to some hundreds of meters. Recommendation problem is viewed as a N-top recommendation but they also depend on the user’s location.

In [34], the problem defined as power consumption resulted from communication overhead for handheld devices. The communication overhead is introduced by user interaction in information retrieval system. A situation-aware personalized information retrieval system is proposed to diminish user interaction with the system.

Also multipurpose tools have been developed to facilitate the development of context-aware applications. CAPpella[30] is a tool that allows users to build context-aware applications by demonstration. They use supervised learning for adaptation of software to any context. Any context is defined as a value-attribute pair, and a task is assigned to each samples. Samples are provided to software as training input. After training, the software can adapts itself to dynamic context, and implements relevant task.

Sycophant[32] is a context learning calendar application program which learns a mapping from user-related contextual features to application actions. A computer is viewed as a stationary robot with simple sensors such as for motion and speech. The aim

of this agent is to learn users' routine by using machine learning techniques. The proposed approach resembles somewhat to [30].

2.6 Path Prediction

Assume that a context-aware recommendation system producing recommendations in accordance with the user's context and interest is developed. Actually, the recommendations are produced as alternatives to the user's prearranged behaviour. "Prearranged behaviour" means that generally actions of user are more prearranged in real world compared to web applications. In other words, when a user moves from current location to next location, actions of user in next location has been planed before hand and instantly suggesting different alternatives to user will be useless since probably they will not affect the user behaviour. As a result predicting the user's next location becomes important. If it is possible to predict the user's next location in mobile environment, the related recommendations can be produced beforehand, which also becomes more valuable.

The problem of prediction of user movement in mobile environment is called as "path prediction". The mobile environment which is a plane is represented by constant nodes. Motions of the mobile users are represented by transitions between these nodes. Firstly the system is trained with the users' motion history. After training process, current user's node or previous transitions are provided to system as an input, then the system predicts user's next node as an output.

Such approaches generally used to solve resource allocation problem in mobile network. Mobile network is represented by base station, and each based station defines a cell. Thus, a user's movement can be represented by transitions between cells on the plane. Predicting the probabilities that a mobile user will be active in other cells at future moments poses a significant technical challenge to network resource management in

wireless environment. The probability information of users' next cell can be used to achieve maximum resource utilization.

In [35] an approach is proposed for prediction of user mobility profile. They propose a novel adaptive fuzzy logic inference system to estimate and predict the probability information for wireless communications networks. The estimation is based on measured pilot signal strengths at the mobile user from a number of nearby base stations, and the prediction is obtained using the recursive least square algorithm.

Another work on path prediction based on Learning Automaton is presented in [36]. The learning automaton, based on a properly structured knowledge, assigns probability values to the neighbourhoods of the currently visited cell. The highest probability denotes the most likely to be visited cell. Two learning automatons, the global and per-user, are used. Obviously first one accepts all mobile users' path histories as training input, and the other accepts per user's path histories as training input. The output of two algorithms is combined with a proper weight for prediction of user's next cell.

In [37] path prediction is used to improve TCP performance in wireless network. Proposal is based on stochastic datagram relocation. Traffic destined to the mobile terminal is tunnelled to and cached into adjacent cells according to the output of a path prediction algorithm. To reduce the associated overhead, only percentages of inbound traffic are copied to the cell's neighbourhood on the basis of estimated probabilities. The time scheduling for datagram relocation is also taken into account.

We design a method combining approaches from three different field, which are recommendation systems, context-awareness and path prediction. Our methodology, which is based on FCM, stochastic learning automaton, UFDHC and generation of recommendation respectively, is introduced in a detailed manner in Chapter 3.

CHAPTER 3

PROPOSED RECOMMENDATION SYSTEM

Through our survey the “web-based recommendation”, “context-awareness” and “path prediction” were investigated respectively. Context-awareness and prediction in context spaces are new for the recommendation task. We use in this thesis the Fuzzy C-Means clustering and Stochastic Learning Automaton to handle context-awareness and prediction. Also, Unsupervised Fuzzy Divisive Hierarchical Clustering method is used to construct the users’ interest profile for ontological categories.

3.1 Overall Architecture

Block diagram of overall architecture is represented in Figure 3.1. The overall architecture is denoted by two main part, which are offline and online parts. The recommendation system proposed for the mobile domain consists of three sub parts, which are “context-awareness”, “prediction of user’s motion” and “user profiling”. Each sub part is represented by “Fuzzy C-means Clustering”, “PUA and GA” and “UFDHC” in offline part respectively and the output of each part is used in online part. The online part of the figure represents a live user experience.

Context-awareness means user’s context, which are contextual information such location and time, is used in generation of recommendation. In other words, items/actions are filtered through the user’s context, and then only those items/actions matching the user’s context are recommended. Therefore each item should be defined by a context profile. Firstly users’ actions have to be separated into subgroups (clusters) according to closeness of their occurrence in the context space. We achieve this using Fuzzy C-means

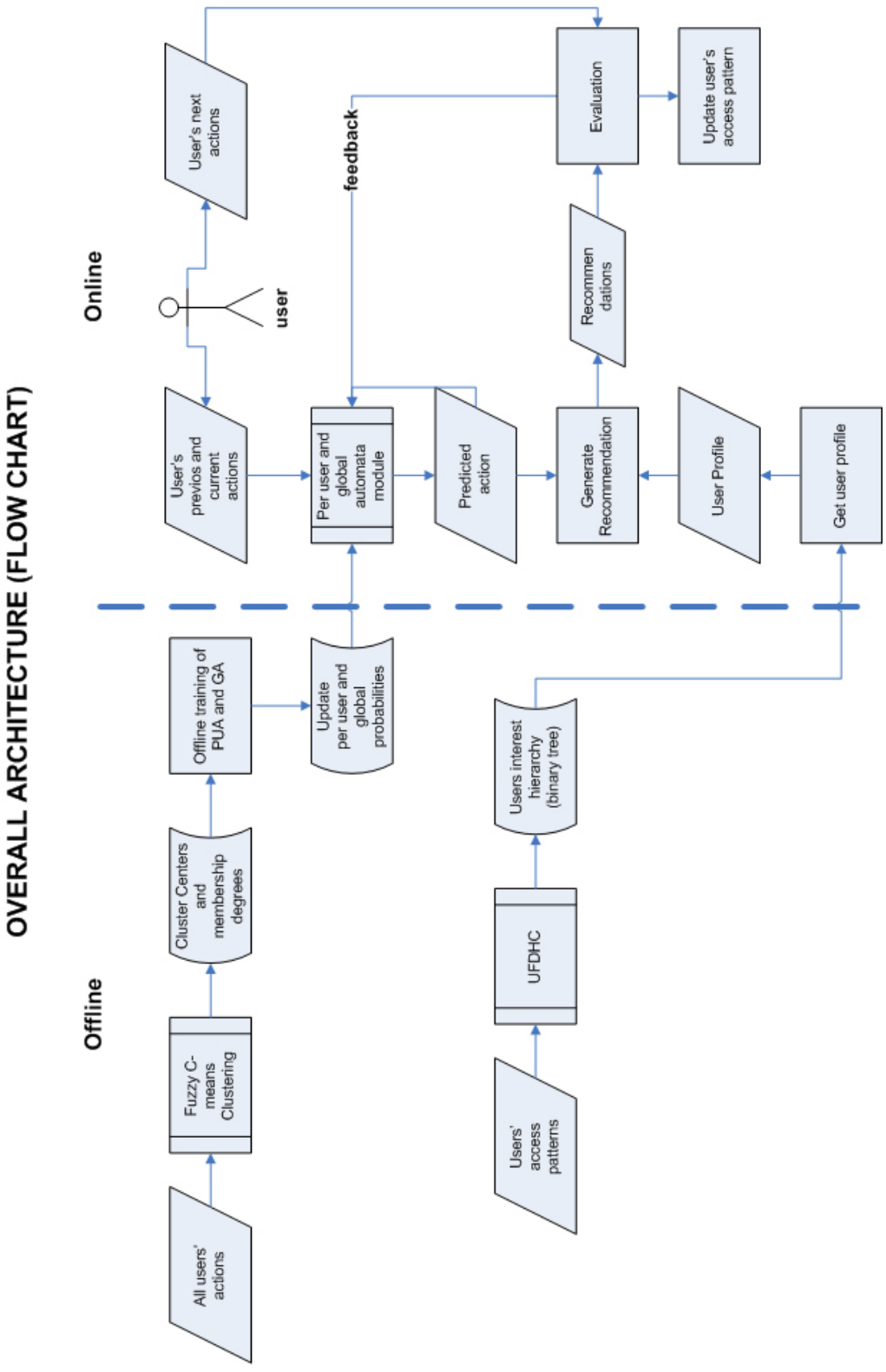

Figure 3.1 Overall Architecture

Clustering (FCM). In Figure 3.1 all users' actions are input to FCM and cluster centres and membership degrees are outputs. Actions contained in a cluster are represented by a single context profile which is the point called cluster centre. The context profile means that the actions contained in the cluster can occur at any point of that class, which is close to the cluster centres.

We consider all users move in a context space. If it is possible to predict the user's motion, the recommendation system can generate the recommendations previously, thus, it will be more useful to recommend actions related to the users' next status in the context space. The task of the second part called "prediction of user's motion" is to predict the user's motion in the context space. By means of cluster centre, the context space is represented by fixed points, and then motions of any user are defined as transitions between clusters. Therefore, we can predict the users' transitions between clusters beforehand. We use stochastic learning automaton (LA) for such estimation. To train LA, we have two alternatives in Figure 3.1: offline and online training. In offline training, LA (both per user and global automaton) is trained automatically using the training sample data (this data is produced by synthetic data generator) stored previously in database. In online training, training samples are provided manually via graphical user interface on behalf of a selected user. The general training of LA is as follows. The previous transition, which is represented by the previous cluster id and current cluster id, of the user is input to LA. Outputs are transition probabilities from current cluster to other clusters, each of them being candidate next cluster. To predict the next cluster among all these candidates Roulette Wheel Selection algorithm is used, to select most probable transition according to their probabilities of the candidates. Selected transition represents the predicted next cluster. According to feedback received from the user, LA adapts it self.

There exist heterogonous categories of items/action in the mobile domain. Therefore, probably each user will have interest profile on those categories. Since user interest profile is information showing the interest degrees of the user to each category, such

information should be exploited through generation of recommendations. To extract the users' profiles, we use Unsupervised Fuzzy Divisive Hierarchical Clustering (UFDHC). In Figure 3.1 users' access patterns to the categories are input to UFDHC and the output of UFDHC is binary tree of the clusters. Each cluster is identified by a profile which summarizes the categorical preferences of the users strongly belonging to the cluster.

The online part of Figure 3.1 represents a live user experience. In the online part the outputs of FCM, LA and UFDHC are exploited to generate recommendation for the user placed at the upper right-hand corner of the figure. According to the user's previous and current clusters (previous transition), LA predicts the user's next cluster (next transition). The recommendations are generated based on two data: next cluster id predicted by LA and the user's profile retrieved from binary tree produced by UFDHC. The user's profile consists of percentage of contribution degree of each category. Among the categories comprising the user's profile, three categories are determined by Roulette Wheel. The higher percentage of contribution degree the categories have, the more chances to be selected they have. Therefore, the recommendation set presented to the user consists of those actions, which are included in one of three categories and being member of the predicted next cluster. According to response (feedback) received from the user, the system adapt it self. Firstly, whether the user has moved to predicted cluster or not is evaluated by LA. If the predicted cluster is true, then the feedback is treated as positive, otherwise the feedback treated as negative. Secondly, the user's access pattern is updated according to action selected by the user, and recommendation process terminates.

3.2 Fuzzy C Means Clustering for Context-Awareness

We assume that contexts consist of three attributes: x coordinate, y coordinate and time. These three attribute constitute a context space with three dimensions. Also all actions of the mobile users are located in the context space. Actions of a mobile user can be defined as daily activities, for example, going restaurants, shopping centres, cinemas,

historical place … etc. Here, we assume that all users are tractable in the context space. When a user does an action, raw data related to this action is illustrated below.

User 1 went **Restaurant 1** at **12:35** on **Friday**.
User 1 went **Cinema 1** at **18:35** on **Friday**.
User 1 went **Bar 1** at **00:35** on **Saturday.**
User 2 went **Shopping Centre 1** at **18:35**.

Above we see Restaurant 1, Cinema 1...etc, and each of them refers to occurrence of a specific action. These raw data is stored in following form (Figure 3.2):

Figure 3.2 Structural representation of user action. Coordinate X and Y refer to a point where action is done.

Firstly, action histories of all users are retrieved from database as raw data then they are converted into vector form. Finally, the vector set obtained from conversion is accepted as training dataset by Fuzzy C-Means (FCM) algorithm. Block diagram of FCM is represented in Figure 3.3.

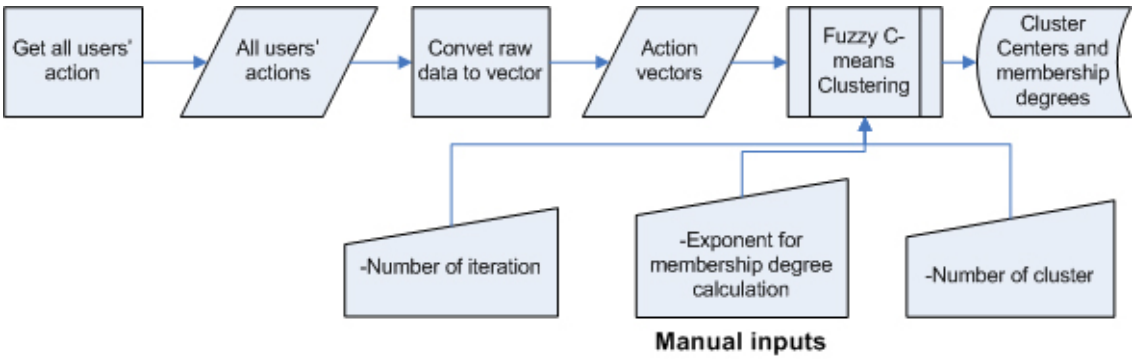

Figure 3.3 Flow chart of FCM

Clustering is a method for dividing scattered groups of data into several groups. It is commonly viewed as an instance of unsupervised learning. The grouping of patterns is then accomplished through the clustering by defining and quantifying similarities between the individual data points or patterns. The patterns that are similar to the highest extent are assigned to the same cluster. Clustering analysis is based on partitioning a collection of data points into a number of subgroups, where the objects inside a cluster (a subgroup) show a certain degree of closeness or similarity [38].

Fuzzy c-means [39] is a method of clustering which allows one piece of data to belong to two or more clusters. It is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N}\sum_{j}^{C} u_{ij}^{m} \left\| a_i - c_j \right\|^2, \qquad 1 \le m < \infty \qquad \text{(Eq. 3.1)}$$

where m is any real number greater than 1, u_{ij} is the degree of membership of a_i in the cluster j, a_i is ith of 3-dimensional(x, y and time) action sample, c_j is the centre of the cluster, and $\|*\|$ is any norm expressing the similarity between any measured data and the centre. Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centres c_j by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C}\left(\frac{\left\| a_i - c_j \right\|}{\left\| a_i - c_k \right\|}\right)^{\frac{2}{m-1}}}, \qquad \text{(Eq. 3.2)}$$

$$c_i = \frac{\sum_{i=1}^{N} u_{ij}^{m} . a_i}{\sum_{i=1}^{N} u_{ij}^{m}}, \qquad \text{(Eq. 3.3)}$$

This iteration will stop when $\max_{ij}\left\{\left|u_{ij}^{(k+1)}-u_{ij}^{(k)}\right|\right\}<\varepsilon$, where ε is a termination criterion between 0 and 1, whereas k is the iteration steps. This procedure converges to a local minimum or a saddle point of J_m.

The algorithm is composed of the following steps:

1. Initialize $U=\left[u_{ij}\right]$ matrix, $U^{(0)}$
2. At k-step: calculate the centres vectors $C^{(k)}=\left[c_j\right]$ with $U^{(k)}$
3. Update $U^{(k)}$, $U^{(k+1)}$
4. If $\left\|U^{(k+1)}-U^{(k)}\right\|<\varepsilon$ then stop; otherwise return to step 2.

The outputs of FCM are the matrices C and U. There exists a row for each a_i in U and each element in *ith* row represents to membership degree to each cluster. More semantically, each element of C is a point representing an action density in the context space and each action a_i is a member of each cluster with a membership degree u_{ij}.

Figure 3.4, which is generated by our simulator of which details are found in Chapter 4, shows a cluster and the actions which is a member of this cluster with the highest membership degree.

Ones the matrices C and U are created, membership degrees to each cluster can be calculated for any user moving in the context space. Then, for example it is possible to say that any user is a member of that cluster having highest membership degree in context space and actions being member of that cluster are possible activities. Hence, in recommendation process, the recommendation can be produced for only those actions. In Figure 3.5 a user with highest membership is showed.

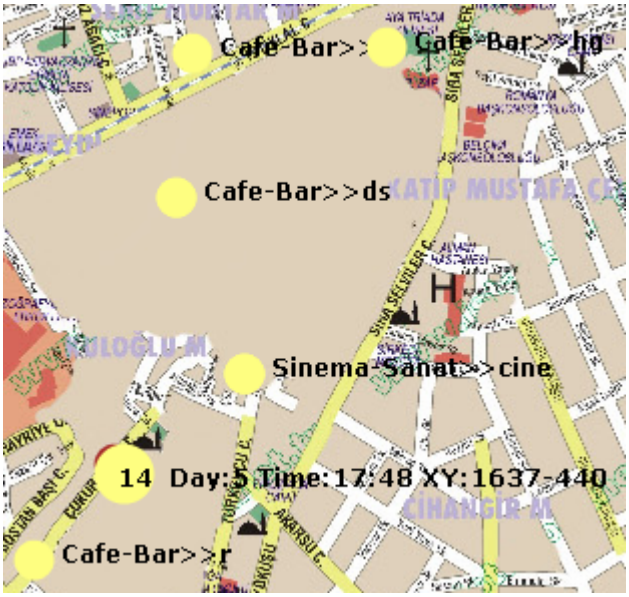

Figure 3.4 A cluster example generated by the simulator. On the figure big yellow point (14 Day: 5 Time:17:48 XY: 1637-440) represents a cluster centre. It means that actions become denser on those places marked with small yellow point, in the neighbourhood of cluster centre representing day: Saturday time: 17:48 XY: 1637-440.

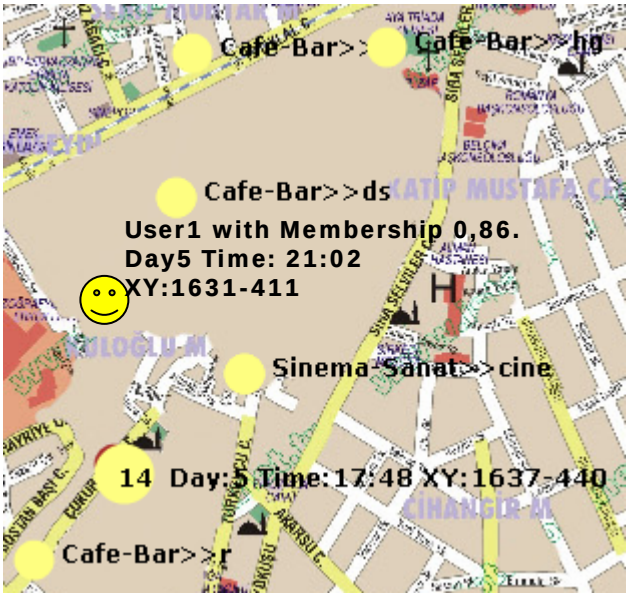

Figure 3.5 A cluster example generated by the simulator. On the figure User 1 placed on a point (Day:5 Time: 21:02 XY:1631-411) with 0.86 membership degree to cluster14. Small yellow points constitute possible alternatives for User 1.

Note that it is possible to produce recommendation instantly as mentioned above. But our aim is to produce recommendation beforehand. If it is possible to predict the user's current cluster in the context space way before its occurrence, the produced recommendations will be more valuable.

3.3 Learning Automaton for Prediction in the Context Space

In section 2.6 a detailed investigation is given on path prediction applications. Remember that mobile network is represented by cells and a user's movement is represented by transitions between cells on the plane. Similarly, the context space is represented by fixed nodes, that are clusters, and a user's motion can be represented by transitions between clusters in the context space. Thus, it is possible to adopt the same algorithms used in path prediction.

For prediction task "Stochastic Learning Automaton" used in [36] is adopted in our case. Stochastic Learning Automaton is finite state adaptive systems that interact continuously in an iterative fashion with the environment. As a result of FCM whole context space is represented by fixed points called cluster centres, and then movement of any user can be represented by transitions between cluster centres. The transitions between cluster centres are considered as a finite number of states mentioned above.

An automaton is a machine or control mechanism designed to automatically follow a predetermined sequence of operations. The term stochastic emphasizes the adaptive nature of the automaton we describe here. The automaton described here does not follow predetermined rules, but adapts to changes in its environment. This adaptation is the result of the learning process. Learning is defined as any permanent change in behavior as a result of past experience, and a learning system should therefore have the ability to improve its behavior with time, toward a final goal. In a purely mathematical context, the goal of a learning system is the optimization of a functional not known explicitly.

The stochastic automaton attempts a solution of the problem without any information on the optimal action (initially, equal probabilities are attached to all the actions). Initially, an input is provided to the automaton from the environment. This input triggers Roulette Wheel Selection, and Roulette Wheel selects one of a finite number of candidate responses, which are possible transitions, from the automaton. Once transition is selected, the response from the environment is observed, action probabilities are updated based on that response, and the procedure is repeated. Through a probabilistic, trial-and-error response process they learn to choose or adapt to a behaviour which generates the best response.

The simplest selection scheme is roulette-wheel selection, also called stochastic sampling with replacement [42]. This is a stochastic algorithm and involves the following technique. The individuals (candidate transitions) are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its probability. A random number is generated and the individual whose segment spans the random number is selected. The higher probability the individuals have, the more chances to be selected they have.

Stochastic Learning Automata are generally considered as robust but not being very efficient learners. They are relatively easy to implement. Generally, the operation of the learning automaton is based on a state transition matrix, which contains the one-step transition probabilities P_{ij} from the current state i to the next state j. Different approaches have been proposed for the updating of the state transition matrix after the reception of environment feedback. In this paper we adopt the behaviour of a Linear Reward-Penalty (L_{R-P}) Scheme. When the automaton selects the right response, the positive feedback received by the environment causes the respective state transition to be "rewarded" (i.e., its probability is increased by some pre-arranged step) while the probabilities of the state transitions that were not selected (remaining transitions from the same state) are "penalized" (decreased) uniformly to keep the probability sum to 1. If the proposed response is not appropriate (i.e., a negative feedback was received from

environment) a reverse approach is followed; the probability value of the selected transition is "penalized", while the remaining transitions are evenly rewarded to balance the decrease. This behaviour is shown in (Eq. 3.4 and Eq. 3.5):

Transition $(i \rightarrow j)$ received positive feedback:

$$P_{ij} = P_{ij} + w(1 - P_{ij}) \text{ and } P_{ik} = P_{ik}(1 - w), k \neq j \quad \text{(Eq. 3.4)}$$

Transition $(i \rightarrow j)$ received negative feedback:

$$P_{ij} = P_{ij} - w'(1 - P_{ij}) \text{ and } P_{ik} = P_{ik}(1 + w'), k \neq j \quad \text{(Eq. 3.5)}$$

In (Eq. 3.4 and Eq. 3.5), w denotes the rewarding step while w denotes the penalizing step. Those two values may be equal or different. Upon invocation, the automaton selects, as candidate future state, the state with the highest probability. After consecutive interactions with the environment, some state transitions will have probabilities close to 1 while others will have near-zero values (convergence).

Two learning automata are exploited. One called per-user automaton (PUA) uses only the per-user recorded information and other called global automaton (GA) uses the information of all users. Structural representation of PUA and GA are shown in Figure 3.6 and Figure 3.7 respectively.

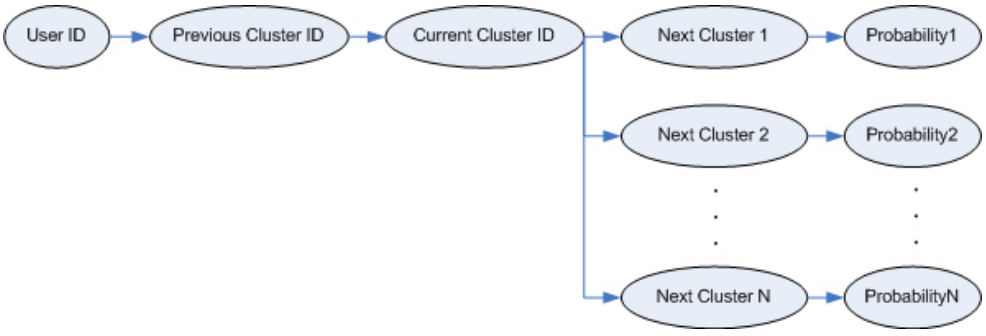

Figure 3.6 Structural representation of PUA.

Whenever a prediction is requested for User ID (User ID is not needed for GA), Previous Cluster ID and Current Cluster ID, probabilities for each next are provided. If

no appropriate state transitions were found then a new entry is fed into the database. If the automaton decision is correct (positive feedback) then the entry field that contributed to this decision is rewarded while the probability values of the remaining fields are reduced (penalized) as discussed.

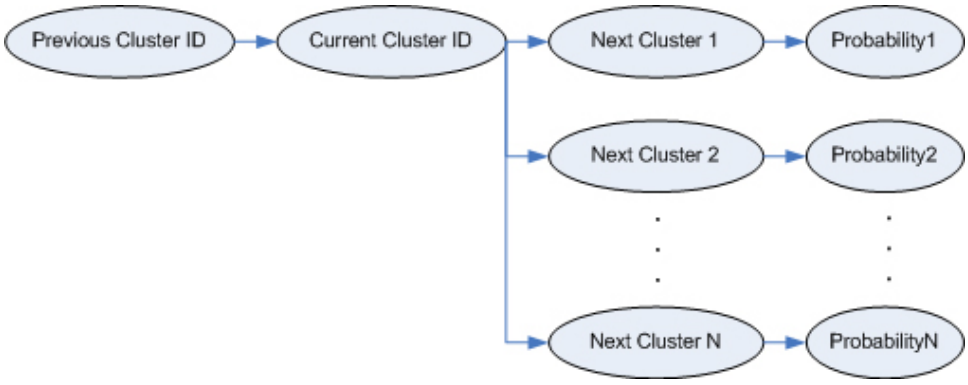

Figure 3.7 Structural representation of GA.

Two learning automata, the Global (GA) and the Per-User (PUA), operate simultaneously on top of two itinerary databases which have the layouts shown in Figure 3.6 and Figure 3.7 respectively, for some specific user. The set of entries in Figure 3.7, show the probabilities for specific cluster crossings (i.e., Previous Cluster - Current Cluster - Future Cluster), for all users covered by the system.

Block diagram of PUA and GA is presented in Figure 3.8. When a prediction is requested, for a specific user, both automata (per-user and global) are consulted. The adopted scheme of combining the outcomes of both automata is termed Weighted Automaton (WA). The WA practically, combines the per-cell probabilities returned by the two automata using predefined weights. (Eq. 3.6) shows the calculation performed by WA. The reward/penalize procedure is applied independently to the PUA and the GA. Figure 3.9 shows a demo scenario for prediction in the context space.

$$WA(previous, current, next, userID) = a.GA(previous, current, next) + (1-a).PUA(previous, current, next, userID), \quad \text{(Eq. 3.6)}$$

where a is a weight.

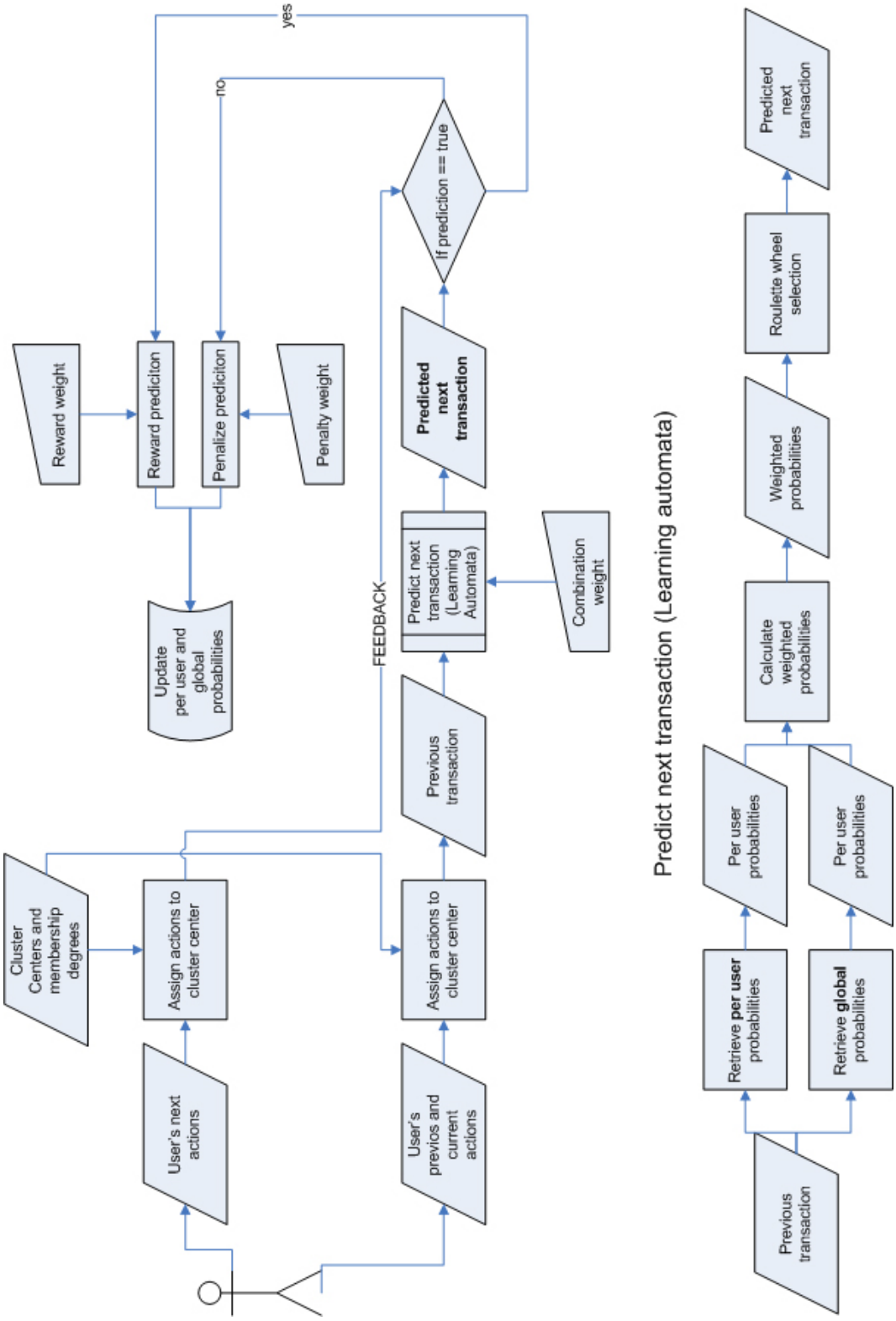

Figure 3.8 Flow chart of PUA and GA

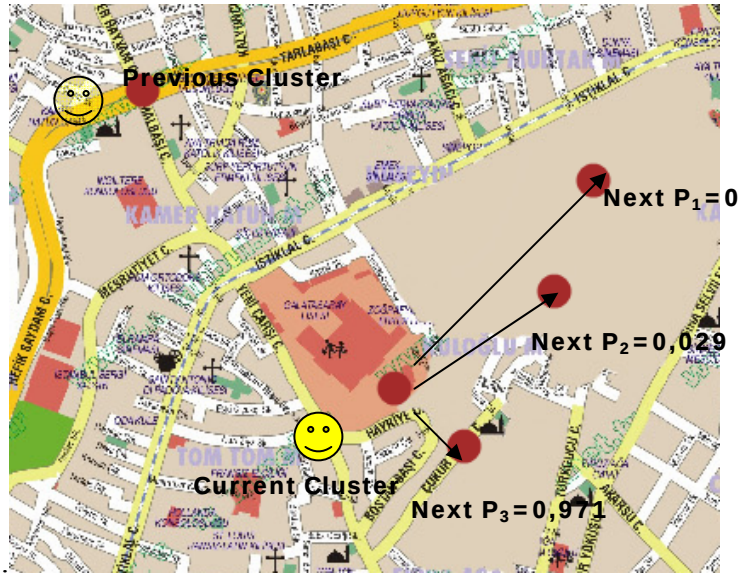

Figure 3.9 Prediction in the context space.

Firstly the clusters have been constructed and the context space has been represented by the clusters. As a second step learning automaton algorithm has run and prediction of users transition has been obtained. Now the remaining step is to implement a user profiling method.

3.4 Unsupervised Fuzzy Divisive Hierarchical Clustering for User Profiling

Throughout Sections 3.1 and 3.2, we have dealt with context-awareness and motion prediction. Consider a user whose next status (context) has been predicted accurately. Context-awareness selects the set of actions that fits the user next context. But not all of actions included in this set satisfy the user's interest. A subset of actions satisfying the user interest should be derived from this set through the user profiling. Generally, user profiling can be defined as learning of user preference or deduction of users' profiles from historical data about the users' behaviours.

The approach proposed in [19] will be adopted to build users' profiles. The proposed approach is called as Unsupervised Fuzzy Divisive Hierarchical Clustering (UFDHC). In that mentioned reference, authors apply this method on a web site consisting of many ontological categories. Each user is represented by the number of access to each category. Access log data are input to UFDHC. The output of UFDHC is a binary tree of user groups characterized by a set of common interests and represented by a prototype, which corresponds to the profile of the users of the group. We use the same approach for profiling. Similarly, the access logs to each category are stored for each user, and they are used as input. The output is the binary tree of the users' profiles (or user clusters) representing the interest degree to each category in the mobile domain. Block diagram of UFDHC is represented in Figure 3.10.

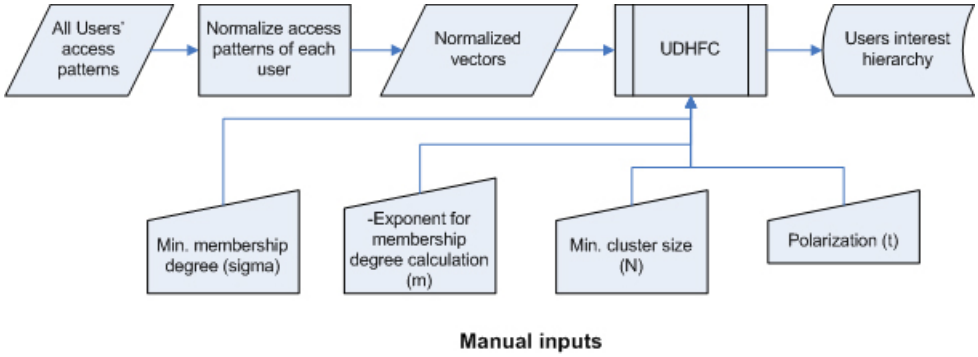

Figure 3.10 Flow chart of UFDHC

Let $X = [\underline{x}_k]$ be the vector of N users. Each user represented as a vector $\underline{x}_k = [x_{k,1}, ..., x_{k,M}]$ in the space $\Re^M$ of the M categories. The coordinates of each vector correspond to the number of access to each category. In [19], they observe that, in the web oriented profiling perspective, the behaviour of a user is more accurately described by the relative orientation of the vector rather than its magnitude. Indeed, two users who access the same topics with the same relative percentage, though a different number of times, can be considered as samples of same behavioural profile. This observation leads them to state that the more similar two users, the higher the value of the cosine of the angle α formed by the corresponding vectors is. Since the coordinates of $x_{k,j}$ of each

vector $\underline{x}_k$ vary on positive values, the cosine can assume only values in $[0,1]$. Thus, they define dissimilarity $d^c(\underline{x}_i, \underline{x}_j)$ between two users $\underline{x}_i$ and $\underline{x}_j$ as

$$d^c(\underline{x}_i, \underline{x}_j) = \sqrt{1 - \cos(\alpha)}, \qquad \text{(Eq. 3.7)}$$

where $d^c(\underline{x}_i, \underline{x}_j)$ is the cosine distance, and

$$\cos(\alpha) = \frac{\underline{x}_k . \underline{x}_j}{\left\| \underline{x}_k \right\|_2 . \left\| \underline{x}_j \right\|_2}, \text{ with } \|.\|_2 \text{ the Euclidean norm,} \qquad \text{(Eq. 3.8)}$$

is the cosine of angle formed by $\underline{x}_i$ and $\underline{x}_j$. To speed up the computation, the users' access pattern vectors are normalized.

The UFDHC method is a fuzzy divisive hierarchical method, that is, it determines a structure in the data in the form of a binary tree by firstly detecting large clusters and then splitting them as much as necessary to generate a final objective classification. A decomposition criterion ensures that only real clusters are included in the hierarchy and consequently, determines the optimal number of clusters in data.

Let A be a fuzzy cluster over dataset $X = [\underline{x}_k]$ and $P = \{A_1, A_2\}$ a binary fuzzy partition of A. It is assumed that P describes real clusters if the following conditions are satisfied.

Firstly, there exist at least $\overline{N}$ users (unlike only 1 in the original version of UFDHC), with $\overline{N}$ generally much smaller than the total number of users, with membership degree to either A_1 and A_2 higher than a prefixed threshold σ, with $\sigma > 0.5$ (unlike equal to 0.5 in original version of UFDHC).

Secondly, if the polarization degree $R(P)$ is larger than t, where $t \in [0.5,1[$ is an appropriate threshold $A(\underline{x}_k)$, $A_1(\underline{x}_k)$ and $A_2(\underline{x}_k)$ are memberships to fuzzy clusters A, A_1 and A_2, respectively.

$$R(P) = \frac{\sum_{\underline{x} \in X} \max(A_1(\underline{x}_k), A_2(\underline{x}_k))}{\sum_{\underline{x} \in X} A(\underline{x}_k)} \qquad \text{(Eq. 3.9)}$$

$R(P)$ is associated with the presence of cluster structure in the data. The higher polarization degree, the better the quality of the partition is. The extreme degrees $R(P) = 0.5$ and $R(P) = 1$ correspond to no structure in A and complete separation between the two classes A_1 and A_2, respectively.

The value of threshold t is a critical application dependent parameter. The value of t must be fixed taking into account the degree of fuzziness of the dataset being considered. High values of t allow us to discover cluster structure in (nearly) crisp datasets. On the contrary, low values of t allow us to correctly partition (more or less) fuzzy datasets. In particular, too high value of t may not identify any real cluster structure with a high fuzziness degree; whereas a too low value of t may produce false partitions (i.e. not corresponding to a real clusters).

To determine binary partition $P = \{A_1, A_2\}$ of a given set A at each level of hierarchy, the modified UFDHC algorithm uses generalised version, denoted Generalized Fuzzy C-Means (GFCM) algorithm, of the well-known Fuzzy C-Means (FCM) proposed by Bezdek [39]. In GFCM, the constraint

$$\sum_{i=1}^{C} A_i(\underline{x}_k) = 1 \qquad \text{(Eq. 3.10)}$$

where C is number of clusters, imposed by FCM is replaced by

$$\sum_{i=1}^{C} A_i(\underline{x}_k) = A(\underline{x}_k), \text{ with } 0 \le A(\underline{x}_k) \le 1. \qquad \text{(Eq. 3.11)}$$

In order to obtain the fuzzy partition P of fuzzy class A we assume that fuzzy class A_i is represented by a prototype $\underline{v}_i$, which, in our case, is a point on the hypersphere with unitary radius in $\Re^M$. Dissimilarity $D(\underline{x}_k, \underline{v}_i)$ between the point $\underline{x}_k$ and prototype $\underline{v}_i$ is defined as the squared cosine distance with respect to the fuzzy class A_i as follows:

$$\begin{aligned} D(\underline{x}_k, \underline{v}_i) &= \min(A_i(\underline{x}_k), A_i(\underline{v}_k))^m . d^c(\underline{x}_k, \underline{v}_k)^2 \\ &= (A_i(\underline{x}_k))^m . d^c(\underline{x}_k, \underline{v}_k)^2 \end{aligned} \text{ with } m \in [1, \infty). \qquad \text{(Eq. 3.12)}$$

The error $J(P,V)$ made in representing the fuzzy partition P by the set $V = [\underline{v}_1, ..., \underline{v}_C]$ is defined as

$$J(P,V) = \sum_{k=1}^{N} \sum_{i=1}^{C} D(\underline{x}_k, \underline{v}_i), \qquad \text{(Eq. 3.13)}$$

where N is the optimal number of points. The optimal fuzzy partition of A and its representation V are obtained as a minimum of criterion function $J(P,V)$. Therefore, an iterative method based on minimization of the functions $J(P,\cdot)$ and $J(\cdot,V)$ is used. To minimize $J(P,\cdot)$, Lagrange multiplier method is applied with the constraint denoted by (Eq. 3.11), and the following formula is obtained:

$$A_i(\underline{x}_k) = \frac{A(\underline{x}_k)}{\sum_{j=1}^{C} \left(\frac{d^c(\underline{x}_k, \underline{v}_i)}{d^c(\underline{x}_k, \underline{v}_j)} \right)^{\frac{2}{m-1}}} \qquad \text{(Eq. 3.14)}$$

To minimize $J(\cdot,V)$, Lagrange multiplier method is applied again with the following constraint,

$$\sum_{f=1}^{M} v_{i,f}^2 = 1 \qquad \text{(Eq. 3.15)}$$

and the following formula is obtained:

$$v_{if} = \frac{\sum_{k=1}^{N} (A(\underline{x}_k))^m \underline{x}_{kf}}{\sqrt{\sum_{t=1}^{M} \left(\sum_{k=1}^{N} (A(\underline{x}_k))^m \underline{x}_{kt} \right)^2}} \qquad \text{(Eq. 3.16)}$$

The iterative process starts with an arbitrary partition of P, then (Eq. 3.15) and (Eq. 3.16) are used iteratively. The process stops when two successive partitions are close enough. It can be noted that if $A = X$ then GFCM becomes the FCM.

The GFCM algorithm starts from the computation of an arbitrary fuzzy partition $P = \{A_1, A_2\}$ of the dataset X. If the partition P does not satisfies the two conditions 1. and 2., the two fuzzy sets A_1 and A_2 do not describe real clusters. It means that there is no cluster structure in X and the data, therefore, comprise a single cluster. The process then ends. Otherwise, a binary partition for each of the two cluster of P is computed. If the binary partition of a cluster does not represent real sub-clusters then the cluster is marked to avoid subsequent attempts to split it. When no partition can be generated, the procedure terminates.

The choice of the value of σ on the membership values and N on the number of users in first condition, and the choice of threshold t on the polarization degree used in second condition can not leave the value of fuzzification constant m out of consideration. Constant m determines the degree of fuzziness of the partition: the closer m is to 1, the

crisper the partition is. Indeed, if $m \to 1$, the membership degrees of any pattern to any cluster tend to be either 0 or 1. On the other hand, if $m \to \infty$ the membership degrees of any pattern to any cluster tent to be equal to $1/C$, thus producing highest level of fuzziness. Thus, if the GFCM with m close to 1 is executed, very crisp partition characterized by a high polarization degree is expected. In such case, to avoid the generation of too high number of partitions, threshold t and σ, and $\bar{N}$ can not be chosen too low.

Once the binary tree is produced, any user represented by $\underline{x}_k$ can be classified using the following method. The current node, and the left and the right children of current node are represented as *CN*, *LN* and *RN*, respectively. Let A, A_l and A_r be the clusters contained, respectively, in *CN*, *LN* and *RN*.

1. Fix a real value $\eta \in [0.5,1]$. Initialize *CN* to the root of the binary tree.
2. If there exist the left child node *LN* and right child node *RN* of the current node CN, then compute the membership degrees $A_l(\underline{x}_k)$ and $A_r(\underline{x}_k)$ of the user $\underline{x}_k$ to the two cluster using (Eq. 3.15); otherwise classify user $\underline{x}_k$ as belonging to A and terminate.
3. If $A_l(\underline{x}_k)$ is larger than η, then assign *LN* to *CN* and go to step 2;
4. If $A_r(\underline{x}_k)$ is larger than η, then assign *RN* to *CN* and go to step 2;
5. If neither $A_l(\underline{x}_k)$ nor $A_r(\underline{x}_k)$ are larger than η, then classify user $\underline{x}_k$ as belonging to A and terminate. If the current node is root, then user $\underline{x}_k$ is judged to be not classifiable.

Each cluster is identified by a profile which summarizes the ontological preferences of the users strongly belonging to the cluster. In general, in each profile, only a restricted of categories has a relevant number of accesses. This set constitutes the fingerprint of the profile and identifies the interests shared by the members strongly belonging to the cluster. To highlight the fingerprint, the profile is presented as follows: first, $\underline{v}_i$ is

transformed into $\underline{\tilde{v}}_i = \left[100 \cdot v_{i1}^2, ..., 100 \cdot v_{iM}^2\right]$. Each component $\underline{\tilde{v}}_{ij}$ represents the percentage of contribution of category j on the user preference. Consequently, in recommendation, the actions included in category with higher contribution are taken into account.

The implementation medium of the proposed architecture for the recommendation system will be introduced in Chapter 4.

CHAPTER 4

DESIGN AND IMPLEMENTATION

In this chapter, the design and implementation of the approach proposed in Chapter 3 is introduced with all its practical details. The component diagram of simulator is shown in Figure 4.1.

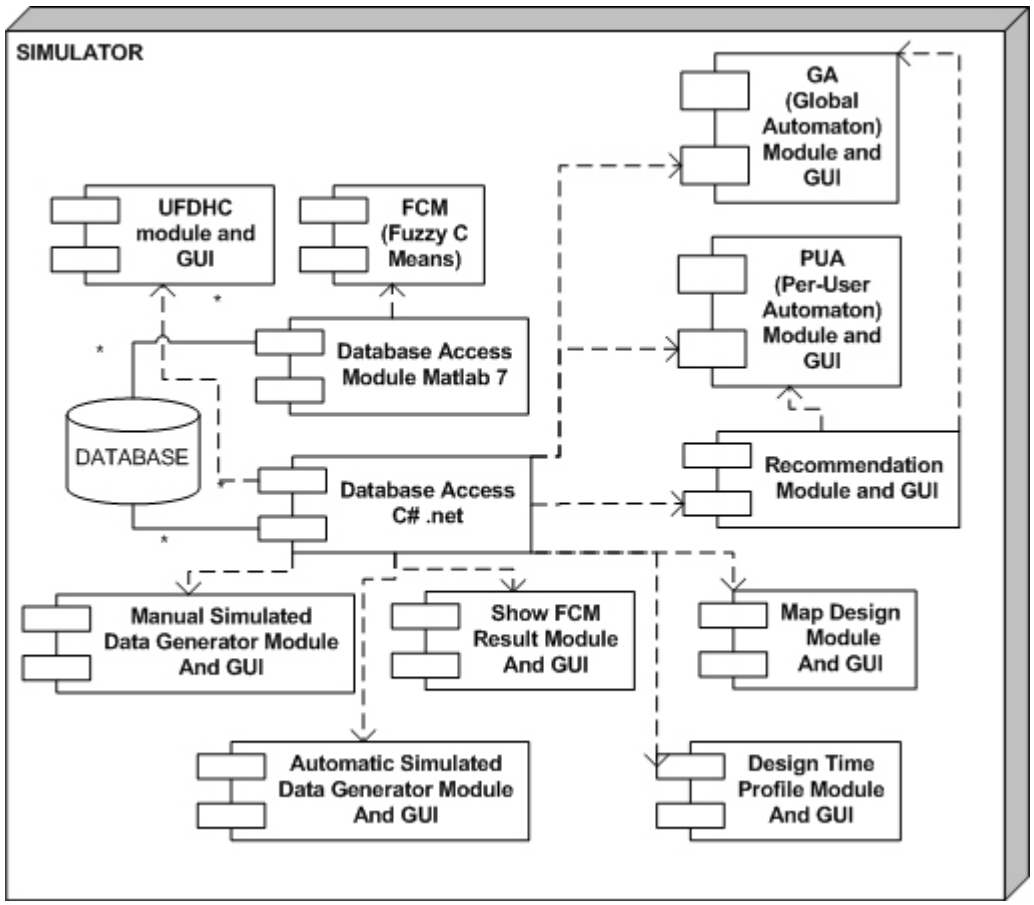

Figure 4.1 Component diagram

In Figure 4.1 all modules except FCM and Database Access Matlab 7 modules were developed with C# .net. FCM and Database Access Matlab 7 modules were developed

with Matlab 7.0 Release 14. Two database access modules, which are Database Access C# .net and Database Access Matlab 7.0, exist. These modules are used by others to communicate with database. The remaining modules represent the business logic layer and respective Graphical User Interface (GUI). The details of each component are given in following sections.

4.1 Database

The simulator works on a database which is a MS SQL Server 2000. All data produced and used are stored on a single database named USERS_ACTIONS_DB. The diagram of the database is provided in Figure 4.2. The task of each table is defined as follows.

T_Clustering_Result, T_Cluster_Centers, T_Cluster_Action_Membership and T_Cluster_Action tables are used by FCM to store the outputs. The T_Clustering_Result table stores a result id for output set from each run of FCM module. The points representing the cluster centres are stored in the T_Cluster_Centers table. The T_Cluster_Action_Membership stores the membership degrees for action histories. The T_Cluster_Action stores members, which are actions, of each cluster.

T_Users, T_Action_History, T_Action and T_Category tables store data related to actions and users. All users are stored in the T_Users table. The T_Action_History table stores action histories of each user. The actions placed on the map are stored in the T_Action table. All categories defining the actions are stored in the T_Category table.

T_Peruser_Probability and T_Global_Probability tables are used by per user and global automaton to store per user and global probabilities respectively.

T_Time_Point, T_Time_Profile and T_Time_Point_Profile tables store biases used by automatic simulated data generator. The details of automatic simulated data generator are given in section 5.1. The points defined on time dimension are stored in the

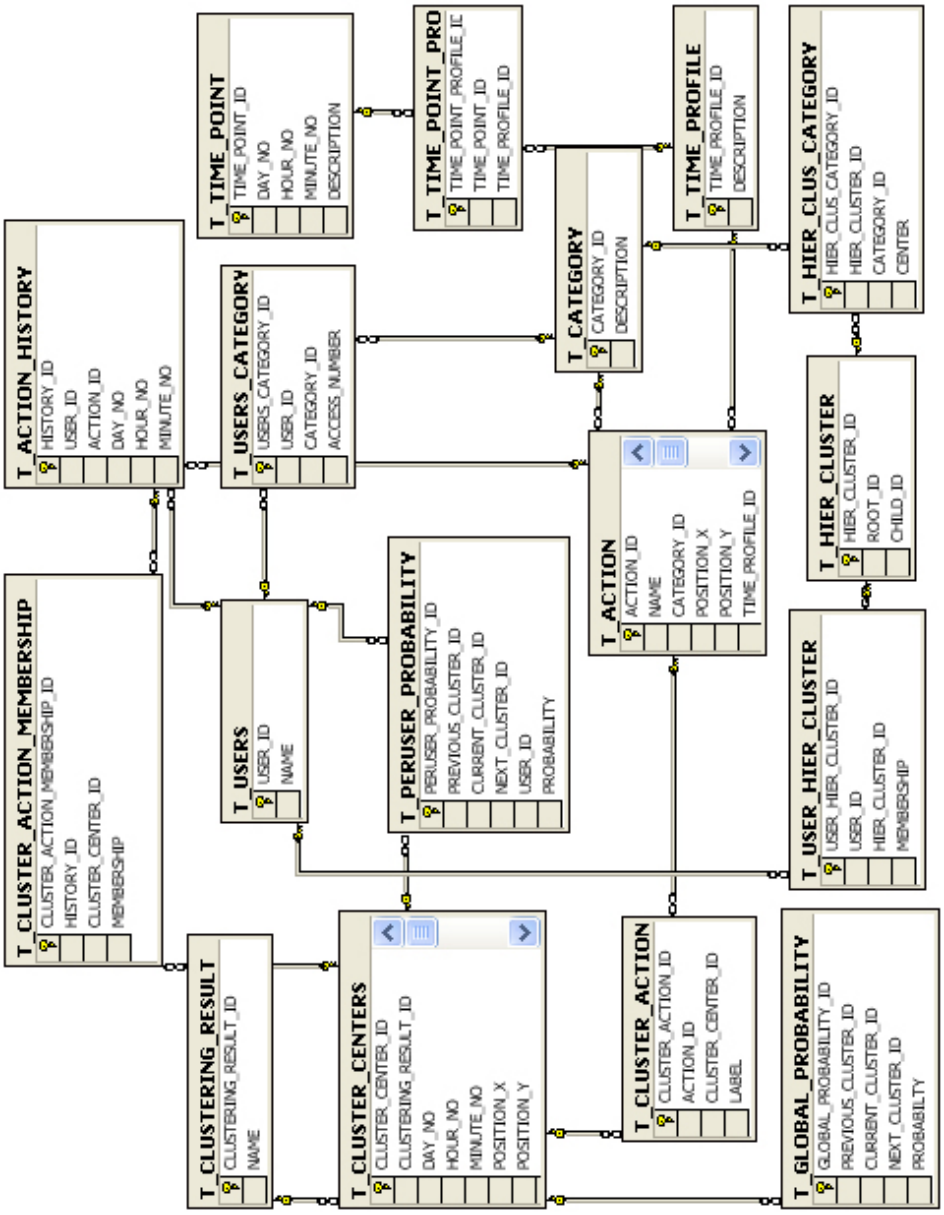

Figure 4.2 Database diagram

T_Time_Point table. The time profiles defined for each action are stored in the T_Time_Profiles table. The time points defining each time profile are stored in the T_Time_Point_Profile table.

T_Users_Category, T_User_Hier_Cluster, T_Hier_Cluster and T_Hier_Clus_Category tables store input and output of UFDHC. For each user, the T_Users_Category table

stores the number of access to each category, which is input to UFDHC. The T_User_Hier_Cluster table stores the members of each cluster produced by UFDHC. Each row of the T_Hier_Cluster defines a cluster and its place in the hierarchical tree. The contribution degree of each category on each cluster profile is stored in T_Hier_Clus_Category table.

4.2 Map Design Module and GUI

Map design interface, a windows form based user friendly GUI, was developed with C# .net. GUI adopts a map as a layout. It is the map of the Bosphorus gaining the sides of Eminönü, Fatih and Beyoğlu of Istanbul. The screen shot of GUI is provided in Figure 4.3.

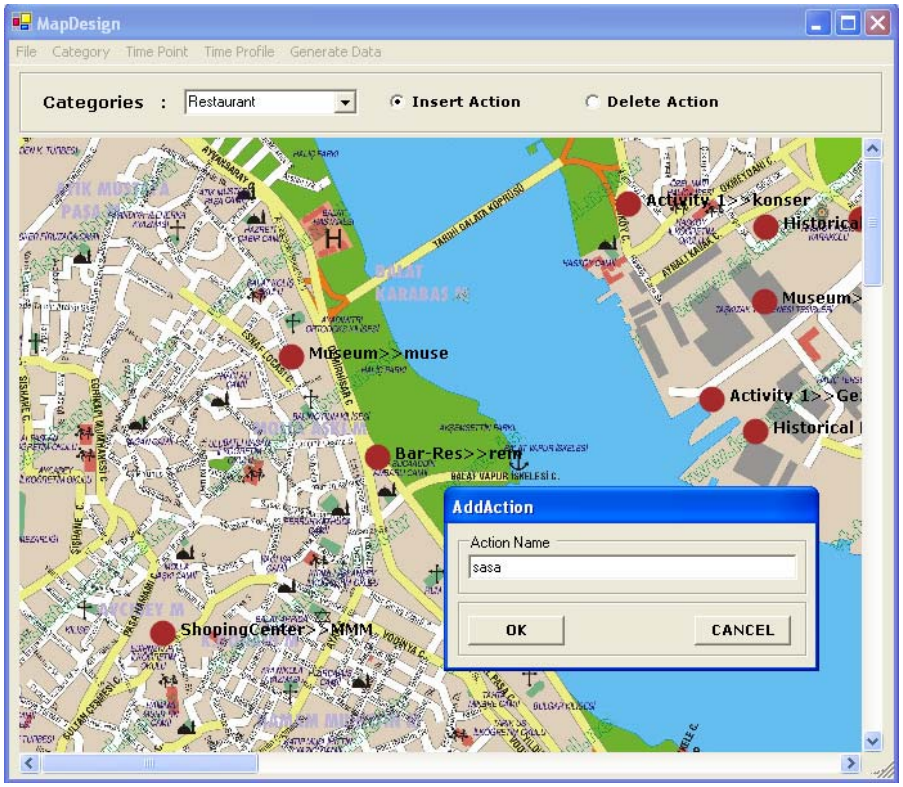

Figure 4.3 Screen Shot of Map Design Graphical User Interface

The functions of GUI are as follows:

- It allows to design of action (places) on map dynamically (It is possible to rearrange the place located on map by adding and deleting functionalities).
- It allows creating new action categories (also deleting and updating functionalities are provided).

4.3 Manual Simulated Data Generator Module and GUI

Simulated data generator interface, a windows form based user friendly GUI, was developed with C# .net. GUI adopts a map as a layout and shows the action designed with map design interface. The screen shot of GUI is provided in Figure 4.4.

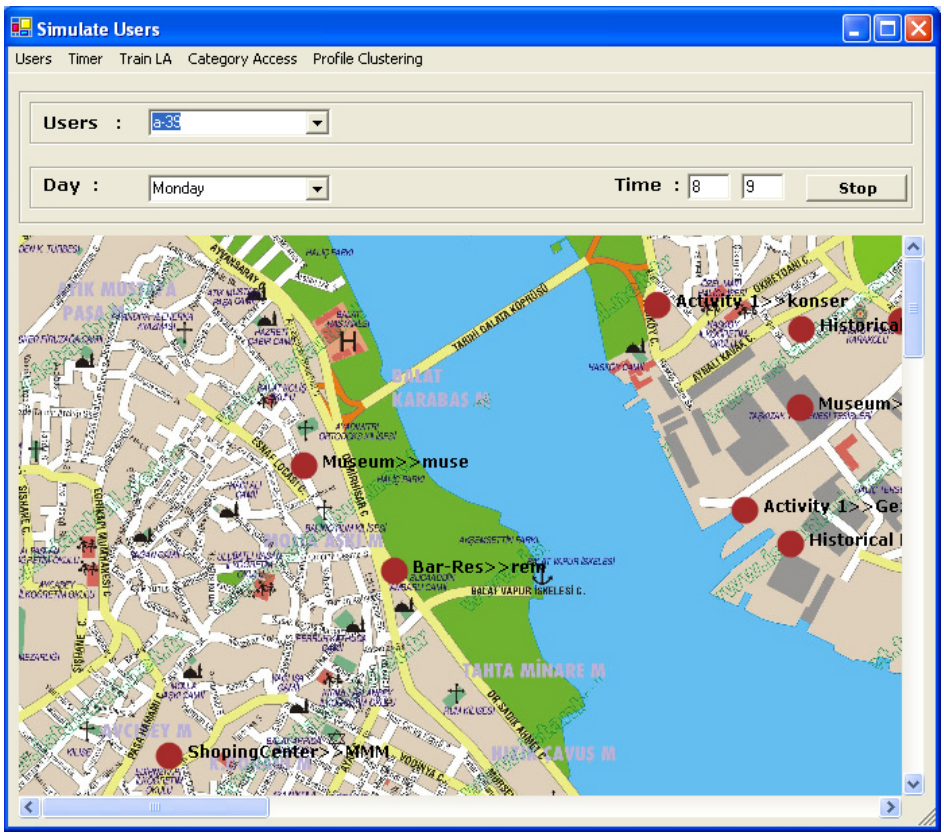

Figure 4.4 Screen Shot of Manual Simulated Data Generator Graphical User Interface

The functions of GUI are as follows:

- It allows user to manually produce synthetic data by clicking actions on the map and stores them into database.
- It allows adding a new simulated user and deleting old users.
- Also GUI of learning automaton train was placed on this GUI.

4.4 Automatic Simulated Data Generator Module and GUI

Automatic simulated data generator interface, a windows form based user friendly GUI, was developed with C# .net. The screen shot of GUI is provided in Figure 4.5. The inputs of this module are:

- Variance: determines the size of neighbourhood of time point.
- User count: the number of users selected randomly by generator to produce simulated data on behalf of them.

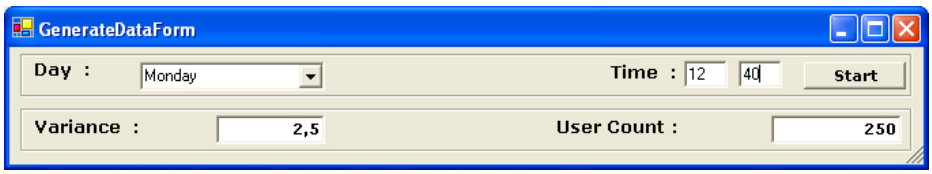

Figure 4.5 Screen Shot of Automatic Simulated Data Generator Graphical User Interface

4.5 FCM

FCM (fuzzy c means) module was developed with Matlab 7.0 Release 14. FCM function of Fuzzy Logic Toolbox was used. It gets the all users' action from database, produces C and U, and then stores them into database. The inputs of this module are:

- Data set to be clustered; each row is a sample data point(user action)
- Number of clusters (greater than one)

The outputs of this function are:

- Matrix of final cluster (in section 3.1 it is called C) centres where each row provides the centre coordinates
- Final fuzzy partition matrix (or membership function matrix, in section 3.1 it is called U)
- Values of the objective function during iterations

Function uses an additional argument variable, options, to control clustering parameters, introduce a stopping criterion, and/or set the iteration information display.

- options(1): exponent for the partition matrix U (default: 2.0)
- options(2): maximum number of iterations (default: 100)
- options(3): minimum amount of improvement (default: 1e-5)
- options(4): info display during iteration (default: 1)

4.6 GA and PUA Modules and GUI

GA (Global Automaton) and PUA (Per-User Automaton) modules were developed with C# .net. They gets cluster centres produced by FCM from database, and assign them to the users' actions to construct the user transitions between cluster centres. Then train the GA and PUA. The screen shot of GUI is provided in Figure 4.6. The inputs of this module are:

- Penalty weight which determines contribution degree of the negative feedback received from the user.

- Reward weight which determines contribution degree of the positive feedback received from the user.
- PUA and GA weight which determines contribution degrees of per user probabilities and global probabilities in prediction.
- Cluster centre set which represents the context space.

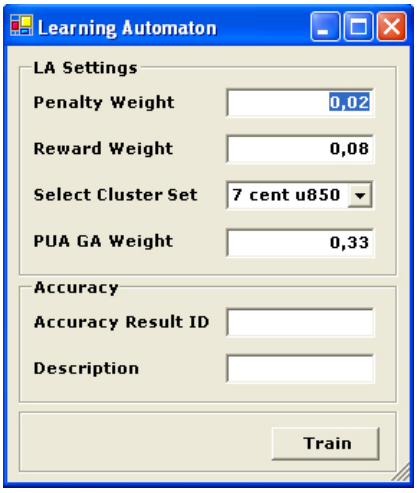

Figure 4.6 Screen Shot of PUA and GA Graphical User Interface

The outputs of this function are probability values assigned to each possible transition between the cluster centres.

4.7 Recommendation Module and GUI

This GUI, a windows form based user friendly GUI, was developed with C# .net. It was designed to see visually and instantly whether developed modules work correctly. Two windows forms are presented to user. One (Figure 4.7) allows the user to select a cluster centre, and the other (Figure 4.8) allows the user to see recommended actions and select an action related to selected cluster.

On the lower left-hand side of Figure 4.7, we see four labels which represent previous cluster id, current cluster id, predicted cluster id and selected cluster id respectively. The

previous and current cluster id represent the user's previous transition and the coordinates of those cluster centres. The predicted cluster id shows the next cluster id predicted by LA according to the previous and current cluster id. Also, the coordinates of this cluster id and its transition from the current cluster id to the predicted cluster id probabilities for PUA and GA are given. The selected cluster id shows the next cluster selected by the user. Also, the coordinates of this cluster id and its transition

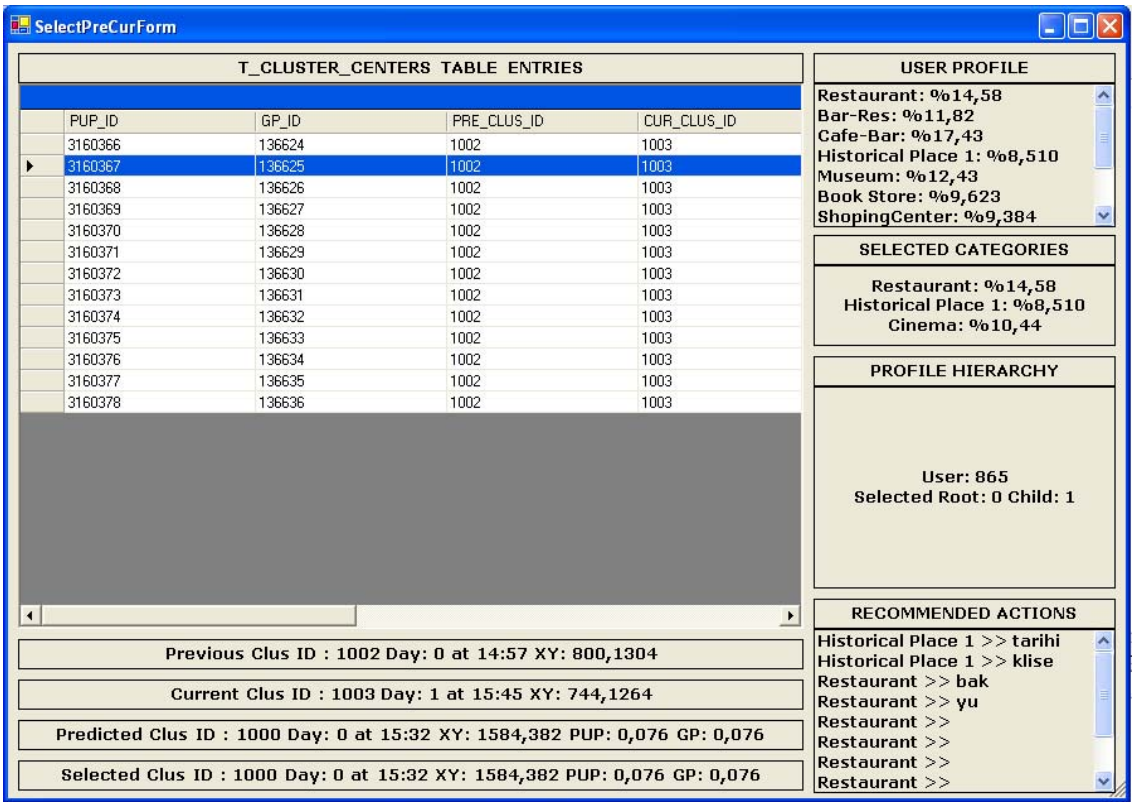

Figure 4.7 Screen Shot of Recommendation Graphical User Interface1

probabilities from the current cluster id to the selected cluster id for PUA and GA are given.

On the upper left-hand side of Figure 4.7, we see a data grid control called T_Cluster_Centers Table Entries. It allows the user to select his/her next cluster id, which is next transition, via clicking on the rows manually. According to his/her selection, the label showing the selected cluster id changes its status automatically.

On the right-hand side of Figure 4.7, we see four labels which represent user profile, selected categories, profile hierarchy and recommended actions respectively. The user profile label shows the profile of the cluster, which the user is a member of, which is produced by UFDHC. Each row of the profile shows the percentage of contribution of each category on the user preference. The selected categories label represents the categories selected by Roulette Wheel Selection algorithm according to the percentage of contribution of each category. The profile hierarchy label represents the user id and the layer in the hierarchical tree. For example, the root: 0 child 1 defines that this profile is a child of root cluster. The recommended action label represents the recommended actions which is included in one of the categories given in selected categories label. According to the user's selection in Figure 4.7, the recommendations to the user are shown graphically in Figure 4.8.

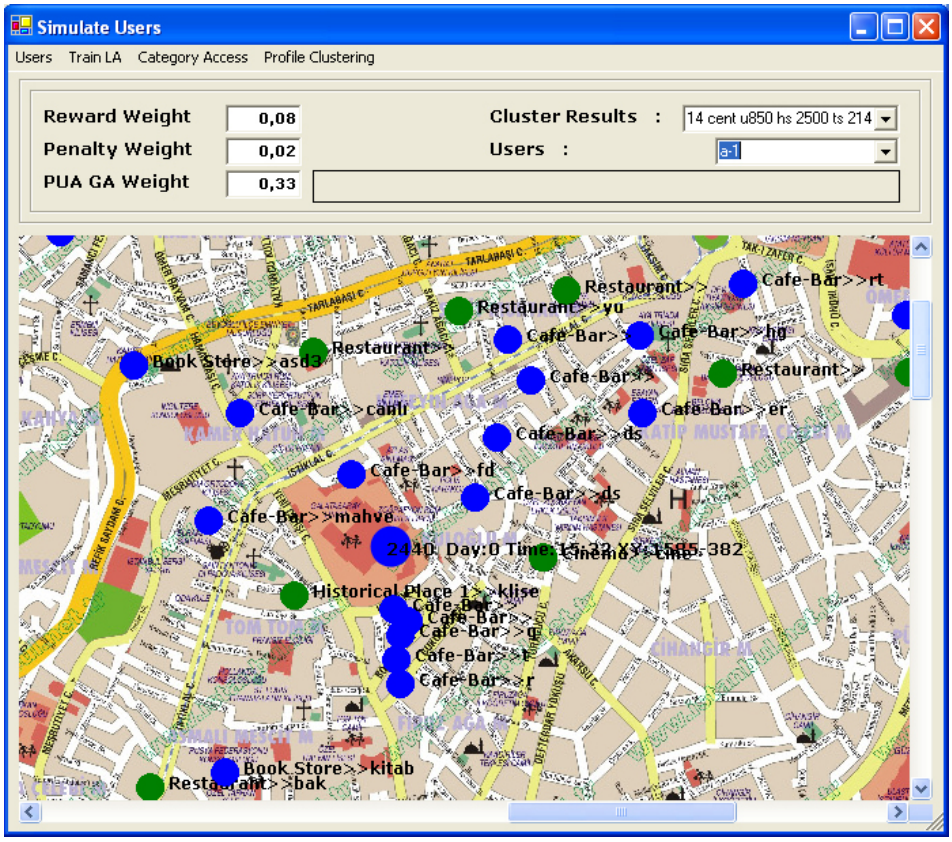

Figure 4.8 Screen Shot of Recommendation Graphical User Interface2

The points represent the action included in the selected cluster. But only the green points are the recommended actions matching the user profile. When the user clicks on one of the actions defined by the points on the map, that action is implemented by that user. Thereupon second click on the map, the GUI given in Figure 4.7 is presented to the user again. The inputs of this module are:

- Penalty weight which determines contribution degree of the negative feedback received from the user.
- Reward weight which determines contribution degree of the positive feedback received from the user.
- PUA and GA weight which determines contribution degrees of per user probabilities and global probabilities in prediction.
- Cluster results which represent the context space.
- Users who run the demo.

4.8 UFDHC Module and GUI

UFDHC module was developed with C# .net. It produces binary tree of the users' profile. A GUI (Figure 4.9) allows setting the manual inputs.

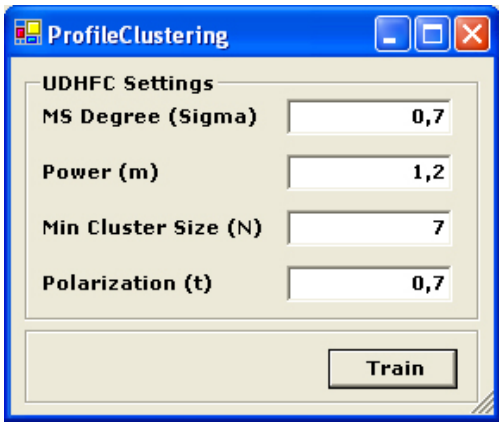

Figure 4.9 Screen Shot of UFDHC Graphical User Interface

The inputs of this module are:

- MS Degree (sigma): a prefixed threshold σ with $\sigma > 0.5$ restricting the number of the user included in each cluster of the binary tree produced by UFDHC.
- Power (m): m determines the degree of fuzziness of the binary partition.
- Min Clus Size (N): at least $\bar{N}$ users have to exist in each cluster.
- Polarization (t): a prefixed threshold with $t \in [0.5,1[$. The polarization degree of each binary partition in the tree have to be larger than binary t.

The output of this module is a binary tree of the profiles.

4.9 Show FCM Result Module and GUI

This GUI, a windows form based user friendly GUI, was developed with C# .net. It allows the seeing the results of FCM on a map visually. The screen shot of the GUI is presented in Figure 4.10. The inputs of this module are:

- Results: Cluster result set which represents the context space.
- Centers: Cluster centres of the selected cluster result set.
- Show Cluster option: if this option is checked, the cluster centre selected from the Centers list box will be shown on the upper left-hand side of the map. If the user clicks on map upon appearance of the cluster centre, a colour dialog box is provided. All actions belonging to the cluster centre are highlighted with selected colour from colour dialog box.
- Show Member option: if this option is checked, according to point's coordinates defined by user click on the map and time defined by timer, the membership degrees of that point to each cluster centre will be listed in Membership to each cluster list box.
- Timer: defines the time flow while the user is clicking on the map.

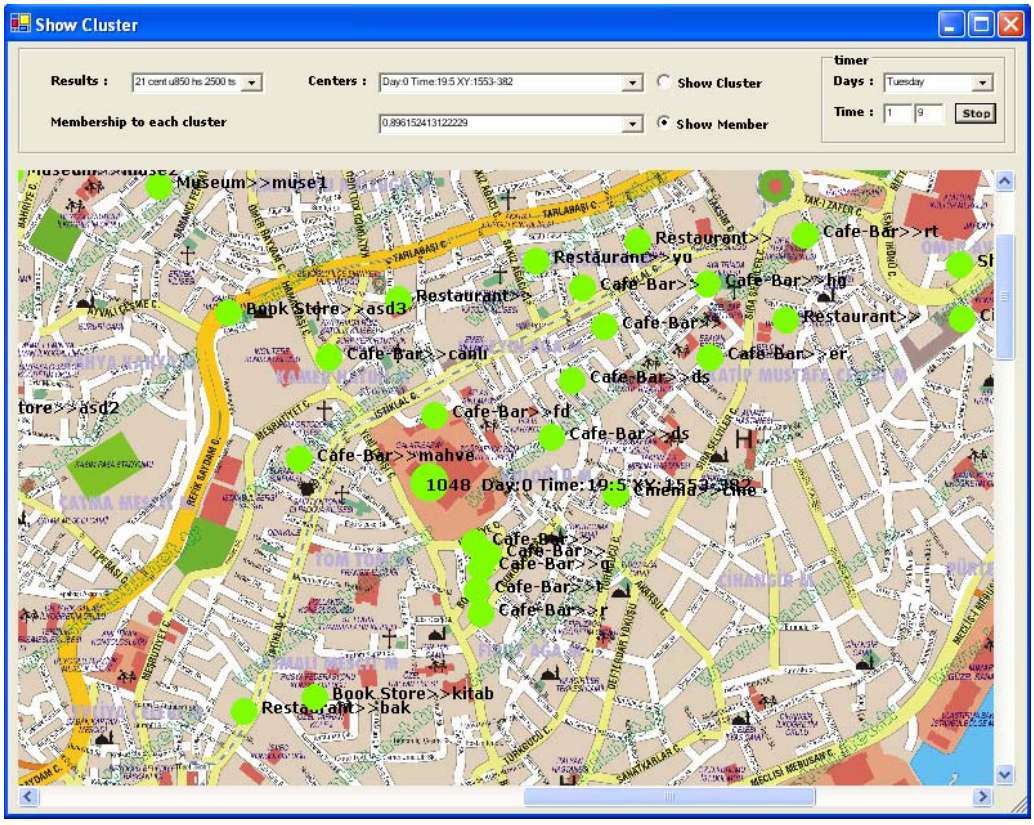

Figure 4.10 Screen Shot of Show FCM Results Graphical User Interface

4.10 Design Time Profiles Modules and GUI

This GUI, a windows form based user friendly GUI, was developed with C# .net. It allows the assign biases (designing time profiles) to action for generation of simulated data. The screen shot of the GUI is presented in Figure 4.11.

The GUI provides four data grid controls, which are T_Time_Point Table Entries, T_Time_Profile Table Entries, T_Time_Profile_Point Table Entries and T_Action Table Entries. T_Time_Point Table Entries shows the time points stored in T_Time_Point table in the database. The Add and Delete buttons given on the left-hand side allow the user adding and deleting rows.

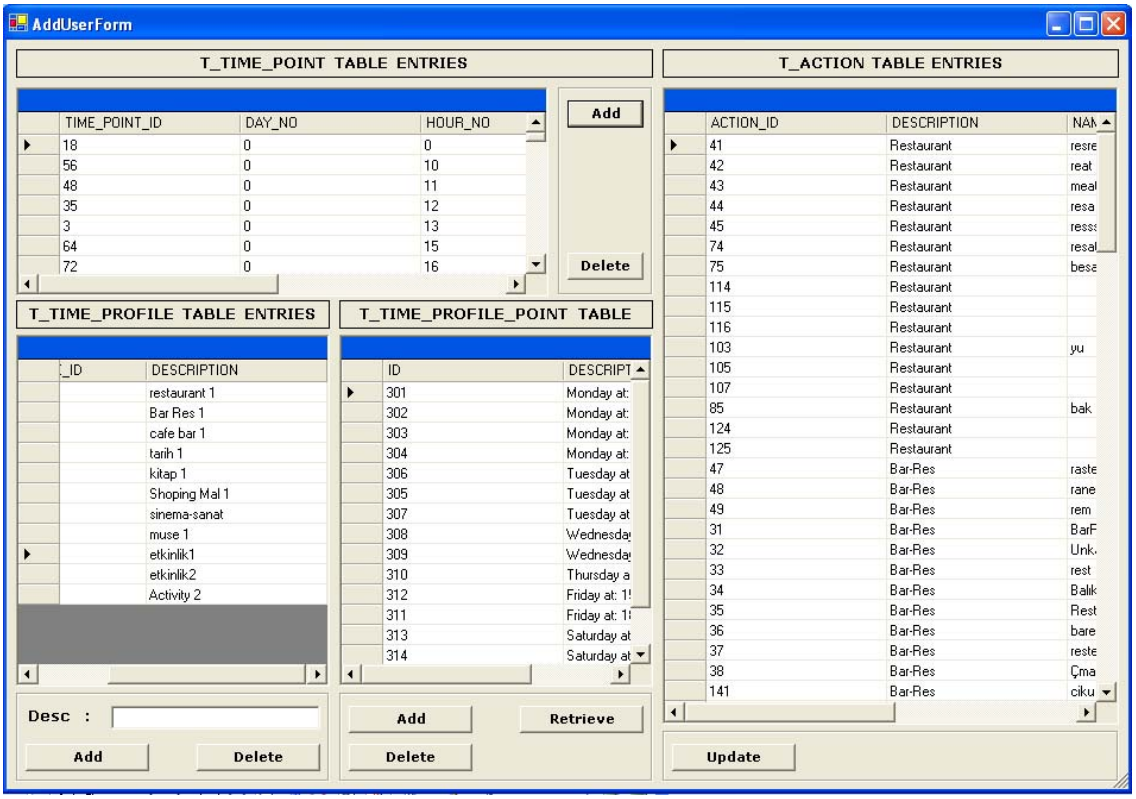

Figure 4.11 Screen Shot of Design Time Profile Graphical User Interface

T_Time_Profile Table Entries shows the time profiles stored in T_Time_Profile table. The Add button allows the user to define a new profile. The Delete button allows the user to delete a selected profile.

T_Time_Profile_Point Table Entries shows the points assigned to the time profile from T_Time_Profile Table Entries data grid, thereupon clicking the Retrieve button. Any time point from T_Time_Point Table Entries can be assigned to any time profile from T_Time_Profile Table Entries by clicking Add button. On other hand, a time point assigned to any time point can be deleted by clicking Delete button.

T_Action Table Entries lists all action stored in T_Action table. The time profiles of each action can be updated to a selected time profile clicking on Update button.

4.11 Database Access Module .net and Database Access Module Matlab 7

Database Access Module .net, a Windows DLL, was developed with C# .net. PUA, GA, Simulated Data Generator Interface, Map Design Interface and Test Interface use this DLL for any database operation (update, delete, insert and select). Data Access Module Matlab 7, consisting of several m-file, was developed with Matlab 7.0 Release 14. FCM module uses this module for any database operation (update, delete, insert and select).

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Simulated Data Generator

To test the proposed recommendation architecture generated in this thesis, we use simulated data since any real data is not available. A simulated data generator module was developed to produce the test dataset. Two options are provided to generate data: one is that the data can be produced manually via clicking on a visual map, and the latter is that the data can be generated automatically with some predefined biases and randomness. We direct the reader to play with the demo copy of our architecture, provided in the enclosed CD.

In the manual process of data generation, a graphical user interface (GUI) is provided. Figure 4.4 shows screen shot of the GUI. All actions are presented with a point on a map visually. Also, some controls showing the flow of time through the days are placed on the GUI. When clicking an action on the map, the action is stored in the database as an action history row on behalf of the selected user. In this way, simulated data can be generated one by one according to the user clicking on the map.

Also, it is possible to generate simulated data in an automatic manner. Automatic generation of simulated data has an advantage that thousands rows of data can be generated in several minutes. Automatic generation process is based on some randomness and biases. The biases are used to make the results of FCM and LA more meaningful. The biases are only defined on the time dimension of the context space as follows. Remember that the cluster centres, which are results of FCM, define context profiles for their members. Consider a case that simulated data is generated in a purely random manner: at randomly selected time, a randomly selected action is implemented

by a randomly selected user, and then all action samples will be distributed on the time dimension randomly. Therefore, the time profile of the cluster centres will be unreasonable. For example, visiting a historical place can be defined by an unreasonable context profile like “on Sunday at 02:00 am” or going a bar can be defined by an unreasonable context profile like “on Monday at 08:00 am”. To remove the possibility of such unreasonable results, we use some predefined biases, which are predefined context profile (time profile) on the time dimension for each action. A time profile for an action consists of points on a one week timeframe, and each point means that this action can occur in predefined neighbourhood of this point. The size of the neighbourhood is configured manually, before the generation of simulated data. In Table 5.1, an example of a time profile assigned to an action is shown.

Table 5.1 An example of a time profile

No	**Day**	**Hour**	**Minute**
1	Monday	12	00
2	Monday	13	20
3	Tuesday	12	00
4	Tuesday	18	15
5	Wednesday	13	20
6	Wednesday	18	15
7	Thursday	16	6
8	Friday	17	52
9	Saturday	15	51
10	Sunday	12	28

The data generator works as follows. It progresses on time points defining the time profiles for actions, respectively, as the times goes throughout a week. At each point, the actions defined on that point are query from database, and then a user is selected randomly from a user subset and assigned to each action. Note that only a subset of users is used. Before running the generator, the size of this subset is determined. The size of

this subset determines the density of action occurrences per user along a week. Namely, the larger the size of the user subset is, the sparser action occurrences per user in the dataset are. The users included in this subset are selected randomly from all users set. The screen shot of automatic simulated data generator is presented in Figure 4.5.

5.2 The Test Results of the Proposed Approach

5.2.1 Fuzzy C-Means Clustering

The details of the FCM have been given in Section 3.1. 21485 rows of dataset have been produced by automatic simulated data generator, and each row refers to an action occurring in the context space. Remember that each sample is represented by a three dimensional vector: x, y and t. The scale of x and y is determined by the width and height of a map used to place the actions respectively. The map is a 2409x2143 pixel image representing a particular part of a city, more precisely the Eminönü, Fatih and Beyoğlu areas of Istanbul. The location of an action placed on this map is represented by a pair of pixel index (x, y). Thus, the scales of dimensions x and y on the context space are restricted by [0, 2409] and [0, 2143], respectively. Similarly, the dimension t is represented with one week and restricted by a scale with [0, 2500x7]. Thus, an occurrence of any action represented by day, hour and minute corresponds to a point in the interval [0, 2500x7]. It is expected that when the context space is separated into 7 clusters, each day of week will corresponds to only one cluster centre.

The dataset with 21485 rows is input to data set with following settings: “max. number of iteration”=100, “exponent for membership the matrix U”=2, “number of the clusters”= 7. Results are shown in Table 5.2. The FCM assign one cluster to per day. For weekday, the cluster centres occurs nearly at the same point on each day.

The results of FCM using the same input dataset with “number of the clusters”= 14 are shown in Table 5.3. Generally, each day is represented by two clusters which are placed

different location on X and Y. Also note that, on Monday, Tuesday and Wednesday, t coordinate of two cluster centres are placed close together.

Table 5.2 Results of the FCM with following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 7, "scale of t"=[0, 2500x7].

No	X	Y	Day	Hour	Minute	Number of Members
1	1149	883	Monday	14	57	128
2	1153	837	Tuesday	15	6	112
3	1061	895	Wednesday	15	55	112
4	1080	906	Thursday	16	6	112
5	1118	876	Friday	17	52	112
6	963	1138	Saturday	15	51	116
7	894	1143	Sunday	12	28	66

Table 5.3 Results of the FCM with following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 14, "scale of t"=[0, 2500x7].

No	X	Y	Day	Hour	Minute	Number of Members
1	1585	382	Monday	15	32	54
2	801	1304	Monday	14	58	74
3	1576	394	Tuesday	16	18	50
4	744	1264	Tuesday	15	45	62
5	684	1243	Wednesday	17	22	64
6	1607	382	Wednesday	16	21	50
7	1033	940	Thursday	15	45	112
8	1293	1533	Friday	13	47	29
9	553	1079	Friday	17	55	42
10	1598	357	Friday	19	22	47
11	554	998	Saturday	17	23	55
12	1115	1624	Saturday	13	49	40
13	760	1519	Sunday	15	21	46
14	1540	401	Sunday	0	8	43

The scale of t is increased to [0, 3500x7]. The results of FCM using the same input dataset with "number of the clusters"= 14 are shown in Table 5.4. It is inferred that the cluster centres are placed at separated coordinates on "t" as the scale of "t" becomes larger. For example, Monday is separated by 4 cluster centre coordinate on "t", and the x and y coordinates of the cluster centres becomes closer to each other.

Table 5.4 Results of the FCM with following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 14, "scale of t"=[0, 3500x7].

No	X	Y	Day	Hour	Minute	Number of Members
1	507	1178	Monday	17	49	50
2	1497	343	Monday	13	0	35
3	1592	401	Monday	19	9	46
4	1274	1557	Monday	11	50	39
5	743	1252	Tuesday	16	37	63
6	1571	405	Tuesday	17	0	50
7	692	1230	Wednesday	17	38	66
8	1578	408	Wednesday	15	9	58
9	711	1251	Thursday	16	37	62
10	1544	424	Thursday	15	53	51
11	815	1226	Friday	16	58	77
12	1524	458	Friday	21	16	49
13	947	1127	Saturday	16	38	116
14	848	1202	Sunday	14	11	66

The scale of t is decreased to [0, 2000x7]. The results of FCM using the same input dataset with "number of the clusters"= 14 are shown in Table 5.5. It is inferred that the cluster centres are placed at separated coordinates on "x and y" as the scale of "t" becomes more restricted. For example, Monday is separated by 4 cluster centres. But their coordinates on "t" are closer compared to Table 5.4, and the x and y coordinates of the cluster centres becomes more distant from each other. Determining which scale of "t" should be used is a manual process and application dependent. We only observe the

results of chances in scale. Since simulated data is used in this test, it is not important to discuss choosing a proper scale anymore. Similarly, determining the number of cluster is manual process, and depends on size and properties of the dataset. In our case, the properties of the dataset can be defined as: number of the action, granularity of the categories and scope of the context space.

Using the same dataset, the FCM produces another results with the following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 21, "scale of t"=[0, 2500x7]. The results are shown in Table 5.6. Also, some clusters of the results presented in Table 5.6 can be seen visually on the map in APPENDIX A. It will make the cluster definition clearer.

Table 5.5 Results of the FCM with following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 14, "scale of t"=[0, 2000x7].

No	X	Y	Day	Hour	Minute	Number of Members
1	479	1190	Monday	17	44	34
2	1831	322	Monday	17	9	38
3	1066	331	Monday	15	23	27
4	1319	1597	Monday	12	37	36
5	1169	811	Tuesday	17	21	112
6	690	1239	Wednesday	17	11	64
7	1617	377	Wednesday	16	31	49
8	988	1000	Thursday	16	15	112
9	1572	388	Friday	18	40	51
10	742	1257	Friday	16	51	63
11	518	1037	Saturday	17	6	46
12	1526	380	Saturday	23	3	45
13	1127	1635	Saturday	13	35	36
14	759	1564	Sunday	15	1	44

Table 5.6 Results of the FCM with following setting: "max. number of iteration"=100, "exponent for membership the matrix U"=2, "number of the clusters"= 21, "scale of t"=[0, 2500x7].

No	X	Y	Day	Hour	Minute	Number of Members
1	528	1372	Monday	18	35	42
2	1553	382	Monday	19	5	38
3	462	806	Monday	14	9	29
4	1370	1627	Monday	11	55	35
5	1847	290	Monday	13	45	25
6	473	1131	Tuesday	17	34	40
7	1261	1562	Tuesday	14	16	25
8	1615	310	Tuesday	14	14	39
9	1574	432	Tuesday	22	18	32
10	652	1249	Wednesday	17	44	64
11	1615	373	Wednesday	17	8	50
12	1581	376	Thursday	15	42	50
13	668	1252	Thursday	16	8	63
14	1530	329	Friday	15	48	48
15	512	1144	Friday	18	20	40
16	1313	1606	Friday	14	7	26
17	1558	426	Saturday	1	21	35
18	1115	1636	Saturday	13	59	38
19	536	1050	Saturday	17	49	53
20	1547	404	Sunday	0	45	44
21	758	1546	Sunday	15	23	46

5.2.2 Stochastic Learning Automata

For testing the offline training of LA, the dataset used in testing of FCM has been used. Only one metric has been used in test. This metric is the "Precision Metric" mentioned in [41], and defined as follows. Precision is defined as the ration of number of true prediction to number of prediction, shown in (Eq. 5.1):

$$P = \frac{Numer\,Of\ True\,Prediction}{Numer\,Of\ Prediction} \qquad \text{(Eq. 5.1)}$$

Remember that in section 3.2 two learning automaton (LA) called Per LA and Global LA are mentioned. They are combined with a weight called a, to work together. Also, there exist weights for penalty and reward, which are represented by w' and w respectively. The sensitivity of P to these three weights (a, w' and w) will be discussed through this section.

The GA is trained with transition data where the context space is represented by 7 cluster centres (FCM separates the context space into 7 clusters). Remember that automatic simulated data generator produces the data consecutively along a week. In the context space defined by 7 points (cluster centre), each cluster centre corresponds to a day. If an action occurrence of a user is dense enough, almost all users will tract the same path between the cluster centres through a week. So, at the end of the training of the GA, it is expected that the GA will predict the users' transitions with high precision.

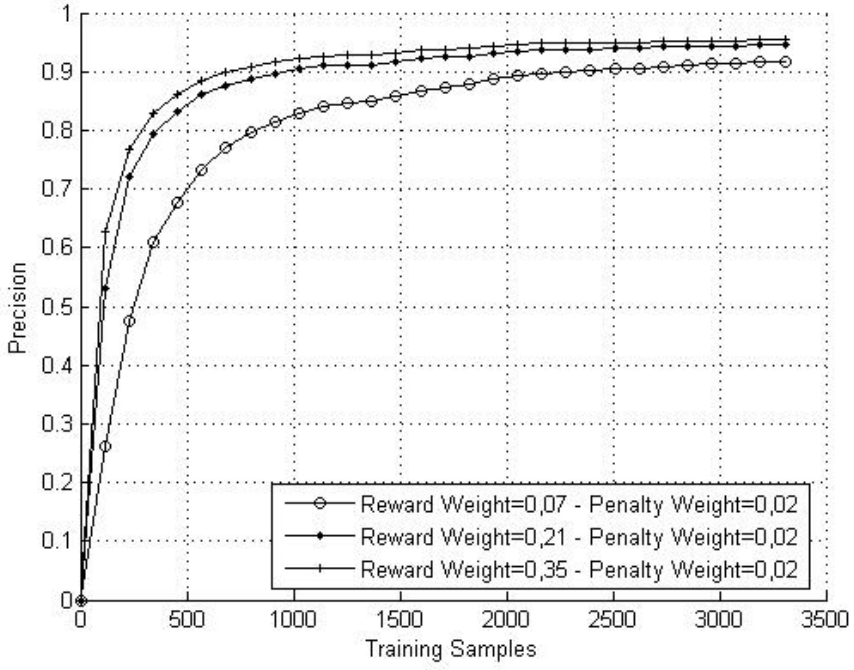

Figure 5.1 Precision graphic of the GA in the context space defined by 7 cluster centres. Each graphic presents the chance of precision of the GA in training period with different reward weights.

As seen in Figure 5.1, each graphic converges to the high precision value. Note that when the reward weight is increased from 0.07 to 0.21, we observe a significant improvement in convergence of the precision. On the other hand, when the reward weight is increased from 0.21 to 0.35, the improvement is trivial.

The changes in penalty weight do not improve or decline the performance considerably. The test results with different penalty weight (0.02, 0.05 and 0.1) are shown in Figure 5.2

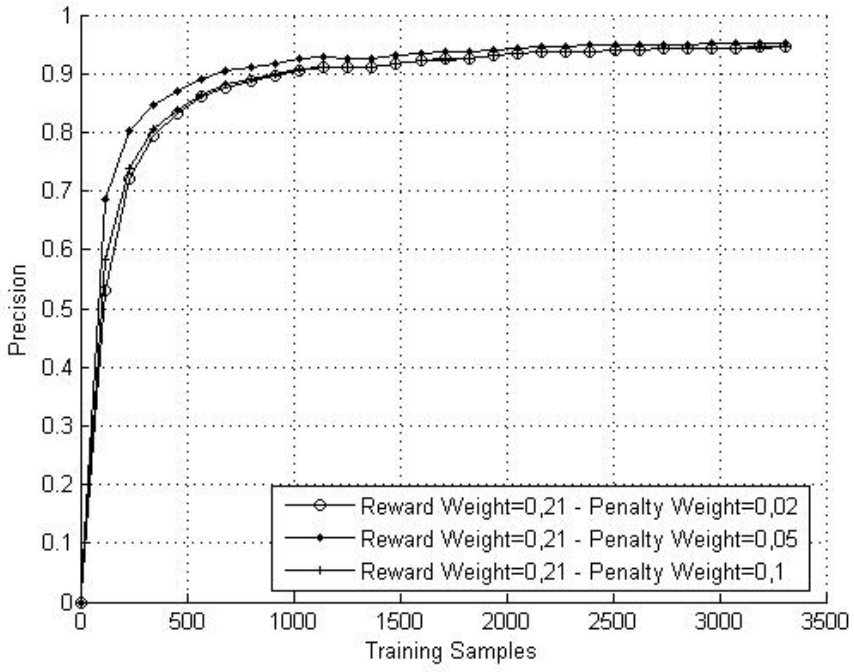

Figure 5.2 Precision graphic of the GA in the context space defined by 7 cluster centres. Each graphic presents the chance of precision of the GA in training period with different penalty weights.

The GA is trained with transition data where the context space is represented by 14 cluster centres. If the number of the cluster centres is increased to 14, it will introduce new alternative paths through a week, and the paths of the users will become more different from each other. Since the GA learns the global flow of the users' motion

through a week, the precision of the GA will decrease. The results of the GA where the context space is defined by 14 cluster centres is shown in Figure 5.3

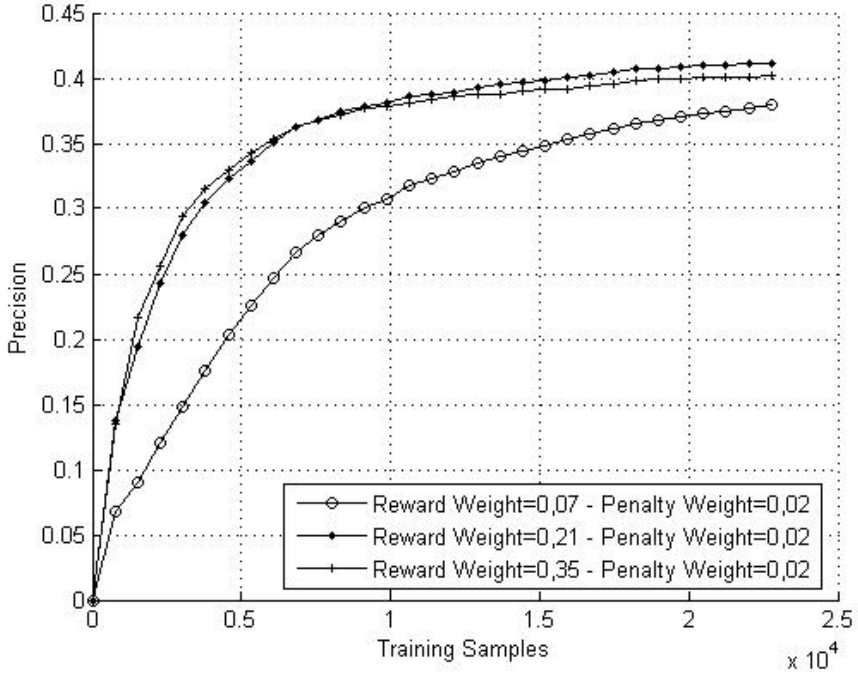

Figure 5.3 Precision graphic of the GA in the context space defined by 14 cluster centres. Each graphic presents the chance of precision of the GA in training period with different reward weights.

In Figure 5.3, three different graphics are shown. They are results of the training of the GA with different reward weight; 0.07, 0.21 and 0.35, respectively. The figure shows that when the GA is trained with reward weight 0.07, the graphic does not converge to a constant value, and more training samples are required. The results with reward weight 0.21 and 0.35 are somewhat similar to each other. Both of them converge to a constant value. Also, it is observed that there exists a significant decrease in precision compared to the results of the context space represented by 7 cluster centre (Figure 5.2). This decrease is due to the increase of the number of the cluster centres from 7 to 14.

Similar to the previous case, if the number of the cluster centres is increased to 21, it will introduce new alternative paths through a week. And the GA is trained with transition data where the context space is represented by 21 cluster centres. Almost all users will tract the same path between the cluster centres through a week where the context space is represented by 7 cluster centres, since each day of week is represented by only cluster. So, the GA will predict the users' transitions with high precision. Each day of week will be represented by more clusters where the context space is represented by 21 cluster centres. In the second case, since each user has more selection alternative for each day of week, each user's path becomes more different compared to others. In other words, the more clusters the context space is represented by, the more different paths appear through a one week. Consequently, the precision of the GA will decrease compared to previous case where action space is represented by 21 cluster centres. The result of the GA where the context space is defined by 21 cluster centres is shown in Figure 5.4.

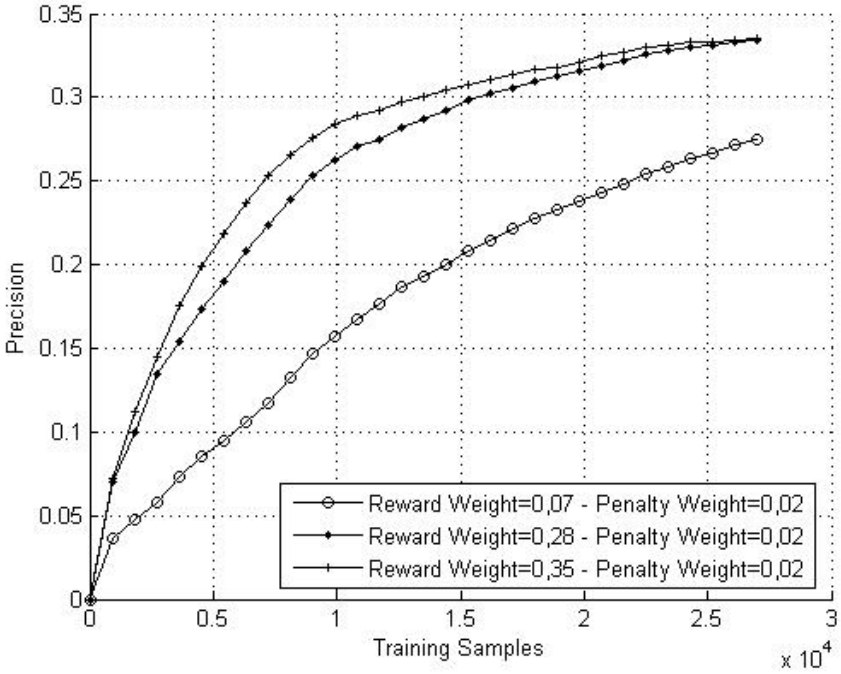

Figure 5.4 Precision graphic of the GA in the context space defined by 21 cluster centres. Each graphic presents the chance of precision of the GA in training period with different reward weights.

In Figure 5.4, three different graphics are shown. They are results of the training of the GA with different reward weight; 0.07, 0.28 and 0.35, respectively. The figure shows that when the GA is trained with reward weight 0.07, the graphic does not converge to a constant value, and more training samples are required. The results with reward weight 0.28 and 0.35 somewhat similar to each other. Both of them converges a constant value. Also, it is observed that there exists a significant decrease in precision compared to results of the context space represented by 14 cluster centre (Figure 5.3). This decrease results from the increase of the number of the cluster centres from 14 to 21.

Above we presented the results of the GA with three different numbers of cluster centres; 7, 14 and 21 respectively. In the first test with 7 cluster centres, the precision converge very high value, since almost all users follow the same or similar paths. In other words, global flow between clusters, which is extracted by the GA, covers almost all users' flows. Obviously, this case will not fit the scenario in a real mobile environment. As the number of the cluster centres representing the context space increases, namely, the possibility of occurrence of different path increase, the precision value decreases.

The result shown in Figure 5.4 (21 cluster centres) is the best one fitting a real scenario. The precision of the GA converges nearly 0.33. To predict user motion with high precision, two learning automaton work together. The other one is called the PUA. The PUA learns per user's flow between the cluster centres. But, to test the PUA, we can not use simulated data produced automatically since the data produced for per user is haphazard and do not include any routine. Consequently, the PUA can not extract any flow from per user data. To test the PUA, a fixed path is determined, and it is assumed that a chosen user follows this fixed path at the turn of each week in the context space. Hence, we can observe the results of the PUA.

Remember that the GA and PUA are combined with weight a (Eq. 3.6), and the resulting automaton called as weighted automaton (WA). Value of a, penalty weight and

reward weight is chosen as 0.2, 0.02, and 0.21, respectively. 12 cluster centres are selected randomly and sorted according to their time. Thus, the path of the user consists of 12 fixed points. After following this path 30 times, the precision graphic of the weighted automaton is as follows (Figure 5.5):

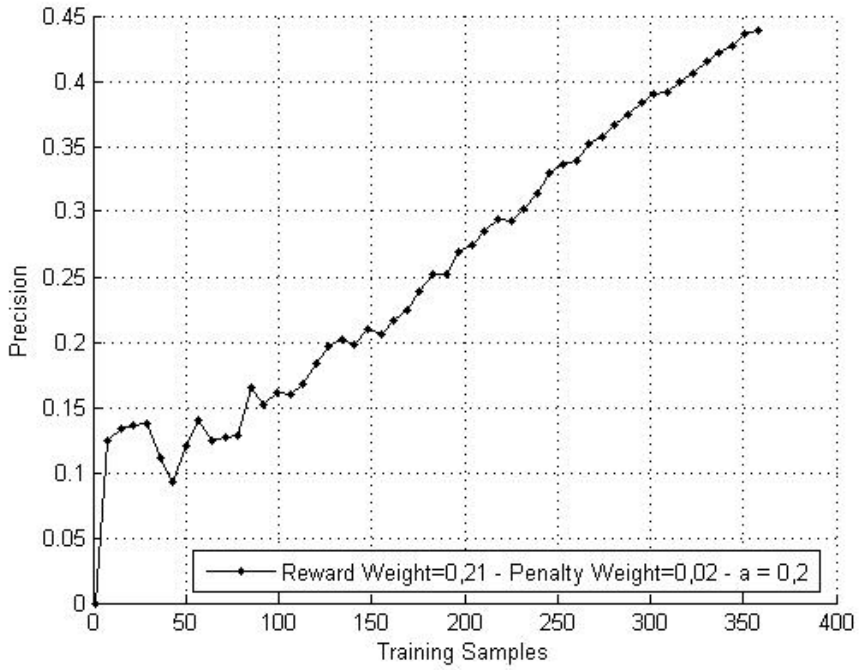

Figure 5.5 Precision graphic of the WA with 12x30=360 training sample in the context space defined by 21 cluster centres.

In Figure 5.5, the precision result is presented with 12x30=360 training sample. Up to 100 training samples, the graphic varies around precision value 0.12, and approximately, mean value of precision in this interval is 0.12. This behaviour results from the contribution of the GA. After 100th samples, the precision increases up to 0.44 at 360. It means that the PUA starts to learn the user's routine increasingly. Obviously, if the user continues itinerating on the same path, the precision value will go up to higher values.

The value of a determines the contribution degree of the GA to the WA. In other word, if the value of a increased, the contribution degree of the GA will become higher. After

increasing a from 0.2 to 0.5, the graphic of precision of the weighted automaton is shown in Figure 5.6. Up to 50 training samples, the graphic varies around precision value 0.25, and approximately, mean value of precision in this interval is 0.25. It is observed that the initial precision value has increased and the initial interval has shrunk compared to Figure 5.5. This increase and shrinking are due to the increase of the contribution degree of the GA. On other hand, after 50th sample, the learning curve has decreased compared to Figure 5.5, because of the decrease of the contribution degree of PUA.

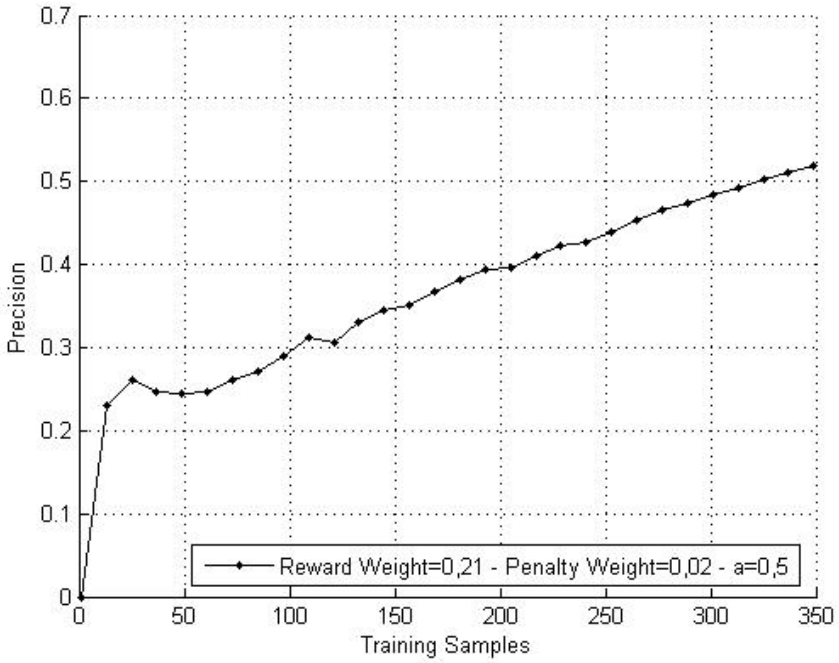

Figure 5.6 Precision graphic of the WA with 12x30=360 training sample in the context space defined by 21 cluster centres.

After presenting the path 70 times, the graphic of precision of the weighted automaton is shown in Figure 5.7. Since the itinerary of the user does not change, the precision value still goes up. Finally, after training with 230x12 samples, the precision value converges to a constant as presented in Figure 5.8.

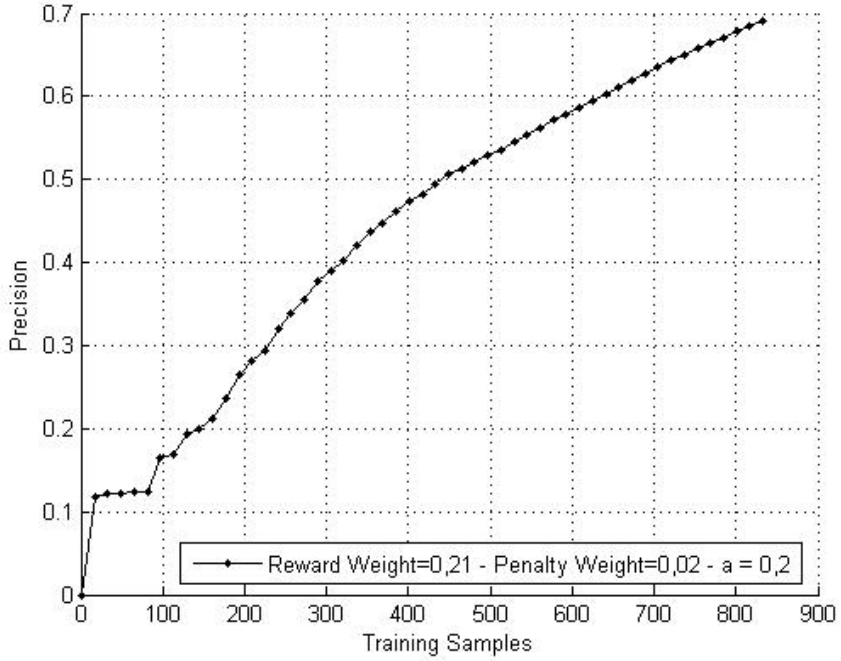

Figure 5.7 Precision graphic of the WA with 12x70=840 training sample in the context space defined by 21 cluster centres.

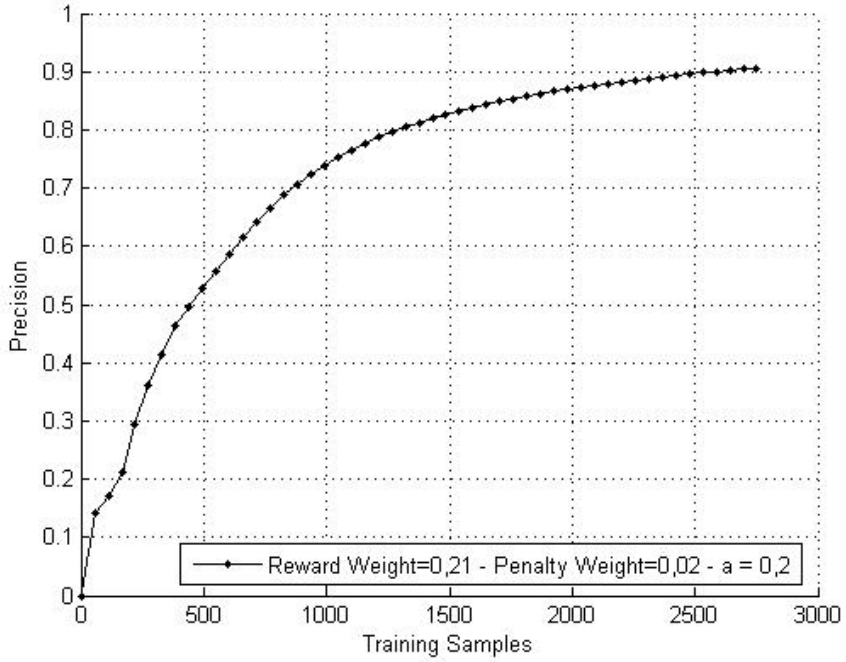

Figure 5.8 Precision graphic of the WA with 12x230=2760 training sample in the context space defined by 21 cluster centres.

Note that a single GA can not predict the user motion with high precision. When the PUA and the GA are combined, precision value reaches high value. Here, the task of the GA can be described as an assistant that assist the PUA on initial conditions where enough training samples are not available yet to train the PUA. It is obvious that in the real case, the user will not follow the same path permanently. On other hand, per user's path includes somewhat randomness and routine, and which refers to per user's path profile. The task of PUA is to learn these per users' path profiles.

5.3 Unsupervised Fuzzy Divisive Hierarchical Clustering

In section 3.3, the details of the UFDHC were given. The result of the UFDHC depends on four parameter: σ, $\bar{N}$, m and t. $\bar{N}$ is the minimum number of users contained in any cluster with membership degree to either A_1 or A_2 higher than a prefixed threshold σ. m is the fuzzification constant which determines the degree of fuzziness of the partition. And t is the threshold for the polarization degree. Here, we will test the sensitivity of the binary tree to these parameters.

The test dataset has been generated automatically by the simulated data generator. The simulator generator produces simulated action dataset on behalf of randomly selected users. After that an action from any category is executed by any user, the access count of that user to that category is increased by one. Any bias is not used, when constructing the users' access patterns to categories. So the access patterns of the users can resemble uniform distribution to some degree. Namely, access patterns of the users may not be very different from each other. The module has been tested with 1200 users and 11 categories, and results are as follows. At each node in the binary, only the categories exceeding %10 preference degree are represented.

In Figure 5.9, the results of the UFDHC with setting $\sigma = 0.7$, $\bar{N} = 15$, $m = 1.2$ and $t = 0.7$ are represented. Each node represents a fuzzy cluster, and each cluster is identified by a

prototype which summarizes the preference of the users strongly belonging to the cluster, thus identifying the profile of its typical members.

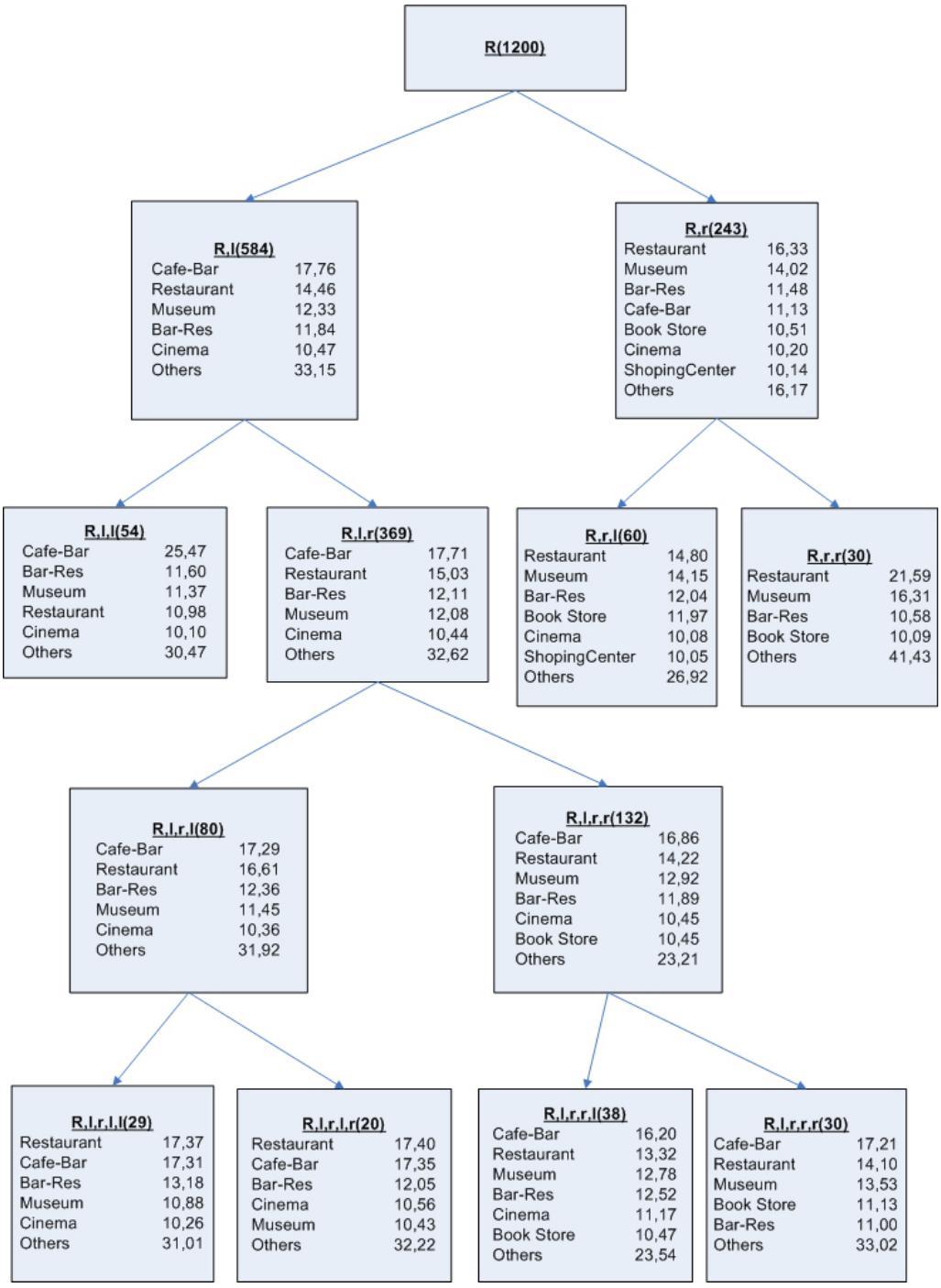

Figure 5.9 The result of the UFDHC with following setting: $\sigma = 0.7$, $\overline{N} = 15$, $m = 1.2$ and $t = 0.7$.

The number of users belonging to the cluster and percentages with which the categories contribute to the user profile are given within each node in Figure 5.9. There exist 1200 users at the top of the hierarchy, where they separate into two branch. 373 users are not classified as a member of any cluster in the hierarchy since they can not exceed the threshold σ and $\bar{N}$.

Remember that the constant m determines the degree of fuzziness. The closer m is to 1, the crisper the partition is. Thus, if we execute the UFDHC with m close to 1, we expect very crisp partitions characterised by a high polarization degree. When m increased to 1.5, the UFDHC can not produce a binary tree, since the polarization degree of any produced clusters can exceed threshold $t = 0.7$. For the values of $m = 1.4$ *and* $m = 1,3$ the

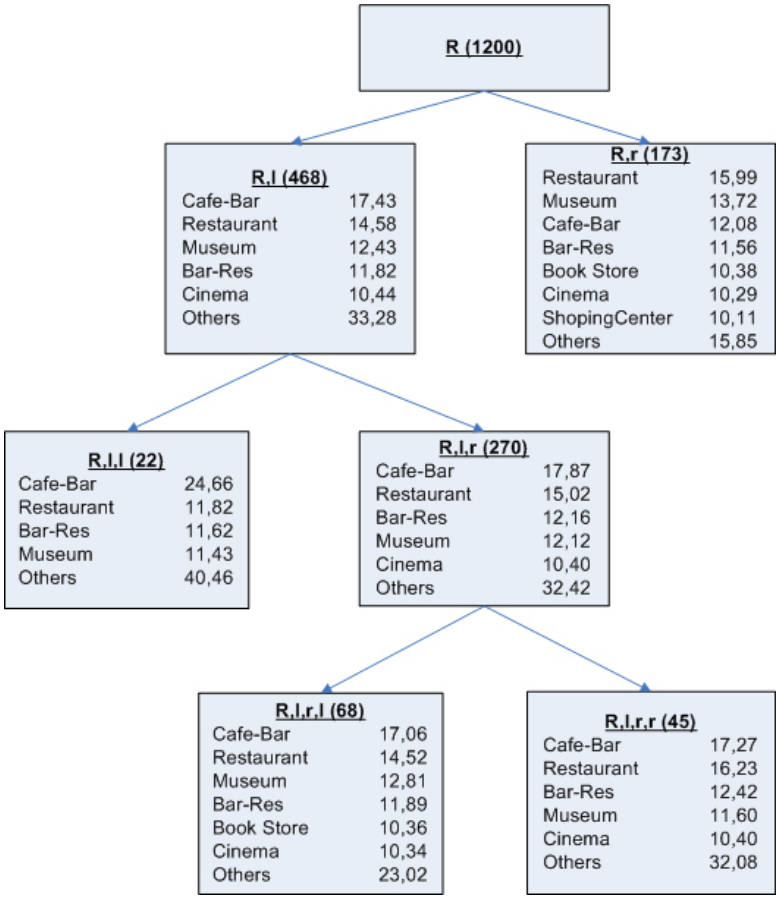

Figure 5.10 The result of the UFDHC with following setting: $\sigma = 0.7$, $\bar{N} = 15$, $m = 1.25$ and $t = 0.7$.

result does not change. When $m = 1.25$, the UFDHC has produced a binary tree as shown in Figure 5.10. As seen in the figure, some branches of the tree have been removed, since their polarization degrees could not exceed the threshold.

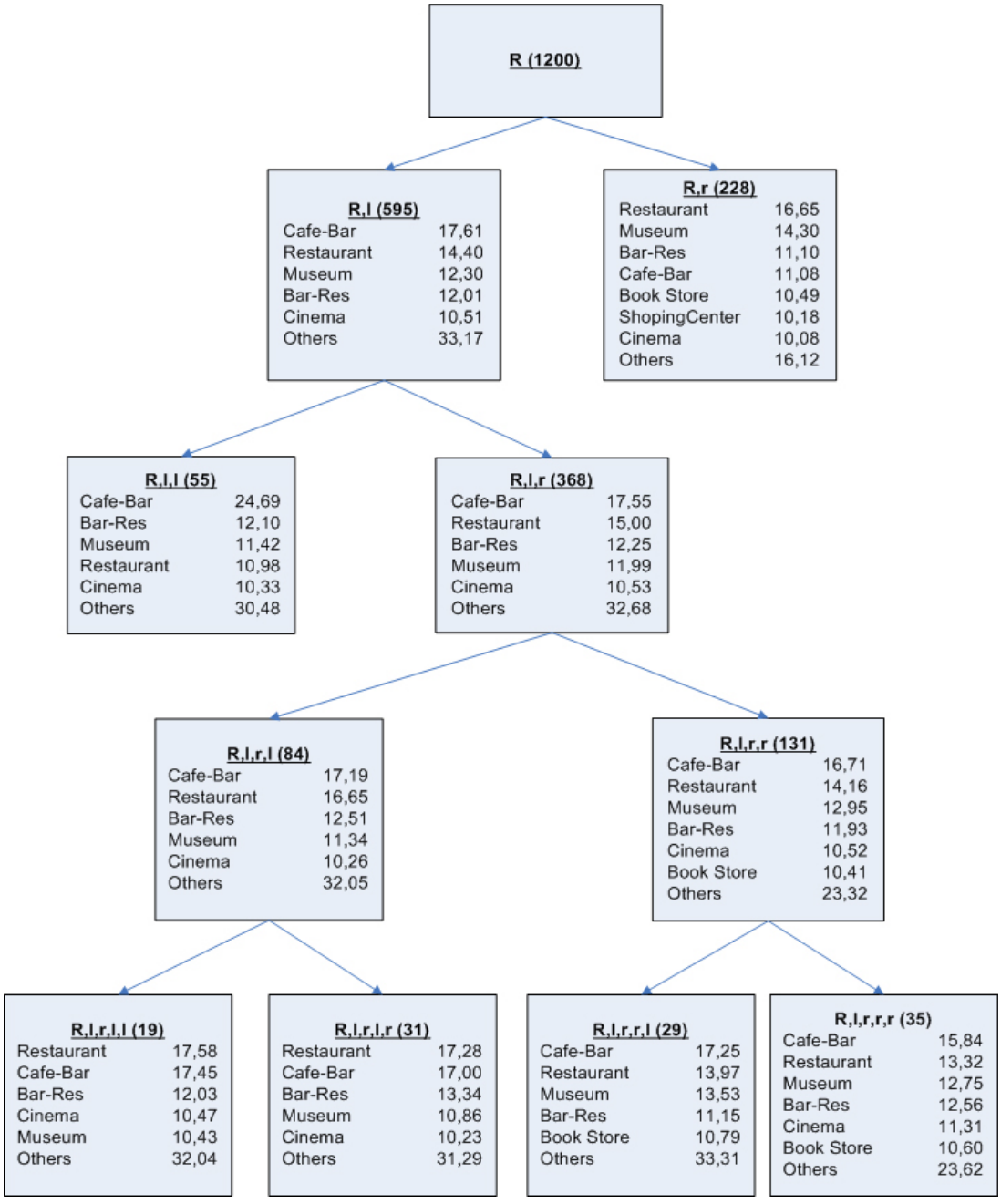

Figure 5.11 The result of the UFDHC with following setting: $\sigma = 0.7$, $\bar{N} = 15$, $m = 1.2$ and $t = 0.8$.

The value of t must be fixed taking into account the degree of fuzziness of the data set being considered. High values of t allow us to discover cluster structure in crisp datasets. On the contrary, low values of t allow us to more correctly partition fuzzy datasets with suitable fuzzy partition. When t is configured as $t = 0.8$ and the others $\sigma = 0.7$, $\bar{N} = 15$, $m = 1.2$, the output of UFDHC is shown in Figure 5.11. Note that child clusters of the cluster R,r(228) are removed from the tree, since the partitioning can not exceed the polarization threshold.

If we execute the UFDHC with $m = 1.1$, we expect very crisp partition characterised by a high polarization degree. The other parameters are configured as follows: $\sigma = 0.9$, $\bar{N} = 50$, $t = 0.9$. The result of the UFDHC with these settings is presented in Figure 5.12. Despite the parameters σ, $\bar{N}$ and m are set to quite high value, a binary tree consisting of 8 cluster has been produced.

In this test, we have only shown the effects of manual inputs t, σ, $\bar{N}$ and m on the resulted binary tree. We have not discussed on which results are better or not. We would like to point out that the values of these parameters are critical application-dependent parameters. So depending on the properties of the dataset, the values of them can change relatively.

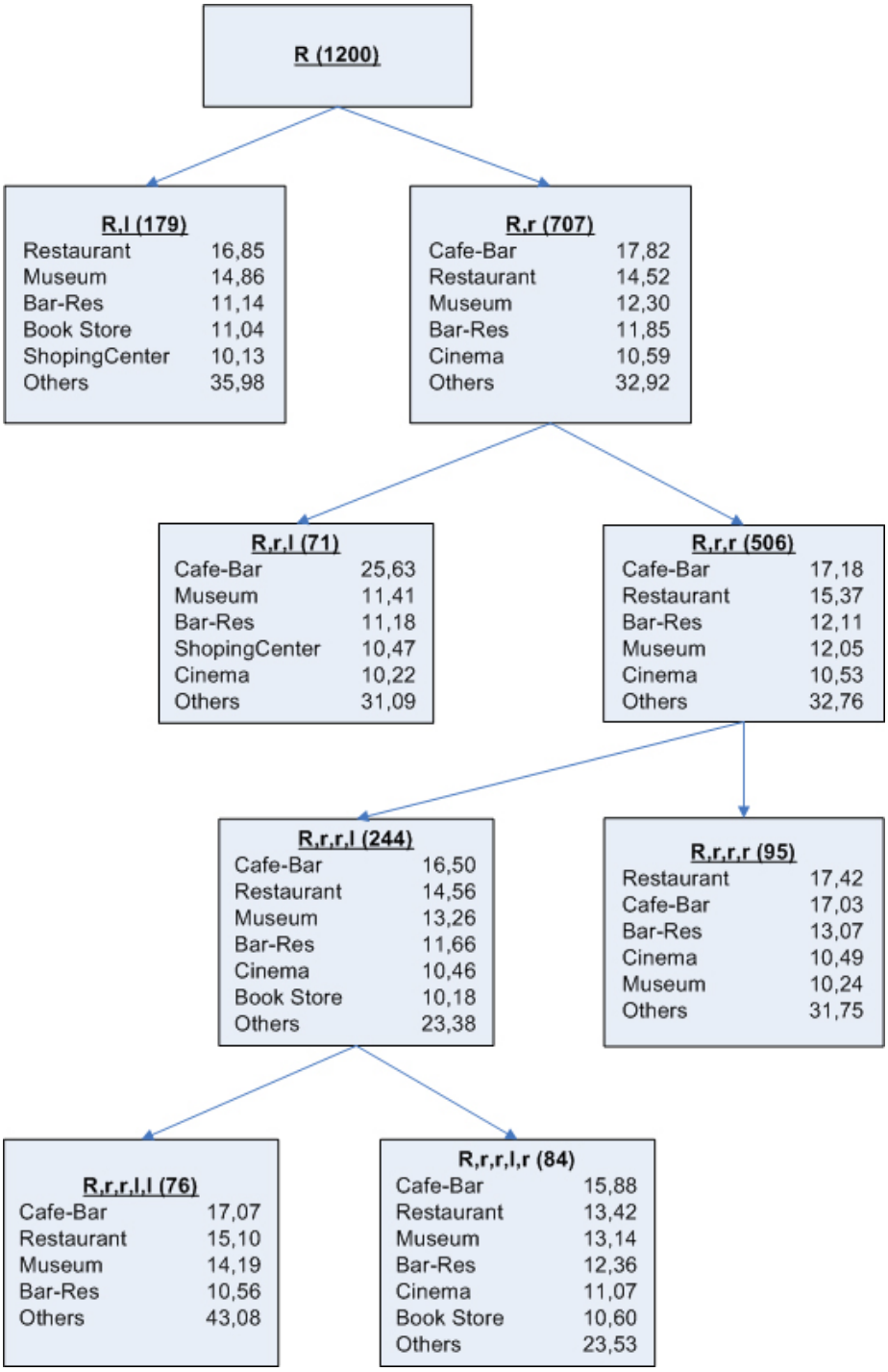

Figure 5.12 The result of the UFDHC with following setting: $\sigma = 0.9$, $\bar{N} = 50$, $m = 1.1$ and $t = 0.9$.

5.4 A Demonstrative Complete Run

In this section, a demonstrative complete run of simulator is given step by step. It will make the entire scenario clearer. The only one FCM output given in Table 5.3 is used throughout this section. This FCM output represents the context space with 14 cluster centres.

Firstly, we have trained the GA. The precision graphic of the training period is given in Figure 5.13.

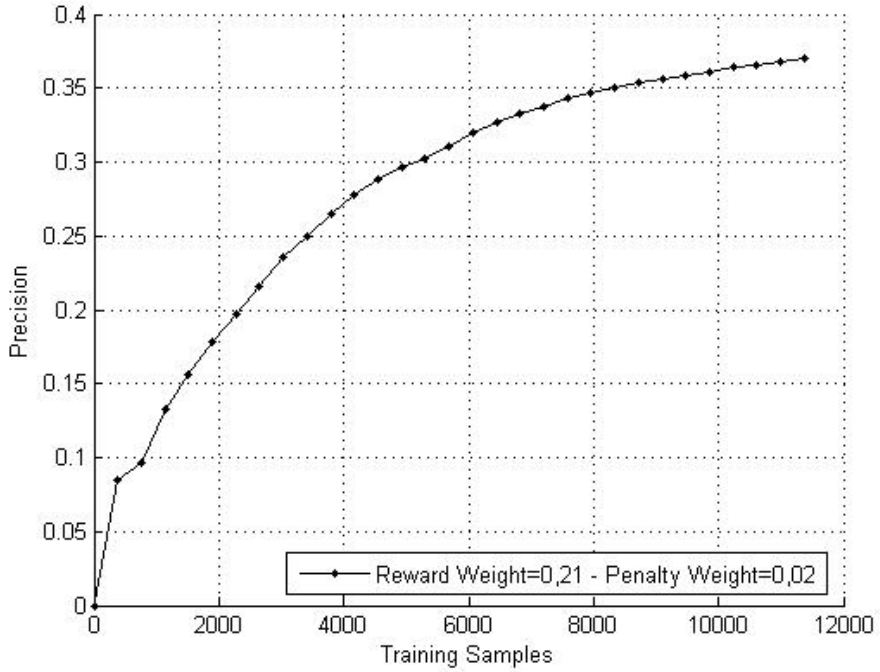

Figure 5.13 Precision graphic of the GA in the context space defined by 14 cluster centres.

Secondly, we have run the UFDHC with following setting: $\sigma = 0.7$, $\overline{N} = 15$, $m = 1.25$ and $t = 0.7$. The output is given in Figure 5.14.

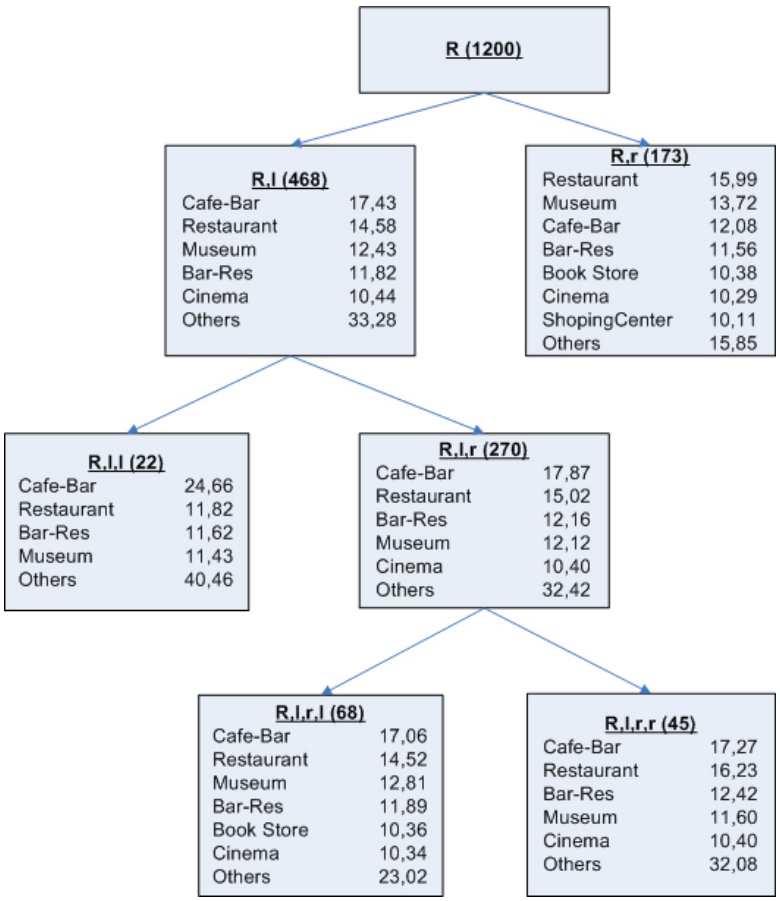

Figure 5.14 The result of the UFDHC with following setting: $\sigma = 0.7$, $\overline{N} = 15$, $m = 1.25$ and $t = 0.7$.

Thus, after training of the GA and the UFDHC, the offline training of the recommendation system have been completed. In the online part of the recommendation system, the experience of a user will be introduced, using Recommendation Module and GUI given in the section 4.7 and the outputs produced in the offline part.

Initially, the GUI represented in Figure 5.15 is provided to the user. The Reward Weight, Penalty Weight and PUA GA Weight have been defined as 0.21, 0.02 and 0.03 respectively. The context space has been grouped into 14 clusters from Cluster Result list box. Finally, the user identified by a-0 has been selected from Users list box. Upon clicking on the map, another GUI given in Figure 5.16 is presented to the user. On the GUI, a data grid shows the cluster centres. From the data grid, we have selected two

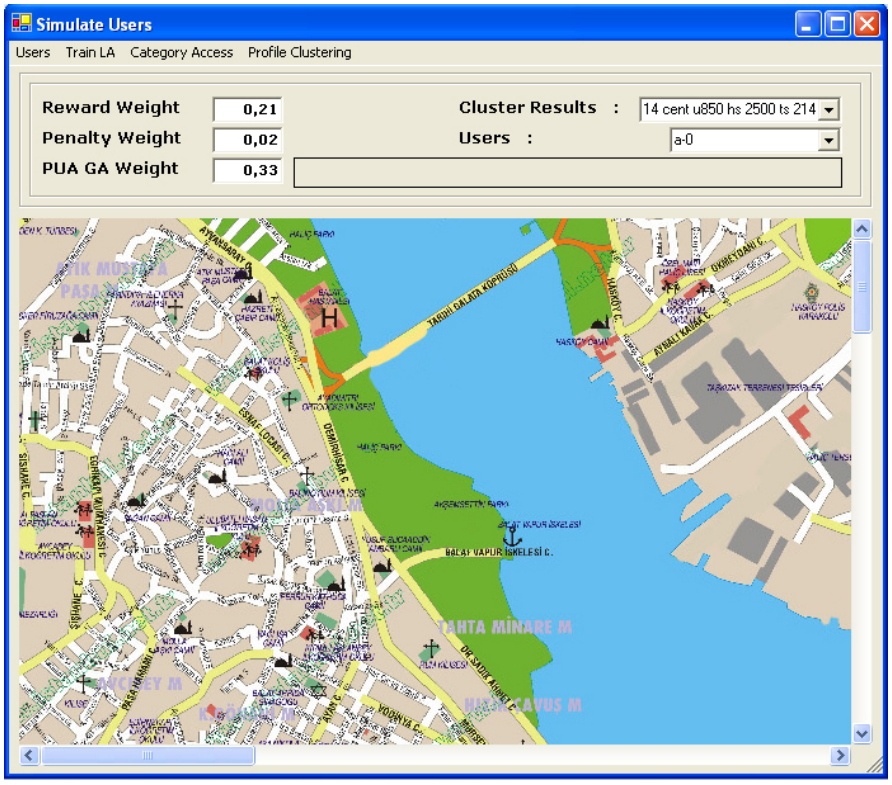

Figure 5.15 The GUI initially presented to the user.

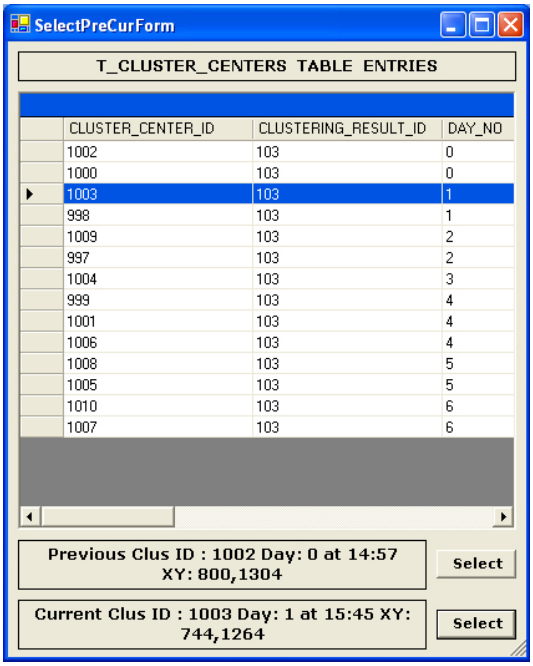

Figure 5.16 The GUI initially presented to the user.

cluster centres, via clicking on Select buttons, then closed the GUI. The selected cluster centres' ids are previous cluster id: 1002 and previous cluster id: 1003. The Recommendation Module assumes that the user has already been in these clusters.

The GUI given in Figure 5.15 appears again. After clicking on the map, the GUI given in Figure 5.17 is presented to the user.

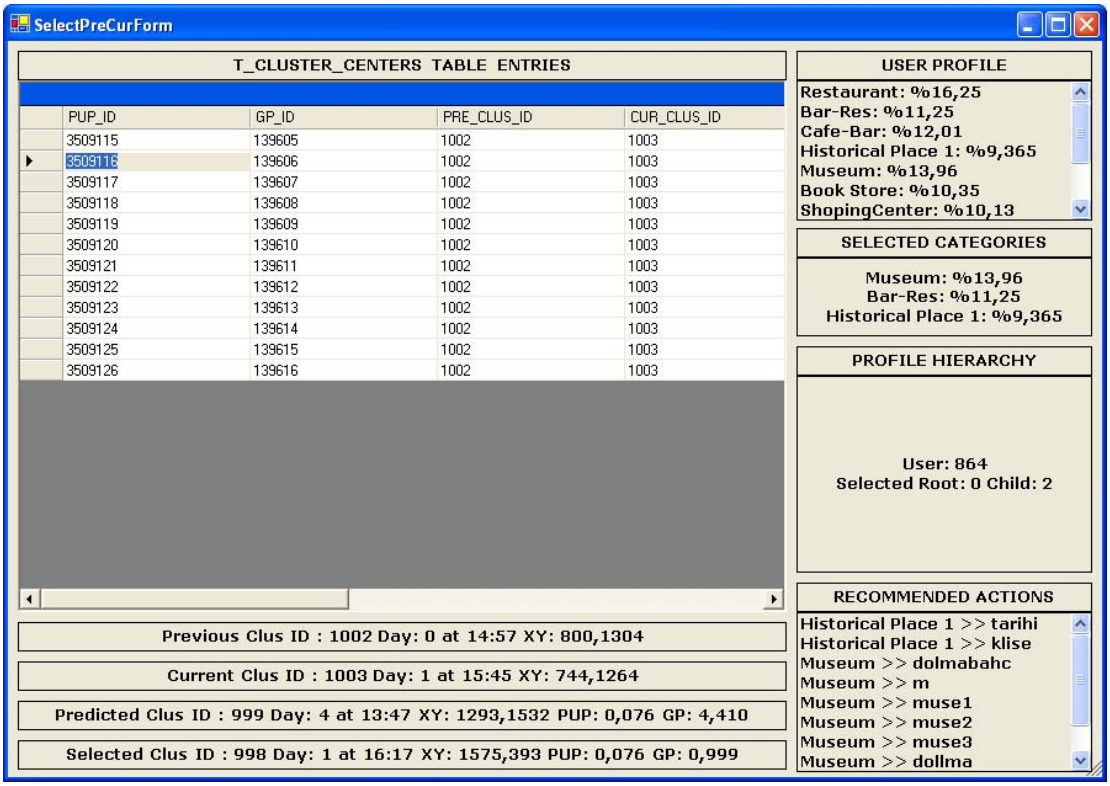

Figure 5.17 The recommendation GUI.

On the lower left-hand side of Figure 5.17, we see four labels which represent previous cluster id, current cluster id, predicted cluster id and selected cluster id respectively. The previous and current cluster id represent the user's previous transition and the coordinates of those cluster centres. The predicted cluster id shows the next cluster id predicted by LA according to the previous and current cluster id, which are 1002 and 1003. The selected cluster id shows the next cluster selected by the user. The recommendation module assumes that the user will go to selected cluster.

On the upper left-hand side of Figure 5.17, we see a data grid control called T_Cluster_Centers Table Entries. It allows the user to select his/her next cluster id, which is next transition, via clicking on rows manually. According to his/her selection, the label showing the selected cluster id changes its status automatically. We have selected the cluster id 998.

On the right-hand side of Figure 5.17, we see four labels which represent user profile, selected categories, profile hierarchy and recommended actions respectively. The user profile label shows the profile of the cluster, which the user is a member of, which is produced by UFDHC. Each row of the profile shows the percentage of contribution of each category on the user preference. The selected categories label represents the categories selected by Roulette Wheel Selection algorithm according to the percentage of contribution of each category. The profile hierarchy label represents the user id and the layer in the hierarchical tree. For example, the root: 0 child 2 defines that this profile is a child of root cluster. The recommended action label represents the recommended actions which is included in one of the categories given in selected categories label.

After closing the GUI, the recommendations to the user are shown graphically in Figure 5.18. Actually, the figure represents the same GUI in Figure 5.15, but now it represents the cluster centre 998 and the actions belonging to the cluster 998. The points represent the action included in the selected cluster. When the user clicks on one of the actions on the map, that action is implemented by that user. Upon second click on map, the recommendation GUI appears again as in Figure 5.19. Now the cluster ids 1003 and 998 are defined as previous and current clusters, and a predicted cluster presented to the user based on these cluster ids. Similar to the previous case, the user profile label shows the profile of the user, the selected categories label represents the selected categories, and the recommended action label represents the recommended actions. After selecting a cluster id and then closing the GUI, the recommendations will be shown on the map again. And this process goes on repeatedly.

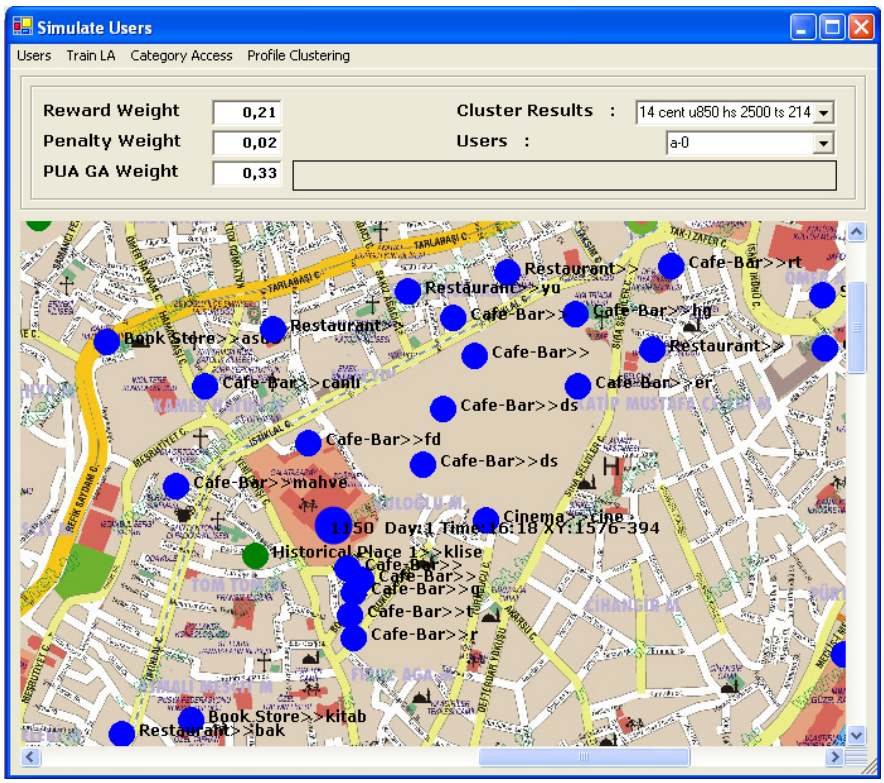

Figure 5.18 The GUI showing the recommendations visually.

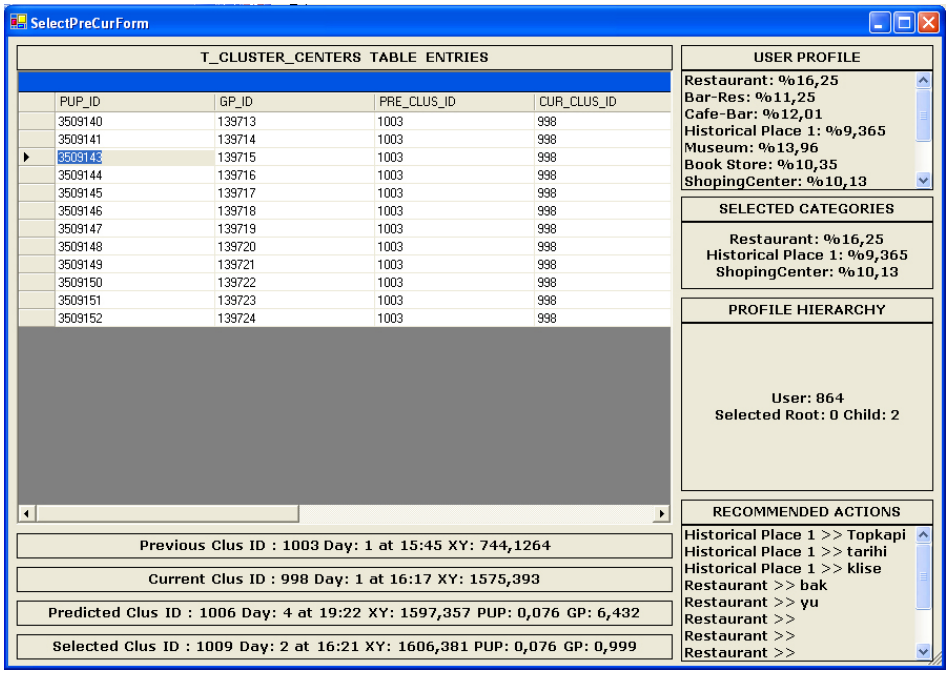

Figure 5.19 The recommendation GUI.

We have followed the path consisting of cluster ids given as follows: 1002, 1003, 998, 1009, 997, 1004, 999, 1008, 1007, 1002 and 1003. After following the same path 7 times repeatedly, the precision graphic is given in Figure 5.20.

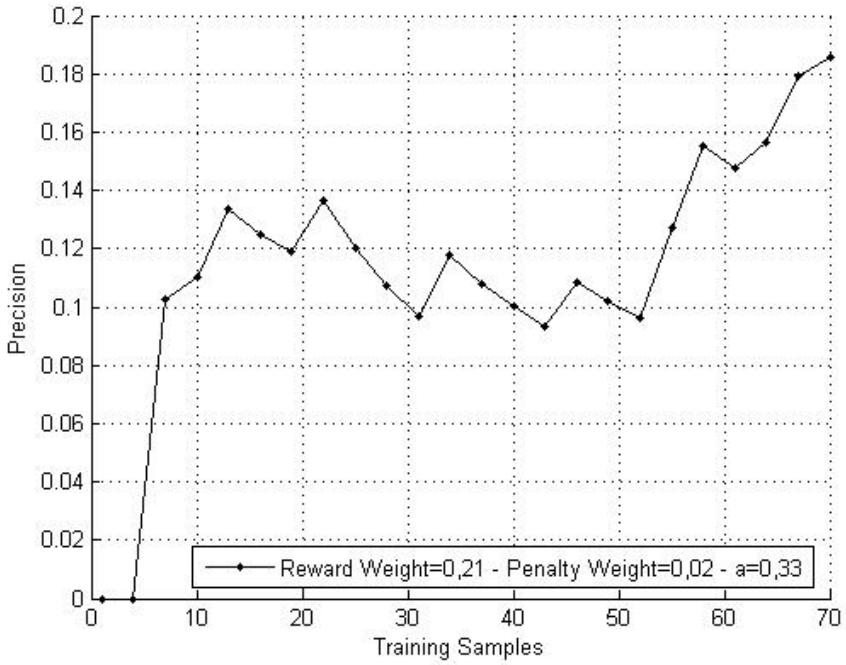

Figure 5.20 Precision graphic of the WA with 11x7=77 training sample in the context space defined by 14 cluster centres.

Figure 5.20 show that after 50 samples the system, the precision starts increasing. It means WA starts learning the user behaviour.

CHAPTER 6

CONCLUSION

6.1 Summary and Conclusive Remarks

In the scope of this thesis, we try to propose a recommendation system for the mobile environment. Generally, the recommendation system has been used in web based applications, and too many recommendation systems have been proposed for Information Retrieval and E-Commerce (related works are presented in Section 2.1). Through searching web based recommendation system in the literature, information related to general definition of recommendation problem for web environment is gathered. But in our case, the problem definition is somewhat different. Additional characteristic properties of the mobile environment take the recommendation problem beyond the web environment. These properties are called as mobility and ubiquity, and addition definitions resulting from these properties have been presented in details in Section 1.2. Finally, two new problems have been introduced in addition to pure recommendation problem (user profiling): "context-awareness" and "motion prediction in context space". The works related these concepts have been presented in Section 2.2 and Section 2.3.

A method, which handles "context-awareness" and "motion prediction in context space" in a combined manner, is proposed. The method combines the two methods called Fuzzy C-Means (FCM) and Stochastic Learning Automaton (LA). On the other hand, "user profiling" is handled separately, using a method called Unsupervised Fuzzy Divisive Hierarchical Clustering (UFDHC) for extracting users' profiles. The data derived from the users' action logs is presented as an input vector set to FCM. The FCM separates the action set into clusters defined by cluster centres. Each cluster centre represents context

profile for its members. Also, by means of cluster centre, the context space is represented by fixed points, and then motions of any user are defined as transitions between clusters which are predicted by the Learning Automaton. The UFDHC works separately. The output of UFDHC is binary tree of the clusters. Each cluster is identified by a profile which summarizes the ontological preferences of the users strongly belonging to the cluster. The recommendations are generated based on two data: next cluster id predicted by LA and the user's profile retrieved from binary tree produced by UFDHC. Therefore, the recommendation set presented to the user consists of those actions fitting the user's profile while being member of the predicted next cluster. The details of the methodology have been presented in Chapter 3.

To show how the proposed approach works, a simulator was developed. The implementation details have been presented in Chapter 4. Basically, the simulator consists of three main modules: FCM, LA and UFDHC. Each module was tested separately. All tests are based on simulated data. So a simulated data generator was developed to produce test data. In generation of the simulated data, some biases are used to imitate the data expected to occur in real case. The simulated data generator is explained in details in Section 5.1. The test result of each component is presented through the Section 5.2, Section 5.3 and Section 5.4.

The context space has been divided into 7, 14 and 21 clusters, respectively. The higher number of cluster the context space is divided into, the more detailed context profile each action is represented by. The FCM produce expected and reasonable results which shows that the module works well with simulated data.

In prediction module two LA (PUA and GA) work collaboratively. The simulated dataset representing transitions of the users between clusters was produced from the FCM results. This dataset was used to test the GA. The results show that the GA learns global transitions in the dataset. To test the PUA, we can not use simulated data produced automatically, since the data produced for per user is haphazard and do not

include any routine. Consequently, the PUA can not extract any flow per user data. To test the PUA, a fixed path is determined, and it is assumed that a chosen user follows this fixed path at the turn of each week in the context space. The results show that the combination of GA and PUA works well in this test.

LA was used for prediction, because of its simplicity. Indeed, generally, LA is not considered as very efficient learning method. And we think this test is not sufficient to prove that LA is a proper choice. It should be tested with more realistic data.

The output of UFDHC is a binary tree of user groups characterized by a set of common interests and represented by a prototype, which corresponds to the profile of the users of the group. Several binary trees have been produced with different manual settings, and the effects of the manual settings have been observed. Generally, it is inferred that the produced clusters have not distinctive profiles. Namely, the profiles resemble each other to a great extent, since the simulated data generator produces the users' access pattern data randomly. And this randomness causes uniform distribution on produced data. However, the results are sufficient to prove that the UFDHC work well for extraction of the users' profiles. Furthermore, the members of each cluster have similar interest on ontological category, and the collaboration can be established between the same cluster's members. Namely, the combination of UFDHC with pure collaborative filtering methods may produce better result where the contents consist of multiple categories.

Each component of the recommendation system was verified one by one using simulated data. Then, the components was integrated for recommendation task and tested with simulated data. The test results verify that the integrated system works well. But we don't think that the integrated system is validated for more realistic cases, since the test data produced by the data generator is not enough realistic to validate the proposed architecture.

6.2 Future Works

To validate the proposed architecture for real cases, we should test it with a realistic dataset. The realistic dataset can be obtained directly from real user or produced by more realistic simulated data generator. Obtaining the dataset from the real users or the design and development of more realistic data generator remain as future work.

Also, after testing with realistic data, the research on alternative methods for prediction task, and the research on the combination of UFDHC and pure collaborative filtering methods remain as future works.

REFERENCES

[1] Harry Chen, Sovrin Tolia. Steps Towards Creating a Context-Aware Software Agent System. HP Laboratories Palo Alto HPL-2001.

[2] Stuart Edward Middleton. Capturing knowledge of user preferences with recommender systems. A thesis submitted for the degree of doctor of philosophy in the faculty of engineering and applied science department of electronics and computer science in university of Southampton. May 2003.

[3] Danny POO, Brain CHNG and Jie-Mein GOH.A Hybrid Approach for User Profiling. Proceedings of the 36th Hawaii International Conference on System Sciences. 2002 IEEE.

[4] Kaname Funakoshi, Takeshi Ohguro. A content-based collaborative recommender system with detailed use of evaluations. Fourth International Conference on knowledge-Based Intelligent Engineering System 2000 IEEE.

[5] Michael J. Pazzani. A Framework for Collaborative, Content-Based and Demographic Filtering Artificial Intelligence Review Volume 13 , Issue 5-6 Special issue on data mining on the Internet Pages: 393 – 408, 1999

[6] Mukund Deshpande, George Karypis. Item-based Top-N Recommendation Algorithms. ACM Transaction on Information Systems, Vol. 22, no. 1, January 2004.

[7] Tong Zhang, Vijay S. Iyengar. Recommender Systems Using Linear Classifiers. Journal of Machine Learning Research 2 (2002)

[8] HOFMANN, T. Latent Semantic models for collaborative filtering. ACM Transactions on Information Systems, Vol. 22, No. 1, January 2004.

[9] Rong Jin, Joyce Y. Chai, Luo Si. An Automatic Weighting Scheme for Collaborative Filtering. SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK. Copyright 2004 ACM.

[10] Kai Yu, Volker Tresp, Shipeng Yu. A Nonparametric Hierarchical Bayesian Framework for Information Filtering. SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK. Copyright 2004 ACM

[11] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, David M. Pennock. Methods and Metrics for Cold-Start Recommendations. SIGIR'02, August 11-15, 2002, Tampere, Finland. Copyright 2002 ACM

[12] Kai Yu, Wei-Ying Ma, Volker Tresp, Zhao Xu, Xiaofei He, HongJiang Zhang, Hans-Peter Kriegel. Knowing a Tree from the Forest: Art Image Retrieval using a Society of Profiles. MM'03, November 2–8, 2003, Berkeley, California, USA. Copyright 2003 ACM.

[13] Justin Basilico, Thomas Hofmann. Unifying Collaborative and Content-Based Filtering. Appearing in Proceedings of the 21 st International Conference on Machine Learning, Banff. Canada, 2004.

[14] Byoung-Tak Zhang and Young-Woo Seo. Personalized Web-Document Filtering Using Reinforcement Learning. Applied Artificial Intelligence, 15:665-685, 2001.

[15] Maria J. Martin-Bautista and Maria-Amparo Vila. Building adaptive user profiles by a genetic fuzzy classifier with feature selection. 0-7803-5877-5/00 IEEE.

[16] By Hyoung R. Kim and Philip K. Chan. Learning Implicit User Interest Hierarchy for Context in Personalization Proceedings of the 8th international conference on Intelligent user interfaces, 101 - 108 ACM, 2003

[17] Matthew R. McLaughlin and Jonathan L. Herlocker A Collaborative Filtering Algorithm and Evaluation Metric that Accurately Model the User Experience SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK. Copyright 2004 ACM 1-58113-881-4/04/0007.

[18] Mira Kwak, Dong-Sub Cho. Collaborative Filtering With Automatic Rating For Recommendation. 2001 IEEE.

[19] Beatrice Lazzerini, Francesco Marcelloni, Marco Cococcioni. A System Based on Hierarchical Fuzzy Clustering for Web Users Profiling, IEEE SMC 2003, Washington D.C., USA, October 2003, pp. 1995-2000.

[20] ZAN HUANG, HSINCHUN CHEN, and DANIEL ZENG. Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. ACM Transactions on Information Systems, Vol. 22, No. 1, January 2004.

[21] Costin Barbu, Marin Simina. Information Filtering Using the Dynamics of the User Profile. American Association for Artificial Intelligence 2002

[22] Qiubang Li and Rajiv Khosla. An Adaptive Algorithm for Improving Recommendation Quality of E-Recommendation Systems. 2003 IEEE

[23] Thomas P. Moran and Paul Dourish. Introduction to This Special Issue on Context-Aware Computing. Special Issue of Human-Computer Interaction, Volume 16, 2001

[24] Arkady Zaslavsky. Mobile Agents: Can They Assist with Context Awareness Proceedings of the 2004 IEEE International Conference on Mobile Data Management (MDM'04) 0-7695-2070-7/04

[25] Gellersen, H., Schmidt, A., Beigl, M. Multi-Sensor Context-Awareness in Mobile Devices and Smart Artefacts, Mobile Networks and Applications 7 (2002) 341–351

[26] Vagan Terziyan, Oleksandra Vitko. Bayesian Metanetworks for Modelling User Preferences in Mobile Environment, KI 2003, 370-384.

[27] Per Persson, Fredrik Espinoza, Petra Fagerberg, Anna Sandin and Rickard Cöster. GeoNotes: A Location-based Information System for Public Spaces. CHI '03 extended abstracts on Human factors in computing systems, 828-829, ACM, 2003

[28] Hee Eon Byun and Keith Cheverst. Exploiting User Models and Context-Awareness to Support Personal Daily Activities. Distributed Multimedia Research Group, Department of Computing, Lancaster University 2000.

[29] James D. Carswell, Alan Eustace, Keith Gardiner, Eoin Kilfeather, Marco Neumann. An Environment for Mobile Context-Based Hypermedia Retrieval. Proceedings of the 13th International Workshop on Database and Expert Systems Applications (DEXA'02) 1529-4188/02.

[30] Ian Li. a CAPpella: Prototyping Context-Aware Applications by Demonstration. Summer Undergraduate Program in Engineering Research at Berkeley http://www.eecs.berkeley.edu/Programs/ugrad/superb/papers2003/Ian%20li.pdf,11-2005

[31] John Canny. Some Techniques for Privacy in Ubicomp and Context-Aware Applications. Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information, 238-245, 2002

[32] Sushil J Louis, Anil Shankar. Context Learning Can Improve User Interaction. IEEE International Conference on Information Reuse and Integration (IEEE IRI-2004)

[33] Mauro Brunato and Roberto Battiti. PILGRIM:A Location Broker and Mobility-Aware Recommendation System p. 265, First IEEE International Conference on Pervasive Computing and Communications (PerCom'03), 2003

[34] Stephen S. Yau, Huan Liu, Dazhi Huang and Yisheng Yao. Situation-Aware Personalized Information Retrieval for Mobile Internet. Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC'03) 0730-3157/03

[35] Xuemin Shen, Jon W. Mark and Jun Ye. User mobility profile prediction: An adaptive fuzzy inference approach. Wireless Networks 6 (2000) 363–374 J.C. Baltzer AG, Science Publishers.

[36] M.Kyriakakos, S.Hadjiefthymiades, N.Frangiadakis, L.Merakos. Multi-user Driven Path Prediction Algorithm for Mobile Computing. Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA'03) 1529-4188/03 $17.00 © 2003 IEEE

[37] Stathes Hadjiefthymiades, Stamatis Papayiannis, and Lazaros Merakos, University of Athens. Using Path Prediction to Improve TCP Performance in Wireless/Mobile Communications. IEEE Communications Magazine August 2002

[38] Pornphan Dulyakarn and Yuttapong Rangsanseri. Fuzzy C-Means Clustering Using Spatial Information with Application to Remote Sensing. Paper presented at the 22nd Asian Conference on Remote Sensing, 5-9 November 2001, Singapore. Copyright (c) 2001, Center of Remote Imaging, Sensing and Processing (CRISP), National University of Singapore; Singapore Institute of Surveyors and Valuers (SISV); Asian Association on Remote Sensing (AARS).

[39] Bezdek, J.C. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York. 1981.

[40] Unsal, Cem. Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles Approach. A thesis submitted for the degree of doctor of philosophy in the Virginia Polytechnic Institute and State University, 1998.

[41] J.L. Herlocker, J.A. Konstan, L.G. Terveen and J.T. Rield. Evaluating Collaborative Filtering Recommender System. ACM Transactions on Information Systems, Vol. 22, No. 1, Pages 5–53 January 2004.

[42] Baker, J.E. Reducing Bias and Inefficiency in the Selection Algorithm. In [ICGA2], pp. 14-21, 1987.

[43] Amazon.com, Books, www.amazon.com, last date accessed, 20.12.2005

[44] NEC Research Institute, Steve Lawrence, Lee Giles and Kurt Bollacker, Citeseer.IST Scientific Literature Digital Library, http://citeseer.ist.psu.edu/, last date accessed, 20.12.2005

[45] S. J. Soltysiak and I. B. Crabtree, Automatic learning of user profiles - towards the personalisation of agent services. BT Technol J Vol 16 No 3 July 1998

APPENDIX A

THE RESULTS OF FCM WITH 21 CLUSTERS ON THE MAP

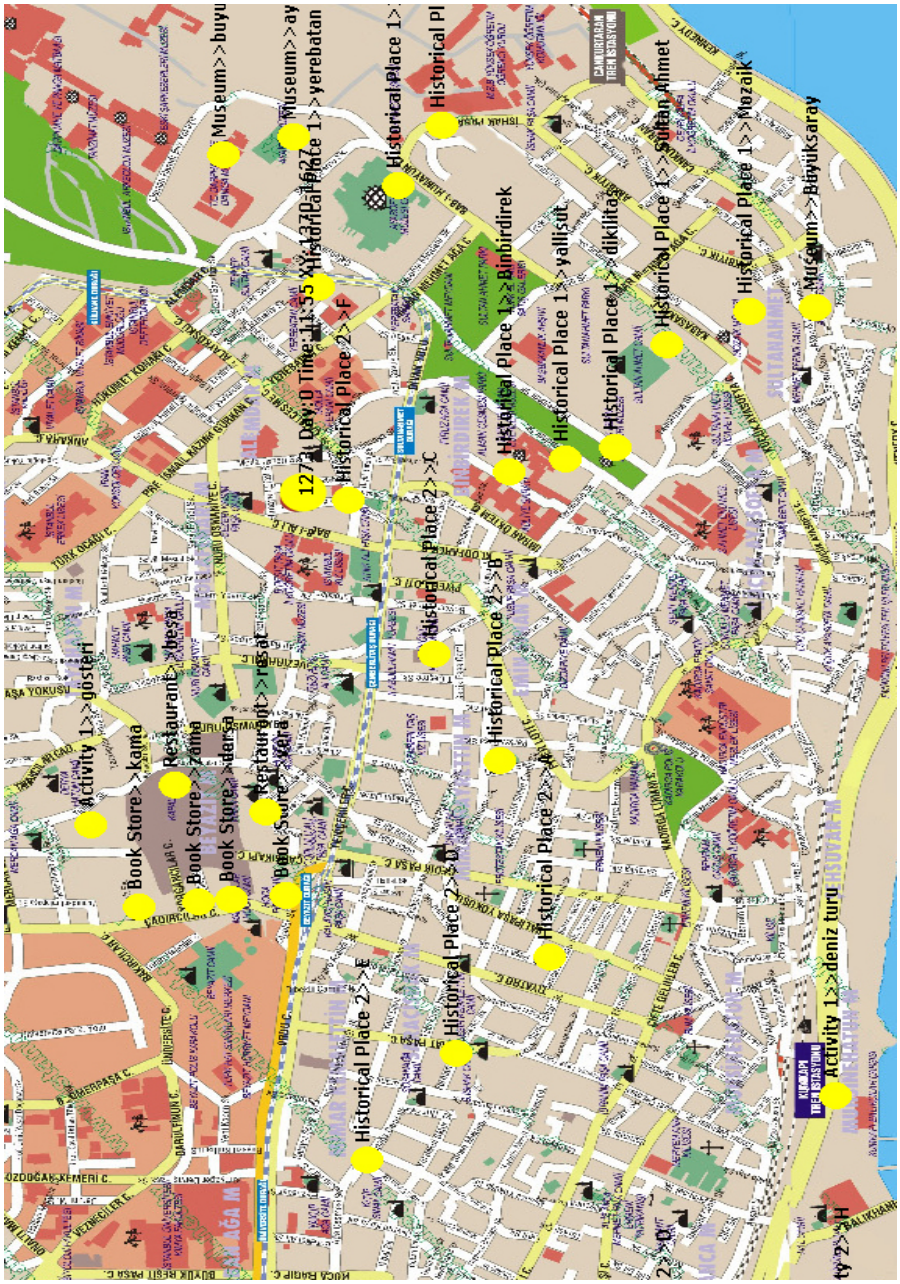

Figure A.1 Cluster represented with X:1370, Y:1627 on Monday at 11:55

Figure A.2 Cluster represented with X:1847, Y:290 on Monday at 13:45

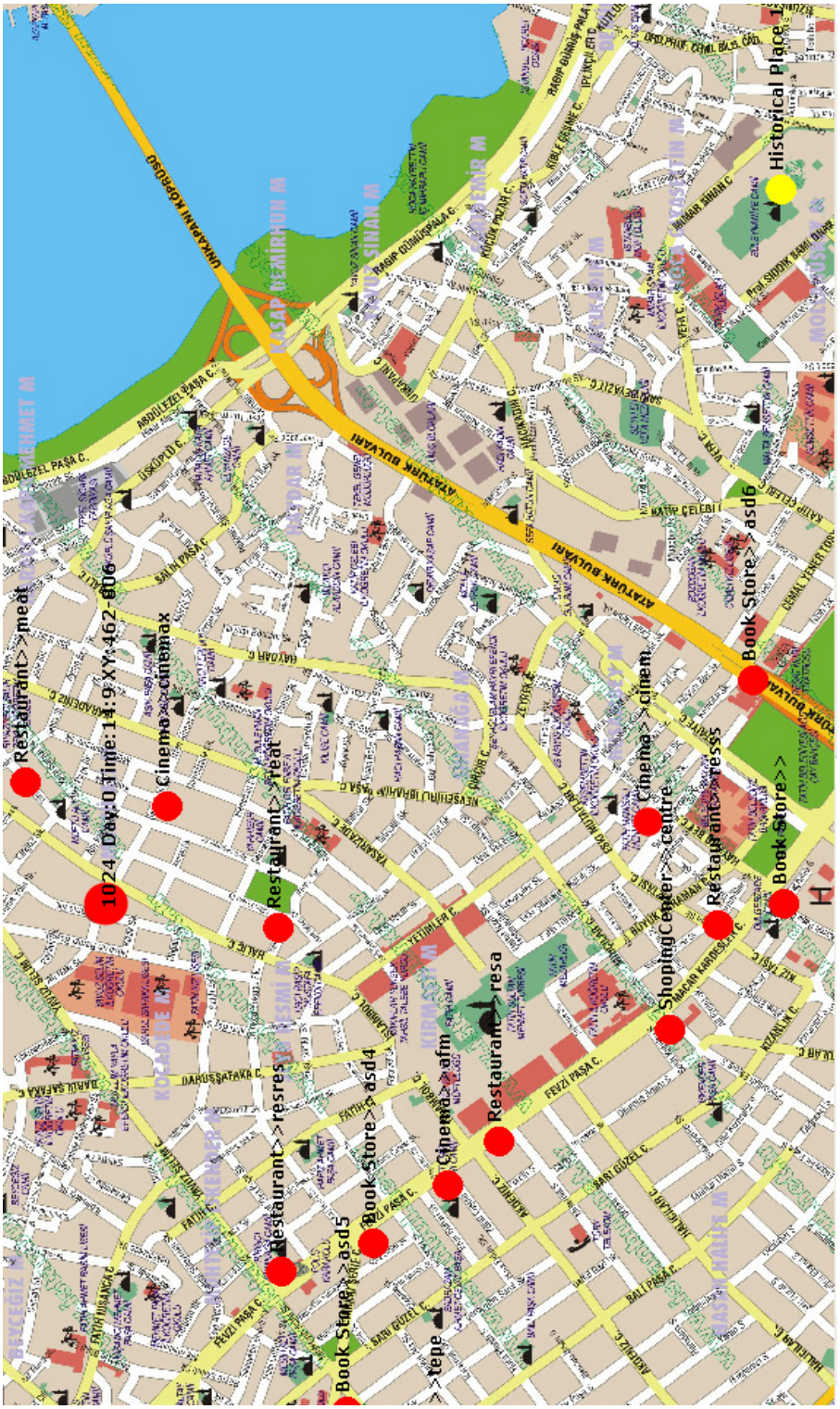

Figure A.3 Cluster represented with X:462, Y:806 on Monday at 14:09

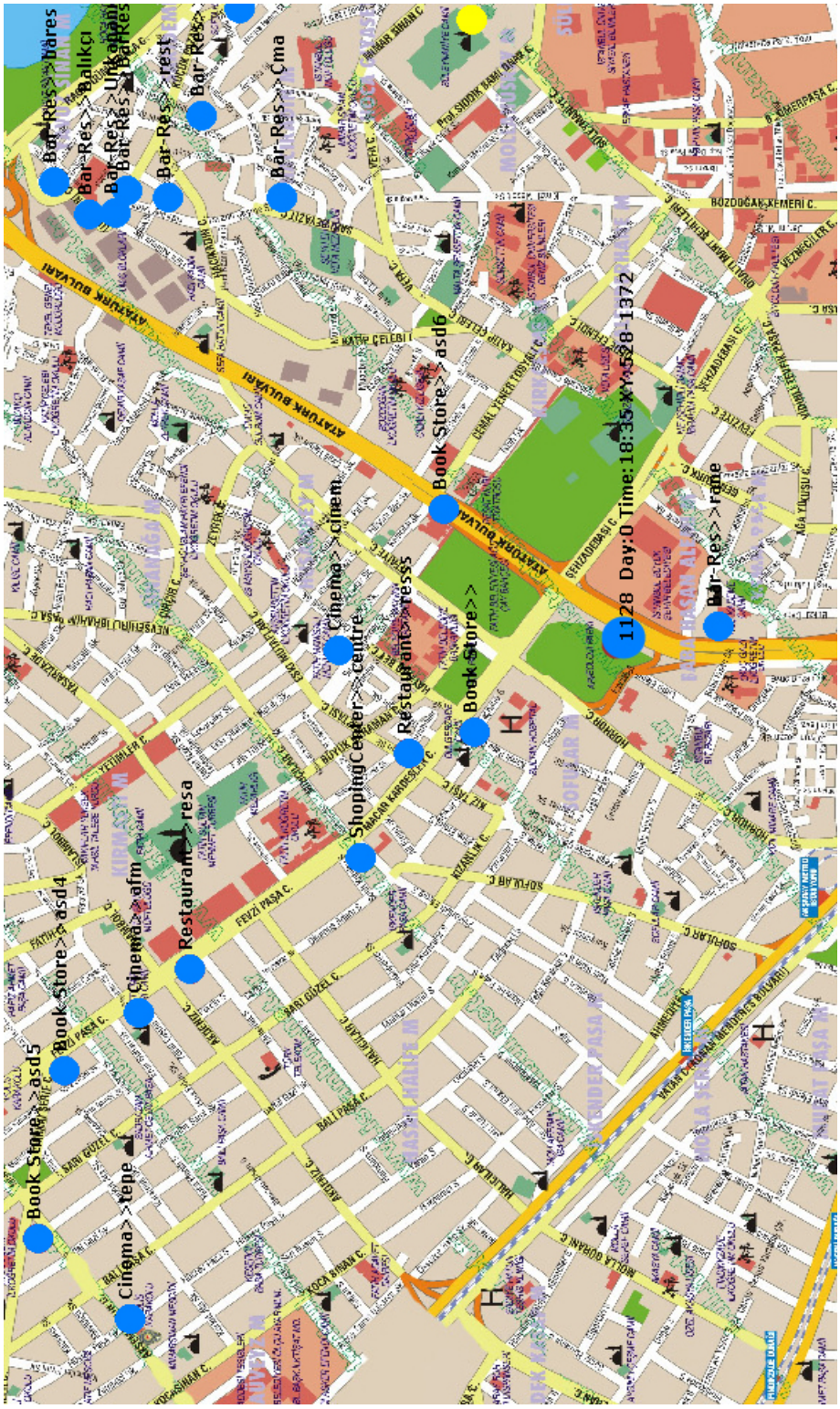

Figure A.4 Cluster represented with X:528, Y:1372 on Monday at 18:35