A Methodology for Predicting Agent Behavior by the Use of Data Mining Techniques

Andreas Symeonidis and Pericles Mitkas

Department of Electrical and Computer Engineering,
Aristotle University of Thessaloniki,
GR541 24 Thessaloniki, Greece
Tel.: +30-2310-996399, Fax: +30-2310-996398
asymeon@danae.ee.auth.gr, mitkas@eng.auth.gr

Abstract. One of the most interesting issues in agent technology has always been the modeling and enhancement of agent behavior. Numerous approaches exist, attempting to optimally reflect both the inner states, as well as the perceived environment of an agent, in order to provide it either with reactivity or proactivity. Within the context of this paper, an alternative methodology for enhancing agent behavior is presented. The core feature of this methodology is that it exploits knowledge extracted by the use of data mining techniques on historical data, data that describe the actions of agents within the MAS they reside. The main issues related to the design, development, and evaluation of such a methodology for predicting agent actions are discussed, while the basic concessions made to enable agent cooperation are outlined. We also present κ-Profile, a new data mining mechanism for discovering action profiles and for providing recommendations on agent actions. Finally, indicative experimental results are apposed and discussed.

1 Introduction

1.1 Web Personalization and Agent Behaviors

The core objective of agent action-based personalization is the discovery of a *recommendation set*, that will better predict the behavior of the agent "in action". Although such a topic is quite intriguing, going briefly through related bibliography, one can easily identify a lack of publications on agent action identification and prediction, since it is inherently complicated. However, this problem is very similar to web personalization, where a great number of research efforts have been published. By drawing an analogy to web personalization, agent behavior prediction may be feasible.

Rather popular approaches for web personalization include collaborative filtering [1],[2] and Web usage mining [3],[4],[5]. Advancing to more elaborate infrastructures, web usage mining systems have been built [6],[7] to discover interesting patterns in the navigational behavior of users [8]. Nevertheless, none of the above approaches had proven sufficient for providing successful personalization

V. Gorodetsky, J. Liu, and V.A. Skormin (Eds.): AIS-ADM 2005, LNAI 3505, pp. 161–174, 2005.

of users, and, correspondingly, of agents, until aggregation usage profiles were introduced. Several research groups [9],[10],[11] adopted this approach to identify association rules [12],[13], sequential patterns, pageview clusters and transaction clusters between users [9]. One of the most popular models for personalization and prediction, though, is the one proposed by Mobasher [14] and Mobasher, Cooley, & Srivastava [15],[16] and this is the model that has been adopted for the agent behavior prediction methodology to be built upon.

1.2 Agent Prediction System Prerequisites

The first task in such prediction systems is the collection of all necessary historical data. In order to predict the behavior of an agent, only information on its previous actions is needed. This is not necessarily the case, though, when creating behavior profiles. These profiles, which are extracted by the use of DM techniques, are the projection of "agent experience" on the data.

With respect to the dataset employed for predicting agent actions and for creating profiles, alternative architectures may occur for the prediction system. In a MAS, for example, each agent could maintain a history record for its own actions, in order to predict its own actions. In such a system, the predictions would be solely based on each agent's own "experience"; an agent with a no (limited) historical record, would, thus, produce no (poor) predictions. An alternative architecture could allow the historical records of all agents to be monitored by an appointed agent. Profiles would then be created on the whole of the agent action history, discarding nevertheless personalization. This is why we believe that a system for predicting agent actions should allow both personalization and collaboration between agents, in order to exploit other agents' experience, in case a agent specific cannot create good predictions.

In general, the main goal of a prediction system is the improvement of performance of a wider framework, which specifies the exact development principles and agent interactions of the prediction system.

Taking all these factors into account, a methodology for building a framework for predicting agent actions should comprise the following steps:

1. Model agent actions
2. Develop a mechanism for monitoring agent actions
3. Preprocess data, in order to incorporate domain understanding
4. Develop an appropriate DM algorithm for extracting agent action profiles
5. Develop a mechanism for storing and retrieving profiles
6. Develop an agent action recommendation interface

Figure 1 illustrates the functional architecture of such a system. It comprises two modules: a) an offline module, where action preprocessing, DM application, and profile creation takes place and, b) an online module, where the agent actions are recorded real-time and the recommendation interface is applied.

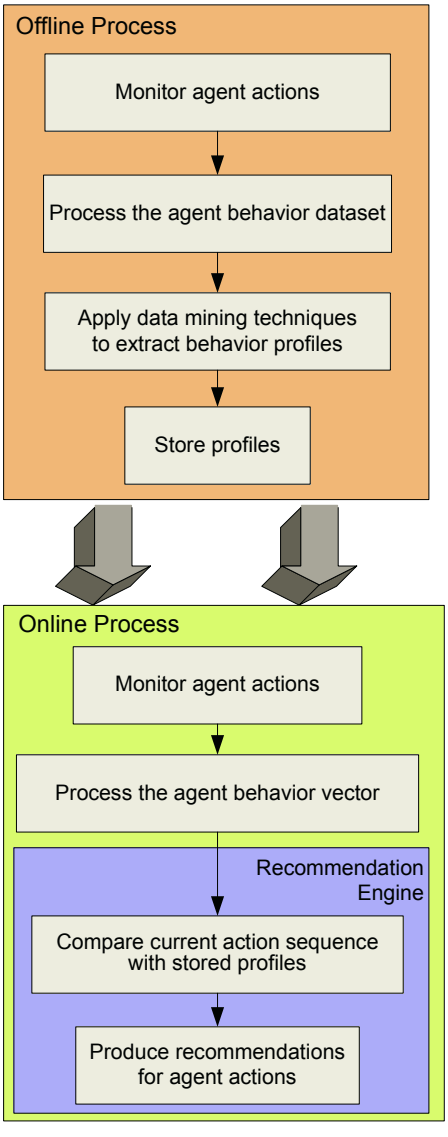

Fig. 1. The basic functionality of an agent prediction system

The rest of the paper is structured as follows: Section 2 deals with all issues related to the modeling of agent actions and presents the developed data mining mechanism. Section 3 presents the evaluation framework and the metrics used. Finally, section 4 provides indicative experimental results and concludes the paper.

2 Predicting Agent Behavior

2.1 The Prediction Mechanism

Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of possible actions that an agent may take during its execution phase. Parameter n varies from one operation cycle[1] to another. Let us also consider a set of m agent action bundles, $B = \{b_1, b_2, \ldots, b_m\}$, where each action bundle $b_i \in B$ is a subset of P, and describes the actions taken by an agent throughout one operation cycle. Since all agent action bundles are defined on the action space P, each vector b has a length n and is of the form:

$$b =< w(p_1, b), w(p_2, b), ..., w(p_n, b) > \quad (1)$$

where $w(p_i, b)$ is a weight associated with action $p_i \in P$ and $0 \leq w(p_i, b) \leq 1$.

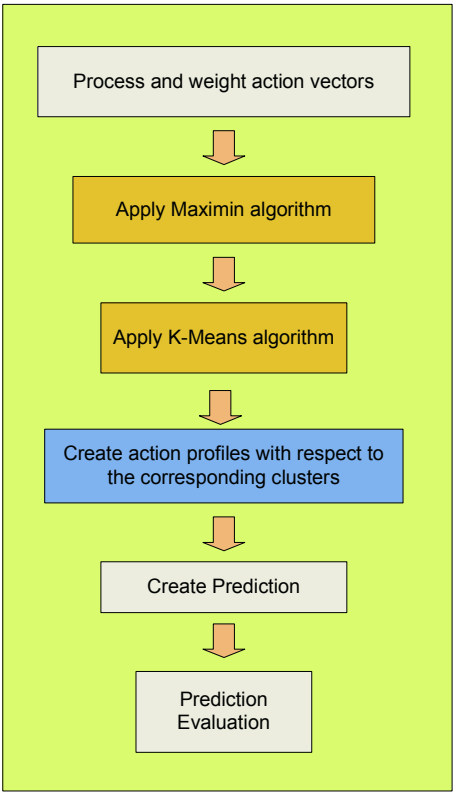

Fig. 2. The κ-Profile mechanism

[1] We define an operation cycle as the sequence of actions performed by an agent in its effort to accomplish the specific task it has been assigned to.

Let us consider a vector b that describes the actions taken by an agent within one operation cycle. The j^{th} component $w(p_j, b)$ of this vector will be 0, if the agent has never taken action p_j, while $w(p_j, b) \neq 0$ when the agent has taken action p_j. In fact, the value of $w(p_j, b)$ increases if the action under consideration is of great importance. Let us now consider an operation cycle that has not yet terminated. Its corresponding vector has, in general, more null elements than a vector of an already terminated cycle. The scope of the prediction mechanism is to determine the vector components (i.e., the agent actions) that have a high probability of occurrence in the remainder of an ongoing operation cycle. This capability will effectively narrow the options of an agent to the most suitable ones and, thus, reduce the time required to compute its next step. The weight calculated for each action denotes the probability of its appearance in the corresponding operation cycle.

A mechanism for discovering common action patterns, which we call κ-Profile is employed to extract the recommendations. The κ-Profile involves a sequence of steps shown in Figure 2. It utilizes the Maximin [17] and K-Means [18] algorithms, in order to cluster the set of vectors B.

After clustering has been performed, a set of action bundle clusters, $BC = \{c_1, c_2, ..., c_k\}$ is extracted, where each c_i is a subset of B. For each action bundle cluster $c_i \in BC$, a representative vector is defined as the **cluster profile** cp_i. Vector cp_i is the vector closer to the cluster center. The components of cp_i that fall below a certain threshold μ are nullified. This way, only the most representative of the events in the cluster are taken into account. In the case of cluster c_i, for example, the profile cp_i is the set of $< action - weight >$ pairs, as defined by Equation 2:

$$cp_i = \{< p, weight(p, cp_i) > \mid p \in P \text{ and } weight(p, cp_i) \geq \mu\} \tag{2}$$

The $weight(p, cp_i)$ of action p within profile cp_i is calculated using Equation 3:

$$weight(p, cp_i) = \frac{1}{|c_i|} \sum_{b \in c_i} w(p, b) \tag{3}$$

where $w(p, b)$ is the weight of action p in action bundle $b \in c_i$. It is, therefore, obvious that each profile can be also represented as a vector of length n. Considering that the same representation mechanism is employed for the ongoing operation cycle vector, both profiles and action bundles can be manipulated as n-dimensional vectors in the action space. For example, a profile C, can be represented as:

$$C = \left\{w_1^C, w_2^C, ..., w_n^C\right\} \tag{4}$$

where

$$w_i^C = \begin{cases} weight(p_i, C), & p_i \in C \\ 0, & otherwise \end{cases} \tag{5}$$

Similarly, the ongoing operation cycle can be represented as a vector $S =< s_1, s_2, ..., s_n >$, where s_i is a weight denoting the significance of action p_i in the current cycle. The weighted values are calculated in the same manner as

the weights for action vectors b are calculated (e.g. $s_i = 0$ if the agent has not taken action p_i). In both cases, a *Fuzzy Inference System* (FIS) is employed to produce the weights, which enables the incorporation of domain understanding into the prediction mechanism. The comparison of a certain profile with the vector representing the ongoing operation cycle is performed using the cosine coefficient, a metric widely used in information retrieval problems. $match(S, C)$, defined in Equation 6, calculates the cosine of the angle of the two vectors S and C, by normalizing their dot product with respect to their moduli.

$$match(S, C) = \frac{\sum_k w_k^C \cdot s_k}{\sqrt{\sum_k (s_k)^2 \times \sum_k (w_k^C)^2}} \tag{6}$$

The actions that the prediction system recommends are determined through a *recommendation score*, defined for each action p_i in each of the already calculated profile vectors. This score is dependent on two factors:

1. The overall similarity of the current vector to the profile, and
2. The average weight of each action p_i in the profile

Given a profile C and an operation cycle vector S, the recommendation score $Rec(S, p_i)$ is calculated for each action p_i, according to Equation 7.

$$Rec(S, p_i) = \sqrt{weight(p_i, C) \cdot match(S, C)} \tag{7}$$

Finally, a Next Action Recommendation set, $NAR(S)$, is compiled for the current action bundle S, containing only actions with recommendation scores that exceed a certain threshold, ρ, for all profiles. That is:

$$NAR(S) = \left\{ p_i \mid \mathrm{Rec}(s, w_i^C) \geq \rho \right\} \tag{8}$$

For each action appearing in more than one vectors, the maximum recommendation score is selected, from the corresponding profile. This way optimal coverage is achieved. Table 1 illustrates an example of a recommendation on action vector S, based on profile C. Recommendation is produced only on vector components that have a null value on S and a non-null value on C. For action p_2, for example, a recommendation score on S is calculated, according to Equation 7. In this case, $Rec(S, p_i) = \sqrt{0.375 \cdot match(S, C)}$, where $match(S, C) = 0.52$. Finally, $Rec(S, p_2) = 0.442$.

2.2 Applying κ-Profile on MAS

The κ-Profile mechanism aims to predict future agent actions within an operation cycle, based on knowledge of prior actions of this and/or similar agents. For the ongoing operation cycle, a z-size window is employed. That is, only the last z actions of the agent can influence the outcome of the recommendation process. κ-Profile can be easily adapted to predict agent behavior, mapping the vector elements to agent actions.

Table 1. Recommending the next action

Actions P	Profile Vector C	Vector S	$Rec(S, p_i)$
p_1	0	0	0
p_2	0.375	0	0.442
p_3	0	0	0
p_4	0.3	0.6	0.395
p_5	0	0.4	0

Let us consider an agent authorized to execute a number of functions on files, as determined by its action space $P = \{Select, Open, Modify, Save, Close\}$. Binary weights are assigned to the elements of the action vectors. Let us consider a profile $C = \{Open, Modify, Save\}$. The corresponding profile vector is, thus, $cp_C = [0\ 1\ 1\ 1\ 0]$. In case actions $\{Open\}$ and $\{Modify\}$ are executed during the current cycle and the prediction window has been set to $z = 2$, action $\{Save\}$ will be recommended, according to profile C (Table 2).

Table 2. An example on predicting the next action

Action	Profile	Recent History
Select	0	0
Open	1	1
Change	1	1
Save	1	0
Close	0	0

Although excluding the two actions $\{Select\}$ and $\{Close\}$ from a set of five possible actions does not seem interesting, an equivalent reduction of the candidate space in a system with a large number ($n \geq 100$) of options would be rather significant. κ-Profile provides this pruning mechanism through the grouping of action bundles into clusters and the identification of actions that are likely to occur next. These actions are determined by their degree of participation into the profile(s) taken into account, and by the similarity of the ongoing operation cycle vector to it(them).

In our example, the participation degree of action $\{Save\}$ is 1. This action is proposed to be the next action for the ongoing operation cycle vector with an $1\times\{vectorsimilarity\}$ similarity measure. The similarity measure is always < 1, since the current action bundle is generally different from the profile used for prediction. It should be denoted that the quality of the prediction is also related to the size of the historical dataset (the bigger the historical dataset, the better the prediction). A bigger dataset offers more training options, often leading to more accurate predictions.

In a sense, κ-Profile produces a type of association rules. For the current example, the rule would be:

$$\{Open\} \wedge \{Change\} \longrightarrow \{Save\} \tag{9}$$

Nevertheless, in the case of association rules, their participation degree (confidence) could be equal to 1, expressing certainty on the occurrence of the action.

The representation mechanism in our example is quite simple, since the weighted values are either 0s or 1s (the action has not/has been taken). In general, the mechanism is much more complicated and several parameters need to be considered (i.e., action frequency, action timing, etc.). The major advantage of κ-Profile is that it can manipulate vectors with fuzzy constituent values (fuzzy degrees of participation). In that case, the representation mechanism assigns values within the specified fuzzy interval, a flexibility that makes the κ-Profile mechanism suitable for a wide area of applications. Important issues that deserve further study include the agent action model and the specification of the operation cycle goal. These issues are discussed next.

2.3 Modeling Agent Actions in an Operation Cycle

Let us consider some multi-agent application and let $P = \{p_1, p_2, \ldots, p_n\}$ be a finite set of possible agent actions. In general, agent actions are asynchronous events occurring at times T_i. The time interval between two actions is, in most cases, of great importance. In the case of a web application, for example, the time interval between two successive page visits is the time the user spent on the first site (possibly exposed to electronic advertising). In order to monitor an agent operation cycle we use an ordered, variable-length vector Π, whose elements are pairs of the form $< action, executiontime >$. The time intervals between consecutive actions can be easily calculated. In fact, proper processing of Π can produce useful information for the operation cycle.

As an example, let us consider a system where the set of possible actions is $P = \{p_1, p_2, p_3\}$. Figure is a representation of an operation cycle with five agent actions occurring at times T_0 to T_4. Table 3 shows vector Π for this example.

Table 3. A vector representing the operation cycle

Vector Π
$< p_1, T_0 >$
$< p_3, T_1 >$
$< p_2, T_2 >$
$< p_3, T_3 >$
$< p_1, T_4 >$

According to the previous analysis, two things have to be determined, as far as the operation goal is concerned: a) the goal itself and, b) a terminating condition for the operation cycle. For an internet-based MAS, the operation goal

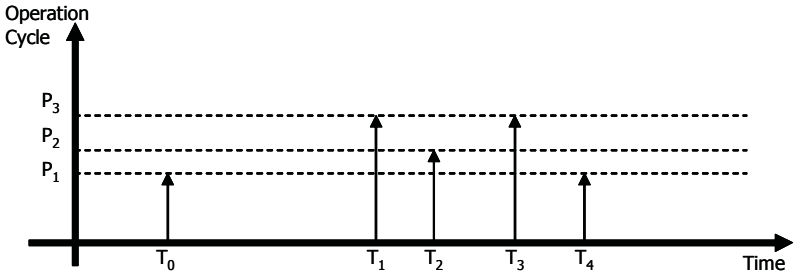

Fig. 3. The evolution of an operation cycle

would be the transition of an agent to one of the available web pages of the site (each transition is considered to be an action). In this case, the prediction mechanism can be quite straightforward, because the operation cycle itself has to be predicted and no terminating condition needs to be specified. During the online process, only an action window (z) is needed for the system to predict. This is not the case, though, for the offline process. where operation cycle vectors must be created and stored. Since the profile vectors must be of equal length, a terminating condition is needed. This condition can be related to either time or change of application status, or even both. Thorough analysis has led to the decision that the optimal choice in such systems with "predicting" capabilities would be the definition of an agent action signaling the end of the current operation cycle. In the web traversing example, the terminating action could be the end of the user's web navigation.

2.4 Mapping Agent Actions to Vectors

Another important issue for the prediction mechanism is the transformation of the operation cycles to agent action vectors. κ-Profile mechanism requires equal-size vectors, with their elements (weights) valued in the [0,1] interval (Table 4). Binary weights could be assigned in the simplest of cases, merely denoting the execution or not of an action in a specific operation cycle. This approach is not suitable, however, for the cases when the same action is executed more than once within the same operation cycle.

Table 4. Mapping agent actions to vectors

Vector π		Weighted Vector
$< p_1, T_0 >$		β_1
$< p_3, T_1 >$	$\longrightarrow$	β_2
$< p_2, T_2 >$		β_3
$< p_3, T_3 >$		β_4

It should be emphasized that the representation mechanism is closely related to the scope of the specific MAS. In a MAS focused on the analysis of consumer

behavior, for example, the items and their quantities purchased within the same transaction would be of interest. In other types of applications, the time interval between actions is important (e.g. the browsing duration of a specific web page). It is, therefore, obvious that thorough analysis is is required in order to achieve the proper modeling of the problem at hand.

The κ-Profile has been designed to recommend only actions of high importance. The mechanism perceives the importance of an action within the operation cycle through its weight (the corresponding element's weight). The action weights are calculated by the use of a FIS, that deals successfully with the issue of incorporating domain understanding. During fuzzification, the related parameters have to be defined and tuned, while the fuzzy rule base has to be created.

If the prediction system is developed following the methodology presented in Chapter 5, the only parameters that have to be defined are:

- The finite set of actions P
- The goal of the operation cycle and the terminating condition, and
- The parameters of the FIS, and their influence on the MAS performance.

3 Evaluating Efficiency

The efficiency of improving agent intelligence through data mining on agent behavior data can be measured at two levels:

a) the profile level, and
b) the prediction level

The behavior profiles extracted by the κ-Profile have to undoubtedly represent related behaviors among agents acting within the same application. If the profiles are successful, then **information personalization** can be achieved, which is of great importance to the performance of the system.

3.1 Profile Efficiency Evaluation

In order to evaluate profile efficiency, we can follow the analysis proposed by Perkowitz and Etzioni [5]. According to this analysis, an agent that has taken an action of the profile, will also take another action of the same profile, within the ongoing operation cycle. That is, if B is the set of profile vectors and cp_i is a profile, then B_{cp_i} is a subset of B, whose elements (b_j) contain at least one of the actions in cp_i. First, we must calculate the *average visit percentage* (AVP) for extracted profiles. The AVP of profile cp_i with respect to all the profile vectors is in this case given by Equation 10.

$$AVP = \sum_{b \in B_{cp_i}} \frac{(\overrightarrow{b} \cdot \overrightarrow{cp_i})}{|b|} \tag{10}$$

The *weighted average visit percentage*, $WAVP$ metric is then calculated by dividing AVP with the sum of all profile pr elements' weights (Equation 11).

$$WAVP = \frac{\left(\sum_{b \in B_{cpi}} \frac{\overrightarrow{b} \cdot \overrightarrow{cp_i}}{|b|} \right)}{\left(\sum_{p \in cp_i} weight(p, cp_i) \right)} \tag{11}$$

Values of $WAVP$ closer to 1 indicate better profile efficiency.

3.2 Prediction System Efficiency Evaluation

Several different approaches can be followed, in order to evaluate the efficiency of a prediction system. Nevertheless, the primitives of a MAS impose an approach that is somewhat different to the ones used for estimation and classification systems. A recommendation engine, like the one described, does not require the rigidity of such systems. More than one suggestions, equally important, can be produced. Therefore, the prediction itself cannot be considered as a valid evaluation metric, and this the reason for not employing the classic evaluation metrics, *precision* and *coverage*.

Instead, the following approach has been adopted:

Let $AS = s_1, s_2, ..., s_n$ be the test set, i.e. a set of action vectors that were not used during the application of the prediction mechanism on the initial data (training set). Let R be the set of actions proposed by the prediction system for action s. In the case $R \cap s \neq 0$, then we consider the recommendation to be a success. The metric for evaluating the efficiency of the recommendation system is then $PSSR$ (Prediction System Success Rate), which is defined as the percentage of successful recommendations ($SRec$) to the sum of recommendations made ($ORec$):

$$PSSR = \frac{SRec}{ORec} \tag{12}$$

Values of $PSSR$ closer to 1 indicate a better prediction efficiency.

4 Experimental Results

In order to demonstrate the added-value of the developed prediction mechanism and the diffusion of knowledge extracted by the use of DM techniques on agent behavior data, we have incorporated κ-Profile into the DM module[2] of the Agent Academy [20], a platform for building multi-agent systems and for enhancing their intelligence by the use of data mining techniques. The dataset used contained web traversing data, provided by a large web site. In our demonstrator, the action was defined as the transition to one of the available pages

[2] Data Miner [19] is one of the core components of the Agent Academy platform. It can also operate as a stand-alone DM suite, incorporating domain knowledge and supporting semantics. Available at: http://sourceforge.net/projects/aadataminer.

in the web site, while the parameters that were considered of importance (*time* and $frequency$)were defined appropriately.

Each one of the web site pages mapped to an agent action. The web site used has 142 discrete web pages, i.e. 142 possible agent actions. The set of action bundle vectors, B, contained 208 elements (b vectors). This set was provided as the training set to the κ-Profile mechanism. At first the *maximin* algorithm identified $K = 8$ clusters, and along with the most representative vectors of each cluster, the $K - Means$ algorithm was applied. The resulting clusters, along with the corresponding set percentage, are illustrated in Table 5. As can be easily seen, some of the produced clusters have very low vector membership. This is a result of the layout of the specific web site.

Table 5. The resulting vector clusters and their percentage distribution

Cluster	Vector
Cluster 1	56.73%
Cluster 2	4.33%
Cluster 3	∼2%
Cluster 4	∼3%
Cluster 5	24.5%
Cluster 6	2.4%
Cluster 7	1.44%
Cluster 8	4.33%

Using the resulting clusters, κ-Profile has identified the most representative actions, therefore constructing eight action clusters, which in fact comprise the agent profile set. Based on these profiles, the recommendation engine produced, in real-time, the agent suggestions. Table 6 illustrates the actions that comprise the profile of cluster 4, along with their normalized weights within the cluster.

The $WAVP$ metric was applied on the extracted profiles, in order to identify their efficiency. The $WAVP$ metric was calculated to be 78% for the first of the eight profiles extracted, 72.5% for the second profile, while for the rest of the profiles it stayed above a respectable 67%.

As far as the efficiency of the prediction mechanism is concerned, the *PSSR* metric was calculated to be 72.14%, a success rate that was considered satisfactory, since the action space was $n = 142$, while the maximum profile size was $m = 8$ (8 actions maximum). Taking under consideration the fact that the recommendation window was set to $z = 3$ (last three actions of the agent), in almost three out of four cases, the prediction mechanism proposed an action that the agent subsequently followed (a web page that the user chose to visit).

Table 6. The actions that comprise the profile of cluster 4

Action	Vector
p_{67}	0.9998
p_{86}	0.9999
p_{15}	0.8352
p_{82}	0.7827
p_{13}	1.0
p_{11}	0.9788
p_{10}	0.7273
p_{9}	0.8992
p_{100}	0.8264
p_{77}	0.7892
p_{8}	0.9999
p_{76}	1.0
p_{7}	0.9999
p_{4}	0.8307

5 Conclusions and Future Work

The main objective of this paper was the analysis of all issues involved in the development (through the methodology presented) of a system that exploits the results of data mining on agent behavior data, in order to predict their posterior actions. Through the analysis conducted, we have shown that data mining techniques can, indeed, be exploited for discovering common behavioral patterns between agents, as long as the problem is modeled appropriately and the system infrastructure entails the features presented in section 1. This way data preprocessing is possible and a suitable DM mechanism can be applied, for agent behavior profiles to be extracted. From that point on, agent behavior can be predicted.

References

1. Konstan, J.A., Miller, B.N., Maltz, D, Herlocker, J.L., Gordon, L.R., Riedl, J: GroupLens: Applying Collaborative Filtering to Usenet News. Communications of the ACM. **40** (1997) 77–87
2. Herlocker, J.L., Konstan, J.A., Borchers, B., Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Theoretical Models (1999) 230–237
3. Nasraoui, O., Frigui, H., Joshi, A., Krishnapuram, R.: Mining Web Access Logs Using Relational Competitive Fuzzy Clustering. In: Proceedings of the Eight International Fuzzy Systems Association World Congress. (1999)
4. Spiliopoulou, M., Pohle, C., Faulstich, L.: Improving the Effectiveness of a Web Site with Web Usage Mining. In: WEBKDD. (1999) 142–162

5. Perkowitz, M., Etzioni, O.: Adaptive Web Sites: Automatically Synthesizing Web Pages. In: AAAI/IAAI. (1998) 727–732
6. Cooley, R., Mobasher, B., Srivastava, J.: Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information Systems **1** (1999) 5–32
7. Zaïane, O.R., Xin, M., Han, J.: Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. In: Advances in Digital Libraries. (1998) 19–29
8. Mobasher, B., Dai, H., Luo, T., Nakagawa, M., Witshire, J.: Discovery of aggregate usage profiles for web personalization. In: Proceedings of the WebKDD Workshop. (2000)
9. Shahabi, C., Zarkesh, A.M., Adibi, J., Shah, V.: Knowledge Discovery from Users Web-Page Navigation. In: RIDE. (1997)
10. Schechter, S., Krishnan, M., Smith, S.: Using path profiles to predict HTTP requests. Computer Networks and ISDN Systems. **30** (1998) 457–467
11. Banerjee, A., Ghosh, J.: Clickstream Clustering using Weighted Longest Common Subsequences. (2001)
12. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In Bocca, J.B., Jarke, M., Zaniolo, C., eds.: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, Morgan Kaufmann (1994) 487–499
13. Agarwal, R.C., Aggarwal, C.C., Prasad, V. V. V.: A Tree Projection Algorithm for Generation of Frequent Item Sets. Journal of Parallel and Distributed Computing **61** (2001) 350–371
14. Mobasher, B.: A Web personalization engine based on user transaction clustering. In: Proceedings of the 9th Workshop on Information Technologies and Systems. (1999)
15. Mobasher, B., Srivastava, J., Cooley, C.: Creating Adaptive Web Sites Through Usage-Based Clustering of URLs. (1999)
16. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on Web usage mining. Communications of the ACM. **43** (2000) 142–151
17. Looney, C. G.: Pattern Recognition Using Neural Networks: Theory and Algorithms for Engineers and Scientists. Oxford University Press (1997)
18. McQueen, J.B.: Some Methods of Classification and Analysis of Multivariate Observations. In Cam, L.M.L., Neyman, J., eds.: Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability. (1967) 281–297
19. Symeonidis, A.L., Mitkas, P.A., Kehagias, D.: Mining patterns and rules for improving agent intelligence through an integrated multi-agent platform. In: Proceedings of the 6th IASTED International Conference on Artificial Intelligence and Soft Computing. (2002)
20. Mitkas, P.A., Kehagias, D., Symeonidis, A.L., Athanasiadis, I.: A Framework for Constructing Multi-Agent Applications and Training Intelligent Agents. In: Proceedings of the 4th International Workshop on Agent-Oriented Software Engineering, Springer-Verlag (2003) 1–16