Automatic Recognition of Learner Groups in Exploratory Learning Environments

Saleema Amershi and Cristina Conati

Dept. of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, BC, V6T 1Z4, Canada
{samershi, conati}@cs.ubc.ca

Abstract. In this paper, we present the application of unsupervised learning techniques to automatically recognize behaviors that may be detrimental to learning during interaction with an Exploratory Learning Environment (ELE). First, we describe how we use the *k*-means clustering algorithm for off-line identification of learner groups with distinguishing interaction patterns who also show similar learning improvements with an ELE. We then discuss how a *k*-means on-line classifier, trained with the learner groups detected off-line, can be used for adaptive support in ELEs. We aim to show the value of a data-based approach for recognizing learners as an alternative to knowledge-based approaches that tend to be complex and time-consuming even for domain experts, especially in highly unstructured ELEs.

1 Introduction

Exploratory learning environments (ELEs) provide facilities for student-led exploration of a target domain with the premise that active discovery of knowledge promotes deeper understandings than more controlled instruction. Through the use of graphs and animations, algorithm visualization (AV) systems aim to better demonstrate algorithm dynamics than traditionally static media, and there has been interest in using them within ELEs to promote interactive learning of algorithms [1,8]. Despite theories and intuitions behind AVs and ELEs, reports on their pedagogical effectiveness has been mixed [5,8]. Research suggests that their pedagogical effectiveness depends upon the way in which these systems are used, which in turn is influenced by distinguishing student characteristics such as meta-cognitive abilities [5] and learning styles [11,8]. For example, some students often find such unstructured environments difficult to navigate effectively and so they may not learn well with them.

Such findings highlight the need for ELEs in general, and specifically for ELEs that use interactive AVs, to provide adaptive support for students with diverse abilities or learning styles. This is a challenging goal because of the difficulty in observing distinct student behaviors in such highly unstructured environments. The few efforts made towards this goal mostly rely on hand-constructing detailed student models that can monitor student behaviors, assess individual needs, and inform adaptive help facilities. This is a complex and time-consuming task that typically requires the collaborative efforts of domain, application and model experts. For example, Bunt et al. [5] hand-constructed a Bayesian network to model student exploration and

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 463 – 472, 2006.

understanding in the Adaptive Coach for Exploration, an ELE for mathematical functions. Model construction required enumerating all possible exploration cases and domain concepts in the form of network nodes, specifying all node interdependencies, and manually estimating multi-valued conditional probability tables for each node. Though the model can be used to provide students with personalized help in exploration, this entire process would have to be repeated for each new application. In the absence of experts or in large applications, the difficulties of constructing such models can be exacerbated.

To circumvent the difficulties in hand-constructing models, some researchers have turned to the field of machine learning [4,2]. Typically, supervised learning algorithms are used to approximate functions that map observable input data to observable output data such as the correctness of student answers [4]. These functions can then predict student behavior and inform adaptive facilities. However, when output values are unobservable, domain experts must resort back to manual labeling to supply them [2]. This is again time-consuming and can be error prone.

We explore an alternative method for informing adaptive support for ELEs by employing *k*-means [6], an unsupervised machine learning technique to recognize patterns of student behaviors that may affect learning. Because ELEs provide unconstrained environments for exploration, the space of user behaviors can be very large, and characteristic behaviors of different learner types may not be obvious or easily observable even by application experts. For this reason we use *k*-means clustering for the automatic, off-line recognition of user groups. Once groups are detected, we analyze similarities within and dissimilarities between them in terms of both learning and interface usage, to show that clustering identifies meaningful patterns along these two dimensions. Then, we show how these distinct learner groups can be used for on-line classification of individual users. The long-term goal is automatic interface adaptation to encourage effective behaviors and prevent detrimental ones.

We begin by discussing related work. Next, we illustrate the ELE and the experimental data we use. Then, we describe and present results on the use of *k*-means for both the off-line identification of learner groups as well as for on-line recognition. We conclude with a summary and suggestions for future research.

2 Related Work

Gorniak and Poole [7] identify several issues concerning the development of sophisticated user or application models for intelligent user interfaces, including limited transferability across applications and effort required to hand construct the models. They address these concerns by learning a stochastic state space model of application use from user interactions with an earlier version of the same pedagogical tool that we use here, the CIspace Constraint Satisfaction Problem applet (see Section 3). The model they propose can predict future user actions, but unlike our work, it does not allow for assessing quality or relevance of these actions for the interaction goals.

Closely related to our work is research in the emerging field of educational data mining (e.g. [3]). For example, in [9] the authors cluster student responses to multiple choice tests and then analyze the clusters to assess student understanding and misconceptions. Our work differs in that we aim to cluster on higher dimensional interaction

behaviors as opposed to answers to questions, as ELEs avoid this type of structured learning tasks by nature. In [12], the authors use clustering on behaviors, specifically action frequencies of students using a collaborative learning tool. These clusters provide behavioral summaries to instructors who can then interpret the results to manage student collaborations. In our research, we take into account temporal data as well as activity frequency when clustering interactions. We also take this process one step further by demonstrating how detected clusters can be used to provide automatic, online adaptive support.

3 Experimental Data

The ELE we use as a testbed for our approach is the Constraint Satisfaction Problem (CSP) Applet, one of a collection of interactive AV tools for learning common Artificial Intelligence algorithms called CIspace [1]. Algorithm dynamics are interactively demonstrated on graphs by the use of color and highlighting, and graphical state changes are reinforced through textual messages (see Figure 1 for an example).

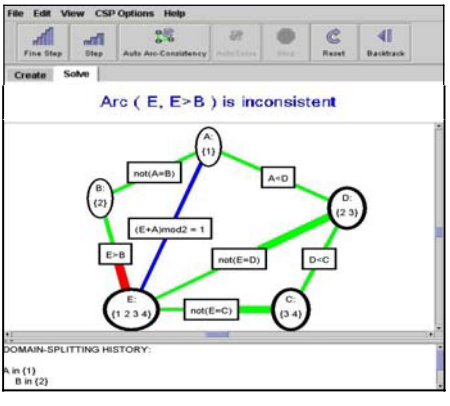

Fig. 1. CSP applet with example CSP

A CSP consists of a set of variables, variable domains and a set of constraints on legal variable-value assignments. The goal is to find an assignment that satisfies all constraints. The CSP applet illustrates the Arc Consistency 3 (AC-3) algorithm for solving CSPs [10] represented as networks of variable nodes and constraint arcs. AC-3 iteratively makes individual arcs consistent by removing variable domain values inconsistent with a given constraint until all arcs have been considered and the network is consistent. Then, if there remains a variable with more than one domain value, a procedure called domain splitting can be applied to that variable to split the CSP into disjoint cases so that AC-3 can recursively solve each case or sub-network.

The CSP applet provides several mechanisms for interactive execution of the AC-3 algorithm, accessible through the toolbar shown at the top of Figure 1 or through direct manipulation of graph elements. These mechanisms include:

- *Fine Stepping*. Cycles through three detailed algorithm steps: selecting an arc, testing it for consistency, and removing variable domain values when necessary.
- *Direct Arc Clicking*. Allows the user to decide which arc to test, and then performs three *Fine Steps* on that arc to make it consistent.
- *Auto Arc Consistency (Auto AC)*. Automatically *Fine Steps* through the network.
- *Stop*. Stops *Auto AC*.
- *Domain Splitting (DS)*. Allows the user to select a variable domain to split, and specify a sub-network for further application of AC-3.
- *Backtracking*. Recovers the alternative sub-network set aside by *DS*.
- *Resetting*. Resets the CSP network to its initial state.

The data we use for this research was obtained from a previous experiment investigating the effects of studying sample problems with the CSP applet [1]. The experiment typified a study scenario in which students first learn from text based materials, and then study relevant sample problems with the applet. We use the following data collected from 24 students who participated in the study: time-stamped logs of user interactions with the applet, and learning gains computed from pre and post test scores. From the logged data we obtained 1931 actions of users over 205.3 minutes.

4 Behavior Recognition Through *K*-Means Cluster Analysis

Clustering is a class of machine learning techniques used for automatically recognizing patterns in unlabelled data. Clustering operates on data points (feature vectors) in a feature space. Features can be any measurable property of the data. Similarities correspond to distances between data points in the feature space. We use a popular clustering algorithm, *k*-means [6], on our experimental data.

K-means clustering takes as input feature vectors and a user-specified *k* value, and returns *k* clusters of feature vectors. From our logged data, we have 24 feature vectors corresponding to the 24 study participants. Typically the *k* value is determined by intuition about the data or through cross-validation. We experimented with *k* set to 2 and 3 because our data set was relatively small and so we expected to find only a few clear groups with distinct learning outcomes.

Initially, *k*-means randomly selects *k* data points to be the current cluster means. The remaining data points are then assigned to the cluster whose mean minimizes some specified distance metric. Here we minimize Euclidean distances in a 21 dimensional, normalized feature space where the dimensions are the average frequencies of use, and the mean and standard deviations of the pause durations after use of the seven mechanisms described in Section 3. The two latter dimensions have been chosen to capture both the speed of use (which could indicate student attention) and its selectiveness, since varied speed may indicate planned rather than impulsive or inattentive behavior and may not be as detrimental for learning. After all data points are assigned to a cluster, new cluster means are computed from these groupings. The process then repeats for a given number of iterations or until there are little or no changes in the clusters.

K-means can often converge at local optima depending on the selection of the initial cluster means and so several trials are typically executed and the highest quality clusters are used. We executed 20 trials of *k*-means on the data and measured quality

by selecting the clusters that resulted in the lowest Euclidean Sum of Squares and highest Euclidean distance between the clusters as the solution groups [6].

Results (*k*=2). A statistically significant[1] ($p<.006$) difference was found in learning gains (pre to post test improvements) between students in the two clusters found by *k*-means with *k*=2. In terms of practical significance [13], the magnitude of this difference in learning gains is large[2] (Cohen's $d=1.48$).

In order to characterize the different learner groups found, we examined the differences between the groups on each of the 21 dimensions. Figures 2 and 3 show the *frequency* and *pause duration* dimensions that were found to have significant ($p<.05$) or marginally significant ($p<.07$) statistical differences, and significant practical differences (Cohen's $d\geq 0.8$) between the group with high average learning outcomes (HL) and the group with low average learning outcomes (LL).

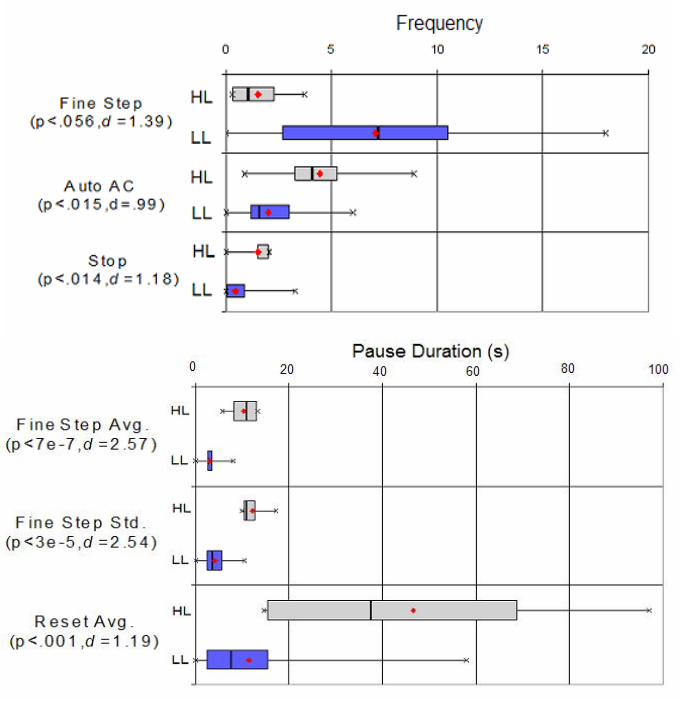

Fig. 2 and 3. Dimensions with significant differences between HL and LL (*k*=2). Fig. 2 (above) shows box plots of frequencies[3], and Fig. 3 (below) shows plots of pause durations.

The results on the use of the *Fine Step* feature are quite intuitive. It is reasonable that students who *Fine Stepped* frequently and consistently too quickly (given by the

[1] Unless otherwise stated, all tests for significance are one-tailed Student's t-tests.

[2] Cohen's standard suggests *d*=.2, *d*=.5 and *d*=.8 are small, medium, and large effects resp.

[3] *Fine Step* is plotted in actions per minute, whereas *Auto AC* and *Stop* are plotted in actions per 10 minute intervals because *Auto AC* runs AC-3 in its entirety and so fewer *Auto ACs* are typically performed, and *Stop* is used when running *Auto AC*.

combination of low *Fine Step* pause average and standard deviation) may be over using this feature mechanically, without pausing long enough to consider the effects of each *Fine Step*. Such behavior may negatively affect learning, as is evident with the LL group. The higher frequency of *Auto AC* by the HL group in isolation appears unintuitive, but in combination with the higher frequency of *Stopping*, this behavior suggests that students could be using these features to forward through the AC-3 algorithm in larger steps and to analyze it at a coarser scale. The HL group also paused longer after *Resetting* than the LL group. With the hindsight that these students were successful learners, we can interpret this behavior as an indication that they were reflecting on each problem more than the LL group. However, without prescience of learning outcomes, it is likely that an application expert or educator observing the students would overlook this less obvious behavior.

Results (k=3). For *k* set to 3, significant differences in learning gains were found between one group with high learning gains and the two other groups with lower learning gains ($p<.014$ and Cohen's $d>1.4$ in both cases). We will call these groups 'HL', 'LL1' and 'LL2'. No significant differences in learning gains were found between the two groups with low learning outcomes, suggesting that students may use/misuse pedagogical software in a variety of distinctive ways.

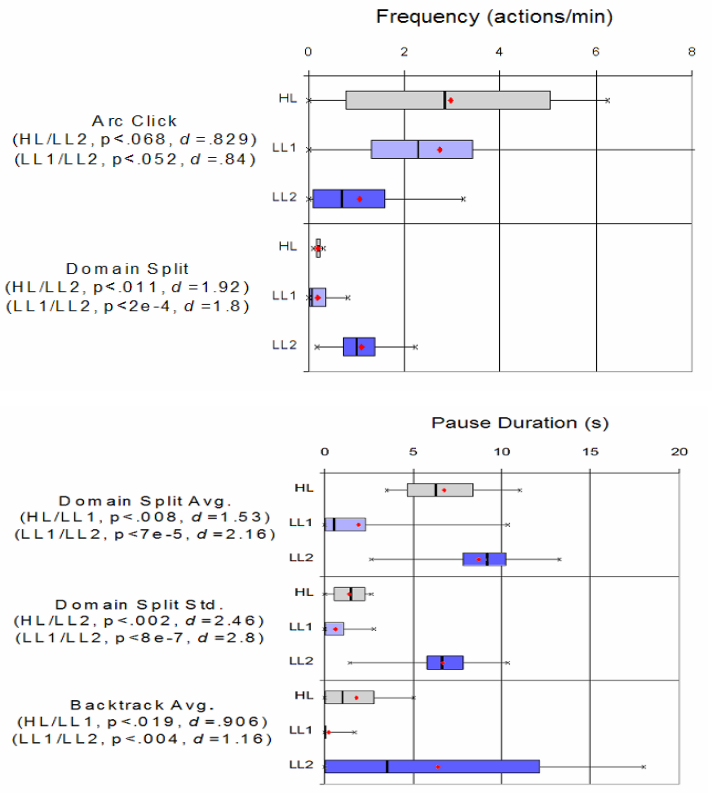

Fig. 4 and 5. Dimensions with significant differences between HL, LL1 and/or LL2 (*k*=3). Fig. 4 (above) shows box plots of frequencies, and Fig.5 (below) shows pause durations.

The same distinguishing behaviors identified by k=2 were replicated[4] between the HL group and both LL groups. For instance, both the LL1 and LL2 group had a significantly higher frequency of *Fine Stepping*, and a significantly and consistently shorter average pause duration after *Fine Stepping* than the HL group. This clustering also revealed several additional patterns, not only between the HL and LL groups, but also between the two LL groups, indicating that k=3 was better at discriminating between different behaviors. Figures 4 and 5 show the additional *frequency* and *pause duration* dimensions for which we found significant/marginally significant statistical and practical differences between the groups.

Interestingly, no differences were found on the frequency of use of the *Arc Click* feature between group HL and LL1, but differences were found along this dimension between HL and LL2 as well as LL1 and LL2. The HL and LL2 difference is reasonable because this feature involves more active engagement on the part of the learner and so using it more often could entail increased learning. However, the LL1 group used this feature comparably as frequently as the HL group but had significantly lower learning outcomes, suggesting that the LL1 students may be using it, but only passively. This is consistent with the passive operation of the *Fine Step* feature exhibited by the LL1 group, not shown in Figure 4 but analogous to the results presented in the previous section. Similarly, the HL group used the *Domain Splitting* feature as frequently as the LL1 group, but paused longer after each split on average. This feature is intended to require thought about efficiency in solving a CSP given different possible splits, and so longer pauses may be needed to thoroughly consider the choices. The LL2 group, however, paused longer and more selectively after *Domain Splitting* than the LL1 group, yet still had low learning gains. The LL2 group is also characterized by significantly longer pauses after *Backtracking* than LL1, and so in this case, the very long pauses may indicate that these students were confused about these applet features or the concepts of domain splitting and backtracking. Once again, these behaviors may be difficult to identify through mere observation.

Discussion. Though our sample size is small, and as a result the power to achieve statistical significance is reduced, k-means is still able to detect groups of users that achieved statistically and, arguably more importantly [13], practically different learning outcomes. And several of the behavioral differences found reasonably explained both effective and poor learning outcomes. However, as expected, some findings were less intuitive, requiring consideration of combinations of dimensions (as k-means does to determine its clusters). This makes interpreting meaningful characteristics a complex task, even for application experts, and highlights the benefits of using clustering to automatically identify learner groups.

5 On-Line Classification Through K-Means Classifier

Understanding the effectiveness of a student's behavior for learning is mostly useful if an ELE can provide adaptive support to improve behavior *while* the student is interacting with the system. Here we discuss the use of a version of k-means for on-line learner classification that can help provide this real time adaptive support.

[4] For space considerations these are not shown in Figs. 4 and 5. Refer to Figs. 2 and 3.

Once *k*-means clustering has found learner groups off-line, an on-line *k*-means classifier can incrementally update a student's classification within these groups as the student interacts with the applet. As interface actions are observed, the student's 21D feature vector is updated to reflect the new observation and classification is computed by simply recalculating the distances between the updated vector and the cluster means. The vector is then assigned to the cluster with the nearest mean.

We use leave one out cross validation (LOOCV) to see how the classifier generalizes to unseen data. We performed a 24 fold LOOCV for both the *k*-means classifiers with k=2 and k=3. For each trial the training data consists of the *k*-means clusters found off-line (Section 4) with one student removed, and the test data is the logged interface actions of the removed student.

Results (k=2 and k=3). Figure 6 shows the percentage of correct classifications as a function of the percentage of actions seen by the *k*-means classifiers over time. Note that we should not expect to achieve 100% accuracy after seeing all the actions because we are not re-clustering the data on-line, instead we are classifying incoming data given the clusters found off-line by *k*-means over the data points given by LOOCV. Thus, some error is expected reflecting the possibility of different clusters being found with one data point removed. The trend for k=2 suggests that this classifier's accuracy improves as more evidence is accumulated. More notably, the accuracy of the classifier is already around 80% after seeing only 10% of the actions.

It should be noted that there is an unbalance in classification accuracy between the individual clusters. With k=2, the accuracy is higher (93.5%) for the group with low learning gains (see Table 1, first row). This means that while this method currently would be very effective in detecting behaviors that eventually result in suboptimal learning, it would more often interfere with learners that show these behaviors sporadically but may eventually be successful. However, the lower accuracy (62.4%) for classification of the high learning gains groups' actions is likely the consequence of our small sample size. Only four students were clustered in this group, and so removing even one student for LOOCV purposes may produce different clusters than those found using all the data. Thus with small data sets, larger clusters may be more stable [6], suggesting that the accuracy of the classifier would increase with more training data. Further investigation is necessary to evaluate this hypothesis, but the fact remains that even with the current limited amount of data, *k*-means with k=2 reaches very high accuracy in detecting behaviors that eventually result in suboptimal learning.

Table 1. Classification accuracy for individual clusters. Clusters names appear with the number of members within that group.

	Learner Groups/number of available data points				
Classifier	HL/4	LL/20	LL1/8	LL2/12	Overall/24
k=2	62.4%	93.5%			88.3%
k=3	63.3%		62.1%	85.7%	74.1%

As Figure 6 shows, the trend for k=3 initially improves but then dips slightly as student actions are observed. Overall, this *k*-means classifier was able to detect the correct group labels 74.1% of the time (see Table 1, second row). For the high

average learning gains group, classification accuracy was 63.3%. For the two low average learning gains groups, the classification accuracies were 62.1% and 85.7% respectively. Group sizes decrease as the number of clusters increase and so the classifier with k=3 faces the same problem as the high learning gains group does for k=2 discussed above. The group with the highest classification accuracy was the group with 12 members, further supporting the hypothesis that lower accuracy for the other groups is an artifact of fewer data points.

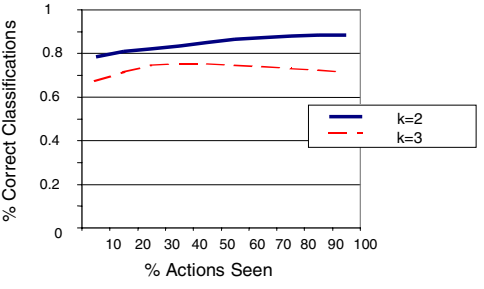

Fig. 6. *K*-means Classifier Performance Over Time

Discussion. From the results of the on-line *k*-means classifiers we can assert that this technique for classifying student actions appears to be a promising avenue worth further investigations. Despite the limited amount of available data, the classifiers were able to achieve reasonable accuracy, especially in detecting behaviors detrimental for learning, even after seeing only 10% of a student's overall actions. An adaptive ELE could use these classifications for interface adaptations to promote more effective behavior. For example, the ELE could employ a multi-layered interface design, where each layer's features are tailored to facilitate learning for a given learner group. Based on a learner's classification, the ELE could select the most appropriate interface layer. For instance, the ELE may select a layer with *Fine Step* disabled or with a subsequent delay to encourage careful thought for the students in either of our "low learning" groups, or could choose a layer with additional textual explanations of *Domain Splitting* and *Backtracking* for students classified in our LL2 group.

6 Conclusion and Future Work

In this work we have explored a data-based approach to automatic behavioral recognition in an ELE that uses interactive AVs to help students learn constraint satisfaction algorithms. We have described the use of *k*-means clustering to detect groups of learners with distinct interaction patterns and with significantly different learning outcomes. The clusters identified were then used for on-line behavioral classification. We found that this approach achieved good accuracy even after seeing only a small fraction of student actions, despite the low amount of data available for training.

The next step of this research is to collect more data and verify that this will substantially improve *k*-means performance. In addition, we plan on experimenting with

other unsupervised techniques including a probabilistic version of *k*-means called Expectation Maximization. We also intend to examine how well our approach transfers to other educational applications with different input dimensions including eye tracking and physiological data. Finally, we wish to design an adaptive support facility that takes as input on-line classification information, and empirically evaluate the classifier's effectiveness in a real world setting.

References

1. Amershi, S., Arksey, N., Carenini, G., Conati, C., Mackworth, A., Maclaren, H., Poole, D.: Designing CIspace: Pedagogy and Usability in a Learning Environment for AI. Innovation and Technology in CS Education (2005) 178-182
2. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Detecting Student Misuse of Intelligent Tutoring Systems. Intelligent Tutoring Systems (2004) 531-540
3. Beck, J.E. (ed.): Educational Data Mining: AAAI Workshop. WS-05-02 (2005)
4. Beck, J.E.: Engagement Tracing: Using Response Times to Model Student Disengagement. AI in Education (2005) 88-95
5. Bunt, A., Conati, C., Huggett, M., Muldner, K.: On Improving the Effectiveness of OLEs Through Tailored Support for Exploration. AI in Education (2001)
6. Duda, R., Hart, P., Stork, D.: Pattern Classification. 2nd edn. Wiley-Interscience, NY (2001)
7. Gorniak, P.J., Poole, D.: Building a Stochastic Dynamic Model of Application Use. Uncertainty in Artificial Intelligence (2000) 230-237
8. Hundhausen, C.D., Douglas, S.A., Stasko J.T.: A Meta-Study of Algorithm Visualization Effectiveness. J. Visual Languages and Computing 13, 3 (2002) 259-290
9. Hunt, E., Madhyastha, T.: Data Mining Patterns of Thought. In Beck, J.E. (ed.): AAAI Workshop on Educational Data Mining (2005)
10. Poole, D., Mackworth, A, Goebel, R.: Computational Intelligence: A Logical Approach. Oxford University Press, New York (1998)
11. Stern, L., Markham, S., Hanewald, R.: You Can Lead a Horse to Water: How Students Really Use Pedagogical Software. Innovation and Technology in CS Ed. (2005) 246-250
12. Talavera, L., Gaudioso, E.: Mining Student Data to Characterize Similar Behavior Groups In Unstructured Collaboration Spaces. Workshop on AI in CSCL, European Conference on Artificial Intelligence (2004) 17-23
13. Vicente, K.J., Torenvliet, G.L.: The Earth is Spherical ($p<0.05$): Alternative Methods of Statistical Inference. Theoretical Issues in Ergonomics Science 1, 3 (2000) 248-271