Jl. of Interactive Learning Research (2007) **18***(2)*, 185-205

Results from Action Analysis in an Interactive Learning Environment

OLIVER SCHEUER
German Research Center for Artificial Intelligence DFKI, Germany
oliver.scheuer@dfki.de

MARTIN MÜHLENBROCK
European Patent Office, The Netherlands
martin@muehlenbrock.de

ERICA MELIS
German Research Center for Artificial Intelligence DFKI, Germany
erica.melis@dfki.de

Recently, there is a growing interest in the automatic analysis of learner activity in web-based learning environments. The approach and system SIAM (System for Interaction Analysis by Machine learning) presented in this article aims at helping to establish a basis for the automatic analysis of interaction data by developing a data logging and analysis system based on a standard database server and standard machine learning techniques. The contribution is the integration of components which are appropriate for large amount of data. The analysis system has been connected to the web-based interactive learning environment for mathematics, **ActiveMath**, but is designed to allow for interfacing to other web-based learning environments, too. The results of several usages of this action analysis tool are presented and discussed. They indicate potentials for further development and usages.

Introduction

Recently, there is a growing interest in the automatic analysis of learner interaction data with web-based learning environments. This is largely due to the increasing availability of log data from learning environments and in particular from web-based ones. The potential outputs include the detection of regularities and deviations in the learners' or teachers' actions as well as

building of models which can predict learner features from log data. The objective is to use those outputs to support teachers and learners by providing them with information that helps to manage their learning and teaching or to use the information for adaptation actions of eLearning systems.

Commercial systems such as WebCT, Blackboard, and LearningSpace already give access to some information related to the activity of the learners including some statistical analyses, and provide teachers with information on course attendance and exam results. With this information already being useful, it only represents the tip of the iceberg of what might be possible by using advanced technologies.

This upcoming research area, (i.e., addressing the automatic analysis of learner interaction data), is related to several well-established areas of research including intelligent tutoring systems, web mining, statistics, and machine learning, and can build upon results form these fields for achieving its objectives. In contrast to intelligent tutoring systems, learner interaction analysis does not rely on models of the learner or of the domain knowledge since these are heavy to build and maintain.

In this regards, learner interaction analysis is comparable to website data mining but with a specific perspective on learning settings and with the availability of pedagogical data that usually are not available in web mining applications that are mostly based on click-through data only. Click-through data streams only allow for a rather shallow analysis, but with the inclusion of pedagogical data, more advanced techniques can be adopted from the field of machine learning, for example, in order to learn models of individual behavioural and non-observable variables that can predict the non-observable variables from behaviour of future students .

Although a number of open questions have already been tackled (Arroyo, Murray, & Woolf., 2004; Heiner, Beck, & Mostow, 2004; Merceron & Yacef , 2003; Merceron, Oliveira, Scholl, & Ullrich 2004; Mostow, 2004; Oliveira & Domingues, 2003; Zhang & Lu, 2001), there is not yet a systematic approach in analysis interaction data from huge learner action logs nor are there common architectures. The approach presented in this article aims at helping to establish a basis for the automatic analysis of interaction data by developing a data logging and analysis system based on a standard database server and standard machine learning techniques. This work was conducted in context of the iClass project which aimed at developing an intelligent, cognitive-based e-learning system, to enhance the iClass system with profiling capabilities.

Because the iClass system was still under development when this research was carried out, our analysis system has been connected to the web-based interactive learning environment for mathematics **ActiveMath** which provides the log files. However, the system is designed for interfacing to other web-based learning environments, too, which will enable us to connect it with the upcoming iClass system. It has been tested with a medium scale experiment

in which four classes of a secondary school participated throughout a school term of five months on a weekly basis as well as on a large scale experiments in context of an introductory mathematics course at a UK-based University.

This article is organized as follows. In the following section, the SIAM system will be described, which is comprised of a learning environment, a logging component, and an analysis component. Subsequently, two studies will be presented: In the first one, the system's capabilities to estimate students' performance and gender, and in the second one the relationship between students' behaviour and cognitive style was investigated.

The Action Analysis System SIAM

The SIAM system is comprised of three major parts, that is, a learning environment, an action logging component, and an action analysis component (see Figure 1). These system parts will be described in more detail in this section. In addition to these components, there are also three data repositories involved in storing and providing information: a learning material database, a set of user log files, and a database containing logs and possibly additional user data and context data.

The analysis subsystem has been implemented by using standard technology such as Java and mySQL, which are available for a number of platforms and operating systems, together with the suitable drivers for database

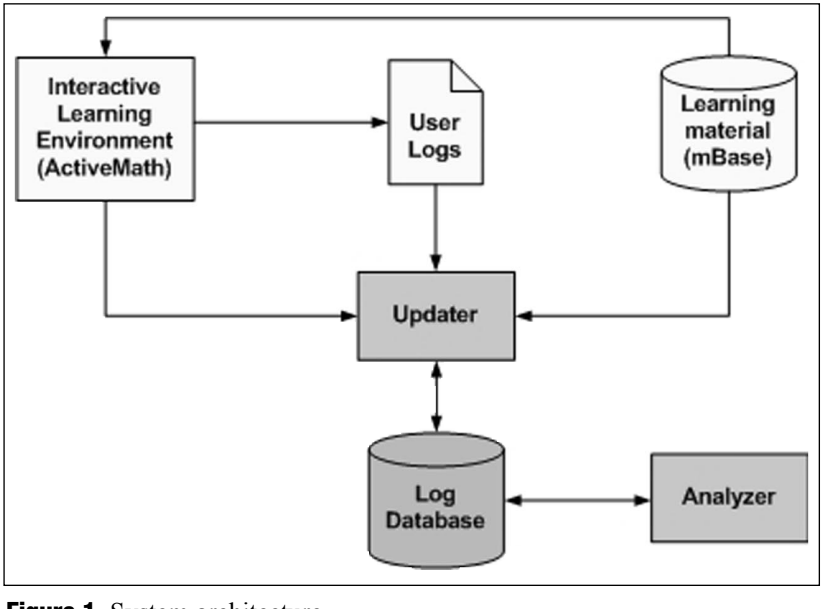

Figure 1. System architecture

connectivity. In addition, the Analyzer is based on the Weka (Witten & Frank, 1999) and YALE (Fischer, Klinkenberg, Mierswa, & Ritthoff, 2002) toolkits, which provide tools for visualizing and exploring data as well as means for integrating machine learning functionality into applications.

As a starting point, the web-based learning environment **ActiveMath** has been used to provide a testbed for developing and testing the action logging and analysis components. However, these components have been designed to be mostly independent of a specific learning environment, allowing for providing the same logging and analysis functionality to other learning environments.

ActiveMath Learning Environment

ActiveMath is a web-based learning environment that dynamically generates interactive courses adapted to the student's goals, preferences, capabilities, and prior knowledge (Melis et al., 2001; Melis et al., 2006). The content is represented in a XML-knowledge language for an educational context which greatly supports reusability and interoperability of the encoded content. **ActiveMath** supports individualized learning material in a user-adaptive environment, active and exploratory learning by using (mathematics) service tools and with feedback (see Figure 2).

For different purposes and for different users, the learning material and its presentation can be adapted: the selection of the content, its organization, and the means for supporting the user have to be different for a novice and an expert user, for an engineer and a mathematician, for different learning situations such as a quick review and a full study. Since there is no way of knowing in advance the goals, the profile, and the preferences of any user when designing the system, **ActiveMath** builds on adaptive course generation.

One component of **ActiveMath** – its event framework – is especially relevant to the SIAM system since it provides the information to be logged. The event framework (Melis et al., 2006) realizes a publish-subscribe scenario and governs the asynchronous communication between system components and even communication of remote services. Events are a mechanism for a powerful and flexible, yet rather loose integration of components.

An example for event publication is the following: when the learner finished working on an exercise, the exercise subsystem issues an event. The event carries information describing the learner, the identifier of the exercise, the success rate, and the time stamp of the event. Listeners that can subscribe to such an event can be the learner model as well as the suggestor of the tutorial component. A component that publishes events is called an *event source*. A component that subscribes to the events published by an event source is called a *listener* which receives event messages from the event source.

In contrast to a full-fledged messaging model, events remain anonymous rather than being sent from a specific sender to a specific recipient: when publishing an event, the event source is usually not aware who is listening

to the events (only the module managing the subscriptions is). Moreover, usually the listener does not care which component or module created the event, it only knows where to subscribe to the events it is interested in.

The Action Logging Component receives the trace of user actions by subscribing to the event framework for relevant events. To return to the example above, when for example, the learner finished working on an exercise, the generated event will be received by the Action Logging Component.

Figure 2 shows the **ActiveMath** user interface. The left panel shows the table of contents (TOC) of the currently selected book (in **ActiveMath**, the term book is used as a metaphor for a course). The contents are organized in a hierarchy with expandable and collapsible nodes. The lowest level is constituted by single book pages which can be selected. A colored bullet left-hand side of each book page item indicates the system's belief in the student's mastery for the respective contents.

Right of the TOC the currently selected book page is presented. Each page consists of a sequence of learning items, which may be reading material, exercises, or interactive examples. Exercises are displayed in a separate window, when the *Start exercise* link is clicked. Relevant concepts occurring in the texts are hyperlinked; their selection opens a window containing a concept definition or other additional information. In the upper-right of each

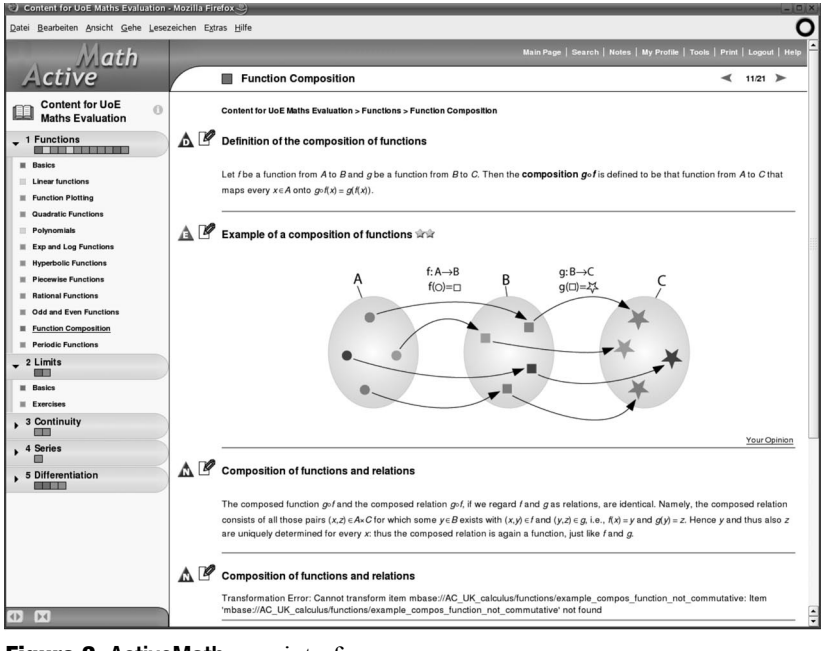

Figure 2. **ActiveMath** user interface

displayed learning item is a *Note* icon, where students can access the notes functionality. Students can use it to annotate learning items with private or public comments, or to read already existing notes.

In the lower part of the page a previous- and a next-button are located. As alternative to the TOC, from which pages can be accessed based on a hierarchical overview, these buttons allow movement through the contents in a predefined linear sequence.

Finally, in the upper-part of the screen, the menu bar is located which offers functionalities that are independent of specific contents and which are generally available. This includes a *Menu* link to return to the main page were books can be selected, a *Dictionary* link which opens a window where search queries can be submitted and a *Logout* button to finish the current session.

Action Logging Component

For each learner, the **ActiveMath** environment generates an online log that lists user actions in the learning environment in terms of general information such as time, type of action, user name, and session number, as well as specific information including which page has been presented to the user, which item has been seen by the user, which exercise has been tackled and solved or not solved.

The Updater (see Figure 1) receives event information on the users' actions from the learning environment, and transforms every user event into one or more corresponding database tables. Usually, the Updater receives the information online from the event queue, but it can also read in files with log data that have been generated in an offline mode.

The Log Database (see Figure 1) is at the center of the SIAM system. It contains not only representations of the raw data in the user logs (see Table 1), but also has tables that hold the results of the analysis. Moreover, it contains tables for additional background knowledge concerning the users, context, or courses among others (see Table 2).

The basic level of the database, which corresponds to the raw log data, is organized in tables that represent generic event information as well as event-specific data. The structure of these tables has been designed closely to the events specification, since this allows for simpler updating operations when the event subsystem is changed or replaced by another system. Table 1 lists the basic event tables together with their fields and a short description. In addition, as shown in Table 2, the database includes tables that hold additional information on the users and sessions such as gender and holiday periods, respectively, as well as tables that are derived from these by means of database queries.

In addition, the Updater provides some data completion functionality. Every now and then for some tables the information is not complete. For instance, most users do not log out of the learning system explicitly by using the button provided in the user interface, but simply close the browser or

Table 1
Log Database Schema for Basic Level

Table	Attributes	Description
Event	*eventId* *timestamp* *source* *session* *userId* *type*	Generic information for all events
eventUserCreated	*eventId* *userName*	Registration of a new user to the system
eventUserLogged-In	*eventId* *ip* *userAgent*	User logs into the system; start of a new session
eventUserLogged-Out	*eventide*	User logs out of the system; end of a session
eventPage-Presented	*eventId* *book* *page*	System presents a requested book page to the user
eventExercise-Started	*eventId* *exercise*	User starts an exercise
eventExerciseStep	*eventId* *userInput*	User submits an input for an exercise
eventExercise-Finished	*eventId* *exercise* *successRate*	Exercise is finished
eventMastery-Changed	*eventId* i*tem* *masteryDimension* *oldValue* *newValue*	System changes the mastery level of an user
eventItemPresented	*eventId* *item* *itemType*	Presentation of a learning item to the user
eventUserProperty-Changed	*eventId* *propertyNam* *oldValue* *newValue*	Indicates changes of some user meta data
eventItemSeen	*eventId* *duration* *item*	Gives a more fine-grained resolution (on item level) of the user's focus (uses information of an eye-tracker)

shut down their computer. In this case usually no event is generated concerning the logout. The corresponding logout table is enhanced by information that is derived from the other events the user created and on heuristics concerning pauses and open hours among others. This information is automatically added to the login table, but is marked as derived information to be distinguishable from the original log data.

Since log files can grow very large, a tool for a realistic usage (rather than for small-scale academic purposes only) needs to built on a database. For us this was a major design decision but only few research prototypes developed so far take this need into consideration.

Action Analysis Component

The Analyzer (see Figure 1) performs data aggregation and evaluation in terms of the queries to the log database. It also incorporates a number of machine learning methods for automatically analyzing the data and takes the data from the Log Database as an input. If needed, adjustments and preferences can be input by the engineer who is running the analysis.

In addition of getting a better insight into the underlying relationships in the data, the results of the analysis can be used for the prediction and classification of future sessions. Up to now, the Analyzer is not an integrated, fully automated component but consists of a set of SQL-query scripts that preprocess the raw action data, and the machine learning tool box YALE (Fischer et al., 2002) to execute the actual machine learning analysis.

YALE offers a wide range of operators for performing data pre-processing (discretisation, filtering, normalisation and others), learning (more than 100 learning algorithms), validation (e.g., cross-validation, several performance evaluation criteria) and other analysis-relevant tasks. In the following, a selection of learning schemes used in the Action Analysis Component is presented:

Decision tree learner: Many machine learning methods provide their output in an intelligible, human readable form. For instance, methods for generating decision tress from data, such as C4.5 (Quinlan, 1993), allow for a tree-shaped representation of the learning results. A decision tree is constructed by the algorithm first selecting an attribute to place at the root node of the tree and make one branch for each possible value. This splits up the example set into subsets, one for every value of the attribute. The attribute is selected in a way that maximizes the information gain by the chosen attribute. This process is repeated recursively for each branch, using only those instances that actually reach the branch. If at any time all instances at a node have the same classification, the developing of that part of the tree is stopped.

Rule Learner: Rule learning algorithms generate a set of prediction rules each consisting of a class values and its condition. The PRISM rule learner, for instance, constructs rules in the following way: After choosing a class value the algorithm starts with an empty rule and adds iteratively new conditions of the form 'attribute X has value Y.' Each added condition narrows the scope of the rule because less trainings instances will match the extended condition. On the other hand, the attribute-value pairs are chosen in a way which increases the accuracy of the rule (a higher share of the matched instances will be correctly classified). If the rule is perfect (100% accuracy) the algorithms goes over to construct the next rule. To cover all trainings

instances, several rules for one class value might be necessary. This procedure is repeated for all class values.

Naïve Bayes: Naïve Bayes is a widely-used probabilistic classifier, based on Bayes' Theorem. It assumes that all attributes are independent (therefore naive). Dependencies between attributes might comprise the quality of computed models.

Support Vector Machine (SVM): Support Vector Machine algorithms try to find hyperplanes in the attribute space which optimally separate instances of different classes. Optimality is achieved by maximising the margins between classes (the margin is the minimum distance between class instances and the separating hyperplane). Because more complex patterns can not be captured with a linear separation, SVM apply the *kernel trick*: instead of changing the SVM core algorithm, the attribute space is transformed by defining a non-linear kernel.

Boosting algorithms: Boosting algorithms (e.g., AdaBoost) are meta-algorithms which build on so-called weak learners that are rather simple learning methods, for instance a Decision Stump algorithm (computes a 1-level decision tree). The boosting algorithm iteratively applies the weak learner to the training instances to compute a set of predictive models. Initially, all instances are equally weighted. In each iteration, weights of the falsely classified instances gets increased and of correctly classified ones decreased. In the subsequent iteration the learner will try to find a model which takes the falsely classified ones more into account. The complete model of the whole boosting procedure combines all the computed simple models, weighted by their accuracy.

Problems and Solutions

Although the SIAM system presented here considerably facilitates the process of action log analysis there are still some open questions and problem to be tackled which will have to be addressed in future.

Up to now, pre-processing is executed in a semi-automated way: before starting the analysis the existing SQL scripts are adjusted according to the purpose of the current investigation and then applied by the Analyzer component to the raw data. This indicates further potential for automation: it has to be found out which usage aspects are generally relevant to the analysis process, independent of specific purposes of individual investigations. For example, students' success rates, the number of pages read or their overall online time will in most investigations play a role. The computation of such aspects could be completely automated. The computation of aspects which are specific to individual investigations will still need manual adjustments.

A further candidate for automation is the inclusion of external, non-behavioural data which are not contained in the action logs (e.g., gender, questionnaire data, pre- and post test results). Currently, this essential infor-

mation is manually added to the database, from spreadsheets provided by teachers. It has to be investigated which interfaces can be offered to teachers to automatically feed the database.

There is ongoing work concerning the inclusion of pedagogical and domain data. State-of-the art e-learning systems use learning objects which are annotated with pedagogical or domain-specific metadata. Including this information will allow more sophisticated analyses. For example, performances can be analyses for different topics or different levels of difficulty.

Related Work

Merceron and Yacef (2005) present a case study how educational data can be mined to support teachers to understand their students' learning and students to promote self-reflection. The analysis was based on data of the Logic-ITA system, a web-based tutoring tool to practise logical proofs. The conducted analyses included association rule algorithm that can be used to identify sets of mistakes that often occur together, or a decision tree algorithm to predict final exam marks given students prior usage behaviour. They used their tool for advanced data analysis in education (TADA-Ed) (Benchaffai, Debord, Merceron, & Yacef, 2004) to carry out the analyses.

There are a number of tools which are more concerned with an appropriate presentation of student action data than with an automated analysis. For instance, The LISTEN Reading Tutor is intended to help children learning to read. The system displays stories, sentence by sentence, which children then have to read aloud. Children's utterances as well as other student-tutor interaction are logged into a relational database. Mostow et al. (2004) present a tool to browse the student-tutor interaction data of the LISTEN Reading Tutor which are presented in a hierarchical tree structure.

The Data Recording and Usage Interaction Analysis in Asynchronous Discussions System (D.I.A.S.) presented by Bratitsis and Dimitracopoulou (2005) is intended to improve asynchronous discussions. The usage data is stored in a database system. From the raw data, meaningful statistics (so-called interaction analysis indicators) which give account of passive and active participation, and thread initiations, are extracted by means of SQL queries. These indicators can be presented to discussion participants, but also to monitoring persons as teachers by a visualisation module. Discussion participants may benefit from an increased awareness of their own actions (metacognition) as well as those of other participants. Teachers can identify problems and, if necessary, intervene.

Using Action Analysis

The Action Analysis System has been investigated in a mathematics courses in a secondary school and at a UK-based University. The objectives of those tests have been (1) to collect experience with the SIAM system and to find possible ways to improve it, (2) to conduct the actual analyses to get some insight

how students use the **ActiveMath** and to uncover relationships between individual (e.g., gender and cognitive style), behavioural and external factors (teacher).

Estimating Performance and Gender

The SIAM system has been tested in a medium sized experiment in a mathematics course in a secondary school. About 70 students from three different courses used the learning environment for a period of five months. A further course of about 20 students were taught the same subject but in the traditional classroom manner. The other three courses used the **ActiveMath** learning environment on a weekly basis in two-hour lessons. During the online course, each class was split into two subgroups using different computer rooms. Many students were already familiar with computers, but a considerable number needed further instruction even for basic operations such as login.

A preliminary evaluation of the logged data after a first couple of sessions showed some problems in the quality of the data. For instance, instead of registering with the **ActiveMath** system only in the very first session and using the created user account in the sequel, a large number of students created a new account including a new user name for each session, which makes difficult the longitudinal analysis of the data. The problem was resolved by having the students create only one account and making the registration procedure inaccessible for them after that.

Figure 3 provides an overview on the data that shows the number of events related to the hours of the day. Clearly, the major amount of events was created during lesson hours between 9 am and 2 pm (14 h), but some events were

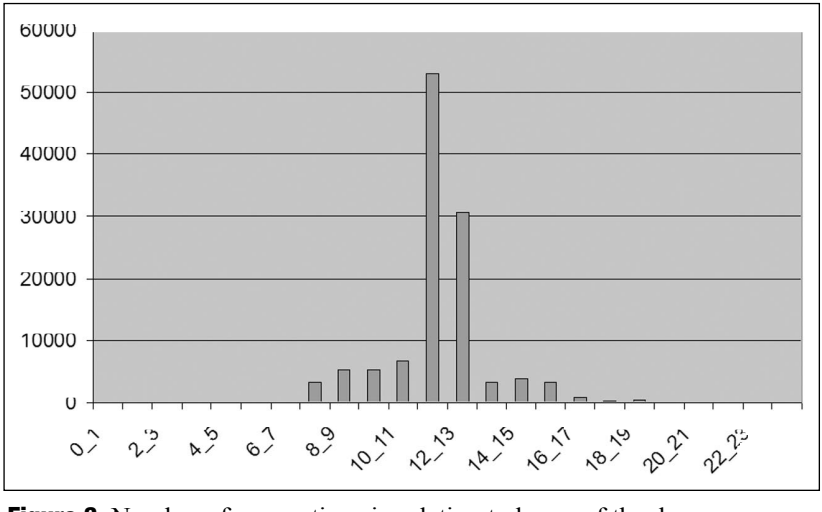

Figure 3. Number of user actions in relation to hours of the day

generated earlier or later in the day. Some of them are due to a small number of students using the system in off-time, and most of them are due to teachers and system administrators preparing or evaluating the system.

At the end of the term, a written post test has been done with the students to assess what they had learnt. The results were added to the database manually as well as some further information on gender, teacher, and so forth. Further information was semi-automatically generated by the Analyzer from the log data and added to the database. For each student the information in Table 2 has been gathered for further analysis. For anonymity reasons the students used arbitrary user names in the learning environments, and they were to give these user names also in the post test. However, in one course the students put down their real names on the test sheets, a fact which makes the linking to their log data impossible. Finally, 25 student records were complete and clean enough for being used in the further analysis.

Figures 4 and 5 show the decision trees that were generated for characterizing the attributes post-test (with values low, medium, and high) and gender (with values male and female), respectively. The decision tree for post test (see figure 4) shows that for predicting the result the teacher is most influential, that is, with teachers Mr. E. and Mr. G. the post test result is expected to be low. However, with Mr. K. the test result is high if the student is in class 6a, would be low if he was in class 6b and male, and medium if she was in class 6b and female. Similarly, the decision tree for gender (see Figure 5) depicts that when the post test result is bad or medium the user would be male

Table 2
Data From Manual Input and Gathered Automatically

Attribute	Input	Comment
Gender	manual	
Class	manual	Course, each comprised of about 20 students
Teacher	manual	Each class has been split into two subgroups with each being taught by another teacher
post test	manually	Results in the post test done in writing
Exercises started	semi-automatic	
Exercises finished	semi-automatic	
successful exercises	semi-automatic	
reading actions	semi-automatic	
solving actions	semi-automatic	
Dictionary	semi-automatic	Whether the student used the dictionary for lookup
Integration	manual	Whether the student is handicapped
off time	semi-automatic	Whether the student accessed the learning environment beyond lesson hours

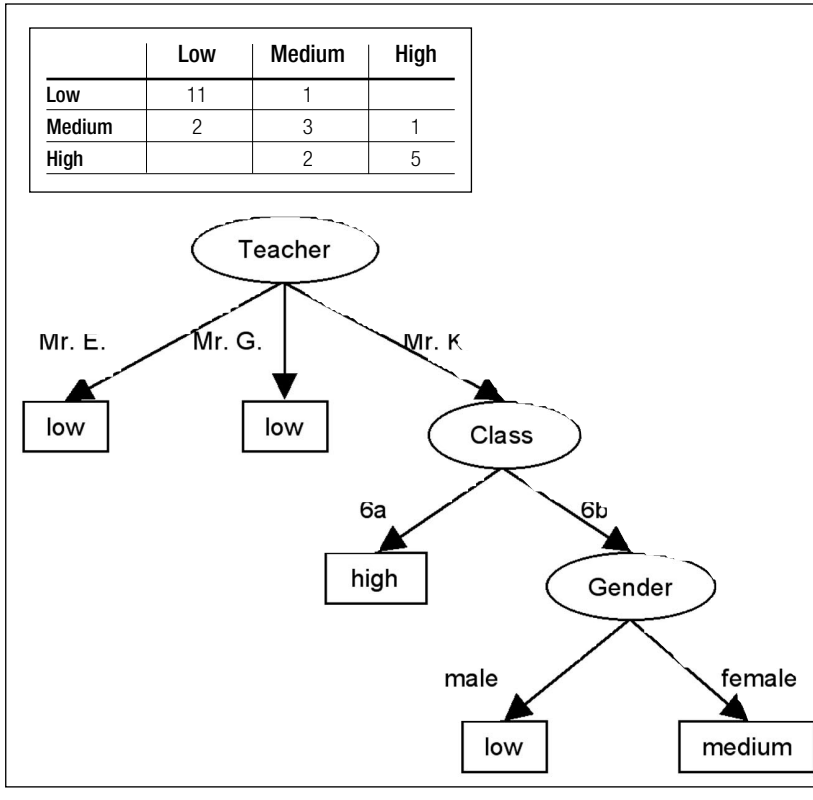

Figure 4. Decision tree for post test and corresponding confusion matrix

or female, respectively, and if it was high the reading activity would indicate a male use if it was low and a female user if it was medium or high. This is an interesting results that can be interpreted as indicating that female users more successful in tests and a harder working concerning reading material. Importantly, these attributes have solely been selected by the learning algorithm from the ones in Table 2, which means that other attributes such as successful exercise solving and off time system usage were of minor relevance.

Figures 4 and 5 also give the quality of the decision trees in terms of confusion matrices. A confusion matrix displays the result of testing the decision tree with data as a two-dimensional matrix with a row and a column for each class. Each matrix element shows the number of test examples for which the actual class is the row and the predicted class is the column. Good results correspond to large numbers down the main diagonal elements and small, ideally zero, off-diagonal elements. Hence the confusion matrices in Figures 4 and 5 indicate not ideal, but quite good decision trees.

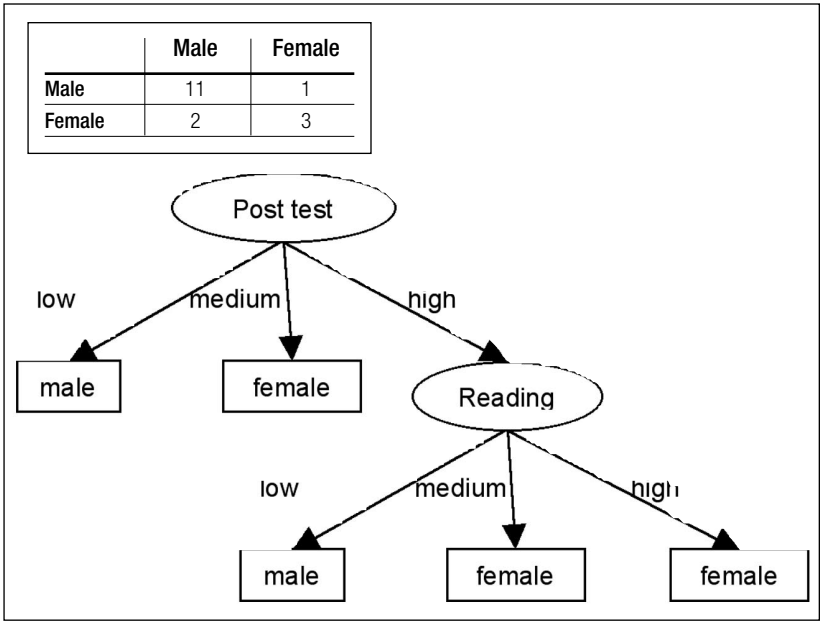

Figure 5. Decision tree for gender and corresponding confusion matrix

CONCLUSION

In this study, a decision trees were computed capable of predicting gender and performance. These analyses indicate possible usages of the SIAM system albeit, due to the small sample size, the expressiveness and generality of the presented results are limited. One reason for this fact is the high number of instances which had to be excluded from the analysis. This underlines the importance of a careful preparation.

Estimating Cognitive Styles

A further study was conducted in the context of an introductory mathematics course for computer science students at a UK-based University with around 300 participating students. Its main objective was to investigate the relationship between students' cognitive styles and their observable behaviour in the **ActiveMath** system by means of machine learning.

Introduction: Cognitive styles

Cognitive styles describe an "individual's preferred and habitual approach to organising and representing information" (Riding & Rayner, 1998). Among the considerable number of existing style theories, especial-

ly Witten's idea of field-dependency (Witten & Frank, 1999) attracted attention in the scientific community and was subject to numerous studies. Field-dependent individuals are attributed with the tendency to rely on an external frame of reference (the field) when tackling a task whereas field-independent ones make more use of an internal mental model. As an implication for instruction, field-dependent students may benefit to a larger extent from external help and guidance whereas field-independent ones can take advantage from instructional settings which allow a higher degree of freedom.

The advent of hypertext gave rise to a new strand of research which investigated how subjects with different cognitive styles can cope with differently structured environments. Hypertext environments provide a nearly ideal testbed for examining these relationships: field-dependent and field-independent subjects may show differences in navigation behaviour and tool use, and their performance may differ depending on the characteristics of the underlying hypertext system (linear vs. non-linear structure, offered tools, offered navigational support). Chen and Macredie (2002) give a comprehensive overview of empirical studies concerning the relationship between field-dependency and hypermedia navigation and their findings.

What can be the benefit of knowing students' cognitive style? There are a number of studies which address this question by comparing scores of subjects performing a task in an environment which corresponds to their cognitive style (matching condition) with scores of subjects in a non-matching environment (Ford & Chen, 2001; Witten, 1999). The results essentially support the hypothesis that matching students' style does have a positive effect on performances. Therefore, cognitive styles gained some attraction in the adaptive hypermedia community: tailoring content structuring, presentation style and navigation support to individuals' cognitive style may be a promising approach to improve usability, and in the area of hypertext learning, to increase students' learning gains.

Traditionally, cognitive styles are determined by means of questionnaires and psychological tests. A machine learning model could directly derive styles from behavioural data and would make the application of additional instruments dispensable.

Pre-study: Hypotheses Building

The quality of a machine learning model is strongly dependent on the set of attributes it is generated from. For the purposes of this study, attributes should reflect behavioural aspects which are relevant for the decision between field-independence and field-dependence, that is, attributes in which field-independent subjects significantly differ from field-dependent ones.

Therefore, a pre-study was conducted to review empirical findings reported in the relevant literature. We found evidence that the following aspects are linked with field-dependence: Linearity of navigation, revisita-

tion behaviour, material coverage, pace, home page visiting behaviour, use of hyperlinks, exercise behaviour and tool usage. The analysis will show if these results can be replicated or not.

Experimental Setting

Around 300 students of a UK-based University participated in an introductory course in mathematics for computer scientists. For one part of the course, the instruction was based on the **ActiveMath** system. The computer-mediated contents covered the topics functions, matrices and graphs. **ActiveMath** was partly used in supervised tutorial sessions, partly on students' own discretion. The supervised sessions covered four tutorial weeks: week 1 was mainly devoted to get to know the system, week 2 and 3 to learn the material, and week 4 to revise for exam preparation.

Additionally, students were asked to take part in the Cognitive Style Analysis (CSA), a computerised test to determine their cognitive styles. Low test scores correspond to a field-dependent (FD) cognitive style, high scores to a field-independent (FI) cognitive style.

ANALYSIS AND RESULTS

Based on the results of the pre-study, a set of attributes, describing students' behaviour in the **ActiveMath** system, was computed for each supervised session (see Table 3, in parentheses the hypotheses they refer to). These data, combined with the CSA results of the respective students, constituted the input for the machine learning algorithms.

In a first processing step, inappropriate data was filtered out (students not taking part in the CSA test, outliers in terms of extremely short online time (< 20 minutes), small number of pages (< 3), small number of exercises finished (<3)).

Several machine learning schemes were applied in order to compute a predictive model (Naïve Bayes, C4.5 Decision Tree, Support Vector Machine (SVM), PRISM Rule Learner, AdaBoost with Decision Stump). Because no satisfactory result could be achieved (all models were outperformed by a simple choice of the most frequent class with 43 % accuracy), a series of variations in the machine learning setup were conducted (e.g., analysing every tutorial week alone, several other preprocessing steps, variation of parameters of machine learning algorithms). The varied setups did not lead to better performances.

Pearson's correlation coefficient r was computed to check whether in the pre-study hypothesised relationships between field-dependence and behaviour can be confirmed. Also the application of Pearson's correlation coefficient identified only three significant relationships ($p<0.05$), namely positive correlation between field-independence and the notes creation, notes access and

Table 3
Attributes Extracted From Students' Action Log Traces

Attribute	Description
tocRate	ActiveMath offers two means for navigating through contents: Students can either select pages directly in the TOC panel, or use the previous and next buttons. This attribute gives the ratio between TOC and previous-next moves. (**_Linearity**)
stratum	A graph-theoretical measure indicating the linearity of a student's navigation graph. A detailed description of its computation and the rationale behind is provided by McEneaney (2001). (**_Linearity**)
compactness	The non-linear counterpart to the stratum measure. It quantifies the degree of connectedness of the student's navigation graph (see also McEneaney, 2001). (**_Linearity**)
recurrenceRate	The average number of times the student returns to a page already seen. (**_Revisitations**)
homepageRate	Number of home page visits divided by the number of book page requests. (**_Home page**)
notesCreationRate	ActiveMath offers a 'Notes' function for annotating learning items. These notes can also be made public and are the main communication vehicle in ActiveMath. The attribute is given relative to the number of learning items seen. (**_Tool usage**)
notesAccessRate	Amount of accessed notes relative to the number of learning items seen. (**_Tool usage**)
numSearch	Number of search queries. (**_Tool usage**)
linkClickRate	Key terms occurring in the content material are hyperlinked. Clicking these hyperlinks opens a window containing additional information, for instance a concept definition. The attribute gives the number of clicked key terms relative to the number of pages visited. (**_Hyperlink**s)
avgNumExAttempts	Average number of attempts needed to solve an exercise (aborted exercises are excluded). This attribute gives at the same time account of the exercise performance. (**_Exercises**)
exFinishingRate	Exercises can be aborted by closing the exercise window. This attribute gives the relative amount of non-aborted exercises. (**_Exercises**)
exTacklingRate	Number of exercises tackled relative to the number of exercises seen. (**_Material coverage**)
numPages	Reading material and exercises were located on different pages. This attribute gives the absolute number of reading material pages visited. (**_Material coverage**)
timePerEx Attempt	Time spent on exercise pages divided by the number of exercise steps. (**_Pace**)
timeFirstEx Attempt	Typically, the first attempt needs more time than the subsequent attempts. This attribute gives the median time between exercise start and the first submitted answer. (**_Pace**)
timePerPage	Median time per book page (only reading material). (**_Pace**)

exercise tackling rate. Also, these results cannot be regarded as convincing because they are not constant over time (only observable in one of four weeks).

DISCUSSION

The presented study investigated the relationship between students' behaviour in a hypertext-based learning system and their field-dependence cognitive style. The behavioural aspects under consideration were derived from findings reported in the relevant literature. Several machine learning schemes were applied and Pearson's correlation coefficient was computed but none of them were capable to replicate findings of prior studies. The possible reasons are discussed in the sequel.

In this study Riding's CSA test was used to measure students' field-dependence. Most of the findings presented in the pre-study are based on the instruments Embedded Figures Test (EFT) or Group Embedded Figures Test (GEFT). It is not clear in how far findings based on the classical tests EFT and GEFT can be applied to subjects classified according to CSA scores. Zang and Noyes (2006) report on two studies: in the first one, only moderate correlations between CSA and GEFT results could be observed, in the second one neither EFT nor GEFT scores were correlated with CSA scores. It is not very surprising that no strong correlations are discovered because one motivation for developing the CSA instrument was to overcome methodological flaws of EFT and GEFT. On the other hand, also the results from CSA-based studies could not be replicated.

It might be possible that other individual differences not considered in the analysis also play an important role. There is some evidence that prior knowledge (Last, O'Donnell, & Kelly, 2001), gender (Ford & Chen, 2000), cultural background (Kralisch & Berendt, 2004) and computer experience (Ford & Chen, 2000) also affect hypertext navigation, tool use and performance. The omission of some of these factors might prevent a reliable diagnosis. The influence of computer experience in combination with the field-dependence style on search and navigational behaviour was investigated in a study by Kim (2000). Significant differences between FD and FI subjects could only be observed for inexperienced subjects. Due to the fact that all participants of the here presented study took courses in computer science we can assume rather experienced subjects where only marginal behavioural differences exist.

A further reason might be a lack of opportunities to display different behaviour. Exercises consisted only of one step and were largely solved without problems (74% of all exercises were solved in the first of three possible attempts). More complex exercises, possibly with a help or a hint button, would have allowed a deeper analysis of problem solving behaviour. The contents were adapted from a paper script with the consequence of a predominantly linear structure with only few hypertext features. This structural restrictedness of the hypertext environment might have led to similar navigation strategies of FDs and FIs. In a study presented by Ford & Chen (2000) FD subjects made greater use a topic maps for navigating than FI

ones, presumably to capitalise on structural information provided there. This suggests that FD subjects will also use **ActiveMath's** table of contents (TOC) to navigate. On the other hand, the TOC is the only navigation control allowing non-linear navigation. Hence both, FD and FI subjects, might appreciate the TOC, albeit, due to different reasons. A look into the data supports this assumption: Pearson's correlation coefficient between CSA score and TOC rate (r =.00), and between CSA score and stratum (r=.03) indicate the absence of such straight connections in the analysed sample.

A further possible problem is linked with the data logging process. Web-browsers offer back and forward buttons which allow movement between already visited pages. These pages are usually not requested from the web-server but retrieved from the local browser cache. Because the server side is not involved in this process, these navigational moves are not available to the logging component and are missing in the action traces. This might lead to complications because back-button usage is one of the most prominent navigation actions. In a study by Tauscher & Greenberg (1997) the back button accounted for 30% of all navigation actions. Not only that a possible back button attribute might contribute to a better result, the omission of back button information also distorts the measurements stratum, compactness, recurrence rate, home page rate, time per page and time per exercise attempt.

CONCLUSION

In this study, the problem of automatically diagnosing students' field-dependence style by means of machine learning techniques is addressed. For the application of styles in adaptive hypermedia systems, a reliable diagnosis is fundamental because no adaptation is still better than a false adaptation. A set of behavioural attributes was defined, based on promising empirical findings reported in literature. The applied techniques turned out not to be capable of deriving a reliable predictive model. A subsequent analysis using Pearson's correlation coefficient could also not confirm the results of prior studies. Several potential reasons thereof were discussed. Especially, general concerns arise whether cognitive styles can be diagnosed from a certain behaviour which is undoubtedly the product of a multitude of individual factors.

Summary

In this article, the SIAM system is presented for the automatic analysis of user actions in web-based learning environments. It has been used in investigations with students from an introductory mathematics course to explore user behaviour in **ActiveMath** in relation to non-observable variables. The relationships between cognitive style and students' behaviour found in prior studies could not be replicated.

A decision tree has been generated that reveals some interesting underly-

ing aspects in the log data, which can be used for prediction with new users of the system.

Acknowledgement

This publication was funded by the iClass project in the 6th Framework Programme of the European Union (Contract IST-2003-507922). The author is solely responsible for its content. The data used in the second investigation was collected in a study designed, led, and conducted by Maria Margeti as part of her doctoral work at the Institute of Education, University of London.

References

Arroyo, I., Murray, T., & Woolf, B. P. (2004). Inferring unobservable learning variables from students' help seeking behaviour. In *Proceedings of the workshop Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes at the International Conference on Intelligent Tutoring Systems ITS 2004*, (pp. 29-38), Maceio, Brasil, August.

Benchaffai, M., Debord, G., Merceron, A., & Yacef, K. (2004). TADA-Ed, a tool to visualize and mine students' online work. In Collis, B. (ed.), *Proceedings of the International Conference on Computers in Education ICCE 2004*, (pp. 1891-1897), Melbourne, Australia.

Bratitsis, T., & Dimitracopoulou, A. (2005). Data recording and usage interaction analysis in asynchronous discussions: The D.I.A.S. system. In *Proceedings of the workshop on Usage Analysis in Learning Systems at the 12th International Conference on Artificial Intelligence in Education AIED 2005*, (pp. 17-24), Amsterdam, The Netherlands, July.

Chen, S. Y., & Macredie, R. D. (2002). Cognitive styles and hypermedia navigation: Development of a learning model. *Journal of the American Society Information Science and Technology, 53*(1), 3-15.

Fischer, S., Klinkenberg, R., Mierswa, I., & Ritthoff, O. (2002). *Yale: Yet another learning environment - Tutorial*. Technical Report No. CI-136/02, Collaborative Research Center 531, University of Dortmund.

Ford, N., & Chen, S.Y. (2000). Individual differences, hypermedia navigation and learning: An empirical study. *Journal of Educational Multimedia and Hypermedia, 9*(4), 281-311.

Ford, N., & Chen, S.Y. (2001). Matching/mismatching revisited: An empirical study of learning and teaching styles. *British Journal of Educational Technology. 32*(1), 5–22.

Heiner, C., Beck, J., & Mostow, J. (2004). Lessons on using ITS data to answer educational research questions. In *Proceedings of the workshop Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes at the International Conference on Intelligent Tutoring Systems ITS 2004*, (pp. 1-9), Maceio, Brasil, August.

Kim, K. S. (2000). Effects of cognitive style on web search and navigation. In *Proceedings of the Conference on Educational Multimedia, Hypermedia & Telecommunications ED-MEDIA 2000*, (pp. 496-501), Montreal, June.

Kralisch, A., & Berendt, B. (2004). Cultural determinants of search behaviour on websites. In *Proceedings of the Sixth International Workshop on Internationalisation of Products and Systems: Designing for Global Markets*, (pp. 61-74), Vancouver.

Last, D. A., O'Donnell, A. M., & Kelly, A. E. (2001). The effects of prior knowledge and goal strength on the use of hypermedia. *Journal of Educational Multimedia and Hypermedia, 10*(1), 3-25.

McEneaney, J. E. (2001). Graphic and numerical methods to assess navigation in hypertext. *International Journal of Human-Computer Studies, 55*, 2001, 761-786.

Melis, E., Büdenbender, J., Andres, E., Frischauf, A., Goguadze, G., Libbrecht, P., Pollet, M., & Ullrich, C (2001). ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education, 12*(4), 385-407.

Melis, E., Goguadze, G., Homik, M., Libbrecht, P., Ullrich, C., & Winterstein, S. (2006). Semantic-aware components and services of ActiveMath. *British Journal of Educational Technology, 37*(3), 405-423.

Merceron, A., & Yacef, K. (2003). A web-based tutoring tool with mining facilities to improve learning and teaching. In *Proceedings of the 11th International Conference on Artificial Intelligence in Education AIED 2003*, (pp. 201-208), IOS Press.

Merceron, A., Oliveira, C., Scholl, M., & Ullrich, C. (2004). Mining for content re-use and exchange - Solutions and problems. In *Poster Proceedings of the 3rd International Semantic Web Conference ISWC 2004*, (pp. 39-40), November.

Merceron, A. & Yacef, K. (2005). Educational data mining: A case study. In *Proceedings of the 12th International Conference on Artificial Intelligence in Education AIED 2005*, (pp. 467-474), Amsterdam, The Netherlands, IOS Press.

Mostow, J. (2004). Some useful design tactics for mining ITS data. In *Proceedings of the workshop Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes at the International Conference on Intelligent Tutoring Systems ITS 2004*, (pp. 20-28), Maceio, Brasil, August.

Mühlenbrock, M. (2005). Automatic action analysis in an interactive learning environment. In *Proceedings of the workshop on Usage Analysis in Learning Systems at the 12th International Conference on Artificial Intelligence in Education AIED 2005*, (pp. 73-80), Amsterdam, The Netherlands, July.

Oliveira, C., & Domingues, M. (2003). Data warehouse for strategic management of an elearning system. In *Proceedings of the 2nd International Conference on Multimedia and ICT in Education m-ICTE 2003*, (pp. 627-631), Badajoz, Spain, December.

Quinlan, R. (1993). C4.5: *Programs for machine learning*. Morgan Kaufmann Publishers, San Mateo, CA.

Riding, R., & Rayner, S. (1998). *Cognitive styles and learning strategies*. David Fulton Publishers, London.

Tauscher, L., & Greenberg, S. (1997). Revisitation patterns in World Wide Web navigation. In *Proceedings of the Conference on Human Factors CHI 1997*, (pp. 399-406), ACM Press.

Witten, I. H., & Frank, E. (1999). *Data mining*. Morgan Kaufmann, San Francisco.

Zhang, J., & Lu, J. (2001). An educational data mining prototype. In *Proceedings of the 10th International Conference on Artificial Intelligence in Education AIED 2001*, (pp. 616-618), IOS Press.

Zhang, M., & Noyes, J. (2006). Riding's cognitive style modal: A stable construct? In *Proceedings of the 11th annual conference of the European Learning Styles Information Network ELSIN*. University of Oslo, Norway.