Automated Ranking of Hints

Sylvia Encheva

Abstract—The importance of hints in an intelligent tutoring system is well understood. The problems however related to their delivering are quite a few. In this paper we propose delivering of hints to be based on considering their usefulness. By this we mean that a hint is regarded as useful to a student if the student has succeeded to solve a problem after the hint was suggested to her/him. Methods from the theory of partial orderings are further applied facilitating an automated process of offering individualised advises on how to proceed in order to solve a particular problem.

Keywords—Decision support services, uncertainty management, partial orderings.

I. INTRODUCTION

I NTELLIGENT Tutoring Systems (ITS) facilitate students learning. They also provide fine grained tracking and assessment of students cognitive and metacognitive behaviors, [12].

An ITS is defined in [23] as a "computer system" based on "artificial intelligence" designed to deliver content and provide feedback to its user. In [24] an ITS is considered to be a software agent that provides customised and adaptable instructions and feedback to learners. The topic received a lot of attention from the research community since the 1980’s. However it seems that the process of providing adequate help to users based on their individual responses is still open.

Usually the level of usefulness of a service is established by sending users questionnaires and summarising the obtained responses. Naturally new users lack indept understanding of the subject they study and cannot really judge to which extend a particular hint is facilitating the learning process. What they actually express is their overall likings of the tool. Among other factors that influence their responses are friends’ opinions on the matter, student’s degree of interest in that subject, honesty, i.e. are their responses anonymous or a sender can be tracked down and so on. A response time model for bottom-out hints is presented in [18]. The presented model predicts learning and at the same time captures behaviours related to self-explanation.

This work is intended to facilitate automated provision of hints via an intelligent tutoring system, applying mathematical methods from partially ordered sets. Our approach to evaluate hints will avoid the influence of the above mentioned subjective factors since a hint in this work is regarded as useful to a student if the student has succeeded to solve a problem after using it.

The rest of the paper is organised as follows. Section II contains definitions of terms used later on. Section III explains how to rank hints according to personal responses and Section IV is devoted to a system description. Section V contains the conclusion of this work.

II. BACKGROUND

The process of delivering hints in an intelligent tutoring system has been discussed in [22]. A taxonomy for automated hinting is developed in [19]. The role of hints in a Web based learning systems is considered in [7].

Research-based good practice addressing the pedagogical, operational, technological, and strategic issues faced by those adopting computer assisted assessment is described in [8], [9], and [10]. Expert and theoretical knowledge about the use of technology for assessment is offered in [14]. A method enabling the instructor to do a post-test correction to neutralize the impact of guessing is developed in [11]. The theory and experience discussed in the above listed literature was used while developing our assessment tools.

A personalized intelligent computer assisted training system is presented in [15]. An intelligent tutoring system that uses decision theory to select the next tutorial action is described in [13]. A model for detecting student misuse of help in intelligent tutoring systems is presented in [2], [3]. An investigation of whether a cognitive tutor can be made more effective by extending it to help students acquire help-seeking skills can be found in [12].

A major issue in Intelligent tutoring systems is off-task student behavior known as ‘gaming’ aiming at quick advancement through a curriculum, [20]. Gaming is the systematic use of tutor feedback and help methods as a means to obtain correct answers with little or no work, [20]. A proliferation of hint abuse (e.g., using hints to find answers rather than trying to understand) was found in [1] and [12]. However, evidence that used appropriately, on-demand help can have a positive impact on learning was found in [16], [17], and [21].

A. Ordered Sets

Two very interesting problems are considered in [5], namely the problem of determining a consensus from a group of orderings and the problem of making statistically significant statements about ordering.

Definition 1: An ordered set (or partially ordered set or poset) is an ordered pair \((\mathcal{P}, \leq)\) of a set \(\mathcal{P}\) and a binary relation \(\leq\) contained in \(\mathcal{P} \times \mathcal{P}\) called the order (or the partial order) on \(\mathcal{P}\) such that

1) The relation is \(\leq\) reflexive. That is, each element is related to itself;

\[ \forall p \in \mathcal{P} : p \leq p \]
2) The relation \( \leq \) is antisymmetric. That is, if \( p \) is related to \( q \) and \( q \) is related to \( p \), then \( p \) must equal \( q \):

\[
\forall p, q \in P : ([p \leq q] \land [q \leq p]) \Rightarrow (p = q)
\]

3) The relation \( \leq \) is transitive. That is, if \( p \) is related to \( q \) and \( q \) is related to \( r \), then \( p \) must equal \( r \):

\[
\forall p, q, r \in P : ([p \leq q] \land [q \leq r]) \Rightarrow (p \leq r).
\]

A relation \( R \) is an *indifference* relation when given \( A \equiv B \) neither \( A > B \) nor \( A < B \) has place in the componentwise ordering. A partial ordering whose indifference relation is transitive is called a *weak ordering*.

If given two alternatives, a person is finally choosing only one. The natural extension to more than two elements is known as the 'majority rule' or the 'Condorcet Principle'. A relation \( R(L_1, L_2, \ldots, L_n) \) is constructed by saying that the pair \((a, b) \in R\) if \((a, b)\) belong to the majority of relations \( L_i \).

Next three linear orderings \( a \ b \ c \), \( b \ c \ a \), and \( c \ a \ b \) leading to

\[
R = \{(a, b), (b, c), (c, a)\}
\]

(three-way tie), illustrate the 'paradox of voting'.

A 'social welfare function' maps \( k \)-tuples of the set of linear orderings of any \( b \subset A \) to single linear orderings of \( B \), where \( A \) is a set of at least three alternatives.

Two elements \( a \) and \( b \) where \( a \neq b \) and \( a, b \in P \) are comparable if \( a \leq b \) or \( b \leq a \), and incomparable otherwise. If \( \forall a, b \) where \( a, b \in P \) are comparable, then \( P \) is chain. If \( \forall a, b \) where \( a, b \in P \) are incomparable, then \( P \) is antichain.

### B. Fuzzy Functional Dependencies

**Definition 2:** Let \( r \) be a fuzzy relation instance on a scheme \( R(A_1, \ldots, A_n) \), and \( U \) be the universal set of attributes \( A_1, \ldots, A_n \), where \( X, Y \subset U \). A fuzzy relation instance \( r \) satisfies the fuzzy functional dependency \( (F \in D)X \rightarrow Y \) if, for any \( t_i, t_j \in r \),

\[
SS(t_i[X], t_j[X]) \leq SS(t_i[Y], t_j[Y]).
\]

**Inference rules**

- **Reflexive rule:**
  
  if \( y \subseteq x \) then \( x \rightarrow y \) holds.

- **Augmentation rule:** \( (x \rightarrow y) \Rightarrow (xz \rightarrow yz) \).

- **Transitivity rule:** \( (x \rightarrow y, y \rightarrow z) \Rightarrow (x \rightarrow z) \).

### III. Partial Ordering

In this scenario students are suggested to solve problems via an intelligent tutoring system. The system provides assistance in a form of hints on a user request. Three types of hints called \( a, b, \) and \( c \) respectively are available. They can contain f. ex. theoretical rules, hints and solutions of similar problems. The students' responses are saved in a database. The goal is to find out in which order hints related to a particular problem should be presented to a new student and in which order hints should be presented to a student who has been using the system when the student is requesting help with respect to a new problem. The system recognises a hint as a useful one if a student provides a correct answer after that hint has been presented to her/him.

Using nested parentheses one can represent the partial ordering structure in an application internally. Let the alphabets in this definition be \( (., a, b, c) \). The use of parentheses is to signify the position of ordering of a particular element 'a', 'b', or 'c', where

- \( (.) \) signifies level 1 - the top most level,
- \( (((())))) \) signifies level 2, and
- \( ((((())))) \) signifies level 3.

Thus f. ex., the partial ordering in Fig. 1 is represented as \( (a(b(c))) \), in Fig. 2 is represented as \( (a(b(c))) \), in Fig. 3 is represented as \( (ab(c)) \), and in Fig. 4 is represented as \( (a(b)c) \).

According to the theory of partial orderings the hints \( a, b, \) and \( c \) can be related as in Fig. 1, Fig. 2, Fig. 3, and Fig. 4. Note that the set \( a, b, c \) is neither a chain nor an antichain, i.e. a pair of different elements can be either comparable or incomparable.

The ordering in Fig. 1 implies that

- hint \( a \) is helpful for the majority of students,
- hint \( b \) is helpful to a smaller number of students, and
- hint \( c \) is helpful to even smaller number of students.

Fig. 2 illustrates a partial ordering where hint \( a \) is the most helpful of the three hints while hints \( b \) and \( c \) are of similar level of helpfulness.

The ordering in Fig. 3 implies that hint \( c \) is less helpful of the three, while hint \( a \) and \( b \) turn to be equally useful.

The ordering in Fig. 4 implies that hint \( a \) is helpful to most of the students, hint \( b \) is helpful to a lesser number of students, and hint \( c \) has not helped anybody.
IV. SYSTEM DESCRIPTION

A system prototype is build as a Web-based application using Apache HTTP server [26], mod_python module [27] and SQLite database [28]. The mod_python module provides programmable runtime support to the HTTP server using Python programming language. The whole application components are

1) Web-based user interface,
2) application logic and application interfaces written in Python, and
3) relational database.

The system architecture is presented in Fig. 6. The system contains also

- an interactive theoretical part (lecture notes),
- problems to be solved at home,
- tests assessing recall of facts, the ability to map the relationship between two items into a different context, and conceptual thinking, asking students to evaluate consequences and draw conclusions, and
- various quizzes.

The users, i.e. expert tutors, teachers, and students interact with the system using Web forms. Before any interaction with the system can take place, a user needs to be authenticated first. Experts and teachers can submit and update data, while students can only view information.

For a particular subject, an expert tutor will first submit data that will be used to construct a data table.

The system will then check that there are no duplicate attribute combinations and insert the context data into the database.

The system provides recommendations on whether or not a student needs to take additional classes (courses) based on fuzzy dependencies.

V. CONCLUSION

Evaluating hints based on students’ responses is more accurate than using questionnaires. Such an approach is not affected by subjective opinions and provides useful feedback to content developers.

The presented automated tutoring system is a response to the increased demand for the necessity of developing
effective learning tools that can be smoothly integrated in the educational process.

The system uses each student’s diagnostic reports on misconceptions, and lack of knowledge and offers advice in the form of additional reading, hints and tests and recommends an interaction with the human tutor when needed.

REFERENCES


