Adaptive Assessment and Training Using the Neighbourhood of Knowledge States⋆

Cornelia E. Dowling, Cord Hockemeyer, Andreas H. Ludwig

Technische Universität Carolo Wilhelmina zu Braunschweig
Institut für Psychologie
Abteilung für Mathematische und Sozialpsychologie
38092 Braunschweig, Germany
E–Mail: { C.Dowling | C.Hockemeyer | A.Ludwig } @ tu–bs.de

Abstract. In this paper, we suggest methods for the adaptive assessment and training of students. The main idea is to apply the theoretical concept of the neighbourhood of a knowledge state to adaptive training. We also describe improvements of existing assessment algorithms. These ideas are implemented in a multi–platform tutoring system shell. This shell has been applied to two different fields of knowledge.

1 Introduction

Doignon and Falmagne (1985) suggested a mathematical framework for the adaptive assessment of knowledge. Subsequently, deterministic and probabilistic algorithms were developed which are suitable for posing questions to a student dependent on his/her individual knowledge (Falmagne and Doignon, 1988a,b). Such an adaptive assessment algorithm requires a large pool of *items* from a specified field of knowledge. These items are, for example, classes of equivalent examination questions as they are used in schools.

For each individual student, a small number of questions are sequentially selected from this large item pool, and presented to the student. With each new question, the student's answers to the previous questions are taken into account. In this manner, selecting new questions is increasingly adapted to the student's individual knowledge. As a result of assessing the student, we obtain the student's *knowledge state*. A knowledge state has been defined as the subset of items from the large item pool that can be mastered by the student (Doignon and Falmagne, 1985). Such an adaptive procedure makes use of *prerequisite relationships* between the items. Prerequisite relationships are, for example, obtained by querying experts with the help of a computerized procedure (Dowling and Kaluscha, 1995). The following example illustrates a set of items and their prerequisite relationships.

⋆ Published in Frasson, Cl., Gauthier, G., & Lesgold, A. (1996) *Intelligent Tutoring Systems*, Proceedings of the ITS'96, Montréal, Canada, *Lecture Notes in Computer Science*, vol. 1086, pp. 578–586.

Example 1. We show five items from a project, in which we developed a system with 77 items from the field of fractions (Baumunk, 1995).

a Subtraction of fractions without a common denominator.
b Addition of mixed numbers.
c Addition of fractions without a common denominator.
d Addition of fractions with a common denominator.
e Finding the least common multiple.

For these items, one expert determined the prerequisite relationships represented by the following statements:

1. *If a student masters item a, then he/she masters at least one of its prerequisites b, or c.*
2. *If a student masters item b, then he/she also masters its prerequisite d.*
3. *If a student masters item c, then he/she also masters all of its prerequisites d and e.*

If these prerequisite relationships are valid, then only those subsets of the item set $\{a, b, c, d, e\}$ can be considered to be knowledge states which do not "contradict" the statements 1, 2, and 3 above. For example, the set $\{a, b\}$ is not a knowledge state because it contains item b, but not its prerequisite item d, and hence contradicts statement 2. A subset contradicting statement 1 is, for example, the set $\{a, d, e\}$ which is not a knowledge state, either. Examples of knowledge states are the sets $\{d\}$, $\{b, d\}$, and $\{b, d, e\}$.

For a given set of prerequisite relationships, the set of all knowledge states constitutes a *knowledge space*. A knowledge space $\mathcal{K}$ is a family of subsets of a finite item set Q which contains the empty set $\emptyset$ and the item set Q itself, and which is closed under union, i. e. the set union of each subset of the family $\mathcal{K}$ is an element of $\mathcal{K}$.

Example 2. The set of all knowledge states which are possible if the prerequisite relationships in Example 1 hold, are illustrated by the Hasse–diagram of Fig. 1, which shows that the set is closed under union. The line surrounding some of the states marks the neighbourhood of the knowledge state $\{b, d\}$. The concept of the neighbourhood of a knowledge state will be introduced below.

In one of their probabilistic assessment procedures, Falmagne and Doignon (1988b) use the concept of the *neighbourhood* of a knowledge state which comprises all other states with a distance of at most one. The distance $d(K, K')$ between two knowledge states K and K' is equal to the size of their symmetric set difference, $d(K, K') = |K \Delta K'| = |(K \setminus K') \cup (K' \setminus K)|$. They use this concept to define the *fringe* of a knowledge state which is defined to be the set of items by which the knowledge state differs from its neighbours.

The main suggestion of this paper is to apply the concept of the fringe of a knowledge state not only to assessment but also to training. Suppose that, for each item, we have several equivalent problems which are designed to test whether this item is mastered or not. Equivalent problems for testing item a

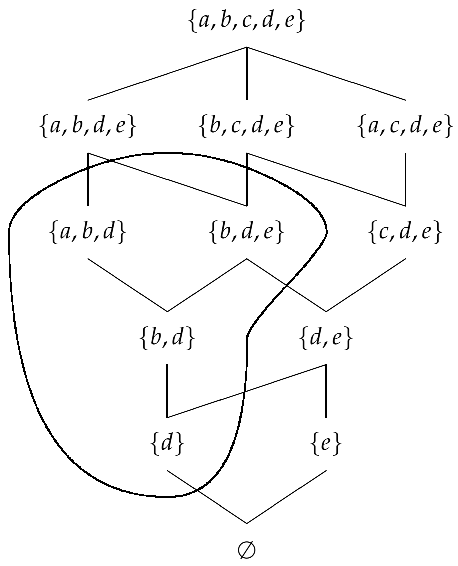


Fig. 1. Neighbourhood of knowledge states

in Example 1 are, for instance, $\frac{3}{10} - \frac{1}{4}$, and $\frac{2}{15} - \frac{1}{10}$. A method that we recommend for selecting problems for training is to take items from the fringe of the student's knowledge state, and to offer problems belonging to these items as exercises. In that manner we are able to adapt the exercises to his/her individual knowledge.

Example 3. Consider the knowledge space $\mathcal{K}$ from Example 2. The neighbourhood of the knowledge state $K = \{b, d\}$ is the set $N(K) = \{\{d\}, \{b, d\}, \{a, b, d\}, \{b, d, e\}\}$. As Fig. 1 shows this is the set of all states which can be reached from the knowledge state $K = \{b, d\}$ in 0 or 1 steps following the edges of the Hasse–diagram. The fringe of the knowledge state K is the set $F(K) = \{a, b, e\}$.

For a student with the knowledge state $\{b, d\}$, it makes sense to practice item b, since it is the item which has been learned most recently. The items a and e are those new items which are recommended to be learned next, since all of their prerequisites are known.

In Sect. 2, we suggest an efficient procedure for the assessment of students' knowledge states and for computing their fringes. In Sect. 3, we describe a robust multi–platform tool which we use for adaptive assessment and training in the neighbourhood of knowledge states, and as a system for authoring new problems. The system is a shell which enables the usage of different assessment and training algorithms with different structures of prerequisite relationships and sets of items.

2 Efficient Procedures for the Adaptive Assessment and Training

Applications of knowledge space theory to different fields of knowledge have shown that the resulting knowledge spaces can grow very large, and nevertheless be efficient for assessment (Baumunk, 1995). For these large knowledge spaces, the algorithms by Falmagne and Doignon (1988a,b) for assessing a knowledge state and for computing its fringe need a large amount of computer memory, and computing time. In this section, we introduce more efficient algorithms for the adaptive assessment, and for determining the fringe of a knowledge state. The structural information about the knowledge domain represented by knowledge spaces can also be represented by other, equivalent structures used by the new algorithms.

A knowledge state K is called *minimal* for an item $q \in K$ if there exists no other knowledge state K' which contains item q, and is a subset of K. A knowledge state can be minimal for several items and, vice versa, an item can have several minimal states. These minimal states cannot be built as a union of other knowledge states. The set of all knowledge states which are minimal for at least one item is called the *basis* of a knowledge space. The basis is the smallest family of knowledge states from which the complete knowledge space can be reconstructed by closure under union (Doignon and Falmagne, 1985).

Example 4. Figure 1 shows a knowledge space $\mathcal{K}$ on the item set $Q = \{a, b, c, d, e\}$. The basis $\mathcal{B}$ of this knowledge space is the set of the knowledge states $\{d\}$, $\{e\}$, $\{b, d\}$, $\{a, b, d\}$, $\{c, d, e\}$, and $\{a, c, d, e\}$. These states are minimal for the items d, e, b, a, c, and a, respectively.

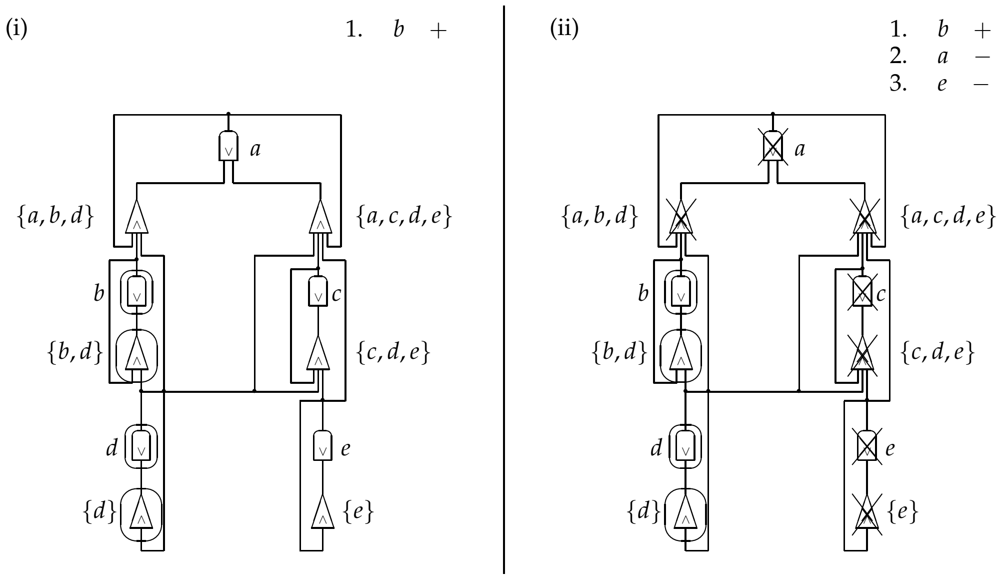
The main idea behind the basis–based assessment algorithm is to interpret the basis as a set of rules, as follows (Hockemeyer, 1993). For any set of items, we say that a student *masters* the set of items if and only if he/she masters all the items within the set. Since each knowledge state can be constructed as a union of elements of the basis, the following two statements are valid:

(i) *A student masters an item q if and only if he/she masters at least one of its minimal states.*
(ii) *A student masters an element of the basis if and only if he/she masters all the items contained in this element.*

Our assessment procedure constructs a constraint network in the following manner. The variables of this network are the mastery of the items, as well as the mastery of the elements of the basis. The relations between these variables are statements of the form (i) and (ii) above. The variables can have one out of three values: *"is mastered"*, *"is not mastered"*, and *"mastery is unknown"*. The assessment procedure can now be considered as solving a constraint satisfaction problem.

Simulation studies have shown that, as a rule, the space–based assessment procedure developed by Falmagne and Doignon (1988a,b) tests a slightly smaller number of items to complete an assessment than the basis–based procedure. On the other hand, the basis–based procedure needs much less computing time and memory than the space–based procedure (Hockemeyer, 1993).

Example 5. Consider the knowledge space used throughout this paper. The left side of Fig. 2 shows the constraint network with the items and basis elements as described above. Suppose that item b is presented to the student first, and that the student answers the question correctly. Then we can conclude that he/she also masters the basis element $\{b, d\}$, and hence, also the item d and its minimal element $\{d\}$. These items and basis elements are marked with an oval in Fig. 2(i). The items b and d are not considered any longer, since we know that the student masters them. Suppose that item a is asked next and that the student gives a false answer. We can now conclude that he/she does not master the basis elements $\{a, b, d\}$ and $\{a, c, d, e\}$, either. Assume that we choose item e next and that the student does not master this item. Now, we can infer that he/she does not master the basis elements $\{e\}$ and $\{c, d, e\}$ and hence, that he/she does not master item c either. The right side of Fig. 2 shows the resulting assessment state. We know for all items and basis elements whether the student masters them or not and, therefore, have finished the assessment. The student's knowledge state can be determined as the set of all items or the union of all basis elements which are marked as mastered by the student. This state is the set $\{b, d\}$.



Knowledge state assessed: $\{b, d\}$.

Fig. 2. Knowledge assessment using the basis of a knowledge space

For selecting the training problems from the fringe of a knowledge state, we also need a basis–based procedure to compute it. There are two ways for different cases to test whether or not an item q belongs to the fringe of a knowledge state. (i) An item q which is not a member of the knowledge state K is an element of the fringe $F(K)$ if there is a minimal element B for the item q such that $B \setminus \{q\} \subseteq K$. (ii) An item q which is a member of the knowledge state K is an element of the fringe $F(K)$ if the set difference of the knowledge state K and the union of specific basis elements is equal to the set $\{q\}$. These basis elements are those which are subset of K but do not contain the item q (Hockemeyer, 1997).

Example 6. Consider the knowledge space and the basis from the Examples 2 and 4. To compute the neighbourhood of the knowledge state $K = \{b, d\}$, we make a case distinction between the items b and d which are elements of the state K, and the items a, c, and e which are not. For item a, we have two minimal elements, $\{a, b, d\}$ and $\{a, c, d, e\}$. Since the set difference $\{a, b, d\} \setminus \{a\}$ is a subset of K, item a is a member of the fringe $F(K)$. The item c has one minimal element, $\{c, d, e\}$. The set $\{d, e\} = \{c, d, e\} \setminus \{c\}$ is not a subset of K. Therefore, c is not a member of $F(K)$. For item e, we get $\{e\} \setminus \{e\} = \emptyset$. The empty set $\emptyset$ is a subset of K and, hence, e is an element of the fringe $F(K)$.

The items b and d are members of the knowledge state K. Therefore, we have to use the other procedure for these items. For item b, there exists one element $\{d\}$ of the basis which is subset of $\{b, d\}$ and does not contain item b. We determine the ordinary set difference $\{b, d\} \setminus \{d\} = \{b\}$. Therefore, the item b is an element of the fringe $F(K)$. For item d, there exists no element of the basis which is a subset of $\{b, d\}$ and does not contain item d. So, we have to compute the set difference of the knowledge state K and the empty set $\emptyset$. This is the state K which is unequal to the set $\{d\}$. Therefore, item d is not a member of the fringe $F(K)$. As a result, we obtain the set $\{a, b, e\}$ to be the fringe $F(K)$.

3 A Tool for Assessment and Training

To put the algorithms introduced above into practice, Ludwig (1995) and Winkelmann (1995) developed a shell for the adaptive testing and training, ADASTRA[1]. ADASTRA provides a graphical user interface, and a programming language for computer–based problems. The motivation for building this adaptive assessment and training tool was to be independent of professional software distributors, and to be able to run it on different platforms.

The problems can be presented either in *assessment mode* or in *training mode*. In both modes, ADASTRA presents the problems on the screen, and evaluates the student's input. The training mode offers *feedback* which informs the student on the type of his/her errors. In this mode, the student is also able to request

[1] ADASTRA is implemented in C++ with the wxWindows 1.5 library and runs under MS–Windows 3.11 and Linux. The wxWindows library was developed by Julian Smart et al., University of Edinburgh, Artificial Intelligence Applications Institute, 80 South Bridge, Edinburgh, UK.

help texts. The language for programming testing and training problems contains elements for the layout of the problems, as well as for the evaluation of the students' answers. These elements are either general or specific for the knowledge domain. An example of a domain specific function is *'fraction(...)'* which can be used for displaying its arguments above and under the line. ADASTRA includes components for presenting the problems, and a development environment for the author of the problems. The former is called *problem presenter*, the latter *authoring environment*. The architecture of the system ADASTRA is shown in Fig. 3.

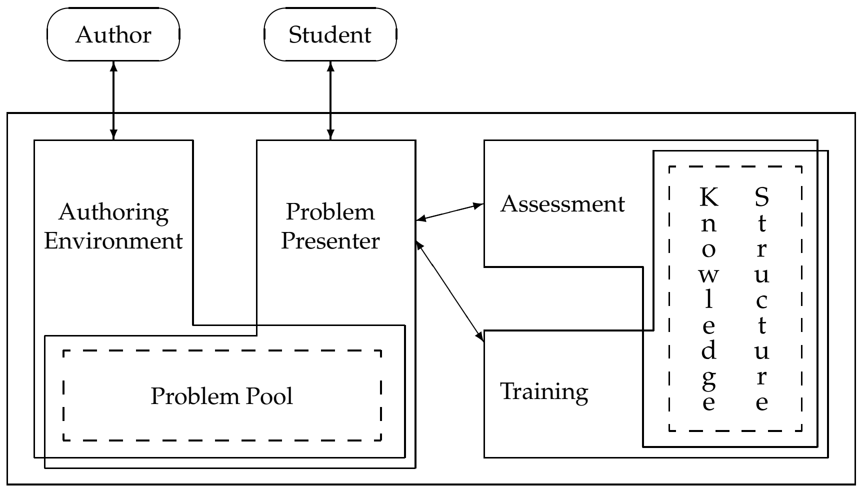


Fig. 3. Architecture of the system ADASTRA

- The authoring environment is a frame, providing file handling, editing, and testing of the problem programs. Additionally, the author can specify which problem programs belong to an item. In the following, the set of all problem programs is called the *problem pool*.
- The problem presenter is the assessment and training environment for the student. It communicates with the assessment or training object, and subsequently presents the problems to the student. This communication is described in more detail below.
- The *assessment object* encapsulates the implementation of an assessment algorithm as, for example, described in Sect. 2. The implemented assessment algorithm accesses the knowledge structure as shown in Fig. 3, and communicates with the problem presenter via the assessment object.
- Figure 3 shows, that the *training object* encapsulates the implementation of the training algorithm described in the Sects. 1 and 2. The implemented training algorithm also accesses the knowledge structure, and communicates with the problem presenter via the training object.

ADASTRA stores the relation on problem programs and items. The problem presenter uses this information. This design has the advantage that the interface between assessment and training object and the problem presenter is simple. In the direction from assessment or training object to problem presenter only an item number is passed and in the reverse direction only a truth value is transmitted. The problem presenter does not access the knowledge structure, and the assessment and training objects do not access the problem pool. The assessment object and its implementation of the assessment algorithm can be exchanged for another one as can the training object with its implementation of the training algorithm and the knowledge structure.

The problem programs are written in the programming language ProLa, an abbreviation for "Problem Language". In each ProLa program, three tasks are specified: (i) Presenting a problem and optionally offering some help text, (ii) reading the student's input, and (iii) checking a student's input for correctness. In step (iii) ProLa programs can display feedback to the student. Figure 4 illustrates how the student sees a problem on the screen. This problem belongs to the item (a) *Subtraction of fractions without a common denominator* from Example 1. It also shows the help frame which is displayed whenever the student clicks on the help button in training mode.

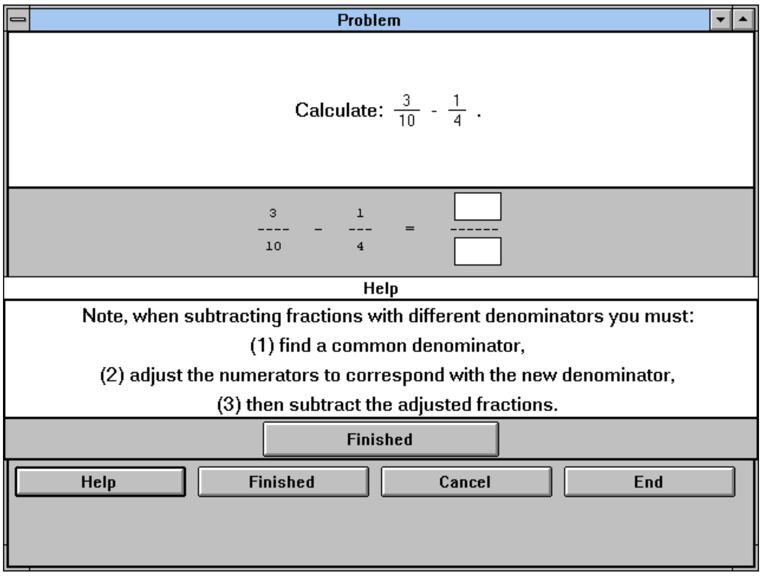


Fig. 4. Display of a problem after clicking on the help button

Presently ADASTRA runs with 77 items from the field of *fractions*. In a second project, a prototype of an assessment and training system in the domain of tax law is being developed.

4 Discussion

We have described an efficient system for the adaptive assessment and training based on the theory of knowledge spaces. It is designed such that the knowledge domains, the knowledge structure representing the prerequisite relationships, and the assessment module can be exchanged easily. Since the teachers are heavily involved in selecting the items, the resulting problem pool is close to the actual curriculum. There are several aspects in which this approach to assessment and training differs from tutoring systems based on cognitive theories as, for example, described by Anderson et al. (1995). The concept of the item as it is used in knowledge space theory integrates both, procedural as well as declarative knowledge. The level of granularity of the items is so fine that items which build on each other differ only very little. Therefore, suitable help can be offered to the student even without cognitive modeling.

Bibliography

Anderson, J. R., Corbett, A. T., Koedinger, K. R., and Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*, 4:167–207.

Baumunk, K. (1995). Die empirische Erhebung einer Struktur von Schulwissen im Fach Mathematik. Diplomarbeit, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany.

Doignon, J.-P. and Falmagne, J.-C. (1985). Spaces for the assessment of knowledge. *International Journal of Man-Machine Studies*, 23:175–196.

Dowling, C. E. and Kaluscha, R. (1995). Prerequisite relationships for the adaptive assessment of knowledge. In Greer, J., editor, *Artificial Intelligence in Education, 1995*, pages 43–50, Charlottesville, VA. Association for the Advancement of Computing in Education (AACE).

Falmagne, J.-C. and Doignon, J.-P. (1988a). A class of stochastic procedures for the assessment of knowledge. *British Journal of Mathematical and Statistical Psychology*, 41:1–23.

Falmagne, J.-C. and Doignon, J.-P. (1988b). A Markovian procedure for assessing the state of a system. *Journal of Mathematical Psychology*, 32:232–258.

Hockemeyer, C. (1993). Wissensdiagnose auf Wissensräumen. Diplomarbeit, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany.

Hockemeyer, C. (1997). Using the basis of a knowledge space for determining the fringe of a knowledge state. *Journal of Mathematical Psychology*, 41:275–279.

Ludwig, A. H. (1995). Eine themenorientierte Programmiersprache für die Präsentation und Auswertung computergestützter Test– und übungsaufgaben. Diplomarbeit, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany.

Winkelmann, J. (1995). Ein objektorientiertes Maschinenmodell für die Präsentation und Auswertung computergestützter Test– und übungsaufgaben. Diplomarbeit, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany.