

Skill Assessment in Problem Solving and Task Simulation

Luca Stefanutti

Department of Psychology, University of Graz, Austria
luca.stefanutti@uni-graz.at

Dietrich Albert

Department of Psychology, University of Graz, Austria
dietrich.albert@uni-graz.at

Abstract: Simulated learning environments provide an efficient means for improving individual skills in specific problem solving and learning situations. One crucial aspect of an optimal system for simulated training environments is its capability to keep track of the improvements of the user along the whole training process. In this paper we present a set-theoretical formal framework that can be applied for the efficient assessment of the skills of an individual in a simulated learning environment. The basic concept underlying our approach is that of a functional skill mapping of the simulated learning environment through *problem spaces*.

Key Words: skill assessment, problem solving, simulated learning environment, knowledge space, problem space

Category: J.4 Social and Behavioral Sciences

1 Introduction

Simulated training environments, like virtual reality or computerized training systems, provide an efficient means for improving individual skills in specific problem solving and learning situations. A simulated environment is usually cheaper and safer than a real one. Moreover, simulation allows reversibility of the user's action (i.e., the user can always 'redo' or 'undo' an action or a move), which is not always possible in a real environment. Furthermore, in a simulated environment, complex cognitive tasks can be decomposed into simpler sub-problems that are well-suited to the actual skills and competencies of the learner (see e.g., [Lee & Anderson, 2001]). This facilitates inductive/deductive reasoning, exercise and insight which, in turn, allow the user to learn new skills and competencies (or to strengthen existing ones) in the domain of the problem. An optimal training system is a training system which triggers and keeps this virtuous circle steady along the whole learning process.

The development of simulated training environments is a necessary, but not a sufficient condition for an optimal and efficient training of a learner in complex cognitive tasks. One crucial aspect of an optimal system for simulated training environments is its capability to keep track of the improvements of the user

along the whole training process. This implies a dynamic adaptation of the system to the user's skills and performance (personalisation) so that her/his motivation and mental activity remains at an optimal level during the whole training session. One mechanism at the basis of this adaptation is performance and skill assessment and monitoring.

The basic idea underlying our approach is a functional (skill) mapping of the simulated learning/training environment. When the learner enters the simulated environment, s/he finds her/himself in some initial state, and her/his objective is to move to some final (solution) state by performing appropriate actions, operations and moves. From a cognitive perspective, when the user tackles a new problem, s/he uses a number of strategies that involve, among others, inductive and deductive reasoning, learning by trials and errors and insight. Whatever the strategies are, to solve a problem the user performs a sequence of (either mental or concrete) operations that allow her/him to move from one state to another until the final (solution) state of the problem is reached [Simon & Reed, 1976]. In the formal framework that we present in the next section a *problem space* is a set Q of *problem states* connected by operations, and a *problem* is an ordered pair $\langle a, b \rangle$ of problem states such that state b can be reached from state a through a suitable sequence of operations.

One obvious consequence is that if an individual is not capable of performing an operation (or a sub-sequence of operations) contained in this sequence then s/he will fail to solve the problem. A failure, however might occur also for other reasons. Problems can be constructed by transitivity: if $\langle a, b \rangle$ and $\langle b, c \rangle$ are both problems, then $\langle a, c \rangle$ is a problem too. We make a distinction between *optimal* and *human* problem solvers. If an optimal problem solver is able to solve both $\langle a, b \rangle$ and $\langle b, c \rangle$, then s/he always solves $\langle a, c \rangle$ by transitivity. This, however, does not hold in general for human problem solvers. In particular, we assume that — as far as a human problem solver is concerned — solving both $\langle a, b \rangle$ and $\langle b, c \rangle$ is not a sufficient condition for solving $\langle a, c \rangle$. The operation that combines $\langle a, b \rangle$ and $\langle b, c \rangle$ (or, in general, many different sub-problems) together is, itself, a mental operation (a kind of meta-operation) that occurs in the mind of the problem solver either by deductive reasoning or by insight.

In our framework a *skill* is regarded as the capability of performing a given sequence of operations, where the elementary sequences are the single operations, and the *skill state* of an individual is the collection of all the skills possessed by this individual. In the next sections we face the question of uncovering the skill state of an individual in an efficient way during the training process. Our approach can be viewed as an extension of knowledge space theory ([Albert & Lukas, 1998]; [Doignon & Falmagne, 1999]) to simulated training environments.

2 Problem spaces

A *problem space* is a labeled directed graph $\mathbf{P} := \langle Q, \Omega, v \rangle$ in which Q is a set of nodes called (problem) states, Ω a set of operations, and $v : Q \times \Omega \rightarrow Q$ a partial function specifying the edges of the graph. Nodes in the graph are labeled by elements in Q , and edges are labeled by elements in Ω .

A *string* on Ω is a sequence $\langle o_1, o_2, \dots, o_n \rangle$ of operations in Ω . In the sequel we use lowercase Greek letters like ' σ ' to denote strings on Ω , and the empty string $\langle \rangle$ is denoted by the letter ϵ , fixed throughout. The *concatenation* of two strings $\alpha = \langle a_1, a_2, \dots, a_m \rangle$ and $\beta = \langle b_1, b_2, \dots, b_n \rangle$ is the string $\alpha\beta = \langle a_1, a_2, \dots, a_m, b_1, b_2, \dots, b_n \rangle$. The collection of all strings on Ω is the closure Ω^* under concatenation of Ω defined by

$$\Omega^* = \bigcup_{n \geq 0} \Omega^n.$$

Note that $\epsilon\alpha = \alpha\epsilon = \alpha$ holds for all $\alpha \in \Omega^*$.

An extension V of the function v to strings is defined (recursively) as follows: if $q \in Q$, $o \in \Omega$ and $\sigma \in \Omega^*$ then

$$V(q, \epsilon) := q; \tag{1}$$

$$V(q, o\sigma) := V[v(q, o), \sigma]. \tag{2}$$

A pair $\langle a, b \rangle$ in $Q \times Q$ is called a (solvable) *problem* in the problem space \mathbf{P} if and only if $V(a, \sigma) = b$ for some $\sigma \in \Omega^*$. The set of all problems is the binary relation $P \subseteq Q \times Q$ such that, for $a, b \in Q$,

$$aPb \iff V(a, \sigma) = b \text{ for some } \sigma \in \Omega^*.$$

In other words, we say that $\langle a, b \rangle$ is a problem if state b is reachable from state a through some sequence σ of operations in the graph of \mathbf{P} . Moreover, if $V(a, \pi) = b$ for some problem $\langle a, b \rangle$ and some string π , then we say that π *solves* problem $\langle a, b \rangle$. A string solving some problem in P is called a *path* in \mathbf{P} , and the collection of all problems solvable by path π is

$$V(\cdot, \pi) = \{\langle a, b \rangle \in Q^2 : V(a, \pi) = b\},$$

while the collection of all paths (i.e., all strings solving some problem) in Ω^* is

$$\Pi := \{\pi \in \Omega^* : V(\cdot, \pi) \neq \emptyset\}.$$

In the sequel we use the terms *path* and *skill* to denote, essentially, the same kind of objects, namely sequences of operations in Ω solving some problem. However, while *path* is used, more in general, to denote sequences of operations in \mathbf{P} , the

term *skill* denotes a path that ‘belongs’ to some individual. This means that a path π solving some problem $\langle a, b \rangle$ represents the skill of some individual if this last is capable of solving $\langle a, b \rangle$ by means of π . In this sense, every skill is represented by some path, but the opposite does not hold in general.

Then we use the term *skill state* to denote the subset $S \subseteq \Pi$ of all paths that an individual is able to compute. More concretely, if S is the skill state of some individual, and $\pi \in S$ then we say that this individual is able to compute $V(a, \pi)$ for all $a \in Q$ for which $V(a, \pi)$ is defined. Thus one first assumption in our framework is that

[S1] if π is a skill in S then an individual in state S is capable of solving all problems in $V(\cdot, \pi)$.

Our second and third assumptions provide an explicit distinction between an *optimal problem solver* and a *human problem solver*.

[S2] If α and β are two actual paths, and S the skill state of an *optimal problem solver* then:

$$\alpha\beta \in S \iff \alpha, \beta \in S.$$

On the other hand,

[S3] if α and β are two actual paths, and S the skill state of a *human problem solver* then:

$$\alpha\beta \in S \implies \alpha, \beta \in S.$$

Clearly, [S2] is stronger than [S3]. Henceforth, with the term *skill state* we refer to the set of skills of a human problem solver. Thus, $S \subseteq \Pi$ is a skill state if and only if it is consistent with [S3]. We use, instead, the term *optimal skill state* to refer to subsets of Π that are consistent with [S2].

One key concept in our framework is that of *string inclusion*. If ω and π are strings then ω includes π (denoted by $\pi \leq \omega$) if $\omega = \alpha\pi\beta$ for some $\alpha, \beta \in \Omega^*$. String inclusion is reflexive, transitive, and antisymmetric, thus $\langle \Omega^*, \leq \rangle$ is a partially ordered set. It follows from [S3] that any down-set in the partially ordered set $\langle \Pi, \leq \rangle$ is, in fact, a skill state.

3 Skill maps and knowledge states

A *skill map* for the problem space \mathbf{P} is a triplet $\langle P, \Pi, f \rangle$ where $f : P \rightarrow 2^\Pi$ is a mapping such that, for any $\langle a, b \rangle \in P$,

$$f(a, b) = \{\pi \in \Pi : V(a, \pi) = b\}. \quad (3)$$

The collection $f(a, b)$ is called the set of skills assigned to $\langle a, b \rangle$ and it is, simply, the set of all paths solving $\langle a, b \rangle$. If $\langle P, \Pi, f \rangle$ is a skill map for \mathbf{P} and $X \subseteq \Pi$ a set of skills, then we say that $K \subseteq P$ is the *knowledge state delineated* by X if

$$K = \varphi(X) := \{\langle a, b \rangle \in P : f(a, b) \cap X \neq \emptyset\}, \quad (4)$$

where $\varphi : 2^\Pi \rightarrow 2^P$ is called the *disjunctive model of skill maps* ([Doignon, 1994]; [Doignon & Falmagne, 1999]), and the collection of all knowledge states delineated by subsets of Π is the image $\varphi(2^\Pi)$. This collection of knowledge states is what, in theory of knowledge spaces, is called a *knowledge space*.

4 Skill assessment

Knowledge space theory provides efficient procedures for uncovering the knowledge state (and the corresponding skill state) of an individual in an interactive computerized session by means of a knowledge space (see, e.g., [Falmagne & Doignon, 1988]). These procedures can be applied to the framework delineated above.

The above-mentioned assessment procedures have, at least, three nice features that make them appropriate in dynamic and adaptive assessment systems:

- *efficiency*: the knowledge state of an individual can be completely recovered after the presentation of a small fraction of the whole set P of problems;
- *adaptivity*: an individual is never presented with problems that are too difficult or too easy for her/him to solve.
- the procedure is stochastic and always terminates with the best estimate of the knowledge state of an individual also in the case in which this state is not stable (i.e., the user oscillates among two or more states) during the assessment process.

A detailed presentation of these procedures is beyond the scope of this paper; we refer the reader to [Falmagne & Doignon, 1988], [Doignon & Falmagne, 1999], [Dowling & Hockemeyer, 1999] for a comprehensive introduction. We just outline here the basic concepts underlying these methods (we call them the *basic assessment procedure*).

Let K^* be the (true but unknown) knowledge state of a user, and \mathcal{K} be the knowledge space on the set P of problems used for the assessment. At the outset, with no prior information about K^* , the basic assessment procedure starts with the assumption that K^* is one of the states in \mathcal{K} (we denote this by setting $\mathcal{H} = \mathcal{K}$). A first problem $\langle a, b \rangle \in P$ is presented to the user. If the problem is successfully solved by the user, all states $K \in \mathcal{H}$ not containing $\langle a, b \rangle$ are

removed from \mathcal{H} . If the problem is not solved, all states $K' \in \mathcal{H}$ containing $\langle a, b \rangle$ are removed from \mathcal{H} . Then a new step takes place and a new problem is presented until \mathcal{H} contains exactly one knowledge state. This state is the estimate of the true state K^* of the user.

5 An example

In this section we present a simple example application of the formal framework delineated above. Our example is taken from classical problem solving, and the problem is called “buckets of water”. The text of the problem is the following:

“There are two buckets A and B . The capacity of bucket A is 4 liters, and that of B is 3 liters. The two buckets are empty and there is a source of water. How to fill bucket A with exactly 2 liters of water?”

There are 6 alternative operations in this problem: (a) fill bucket A ; (b) fill bucket B ; (c) empty bucket A ; (d) empty bucket B ; (e) pour water from A to B until A is empty or B is full; (f) pour water from B to A until A is full or B is empty. A single state of the problem is denoted by an ordered pair $\langle x, y \rangle$, where x is the amount of water in bucket A , and y is the amount of water in B . Thus, for instance, $\langle 2, 3 \rangle$ denotes the state in which A contains 2 liters, and B contains 3 liters. The initial state of the problem is $\langle 0, 0 \rangle$, and the solution state is $\langle 2, 0 \rangle$. A portion of the problem space of the “buckets of water” is depicted in Figure 1.

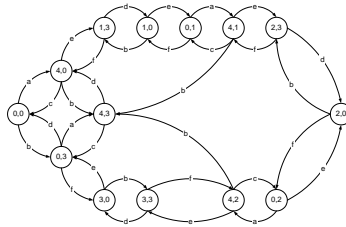


Figure 1: Portion of the problem space of the ‘buckets of water’

In this problem space, the problem $\langle \langle 0, 0 \rangle, \langle 2, 0 \rangle \rangle$ can be solved by two alternative paths, namely, $aedeaed$ or $bfbfcf$. Of course, in the problem space of Figure 1 there are a number of other problems. Some of them, like for instance $\langle \langle 1, 3 \rangle, \langle 4, 1 \rangle \rangle$ and $\langle \langle 0, 3 \rangle, \langle 4, 2 \rangle \rangle$ are sub-problems of $\langle \langle 0, 0 \rangle, \langle 2, 0 \rangle \rangle$.

To construct a knowledge space on this problem space we first derive the partially ordered set $\langle \Pi, \leq \rangle$ of paths, where each path is a sequence of operations in

$S = \{a, b, c, d, e, f\}$ solving some problem (see Section 2). Then, we apply (3) to derive the corresponding knowledge space. This last can then be used for assessing and training a user in solving problems in the problem space of Figure 1. This can be done by applying the assessment procedures discussed in Section 4.

References

- [Albert & Lukas, 1998] Albert, D., and Lukas, J. (1998). *Knowledge Spaces: Theories, Empirical Research, Applications*. Mahwah, NJ: Lawrence Erlbaum Associates.
- [Doignon, 1994] Doignon, J.-P. (1994). Knowledge spaces and skill assignments. In G.H. Fischer and D. Laming (Eds.), *Contributions to Mathematical Psychology, Psychometrics, and Methodology*, (pp. 111–121). New York: Springer-Verlag.
- [Doignon & Falmagne, 1999] Doignon, J.-P., and Falmagne, J.-C. (1999). *Knowledge Spaces*. Berlin, Heidelberg: Springer-Verlag.
- [Dowling & Hockemeyer, 1999] Dowling, C.E., and Hockemeyer, C. (1999). Automata for the assessment of knowledge. *IEEE Transactions on Knowledge and Data Engineering*, to appear.
- [Falmagne & Doignon, 1988] Falmagne, J.-C., and Doignon, J.-P. (1988). A Markovian procedure for assessing the state of a system. *Journal of Mathematical Psychology*, 32, 232–258.
- [Lee & Anderson, 2001] Lee, F.J., and Anderson, J.R. (2001). Does learning a complex task have to be complex? A study in learning decomposition. *Cognitive Psychology*, 42, 267–316.
- [Simon & Reed, 1976] Simon, H.A., and Reed, S.K. (1976). Modelling strategy shifts in a problem-solving task. *Cognitive Psychology*, 8, 86–97.