Zaluski, A. (2001). Knowledge Spaces Mathematica Package. PrimMath 2001 - Mathematica u znanosti, tehnologiji i obrazovanju [Mathematica in Science Technology, and Education] Conference, Zagreb, Sept. 27–28 (pp. 287-325).

Knowledge Spaces *Mathematica* Package

Andrej Zaluski [*]

Abstract

This paper aims to point at possibilities of applying Mathematica within the knowledge space theory and similar areas of cognitive psychology by describing a case in which software requirements of a research project are to be met by designing a Mathematica package - Knowledge Spaces Mathematica Package (KSMP). The initial and main goal of KSMP is to implement the concepts relevant for research on and development of querying on skills procedures. Nevertheless, the ultimate goal of KSMP is set to implement all essential knowledge space theory developments, and consequently not only to provide assistance in theoretical explorations, simulations and teaching, but also to serve as a basis for developing user-friendly Mathematica applications for a knowledge space theory applicant. The first part of the paper gives an overview and an introduction to the knowledge space theory. After the short overview of the research topics, the selected fundamental concepts currently covered by the package are presented. The notion of the prerequisite relationships among items of a knowledge domain is introduced, followed by the core theory concepts: knowledge spaces with their bases, surmise systems and entail relations. The topic of adaptive knowledge assessment is presented next, providing the distinction between the deterministic and the stochastic knowledge assessment procedures. After introducing the concept of an intelligent tutoring system, an adaptive tutoring strategy and some knowledge space theory applications in learning technologies are presented. The topic of establishing the structures, with an emphasis on querying experts procedures, is given, followed by the topic of empirical validation of the established structures. Finally, applications of the theory in cognitive psychology are presented with the emphasis on the competence-performance approach. The second part of the paper is devoted to the package itself. It begins with introducing the Querying Experts on Skills research project (QEOS) that initially motivated the development of KSMP. After presenting the QEOS project

---

[*] The work on Knowledge Spaces *Mathematica* Package was supported by an Ernst-Mach fellowship (BMBWK G-1621/Querying Experts on Skills).

The author thanks Dietrich Albert, Cord Hockemeyer and Christof Kŏrner for support during the research.

software requirements and describing Mathematica as a platform suitable for research and teaching within the field of knowledge spaces, the main and ultimate KSMP goals are given. The package design is presented next, including the package design requirements fulfilling the KSMP goals, as well as the package architecture reflecting the design requirements: two bottom core layers, as well as current individual modules of the upper layers. Additionally, few remarks on the package public functions are given, following by an introduction to the supported data file formats. The next section is related to the interaction with the package. After remarks on the package software and hardware requirements, the forms of the package documentation and information on loading the package are shortly presented. The section ends with an illustration of the package performance. The following section presents three overlapping categories of the further package developments: development of the core, development of the modules, and development of the package applications. The paper ends with concluding remarks that addresses the current contribution of KSMP to the QEOS project, the possible impact of KSMP on using Mathematica in similar areas, as well as some factors determining the ultimate reception of KSMP-based solutions by the knowledge space theory users.

The First Part — An Introduction and Overview of the Knowledge Space Theory

Research topics overview

Since the first published paper on the knowledge space theory (Doignon and Falmagne, 1985), the knowledge spaces have evolved from a knowledge representation language suitable for (computerized) adaptive assessment and adaptive tutoring to an approach within mathematical psychology that includes the whole spectrum of various theoretical research, empirical investigations and applications. The knowledge spaces approach addresses different issues from psychological, educational and cognitive sciences covering aims from building the effective computerized adaptive tutoring systems to the empirical validation of cognitive theories. It is a field with a plenty of ongoing research, and the classical overview (Falmagne et al, 1990) certainly does not give a complete picture of the field achievements today (see Albert and Lukas, 1999; Doignon and Falmagne, 1999). In the time of writing this paper, it seems legitimate to say that the knowledge space theory with its theoretical results, empirical investigations and applications has been giving contributions to the development of learning technologies, cognitive psychology and applied discrete mathematics. In its beginnings, the knowledge space theory was developed with an intention to give a contribution to the computer-assisted instruction systems. Accordingly, a large portion of the research was oriented towards developing adaptive assessment and training methods, as well as towards developing methods for establishing and empirically validating the built knowledge spaces. The relationships between these research topics are depicted in the following figure:

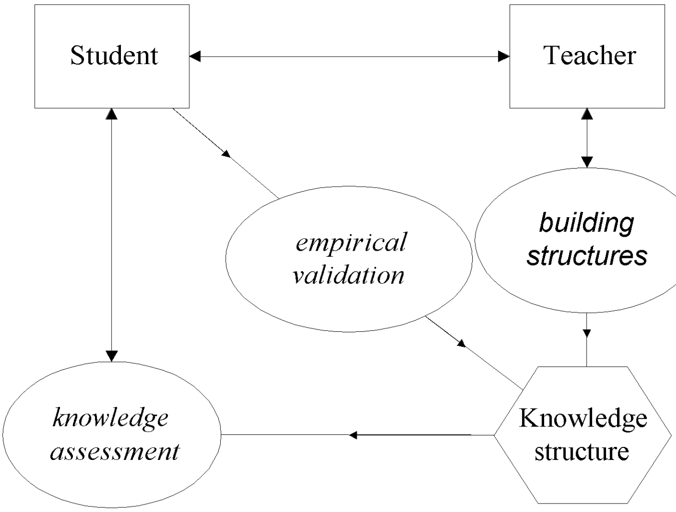

Figure 1. Knowledge space theory research topics

Later, the theory conceived to be applied within the educational context, having a favor towards abstraction with the initial intention to decrease vagueness of concepts (such as for example 'learning path' or 'knowledge state') was equipped in other research with a substantial mostly cognitive content (see Albert and Lukas, 1999). For instance, within the framework of systematic problem construction and information processing the knowledge space theory has been extended with the intention to provide means of validating cognitive theories (Albert and Held 1994, 1999; Albert, Schrepp and Held, 1994).

Knowledge space theory

Knowledge space theory is a formal model of knowledge representation based upon the lattice theory. Three formally equivalent concepts of knowledge spaces, surmise functions and entailments form the core of the theory. For applications in building intelligent tutoring systems, an important property of a knowledge space is well-gradeness. Additionally, the concept of the base of a knowledge space has an important role in the theory applications due to memory limitations of personal computers.

Prerequisite relationships among the items

In the knowledge space theory, the term item denotes any problem or task formulated in a way that enables classifying an examinee's responses either as correct or incorrect. It is assumed here that such responses provide sufficient information to answer whether or not the examinee masters the (tutorial) topics the item reflects. An item is conceived as an empirically testable representation of a knowledge unit, rather than a particular instance of the problem or task in question. Usually, there is a collection of instances assigned to each item. These collections are stored inside a pool of items, which for instance may be a part of an intelligent tutoring system. The theory attempts to represent a particular knowledge domain using a finite set of items. The main goal of the theory is to describe dependencies among solvability of items that arise in a particular knowledge domain for a certain population of examinees. This goal was motivated by the observations that an item often has some other items as its prerequisites, as well as that an item usually may have more differing collections of items as its prerequisites. More precisely, the theory states that in ideal conditions an examinee solves at least one collection of prerequisite items whenever he or she solves the item in question. Here, the ideal conditions denote both the absence of *lucky guesses* and *careless errors*, as well as the solving conditions in which the examinee is not under the time pressure or under an emotional stress.

Knowledge states, knowledge structures and knowledge spaces

Given a knowledge domain, a collection of all the items an examinee is capable of solving in the ideal conditions is called a knowledge state. For a population of examinees, the collection of all the knowledge states captures the organization of knowledge in that domain, and it is called a knowledge structure.

> A pair $(Q, \mathcal{K})$ in which Q is a nonempty set, and $\mathcal{K}$ is a family of subsets of Q containing at least Q and the empty set $\emptyset$, is called a ***knowledge structure***. The set Q is called the ***domain*** of the

> knowledge structure, its elements are referred to as ***items***, and the subsets in the family $\mathcal{K}$ are named ***knowledge states***. Occasionally, besides saying $(Q, \mathcal{K})$ is a knowledge structure, it is also said that $\mathcal{K}$ is a knowledge structure on the set Q. A knowledge structure $(Q, \mathcal{K})$ is called finite when the set Q is finite.

An important class of knowledge structures are knowledge spaces

> A knowledge structure $(Q, \mathcal{K})$ is called a ***knowledge space*** if the family $\mathcal{K}$ is closed under union, that is $\cup\mathcal{F} \in \mathcal{K}$ whenever $\mathcal{F} \subseteq \mathcal{K}$.

Due to careless errors and lucky guesses, a knowledge state is not conceived as an observable entity, what is observable are only the response patterns. In general, different knowledge states occur with different frequencies in a given population of examinees. Therefore, it is more realistic to exchange the deterministic knowledge structures for a probabilistic concept of knowledge structures

> A triple $(Q, \mathcal{K}, p)$ is called a (finite) ***probabilistic knowledge structure*** if $(Q, \mathcal{K})$ is a finite knowledge structure, and the mapping $p\colon \mathcal{K} \to [0, 1]$, $K \in \mathcal{K} \mapsto p(K)$, is a probability function. A function $r\colon 2^Q \times \mathcal{K} \to [0, 1]$, $(R, K) \mapsto r(R, K)$, which specifies the probability of observing the response pattern R for a subject in the knowledge state K, is called a ***response function***. A quadruple $(Q, \mathcal{K}, p, r)$ is called a ***basic probabilistic model***.

The basic probabilistic model is fundamental for the class of stochastic knowledge assessment procedures. Sometimes, it is assumed that the response function r additionally satisfies the *local independence* condition, i.e. for each item $q \in Q$ the careless error probability and the guessing probability are represented by the two constants $\beta_q, \eta_q \in [0, 1]$ (respectively), such that

$$r(R, K) = \Big(\prod_{q \in K \setminus R} \beta_q \Big) \Big(\prod_{q \in K \cap R} (1 - \beta_q) \Big) \Big(\prod_{q \in R \setminus K} \eta_q \Big) \Big(\prod_{q \in Q \setminus (R \cup K)} (1 - \eta_q) \Big) .$$

A basic probabilistic model having a response function satisfying the local independence condition is called a basic *local independence model.*

Bases, surmise systems and entail relations

Every finite knowledge space can be represented in the computer memory using only its base. Having in mind the size of knowledge spaces relevant for practical applications (whose domains may reach hundreds of items), algorithms that are not based on the concept of base can hardly be performed on personal computers.

> Let $\mathcal{F}$ be a family of sets. The family $\mathcal{F}'$ of all sets that are unions of some members of $\mathcal{F}$ is called the ***span*** of the family $\mathcal{F}$, or in other words $\mathcal{F}$ spans $\mathcal{F}'$. With respect to the set inclusion, a minimal family $\mathcal{B}$ of states spanning $\mathcal{K}$ is called a ***base*** of the knowledge structure $(Q, \mathcal{K})$.

By a convention, the empty set is the union of the empty subfamily of $\mathcal{B}$, hence the empty set does not belong to a base. A knowledge structure has a base only if it is a knowledge space.

> Let $\mathcal{B}$ be a base of a knowledge space $(Q, \mathcal{K})$. Then, $\mathcal{B} \subseteq \mathcal{F}$ for any subfamily $\mathcal{F}$ spanning $\mathcal{K}$. If the knowledge space $(Q, \mathcal{K})$ is finite, then it admits exactly one base.

An another fundamental concept is the concept of an atom

> For any item q, a minimal knowledge state containing q is called an ***atom at*** q. A state K is called an ***atom*** if it is an atom at some item q. A knowledge structure $(Q, \mathcal{K})$ is called ***granular*** if for each $K \in \mathcal{K}$, $q \in K$ there exist an atom A at q such that $q \in A \subseteq K$. Every finite knowledge structure is granular.

The concept of atom is useful for specifying a base when it exists

> If a knowledge space has a base, then the base is equal to the collection of all the atoms.

A concept equivalent to the concept of a knowledge space is the concept of a surmise system. It is probably the most obvious from this concept that the theory in general allows more than one learning path towards an item

> Let Q be a nonempty set of items, and let σ be a function from Q into 2^{2^Q}. Each function σ for which it holds that $\sigma(q) \neq \emptyset$ for every $q \in Q$ is called an ***attribution*** (function) on the set Q. For each $q \in Q$, any $C \in \sigma(q)$ is called a ***clause*** of q (in σ), or a ***background*** of q (in σ). Every attribution σ for which the following three conditions hold: for all $q, q' \in Q$ and $C, C' \subseteq Q$, (i) if $C \in \sigma(q)$, then $q \in C$; (ii) if

> $q' \in C \in \sigma(q)$, then $C' \subseteq C$ for some $C' \in \sigma(q')$; (iii) if $C, C' \in \sigma(q)$ and $C' \subseteq C$, then $C = C'$, is called a ***surmise function*** on Q, whereas the pair (Q, σ) is called a ***surmise system***.

Remaining fundamental concepts equivalent to the knowledge spaces are the concepts of entailments and entail relations. These concepts may be the basis for capturing an expert's answers in the process of building a knowledge space using querying procedures.

> Let $(Q, \mathcal{K})$ be a knowledge structure, and $\mathcal{P}$ a relation from 2^Q to Q defined by $A\ \mathcal{P}\ q :\Leftrightarrow (\forall K \in \mathcal{K} \mid A \cap K = \emptyset \Rightarrow q \notin K)$. It holds: (i) if $q \in A \subseteq Q$, then $A\ \mathcal{P}\ q$; (ii) $A, B \in 2^Q \setminus \{\emptyset\}$, $p \in Q$, $(\forall b \in B, A\ \mathcal{P}\ b) \wedge B\ \mathcal{P}\ p \Rightarrow A\ \mathcal{P}\ p$. A relation from $2^Q \setminus \{\emptyset\}$ to the nonempty domain Q that satisfies the conditions (i) and (ii) is called an ***entailment***.
>
> Let $\mathcal{P}$ denote a relation from $2^Q \setminus \{\emptyset\}$ to Q. Let $\mathcal{Q}$ be the relation on $2^Q \setminus \{\emptyset\}$ associated to $\mathcal{P}$ using the equivalence $A\ \mathcal{Q}\ B \Leftrightarrow (\forall b \in B \mid A\ \mathcal{P}\ b)$. This equivalence establishes a one-to-one correspondence between the family of all entailments $\mathcal{P}$ and the family of all relations $\mathcal{Q}$ that fulfill:
>
> $$\begin{aligned} &(i)\ \forall A, B \in 2^Q \setminus \{\emptyset\}, \quad B \subseteq A \Rightarrow A\ \mathcal{Q}\ B;\\ &(ii)\ \forall A, B, C \in 2^Q \setminus \{\emptyset\}, \quad A\ \mathcal{Q}\ B \wedge B\ \mathcal{Q}\ C \Rightarrow A\ \mathcal{Q}\ C;\\ &(iii)\ A, B_i \in 2^Q \setminus \{\emptyset\},\ (i \in I), \quad (\forall i \in I, A\ \mathcal{Q}\ B_i) \Rightarrow A\ \mathcal{Q} \bigcup_{i \in I} B_i; \end{aligned}$$
>
> A relation on $2^Q \setminus \{\emptyset\}$ satisfying the conditions (i), (ii) and (iii) is called an ***entail relation***.

The knowledge spaces, surmise systems and entailments (entail relations) represent equivalent concepts

> Given the same domain Q, there is a one-to-one correspondence between the family of all knowledge spaces $\mathcal{K}$ and the family of all entailments $\mathcal{P}$. Both the entailment $\mathcal{P}$ and the knowledge space $\mathcal{K}$ can be derived from each other using the following equivalences: $A\ \mathcal{P}\ q \Leftrightarrow (\forall K \in \mathcal{K}, A \cap K = \emptyset \Rightarrow q \notin K)$ and $K \in \mathcal{K} \Leftrightarrow (\forall (A, p) \in \mathcal{P}, A \cap K = \emptyset \Rightarrow p \notin K)$.
>
> Given the set Q of items, there exists a one-to-one correspondence between the collection of all granular knowledge spaces $(Q, \mathcal{K})$ and

the collection of all surmise functions σ. This correspondence is given by the equivalence between being an atom at q in $\mathcal{K}$ and being a clause of q in σ.

Although these core concepts are formally equivalent, they stress different aspects of the notion of prerequisite relationships among items, and consequently have different roles in the theory applications.

Knowledge assessment

Knowledge assessment procedures were developed within the educational context, but their axioms and results can be considered from a position of the general theory of systems. Here, a class of systems whose states are represented by subsets of a basic set of features is considered. A given constraint is that in each step only presence of one feature can be tested. The goal is to find a sequence of tests using which the state of a system can be discovered in a minimal number of steps. In the adaptive knowledge assessment, a set of items stands for the basic set of features and an examinee stands for the system. The term *adaptive* refers to the property of testing procedures that the selection of the next item depends on the previous examinee's answers. The utilization of prerequisite relationships existing in a knowledge domain described via a knowledge structure, enables designing the adaptive knowledge assessment procedures. These procedures are said to be *efficient* in the sense that often the number of items presented to an examinee while discovering his or her knowledge state is less than the number of items in the set representing the knowledge domain.

The determination of the knowledge space as a metric space, together with the concept of the neighborhood of a family of states serves as a basis for the deterministic adaptive knowledge assessment procedures. However, this deterministic context cannot be considered as realistic for many applications, in which possibilities of making a careless error or a lucky guess, as well as stochastic oscillations of the examinee's knowledge state during the assessment, have to be allowed. The introduction of the basic probabilistic model into a stochastic Markovian processes' framework provides a suitable basis for developing the stochastic adaptive knowledge assessment procedures (Doignon, 1994b; Doignon and Falmagne, 1987, 1999; Falmagne and Doignon, 1988a,b). After collecting an examinee's answer a computerized procedure either discards certain knowledge states as implausible for the examinee (the Markov chain knowledge assessment procedure) or updates the likelihoods that the examinee resides in a certain knowledge state (the class of continuous Markovian knowledge assessment procedures). Taking this change into account, the procedure selects the most diagnostically informative item for the next question.

One knowledge assessment procedure belonging to the deterministic context is the split-half procedure. The split-half procedure uses the split-half heuristic for selecting the next item, despite the fact that it does not guarantee the minimal number of items presented. At each step, the (*knowledge states*) *marker* collects all the knowledge states that are plausible for the examinee under assessment. A rule using which some item is denoted as the most diagnostically informative, and consequently selected as the next item for presentation, is named the *selection rule*. Given the marker at the step k and a collected examinee's answer, the *updating rule* determines the marker for the step $k+1$. The split-half selection rule tries to balance as much as possible the number of states that would remain in the marker after an incorrect response from an examinee and the number of states that would remain after a correct response.

> Let $(Q, \mathcal{K})$ be a knowledge structure, ${}_k\psi$ the marker in a step k, and ${}_kQ \subseteq Q$ a set of items left to be asked. A collection of all the knowledge states from ${}_k\psi$ that contain an item $i \in {}_kQ$ is denoted by ${}_k\psi_i$, whereas ${}_k\psi_{\bar{\imath}} := {}_k\psi \setminus {}_k\psi_i$ collects all the knowledge states that do not contain the item i. The set I_ψ of all items that the best balance the markers ${}_k\psi_i$ and ${}_k\psi_{\bar{\imath}}$ is given by $I_\psi := \{\, i_\psi \in {}_kQ \mid \big|\, |{}_k\psi_i| - |{}_k\psi_{\bar{\imath}}| \,\big| = \min \}$. After assigning to each item $i_\psi \in I_\psi$ a probability $p(i_\psi) = 1/|I_\psi|$, the split-half selection rule randomly selects some $q := i_\psi$ as the next item to be asked. Accordingly, the updating rule sets the marker ${}_{k+1}\psi$ for the next step $k+1$ either to ${}_k\psi_q$ or ${}_k\psi_{\bar{q}}$ depending on the correctness of an examinee's answer in the step k.

The described adaptive knowledge assessment procedure is called the split-half procedure. Describing the stochastic knowledge assessment procedures is beyond the scope of this introduction.

Applications in education

Intelligent tutoring systems

Intelligent tutoring system may be defined (IEEE LTSC, 2001) as a learning technology system that dynamically adapts a learning content to a learner's specific objectives, needs and preferences using its expertise in instructional methods and topics to be taught. Intelligent tutoring systems represent a class of computer-managed instruction systems that has the ultimate goal to reproduce key features of a human teacher's behavior. Here, the *computer-managed instruction* refers to the general use of computers to register learners, schedule learning resources, control and guide the learning process, as well as to analyze and to report learners' performance (IEEE LTSC, 2001). An intelligent tutoring system besides possessing tutoring strategies and knowledge about a domain, has to be capable to model a learner's state of knowledge. In comparison to *computer-aided learning*, the instructional process is not necessarily driven by a human since the system is also equipped with instructional methods and strategies. Such learning technology is also referred to as *computer-based instruction.* There exists a plenty of ITS architectures in the various domains of knowledge. Most of these architectures are specific for the domain for which they are constructed, and consequently not transferrable to the others. However, in general there is an agreement that each ITS should contain at least four components that interact during the instruction process: the knowledge base, the student model, the diagnostic component, and the teaching component (Albert and Schrepp, 1999). While the knowledge base usually captures experts' knowledge in the form of an expert system, there are differences in conceptualizing the student model.

The knowledge space theory can serve as a basis for developing intelligent tutoring systems since it provides solutions for all the components: the knowledge base in the form of a knowledge structure empirically captured for a population of students, the student model consisting of knowledge states and the basic probabilistic model for modelling learner's responses, the diagnostic component in the form of adaptive knowledge assessment procedures, and the teaching component in the form of tutoring strategies that are adaptive with respect to the prior knowledge of a learner. Albert and Schrepp (1999) present an usage of the knowledge space theory and skill assignments in developing general, knowledge domain independent ITS architecture.

Adaptive tutoring

A tutoring strategy that consists of a selective (computerized) presentation of learning material and exercises, based on the diagnosed knowledge of a student, is called an *adaptive tutoring.* The output of the knowledge assessment procedures is an estimate of the examinee's knowledge. This estimate may be given as a single knowledge state, as a probability distribution on a knowledge structure, or as a collection of knowledge states sometimes additionally accompanied with a probability distribution. In any of these cases, within the educational context such estimates have to be considered as more informative than single-number estimates (e.g. 75 % answers correct) because they provide an insight into which topics of the knowledge domain are already mastered by a student and which have still to be taught. Therefore, these kinds of knowledge estimates are suitable for (adaptive) tutoring. Adaptive tutoring methods can be based on the concepts of a knowledge state's neighborhood and the concept of the fringe of a knowledge state. Suppose a student in a knowledge state K and the domain knowledge represented by a well-graded knowledge space. The collection of all the knowledge states whose distance from the state K is at the most one is called the neighborhood $N(K)$ of the knowledge state K. The set of all the items by which the state K differs from the states belonging to $N(K)$ is called the fringe $K^{\mathcal{F}}$ of the knowledge state K. The items belonging to a fringe are either the items that were last mastered by a student in his or her specific learning path towards the item, or the items that are usually mastered immediately after the item in question by students from the same population. The knowledge corresponding to the first group of items may still be fragile since it was mastered the last. The latter group represents recommended continuations of the student's individual learning path. Therefore, there are reasons for teaching tutorial topics that correspond to both groups of items that form the fringe. Additionally, the items (and corresponding tutorial topics) outside the fringe are considered to be either too easy or too hard for a student possessing the knowledge state in question. The first may cause boredom, the second frustration. Thus, there are reasons against teaching the topics corresponding to the items outside the fringe. Accordingly, one tutoring strategy consists of suggesting for teaching only the topics corresponding to the items belonging to the fringe of the knowledge state that was assigned to a student by the knowledge assessment. For example, suppose a student in the knowledge state $\{b, d\}$ given the knowledge space depicted in the figure 2. Then, the neighborhood $N(\{b, d\})$ is equal to $\{\,\{d\}, \{a, b, d\}, \{b, d, e\}\,\}$, whereas the fringe $\{b, d\}^{\mathcal{F}}$ is equal to $\{a, b, e\}$. The tutoring strategy suggests the topics that correspond to the items a, b, e for teaching.

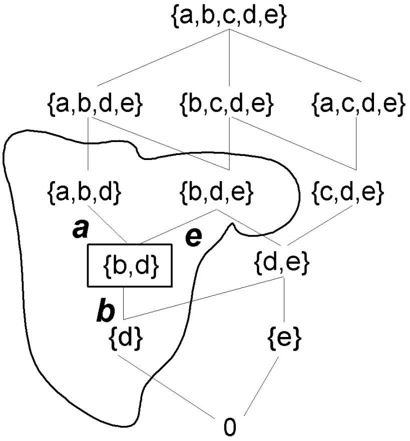

Figure 2. A tutoring strategy based on the fringe of a knowledge state

Learning technology systems developed using the knowledge space theory

There exists a few intelligent tutoring systems built on the knowledge space theory or its extensions. A comprehensive application of the knowledge space theory in the distance learning is *ALEKS* (an acronym for 'Assessment and Learning in Knowledge spaces'), available at the URL http://www.aleks.com. It represents a multilingual educational environment covering topics such as basic arithmetic, algebra and geometry (see Doignon and Falmagne, 1999).

Another interesting application is *RATH* — a relational adaptive tutoring hypertext WWW-environment based on the knowledge space theory (Hockemeyer, 1997; Hockemeyer, Held and Albert, 1998; Hockemeyer and Albert, 1999). *RATH* combines a mathematical model for the structure of hypertext with the knowledge space theory. The teaching material is given to the system as a tutoring hypertext, i.e. hypertext covering knowledge domain lessons and exercises. The single components of the tutoring hypertext and the link structure are registered in a relational database. Utilizing the prerequisite relationships in a knowledge domain, after diagnosing the knowledge state of a student, the system presents to the student only hyperlinks to those lessons for which he or she fulfills all the prerequisites. *RATH* is available at the URL http://wundt.uni-graz.at/rath.

Establishing structures

In the knowledge space theory applications, a critical phase lies in the process of establishing the structures, that is in finding out feasible states from all the possibilities. The so-called empirical approach in building the structures is based on conducting the knowledge assessment on large samples of examinees. After the assessment, noise in the data is eliminated using statistical methods. However, according to Doignon and Falmagne (1999, pp. 105), due to technical reasons, the available statistical methods can be applied only to the sets of items which size is much smaller than the sizes that would be desirable to have in the applications. The next approach in establishing the structures is based on a systematic examination of the content of problems (Albert and Held, 1994). The third approach in building the knowledge structures is based upon consultations with experts from a particular knowledge domain, and a utilization of their domain meta-knowledge. Additionally, it is not necessarily to build a large knowledge structure at once. In certain situations, several domains with accompanied structures may be treated as subdomains of a full domain and combined to construct a structure on the full domain. (Doignon and Falmagne, 1999).

Querying experts

There exists several techniques for establishing knowledge structures by eliciting and acquiring experts' domain meta-knowledge. Generally, an expert is not requested to enumerate the feasible states. His or her awareness of feasible states is assumed to be implicit, and it can be evoked by questions presented by computerized procedures. These procedures also take care of some additional parameters such as consistency and completeness of the expert's answers. The approach of establishing knowledge structures based on experts' judgments on prerequisite relationships among items within a knowledge domain, which are obtained through a computerized procedure, is called *querying experts* (Müller 1989, 1991; Dowling, 1991, 1993a,b; Koppen 1993, 1994; Koppen and Doignon, 1990).

The procedures for querying experts were developed by Koppen and Doignon (1990), Koppen (1993,1994) and by Dowling (1993a,b). Enhancements to these procedures were introduced later by different authors, for example by Dowling and Kaluscha (1995); Dowling, Koch and Quante (1996); and Freitag (1999). A practical querying procedure was suggested by Cosyn and Thiéry (2000). These enhancements concerned different aspects of the procedures, among them also the selection strategies of the next item and the development of user's interfaces to the expert. It has been known from the empirical investigations that only in case of small sets of items it should be expected that experts are capable of directly drawing and/or graphs, or directly stating the prerequisite relationships through surmise systems. In case of the larger sets, an expert's judgments on prerequisite relationships become both incomplete and inconsistent. In order to obtain both complete and consistent judgments of experts, computer software has been developed to assist experts during the process of establishing knowledge structures. The computerized procedures present to an expert to judge statements of the following form:

> "Suppose that a student under examination has just failed all the problems $a_1, a_2, \ldots, a_n$. Is it then (practically) certain that this student will also fail the problem q? You may assume that chance factors, such as careless errors and lucky guesses play no role in the student's performance."

A statement of this form is called a standard *form assertion.* Taking into account the size of interesting sets of items and the number of all possible prerequisite relationships among the items, it becomes obvious that presenting all the possible statements to an expert is not feasible in many situations. Thus, a query procedure should not present the statements that are logical consequences of the statements previously either accepted or rejected by an expert. This property of a query procedure is called *the non-redundancy property.* A standard form assertion that is accepted by an expert or whose acceptance logically follows from the set of previously judged statements is called a *positive inference*, whereas a statement that is rejected by an expert or whose rejection logically follows from the statements previously judged is called a *negative inference.* A standard form assertion that is either positive or negative inference is called an *inference.* The introduced concept of entailments can be used for representing the inferences obtained by querying an expert.

Koppen and Doignon (Koppen, 1993,1994; Koppen and Doignon, 1990) developed a stepwise, block-by-block querying algorithm that assumes limited memory conditions. It divides the set of all possible standard form assertions into the blocks containing all the assertions with a same premise size. A querying process runs block by block. At each query, exactly one standard form assertion is presented to the expert. The expert either accepts or rejects the presented assertion. The algorithm calculates all the inferences and fills in the current block. The process of selecting the next standard form assertion is governed by the principle that the selection should maximize the expected number of inferences in order to minimize the overall number of presented assertions. After a block of assertions is completely filled, the algorithm moves to the next block. The algorithm finishes the querying when all the assertions are inferences. This algorithm presents an assertion with the premise size $k+1$ only after all the assertions with the premise size k are inferences. Dowling (1993) developed a querying algorithm based on the base of a knowledge space. One property of this algorithm is that it is not necessary to present a premise of the size $k+1$ only after all the assertions with the premise size k are inferences. This advantage allowed Kaluscha (Dowling and Kaluscha, 1995) to introduce various heuristics for selecting the next assertion. Both of the algorithms have the properties of non-redundancy and completeness.

Let Q be a knowledge domain whose structure is to be established by querying an expert. Let $\mathcal{P}$ be an entailment that stores the positive inferences during a querying session, and let $\mathcal{N}$ be its complementary relation that stores the negative inferences. At each step, it holds $\mathcal{P} \cup \mathcal{N} = 2^Q \setminus \{\emptyset\} \times Q$. A simple querying procedure puts after each query the corresponding standard form assertions $(A, q) \in 2^Q \setminus \{\emptyset\} \times Q$ either in $\mathcal{P}$ or $\mathcal{N}$. In calculating the inferences, the procedure may, for instance, apply the following rules as reasonable $\big(A, B \in 2^Q \setminus \{\emptyset\},\ q, q' \in Q, \quad B\,\mathcal{P}\,A \Leftrightarrow (\forall a \in A \mid B\,\mathcal{P}\,a)\big)$:

$$A\,\mathcal{P}\,q \wedge B\,\mathcal{P}\,A \wedge (A \cup \{q\})\,\mathcal{P}\,q' \Rightarrow B\,\mathcal{P}\,q' \tag{a}$$

$$A\,\mathcal{P}\,q \wedge A\,\mathcal{N}\,q' \wedge (A \cup \{q\})\,\mathcal{P}\,B \Rightarrow B\,\mathcal{N}\,q' \tag{b}$$

$$A\,\mathcal{P}\,q \wedge B\,\mathcal{N}\,q \wedge (B \cup \{q'\})\,\mathcal{P}\,A \Rightarrow B\,\mathcal{N}\,q' \tag{c}$$

$$A\,\mathcal{N}\,q \wedge A\,\mathcal{P}\,B \wedge (B \cup \{q'\})\,\mathcal{P}\,q \Rightarrow B\,\mathcal{N}\,q' \tag{d}$$

Consequently, after each step there is at least one open standard form assertion less. Unfortunately, these rules do not guarantee the absence of contradictions among the inferences. Namely, the rules (b), (c) and (d) do not find all the negative inferences among the open standard form assertions. In the block-by-block algorithm, all the missing negative inferences can be collected by going through the current block and drawing inferences after assuming for each open standard form assertion that it is a positive inference. If a contradiction is found, this in fact means that this standard form assertion is not an open assertion, but a missed negative inference.

Empirical validation of the established structures

The validity of established knowledge spaces is determined by examining a match between the built spaces and the collected response patterns. The cardinality of a minimal symmetric set difference between an individual response pattern and the knowledge states can be taken as a measure of the individual match. This measure serves as a basis for determining the distance between a knowledge space and the response patterns from a sample of subjects. The various validity coefficients exist, e.g. the distance mean coefficient. An established space is considered to be more valid if it is closer to the collected response patterns. Querying more than one expert given the same set of items, and an integration of the established knowledge spaces into resulting one is usually performed in order to obtain more valid knowledge spaces. At this point, sometimes the trade-off between the validity and the efficiency of knowledge assessment procedures has to be respected. Namely, the validity determined by a validity coefficient is greater in cases when the knowledge spaces contain more knowledge states. On the other hand, the knowledge assessment is more efficient in cases when knowledge spaces contain fewer states. If the purpose of building a knowledge space is its embedment in an intelligent tutoring system, the aim is to establish sufficiently valid knowledge spaces that still support an effective knowledge assessment.

Applications in cognitive psychology

The knowledge space theory approach introduced by Falmagne and Doignon is behavioral one. The primary concern of the authors was building a formal apparatus that would decrease vagueness of the notions used in the educational field. The authors has not ascribed neither cognitive nor psychological interpretations to the concepts. The other researchers equipped the apparatus with a substantial content by introducing concepts that have psychological or cognitive interpretations (see Albert and Lukas, 1999).

Competence-performance approach

In the competence-performance approach (Korossy 1993, 1996a,b, 1997a,b, 1999a,b), domains of knowledge are conceptualized in terms of competencies (skills, abilities, knowledge) and performances. The performance refers to the observable behavior, i.e. to the solution behavior on the sets of problems, whereas the competencies stand for theoretical entities with the purposes of ‘explanation' and prediction of the performance. More precisely, a knowledge domain is represented by a finite, non-empty, structured family of competence states called a competence structure. Competence states, i.e. representations of knowledge or capabilities of individual persons, can be specified as particular subsets of a domain-specific set of elementary competencies. In addition, the same knowledge domain can also be modeled at the observable level, i.e. by the sets of problems satisfying several constraints. Firstly, each problem from the set is solvable exclusively by the knowledge modeled using the competence structure. Additionally, for each pair consisting of a problem and of a competence state, it is possible to determine whether or not the problem can be solved in the competence state, allowing the possibility that the same problem can be solved in the several competence states. Consequently, each person due to his or her competence state is capable of solving only certain problems. A collection of problems one is capable of solving is called a *solution pattern*.

Having an established competence structure and the solvability of a set of problems interpreted in terms of the competence structure, a collection of theoretically expected solution patterns can be derived. A theoretically expected solution pattern is called a performance state. In this sense, a match between performance states and empirically observed solution patterns indicates the validity of the competence structure, providing a way of the empirical validation of a domain-specific cognitive theory. The performance state of a person, obtained on a valid competence structure, allows drawing conclusions on the possible competence states of the person, which in turn makes the competence-performance approach also suitable in the development of intelligent tutoring systems. Using the introduced concepts, it is possible to reinterpret the fundamental concepts of the knowledge space theory. The knowledge structure, a collection of subsets of problems, has been substituted by two related concepts: the competence structure and the performance structure. The *competence structure* refers to a collection of subsets of elementary competencies, such that each elementary competency is contained in at least one subset. The members of a competence structure are called *competence states*. A competence structure that contains both empty set and the set of all elementary competencies, and is also stable under union, is called a *competence space*. A non-empty family of subsets of a finite set of problems, such that each problem is contained in at least one subset, is called a *performance structure*. The members of a performance structure are called *performance states*. A *performance space* is a performance structure both stable under union and having the empty set and the set of all problems as its members. By putting the knowledge spaces in the competence-performance framework, a descriptive model of domain specific solution behavior has been substituted by the theory-oriented model providing means for accounting solution behavior on different sets of problems within a knowledge domain. To ensure usefulness of such an accounting of observed behavior by latent competencies, the number of competencies forming a competence structure should be strictly less than the number of problems forming the performance structure.

The concept of a diagnostic

The goal of the competence-performance approach is to establish meaningful explicit connections between the competence level and the performance level. This can be accomplished using the concept of a diagnostic. A problem from a given set can be interpreted in a competence structure if the competence structure includes all the competence states each of which is sufficient for solving the problem. In this case, the interpretation of a problem given a competence structure can be defined as a mapping from the set of problems to the powerset of the competence structure, which assigns to each problem a collection of competence states such that in each of them the problem is solvable, and the assigned collection differs both from the empty set and the whole competence structure. This mapping is called an *interpretation function*, the given set is called a set interpreted in a competence structure, and the subsets of the competence structure are called *interpretations* of problems in the given competence structures. A subset of a problem set is called a *representation* if there exists a competence state such that the subset includes exactly those problems that are solvable in the competence state. In that sense, a mapping that assigns to each competence state the unique (possibly empty) collection of all problems solvable in the competence state is called a *representation function.* A five-tuple consisting of a competence structure, a set of problems, an interpretation function, together with the representation function and the performance structure is called a *diagnostic.*

Cognitive task analysis

An establishment of the competence structure is usually based on the cognitive task analysis (Korossy, 1999a). The cognitive-task-analysis goal is to identify knowledge components using which it is possible to explain success or failure in solving a sample of typical knowledge domain problems. Firstly, a solution-ways analysis is conducted to collect all possible solution ways for each problem. Secondly, a competence-analysis is performed to establish a family of elementary competencies using which the solution ways can be explained. Finally, each solution way has to be completely expressed through elementary competencies.

The Second Part — Knowledge Spaces Mathematica Package

Motivation for developing the package

QEOS software requirements and *KSMP* goals

Although the establishment of a competence structure is usually based on the cognitive task analysis, it is possible to establish the competence structures by taking advantage of experts' meta-knowledge as in the case of querying on knowledge structures procedures. The research and development of querying procedures within the competence-performance approach is the aim of the *Querying Experts on Skills* (QEOS) research project, which in turn initiated the development of KSMP. Although most concepts implemented in the current version of the package are those closely related to QEOS, the principles guiding the package design as well as the whole package components exceed the initial requirements on the package derived from the project. One of the initial project tasks was to find or create a software environment that would facilitate research and development of such querying procedures. Such a software environment should have primarily been suitable for theoretical explorations and simulation studies. This software environment should also support development of usable applications that implement the developed querying procedures, at least to the extent that ensures an adequate empirical validation of the procedures. Additionally, it would be desirable if the dissemination requirements could be completely fulfilled, that is if the developed applications can be delivered to all applicants of knowledge space software tools. *Mathematica* as an integrated platform suitable for computing, programming and typesetting seemed as an appropriate candidate. Interactive capabilities and *Mathematica*'s potential in teaching were additional arguments facilitating the selection. The standard *Mathematica* packages were not found sufficient for carrying on the research, a need for writing specific program code appeared, and the issue of organizing the produced code has emerged.

Although the initial and main goal of KSMP is to implement the concepts relevant for research on and development of querying on skills procedures, its ultimate goal is to implement all essential knowledge space theory developments. Consequently, KSMP may provide assistance to a researcher in his or her theoretical explorations, simulations and teaching. Additionally, to a software developer KSMP can serve as a basis for developing various knowledge space theory applications, allowing her or him to focus efforts on the tasks such as building task-oriented front-ends to an applicant. This paper addresses parts of KSMP that together achieve some specific functionalities, which are also usually united by a specific interface to a user, as a package application. The QEOS application is conceived as the package application dealing especially with the querying on skills procedures. Unfortunately, only the applicants having access to *Mathematica* may take advantage of using the QEOS application. Nevertheless, KSMP alone without the QEOS application may also facilitate the research on the querying on skills procedures. Once the procedures with satisfying properties are developed, they can be implemented in one of the standard languages as C/C++ and the built tools disseminated also to the applicants that do not have access to *Mathematica*. KSMP is considered to be a work under development.

Mathematica as a platform suitable for research and teaching within the field of knowledge spaces

Providing integrated computing, typesetting and programming capabilities, *Mathematica* offers an attractive environment for research within the field of knowledge space theory. Extending these capabilities with a possibility of presenting both standard findings as well as new results in the form of interactive notebooks to students, makes *Mathematica* quite recommendable. In addition to these two benefits, several other characteristics I found useful. MathLink and the interprocess-communication mechanism enable integrating *Mathematica* with already developed Unix/C knowledge space tools, either by extending these tools using *Mathematica* functions or invoking them from the front-end in a transparent way. These tools are optimized for computations on structures having sizes that are found in relevant empirical applications, and the communication between KSMP and these tools might be beneficial to both. Additionally, possibilities of redirecting computation output to separate notebooks and automatic export to different file formats (both achievable by front-end programming) may further ease various routine tasks. Concerning the development of KSMP, the most influential part of the standard *Mathematica* distribution has been the Skienna's *DiscreteMath‘Combinatorica‘* package (Skienna, 1990).

Package design

Package design requirements

Three different classes of users are assumed to work with the package: applicants, researchers and software developers. The applicants would probably only be interested in using the package inside empirical investigations or for teaching. The researchers would additionally be interested to use it for theoretical explorations and simulation studies, to suit the package to their needs and probably to extend the package with their own concepts. On the other side, the developers would probably be engaged primarily in building the extensions and package applications.

The package design reflects the assumption of the three classes of potential users together with the fact that a whole variety of knowledge space theory concepts might be of interest and therefore implemented. Hence, a major requirement on the package design is to allow developments in several independent directions, potentially by different researchers and developers, as well as to provide means for integration and interoperability of the independent contributions. Another major design requirement refers to ensuring a delivery of only those parts of the package for which individual applicants are interested, i.e. leaving out thematically unnecessary parts, without affecting basic features such as working with data files or connected to the fundamental concepts. This requirement aims to ensure potential deliveries of individual package applications regardless of KSMP deliveries. Parts of the package have also been developed respecting this requirement. The naming conventions applied to the package functions favor the developers and researchers because the KSMP-applicants communication is to be achieved through the fronts-ends of the task-oriented package applications. I think that designing the package in accordance with these requirements results in a software product that can be easily maintained or extended, that might be interesting for most knowledge space theory researchers and applicants because it more or less equally covers all important knowledge space theory concepts, and finally that can be easily used by all the classes of potential users.

Architecture of the package

The package architecture is described through abstract layers. The layers refers to dependencies among the functions and introduction of new abstract data types. Ideally, communication should occur only between entities of the adjacent layers. Consequently, possible future redesigns of the lower layers would entail minimal redesigns of entities of the upper layers, and vice versa.

The KSMP architecture consists of four layers reflecting the package design requirements. The bottom two layers are considered to be the package core since they are to be present in all package applications or package editions. The core's purpose is to ensure accessibility of basic data structures and functions representing the basic knowledge space theory concepts to the upper layers, to ensure a flexible and uniform way of working with data files, and lastly to provide reporting facilities. The third layer and the forth layer consist of modules, i.e. entities largely independent one from each other bearing specific functionalities. The modules of the layer three are conceived to have functionalities corresponding to the different topics of the knowledge space theory. The layer four of the package is reserved for the control-parts and front-ends of various package applications. The package applications utilize different modules of the layer three. Throughout the paper, I sometimes identify the package applications with their fourth layer components and treat them as modules of the package layer four. The package applications represent the basis for composing different task-oriented package editions. The layers and modules consist of components that mostly correspond to different concepts and approaches in the knowledge space theory. This package version brings the core that besides the two layers (KSMP Layer one and KSMP Layer two) incorporates an additional module (KSMP Warp core) only partially wrapped up by the layer one. During the core development, the package main goal and the ultimate goal were equally respected. The layer three brings the following modules: KSMP Simulation module, KSMP Querying module and KSMP Assessment module.

KSMP Layer one

The KSMP Layer one consists of the following components: *Sets, Relations, Functions, Structures, DataFiles, Report* and *Miscellaneous* component. The aim of this layer is to be an interface between standard *Mathematica* data structures and functions at one side and functions implementing fundamental concepts and procedures of the knowledge space theory at the other. The *Datafiles* component is responsible for loading and saving data using different data file formats. The aim of the *Report* component is to provide facilities on producing uniform reports to the functions of the layer two and to the upper modules. These facilities aim to supplement reporting mechanisms of individual functions based on using options. The components *Sets*, *Relations* and *Functions* aim to represent selected concepts from the set theory, theory of relations and theory of functions in a way suitable for utilization by the layer two and the modules of the layer three. The component *Structures* represents the concept of a structure, as it is defined and used in the knowledge space theory. The component *Miscellaneous* is a repository of all public functions that have not found their place in the other components and that might be of interest mostly for developers.

KSMP Warp core

KSMP Warp core represents a recent addition to the layer one. Its relationships to entities of the other layers are currently not strictly defined. The KSMP Warp core aims to provide data structures and data files handling facilities to the package, which would enable usage of KSMP in relevant empirical knowledge space theory applications respecting the memory limitations and CPU strength of personal computers.

KSMP Layer two

The aims of the KSMP Layer two are to implement the fundamental knowledge space theory concepts and to ensure access to the pools of items and skills for the package modules. The layer two consists of the following components: *Items and skills, Theory of relations and order theory, Knowledge structures, Bases, Surmise systems, Entail relations* and *Competence-performance approach.* The layer two components are supposed to serve as an interface between the entities of the layer one and the modules of the layer three. The aim of the component *Items and skills* is to provide data structures and functions for describing items and skills that are to be used by modules while performing different procedures, for example querying an expert on structure of a set of items or assessing knowledge of a student given the structure. The aim of the components *Theory of relations and order theory, Knowledge structures, Bases, Surmise Systems* and *Entail relations* is to implement the corresponding knowledge space theory concepts in the package. The various knowledge space theory approaches for handling latent skills and competencies are to be covered by the components of this layer. This package version covers only the competence-performance approach by the *Competence-performance approach* component.

KSMP Simulation module

The KSMP Simulation module is a third layer module that aims to provide facilities for conducting simulations related to the knowledge space theory topics covered by the other modules. It consists of the two components: *Modeling expert* and *Modeling student*. The aims of these components are to provide facilities for describing the domain meta-knowledge of an expert and the domain knowledge of a student. These descriptions also provide place for uncertainty parameters representing careless errors and lucky guesses of a student or expert, or a subjective certainty in a given response or answer. An expert's meta-knowledge can be described either using a structure or a diagnostic, but also using alternative solutions such as lists of positive, negative and undecidable assertions including different uncertainty parameters.

KSMP Querying module

The KSMP Querying module aims to implement approaches for establishing various knowledge space theory constructs by querying experts. The module in this package version consists of the following components: *Koppen's algorithm, Querying experts on competence-performance diagnostic, Querying Session Sonde*, and *General querying experts* component. The *Koppen's algorithm* component implements procedures for establishing knowledge structures based on the concepts of entail relation and entailment, as well as the Koppen's block-by-block algorithm (Koppen, 1993). The *Querying Experts on a competence-performance diagnostic* currently implements only the straightforward procedure for querying experts on diagnostics. The most important parts of this component are the procedure for querying on a competence space based on Koppen's block-by-block algorithm and the straightforward procedure for querying on an interpretation function. The *Querying Session Sonde* introduces the concept of an observation sonde, whose purpose is to explore the behavior of querying procedures. The *General querying experts* component is conceived as a component that unites all implemented querying approaches and represents an interface between the module and the control parts of package applications residing at the fourth layer. The querying procedures may operate either on a human expert or on a simulated expert. Querying reports having different extent of verbosity may easily be produced due to the global verbosity variable and the verbosity option that majority of the central package functions have.

KSMP Assessment module

The KSMP Assessment module aims to implement approaches to the adaptive knowledge assessment developed within the knowledge space theory. In this package version, this module is included for demonstration purposes and it consists only of a component for *Deterministic knowledge assessment.* The assessment procedures are able to dynamically generate item instances from the item descriptions during the assessment.

KSMP public functions

This version of the package contains more than 300 public functions. Some of them are primitive functions introducing new abstract data types, some of them are responsible for performing complex procedures using a plenty of other functions. Nevertheless, the most of them are left as public. Namely, assuming the three different classes of potential users, the package brings weak restrictions concerning the visibility and scope of the package functions. Presumably, only a few of the public functions are of exclusive interest to the developer. Discussing each or even selected representatives of the package functions is not the intention and is beyond the scope of this paper. For a description of individual package functions, the reader is advised to consult the package documentation.

Supported data file types

In case of KSMP, the data file stores instances of different concepts defined by KSMP, such as for example: the structure, the items description, the skills description etc. KSMP supports three major data file formats *Spacefile, Itemsfile* and *Skillsfile*, all of them being the plain text type. Here, the term data file format refers to internal organization of data inside the data file, whereas the term file type refers to human readability of the files- plain text or binary. The term major data file formats refers to those data file formats that are intended to store instances of the major (both in the sense of fundamental and most used) knowledge space theory concepts. KST and SRBT tools (Unix/C knowledge space theory tools developed independently of KSMP by different authors) stipulate their own data file formats, all of them being the plain text type. KSMP inherits some of the KSBT and SRBT data file formats, for example the *Spacefile* data file format. The KSMP native data file formats are those stipulated directly by KSMP, that is not inherited from the other tools. The examples are the *Itemsfile* and the *Skillsfile* data file formats. The *Itemsfile* data file format stores descriptions of items for purposes of querying experts or knowledge assessment. The *Skillsfile* data file format stores descriptions of skills primarily for purposes of querying experts.

The *Itemsfile* data file format supports three types of items: fill-in items, multiple-choice items (without the fill-in answer alternative), and multiple-choice items with the fill-in answer alternative. Both multiple-choice item types support only static item records in the sense that no part of the item record is subject to change during evaluation. Consequently, an individual multiple-choice item data file record in fact stores an instance of an item. The fill-in item type additionally supports dynamic type records. The item description line of a fill-in item record may contain random macros and the correct answer item line may contain a pure function taking the random macros as parameters. Here, the term random macro refers to denoted parts of the item description record line that serve as parameters for the standard *Mathematica* Random function. The random macros can be evaluated while loading the items. Whenever the random macros are evaluated, the pure function providing the correct answer is also evaluated, and in turn both the item description record line and the correct answer item record line are changed accordingly. Consequently, an individual fill-in item record may store either a specific instance of an item or an item. In the latter case, individual instances are to be generated dynamically by the package functions.

As already mentioned, the *Datafile* component of the package layer one is responsible for appropriate handling of data files. Since KST and SRBT tools are developed in parallel and independently of KSMP, their native data file formats were and are subject to change, the *Datafile* component is designed to be also able to accommodate both existing versions of their native data file formats as well as future changes in the native data file formats of these tools. Here, the term versions of a data file format refers to differences in a same data file format resulting from development of the tools. The same holds for accommodating versions of the KSMP native data file formats. The decision to rely exclusively on the plain text type is made to ensure easy transfers of data between different hardware/operating systems configurations, to facilitate exchange of data with other software products (e.g. *STATISTICA* and SPSS), as well as to preserve compatibility with the KST and SRBT tools and consequently ease a joined usage of KSMP, KST and SRBT in research and teaching. Such a decision was made despite the inferior performance of certain package functions occurring due to more space consuming data file formats in comparison with possible binary type alternatives.

Interacting with the package

Software and hardware requirements

The package has been developed and tested on a Pentium 233 MHz computer equipped with 96 MB of RAM running Windows 2000 Professional operating system. *Mathematica* version 4.0 was used. No effort has been taken to check backwards compatibility with 3.x versions. In general, any hardware/operating system configuration with installed at least version 4.x *Mathematica* may be regarded as sufficient for running the package. Nevertheless, because the package performance is quite affected by the amount of available memory, practical applicability of the package for a specific research investigation will depend on the size of used structures.

Loading the package

The package can be loaded manually, either whole at once or individual modules separately. The first option will be probably used by the applicants in most occasions. The intention of the second is to facilitate independent package development in different thematic directions, as well as to increase the amount of free memory by excluding unnecessary parts from the individual package applications. The package provides the autoloading option that predeclares the package symbols and automatically loads the package whenever one of its functions is called for the first time. Autoloading the package is advised at least to the applicants, because it prevents possible improper working of the package that might occur if one of its functions had been called before the package was actually loaded.

Package documentation

Package documentation covers installation instructions, a technical description of the package, a glossary of all public functions, supported data file types and examples on using the package. The documentation takes two forms: external and internal. The external form refers both to the package manual provided as a PDF document and to help notebooks. The help notebooks can be navigated through the *Mathematica* Help Browser and are accessible under the Add-ons/Knowledge Spaces node. The internal form refers to the information accessible using the ? command. This documentation form not only speeds up the writing of function calls, but also eases a work with the package. Description and usage information on each public package function, as well as the options used by the functions, are available using this help mechanism. Pressing the Ctrl-Shift-K key combination after the (sometimes even partial) name of a package function results in obtaining the usage template for the function. For instance, the information on *QEOSQueryExpertOnCompetenceSpace* function may be obtained by

```
?QEOSQueryExpertOnCompetenceSpace
```

> `QEOSQueryExpertOnCompetenceSpace[E,SkOrExp,options]` queries a human (SkOrExp represents a description of skills) or a simulated expert (SkOrExp represents a simulated expert) on structure of the set `E` of elementary competencies and returns the established structure either as a space or assertions. A valid option is `QueryingAlgorithm` (Koppen [default]). For additional valid options see `QEKQueryExpertByKoppen`.

By default, a functional package installation has both the code files, the help notebooks and the manual under the same directory. In spirit of the literate programming, the intention is to bring both forms of the documentation as close as possible to the code in order to facilitate their maintenance.

An illustration of the package performance

The package contains a plenty of functions having different behavior in the terms of computation time, memory requirements and practical applicability. Skienna (1990) recommends to computationally determine a function's performance due to specifics of *Mathematica* computing mechanisms. Presenting an overview of the package function performances is beyond the scope of this paper. Nonetheless, an illustration related to one of the central package functions is provided in the sequel. The behavior of `QEOSQueryExpertOnCompetenceSpace` in addition to the cardinality of a set of items on which it operates, significantly depends on the 'amount of structure' existing in the knowledge domain under consideration. Such performance dependencies are often found among the knowledge space theory procedures. The procedure acquires information on the structure through the input of a human expert during the querying process. In the simulation studies the whole structure is given in advance as an expert's meta-knowledge. The figure 3 depicts the relationships among the number of items, the amount of structure represented both by the worst case and the best case for the procedure, and the number of questions the procedure presents to an expert on judging. The structure based on the powerset of the set of items makes the worst case, whereas the structure corresponding to the Guttman scale on the items, i.e. a single chain structure of the form $\{\emptyset, \{q_1\}, \{q_1, q_2\}, \ldots, Q\}$, represents the best case because it is assumed that the structures under consideration are discriminative spaces. The curves representing the rest of the structures lie between these two.

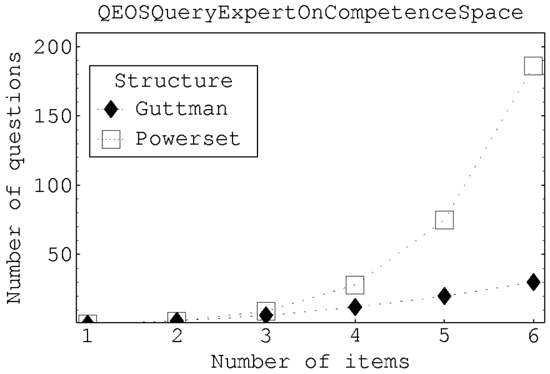

Figure 3. Number of questions presented to an expert by the `QEOSQueryExpertOnCompetenceSpace` function depends on the number of items and the amount of structure in the knowledge domain.

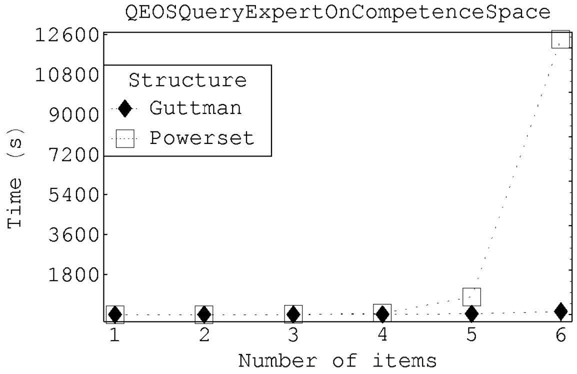

Figure 4. The `QEOSQueryExpertOnCompetenceSpace` simulation study execution times depending on the number of items and the amount of structure in the knowledge domain.

The figure 4 provides a further insight on the function performance by presenting the computation times needed by a simulation study to obtain values for the figure 3. The number of presented questions reflects the amount of structure existing in the knowledge domain. The selection of the next question and the process of drawing inferences are based on the previous expert's answers. Both require computation time, that is clearly depicted by the figure 4. The human experts are reluctant to interact with monotonous, long-lasting procedures. Therefore, the number of presented questions is closely related to the practical applicability of the procedure. The figure 4 shows that even the practical applicability of the package function for simulation studies is greatly affected by the amount of structure. Since the range of performance is indeed large, the expectations on a function performance and its applicability have to be largely based on expectations related to the structure of the knowledge domain under consideration.

Further package developments

Further developments of the package may be classified into three overlapping categories: the development of the core, developments of the modules, and developments of applications.

In the current package version, some functions reach a satisfactory performance, given the limitations of personal computers, only on small sets of items, that is on the sets whose cardinalities do not reach numbers requested by relevant empirical investigations. Therefore, efforts have to be taken to equip the package with data representations, auxiliary functions and swapping file mechanisms that ensure adequate performance of the functions on these sets as well. Further development of the KSMP Warp core is an effort in this direction. Probably, the future versions of the package will include two operating modes of the package: research mode and application mode. Research mode will be suitable for research, teaching, explorations and simulations on small sets of items, whereas the application mode will be suitable for the empirical investigations including the larger sets of items. Besides the further developments of the KSMP Warp core, the functions belonging to the package core layers have also to be upgraded to support the warp core solutions. In addition, a mechanism for an automatic selection between the research mode and the application mode, probably based on the available memory, may be included. Another category of developments consists of extending and enhancing the existing modules, as well as of developing additional ones. This category for instance includes other approaches to skills as components of the KSMP Layer two, or other querying procedures as components of the KSMP Querying module. The last, but not the least, category of further developments consists of completing the QEOS application, developing other package applications, and composing package editions. The further package releases including documentation updates should be available electronically at the package homepage (the URL http://wundt.uni-graz.at/zaluski/ksmp).

Concluding remarks

Both KSMP, the QEOS project and the QEOS application are works in progress. KSMP contributed to research and development of the querying on skills procedures by providing insights on behavior of the developed procedures, e.g. the straightforward querying on competence-performance diagnostic procedure, both by the conducted simulation studies and the applications in the empirical validations of the procedures. Since the KSMP ultimate goal has the broader scope comparing to the package requirements stipulated from the QEOS project, further editions of KSMP may contribute to research in other areas of the knowledge space theory as well. KSMP may be a motivation and a model for developing *Mathematica*-based solutions in cognitive psychology, primarily inside the areas similar to the knowledge space theory as for example in the formal concept analysis, but in other areas of the behavioral, social and educational sciences as well. Presumably, the ultimate reception of KSMP-based software solutions in both knowledge space theory applications and empirical investigations will depend on the success of KSMP in working with large size data structures and on the constraints concerning their dissemination. The satisfactory solution for handling the large size data structures is an open problem for all existing knowledge spaces software tools. However, a disadvantage of KSMP as well as its package applications lies in the fact that *Mathematica* is not a free software, and KSMP cannot be used by the knowledge space theory researchers or applicants who do not have access to *Mathematica*. Up to now, KSMP has not been used as support in teaching the knowledge space theory.

References

[1] Albert, D. (Ed.). (1994). Knowledge structures. New York: Springer-Verlag.

[2] Albert, D., Held, T. (1994). Establishing knowledge spaces by systematical problem construction. In D. Albert (Ed.), Knowledge structures (pp. 78–112). New York: Springer-Verlag.

[3] Albert, D., Held, T. (1999). Component based knowledge spaces in problem solving and inductive reasoning. In D. Albert and J. Lukas (Eds.), Knowledge spaces: theories, empirical research, applications (pp. 15–40). Mahwah, NJ: Lawrence Erlbaum Associates.

[4] Albert, D., Hockemeyer, C. (1997). Adaptive and dynamic hypertext tutoring systems based on knowledge space theory. In B. d. Boulay and R. Mizoguchi (Eds.), Artificial intelligence in education: knowledge and media in learning systems (Vol. Frontiers in artificial intelligence and applications, pp. 553–555).

[5] Albert, D., Hockemeyer, C. (1999). Developing curricula for tutoring systems based on prerequisite relationships. In G. Cumming, T. Okamoto and L. Gomez (Eds.), Advanced research in computers and communications in education: new human abilities for the networked society (Vol. 2, pp. 325–328). Amsterdam: IOS Press.

[6] Albert, D., Kaluscha, R. (1997). Adapting knowledge structures in dynamic domains. In C. Herzog (Ed.), Contributions of the 8th workshop of the GI SIG "Intelligent Tutoring Systems" (pp. 89–100). München: TU München.

[7] Albert, D., Lukas, J. (1993). Knowledge assessment based on skill assignments and psychological task analysis. In G. Strube and K. F. Wender (Eds.), The cognitive psychology of knowledge (Vol. 101 of Advances in Psychology, pp. 139–160). Amsterdam: North-Holland.

[8] Albert, D., Lukas, J. (Eds.). (1999). Knowledge spaces: theories, empirical research, applications. Mahwah, NJ: Lawrence Erlbaum Associates.

[9] Albert, D., Schrepp, M. (1999). Structure and design of an intelligent tutoring system based on skill assignments. In D. Albert and J. Lukas (Eds.), Knowledge spaces: theories, empirical research, applications (pp. 179–196). Mahwah, NJ: Lawrence Erlbaum Associates.

[10] Albert, D., Schrepp, M., Held, T. (1994). Construction of knowledge spaces for problem solving in chess. In G. H. Fischer and D. Laming (Eds.), Contributions to mathematical psychology, psychometrics and methodology (pp. 123–135). New York: Springer-Verlag.

[11] Cosyn, E., Thiéry, N. (2000). A practical procedure to build a knowledge structure. Journal of Mathematical Psychology, 44, 383–407.

[12] Degreef, E., Doignon, J.-P., Ducamp, A., Falmagne, J.-C. (1986). Languages for the assessment of knowledge. Journal of Mathematical Psychology, 30, 243–256.

[13] Doignon, J.-P. (1994a). Knowledge spaces and skill assignments. In G. H. Fisher, Laming, D. (Ed.), Contributions to mathematical psychology, psychometrics and methodology (pp. 111–121). New-York: Springer-Verlag.

[14] Doignon, J.-P. (1994b). Probabilistic assessment of knowledge. In D. Albert (Ed.), Knowledge structures (pp. 1–56). New York: Springer-Verlag.

[15] Doignon, J.-P., Falmagne, J.-C. (1985). Spaces for the assessment of knowledge. International journal of man-machine studies, 23, 175–196.

[16] Doignon, J.-P., Falmagne, J.-C. (1987). Knowledge assessment: a set theoretical framework. In B. Ganter, R. Wille and K. E. Wolfe (Eds.), Beiträge zur Begriffsanalyse: Vorträge der Arbeitstagung Begriffsanalyse. Mannheim: BI Wissenschaftsverlag.

[17] Doignon, J.-P., Falmagne, J.-C. (1988). Parametrization of knowledge structures. Discrete applied mathematics, 21, 87–100.

[18] Doignon, J.-P., Falmagne, J.-C. (1997). Well-graded families of relations. Discrete mathematics, 173, 35–44.

[19] Doignon, J.-P., Falmagne, J.-C. (1999). Knowledge spaces. Berlin: Springer-Verlag.

[20] Dowling, C. E. (1991). Constructing knowledge structures from the judgements of experts. Unpublished Habilitationsschrift, Technische Universität Carolo-Wilhelmina, Braunschweig.

[21] Dowling, C. E. (1993a). Applying the basis of a knowledge space for controlling the questioning of an expert. Journal of Mathematical Psychology, 37, 21–48.

[22] Dowling, C. E. (1993b). On the irredundant construction of knowledge spaces. Journal of Mathematical Psychology, 37, 49–62.

[23] Dowling, C. E. (1994a). Combinatorial structures for the representation of knowledge. In D. Albert (Ed.), Knowledge structures (pp. 57–76). New York: Springer-Verlag.

[24] Dowling, C. E. (1994b). Integrating different knowledge spaces. In G. H. Fischer and D. Lamings (Eds.), Contributions to mathematical psychology, psychometrics and methodology (pp. 149–158). New York: Springer-Verlag.

[25] Dowling, C. E., Hockemeyer, C. (1999). Integrating knowledge spaces obtained by querying different experts. Electronic notes in discrete mathematics, 2.

[26] Dowling, C. E., Hockemeyer, C. (to appear). Automata for the assessment of knowledge. IEEE transactions on knowledge and data engineering.

[27] Dowling, C. E., Hockemeyer, C., Ludwig, A. H. (1996). Adaptive assessment and training using the neighborhood of knowledge states. In C. Frasson, G. Gauthier and A. Lesgold (Eds.), Intelligent tutoring systems (Vol. 1086 of Lecture notes in computer science, pp. 578–586). Berlin: Springer-Verlag.

[28] Dowling, C. E., Kaluscha, R. (1995). Prerequisite relationships for the adaptive assessment of knowledge. In J. Greer (Ed.), Artificial intelligence in education. Charlottesville, VA: Association for the Advancement of Computing in Education (AACE).

[29] Dowling, C. E., Koch, U., Quante, K. A. (1996). A new interface for querying experts on prerequisite relationships. In J. Grundy and M. Apperley (Eds.), Proceedings of the 6th australian conference on computer-human interaction (OzCHI 96). Los Alamitos, CA: IEEE Computer Society Press.

[30] Düntsch, I., Gediga, G. (1995). Skills and knowledge structures. British Journal of Mathematical and Statistical Psychology, 48, 9–27.

[31] Düntsch, I., Gediga, G. (1996). On query procedures to build knowledge structures. Journal of Mathematical Psychology, 40, 160–168.

[32] Düntsch, I., Gediga, G. (1995). Skills and knowledge structures. British Journal of Mathematical and Statistical Psychology, 48, 9–27.

[33] Falmagne, J.-C. (1989). Probabilistic knowledge spaces: a review. In F. S. Roberts (Ed.), Applications of combinatorics and graph theory to biological and social sciences (Vol. 17 of IMA). New York: Springer-Verlag.

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

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

[36] Falmagne, J.-C., Koppen, M., Villano, M., Doignon, J.-P., Johannesen, L. (1990). Introduction to knowledge spaces: how to build, test and search them. Psychological Review, 97, 201–224.

[37] Freitag, O. (1999). Graphische Darstellung von Expertenurteilen. Unpublished Diplomarbeit, Teschnische Universität Braunschweig, Braunschweig, DE.

[38] Fuhrmann, F. (1995). Experimente zur Validierung von Expertenbefragungen. Unpublished Diplomarbeit, Teschnische Universität Braunschweig, Braunschweig, DE.

[39] Fuhrmann, F., Dowling, C. E. (1996). Der Einfluss von Frageformulierungen auf die Validität von Expertenurteilen. In A. Schorr (Ed.), Experimentelle Psychologie: 38. Tagung experimentell arbeitender Psychologen. Langerich, DE: Pabst Science Publishers.

[40] Held, T., Korossy, K. (1998a). Data analysis as a heuristic for establishing theoretically founded item structures. Zeitschrift für Psychologie, 206, 169–188.

[41] Held, T., Korossy, K. (1998b). Eine Simulationsstudie zu Urteilsfehlern bei Expertenbefragungen. In O. Güntürkün, R. Guski and C. Walter and A. Wohlschläger (Eds.), Experimentelle Psychologie: Beiträge zur 37. Tagung experimentell arbeitender Psychologen. Regensburg, DE: Roderer Verlag.

[42] Hockemeyer, C. (1993). Wissendiagnose auf Wissensräumen., Teschnische Universität Carolo-Wilhelmina, Braunschweig, DE.

[43] Hockemeyer, C. (1997). RATH - a relational adaptive tutoring hypertext WWW environment (Technical report). Graz, Austria: Institut für Psychologie, Karl-Franzens-Universität Graz.

[44] Hockemeyer, C., Albert, D. (1999). The adaptive tutoring system RATH. Paper presented at the ICL99 workshop interactive computer aided learning: tools and applications, Villach, AT.

[45] Hockemeyer, C., Held, T., Albert, D. (1998, June, 1998). RATH - a relational adaptive tutoring hypertext WWW environment based on knowledge space theory. Paper presented at the CALISCE 98: Proceedings of the 4th international conference on computer aided learning in science and engineering, Göteborg, Sweden.

[46] IEEE LTSC. (2001). Draft standard for information technology - learning technology - glossary (draft). IEEE Computer Society, Learning Technology Standards Committee [2001, May, 25].

[47] Kaluscha, R. (1994). Ein effizienter Algorithmus zur Expertenbefragung. Unpublished Diplomarbeit, Teschnische Universität Carolo-Wilhelmina, Braunschweig, DE.

[48] Kambouri, M. (1991). Knowledge assessment: a comparison between human experts and computerized procedure. Unpublished Doctoral Dissertation, New York University, New York.

[49] Kambouri, M., Koppen, M., Villano, M., Falmagne, J.-C. (1994). Knowledge assessment: tapping human expertise by the QUERY routine. International journal of human-computer-studies, 40, 119–151.

[50] Koch, U. (1996). Plattformübergreifende GUI - Entwicklung: Flexible und optimierte Strategien zur Expertenbefragung. Unpublished Diplomarbeit, Teschnische Universität Carolo-Wilhelmina, Braunschweig, Braunschweig, DE.

[51] Koppen, M. (1993). Extracting human expertise for constructing knowledge spaces: an algorithm. Journal of Mathematical Psychology, 37, 1–20.

[52] Koppen, M. (1994). The construction of knowledge spaces by querying experts. In G. H. Fischer and D. Lamings (Eds.), Contributions to mathematical psychology, psychometrics and methodology (pp. 137–147). New York: Springer-Verlag.

[53] Koppen, M., Doignon, J.-P. (1990). How to build a knowledge space by querying an expert. Journal of Mathematical Psychology, 34, 311–331.

[54] Körner, C. (1998). Comprehension of visualized ordered sets - an empirical approach based on the knowledge space theory. Unpublished Inauguraldissertation, Karl-Franzens-Universität Graz, Graz, Austria.

[55] Korossy, K. (1993). Modellierung von Wissen als Kompetenz und Performanz. Unpublished Inauguraldissertation, Universität Heidelberg, Heidelberg, DE.

[56] Korossy, K. (1996a). Kompetenz und Performanz beim Lösen von Geometrie Aufgaben. Zeitschrift für Experimentelle Psychologie, XLIII (2), 279–318.

[57] Korossy, K. (1996b). A qualitative-structural approach to the modeling of knowledge (Bericht aus dem Psychologischen Institut 83). Heidelberg: Universität Heidelberg.

[58] Korossy, K. (1997a). Extending the theory of knowledge spaces: a competence-performance approach. Zeitschrift für Psychologie, 205, 53–82.

[59] Korossy, K. (1997b, 24.–27. März 1997). Kompetenz und Performanz beim Lösen von Aufgaben aus der elementaren Teilbarkeitslehre. Paper presented at the Experimentelle Psychologie: Abstracts der 39. Tagung experimentell arbeitender Psychologen, Berlin.

[60] Korossy, K. (1999a). Modeling knowledge as competence and performance. In D. Albert and J. Lukas (Eds.), Knowledge spaces: theories, empirical research, applications (pp. 103–132). Mahwah, NJ: Lawrence Erlbaum Associates.

[61] Korossy, K. (1999b). Organizing and controlling learning processes within competence-performance structures. In D. Albert and J. Lukas (Eds.), Knowledge spaces: theories, empirical research, applications (pp. 157–178). Mahwah, NJ: Lawrence Erlbaum Associates.

[62] Lukas, J., Albert, D. (1999). Knowledge structures: what they are and how they can be used in cognitive psychology, test theory, and the design of learning environments. In D. Albert and J. Lukas (Eds.), Knowledge spaces: theories, empirical research, applications. Mahwah, NJ: Lawrence Erlbaum Associates.

[63] Maeder, R. E. (1996). The mathematica programmer II. San Diego, CA: Academic Press, Inc.

[64] Maeder, R. E. (1997). Programming in *Mathematica* (3rd ed.). Reading, MA: Addison-Wesley Publishing Company.

[65] Müller, C. E. (1989). A procedure for facilitating an expert's judgments on a set of rules. In E. E. Roskam (Ed.), Mathematical psychology in progress (pp. 157–170). Berlin: Springer-Verlag.

[66] Müller, R., Regenwetter, M. (1991). A procedure for facilitating an expert's judgments on a set of rules (Official report of the research project "Systeme dynamiques simules par ordinateur"). Luxemburg: Ministere de l'Education Nationale, Luxembourg.

[67] Quante, K. A. (1996). Plattformübergreifende GUI Entwicklung: Integration und Optimierung eines Programms zur Expertbefragung. Unpublished Diplomarbeit, Technische Universität Carolo-Wilhelmina, Braunschweig, DE.

[68] Schrepp, M. (1999a). Extracting knowledge structures from observed data. British Journal of Mathematical and Statistical Psychology, 52, 213–224.

[69] Schrepp, M. (1999b). On the empirical construction of implications between bi-valued test items. Mathematical social sciences, 38(361–375).

[70] Schrepp, M., Held, T. (1995). A simulation study concerning the effects of errors on the establishment of knowledge spaces by querying experts. Journal of Mathematical Psychology, 39, 376–382.

[71] Skienna, S. (1990). Implementing discrete mathematics: combinatorics and graph theory with *Mathematica*. Redwood City, CA: Addison-Wesley Publishing Co.

[72] Strube, G., Wender, K. F. (1993). The cognitive psychology of knowledge (Vol. 101 of Advances in Psychology). Elsevier.

[73] Suck, R. (1998). Ordering orders. Mathematical social sciences, 36, 91–104.

[74] Suck, R. (1999a). The basis of a knowledge space and a generalized interval order. Electronic notes in discrete mathematics, 2.

[75] Suck, R. (1999b). A dimension-related metric on the lattice of knowledge spaces. Journal of Mathematical Psychology, 43, 394–409.

[76] Theuns, P. (1998). Building a knowledge space via boolean analysis of co-occurrence data. In C. E. Dowling, F. S. Roberts and P. Theuns (Eds.), Recent progress in mathematical psychology (pp. 173–194). Hillsdale, USA: Lawrence Erlbaum Associates.

[77] Unnewehr, J. (1992). Benutzerhandbuch Prozeduren zur Wissendiagnose (Bericht aus dem Psychologischen Institut 74). Heidelberg: Universität Heidelberg.

[78] Villano, M. (1991). Computerized knowledge assessment: building the knowledge structure and calibrating the assessment routine. Unpublished Doctoral Dissertation, New York University, New York.

[79] Villano, M. (1992). Probabilistic student models: Bayesian belief networks and knowledge space theory. Paper presented at the Intelligent tutoring systems, 2nd international conference, New York.

[80] Villano, M., Bloom, C. (1992). Probabilistic student modeling with knowledge space theory (Technical report contract No. F33615-91-C-0002). Brooks AFB, TX: Technical training research division, human resources directorate.

[81] Wille, R. (1999a). Formal concept analysis. Electronic notes in discrete mathematics, 2.

[82] Wille, R. (1999b). Mathematical support for empirical theory building. Electronic notes in discrete mathematics, 2.

[83] Wolfram, S. (1999). The *Mathematica* book (4th ed.): Cambridge University Press.

COGNITIVE PSYCHOLOGY SECTION, DEPARTMENT OF PSYCHOLOGY, UNIVERSITY OF GRAZ, UNIVERSITÄTSPLATZ 2/3, A-8010 GRAZ, AUSTRIA
e-mail: `andrej.zaluski@uni-graz.at`