

6

An Empirical Test of a Process Model for Letter Series Completion Problems

Martin Schrepp

Ruprecht-Karls-Universität Heidelberg

The connection between the theory of knowledge spaces and formal models of cognitive problem solving processes is examined. This connection is described for a model of the cognitive processes of subjects in solving letter series completion problems, which is based on ideas from Simon and Kotovsky (1963). The model is formulated as an algorithm, depending on its ability to solve letter series completion problems on two parameters. Different abilities of subjects in solving such problems can be represented within the model by different choices for these parameters. It is shown that the model determines surmise relations on sets of letter series completion problems. These surmise relations, respectively the corresponding quasi-ordinal knowledge spaces, are used in two experimental investigations to test the underlying process model empirically. The results of these experiments show that the surmise relations derived from the process model are able to predict the difficulty of letter series completion problems in a satisfactory manner.

INTRODUCTION

The knowledge of a subject about a special knowledge domain is identified in the theory of knowledge spaces with his or her ability to solve problems from this domain. Such a formalization of human knowledge has the advantage to allow a precise mathematical treatment of this concept, but totally neglects cognitive processes or abilities on which the solution behavior of a subject is based. Therefore, the question arises if it is possible to extend the theory of knowledge spaces developed by Doignon and Falmagne (1985, 1998) in a way that allows the description of such underlying cognitive solution processes.

In this chapter, we examine the connection between the theory of knowledge spaces and process models of the cognitive problem solving process of subjects. We deal especially with the question of how knowledge structures, i.e. sets of knowledge states, can be derived from process models.

A process model describes in detail the cognitive processes subjects use to solve problems. So, it seems possible to derive the set of all solution patterns or knowledge states that are consistent with the assumptions about solution processes contained in the process model. The set of knowledge states consistent with a process model forms the knowledge structure corresponding to that model.

The investigation of the connection between knowledge structure and process model has two applications. First, formal models of the cognitive problem solving process can be used to establish knowledge structures, which can serve as a basis for an efficient assessment of knowledge (see for example Falmagne & Doignon, 1988). Second, process models can be tested empirically through a comparison between the knowledge structures corresponding to them and observed data patterns. We concentrate in this chapter mainly on the second application.

We examine the connection between knowledge structure and process model for a formal model of the cognitive problem solving processes of subjects in the field of letter series completion problems (short lsc-problems). This model, which is especially able to describe interindividual differences in the ability of subjects to solve such lsc-problems, is based on ideas from Simon and Kotovsky (1963).

A general approach to describe the connection between the theory of knowledge spaces and process models can be found in Schrepp (1993).

LETTER SERIES COMPLETION PROBLEMS

In lsc-problems the subjects task is to identify a regularity or rule in a presented sequence of letters of the alphabet and to use this rule to extend the sequence. Such lsc-problems are often used in intelligence tests (e.g., in Thurston's primary mental abilities test) to measure the ability of inductive reasoning.

As an example for a lsc-problem, look at the sequence $axbxcxdx$. What is the

rule used to generate this sequence and how can we find a continuation?

First, we see that every letter at an even position in the sequence is an x . Second, the letter at the odd position j is the immediate successor in the alphabet of the letter at position $j - 2$. The continuation in accordance to this rule is $exfx$, and so on.

From a strong logical standpoint there exists no unique solution for such lsc-problems. It is, for example, also possible to continue the letter sequence shown above by $axbxcxdx$ and so on. This continuation is based on the rule "Repetition of the letter block $axbxcxdx$ " and therefore also consistent with the given sequence. But such a continuation is less obvious than the one described above. As Simon and Kotovsky (1963) pointed out, it is easy to find consensus about the "correct" continuation in the lsc-problems commonly used. This correct continuation is in some sense simpler or more natural than other continuations.

To analyse the cognitive processes subjects use to detect regularities in sequences of letters a formal representation of such regularities must be formulated. Such a formal system to represent lsc-problems was introduced by Klahr and Wallace (1970).

The letter sequences of the commonly used lsc-problems are a mixture of simple sequences. In this simple sequences, every letter has a specific relation to his predecessor in the sequence. As an example, the letter sequence $axbxcxdx$ consists of the simple sequences $abcd$ and $xxxx$. Such a simple sequence can be characterized by the relation that holds between two letters and the first letter in the sequence. Therefore $abcd$ can be described by the relation "is next letter in the alphabet" and the first letter a and $xxxx$ can be described by the relation "is same letter as" and the first letter x . This description is, in the following, written by a symbol for the relation followed by the first letter in brackets. So we write $N(a)$ (N for next) to describe the sequence $abcd$ and $S(x)$ (S for same) to describe the sequence $xxxx$.

To represent a letter sequence it is sufficient to describe the simple sequences it consists of and the order in which letters from these simple sequences are chosen. This can be done by writing down the relation symbols of the simple sequences in the order of their occurrence in the letter sequence followed by the first letters of the simple sequences in brackets in the same order. We can describe our example $axbxcxdx$ by $NS(ax)$. We call such a representation of a letter sequence the *pattern description* of the sequence. This pattern description is, like the continuation of the sequence, not unique. We call the number of relational symbols in the pattern description of a letter sequence the *period* of the sequence.

In this chapter we use also the three relations "is predecessor in the alphabet" (symbolized by P), "is double-next letter in the alphabet" (symbolized by D), and "is triple-next letter in the alphabet" (symbolized by T) to describe letter sequences. As an additional example $NPDT(ekad)$ describes the sequence $ekadfcjggiej$. . . of letters.

Notice that these relations are defined cyclic concerning the alphabet. We

have, for example, $z Na$, $a Pz$, $y Da$, and $x Ta$. We call a relational dependency between two letters *cyclic* if these letters lie on different ends of the alphabet. We mark a relation in the pattern description of a lsc-problem by a $*$ if a cyclic dependency must be recognized to continue the sequence correct and call such a relation in the following a *cyclic relation* in this pattern description. For example, the letter sequence *fiehdgcfbead* is characterized by the pattern description $P^*P(fi)$, because for the continuation of this sequence, the cyclic dependency aPz must be recognized.

In the next section we describe the cognitive processes subjects use to fix a pattern description¹ of a lsc-problem.

A MODEL FOR THE SOLUTION PROCESS

In this section, we describe a model of the cognitive processes on which the solution behavior of subjects solving lsc-problems is based. To construct this model we use ideas from the work of Simon and Kotovsky (1963). These authors assume that subjects continue letter sequences by fixing a pattern description of such sequences. This pattern description is then used to determine the correct continuation. The central process in solving lsc-problems is, therefore, the construction of a pattern description for a given letter sequence. Such a pattern description is, accordingly to Simon and Kotovsky (1963), constructed by recognizing relations between specific letters in a letter sequence.

We start with a given letter sequence $x_1 \dots x_n$. The process of constructing a pattern description for this sequence consists of two subprocesses.

The first subprocess tries to find out the period p of the letter sequence $x_1 \dots x_n$. For this purpose, it searches for a simple subsequence of $x_1 \dots x_n$, that means, for a sequence $x_j x_{j+p} x_{j+2p} \dots$, in which every letter has a specific relation R to his predecessor in this subsequence. To detect such a subsequence the process uses a number of comparisons between single letters of $x_1 \dots x_n$. The second subprocess uses the information about the period to detect all the simple subsequences a given letter sequence consists of, i.e. the pattern description of that letter sequence.

The first subprocess starts with comparing for $j = 1, 2, 3, \dots, k$ the letter x_1 with x_{j+1} . The constant k denotes the largest natural number less than $n/2$. If $x_1 R x_{j+1}$ the process checks if also the dependencies $x_{j+1} R x_{2j+1}$, $x_{2j+1} R x_{3j+1}, \dots$ holds. If this is the case, then $x_1, x_{j+1}, x_{2j+1}, \dots$ is a simple subsequence and j is the expected period of the letter sequence. If the first subprocess has up to this point not found such a simple subsequence, it compares

¹ The idea to fix a special representation of regularities underlying letter sequences and to use this representation to describe the cognitive processes of subjects solving lsc-problems is due to Simon and Kotovsky (1963). The representation they used is a little different from the one in Klahr and Wallace (1970), which we described above.

for $j = 2, 3, \dots, k$ the letter x_2 with x_{j+2} . If $x_2 R x_{j+2}$ holds for a relation R , then the first subprocess checks if $x_{j+2} R x_{2j+2}$, $x_{2j+2} R x_{3j+2}$, \dots also holds. If this is the case, then $x_2, x_{j+2}, x_{2j+2}, \dots$ is a simple subsequence and j the expected period of x_1, \dots, x_n . If the process has up to this point not found a simple subsequence it starts the same comparison process for x_3 , and so on.

The second subprocess starts if the first one had successfully detected a simple subsequence and period j . This second subprocess tries to find the additional simple subsequences which are contained in the given letter sequence. This means it searches for relations R_1, \dots, R_j with:

$$\begin{array}{cccc} x_1 R_1 x_{j+1} & , & x_{j+1} R_1 x_{2j+1} & , & x_{2j+1} R_1 x_{3j+1} & , & \dots \\ \vdots & & \vdots & & \vdots & & \vdots \\ x_j R_j x_{2j} & , & x_{2j} R_j x_{3j} & , & x_{3j} R_j x_{4j} & , & \dots \end{array}$$

If the second subprocess is able to find such relations, then $R_1 \dots R_j(x_1 \dots x_j)$ is a pattern description of the given letter sequence $x_1 \dots x_n$. If the second subprocess fails in finding such relations, then the process continues with the first subprocess at the point this process is interrupted.

Up to now, we have only described the basic algorithm. This algorithm is able to detect a pattern description for every letter sequence if it is able to use all necessary relations in the first and respectively second subprocess. To model interindividual differences in the ability of subjects to solve lsc-problems, we have to introduce restrictions about the relations the algorithm can use. These restrictions are formulated by parameters λ_1 and λ_2 .

In this chapter, we restrict ourselves to lsc-problems that can be described by using only the relations S, N, P, D, T in their pattern description. But a generalization to lsc-problems that must be described by other relations is straightforward.

Let λ_1 be a 6-tuple $\langle f_S, f_N, f_P, f_D, f_T, f_Z \rangle$, where f_S, f_N, f_P, f_D, f_T are elements of $\{0, \dots, k\}$ and f_Z is an element of $\{0, 1\}$. We interpret f_R for $R \in \{S, N, P, D, T\}$ as the maximal number of letters in a letter sequence that can lie between x_i and x_j , so that $x_i R x_j$ could be detected by the first subprocess. Therefore, a relation $R \in \{S, N, P, D, T\}$ can be used in the first subprocess only for comparisons $x_i R x_j$ between single letters x_i and x_j if $f_R \geq j - i - 1$. The value f_Z determines the ability of the first subprocess to detect a cyclic relational dependency between two letters. $f_Z = 0$ indicates that the first subprocess can detect only noncyclical relational dependencies. $f_Z = 1$ indicates that there is no such restriction.

For the relations used in the second subprocess, we made a similar assumption. Let λ_2 be a 6-tuple $\langle s_S, s_N, s_P, s_D, s_T, s_Z \rangle$, where s_S, s_N, s_P, s_D, s_T are elements of $\{0, \dots, k\}$ and s_Z is an element of $\{0, 1\}$. We interpret s_R for $R \in \{S, N, P, D, T\}$ as the maximal number of letters in a letter sequence that can lie between x_i and x_j so that $x_i R x_j$ could be detected by the second sub-

process. As in the first subprocess, s_Z determines the ability to detect cyclic relational dependencies.

We assume additionally that $f_S \leq s_S, f_N \leq s_N, f_P \leq s_P, f_D \leq s_D, f_T \leq s_T$, and $f_Z \leq s_Z$. This assumption can be interpreted in the sense that the ability to detect relations between letters increases in the second subprocess. This natural assumption reflects the observation that lsc-problems are much easier to solve if the period has already been detected because the knowledge about the period can be used to search for relations that are, without this information, very hard to find.

It is also very natural to assume $f_S \geq f_N \geq f_P \geq f_D \geq f_T$ and $s_S \geq s_N \geq s_P \geq s_D \geq s_T$. This assumption reflects the observation that the relations themselves could be ordered in respect to the difficulty to detect them between single letters. For example, $f_S \geq f_N$ and $s_S \geq s_N$ means that it is easier to detect the relation S than to detect the relation N .

The ability of the described algorithm to construct pattern descriptions of letter sequences depends on the two parameters λ_1 and λ_2 . We demonstrate this dependency by an example. Define $\tau_1 := \langle 2, 0, 0, 0, 0, 0 \rangle$, $\tau_2 := \langle 2, 2, 0, 0, 0, 0 \rangle$ and $\sigma_1, \sigma_2 := \langle 2, 1, 0, 0, 0, 0 \rangle$. For the choice $\lambda_1 = \tau_1$ and $\lambda_2 = \tau_2$ the algorithm is able to construct the pattern description $SNS(urt)$ of the letter sequence *urtustutt* and fails in constructing the pattern description $NN(ad)$ of the letter sequence *adbecfdgeh* (because $f_N = 0$ indicates that the first subprocess can not detect the relation N). For the choice $\lambda_1 = \sigma_1$ and $\lambda_2 = \sigma_2$ the converse holds. The algorithm can construct $NN(ad)$ and fails in constructing $SNS(urt)$ (because $s_N = 1$ indicates that the second subprocess is not able to detect the relation $x_i N x_j$ if $j - i \geq 2$).

We interpret the algorithm for each choice of the two parameters λ_1 and λ_2 as a model for the cognitive processes of a specific subject in constructing pattern descriptions of letter sequences. So, a subject is in our model represented by a tuple $\langle \lambda_1, \lambda_2 \rangle$ of parameters. Our last two assumptions concerning the values f_R and s_R for $R \in \{S, N, P, D, T, Z\}$ restrict the possible parameter combinations $\langle \lambda_1, \lambda_2 \rangle$.

Formulations of this model and two similar models as production systems can be found in Schrepp (1993).

A detailed description of the algorithm is given by the flow chart in Fig. 1.

IMPLICATIONS ON THE DIFFICULTY OF LETTER SERIES COMPLETION PROBLEMS

The algorithm introduced in the previous section together with our assumptions concerning the possible parameter combinations can be used to derive a surmise relation \sqsubseteq on a set of lsc-problems Q . We use the surmise relation \sqsubseteq and the corresponding quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq}$ in the next section for an em-

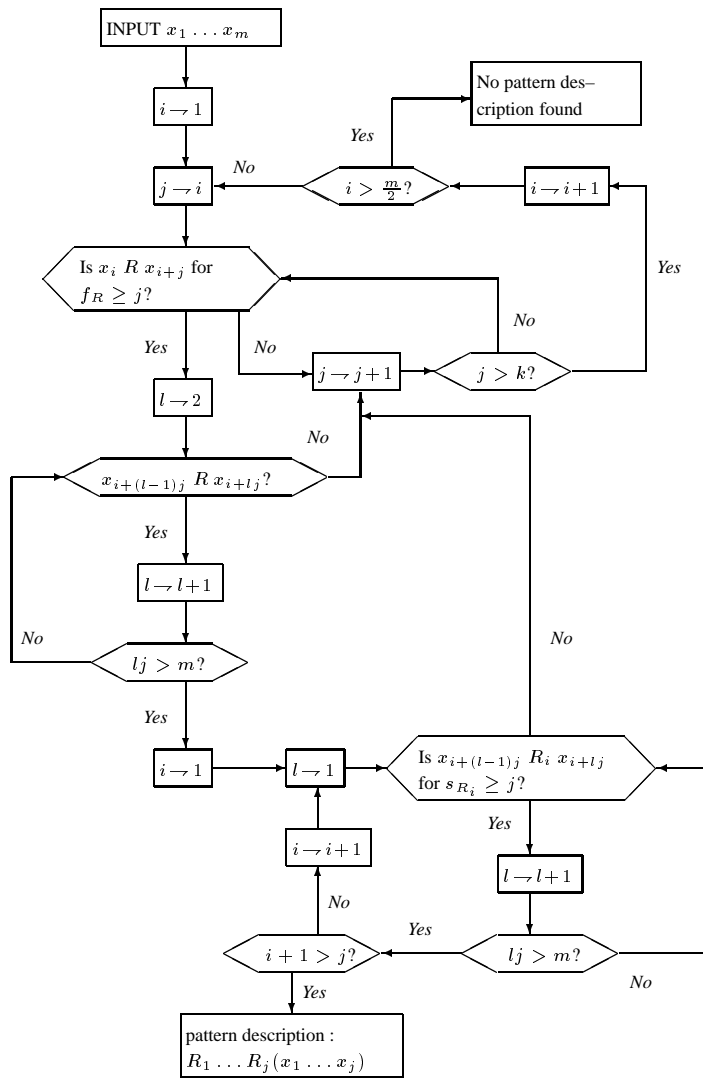


FIG. 1. Flow chart of the algorithm for the construction of a pattern description for a given letter sequence $x_1 \dots x_n$. The symbol \rightarrow should be interpreted as assigning a value to a variable, so $l \rightarrow l + 1$ means that the new value of the variable l is given by the old value of that variable plus one.

pirical test of our approach to describe interindividual differences of subjects in solving lsc-problems.

An additional assumption is needed to derive a surmise relation on a set Q of lsc-problems from our algorithm and our restrictions concerning the possible parameter values. The algorithm describes only how subjects fix a pattern description of a given letter sequence and not how they use this pattern description to continue the sequence. We assume in the following that a subject who has successfully constructed a pattern description of a letter sequence is also able to continue this sequence in accordance to that regularity without making errors. Therefore a lsc-problem should be solved correctly, if and only if a corresponding pattern description is found.

To construct a surmise relation on a set of lsc-problems we have to clarify the dependency between the possible parameter values for λ_1 and λ_2 and the ability of the algorithm to solve a lsc-problem.

Let $x_1 \dots x_n$ be a letter sequence with pattern description $R_1 \dots R_m(x_1 \dots x_m)$, $R \in \{S, N, P, D, T, N^*, P^*, D^*, T^*\}$, $\lambda_1 = \langle f_S, f_N, f_P, f_D, f_T, f_Z \rangle$ and $\lambda_2 = \langle s_S, s_N, s_P, s_D, s_T, s_Z \rangle$. The algorithm is able to construct $R_1 \dots R_m(x_1 \dots x_m)$ for $R_i \in \{S, N, P, D, T, N^*, P^*, D^*, T^*\}$ from the given sequence $x_1 \dots x_n$, if and only if the following conditions are fulfilled:

- a) $\exists i \in \{1, \dots, m\} (f_{R_i} \geq m \wedge (R_i \in \{N^*, P^*, D^*, T^*\} \rightarrow f_Z = 1))$
- b) $\forall i \in \{1, \dots, m\} (s_{R_i} \geq m \wedge (R_i \in \{N^*, P^*, D^*, T^*\} \rightarrow s_Z = 1))$.

Condition a) corresponds to the fact that the first subprocess is able to find a simple subsequence in $x_1 \dots x_n$. Condition b) corresponds to the ability of the second subprocess to detect all other simple subsequences contained in the given letter sequence $x_1 \dots x_n$.

Now, we can build up an order on the pattern descriptions of lsc-problems that reflects our assumptions concerning the influence of the possible parameter combinations on the difficulty of lsc-problems.

Let $\sigma = \langle i_S, i_N, i_P, i_D, i_T, i_Z \rangle$ and $\sigma' = \langle j_S, j_N, j_P, j_D, j_T, j_Z \rangle$ be 6-tuples with $i_S \geq i_N \geq i_P \geq i_D \geq i_T$, $j_S \geq j_N \geq j_P \geq j_D \geq j_T \in \{0, \dots, k\}$, and $i_Z, j_Z \in \{0, 1\}$. We define a partial order \preceq on the set of such 6-tuples by :

$$\sigma \preceq \sigma' \Leftrightarrow i_S \leq j_S \wedge i_N \leq j_N \wedge i_P \leq j_P \wedge i_D \leq j_D \wedge i_T \leq j_T \wedge i_Z \leq j_Z.$$

Assume a letter sequence $x_1 \dots x_n$. Let $\sigma_1(x_1 \dots x_n)$ be the smallest choice for the parameter λ_1 with respect to \preceq , so that the first subprocess of the algorithm can find a simple subsequence in $x_1 \dots x_n$. Let $\sigma_2(x_1 \dots x_n)$ the smallest choice for the parameter λ_2 with respect to \preceq , so that the second subprocess of the algorithm can find all other simple subsequences contained in $x_1 \dots x_n$. It is clear that such smallest choices $\sigma_1(x_1, \dots, x_n)$ and $\sigma_2(x_1 \dots x_n)$ concerning \preceq for these parameters exist in every case. Note that $\langle \sigma_1(x_1 \dots x_n), \sigma_2(x_1 \dots x_n) \rangle$ is

a possible parameter combination accordingly to our assumptions concerning the values f_R and s_R for $R \in \{S, N, P, D, T\}$. As an example for the letter sequence $x_1 := axbxcxdx$ we have $\sigma_1(x_1) = \langle 1, 0, 0, 0, 0, 0 \rangle$ and $\sigma_2(x_1) = \langle 1, 1, 0, 0, 0, 0 \rangle$ and for the letter sequence $x_2 := aobncmdl$ we have $\sigma_1(x_2) = \langle 1, 1, 0, 0, 0, 0 \rangle$ and $\sigma_2(x_2) = \langle 1, 1, 1, 0, 0, 0 \rangle$.

Now we are able to order lsc-problems with respect to their difficulty by defining for letter sequences $x := x_1 \dots x_n$ and $y := y_1 \dots y_m$ a relation \sqsubseteq^* through the condition :

$$x \sqsubseteq^* y \Leftrightarrow \sigma_1(x) \preceq \sigma_1(y) \wedge \sigma_2(x) \preceq \sigma_2(y).$$

So for the two examples x_1 and x_2 we have $x_1 \sqsubseteq^* x_2$ because $\sigma_1(x_1) = \langle 1, 0, 0, 0, 0, 0 \rangle \preceq \sigma_1(x_2) = \langle 1, 1, 0, 0, 0, 0 \rangle$ and $\sigma_2(x_1) = \langle 1, 1, 0, 0, 0, 0 \rangle \preceq \sigma_2(x_2) = \langle 1, 1, 1, 0, 0, 0 \rangle$.

The relation \sqsubseteq^* is as a conjunction of two quasi-orders² also a quasi-order. It reflects our assumptions concerning the dependency between possible parameter combinations and the difficulty of lsc-problems.

Assume $x \sqsubseteq^* y$. The algorithm is for the choices $\lambda_1 = \sigma_1(y)$ and $\lambda_2 = \sigma_2(y)$ able to construct the pattern description of y . But then the algorithm is also able to construct the pattern description of x since $\sigma_1(x) \preceq \sigma_1(y)$ and $\sigma_2(x) \preceq \sigma_2(y)$. Because $\lambda_1 = \sigma_1(y)$ and $\lambda_2 = \sigma_2(y)$ are minimal with respect to \preceq the algorithm is for any choice of the parameters able to solve x if he is able to solve y . Therefore $x \sqsubseteq^* y$ for two lsc-problems x and y can be interpreted as “If the algorithm is for a choice of values for λ_1 and λ_2 able to construct the pattern description of y then he is also able to construct the pattern description of x ”.

A problem is that according to \sqsubseteq^* only the relations in the pattern description of a letter sequence influences the difficulty to construct a pattern description of this sequence. As an example, look at the letter sequences *aabccedg* and *agbickdm*. Their pattern descriptions are *ND(aa)* and *ND(ag)* and therefore with respect to the relations in these pattern descriptions identical. Therefore *aabccedg* \sqsubseteq^* *agbickdm* and *agbickdm* \sqsubseteq^* *aabccedg*. However intuitively it seems much harder to continue the first sequence than the second. The reason is that in the first sequence a lot of relations between letters come in mind that are superfluous for the construction of the pattern description and therefore maybe confusing. For example, in this sequence the first two letters are identical and the third letter is the alphabetical successor of the second letter.

Our algorithm is already able to reflect this intuitively expected difference in difficulty between this examples, since he needs more processing steps to construct a pattern description of *aabccedg* than to construct a pattern description of *agbickdm*. The reason is that the first subprocess detects in solving *aabccedg* the

² Note that the generalization of the partial order \preceq from the set of parameters to the set of letter sequences gives only a quasi-order because different letter sequences may correspond to the same values for σ_1 and σ_2 .

relation S between the first and the second letter. Therefore, it had to find out that this relation does not hold between the second letter and the third. Only after this two steps can the correct dependency N between the first and third letter be detected. In contrast the algorithm detects no relation between first and second letter of *agbickdm*. The correct relation N between the first and the third letter can, therefore, be detected directly. Thus, constructing the pattern description of *aabccedg* needs more processing steps (look at the flow chart in Fig. 1) than constructing the pattern description of *agbickdm*.

The algorithm is able to construct for any parameter combination either a pattern description for both problems or for none of them, but a different number of processing steps is needed in both cases to do so. If we assume a monotonic dependency between processing steps needed to construct a pattern description and solution time the solution of *aabccedg* should take more time than the solution of *agbickdm*. In situations in which the solution time is restricted (as in the experiments we report in the next section) it is therefore consistent with our model that a solution of *agbickdm* is given while no such solution of *aabccedg* can be found.

So, we have to change \sqsubseteq^* in a way that takes into account the number of steps our algorithm needs for constructing the pattern descriptions of letter sequences. This is relatively simple because such differences in the necessary processing steps can only result from detected relations between letters that are not in accordance with the pattern description. For example, in the sequence *aabccedg* the relation S is detected between the first two letters, but this relation is not in accordance with the pattern description $ND(aa)$ of this sequence.

We define a surmise relation \sqsubseteq by $x \sqsubseteq y$ if and only if $x \sqsubseteq^* y$ and for constructing the pattern description of x at least as much processing steps are necessary than for the construction of the pattern description of y ³. For example, we have *agbickdm* \sqsubseteq *aabccedg* but *aabccedg* $\not\sqsubseteq$ *agbickdm*. To find out if $x \sqsubseteq y$ holds or not we have for pairs x, y of lsc-problems with $x \sqsubseteq^* y$ only to count how often the algorithm detects a relation that is not in accordance with the finally constructed pattern description⁴.

EXPERIMENT I

To test our model empirically, a problem set containing 20 lsc-problems was chosen and the surmise relation \sqsubseteq_I based on the principles described in the last section was constructed. We compare \sqsubseteq_I and the corresponding quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq_I}$ with the obtained data patterns of 51 subjects.

³ Note that this procedure for constructing \sqsubseteq from \sqsubseteq^* can not yield intransitivities, because if we have $x \sqsubseteq^* y \sqsubseteq^* z$ and $x \not\sqsubseteq^* z$ we have also $x \not\sqsubseteq y$ or $y \not\sqsubseteq z$. \sqsubseteq is therefore transitive.

⁴ The length of the period of a pattern sequence influences also the number of processing steps needed by the algorithm to construct a pattern description. However, because $x \sqsubseteq^* y$ implies that the period of y is greater or equal to the period of x , we can neglect this influence.

TABLE 1
Lsc-Problems Used in Experiment I.

<i>problem number</i>	<i>letter sequence</i>	<i>pattern description</i>	<i>correct continuation</i>
1	<i>cdc dcdcd</i>	<i>SS(cd)</i>	<i>cdc</i>
2	<i>gohpiqjrks</i>	<i>NN(go)</i>	<i>ltm</i>
3	<i>tbaxtbaxtb</i>	<i>SSSS(tbax)</i>	<i>axt</i>
4	<i>fiehdgcfbe</i>	<i>P*P(fi)</i>	<i>adz</i>
5	<i>abyabxabwab</i>	<i>SSP(aby)</i>	<i>vab</i>
6	<i>adbecfdgeh</i>	<i>NN(ad)</i>	<i>fig</i>
7	<i>jkorklpqlmqt</i>	<i>NNNN(jkor)</i>	<i>mnr</i>
8	<i>ehgjilkmp</i>	<i>DD(eh)</i>	<i>orq</i>
9	<i>urtustuttu</i>	<i>SNS(urt)</i>	<i>utu</i>
10	<i>pvountmslr</i>	<i>PP(pv)</i>	<i>kqj</i>
11	<i>eafgbhicjkl</i>	<i>DND(eaf)</i>	<i>men</i>
12	<i>dhefcgdebcd</i>	<i>PPPP(dhef)</i>	<i>aeb</i>
13	<i>npaoqapraqsa</i>	<i>NNS(npa)</i>	<i>rta</i>
14	<i>axdcxfexhgxi</i>	<i>DSD(axd)</i>	<i>ixl</i>
15	<i>ekofkngkmhk</i>	<i>NSP(eko)</i>	<i>lik</i>
16	<i>jkillminnoip</i>	<i>DDSD(jkil)</i>	<i>pqi</i>
17	<i>fgjggkhglig</i>	<i>N SN(fgj)</i>	<i>mjj</i>
18	<i>wrnfvqmeupld</i>	<i>PPPP(wrnf)</i>	<i>tok</i>
19	<i>dhcbfjedhlgf</i>	<i>DDDD(dhcb)</i>	<i>jni</i>
20	<i>mrnorpqrrst</i>	<i>DSD(mrn)</i>	<i>urv</i>

Method

Subjects

The experiment was conducted with 25 female and 26 male subjects. Their ages ranged from 12 to 53 years. The average age was 26 years. For their participation, the subjects were paid with 12,- DM. The subjects were recruited by an announcement in the local newspaper and the institutes of the university.

Problems

The problem set consists of 20 lsc-problems that are shown in Table 1. We refer to this problem set in the following as Q_I . Some of these problems were constructed especially for the investigation; others were taken from the literature concerning lsc-problems. In each problem, three letters has to be given as a continuation of the sequence by the subjects. Note that in Problem 4 a cyclic relational dependency has to be detected to continue the sequence.

Derived surmise relation

In Fig. 2 the surmise relation \sqsubseteq_I on the problem set Q_I is shown as a Hasse-Diagram. The corresponding quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq_I}$ consists of 276 knowledge states.

Procedure

The investigation was conducted as a group experiment. The size of a group varied between 3 and 11 subjects. First, a general instruction was handed out that explained the problem type “lsc-problems”. Then the subjects had to solve 4 lsc-problems as exercises. After all subjects of a group had finished these exercises, they were informed about the correct solutions.

Then, a second instruction was handed out that informed the subjects about the general experimental conditions. The subjects were instructed not to spend more than 2 minutes on each problem. They were able to control this time themselves on a watch in the room.

After all subjects had finished reading this second instruction the 20 lsc-problems from Table 1 were handed out. Each problem was printed on a separate sheet. The problems were ordered in increasing sequence of their numbers in Table 1. The subjects were instructed to work on the problems in exactly this order and not to go back to problems already finished. As solution they had to write

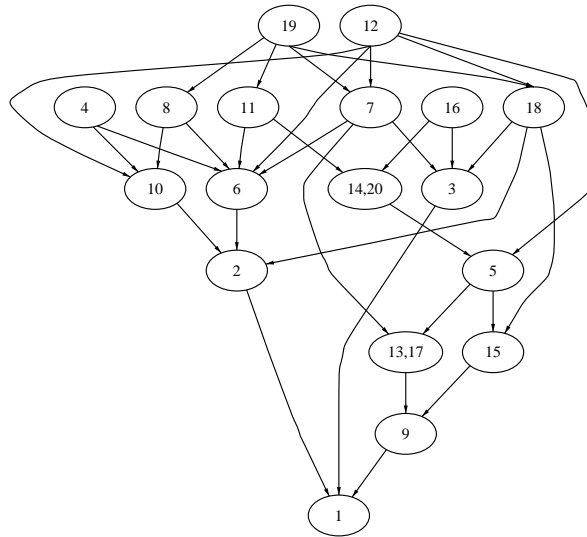


FIG. 2. Hasse-Diagram of the derived surmise relation \sqsubseteq_I on the problem set Q_I .

TABLE 2
Frequency of Symmetric Distances Between Response Patterns
and Closest States of the Quasi-Ordinal Knowledge Space $\mathcal{W}_{\sqsubseteq_I}$ Corresponding to \sqsubseteq_I .

	<i>Distances</i>					<i>Average</i>	
	<i>0</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>Distance</i>
<i>Frequencies</i>	23	14	9	5	0	0	0.92

down three letters as a continuation behind the presented sequence. Detailed informations concerning the experimental procedure of Experiment I can be found in Schrepp (1993).

Results

The average solution frequency was 81 %. This high average solution frequency results mainly from the fact that some problems (1, 2, 3, 5, 9, 10, 11, 13, 15) were solved by a large majority (more than 46) of subjects.

First we compare the observed response patterns with the knowledge space $\mathcal{W}_{\sqsubseteq_I}$. Therefore we look at the symmetric distances⁵ between response patterns and the closest states⁶ of $\mathcal{W}_{\sqsubseteq_I}$ shown in Table 2.

Notice that because $\mathcal{W}_{\sqsubseteq_I}$ includes the whole problem set Q_I and \emptyset the highest theoretically possible symmetric distance between a response pattern and a corresponding closest state in \mathcal{W}_I is 10. Because of the high average solution frequency, the highest possible average symmetric distance of an arbitrary knowledge structure containing Q_I to our data set is 3.8.

To get more information, about the deviations between our data and the theoretically derived surmise relation \sqsubseteq_I , we counted for each pair a, b of problems from Q_I with $a \sqsubseteq_I b$ how often this assertion is violated empirically. A violation of $a \sqsubseteq_I b$ means that b is solved by a subject that fails in solving a . Table 3 shows for each pair a, b of lsc-problems with $a \sqsubseteq_I b$ the number of such violations.

Discussion

Table 2 shows that the symmetric distances between observed response patterns of subjects and the closest states of $\mathcal{W}_{\sqsubseteq_I}$ are rather small. Note that 23 response

⁵ The symmetric distance d between two subsets A and B of Q is defined by $d(A, B) := |A \Delta B| = |(A \setminus B) \cup (B \setminus A)|$.

⁶ A closest state in a knowledge structure \mathcal{W} for an observed response pattern R is a knowledge state W of \mathcal{W} with $d(W, R) = \min\{d(W', R) \mid W' \in \mathcal{W}\}$. Notice that for an observed response pattern there are in general many closest states in a knowledge structure.

TABLE 3
 Violations of Relational Dependencies.
 The i -th Row and the j -th Column of the Table Contains an – if $i \sqsubseteq_I j$.
 If $i \sqsubseteq_I j$ the Number of Observed Violations of this Dependency is Shown.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	-	0	-	2	-	3	0	1	-	4	3	1	-	-	-	-	-	4	0	-
3	-	-	0	-	-	-	0	-	-	-	-	0	-	-	-	0	-	0	0	-
4	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	0	-	-	-	-	-	0	0	-	0	-	0	-	-	0	0
6	-	-	-	4	-	0	2	3	-	-	4	0	-	-	-	-	-	-	0	-
7	-	-	-	-	-	-	0	-	-	-	-	3	-	-	-	-	-	-	3	-
8	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	0	-
9	-	-	-	-	2	-	2	-	0	-	2	1	2	2	2	2	2	2	1	2
10	-	-	-	1	-	-	-	1	-	0	-	0	-	-	-	-	-	-	0	-
11	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	0	-
12	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-
13	-	-	-	-	4	-	1	-	-	-	3	0	0	3	-	2	2	-	0	3
14	-	-	-	-	-	-	-	-	-	-	6	-	-	0	-	6	-	-	0	5
15	-	-	-	-	3	-	-	-	-	-	2	0	-	1	0	2	-	1	1	2
16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-
17	-	-	-	-	6	-	4	-	-	-	6	1	5	5	-	6	0	-	1	5
18	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-	-	-	0	1	-
19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-
20	-	-	-	-	-	-	-	-	-	-	6	-	-	6	-	6	-	-	2	0

patterns (45 %) are states of $\mathcal{W}_{\sqsubseteq_I}$ whereas $\mathcal{W}_{\sqsubseteq_I}$ contains only 276 (0.00026 %) of the 2^{20} possible response patterns.

This is a relatively satisfactory result because we have to consider influences like, for example, lucky guesses, careless errors, fatigue or motivational decrease⁷ during the experimental procedure.

The number of violations of $i \sqsubseteq_I j$ in Table 3 is for all such pairs of problems less or equal to 6 (of 51 subjects). In conclusion, the assumptions of \sqsubseteq_I concerning the difficulty of lsc-problems seems to be in accordance with the data.

⁷ If we assume for example that the probability of both careless errors and lucky guesses is 0.05, then the probability that the response pattern of a subject is equal to his knowledge state is $0.95^{20} \approx 0.35$.

EXPERIMENT II

In our second experiment, a set Q_{II} containing 24 lsc-problems was chosen and the surmise relation \sqsubseteq_{II} on Q_{II} based on the principles derivated from the process model was derived. We compare \sqsubseteq_{II} and the corresponding quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq_{II}}$ with the observed data patterns of 53 subjects.

Method

Subjects

The experiment was conducted with 26 female and 27 male subjects. Their ages ranged from 15 to 64 years. The average age was 26 years. For their participation, the subjects were paid with 12,- DM. The subjects were recruited by an announcement in the local newspaper and the institutes of the University of Heidelberg.

Problems

The problem set Q_{II} consists of 24 lsc-problems which are shown in Table 4. In each problem, the next four letters has to be given as a continuation of the sequence by the subjects. In the problems 5, 20, 22, a cyclic relational dependency has to be detected to continue the sequence.

Derived Surmise Relation

In Fig. 3, the surmise relation \sqsubseteq_{II} is depicted as a Hasse-Diagram. The corresponding quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq_{II}}$ consists of 2942 knowledge states.

Procedure

The investigation was conducted as a single case experiment that was completely controlled by a computer. The experimental program was developed using MEL (see Schneider, 1990).

First, the subjects received some general informations about the problem-type "lsc-problems". This information was followed by instructions about the way in which the problems would be presented on the screen and the usage of the keyboard to type in the solutions. Then, five lsc-problems are presented as an exercise. This exercise should enable the subject to become familiar with lsc-problems and to train the experimental procedure. Feedback about the correct solution was given after each of the five exercise problems.

After a subject had finished the exercises, a second instruction was presented on the screen. The subject was informed in this instruction that he or she had 3

minutes time to solve each of the following 24 problems. The remaining time for a problem was shown in seconds on the screen counting from 180 down to 0. If a subject exceeded the maximal time, the problem presentation was terminated automatically and the problem was counted as unsolved.

Then, the 24 problems from Table 4 were presented. The presentation of a problem was terminated as soon as the subject typed in a continuation (or exceeded the time limit). The subject was able to start the presentation of the next problem by himself through pressing one of the function keys. This enabled the subject to take small breaks between two problems. The order of problem presentation was randomized for each subject. During this experimental phase, no feedback about the correctness of the solutions was presented. For detailed informations concerning this experiment see Schrepp (1993).

TABLE 4
Lsc-Problems Used in Experiment II.

<i>problem number</i>	<i>letter sequence</i>	<i>pattern description</i>	<i>correct continuation</i>
1	<i>bhibhibhibhi</i>	<i>SSS(bhi)</i>	<i>bhib</i>
2	<i>blqcmrdnseot</i>	<i>NNN(blq)</i>	<i>fpug</i>
3	<i>lokojoiohogo</i>	<i>PS(lo)</i>	<i>foeo</i>
4	<i>cmjeolgnisp</i>	<i>DDD(cmj)</i>	<i>kurm</i>
5	<i>apzypxpwpvp</i>	<i>P*S(ap)</i>	<i>uptp</i>
6	<i>wrnvqmupltok</i>	<i>PPP(wrn)</i>	<i>snjr</i>
7	<i>jkjkjilihmg</i>	<i>NPP(jkj)</i>	<i>ngfo</i>
8	<i>dlngoqjrtmuw</i>	<i>TTT(dln)</i>	<i>pxzs</i>
9	<i>ckofdlpgemqh</i>	<i>NNNN(ckof)</i>	<i>fnri</i>
10	<i>difkhmjolqns</i>	<i>DD(di)</i>	<i>purw</i>
11	<i>jknlmnnonp</i>	<i>DDSD(jknl)</i>	<i>pqnr</i>
12	<i>blcmdneofpgq</i>	<i>NN(bl)</i>	<i>hris</i>
13	<i>lpdjnrflpthn</i>	<i>DDDD(lpdi)</i>	<i>rvjp</i>
14	<i>eofngmhlakij</i>	<i>NP(eo)</i>	<i>kilh</i>
15	<i>hhigjfkeldmc</i>	<i>NP(hh)</i>	<i>nboa</i>
16	<i>pvountmslrkq</i>	<i>PP(pv)</i>	<i>jpio</i>
17	<i>jcxglexingxk</i>	<i>DDSD(jcxg)</i>	<i>pixm</i>
18	<i>falxiboxlcrx</i>	<i>TNTS(falx)</i>	<i>odux</i>
19	<i>wrnfvqmeupld</i>	<i>PPPP(wrnf)</i>	<i>tokc</i>
20	<i>ludknwdmpydo</i>	<i>DD*SD(ludk)</i>	<i>radq</i>
21	<i>emrflqgkphjo</i>	<i>NPP(emr)</i>	<i>iinj</i>
22	<i>jkeldmcnboa</i>	<i>NP*(jf)</i>	<i>pzqy</i>
23	<i>lagcmdfengeg</i>	<i>NTPD(lagc)</i>	<i>ojdi</i>
24	<i>ekofkngkmhkl</i>	<i>NSP(eko)</i>	<i>ikkj</i>

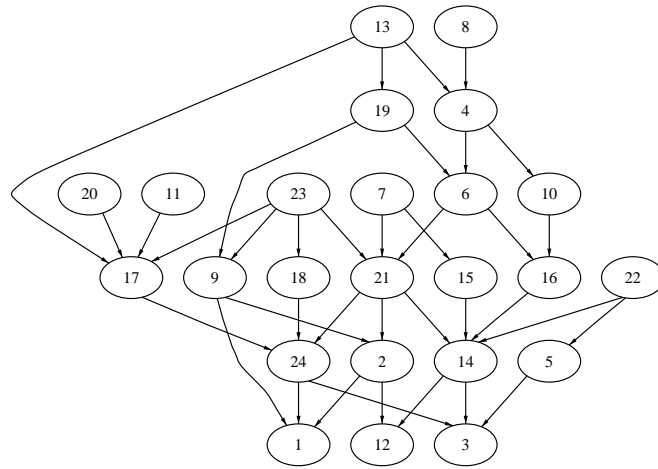
FIG. 3. Hasse-Diagram of the derived surmise relation \sqsubseteq_{II} on the problem set Q_{II} .

TABLE 5
Frequency of Symmetric Distances Between Response Patterns
and Closest States of the Quasi-Ordinal Knowledge Space $\mathcal{W}_{\sqsubseteq_{II}}$ Corresponding to \sqsubseteq_{II} .

Distances						Average	
0	1	2	3	4	5	6	Distance
14	18	12	4	3	2	0	1.43

Results

The average solution frequency over all subjects and all problems was 79 %. As in Experiment I this high average solution frequency results mainly from the fact that some of the problems were too easy for a large majority of subjects participating in the experimental investigation. The problems 1, 2, 3, 5, 9, 12, 14, 15 were solved by more than 46 subjects.

Table 5 shows the frequency of symmetric distances between the observed response patterns and the closest states of the quasi-ordinal knowledge space $\mathcal{W}_{\sqsubseteq_{II}}$. Because $\mathcal{W}_{\sqsubseteq_{II}}$ includes the whole problem set Q_{II} and \emptyset , the highest theoretically possible symmetric distance between an observed response pattern and a corresponding closest state in \mathcal{W}_{II} is 12. Because of the high average solution frequency, the highest possible average symmetric distance of an arbitrary knowledge structure including Q_{II} to our data set is 5.04.

The number of violations of assertions of the form $a \sqsubseteq_{II} b$ is for each such pair a, b of problems from Table 4 listed in Table 6.

TABLE 6
Violations of Relational Dependencies.
The i -th Row and the j -th Column of The Table Contains an - if $i \not\sqsubseteq_{II} j$.
If $i \sqsubseteq_{II} j$ the Number of Observed Violations of This Dependency is Shown.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	-	1	-	1	1	1	-	-	-	1	-	-	-	-	-	1	-	1	-	1	1	
2	-	0	-	0	-	1	0	1	2	-	-	-	1	-	-	-	-	2	-	2	-	2	-	
3	-	-	0	0	2	1	0	0	-	0	0	-	0	0	2	1	1	0	1	0	0	1	0	1
4	-	-	-	0	-	-	-	5	-	-	-	-	4	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	-	-
6	-	-	-	3	-	0	-	2	-	-	-	-	3	-	-	-	-	3	-	-	-	-	-	-
7	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	0	-	-	-	0	-	-	-	-	1	-	-	-	-	0	-
10	-	-	-	7	-	-	-	6	-	0	-	-	6	-	-	-	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-
12	-	1	-	0	-	1	0	0	1	0	-	0	0	0	0	0	-	0	-	1	0	0	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-
14	-	-	-	1	-	4	1	1	-	0	-	-	1	0	5	2	-	2	-	1	5	0	-	-
15	-	-	-	-	-	-	1	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-	-	-
16	-	-	-	5	-	9	-	4	-	5	-	-	3	-	-	0	-	4	-	-	-	-	-	-
17	-	-	-	-	-	-	-	-	-	-	6	-	6	-	-	-	0	-	-	-	-	-	8	-
18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	1	-
19	-	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-	-	0	-	-	-	-	-	-
20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	-
21	-	-	-	6	-	10	3	6	-	-	-	-	4	-	-	-	-	9	-	0	-	4	-	-
22	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-
23	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-
24	-	-	-	3	-	5	2	2	-	-	0	-	2	-	-	-	3	5	4	3	1	-	1	0

Discussion

As Table 5 shows, the symmetric distances between observed response patterns and the closest states of $\mathcal{W}_{\sqsubseteq_{II}}$ are, in almost all cases, rather small. 14 of the observed response patterns (26 %) are states of $\mathcal{W}_{\sqsubseteq_{II}}$ while $\mathcal{W}_{\sqsubseteq_{II}}$ contains only 2942 (0.00017 %) of the 2^{24} possible response patterns.

The observed average distance between response patterns and closest states in Experiment II is higher than in Experiment I. This is not a surprise for two reasons. First, $\mathcal{W}_{\sqsubseteq_{II}}$ is smaller than $\mathcal{W}_{\sqsubseteq_I}$ in the sense that it contains a smaller percentage of the possible response patterns (0.00017 % compared to 0.00026 %) and therefore we expect a higher average distance. Second, the influence of lucky guesses, careless errors, fatigue, or motivational decrease on the resulting distances will clearly increase with the number of problems presented. Because

we have presented more problems in Experiment II than in Experiment I (24 compared to 20), a higher average distance should be observed in Experiment II.

The observed number of violations of relational dependencies is, as Table 6 shows, with a few exceptions ($10 \sqsubseteq 4, 16 \sqsubseteq 6, 21 \sqsubseteq 6, 21 \sqsubseteq 19, 17 \sqsubseteq 23$) rather small (less than 7 of 53 subjects). The assumptions about the difficulty of lsc-problems contained in \sqsubseteq_{II} seems to be in accordance with the data.

GENERAL DISCUSSION

We examined the connection between knowledge structures and models of the cognitive problem solving process for the example of a model for cognitive processes in letter series completion. We demonstrated how knowledge structures, in our special case quasi-ordinal knowledge spaces corresponding to surmise relations, can be derived from this process model and how these derived knowledge structures can be used for an empirical test of the model.

In the derivation of the surmise relation \sqsubseteq from our process model we can distinguish two steps. First, we derived a quasi-order \sqsubseteq^* from the model using only the information concerning the relations in the pattern descriptions of lsc-problems. This first step can be completely characterized as a special skill assignment (see Doignon, 1994). For the construction of a knowledge structure by a skill assignment a set of cognitive skills that are important in the underlying knowledge domain is fixed. The skills are then used to derive the set of all possible knowledge states of subjects by postulating a connection between the ability to solve problems from the knowledge domain and the mastery of these skills. In our special case, the set of skills is given by the set of all possible parameter tuples $\langle \lambda_1, \lambda_2 \rangle$ and the connection between skills and the ability to solve lsc-problems is given by the conditions a) and b). In a second step, we used detailed information about the processing steps of the algorithm to construct the final surmise relation \sqsubseteq from \sqsubseteq^* . The detailed description of the solution process contained in the process model is not used until this second step.

In the following, we discuss if the reported experimental investigations support the described model for cognitive processes in letter series completion. In both experiments, the symmetric distances between the observed response patterns and the corresponding closest states in the quasi-ordinal knowledge spaces derived from the process model were rather small. These observed symmetric distances can be explained by influences like lucky guesses, careless errors or motivational decrease during the experimental procedure.

An additional analysis of the predicted relational dependencies between lsc-problems shows that they are violated empirically only in a small number of cases, with a few exceptions in Experiment II. The assumptions about the difficulty of lsc-problems contained in the surmise relations derived from the process model seems to be in good accordance with the data observed in the two experiments,

even if we take into account the high average solution frequency in these experiments.

But, we have to mention also that the surmise relations \sqsubseteq_I and \sqsubseteq_{II} seems to be not complete, that means there seems to be more relational dependencies between lsc-problems than predicted by these surmise relations. The surmise relations \sqsubseteq on sets of lsc-problems derived from our process model depends strongly on the assumptions about the possible parameter tuples described previously. We have formulated in this section only those restrictions that seem to be very natural in respect to our interpretation of these parameters. Additional restrictions concerning the possible parameter combinations will lead to stricter surmise relation \sqsubseteq' . Such a surmise relation \sqsubseteq' will contain \sqsubseteq but will in general predict additional relational dependencies between lsc-problems.

If we add, for example, the assumption that for the ability of the first and second subprocess of the algorithm to detect the relation S between two letters x_i and x_j the number of letters between x_i and x_j plays no role (formal $f_S, s_S \in \{0, k\}$) the surmise relations \sqsubseteq'_I and \sqsubseteq'_{II} derived from the model will become much stricter.

The surmise relation \sqsubseteq'_I contains under this additional assumption the additional dependencies $3 \sqsubseteq'_I 1, 2, 4, 5, 6, 8, 9, 10, 11, 13, 14, 15, 17, 20$. None of these new relational dependencies is violated empirically in Experiment I. The surmise relation \sqsubseteq'_{II} contains under this additional assumption, the additional relational dependencies $1 \sqsubseteq'_{II} 3, 5, 10, 12, 14, 15, 16, 22$. These new relational dependencies are violated empirically at most once in Experiment II. The observed symmetric distances of $\mathcal{W}_{\sqsubseteq'_I}$ and $\mathcal{W}_{\sqsubseteq'_{II}}$ to the data patterns in our experiments are identical with the observed symmetric distances of $\mathcal{W}_{\sqsubseteq_I}$ and $\mathcal{W}_{\sqsubseteq_{II}}$ to these data patterns, whereas $\mathcal{W}_{\sqsubseteq'_I}$ contains only 216 knowledge states and $\mathcal{W}_{\sqsubseteq'_{II}}$ contains 2919 knowledge states. So, the additional restriction $f_S, s_S \in \{0, k\}$ seems to be well suited by the data and improves the predictions of the model concerning the difficulty of lsc-problems.

Our considerations lead to the following two observations. First, our model predicts not all relational dependencies that seem to be immanent in our data sets. As we have seen this may result from the fact that we have not formulated enough restrictions on the possible parameter combinations. Second, an analysis of the experimental data can be used to search for additional restrictions on possible parameter combinations. Such additional restrictions can be incorporated in the model and can improve the models predictions concerning the difficulty of lsc-problems. We have not enforced such an analysis in detail for our two data sets because the high average solution frequency in our two experiments indicates that many of the presented lsc-problems are too easy for a large majority of subjects participating in these experiments. Therefore, an eventually existing difference in the difficulty of two lsc-problems may not be observable in our data sets.

ACKNOWLEDGEMENTS

This article is based on the authors doctoral dissertation at the University of Heidelberg, supported by a fellowship of the state Baden–Württemberg. The reported experiments were supported by Grant Lu 385/1 of the Deutsche Forschungsgemeinschaft to J. Lukas and D. Albert. I am grateful to B. Hirschmüller and D. Klein for their support in conducting the experimental investigations.

REFERENCES

- Doignon, J. P. (1994). Knowledge spaces and skill assignments. In: G. Fischer & D. Laming (Eds.), *Contributions to mathematical psychology, psychometrics, and methodology* (pp. 111–121). Berlin: Springer.
- Doignon, J. P., & Falmagne, J. C. (1985). Spaces for the assessment of knowledge. *International Journal of Man–Machine Studies*, 23, 175–196.
- Doignon, J.-P., & Falmagne, J.-C. (1998). *Knowledge spaces*. Berlin: Springer.
- Falmagne, J. C., & Doignon, J. C. (1988). A markovian procedure for assessing the state of a system. *Journal of Mathematical Psychology*, 32, 232–258.
- Klahr, K., & Wallace, J.G. (1970). The development of serial completion strategies: An information processing analysis. *British Journal of Psychology*, 61, 243–257.
- Kotovsky, K., & Simon, H.A. (1973). Empirical tests of a theory of human acquisition of concepts for sequential patterns. *Cognitive Psychology*, 4, 399–424.
- Schneider, W. (1990). MEL user's guide: Computer techniques for real time experimentation. *Pittsburgh: Psychology Software Tools*.
- Schrepp, M. (1993). *Über die Beziehung zwischen kognitiven Prozessen und Wissensräumen beim Problemlösen*. [On the relation between cognitive processes and knowledge spaces in problem solving.] Unpublished doctoral dissertation, University of Heidelberg, Germany.
- Simon, H. A., & Kotovsky, K. (1963). Human acquisition of concepts for sequential patterns. *Psychological Review*, 70, 534–546.

