HOW TO MEASURE THE LEARNING PROCESS IN MAN-MACHINE SYSTEMS

Matthias RAUTERBERG and Roger AEPPLI

Work and Organizational Psychology Unit (IfAP), Swiss Federal Institute of Technology (ETH)
Nelkenstrasse 11, CH-8092 Zürich, SWITZERLAND
Tel.: +41-1-6327082, Fax: +41-1-6321186, Email: rauterberg@ifap.bepr.ethz.ch

Keywords:
Learning; cognitive structure; complexity; Petri net

Abstract:
A learning experiment was carried out to investigated the development of mental structures. Six subjects were carefully instructed to operate a commercial database management system (DBMS). The long-term knowledge about general DBMS- and computer experience was measured with a questionnaire once at the beginning of the investigation. On three weeks in a row all subjects had to solve the same task twice repeated in an individual session, overall there are six solutions of the same task. At the beginning of each of the three individual sessions the short-term knowledge about the task and the tool was measured with a short questionnaire. For each task solving process all keystrokes were recorded with a time stamp in a logfile. With a special analysing program the logical structure of each empirically observed task solving process was extracted. This logical structure is given as a Petri net. The behavioural complexity (BC) of this net structure can be measured with the McCabe-measure. With some special assumptions the cognitive complexity (CC) can be derived from the empirically gained BC. The main results are discussed.

INTRODUCTION

Learning increases constantly the complexity of the mental model. This is an irreversible process. One consequence is, that the contextual complexity must increase appropriately to fit the human needs for optimal variety. Based on the empirical result in (Rauterberg, 1993), that the complexity of the observable behaviour of novices is larger than the complexity of experts, we concluded that the behavioural complexity is negatively correlated with the complexity of the mental model. Thus it is possible to estimate the cognitive complexity based on the measurement of the behavioural complexity, the measurement of the system complexity and the measurement of the task complexity (for a more detailed discussion see (Rauterberg, 1992a).

The symbolic representation of the machine system consists of the following elements: 1. Objects (things to operate on), 2. operations (symbols and their syntax), and 3. states (the 'system states'). The mental model of the user can be structured in representing: objects, operations, states, system structure, decision and task structure. A Petri-net can be described as a mathematical structure consisting of two non-empty disjoint sets of nodes (S-elements and T-elements), and a binary flow relation (F). The flow relation links only different node types and leaves no node isolated (Petri, 1980). Petri nets can be interpreted in our context by using a suitable pair of concepts for the sets S (signified by a circle '( )') and T (signified by a square '[ ]') and a suitable interpretation for the flow relation F (signified by an arrow '->'). The main operations (relations) between two Petri nets are *abstraction*, *embedding* and *folding*.

The *folding operation* in the Petri-net theory is the basic idea of our approach. Folding a process means to map S-elements onto S-elements and T-elements onto T-elements while keeping the F-structure. The result is the structure of the performance net. Each state corresponds to a system context, and each transition corresponds to a system operation. This sequence is called a 'process'. If the observable behaviour can be recorded in a complete ...-> (state) -> [transition] -> (state) ->... process description, then the analysis and construction of the net structure of this process are simple: you have only to count the number of all different states and transitions used, or to mark on a list the frequencies of each state and transition used in the process. The aim of the 'folding' operation is to reduce the elements of an observed empirical interaction process to the minimum number of states and transitions, with the reduced number of elements being the 'logical decision structure'. Folding an interactive process extracts the embedded net structure and neglects the information of the amount of repetition and of the sequential order ('time structure').

To measure complexity we use the C_{cycle} metrics of McCabe (1976). With C_{cycle} we have a useful quantitative metric to measure complexity. We are discussing the advantages and disadvantages of four different quantitative metrics in the context of an empirical investigation elsewhere (see Rauterberg, 1992a). The complexity measured with C_{cycle} is defined by the difference of the total number of connections (T: transition) and the total number of states (S: state). The parameter P is a constant to correct the result of Formula 1 in the case of a sequence ($T – S = –1$); the value of P in our context is 1. $C_{cycle} = T – S + P$ with $T \geq (S-1)$. But, what could C_{cycle} mean? McCabe interprets C_{cycle} as the *number of linear independent paths* through the net. Other interpretations of C_{cycle} are *number of holes* in a net or *number of alternative decisions* carried out by the users.

We call the complexity of the observable behaviour the 'behavioural complexity (BC)'. This behavioural complexity can be estimated by analysing the recorded concrete task solving process, which leads to an appropriate task solving solution. The complexity of a given tool (e.g. an interactive system) we call 'system complexity (SC)'. The last parameter we need is an estimation of the 'task complexity (TC)'. The necessary task solving knowledge for a given task is constant. This knowledge embedded in the cognitive structure (CC) can be observed and measured with BC. If the cognitive structure is too simple, then the concrete task solving process must be filled up with a lot of heuristics or trial and error strategies. Learning how to solve a specific task with a given system means that BC decreases (to a minimum = TC) and CC increases (to a maximum = SC). We assume, that the difference (BC–TC) is equal to the difference (SC–CC).

To solve a task, a person needs knowledge about the dialogue structure of the interactive software (measured by SC) and about the task structure (measured by TC). SC is an upper limit for TC ($SC \geq TC$); this means, that the system structure constrains the complexity of the observable task solving space. Now we can state with the constraints ($BC \geq TC$) and ($SC \geq CC$), that: $BC – TC = SC – CC$. To get CC we transform in: $CC = SC + TC – BC$. The parameters SC and TC can be estimated either in a theoretical or in an empirical way. The parameter SC is given by the concrete system structure, so we have to apply a given complexity measure to this system structure (theoretical way). The empirical way to estimate SC is to take the *maximum* of all observed BCs per task. To calculate TC, all a priori descriptions of task structures are usable (theoretical way). If we have empirical data of different task solving processes of different persons, then we can estimate TC using the *minimum* per task of all observed BCs. Given a sample of different complete task solving processes, the best approximation for TC seems to be the minimal solution regarding a specific complexity measurement. One plausible consequence of this assumption is that CC is equal to SC in the case of 'best solution' ($TC= BC$).

A LEARNING EXPERIMENT

Method

Subjects. Six users (6 men; average age of 25 ±3 years) had to solve one task in two different versions on three weeks in a row with exactly seven days between each of the three task solving sessions.

Tasks. The dialog system of a commercial database management system (DBMS) with a character-oriented user-interface (CUI) and a hierarchical menu structure was the test system (for a detailed description see Rauterberg, 1992b). To solve the task, a list must be generated with four different data fields that have to be interactively selected. One data field must be calculated with a set of predefined calculation instructions stored in a disk-file. This list must be outputted to the screen and to the printer. The difference between task-1 and task-1' was only on the formulation level of the instruction, but on the structural level all six tasks were identically.

Procedure. The duration of the actual task solving session was about 15 minutes. At the first session each user had to fill out a questionnaire about his/her general computer experiences and his/her specific experiences with DBMS's and CUI's. Each user was carefully instructed to operate the DBMS, especially to solve the test task (23 ±4 min of instruction time). At the beginning of each task solving session a short questionnaire about the correct task solving process must be answered. During the task solving session each keystroke with a time stamp was recorded in a logfile. Each user needed 68 ±12 minutes for the whole experiment (each session was carried out as follows: two similar tasks, non speed condition, individual sessions, session no. = week no.).

Measures. We measured three different aspects: (1) once the general experience with computer and DBMS (the long-term knowledge), (2) three times the actual knowledge about the task and the tool (the short-term knowledge per week), and (3) six times the task solving time and BC (performance measures per task).

Long-term knowledge. The questionnaire to measure the general computer and DBMS experience consists of twelve single questions. For each question the user has to estimate the number of hours spending on the asked topic in his/her life.

Short-term knowledge. The questionnaire to measure the actual knowledge about the task-tool mapping consists of three lists with 24 DBMS-functions. The user has to solve three sub-tasks by numbering all appropriate functions in the correct sequential order. The correspondence of the answered sequence with the correct solution was calculated with a *similarity ratio* (range 0…1; see Rauterberg, 1995).

Complexity. Based on the logfiles we could measure BC with our analysing tool AMME (cf. Rauterberg, 1993). To estimate CC we have to determine SC and TC. SC can be estimated as the maximum of all observed BC's (e.g., SC = 28). TC can be estimated as the minimum of all observed BC's (e.g., TC = 22; see Figure 1).

Results

Long-term knowledge. On average 2698 ±2534 hrs of general experience with computer technology was measured (including 274 ±601 hrs of DBMS experience, and 1693 ±2112 hrs CUI experience). The product moment correlation between DBMS experience and general experience is $r = .43$ ($p \leq 0.42$, N=6), and between CUI experience and general experience is $r = .85$ ($p \leq 0.03$, N=6).

Short-term knowledge. The similarity ratio as a measure of the actual knowledge about the task-tool mapping ('task tool knowledge') was neither significantly different between the three weeks ($\text{mean}_{\text{week-1}} = 0.89 \pm 0.13$, $\text{mean}_{\text{week-2}} = 0.84 \pm 0.16$, $\text{mean}_{\text{week-3}} = 0.82 \pm 0.22$, $F(2,15) = 0.39$, $p \leq .685$), nor between the three sub-tasks ($\text{mean}_{\text{subtask-A}} = 0.85 \pm 0.16$, $\text{mean}_{\text{subtask-B}} = 0.80 \pm 0.18$, $\text{mean}_{\text{subtask-C}} = 0.90 \pm 0.17$, $F(1,30) = 2.40$, $p \leq .108$). No significant interaction term exists ($F(4,30) = 0.35$, $p \leq .840$).

Task solving time.The difference of the task solving time between task-1 and task-1' is significant ($\text{mean}_{\text{task-1}} = 9.5 \pm 3.8$ min, $\text{mean}_{\text{task-1'}} = 5.2 \pm 0.9$ min, $F(1,10) = 15.3$, $p \leq .003$. There is also a significant difference of the task solving time between the three weeks ($\text{mean}_{\text{week-1}} = 8.6 \pm 4.7$ min, $\text{mean}_{\text{week-2}} = 7.2 \pm 2.9$ min, $\text{mean}_{\text{week-3}} = 6.3 \pm 2.0$ min, $F(2,20) = 3.6$, $p \leq .045$). The interaction term is nearly significant ($F(2,20) = 3.2$, $p \leq .061$). The decline of the task solving time for task-1 over the three weeks corresponds to the well-known learning curve. But the nearly significant interaction term can be interpreted as if all users learned the correct solution already for task-1' in the first session.

Behavioural complexity BC. The difference of BC between task-1 and task-1' is significant ($\text{mean}_{\text{task-1}} = 24.3 \pm 1.7$, $\text{mean}_{\text{task-1'}} = 22.6 \pm 1.4$, $F(1,10) = 12.7$, $p \leq .005$; see Figure 1). All other differences are not significant (see Figure 1). Except one, all users had already a BC of 23 in task-1' of the first session. This result corresponds with the result of the task solving time. All users reached their minimum of BC = 22 in task-1' of the second session. It is important to note that from that moment the variance disappears.

Cognitive complexity CC. CC is exactly with $r = -1.0$ negatively correlated with BC. Therefore, only the difference of CC between task-1 and task-1' is significant, too ($\text{mean}_{\text{task-1}} = 25.7 \pm 1.7$, $\text{mean}_{\text{task-1'}} = 27.4 \pm 1.4$, $F(1,10) = 12.7$, $p \leq .005$). All other differences are not significant (see Figure 1). All users reached their maximum of CC = 28 in task-1' of the second week.

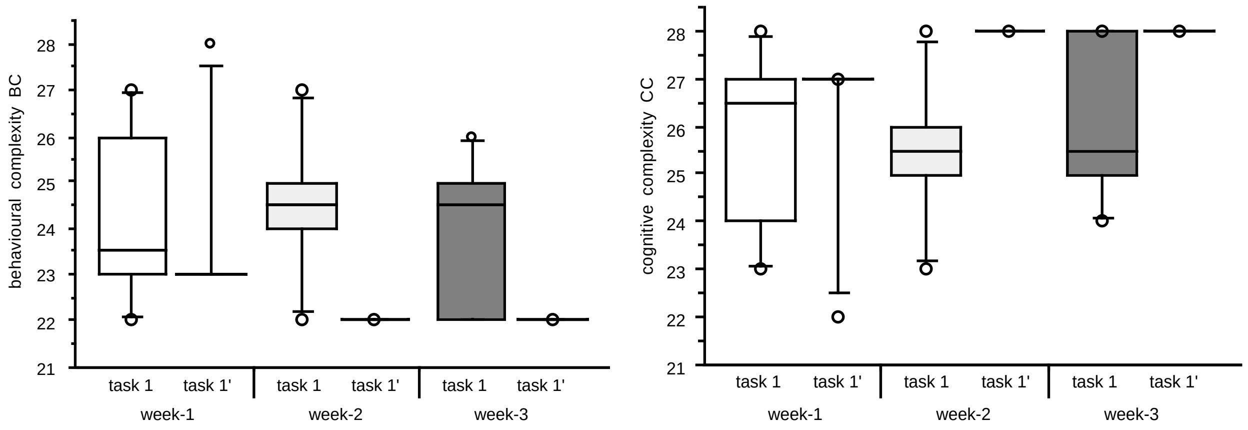

Figure 1. Box plots of the behavioural complexity BC and of the cognitive complexity CC.

CONCLUSION

Three important conclusions can be drawn: (1) Learning how to solve a specific task with a given system means that behavioural complexity (BC) decreases and cognitive complexity (CC) increases. (2) After an exploratory phase all users reached the correct and minimal task solution (minimum of BC = 22, see Figure 1), and the variance between users disappeared. Furthermore, (3) we can conclude from the empirical results that we must discriminate between the *time structure* (Johnson, 1985) and the *logical structure* (Rauterberg, 1995) of a task. The time structure can be measured with the task solving time. The logical structure of a task can be extracted with the special analysing tool AMME (Rauterberg, 1993), and the complexity of this logical structure can be measured with the McCabe-measure (BC) (McCabe, 1976). At week-2 all users learned completely the logical task structure (minimum of BC, no variance, see Figure 1) although the task solving time decreases further on. Learning the time structure means to accelerate the task solving process. One psychological interpretation of the acceleration of the time structure seems to be--if the correct task solving sequence was learned--that self-confidence grows and comes strong in doing the right things at the right time.

REFERENCES

S.L. JOHNSON, "Using mathematical models of the learning curve in training system design", in: *Proceedings of the Human Factors Society 29th Annual Meeting*, Vol. II, 1985, pp. 735-739.

T. MCCABE, "A complexity measure", *IEEE Transactions on Software Engineering*, Vol. SE-2, 1976, pp. 308-320.

C.A. PETRI, "Introduction to general net theory", *Lecture Notes in Computer Science*, Vol. 84, 1980, pp. 1-19.

M. RAUTERBERG, "A method of a quantitative measurement of cognitive complexity", in: G.C. van der Veer, M.J. Tauber, S. Bagnara and A. Antalovits, Eds., *Human-Computer Interaction: Tasks and Organisation* (CUD, Roma, 1992a, pp. 295-307).

M. RAUTERBERG, "An empirical comparison of menu-selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts", *Behaviour and Information Technology*, Vol. 11, No. 4, 1992b, pp. 227-236.

M. RAUTERBERG, "AMME: an automatic mental model evaluation to analyze user behaviour traced in a finite, discrete state space", *Ergonomics*, Vol. 36, No. 11, 1993, pp. 1369-1380.

M. RAUTERBERG, "From novice to expert decision behaviour: a qualitative modelling approach with Petri nets", in: Y. Anzai, K. Ogawa & H. Mori (Eds.) Symbiosis of Human and Artifact: Human and Social Aspects of Human-Computer Interaction. (Advances in Human Factors/Ergonomics, Vol. 20B, pp. 449-454), Amsterdam: Elsevier.