Discriminating Task Solving Strategies Using Statistical and Analytical Methods

Samuel Schluep

Institute of Hygiene and Applied Physiology
Swiss Federal Institute of Technology
Clausiusstr. 25, CH-8092 Zurich
schluep@iha.bepr.ethz.ch
www.iha.bepr.ethz.ch/pages/leute/schluep/schluep.htm

Morten Fjeld

Institute of Hygiene and Applied Physiology
Swiss Federal Institute of Technology
Clausiusstr. 25, CH-8092 Zurich
fjeld@iha.bepr.ethz.ch
www.iha.bepr.ethz.ch/pages/leute/fjeld/fjeld.htm

Matthias Rauterberg

Center for Research on User-System Interaction
Technical University Eindhoven
NL-5612 AZ Eindhoven
rauterberg@ipo.tue.nl
www.tue.nl/ipo/

ABSTRACT

We have recorded the behaviour of several users solving the same task with an interactive database program. Since the number of users exceeds that of strategies, multiple users will have a strategy in common. Our aim is to find groups of users sharing the same strategy. Following each of three methods (correlation, intersection and exclusion) we define a comparative metric among task solving behaviour. For multiple users, we represent these measures by a matrix system, to find groups of users with common strategies. Multidimensional scaling or analytic interpretation indicates distinct user groups.

Keywords

automatic plan recognition, task solving strategies, user behaviour

INTRODUCTION

The aim of this paper is to develop automatic methods finding task solving strategies. Such knowledge is of interest to understand how users behave in a newly designed system, and to thereby give them better support. Also, it may also help understanding how expert users behave in highly complex systems.

Under certain conditions, strategies may also be obtained by protocol analysis (Ericsson and Simon, 1984). Protocol analysis implies manual inspection of video and verbal utterances in addition to log files. With simple tasks, this work can be overcome. For more complex tasks, protocol analysis has proved cumbersome. Semi-automatic generation of process models was studied by Ritter and Larkin (1994). Guided by their work, further principles for automatic recognition of user strategies and plans will be suggested.

In general, a lot of task solving behaviour that is not strictly *task related* can be observed. It is hardly possible to single out the successful *strategy* from the *remaining behaviour*. One approach may be to study many users solving the same task. Since they all solve the same problem, it is likely that their common behaviour is what was required to solve the task.

A *strategy* is defined to be one (of many), possibly error free, successful task solving behavioural sequences for the current system and task. As soon as a complete strategy is accomplished, task solving is over. If users follow different strategies, a group of users may have one strategy in common, an other group a second one.

The *remaining behaviour* contains much information of how the successful strategy is developed. Unsuccessful trials, exploring the capabilities of the applied tools, repetitions and retries are components of a learning process, which may give insights to the cognitive structure of the knowledge about the computer system and the task.

We will investigate measures which relate task solving behaviour among one another and their suitability for grouping users with a common strategy.

In this paper the aim is to develop automatic methods purely based on observable task solving behaviour, applicable with simple as well as with complex tasks. Protocol analysis, human perception and verbalisation will only be used to validate the elaborated, automatic methods.

TASK DOMAIN AND STRATEGIES

An empirical investigation was carried out by Rauterberg (1992) to compare different types of user interfaces. Users had to solve different benchmark tasks with a character-based user interface (CUI) and a desktop user interface (graphical user interface, GUI) of a relational data base program. User behaviour was recorded with log files. In the present paper we use log files of 6 novice {N1,...,N6} and 6 expert {E1,...,E6} users solving the first task on the CUI system.

The task was to find out how many data records there are in a database consisting of files A, B and C. After the required results were found, task solving activity was finished. The goal was to operate the system, so the number of records for each of the files was displayed on the screen and could be read. Three basically different strategies could be used to reach this goal:

Strategy 1 (S1)

From the main menu the user goes to the 'Info' menu. There she or he selects the function 'Datei' by pressing key [d], which displays the number of records and other general information of the 3 database files A, B and C. The total number of lines that are output exceeds the size of the screen, so the first line indicating the number of records of file A is scrolled up and cannot be read. Therefore the user must halt the display by pressing the blank key, read the number of records of file A and continue the display process by pressing any key. Afterwards the requested data for file B and C can be read from the screen. This strategy was used by users N1, N4, N5, N6, E4 and E6.

Strategy 2 (S2)

From the main menu the user goes to the menu 'Liste' to generate a list. Then one of the database files A, B or C is selected. With the function 'Definition' the user defines which data fields shall be listed. After leaving the definition mode the function 'Erstellen' generates the list and displays it to the screen. The program automatically adds the number of records to the end of the list. The user reads the requested data and repeats the procedure with the other database files. This strategy was used by users N2, N3, E1, E2, E5.

Strategy 3 (S3)

From the main menu the user goes to the menu 'Daten' and selects the function 'Ausgabe' by pressing key [a], which is used to find specific data records. The user then enters a question-mark as search criteria, which is used as wild-card character. Pressing the enter key starts the search. As a result the program displays the first record which matches the search criteria and indicates at the bottom line the total number of matching records. Since all records match the wild-card criteria, this number equals the total number of records of file A. The user selects the next database file and repeats the procedure with files B and C. Only user E3 used this strategy.

Remaining Behaviour

All users in the experiment succeeded in solving the task. The effort measured by the number of keystrokes varied among the users, even if they applied the same strategy (Table 1). This indicates that the amount of their remaining behaviour varied among users, since the number of keystrokes is assumed to be constant for each strategy.

Table 1: Bandwidth of the number of keystrokes related to the strategies

Strategy	*User*	*# Keystrokes* *min.*	*max.*
S1	N1, N4, N5, N6, E4, E6	10	71
S2	N2, N3, E1, E2, E5	33	167
S3	E3	73	73

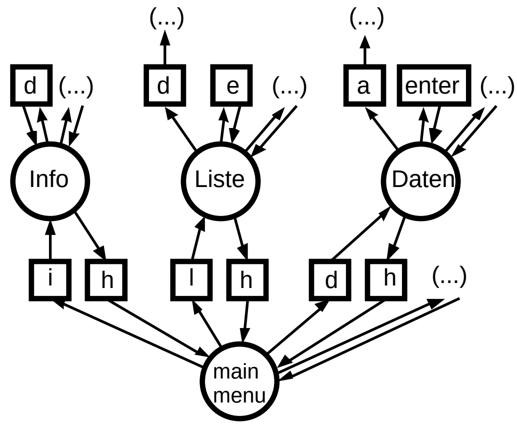
The remaining behaviour can be described as follows:

- *Correction of mistakes.*
- *Retries after unsuccessful trials with or without any changes of the action sequence.*
- *Repetitions.*
- *Exploratory behaviour to find some hint or trying functions which seem to be promising.*
- *Orientation, e.g. determining which mode is active or how to get back to the main menu.*
- *Attempts to apply other strategies, which were abandoned later on.*

DESCRIPTION OF TASK SOLVING BEHAVIOUR

The relational database program can be modelled as a system with 154 dialogue states and transitions which symbolise the possible succession of two states and the action associated with it. Figure 1 shows a fraction of the state transition model of the program. Circles represent dialogue states, squares represent transitions. The letter shown inside the squares indicates the key the user must press to activate the transition and go from one state to another. The same key might be used in different transitions. Specifying the preceding state and the action (i.e. the key to be pressed) identifies a transition unequivocally.

Figure 1: State transition model



User behaviour is represented as a sequence of transitions between system states. A *state-transition-vector* (STV, Formula 1) summarises a subject's task solving behaviour for one task. Each STV element represents the number of activations of a certain transition the user needed to solve the task.

$$\{e_i^p\} \qquad (1)$$

$e_i^p \in N_0$ *STV component*

$p \in \{N_1, ..., N_6, E_1, ..., E_6\}$ *user index*

$i = 1, ..., n$ *transition index*

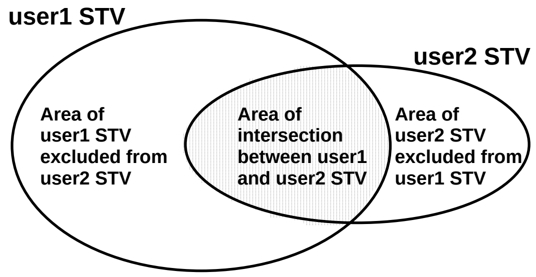
$n = 978$ *number of transitions*

Since the order of activated transitions is not contained in the STV, the order of user actions is only partly conserved. It is stored implicitly, given by the potential sequences of actions which are possible by the system dialogue structure.

METHODOLOGY

The classical method to compare the task solving behaviour of two users is correlation, which is a measure of proximity. Analytical methods offer an alternative. In Figure 2 each ellipse illustrates the set of transitions activated by a user. The intersection area represents similarity (what do two behavioural sequences have in common), the exclusion areas difference (what two behavioural sequences do not have in common), between two STVs.

Figure 2: Intersection and exclusion between user 1 and user 2.



For each comparison method, we elaborate a metric (Table 2). The metric may be symmetrical or asymmetrical (the metric applied from user 1 to user 2 is or is not the same as the metric applied from user 2 to user 1). Application of the metric between all users results in a matrix. The matrix is then analysed by a grouping algorithm.

Table 2: The three methods, metrics, metric nature and grouping algorithms

Method /metric	*Metric order*	*Metric nature*	*Grouping algorithm*
Correlation	CORR	Statistical	Statistical
Intersection	$M_{p,q}^{IS}$, $M_{p,q}^{BIS}$	Analytical	Statistical
Exclusion	$M_{p,q}^{EX}$	Analytical	Analytical

CORR means Pearson correlation; the other metrics are defined by Formulas 2, 3 and 4.

CORRELATION METHOD

Correlation yields proximity of user behaviour. These values are analysed by multidimensional scaling (MDS, Systat, 1989) to indicate groups of users with resembling task solving strategies.

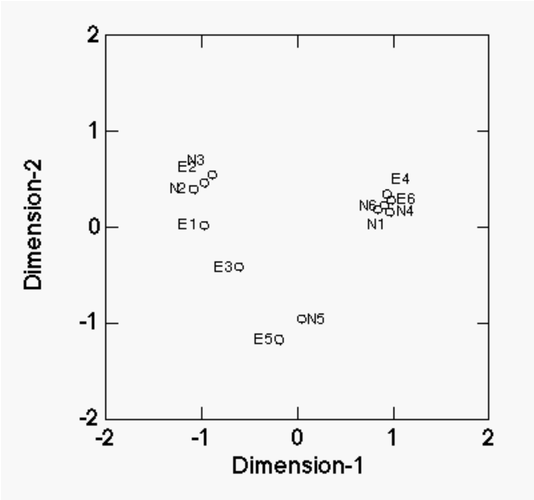
Metric

A Pearson mxm (m=12) correlation matrix relates all STVs with each other and has values between -0.003 (low proximity) and 0.948 (high proximity).

Grouping Algorithm

The correlation matrix interpreted by MDS (dimensions: 2; r-metric: 1; iteration: 50; convergence: 0.005; loss function: Kruskal, regression: monotonic) gives Figure 3, allowing for visual interpretation. The users may be grouped as follows: {N1, N4, N6, E4, E6}, {N2, N3, E1, E2} and {N5, E5}. It is not clear whether E3 should belong to the group {N2, N3, E1, E2} or if it constitutes a separate group. By proportion of variance (RSQ = 0.895), MDS explains some but not all the user data variance.

Figure 3: MDS plot with Pearson correlation matrix gives RSQ = 0.895.



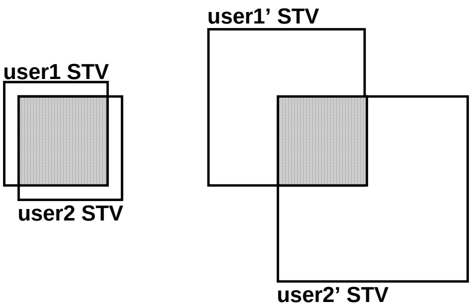
INTERSECTION METHOD

If two users follow the same strategy, the strategy will be within their intersection area which represents similarity.

Metric

Similar behaviour is measured by summing up the smaller STV element values of the two user STVs, thus considering the activation of transitions common to both users.

Figure 4: User1 and user2 STVs represented by rectangles.



In both situations, the intersection is the same, but the similarity between STVs are not. This indicates that a normalisation is required.

User1 and user2 STV can also be seen as sets, represented by *rectangles* (Figure 4). For the intersection area (measured quantity) to be a valid measure of similarity (desired quality), a normalisation is required. It is possible to scale degree of intersection by the larger (max.), the average (mean) or the smaller (min.) sum of the intersection areas. Scaling by the smaller of the areas corresponds to scaling by the maximum possible intersection. Expressed in the state-transition vector space gives the intersection metric of Formula 2.

$$M_{p,q}^{IS} = \frac{\sum_{i=1}^{n} \min\left(e_i^p, e_i^q\right)}{\min\left(\sum_{i=1}^{n} e_i^p, \sum_{i=1}^{n} e_i^q\right)} \qquad (2)$$

$p, q \in \{N_1, ..., N_6, E_1, ..., E_6\}, p \neq q$ *user indices*

These similarity values lie in the range of 0.078 (low similarity) and 0.929 (high similarity).

Ignoring repetitive behaviour is a means to reduce complexity. Replacing each STV component >1, by 1, results in a *binary* state-transition-vector. An intersection metric based on this vector is given by Formula 3. When comparing user behaviour with this metric, the similarity values lie in the range from 0.182 (low similarity) to 0.882 (high similarity).

$$M_{p,q}^{BIS} = \frac{\sum_{i=1}^{n} \min\left(e_i^p \cdot e_i^q, 1\right)}{\min\left(\sum_{i=1}^{n} \min\left(e_i^p, 1\right), \sum_{i=1}^{n} \min\left(e_i^q, 1\right)\right)} \qquad (3)$$

$p, q \in \{N_1, ..., N_6, E_1, ..., E_6\}, p \neq q$ *user indices*

Grouping Algorithm

Interpreting the *normalised intersection matrix* and the *binary normalised intersection matrix* by MDS (dimensions: 2; r-metric: 1; iteration: 50; convergence: 0.005; loss function: Kruskal, regression: monotonic), the users seem to represent three groups, {N2, N3, E1, E2, E5} and {N1, N4, N5, N6, E4, E6} and {E3} (Figure 5 and 6). According to the RSQ = 0.977 / 0.995 we can explain most of the variance within the user data.

Figure 5: MDS with a normalised intersection matrix gives RSQ = 0.977.

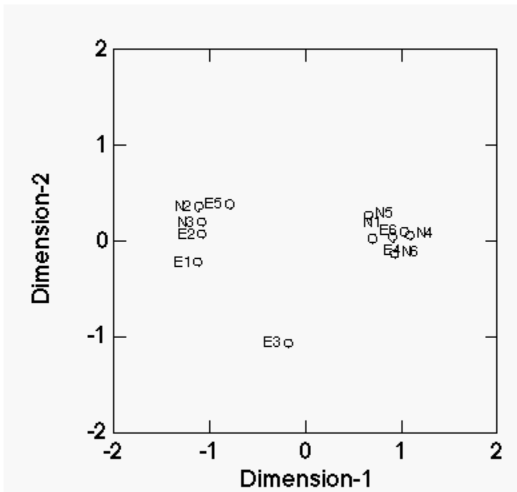
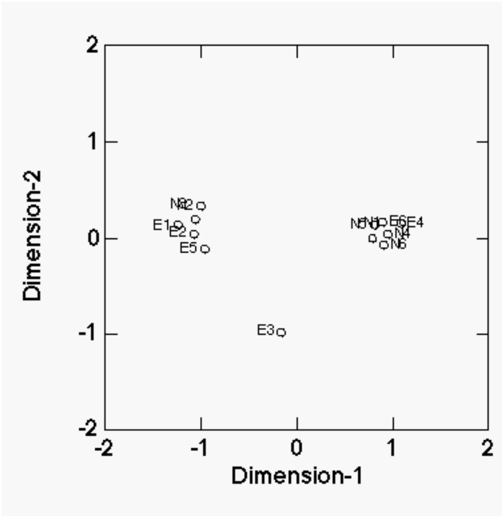


Figure 6: MDS with a binary normalised intersection matrix gives RSQ=0.995.



EXCLUSION METHOD

The exclusion method quantifies differing behaviour. Little difference between user behaviour might indicate that the users apply the same strategy. The difference is measured as the amount of one user's behaviour not shared by the other user. This measure is not symmetric since the excluded behaviour of two users is not identical (see Figure 2). So exclusion gives an asymmetric matrix of distances between users, calling for another grouping algorithm than MDS.

Metric

This method measures the difference between two STVs by estimating how much of one STV (column index in Table 3) is excluded from a second one (row index), giving an m×m (m=12) asymmetric exclusion matrix (Table 3).

$$M_{p,q}^{EX} = \sum_{i=1}^{n} \left| \min\left(e_i^p - e_i^q, 0\right) \right| \qquad (4)$$

$$p, q \in \{N_1, ..., N_6, E_1, ..., E_6\}$$

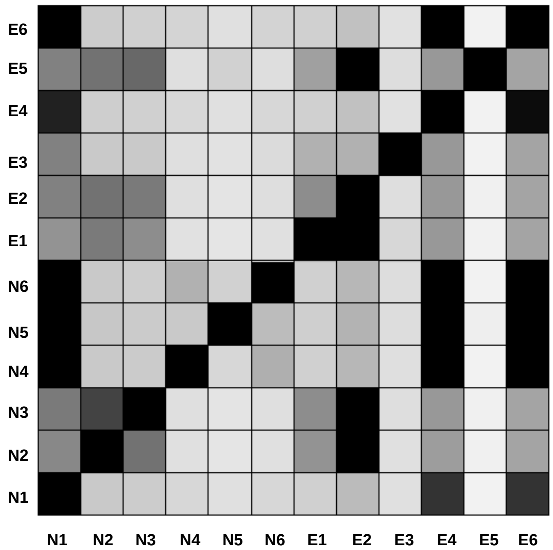
Table 3: Numerical representation of exclusion matrix.

E6	6	43	47	51	70	50	47	35	73	5	171	0
E5	17	15	14	69	48	67	23	7	64	21	0	24
E4	9	44	47	56	70	55	47	35	73	0	171	8
E3	17	41	41	68	77	62	28	28	0	21	162	24
E2	17	15	16	68	81	67	19	0	66	21	143	24
E1	20	16	19	73	85	72	0	7	54	21	147	24
N6	3	41	44	28	48	0	47	30	63	4	166	2
N5	3	39	42	41	0	33	45	29	63	4	132	7
N4	2	41	42	0	55	27	47	30	68	4	167	2
N3	16	11	0	68	82	69	19	4	67	21	138	24
N2	18	0	15	71	83	70	20	7	71	22	143	24
N1	0	41	43	55	70	55	47	32	70	10	168	10
	N1	**N2**	**N3**	**N4**	**N5**	**N6**	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**

Grouping Algorithm

The grey-scale representation (Figure 7) is a visualisation of Table 3. Diagonal elements of Table 3 were directly mapped to the darkest grey-tone. Figure 7 shows to what degree a row user STV excludes a column user STV. Darker matrix elements correspond to lower degree of exclusion.

Figure 7: Grey-scale representation of exclusion matrix. Darker elements mean more exclusion of column STV from row STV.



Ignoring diagonal elements, small exclusion values (Table 3) indicate similarity, and we can derive four similarity relations (Table 4). The first three relations are interrelated, giving one group. Relation four gives a second group. One user is not related to any other user representing a third group. So three groups result: {N1, N4, N5, N6, E4, E6}, {N2, N3, E1, E2, E5} and {E3}.

Table 4: We can derive these four similarity relations.

Relation	*User STVs of each relation*
1	N1 is part of N4, N5, N6, E6
2	E4 is part of N4, N5, N6, E6
3	E6 is part of N4, N5, N6, E4
4	E2 is part of N2, N3, E1, E5

DISCUSSION

For method validation, we did a task based protocol analysis (Ericsson and Simon, 1984), grouping the users according to three distinct strategies (see Table 1).

The correlation method does not correspond fully to the protocol analysis, whereas the intersection and the exclusion method give the same grouping result.

The correlation metric differs from the intersection and exclusion metric particularly by the product of transition activations in the numerator of the formula. A high number of activations of the same transition strongly influences the weight of this transition. This is due to the quadratic quality the correlation metric. Users N5 and E5 illustrate this effect. To solve the task, they applied different strategies, but the correlation method puts them near to one another. Looking closer at their behaviour, it could be seen that both first tried to count the number of records manually by stepping through the whole file. This results in a high number of activations of the ‘next record’ transition, leading to a high correlation coefficient. However, this strategy was abandoned later on because of the large number of records.

The intersection method clearly groups the users according to their strategies. Contrary to the correlation method, remaining behaviour does not seem to have much influence on the grouping.

The exclusion method does not say anything about possible combinations or parts of strategies applied. For such questions, the statistically based methods seem more relevant. This is confirmed by Hanson et. al. (1991), treating class (or: group) assignment with Bayesian methods: "Such classes are also 'fuzzy'; instead of each case being assigned to a class, a case has a probability of being a member of each of the different classes".

CONCLUSION

For the present combination of system, task and user behaviour, it was possible to develop methods grouping users according to their task solving strategy.

FUTURE PERSPECTIVES

Results for one task only were acquired. To make the methods more reliable, it is necessary to evaluate several tasks. For each task the methods will be validated by manual protocol analysis. It is of particular interest to find out if the methods perform with other, more complex tasks.

It is also planned to study learning experiments, in order to recognise the acquisition process of strategies.

REFERENCES

Ericsson, K. A. and Simon H. A. (1984). *Protocol Analysis, Verbal Reports as Data*. Cambridge, Mass.: MIT Press.

Hanson, R., Stutz, J., and Cheeseman, P. (1991). Bayesian Classification Theory. Technical Report FIA-90-12-7-01, NASA, Ames Research Center, AI Branch.

Rauterberg, M. (1992). An empirical comparison of menu selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts. *Behaviour and Information Technology 11*, 227-236.

Ritter, F. E., and Larkin, J. H. (1994). Developing process models as summaries of HCI action sequences. *Human Computer Interaction 9*, 345-383.

SYSTAT Inc. (1989). SYSTAT®: The system for statistics. *SYSTAT program version 7.0.1 for PC.* 93-166.