Chapter 5
Visual, analytic measures of the predictions' fit to the data

This chapter describes a family of graphic displays for analyzing the interpretation of the data with respect to the model. These displays are designed to support the requirements listed in Table 5-17. While many measures presented in Chapter 2 provide useful starting points for analyses, they fail to be completely adequate for analyzing models and data the size we wish to consider, and they fail to summarize the comparison in terms of the model. Because it can present a large amount of information clearly, a graphical approach is better. Three approaches are used.

First, a version of the operator support display invented by Peck and John (1992) is automated. It shows which model actions were supported by data, and their position within the model. Given the aligned data and model actions in the Spa-mode spreadsheet, this display of the support for each operator can be created automatically by the analyst. The analyst can click on the correspondences in the display to learn more about them.

Second, a graphic display that presents the relative processing rate of the model and the subject is provided. Given the aligned data and model actions in the Spa-mode spreadsheet, it too can be created automatically to show the time course of where the model's predictions and the data do and do not correspond because the two have performed different amounts of processing. Here too the analyst is provided with associated information by clicking on the data points, and can find the actions that take disproportionate time and, presumably, effort.

Finally, an environment is provided to assist in editing and designing additional versions of these displays. More displays will be necessary. What is an appropriate display may vary with the model, and there are many ways for the data to not match the model. Several other approaches can now be imagined for displaying the interpretation of the data with respect to the model.

Table 5-17: Requirements supported by the graphic comparison displays and S-mode.

<u>Requirements for analyzing the comparison of the data with the model's predictions.</u>
(a) Show where the data does not match the predictions.
(b) Aggregate the results of the comparison in terms of the model.
(c) Interpret the test results as clues for modifying the model.
<u>Requirements based on integrating the steps and supporting TBPA with a computational environment.</u>
(a) Provide consistent representations and functionality based on the architecture.
(b) The environment must automate what it can.
To support the user for the rest of the task:
(c) Provide a uniform interface including a path to expertise.
(d) Provide general tools and a macro language.
(e) Provide tools for displaying and manipulating large amounts of data.

5.1 Creating the operator support display automatically

The operator support display of Peck and John (1992), previously presented in Figure 2-7, provides a display relating the model's trace to the problem spaces and their hierarchical organization. The indentation of the operators are ordered hierarchically and in order of use during a typical episode. This display, while it was not designed to do so, also supports the requirement to understand the model. The line connecting each trace element begins to show some of the regularities and periodicities in the model. More importantly, however, it shows which model actions are supported by data, and the type of data they are supported by. It also shows where the model's predictions are not matched by data and data that are not predicted. Note, however, that the sequential order of the data is not shown in this display.

The original version of the operator support display took an 8-hour day to construct by hand (Peck, 1992). Figure 5-20 shows the automated version that can be created in minutes from the alignment data in the alignment spreadsheet. There have been a few additions and deletions between Peck and John's display and this version. The automated version presents the course of behavior from left to right, rather than top to bottom. This lets the operator names that are matched to the data to be presented in a more readable form, with complete names, and there is room to indent them to represent their organization by problem space. The automatically created version, when it is presented on-line, is interactive: the analyst can click on each data point or model action to see the other fields of data (such as the verbal utterance) of the actions making up the correspondence. Peck and John (1992) included a summary of the support for each model action and data segment in a column on the margin; the current version does not yet, but should.

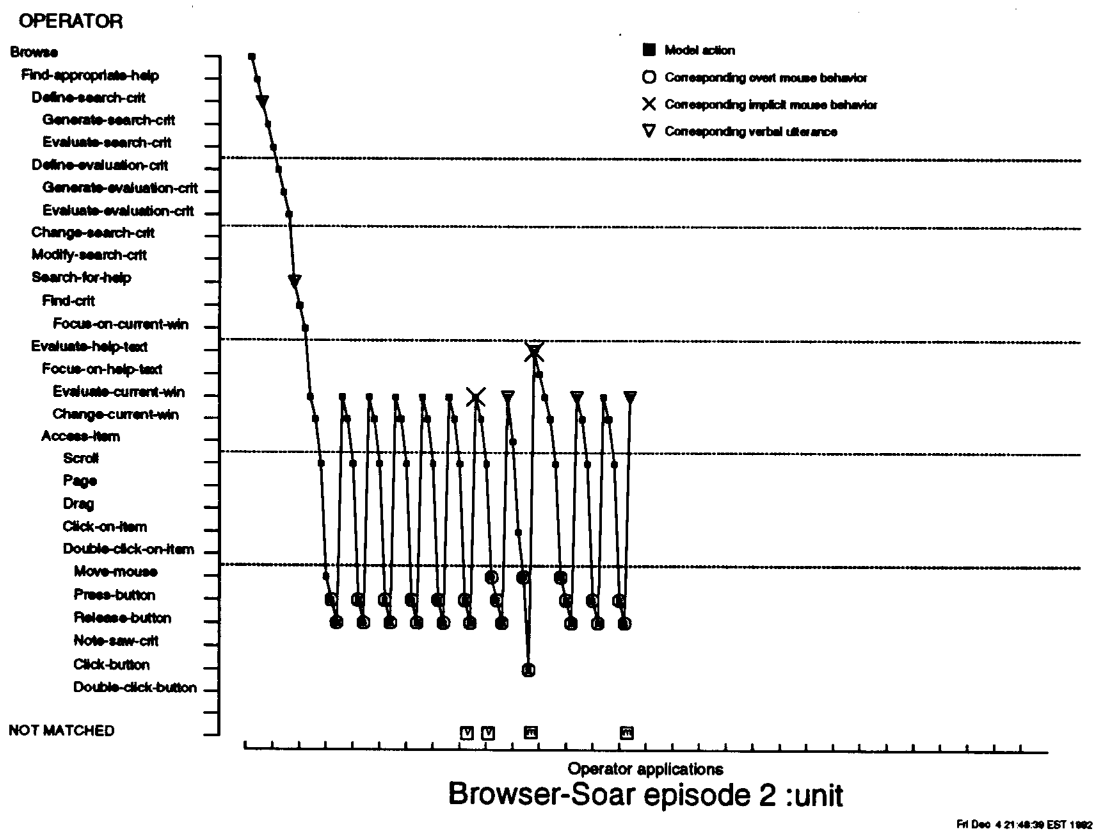

Figure 5-20:
Example operator prediction support display taken from the Unit episode of Browser-Soar. The model's operators are shown on the left-hand side, indented according to their depth in the problem space hierarchy. The connected black squares represent the model's performance. Corresponding data are represented by overlapping symbols. Unmatched data are placed at the bottom of the display as if it matched the *Not matched* operator.

5.2 Understanding the relative processing rate

In order to improve the model the analyst needs to see the global patterns of where the model's predictions do and do not match the data. One way to further understand the sequential predictions of the model would be to view the interpretation of the data with respect to time. A display showing the types of the correspondences (taken from Table 2-5) and the time each action occurred in their respective information streams would emphasize the sequential nature of the model's predictions and the data, the relative processing rate of each, and highlight sections of behavior that could be brought into closer correspondence. This display was initially motivated by the difficulty of understanding the relative rate of actions between the model and subject as they were depicted in the spreadsheet, but seeing that this relationship existed and could perhaps be more clearly displayed.

The order and temporal location of the correspondences between the model's actions and the data can be presented in a display showing their relationship to each other through time. Sakoe and Chiba (1978) first did this for doing speech recognition, matching a model of speech production against the actual recorded speech. Figure 5-21 shows an interpretation of their figure. Their display (and the matching process that generated it) required that each model prediction match a data point, and while it could admit noise, the process did not permit fundamentally wrong actions to be excluded, or an out-of-order match to be included. Cognitive models are usually not yet accurate enough to assume that every action in the model will have a one-to-one match to the data like they assume; multiple subject's actions may match a single model action, and some model actions may not be supported by data. But their display inspires a similar display with a warping function with loosened requirements on the match and with an augmented representation.

5.2.1 A display for comparing the relative processing rate

Figure 5-22 shows a chronometric fit display similar to Sakoe and Chiba's (1978) that presents the warping function for a cognitive process model. Each graphical element (shown in a legend in the upper left) represents a pair of corresponding model and subject actions. The current display presents the correspondences in order that the subject's actions occurred. A similar display presenting them in order of the model's actions could also be created. This display does not include the restriction of one-to-one matches, but allows model actions to match multiple subject actions (it would even support matching multiple model actions to a single data point, but as noted in the review, subject segments should match only one model action, or else they may not be segments).

This display presents the time of the corresponding subject actions on the x-axis in seconds, and the time of the corresponding model actions on the y-axis in terms of decision cycles. The time for both model and subject begin with the first match. Their relationship before the first correspondence cannot be computed. It is possible for later subject actions to match earlier model actions if the subject does not report all actions in order, or if different modalities are reported with different amounts of lag. Unmatched subject actions are unconnected, and put at the bottom of the display along with a label indicating the type of information that was not matched. They are positioned on the subject's time axis with the time they occurred. Overt actions of the model, such as mouse movements, that are not matched, are represented in a similar manner. Unmatched, non-overt model actions, such as internal state transitions, are not displayed. They will not necessarily be matched, so their lack of correspondence with data tells us less according to our theory of measurement (Ericsson & Simon, 1984), and it is best viewed with Peck and John's display of operator support.

This display provides many of the criteria for measures of model fit noted in Chapter 2. Better experiments are favored, for they provide a larger or more informationally dense display of the model's predictions fit to the data set. These denser displays provide more information on how to improve the model, and should be more persuasive. Typical mismatches produce signature patterns on this display; these are noted below.

Identifying outliers: Computing a linear regression on the match. In Figure 5-22 a least squares

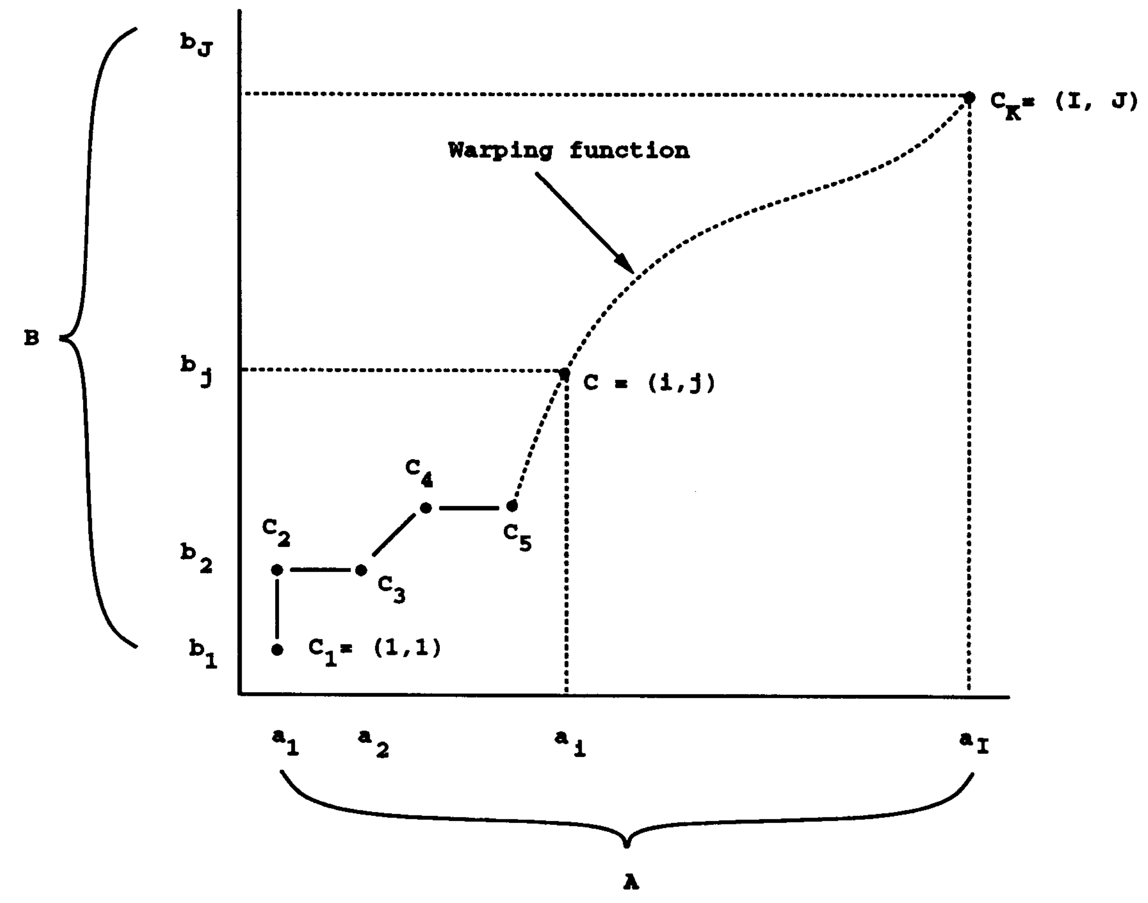

Figure 5-21:
Depiction of Sakoe and Chiba's (1978) correspondence diagram from their speech recognition task. The A axis represents the times of the subjects utterances, and the B axis represents the times of the model's predictions. The places where they correspond are represented by the C terms. The relationship of all the correspondences is seen as a warping function between the axis.

regression based on the correspondences, representing the warping function between the data and the model's predictions is drawn as a solid black line. While in each episode the correspondences may not be well fit by a linear relationship, theoretically the correspondences between the model and data should be a linear relationship. This line can be used in several interesting ways. The regression line helps to show where the fit is poor, highlights outliers, and gives a standard statistic, variance accounted for or r^2, that could be compared on a per subject and per model basis (e.g., Thibadeau, Just, & Carpenter 1982; Just & Carpenter, 1985). It also summarizes the relative processing rate of the model to the data, providing an empirical measure of the theoretical model processing cycle rate in seconds that should be more robust than simply dividing total model time by total subject time.

The regression line also makes several predictions. It can be used to find the mean percentage deviation from predicted for each subject action matched, mean absolute deviation (MAD), and root mean square deviation (RMSD). Measures like these can be used as part of engineering models of human performance to predict human performance (John, 1988), and to predict how accurately the model's predictions will be in the future.

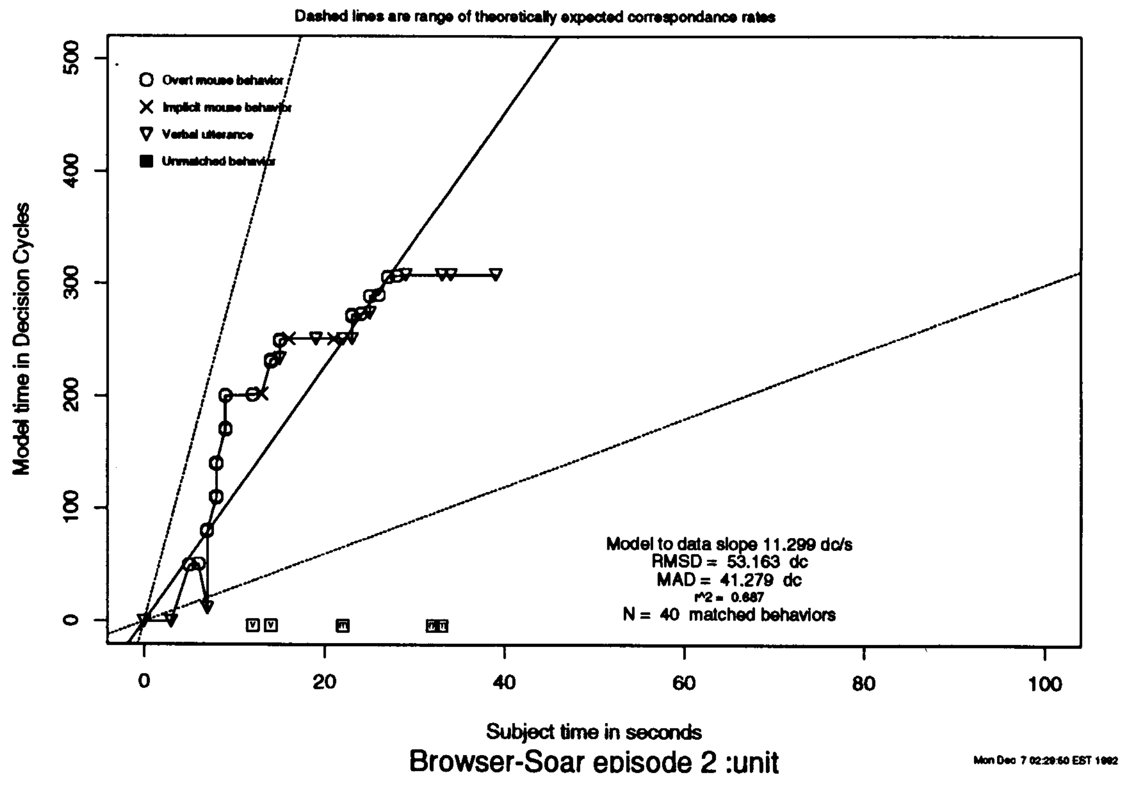

Figure 5-22:
Example relative processing rate display based on decision cycles taken from the Unit episode of Browser-Soar. The straight, solid line is a least-squares regression line through all the correspondences. Its slope is the relative rate between decision cycles and seconds. The dashed lines indicate the expected range for this measure. The location and type of the correspondences are marked on the connected line.

The model's processing rate as measured could also be compared to the theoretical decision cycle rate, but this is not completely known for the architecture used here. The Soar theory predicts the decision cycle rate only within an order of magnitude (Newell, 1990), that the rate of decision cycles will be between 3 and 30 cycles per second. To facilitate comparison with the empirically derived rates, these theoretical rates are presented as dashed lines on the display. Particularly if the regression is redone taking into account the dependent nature of the measurements, the regression results can serve as useful initial measure of the decision cycle rate. Just and Carpenter (1985) perform this analysis for the CAPS architecture, finding a 200 ms cycle rate.

The main use of the regression line, however, is for highlighting the systematic deviations. The rate of match between the two action streams should be a linear relationship, with the slope determined by the relative relationship between the model's cycle time and the actual time in seconds. Points that fall above or below the line indicate sets of behaviors that are not being performed with commensurate effort (or that follow such situations). These outlying points indicate where the model's behavior could be improved. Grant (1962) suggests three ways to use the regression line to find outliers: (a) examine the curve and points as they stand, noticing outliers, (b) draw error (95% confidence) regression lines, and look for points outside them, and (c) draw error bars for each point. In general, examining the plain line for outliers appears to be sufficient.

Signature patterns of model modifications. Table 5-18 lists the signature visual patterns that can appear on this display, and the indications they provide for how the model should be improved. How, or whether to remove them through modifying the model will be based on the purpose of the analysis and other factors. While the analyst can find out information about these outliers by clicking on them, the indication of how to modify the model is indirect. There is not an equation or set of complete rules for prescribing how to modify the model based on the correspondences, or what will happen based on the modifications. The model must be modified and refit. If there are prescriptions that can be drawn from these displays, they are not yet discovered, they will only come from further use and validation through experience with these visual displays.

Table 5-18: Signature correspondence patterns indicating types of model mismatches.

- Horizontal regions in the correspondences line indicate sequences of actions where the model performs the task too quickly relative to the subject. The model's performance may need to be expanded, or it needs to be done in a more cognitively plausible way. Alternatively, the subject may be performing more slowly than the subjects used to develop the model. This may be seen as a limitation of processing capacity of the subject, that the subject's full attention has not been paid to the task, or that the subject is less practiced at that portion of the task.
- Vertical regions in the correspondences line indicate sequences of actions where the model is performing slowly compared with the subject. The model is performing too much work.
- Downward right diagonal lines, indicate sequences (or pairs of actions) where the model and subject may be performing subtasks in different orders. This may indicate an individual difference in the subject in preference order for two operators, or if the actions are of different modalities, it may reflect reporting lag between different subsystems.
- Verbal statements that appear substantially separated to the right of their corresponding overt behaviors, indicate a lag in protocol generation, sometimes making protocols locally retrospective.
- Unmatched subject actions indicate that the model may not be performing the task completely or correctly. They may also indicate that the subject performs the task in an inefficient manner.
- Unmatched model actions indicate unnecessary actions, particularly if the subject performed the task correctly.

Examining learning within an episode. The linear regression line also supports a simple, initial examination of learning within an episode. If the Soar model is run with learning off, and if the subject learns information that transfers within the episode, then the fit of the data to the model's predictions would be concave upwards (the subject's performance speeds up relative to the model's performance). If the fit is concave only at the start of the episode, that indicates unmodeled startup effects.

Limits to the regression line. There are three problems with this regression line and its use. First, and most importantly, while it can point out outliers, it does so in a history dependent manner. If a series of actions represent a poor match, subsequent actions will also appear to be outliers with respect to the regression line. For example, in Figure 5-22, the actions from approximately 20 seconds to 40 seconds follow the same slope as the regression line, but are visibly offset.

Second, while the interpretation and alignment process results in an interpretation that forces the regression through the origin, this distorts the regression assumptions (e.g., Neter, Wasserman, &

Kutner, 1985, Ch. 4). The residuals no longer necessarily sum to zero, and the fit of the later data points is not as good as those near the origin.

Third, the value of r^2 returned by this regression is suspect because the data violate the independent measures assumption. Linear regressions assume that each data point is independent, and in this case, they are not. The regression is fit to a time-series, and the high values of r^2 may not hold when a regression that corrects for the dependencies is performed (Kadane et al., 1981; Larkin, et al., 1986).

<u>Computation of relative processing rate using operator applications as the model's unit of time.</u> In addition to decision cycles, there are other possible theoretically interesting measures of the model's effort. It is possible to modify the time-based comparison display to use a different time unit for the model. Soar models will have at least three main units of model time available to them, decision cycles, elaboration cycles, and operator applications. This is not an exclusive list, the selection of new states and problem spaces, and the rate of chunk creation and application could also be considered as possible units of model time. Figure 5-23 shows an example of using operator applications as the unit of model time. This time measure abstracts away some of the time information. The ratio of decision cycles to operator applications is not specified, and may not in the end be a useful measure. If the effects of decisions to select goals, problem spaces, and states, are relatively minor or correlate highly with operator selections, then the displays will appear to be very similar. If they are divergent, this may have indications for the architecture, or this may merely indicate that operator application rates vary between subjects or tasks. In either case, it tells us more about how to improve the model and illustrates the flexibility of the environment to explore new measures of model fit.

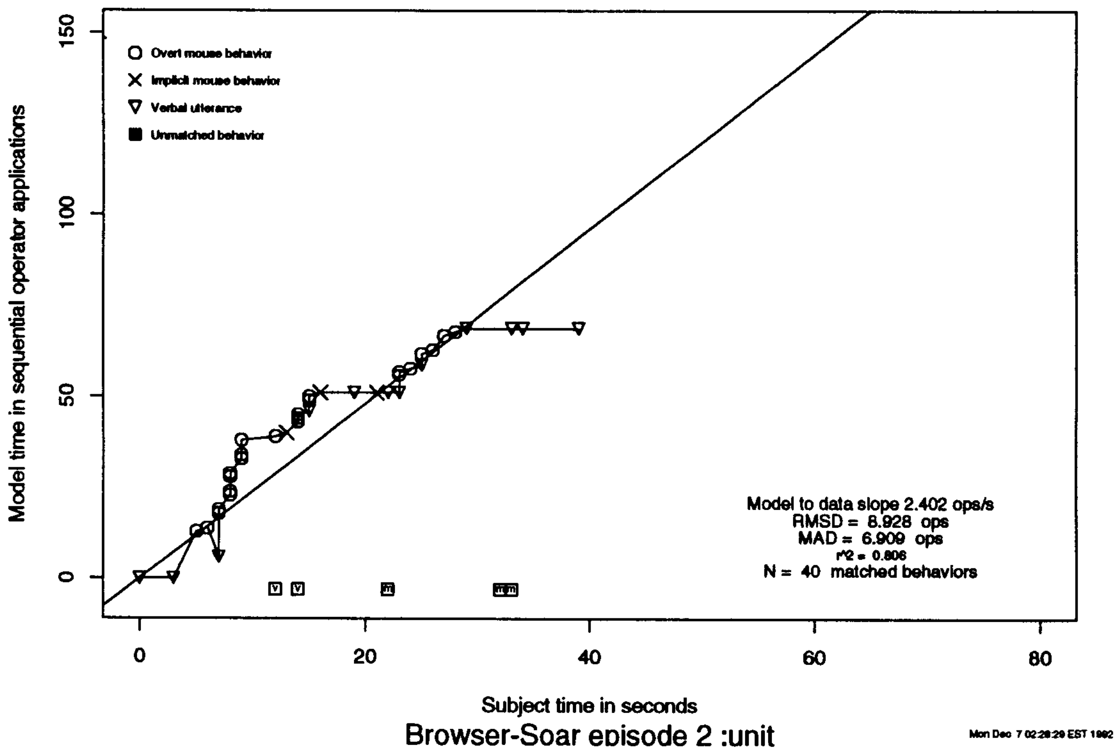

Figure 5-23: Example relative processing rate display based on operator applications taken from the Unit episode of Browser-Soar.

5.2.2 Using the relative processing display to test the sequentiality assumption of verbal protocol production

The relative processing display supports evaluating two features of Ericsson and Simon's (1984) theory of verbal protocol production. The first is the sequentiality assumption, that structures are reported verbally in the order that they enter working memory, and that "Information required as input to some process or operation will be verbalized before the output of that operation is verbalized" (Ericsson & Simon, 1984, p. 233). If the two parts of this assumption hold, subjects are not going to provide a verbal report of something that has sat around in WM for a long time and then delivered, like a letter stuck in a wall in the post-office. This assumption might also be extended to non-verbal protocols.

The second feature that can be examined with the model's predictions of working memory contents is whether the utterances are retrospective or prospective, and the time lag (or lead) of the utterances. Matched overt non-verbal task actions (e.g., clicking a button that has to be clicked to perform the task) can provide fixed data points for computing the offset of the verbal utterances.

In testing both of these features, we are also testing our models. If these reasonable assumptions are consistently violated by the models, in addition to calling into question the assumptions, we must also question the models.

The sequentiality assumption. There are two ways to test the sequentiality assumption. The most direct way is to examine working memory contents directly with neurophysiological tools. This is not yet possible. The other way is to use a model to predict what is appearing in WM. The model's goal stack is a model of what is in working memory, and can be used to test the sequentiality assumption.

The relative processing rates display supports this analysis visually. They represent the correspondences of the protocol segment to the model's predictions as temporally ordered connected symbols. The sequentiality constraint can be quickly checked by examining a display and finding only positive or zero sloped connections between all the protocol segments. Negatively sloped connections between two segments indicate a pair of verbal utterances that violate this assumption.

In addition to verbal protocols, this assumption can be extended to other data streams from the subject, for example, such non-verbal protocols as mouse movements or key presses. The testing process will be the same, except different symbols representing different data streams will be examined.

The cross-modal sequentiality assumption. One might also expect to see a difference in correspondence between verbal utterances and overt task actions. The Ericsson and Simon (1984) theory says that subjects report on information in WM (Ericsson & Simon, 1985, p. 264) along with reporting on inputs before outputs. In valid protocol (their talk-aloud vs. think-aloud utterances), only working memory items used in the task will be reported, and objects with verbal representations will be more easily reported than those that must be translated first (Ericsson & Simon, 1984, pp. 95-100). Therefore, the overt task actions and reports of mental actions may not occur in order. The overt actions (mouse clicks, moves) might not be verbally reported at all, they may not exist in WM or they may not exist in a form that is easily reported verbally. If the overt actions are not reportable verbally, they will still occur as overt actions, while verbalizable information in WM may have to be buffered, or may be reported as the overt actions pre-conditions or post-conditions.

It is not clear what the direction and size of this offset between verbal and non-verbal protocols should be. If the items reported truly are only operator inputs and outputs, then verbal utterances will be presented in order with overt actions, or with a lag, where an overt action occurred while information had not been told yet. If the utterances include goal statements about overt actions to be done, then the utterances may be prospective of the overt behavior.

The amount and direction of lag may be an indicator of protocol quality. If the lag between structures entering working memory and being reported is too long, the protocols are retrospective. The analyst must postulate the uses of long term memory processes that are actually producing the utterances, particularly if the working memory elements reported on have also left the working memory by the time of the report. If the lag is positive the protocols are prospective, and may be including introspective comments. The lag may also indicate tasks (or subtasks) that have a non-verbal representation. The subject's utterances will include covert activity to translate the structure into a verbal representation.

5.3 Creating additional displays

One might now imagine creating numerous types of displays like these, sequential displays based on other time measures of the model and displays that presented the correspondences in a data dependent order. A more extensive sample list is presented in Table 5-19. It is not quite clear in each case what the display will look like, but some of them surely will be interesting and useful. More importantly, it is now possible to consider creating them automatically from the data.

The idea of a single description that shows how the model could be improved now seems small-minded. An analyst trying to understand and improve their model will want many types of displays. In order to create such displays, the analyst will require an underlying system that provides the functions to make them, and an environment for creating and modifying them. S provides the functionality to create the displays. S-mode provides an interface for creating additional displays as functions to be called on a dataset.

Table 5-19: Further displays for summarizing the fit of data to model predictions

- Cumulative model predictions matched over time.
- Scatter plot of correspondence times (rise and run in the relative processing rates display).
- Correspondences with primary order coming from the data instead of the model's predictions.
- Correspondences presented over time with respect to problem spaces or elaboration cycles (instead of operators or DCs) vs. subject time.
- Problem behavior graphs (depictions of states rather than operators).
- A matrix showing production conflicts.

5.3.1 S: An architecture for creating displays

These displays have been implemented as functions in S (Becker, Chambers, & Wilks, 1988; Chambers & Hastie, 1992), an interactive, exploratory, statistics and graphing package. S provides a set of general and easy-to-use facilities for organizing, storing, and retrieving data structures such as matrices. In addition to the numerous built-in numeric and graphing functions, libraries of advanced data analysis routines (such as ANOVAs and logistic regressions) are available from a fairly large and friendly user community. S is programmable, and users can create functions to perform set analyses, to combine smaller analyses, and to create interactive graphic displays. Other flexible, programmable graphing packages, such as GNU-Plot, could also have been used, but S is perhaps the most powerful and appears to have the largest user community.

While S provides all this functionality, it suffers from two disadvantages. First, it does not so much have an awkward interface, but a primitive one: a simple TTY command line. There is no facility to edit the functions in a structured way, treating them as first class objects to be loaded, edited, and run.[5]

[5]S does provide a command that will call a plain editor on a single, named function.

Secondly, S is the only piece of software in the testing environment that is not freely available. This problem is attenuated by the wide distribution of S.

5.3.2 S-mode: An integrated, structured editor for S

In order to create these displays that are functions in S, a structured editor created within GNU-Emacs is provided, called S-mode (Bates et al., 1990; Smith, 1992a). S-mode has been joint work with Kademan, and Bates over the last three years, and Smith for the last nine months. S-mode provides a structured editor to write, load, and edit S programs, and an improved command line interface.

Most analysts will use S to create displays in an edit-test-revise cycle. When programming S functions, S-mode provides for editing S functions in GNU-Emacs edit buffers. Unlike the default use of S, where the editor is restarted every time an object is edited, S-mode uses the current Emacs process for editing. In practical terms, this means that one can edit more than a single function at once, and that the S process is still available for use while editing. Error checking is performed on functions loaded back into S, and a mechanism to jump directly to the error is provided.

S-mode also provides mechanisms for maintaining text versions of S functions in specified source directories. These objects can be manipulated by the user as first class objects, and be examined, edited, and loaded. S-mode provides an interactive command history mechanism, including a quick prefix-search of the history list. To reduce typing, command-line completion is provided for all S objects and keybindings are provided for common functions. Help on individual S functions and on S-mode itself are easily accessible, and a paging mechanism is provided to view them. Finally, an incidental (but very useful) side-effect of S-mode is that a less literal transcript of the session is kept for later saving or editing than S provides by default. A complete listing is available in the manual (Smith, 1992a), and a summary is available in Table 5-20.

Table 5-20: Functionality supported by S-mode.

- Edit S object in a buffer.
- Display help on a function or variable.
- Jump to the S process.
- Load a single line.
- Load the current function.
- Load the current function and go to the S process.
- Load a file of S functions.
- Reformat the current function.
- Complete an object name by querying the S process.
- Move to the beginning (or end) of a function.
- Execute the previous command.
- Insert a function template.
- Automatic matching of parentheses and braces.

5.4 Supporting the global requirements

In addition to the direct requirements of aligning the predictions with the data and starting to interpret their comparison, there are five global requirements that the displays and S-mode also support.

5.4.1 Providing an integrated system

The S-mode and the graphing functions are integrated with the other portions of Soar/MT environment. The data for the displays can be dumped from the spreadsheet into text files and read into S data structures for creating the displays. This step of transferring the data could be more automated, but the path for passing this information is well worn and can be performed relatively quickly. Although the interaction is on the level of files, because of the file manipulation facilities GNU-Emacs provides, it is easier than it sounds.

The functions for creating the displays were designed using S-mode, and S-mode provides a fine environment for routinely calling the functions to make the displays. If at some point an order of magnitude more data is used, on the order of hundreds of subjects, it may be useful to use the batch processing facilities of S to create these displays, where the commands are not executed interactively but as part of a file of commands.

5.4.2 Automating what it can

The graphing functions have automated the display and global analysis to a great extent. What used to take a day to display can now be performed in minutes. Once created, a display can be called with nearly no cost to the analyst, so using many displays to understand a model is possible. S-mode automates many of the actions necessary to create additional displays as S functions. A table similar to Table 6-25 could be created for S-mode.

5.4.3 Providing a uniform interface including a path to expertise

The initial set of displays provided do not require any expertise beyond knowing their inputs, and this is provided with their definitions. As a prototype, this has been an adequate level of documentation. If more users start to use them, the documentation will have to be improved.

S-mode, like Spa-mode, provides a path to expertise similar to that provided by other components in the Soar/MT. S-mode can be menu or keystroke driven. The keystroke bindings of the menu commands are available to the user, and are displayed to him or her after they have been completed.

Documentation is also available as hardcopy. The S-mode manual (Smith, 1992a) and a reference card (Smith, 1992b) are available through the S-mode menus, and obtaining on-line documentation on functions takes only two keystrokes.

5.4.4 Providing general tools and a macro language

S and S-mode provide a very general set of capabilities for performing exploratory analyses and creating new displays for model testing. S commands can be combined to create functions to draw nearly any display imaginable. The creation of these displays is supported through S-mode, as well as applying them to each data set.

Hooks are places to customize a system's behavior by calling a user supplied function at a set point, such as at startup, or after a file has been loaded. The standard set for GNU-Emacs modes have been included with S-mode. The user supplied function, if any, is called when S-mode is loaded or initialized.

5.4.5 Displaying and manipulating large amounts of data

These graphic displays directly support examining a large amount of the model's performance and examining the relationship between the model's predictions and the data. The displays include the ability to examine individual data points based on their location. The displays are based on the model, and can use their representation of the model to make clear which model objects have generated predictions found in the data, and which have not.

Supporting direct manipulation. The displays and S-mode directly support manipulating the main objects of interest on this level, the data points within the displays, and the displays themselves. The functions to create new displays (and their components) are first class objects in S-mode, and allow the analyst to load and manipulate them directly.

On a lower level, the points on the displays are inspectable; clicking on them (after an appropriate function call) will print out the Soar trace, the verbal utterances, their time stamps, and other information, known about that point. Selected sets of points could even get thrown into spreadsheet for further analyses (but currently this is not supported). These interactive displays also serve as a way to understand large data sets by hiding irrelevant fields, but allowing them to be recalled on demand.

5.5 Summary of measures and recommendations for use

The two measures presented here (the operator support display and the relative processing rate display) provide the analyst with ways to view the correspondences between the data and predictions with respect to the model and the time course of the correspondences. Both types of displays provide visual patterns indicating where to improve the model and where the model is consistent with the data.

These displays do not primarily provide local, immediate information about the comparison, but this too are included as a separate block of text on the displays, and the local comparison is also available in the matching tool, Spa-mode.

These displays are not the only ones possible for depicting the comparison, the model's predictions, the data, or various combinations of these, they are only a starting point. There are many ways the model can fail to predict the data, and there are many facets to the relationship between models and the data, so many different types displays will be required by an analyst wishing to improve their model. Further displays can be created using S and the S-mode interface, and several new displays can already be suggested as potentially useful.

Problems with these displays. There remain at least three problems that apply to both types of these displays. When interpreting and using these displays analysts will have to keep in mind: (a) While the architecture treats all operators the same, the modeler and model may have different sized operators (with respect to correspondence to the subject's actions) and different levels of theoretical commitment to particular operators. Some operators may be represent placeholders for complicated operators, such as *read*. These differences are not currently represented. The relative processing rate could vary quite a bit when the grain size changes between operators in this way. (b) Not all analysts will be committed to time based comparisons. They may be interested in other facets of their models, which will need additional displays. (c) Before they are learned, Soar operators are implemented hierarchically. It is not clear how to represent in the displays when hierarchical operators are in effect, and their calling order.

What do these displays have to say about comparing two models? Given these displays, comparing how well the data fits the two models does not come down to comparing two numbers, but can now be based on more analytical (but less straightforward) process of comparing the models' performances in more meaningful ways. As indicated by the displays, the models will have regularities associated with them, places where each model's predictions match the data more closely than the other, but only a metric outside of the comparison can order these comparisons. The models' proponents, however, can now point to diagrams showing the comparison, and describe more clearly and fully how and where their models match the data.

Future work. While these displays use the models and its structures to help in the analysis, the question remains of how to extend them. It may be possible and desirable to incorporate additional components, such as the number of rule firings, the number of matches per rule[6], and to aggregate across subjects or episodes. It would also be desirable to incorporate measures of rule and operator utility more directly, and to understand, display, and manipulate the degrees of freedom in the models.

[6]Although generally Soar models are not described on the level of rules and rule firings, but the higher level cognitive structures, such as operators, that the rules are creating.