GroupAnalyzer: A System for Dynamic Analysis of Group Interaction

Marcial Losada

Shaul Markovitch

EDS Center for Machine Intelligence
2001 Commonwealth Blvd.
Ann Arbor, MI 48105

Abstract

The importance of coding and analysis of group interaction in order to better understand collaborative work has long been recognized. It has been also acknowledged that using such analysis for feedback purposes can enhance collaborative behavior. However, current applications of such methodologies suffer from two major drawbacks: a large time gap between the actual behavior and feedback, and the loss of temporal information arising from the static nature of existing analysis methods. In this paper, we introduce *GroupAnalyzer*, a computerized system for coding and analysis of group meetings. Some of the features that GroupAnalyzer provides are: fast and accurate coding, accurate timing, fast feedback, integrated numerical and graphical representation, and comprehensive dynamic analysis. We conclude by pointing to the potential of GroupAnalyzer for facilitating collaborative behavior.

Introduction

The field of computer-supported collaborative work has received tremendous attention during the last four years [1, 2, 3]. Most of the research done in the field has been concerned with supplying the collaborators with computers in order to improve their collaboration. Relatively few works exist outside of this framework, mostly those concerned with using the computer as a tool to facilitate meetings [4].

In this paper we introduce a concept and a system that expands the scope of the field: using computer technology to code meetings, analyze them and give feedback to meeting participants in order to support a better collaboration between them. The idea of using analyses of coded group behaviors for feedback is not new. A vast amount of research has been done in this area, notably the work of Bales and his associates at Harvard University [5, 6, 7, 8].

However, there are two major limitations with the existing coding and analysis methods. The first limitation is the large time gap between the actual behavior and feedback which stems from the manual method used for coding. The second is the loss of temporal information arising from the static nature of existing analysis methods.

In order to overcome these limitations we designed and implemented GroupAnalyzer, a computerized system which allows efficient coding of meetings following the SYMLOG formalism [8] (and other formalisms in future versions), comprehensive analyses of coded protocols and prompt feedback to the meeting participants on their interactive behavior.

In the next section we will give the background which includes a description of the collaboration laboratory where GroupAnalyzer was implemented and tested, an overview of SYMLOG and a discussion of the limitations of its traditional application. To address these limitations, we will present an overview of the GroupAnalyzer system. Then we will proceed to a detailed description of the coding and analysis modules. We will emphasize the dynamic features of the analysis module, which make it a unique system in terms of providing a kinematic display of group interaction, allowing participants to see the evolution of their behavior through the entire meeting instead of just one overall static picture. We will end with a discussion of some results obtained using GroupAnalyzer to study group dynamics and argue how these findings point to the potential of GroupAnalyzer to support collaborative work.

Background

Two events have preceded and become a driving force in the development of the GroupAnalyzer system. The first is the surge of computer-supported collaborative work which led to the creation of the Capture Lab at the Center for Machine Intelligence in Ann Arbor, Michigan. The second is the development of SYMLOG by Bales and his associates at Harvard University which initiated a resurgence in group dynamics [9].

0073-1129/90/0000/0101$01.00 © 1990 IEEE

The Capture Lab

The Capture Lab is a computer-supported collaborative environment where participants can have fast and easy access to a publicly shared big screen by simply pressing a key on their personal workstations. In addition, they can use these personal workstations to transfer and receive information from the public screen. These workstations are embedded in a conference table in order to facilitate face-to-face interaction [10]. Three video cameras are unobtrusively located behind mirrors and their output is transferred to a monitoring control console in the observation room. This room is separated from the Lab by a one-way mirror. From this room behavioral scientists can observe and code meetings using personal workstations.

When the Capture Lab was created, one of its purposes was to have a state-of-the-art laboratory that would allow us to observe, capture, understand and improve behavior in meetings. In addition to the sophisticated hardware necessary to achieve these objectives, there was a requirement for software that would capture as much of the ongoing group dynamics as possible.

SYMLOG

SYMLOG (a system for the multiple level observation of groups) has been used extensively in several countries for a number of years with excellent results [11]. Computer programs to facilitate the analysis, once the data gathered manually have been entered into the computer, have been developed mainly by Polley (see appendix U in [8]). In his seminal review on group research, McGrath has referred to Bales' research as "among the most consistent and robust findings in the field." [9, p.145].

In this section we will only focus on the essential concepts of SYMLOG necessary to understand GroupAnalyzer. For a comprehensive description of SYMLOG, the reader is referred to [8]. SYMLOG's coding scheme operates within a three-dimensional space. The three spatial dimensions map into three corresponding psychological dimensions. 1) Up-Down (U-D dimension) whose psychological correlate is represented by dominant vs. submissive behavior; 2) Positive-Negative (P-N dimension), psychologically mapped into friendly vs. unfriendly behavior; and 3) Forward-Backward (F-B dimension), whose psychological correlate is task-oriented vs. emotionally expressive behavior.

SYMLOG uses these three dimensions to code interactions at two levels: the behavioral level and the image level (which addresses the issue of the meaning conveyed by behavioral acts and, consequently, is more interpretative than descriptive). In this paper we will focus on the behavioral level. The behavioral level comprises both overt and non-verbal behavior.

A SYMLOG coder must enter by hand the following information: a) the time of the event (approximated to minutes); b) who is the actor or sender; c) who is the receiver of the action; d) a specification of whether the observed behavior was overt or nonverbal; e) the behavioral code; f) a comment describing the behavior briefly. An example of a coded event is:

"10 JHN MAR N B smiles"

This code means that 10 minutes after the meeting started, John smiled to Mary. The N stands for nonverbal behavior. The B is the code for emotional behavior.

The main output of a SYMLOG coded session is a "field diagram" which summarizes the average group behavior by representing each participant as a circle whose radius conveys the level of dominance. The larger the circle, the more dominant the person. The circle is located in a two-dimensional plane whose vertical axis is the F-B dimension and whose horizontal axis is the P-N dimension. Bales argues that this graphical representation of group behavior is more powerful than numerical information: "The visual diagrams seem to allow members to 'externalize' their observations of group process and group relationships in a concrete and observable manner and to speak about them much more easily and directly than they generally can about numerical data" [8, p. 320].

The next two sections will point to two classes of problems with manual SYMLOG coding and analysis.

Coding problems

Slow Coding: Under the traditional manual coding methods, observers produce no more than one or two codes per minute [8]. A slow coding rate is a limitation for two reasons. The obvious one is that less of the meeting's activity is being captured. The less obvious is that a positive correlation between the rate of recording codes and the qualitative goodness of the scoring has been reported [8].

Imprecise Timing: One problem of manual coding is the loss of accuracy in recording time. A description of how time is measured under traditional paper and pencil SYMLOG scoring will convey to the reader this loss of precision: "A digital clock should be visible to all observers for best results...The teacher of the observation team should explain which digits on the clock are to be recorded. The observers may then begin their observation. The observer looks at the group and, when a new person speaks, tries to recall the person's name, checking with the diagram on the back of the sheet if necessary. The observer then looks at the clock and writes the time on the message form..." [8, p. 311].

Difficulty in recalling names: The above quote reveals another limitation of the traditional scoring method; that is, the need to recall the names of the participants in order to enter the three letter code necessary to identify senders and receivers in the scoring sheet. This is Bales' prescription: "The remedy is simple. The observer should look at this person and recall the name repeatedly until the memory sticks..." [8, p. 312]. This imposes a cognitive load on the scorer that can interfere with her or his coding speed.

Analysis Problems

Long delay in feedback: One of the most important limitations of manual SYMLOG scoring is the delay in providing feedback on the group interaction to the participants. At the Harvard Social Relations Laboratory, where SYMLOG was originated and used to train students in group dynamics, it typically required two days to produce a field diagram and give feedback to the group participants [8]. Considering what we know from the literature on behavior modification, for feedback to be most effective it should be given as close to the event as possible [12].

Static analysis: Another important limitation of the traditional scoring system of SYMLOG is that it cannot provide dynamic analysis: one that captures the flow of the meeting and reflects the changes in group members' interactive patterns over time. Such analysis is important for both the researcher and the meeting participants to fully understand the complexity of the interaction patterns that occur in a meeting.

The GroupAnalyzer system was built as an attempt to overcome all of these limitations. The next sections will describe the GroupAnalyzer system by presenting first an overall description of the system and then a more detailed view of its coding and analysis modules.

System Overview

In this section we will give an overview of the main components of the GroupAnalyzer system. The GroupAnalyzer system consists of two main modules: the *coding module* and the *analysis module*. The coding module is the part of the system that enables the coder to create entries in the meeting protocol together with a set of tools for manipulating meeting protocols. The analysis module consists of a set of tools for analysis of meeting protocols. The two modules are independent (see Figure 1). We could disconnect the two modules to create two separate systems, however merging them together into one system allows us to analyze meeting protocols at any stage of their creation.

The Coding Module

The coding module is a system that enables coders to easily transform their observations to processable protocol entries. The interface to the coding module is implemented by a screen which stays on through the whole coding session (see Figure 2). The interface was designed to reflect the physical layout of the meeting room. The big gray rectangle in the middle represents the meeting table while the eight button groups around it represent the participants. The coding screen contains a button for each of the participants. At the beginning of the meeting the coders fill those buttons with the participants names. In addition, each participant button has two arrow buttons attached. The arrow directed toward the participant button represent an action toward that participant. The arrow pointing away from the participant represents an action initiated by that person. Many times an action taken by a participant is aimed at the whole group. In such a case the coder can press the *To Group* button which is equivalent to pressing all the participants *To* buttons.

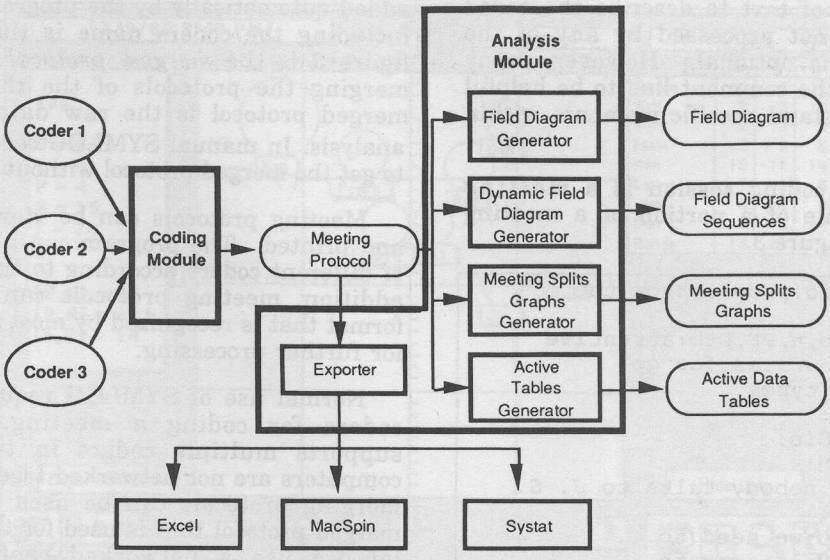

Figure 1. GroupAnalyzer system design.

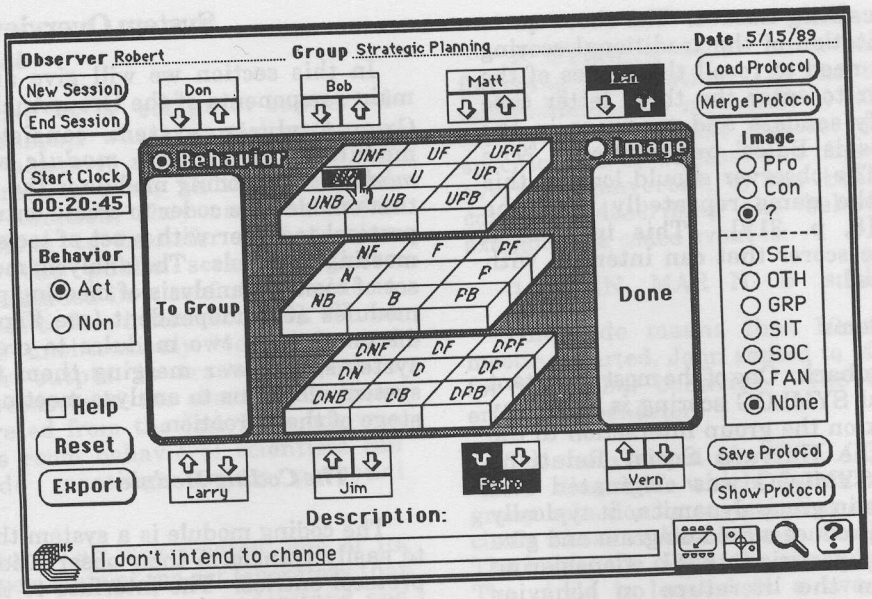

Figure 2. Coding screen.

In the middle of the rectangle that represents the table there is a cube partitioned into three slices. The cube represents SYMLOG's three dimensional space. A cube was originally used by Bales to describe the SYMLOG space. We have borrowed the idea, but sliced the cube so that each of the points in the three dimensional space can be accessed by the coders.

To code an interaction the coder must select at least three buttons. One button for the originator of the interaction, one button for the recipient, and one of the buttons in the cube for the behavioral act. The buttons can be selected in any order. The coder signals the system that an entry is completed by pressing the *Done* button.

At the bottom of the screen there is a box for entering a short line of text to describe the coded event. This line is not processed by any of the statistical tools and is optional. However, many times we have found the comment line to be helpful when trying to understand specific moments within meetings.

The product of a coding session is a *meeting protocol*. An example of a portion of a meeting protocol is shown in Figure 3.

```
1115,Pedro,G,a,F,Gio:people know the
     players
1118,Bob Vern,Pedro,n,PF,Deb:attentive
1125,Don,G,a,F,PAM:writes for gp
1130,Don,G,n,F,Deb:types
1131,Bob,G,a,F,PAM:
1133,Pedro,G,a,DF,Gio:
1139,Bob,G,a,UF,PAM:
1144,Bob,G,a,N,Deb:nobody talks to J. S.
1151,Bob,G,a,F,Deb:
1154,Bob,G,a,UF,Gio:we need to...
1156,Ken,G,a,UN,PAM:interrupts
```

Figure 3. Excerpt from a meeting protocol.

For example, after 1156 seconds have elapsed since the beginning of the meeting, Ken addressed the group (G), it was an overt act (a), coded as UN (dominant and negative), the coder first initials are PAM, and the comment is "interrupts."

The meeting protocol is similar to the one produced by hand in traditional SYMLOG coding. There are two features of the GroupAnalyzer generated protocol that are different from the manually generated protocol. The time stamps in traditional protocols are specified in minutes, whereas GroupAnalyzer uses seconds, automatically time-stamped by the computer. This feature proves to be very useful when fine grain time series analysis is needed. The other feature is the names of the coders at the comment part of each code. The names are added automatically by the program. The reason for including the coders name is that the protocol in figure 3 is the *merged protocol* - the product of merging the protocols of the three coders. This merged protocol is the raw data used for all the analysis. In manual SYMLOG coding it is impossible to get the merged protocol without extensive work.

Meeting protocols can be viewed, stored, loaded and printed. The program can also merge protocols of different coders according to the time stamps. In addition, meeting protocols can be exported in a format that is recognized by most statistical packages for further processing.

Normal use of SYMLOG requires three or more coders for coding a meeting. GroupAnalyzer supports multiple coders in two ways. If the computers are not networked together, the utility for merging protocols can be used for generating the merged protocol that is used for the analysis. In our lab we use a networked configuration, with a merging and analysis server. In this setup the coding machines send all their codes to the merger

machine as they are entered. The merger machine merges the code and performs ongoing analysis so that feedback can be given to the group at any moment. In addition, the networked configuration performs automatic synchronization of the meeting timers of the three coders.

The Analysis Module

The analysis module contains a set of tools for analyzing meetings based on the meeting protocol produced by the coding module. The analysis tools can be categorized into two classes: static analysis tools and dynamic analysis tools. The static analysis tools provide the researcher and participants with the means for looking at the average behavior of the meeting participants in a similar manner to that of the traditional SYMLOG analysis. The dynamic analysis tools help the researcher and participants to understand the dynamics of the meeting: how the behavior of the participants evolved during the meeting.

Static Analysis

GroupAnalyzer provides two tools for static analysis of meeting protocols. The first tool is the field diagram generator. This tool automatically computes and draws the field diagram as described in [8]. The researcher can either produce a field diagram of the whole meeting or produce a field diagram of a selected portion of the meeting protocol. Figure 4 shows an example of a field diagram produced by GroupAnalyzer. In addition to the traditional field diagram which appears at the center of the screen, the researcher can get more information by clicking on various parts of the screen. Clicking on a participant name causes the program to display the numbers that are associated with the circle of that participant. Clicking on the time stamp takes the user to the meeting protocol's line that corresponds to that time stamp. It is also possible to hide the participants names. This facility allowed us to print a version of the field diagram without the names and ask the group members to guess which circle belongs to which name, permitting us to observe what is the participant's perception of her or himself and of other people in the group.

Another tool for static analysis of meetings is the active tables generator. The tool produces a table that shows the number of activities of each type that each of the participants has been involved in. The table is active: clicking on any number within the table displays a list of all the entries in the protocol in which the selected participant exhibits the selected behavior. Figure 5 shows an example of a table produced by this tool.

Dynamic Analysis

The traditional analysis of the SYMLOG data provides the researcher with a view into the average behavior of meeting participants. The problem with such an approach is that it does not show the dynamics of the meetings. GroupAnalyzer supports three methods of looking into the dynamics of meetings.

1) Animated sequences of field diagrams. The dynamic field diagram generator provides the means for the generation, manipulation, and animation of sequences of field diagrams. Figure 6 shows the commands screen for creating field diagrams and field diagram sequences. A sequence is determine by two parameters: *window size* and *window sliding rate*. The window size is the length (in seconds) of the segment of the meeting that is used for each of the

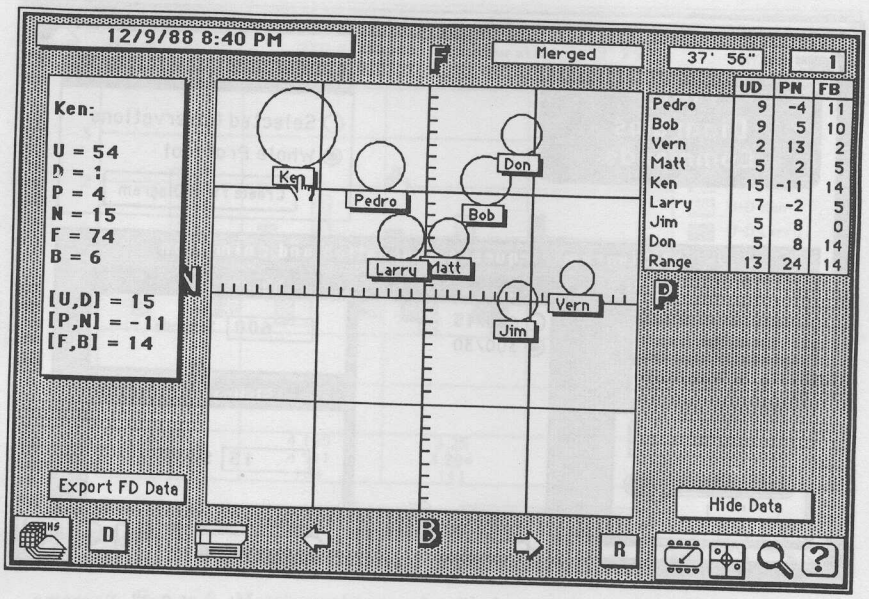

Figure 4. Field diagram.

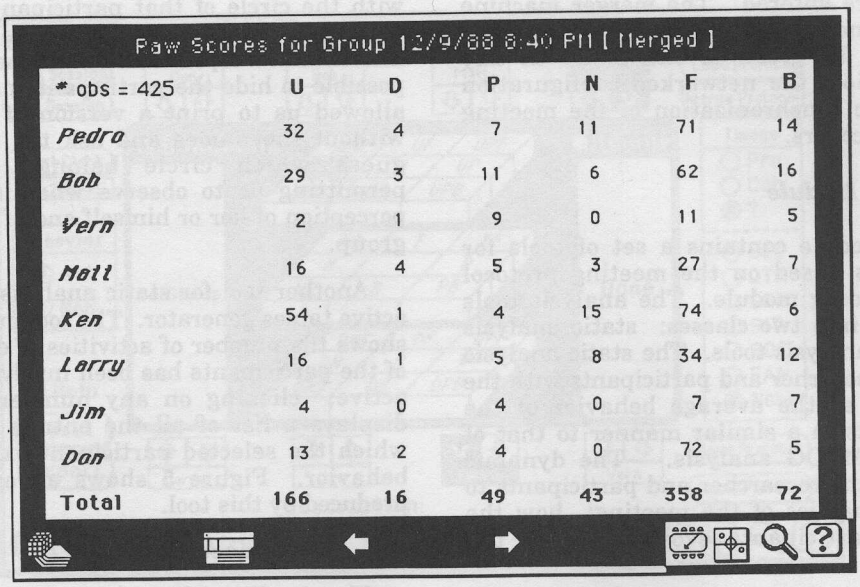

Figure 5. Active table: clicking on any number gives a listing of all related protocol entries.

field diagrams in the meeting. The sliding rate indicates how much the initial point of the segment is increased from one field diagram to the next one. If, for example, the window size is 600 seconds and the sliding rate is 15 seconds, the first field diagram is an average of the first 600 seconds of the meeting, the second field diagram is the average of a window that starts at second 15 and ends at second 615, etc. Smaller sliding rates reduce the effects of new activities on the moving average and make the animation smoother.

2) Meeting splits graphs. While the animated sequences of field diagrams help the researcher (or the meeting participants) to get an intuitive feeling for the dynamics of the meeting, the meeting splits graphs generator provides quantitative information about the flow of the meeting. Figure 7 shows the command screen for producing the graphs. There are three parameters that determine what will be displayed in the graph: the *Participants*; the *Dimensions* to be included, and the *Meeting Splits* which specifies the number of segments into which the meeting will be divided.

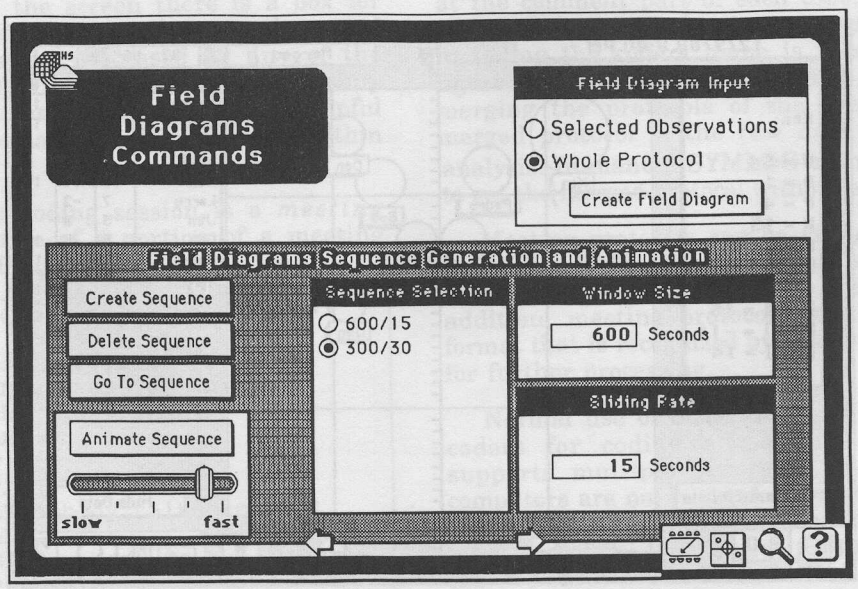

Figure 6. Control screen for sequence generation and animation of field diagrams.

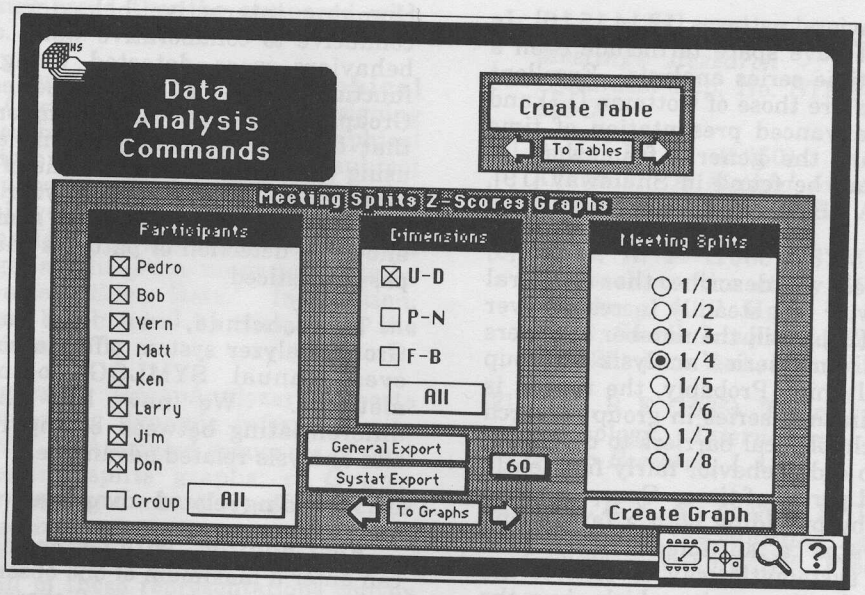

Figure 7. Control screen for meeting splits graphs and generation of active tables.

Figure 8 shows an example of an output produced by the meeting splits graphs generator. One can quickly see an interesting pattern: while Pedro's dominant behavior increases over time, Ken's dominant behavior decreases. The meeting has been split into four segments, each of 9 minutes and 29 seconds. For each meeting split, the mean value for the dimension over the group was calculated. The ordinate is given in z-scores; i.e., the mean over the group for a segment has been subtracted from the raw score and divided by the standard deviation. The graph is active: pointing to a bar displays the statistics for the selected participant.

3) Time Series Analysis. The most sophisticated dynamic analysis that GroupAnalyzer supports is time series analysis. The information contained in the temporal organization of behavior is critical to understand interaction patterns between group members. Simply looking at behavior without any consideration for their temporal order would be like trying to understand music by counting how many different types of notes are there in a score. Behavior is like music, you need to preserve the flow in order to make sense of it. There is today an increasing understanding of the importance of using the appropriate time series analysis tools in order to

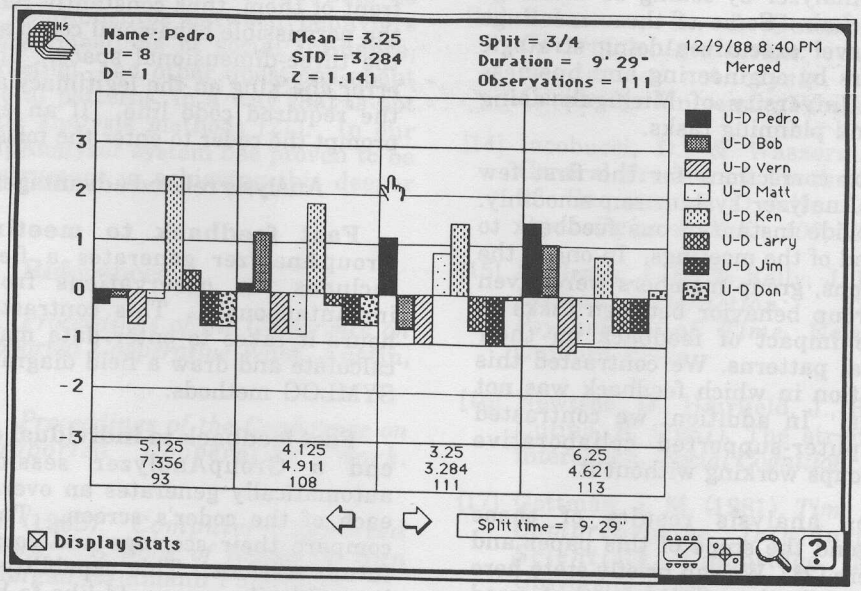

Figure 8. Meeting splits graph with four equitemporal splits.

detect interactive behavioral patterns [13,14,15,16]. In this paper, we do not have space to include even a brief introduction to time series analysis. Excellent introductory textbooks are those of Gottman [17] and Chatfield [18]. An advanced presentation of time series analysis within the general framework of regression methods can be found in Shumway [19], which includes an IBM disk with data and programs.

The use of models to describe the temporal organization of behavior has steadily increased over the last two decades [20]. Still the number of papers with applications of time series analysis to group behavior is practically nil. Probably, the reason is that in order to obtain time series in group research there are difficult technological barriers to overcome; i.e., it is necessary to code behavior fairly frequently and over an extended period of time. GroupAnalyzer solves this problem by providing both a fast coding system that generates a sufficient number of observations and a systematic way to control the sampling interval of the time series which gives the level of granularity at which one can observe interaction patterns.

Currently, GroupAnalyzer can export the generated time series to an external program (SYSTAT). However, we are planning to have GroupAnalyzer generate its own time series analyses including detrending filters, autocorrelation function, partial autocorrelation function, and crosscorrelation function in the time domain. In addition, we expect GroupAnalyzer to do cross-spectral analysis with coherency and phase estimation in the frequency domain.

Discussion

We tested GroupAnalyzer by coding 65 meetings held at the Capture Lab. Some of these meetings were held by high level executives doing strategic planning tasks, others by engineering and business students from the University of Michigan doing typical managerial and planning tasks.

After a few minor corrections for the first few meetings, the GroupAnalyzer system ran smoothly. We were able to provide instantaneous feedback to participants at the end of the meetings. In one of the experimental conditions, group members were given feedback on their group behavior between tasks in order to assess the impact of feedback on their interactive behavioral patterns. We contrasted this with a control condition in which feedback was not given between tasks. In addition, we contrasted groups using computer-supported collaborative technology versus groups working without it.

The time series analysis results of these experiments are beyond the scope of this paper and are reported elsewhere [21]. We can briefly state here that the effect of feedback on groups that used technology was highly significant in terms of changing interactive behaviors considered more conducive to collaborative work. These interactive behaviors were detected using crosscorrelation function analyses of the time series generated by the GroupAnalyzer system. It is important to emphasize that these interactive patterns are not detectable using traditional methods. One of the most powerful features of the GroupAnalyzer is its ability to generate time series data at sampling rates which allow the detection of patterns that otherwise would pass unnoticed.

To conclude, we can affirm that the GroupAnalyzer system offers a series of advantages over manual SYMLOG coding and analyses methods. We now list these advantages, differentiating between coding related advantages and analysis related advantages.

Coding related advantages

Fast coding: With GroupAnalyzer, a fast coder can enter a maximum of one observation in 4 seconds and, on the average, over 6 observations per minute. This is two to three times the rate of coding by hand.

Accurate timing: The GroupAnalyzer coder does not have to worry about entering the time a behavioral event occurs. The computer takes care of it by time-stamping the number of seconds that have elapsed since the meeting started. The clocks in all of the coders' computers are synchronized so that everybody has the same time reading.

Visual mapping of group seating layout: The one-to-one mapping of seating arrangements with the coding display relieves coders of the cognitive load of having to memorize the names of the participants.

Accurate coding: Coders using GroupAnalyzer have the sliced coding cube displayed continuously in front of them, thus constantly reminding them of all the permissible behavioral codes and their position in the three-dimensional space. The program will do error checking on the legitimacy and completeness of the required code line. If an error occurs, it will prompt the coder to enter the missing item.

Analysis related advantages

Fast feedback to meeting participants: GroupAnalyzer generates a field diagram which includes the observations from all the coders instantaneously. This contrasts with the several hours it takes to enter data manually and then to calculate and draw a field diagram under traditional SYMLOG methods.

Fast feedback to individual coders: Once coders end a GroupAnalyzer session, the computer automatically generates an overall field diagram in each of the coder's screen. This allows coders to compare their scorings and correct their biases in future sessions. This capability was in Bales' wish list: "Ideally one would like to have a field diagram produced from the scores of each observer separately,

and these diagrams could then be compared with each other..." [8, p. 344].

Integrated numerical and graphical representation: Our general philosophy in designing GroupAnalyzer has been to produce both graphical and numerical information. All the graphical information is displayed by default. In addition, one has always intuitively easy and fast access to numerical information. For example, in a field diagram one can get all the group numerical data by clicking on the button "Show Data." In addition, individual data can be obtained by clicking on the name of each participant.

Dynamic analysis: GroupAnalyzer supports three ways of presenting dynamic information to meeting participants: a) Animated sequences of field diagrams; b) Meeting splits graphs; c) *Group interaction diagrams* generated by time series analyses of the meeting and representing the cross-correlational links between participants' interactive behaviors [21]. All of these representations convey the flow of interaction in a way that allows the participants to easily see critical moments and patterns in their interactive behavior that are either conducive or detrimental to collaboration.

To recapitulate and conclude, we have highlighted in this paper a dynamic view of group interaction which allows a deeper understanding of the flow of behavioral streams in a meeting. If this understanding is properly fedback to meeting participants, it could prove to be useful in promoting collaboration. This deeper understanding of group interaction was possible because we were able to generate fine grain time series of the interactive behaviors that occur in a meeting, uncovering otherwise hidden patterns in the behavior streams of group dynamics. When we take into account the temporal ordering of behavior, we are able to discover lead-lag relationships between behaviors that can illuminate questions of social influence, collaborative interaction, and other questions about interactive behavioral patterns, in a way that is not possible with static analysis methods [21]. In our research, the GroupAnalyzer system has proven to be a fundamental instrument in achieving this deeper understanding.

References

[1] CSCW (1986). *Proceedings of the Conference on Computer-Supported Cooperative Work*. Austin, TX.

[2] CSCW (1988). *Proceedings of the Conference on Computer-Supported Cooperative Work*. Portland, OR.

[3] Greif, I. (Ed.). (1988). *Computer-Supported Cooperative Work: A Book of Readings*. San Mateo, CA: Morgan Kaufmann Publishers.

[4] Cook, P. G., Ellis, C. A., & Rein, G. L. (1988). *Meetings Research - A Nick Retrospective* (Technical Rep. No. STP-048-88). Austin, TX: MCC.

[5] Bales, R. F. (1950). *Interaction Process Analysis: A Method for the Study of Small Groups*. Cambridge, MA: Addison-Wesley.

[6] Bales, R. F. (1983). SYMLOG: A practical approach to the study of groups. In H. H. Blumberg, A. P. Hare, V. Kent, & M. Davies (Eds.), *Small Groups and Social Interaction* (pp. 499-523). New York, NY: Wiley.

[7] Bales, R. F. (1985). The new field theory in social psychology. *International Journal of Small Group Research, 1*, 1-18.

[8] Bales, R. F., & Cohen, S. P. (1979). *SYMLOG: A System for the Multiple Level Observation of Groups*. New York, NY: Free Press.

[9] McGrath, J. E. (1984). *Groups: Interaction and Performance*. Englewood Cliffs, NJ: Prentice-Hall.

[10] Mantei, M. M. (1988). Capturing the Capture Lab Concepts: A Case Study in the Design of Computer-Supported Meeting Environments. In *Proceedings of the Conference on Computer-Supported Cooperative Work*. Portland, OR.

[11] Polley, R. B., Hare, A. P., & Stone, P. S. (1988). *The SYMLOG Practitioner: Applications of Small Group Research*. New York, NY: Praeger.

[12] Bandura, A. (1986). *Social Foundations of Thought and Action*. Englewood Cliffs, NJ: Prentice-Hall.

[13] Bakeman, R., & Gottman, J. M. (1986). *Observing Interaction: An Introduction to Sequential Analysis*. Cambridge, MA: Cambridge University Press.

[14] Iacobucci, D., & Wasserman, S. (1988). A General Framework for the Statistical Analysis of Sequential Dyadic Interaction Data. *Psychological Bulletin, 103*, 379-390.

[15] McGrath, J. E., & Kelly, J. R. (1986). *Time & Human Interaction: Toward a Social Psychology of Time*. New York, NY: The Guilford Press.

[16] Newtson, D., Hairfield, J., Bloomingdale, J., & Cutino, S. (1987). The structure of action and interaction. *Social Cognition, 5*, 191-237.

[17] Gottman, J. M. (1981). *Time Series Analysis: A Comprehensive Introduction for Social Scientists*. Cambridge, MA: Cambridge University Press.

[18] Chatfield, C. (1984). *The Analysis of Time Series: An Introduction* (3rd ed.). London & New York: Chapman and Hall.

[19] Shumway, R. H. (1988). *Applied Statistical Time Series Analysis.* Englewood Cliffs, NJ: Prentice Hall.

[20] Faraone, S. V., & Dorfman, D. D. (1987). Lag sequential analysis: Robust statistical methods. *Psychological Bulletin, 101*, 312-323.

[21] Losada, M., Sánchez, P., & Noble, E. E. (1989). *Effect of Collaborative Technology and Feedback on Interactive Sequences in Meetings* (Tech. Rep. No. CMI-89-019). Ann Arbor, MI: EDS Center for Machine Intelligence.

Acknowledgements

We gratefully acknowledge Marjorie Horton, Joyce Massey, Paul Scott, Ashby Woolf and two anonymous reviewers for useful comments on an earlier draft of this paper.

We are also indebted to the University of Michigan team of researchers and coders who helped us test GroupAnalyzer, and to Pedro Sánchez for help with the final manuscript.