College of Information Science and Technology

Drexel E-Repository and Archive (iDEA)
http://idea.library.drexel.edu/

Drexel University Libraries
www.library.drexel.edu

Please direct questions to archives@drexel.edu

Thread-based analysis of patterns of collaborative interaction in chat

Murat Cakir, Fatos Xhafa, Nan Zhou and Gerry Stahl
Drexel University
Philadelphia, PA 19104, USA

Abstract. In this work we present a thread-based approach for analyzing synchronous collaborative math problem solving activities. Thread information is shown to be an important resource for analyzing collaborative activities, especially for conducting sequential analysis of interaction among participants of a small group. We propose a computational model based on thread information which allows us to identify patterns of interaction and their sequential organization in computer-supported collaborative environments. This approach enables us to understand important features of collaborative math problem solving in a chat environment and to envisage several useful implications for educational and design purposes.

1. Introduction

The analysis of fine-grained patterns of interaction in small groups is important for understanding collaborative learning [1]. In distance education, collaborative learning is generally supported by asynchronous threaded discussion forums and by synchronous chat rooms. Techniques of interaction analysis can be borrowed from the science of conversation analysis (CA), adapting it for the differences between face-to-face conversation and online discussion or chat. CA has emphasized the centrality of turn-taking conventions and of the use of adjacency pairs (such as question-answer or offer-response interaction patterns). In informal conversation, a given posting normally responds to the previous posting. In threaded discussion, the response relationships are made explicit by a note poster, and are displayed graphically. The situation in chat is more complicated, and tends to create confusions for both participants and analysts.

In this paper, we present a simple mathematical model of possible response structures in chat, discuss a program for representing those structures graphically and for manipulating them, and enumerate several insights into the structure of chat interactions that are facilitated by this model and tool. In particular, we show that fine-grained patterns of collaborative interaction in chat can be revealed through statistical analysis of the output from our tool. These patterns are related to social, communicative and problem-solving interactions that are fundamental to collaborative learning group behavior.

Computer-Supported Collaborative Learning (CSCL) research has mainly focused on analyzing content information. Earlier efforts aimed at identifying interaction patterns in chat environments such as Soller et al. [2] were based on the ordering of postings generated by the system. A naïve sequential analysis solely based on the observed ordering of postings without any claim about their threading might be misleading due to artificial turn orderings produced by the quasi-synchronous chat medium [3], particularly in groups larger than two or three [4].

In recent years, we have seen increasing attention on thread information, yet most of this research is focused on asynchronous settings ([5], [6], [7], [8], [9]). Jeong [10] and Kanselaar et al. [11], for instance, use sequential analysis to examine group interaction in asynchronous

threaded discussion. In order to do a similar analysis of chat logs, one has to first take into account the more complex linking structures.

Our approach makes use of the thread information of the collaboration session to construct a graph that represents the flow of interaction, with each node denoting the content that includes the complete information from the recorded transcript. By traversing the graph, we mine the most frequently occurring dyad and triad structures, which are analyzed more closely to identify the patterns of collaboration and sequential organization of interaction under such specific setting. The proposed thread-based sequential analysis is robust and scalable, and thus can be applied to study synchronous or asynchronous collaboration in different contexts.

The rest of the paper is organized as follows: Section 2 introduces the context of the research, including a brief introduction of the Virtual Math Teams project, and the coding scheme on which the thread-based sequential analysis is based. Section 3 states the research questions we want to investigate. In Section 4 we introduce our approach. We present interesting findings and discuss them to address our research questions and to envisage several useful implications for educational and design purposes in Section 5. Section 6 concludes this work and points to future research.

2. Context of the Research

The VMT Project and Data Collection

The Virtual Math Teams (VMT) project at Drexel University investigates small group collaborative learning in mathematics. In this project an experiment is being conducted, called *powwow*, which extends The Math Forum's (mathforum.org) "*Problem of the Week (PoW)*" service. Groups of 3 to 5 students in grades 6 to 11 collaborate online synchronously to solve math problems that require reflection and discussion. AOL's Instant Messenger software is used to conduct the experiment in which each group is assigned to a chat room. Each session lasts about one to one and a half hour. The *powwow* sessions are recorded as chat logs (transcripts) with the handle name (the participant who made the posting), timestamp of the posting, and the content posted (see Table 1). The analysis conducted in this paper is based on 6 of these sessions. In 3 of the 6 sessions the math problem was announced at the beginning of the session, whereas in the rest the problem was posted on the Math Forum's web site in advance.

Table 1: Description of the coded chat logs.

PoW-wow Session #	Facilitator	Members	Number of Postings	PoW Name	Announced Before?
1	MUR	PIN, GOR, REA, MCP	334	Finding CE	No
2a	GER	AVR, PIN, SUP, OFF	724	Equilateral Triangle Areas	No
2b	MUR	MCP, AH3, REA	204	Equilateral Triangle Areas	No
9	POW	EEF, AME, AZN, LIF, FIR	715	Making Triangles	Yes
10	MFP	AME, FIR, MCP	582	The Perimeter of an Octagon	Yes
18	MFP	AME, KOH, KIL, ROB	488	A Tangent Square and Circle	Yes

Coding Scheme

Both quantitative and qualitative approaches are employed in the VMT project to analyze the transcripts in order to understand the interaction that takes place during collaboration within this particular setting. A coding scheme has been developed in the VMT project to quantitatively analyze the sequential organization of interactions recorded in a chat log. The unit of analysis is defined as one posting that is produced by a participant at a certain point of time and displayed as a single posting in the transcript.

The coding scheme includes nine distinct dimensions, each of which is designed to capture a certain type of information from a different perspective. They can be grouped into two main categories: one is to capture the content of the session whereas another is to keep track of the threading of the discussion, that is, how the postings are linked together. Among the content-based dimensions, conversation and problem solving are two of the most important ones which

code the conversational and problem solving content of the postings. Related to these two dimensions are the Conversation Thread and the Problem Solving Thread, which provide the linking between postings, and thus introduce the relational structure of the data. The conversation thread also links fragmented sentences that span multiple postings. The problem solving thread aims to capture the relationship between postings that relate to each other by means of their mathematical content or problem solving moves (see Figure 1).

Line #	Handle	Statement	Time	Conversation Thread	Conversation	Problem Solving	Problem Solving
45	AVR	Okay, I think we should start with the formula for the area of a triangle	8:21:46		Offer		Strategy
46	SUP	ok	8:22:17	45	Follow	45	
47	AVR	A = 1/2bh	8:22:28		Offer	45	Perform
48	AVR	I believe	8:22:31	47	Extension	47	
49	PIN	yes	8:22:35	47	Setup	47	
50	PIN	i concue	8:22:37	49	Agree	49	Check
51	PIN	concur*	8:22:39	50	Repair Typing		
52	AVR	then find the area of each triangle	8:22:42		Offer	45	Strategy
53	AVR	oh, wait	8:22:54		Regulation		
54	SUP	the base and heigth are 9 and 12 right?	8:23:03		Request		Orientation
55	AVR	no	8:23:11	54	Setup	54	
56	SUP	o	8:23:16		No Code		
57	AVR	that's two separate triangles	8:23:16	55	Critique	55	Reflect
58	SUP	ooo	8:23:19	55	Setup	55	
59	SUP	ok	8:23:20	58	Follow	58	

Figure 1: A coded excerpt from Pow2a.

Each dimension has a number of subcategories. The coding is done manually by 3 trained coders independently after strict training assuring a satisfactory reliability. This paper is based on 4 dimensions only; namely the conversation thread, conversation dimension, problem solving thread, and problem solving dimension.

3. Research Questions

In this explorative study we will address the following research questions:

Research Question 1: What patterns of interaction are frequently observed in a synchronous, collaborative math problem solving environment?

Research Question 2: How can patterns of interaction be used to identify: (a) each member's level of participation; (b) the distribution of contributions among participants; and, (c) whether participants are organized into subgroups through the discussion?

Research Question 3: What are the most frequent patterns related to the main activities of the math problem solving? How do these patterns sequentially relate to each other?

Research Question 4: What are the (most frequent) minimal building blocks observed during "local" interaction? How are these local structures sequentially related together yielding larger interactional structures?

4. The Computational Model

We have developed software to analyze significant features of online chat logs. The logs must first be coded manually, to specify both the local threading connections and the content categories. When a spreadsheet file containing the coded transcript is given as input, the program generates two graph-based internal representations of the interaction, depending on the conversation and problem solving thread dimensions respectively. In this representation each posting is treated as a node object, containing a list of references pointing to other nodes according to the corresponding thread. Moreover, each node includes additional information about the corresponding posting, such as the original statement, the author of the posting, its timestamp, and the codes assigned in other dimensions. This representation makes it possible to study various different *sequential patterns*, where sequential means that postings involved in

the pattern are linked according to the thread, either from the perspective of participants who are producing the postings or from the perspective of coded information.

After building a graph representation, the model performs traversals over these structures to identify frequently occurring sub-structures within each graph, where each sub-structure corresponds to a sequential pattern of interaction. Sequential patterns having different features in terms of their size, shape and configuration type are studied. In a generic format dyads of type C_i-C_j, and triads of type C_i-C_j-C_k where $i<j<k$ are examined in an effort to get information about the local organization of interaction. In this representation C_i stands for a variable that can be replaced by a code or author information. The ordering given by $i<j<k$ refers to the ordering of nodes by means of their relative positions in the transcript. It should be noted that a posting represented by C_j can only be linked to previous postings, say C_i where $i<j$. In this notation the size of a pattern refers to the number of nodes involved in the pattern (e.g. the size is 2 in the case of C_i-C_j). Initially the size is limited to dyads and triads since they are more likely to be observed in a chat environment involving 3 to 5 participants. Nonetheless, the model can capture patterns of arbitrary size whenever necessary. The shape of the pattern refers to the different combinations in which the nodes are related to each other. For instance, in the case of a triad like C_i-C_j-C_k there are two possible type configurations: (a) if C_i is linked to C_j and C_j is linked to C_k , then we refer to this structure as *chain* type; (b) if C_i is linked to C_j and C_i is linked to C_k, then we refer to this structure as *star* type. The dyadic and triadic patterns identified this way reveal information about the local organization of interaction. Thus, these patterns can be considered as the fundamental building blocks of a group's discussion, whose combination would give us further insights on the sequential unfolding of the whole interaction.

The type of the configuration is determined by the information represented by each variable C_i. A variable C_i can be replaced by the author name, the conversation code, the problem solving code, or a combination of conversation and problem solving codes. This flexibility makes it possible to analyze patterns linking postings by means of their authors, and the codes they receive from the conversational or problem solving dimension.

As shown in Table 1, the maximum number of chat lines contained in a transcript in our data repository is about 700 lines, and we analyzed a corpus containing 6 such transcripts for this explorative study. Thus, in this study the emphasis is given to ways of revealing relevant patterns of collaborative interaction from a given data set. Nonetheless, we take care of efficiency issues while performing the mining task. Moreover, there exist efficient algorithms designed for mining frequent substructures in large graphs ([12], [13], [14]), which can be used to extend our model to process larger data sets.

5. Results and Discussion

In this section we show how the computational model presented in this work enables us to shed light on the research questions listed in Section 3.

5.1 Local Interaction Patterns
In order to identify the most frequent local interaction patterns of size 2 and 3, our model performs traversals of corresponding lengths and counts the number of observed dyads and triads. The model can classify these patterns in terms of their contributors, in terms of conversation or problem solving codes, or by considering different combinations of these attributes (e.g. patterns of author-conversation pairs). The model outputs a dyad percentage matrix for each session in which the $(i,j)^{th}$ entry corresponds to the percentage that C_i is followed by C_j during that session. For example, a percentage matrix for dyads based on conversation codes is shown in Table 2. In addition to this, a row-based percentage matrix is computed to depict the local percentage of any dyad C_i-C_j among all dyads beginning with C_i.

Table 3 shows a row-based percentage matrix for the conversation dyads. Similarly, the model also computes a list of triads and their frequencies for each session.

5.2 Frequent Conversational Patterns

For the conversational dyads we observed that there are a significant number of zero-valued entries on all six percentage matrices. This fact indicates that there are strong causal relationships between certain pairs of conversation codes. For instance, the event that an *Agree* statement is followed by an *Offer* statement is very unlikely due to the fact that the *Agree-Offer* pair has a zero value in all 6 matrices. By the same token, non-zero valued entries corresponding to a pair C_i-C_j suggests which C_i variables are likely to produce a reply of some sort. Moreover, C_j variables indicate the most likely replies that a conversational action C_i will get. This motivated us to call the most frequent C_i-C_j pairs as *source-sink* pairs, where the source C_i most likely solicits the action C_j as the next immediate reply.

The most frequent conversational dyads in our sample turned out to be *Request-Response* (16%, 7%, 9%, 9%, 10%, 8% for the 6 powwows respectively), *Response-Response* (12%, 5%, 2%, 4%, 10%, 11%) and *State-Response* (8%, 6%, 4%, 2%, 5%, 16%) pairs. In our coding scheme conversational codes *State, Respond, Request* are assigned to those statements that belong to a general discussion, while codes such as *Offer, Elaboration, Follow, Agree, Critique* and *Explain* are assigned to statements that are specifically related to the problem solving task. Thus, the computations show that a significant portion of the conversation is devoted to topics that are not specifically about math problem solving. In addition to these, dyads of type *Setup-X* (8%, 14%, 12%, 2%, 3%, 4%) and *X-Extension* (14%, 15%, 9%, 7%, 9%, 6%) are also among the most frequent conversational dyads. In compliance with their definitions, *Setup* and *Extension* codes are used for linking fragmented statements of a single author that span multiple chat lines. In these cases the fragmented parts make sense only if they are considered together as a single statement. Thus, only one of the fragments is assigned a code revealing the conversational action of the whole statement, and the rest of the fragments are tied to that special fragment by using *Setup* and *Extension* codes. The high percentage of *Setup-X* and *X-Extension* dyads shows that some participants prefer to interact by posting fragmented statements during chat. The high percentage of fragmented statements strongly affects the distribution of other types of dyadic patterns. Therefore, a "pruning" option is included in our model to combine these fragmented statements into a single node to reveal other source-sink relationships.

5.3 Handle Patterns

Frequent dyadic and triadic patterns based on author information can be very informative for making assessments about each participant's level and type of participation. For instance, Table 4 contrasts two groups, namely Pow2a and Pow2b (hereafter, group A and B, resp.) that worked on the same math problem in terms of their author-dyad percentages. In both matrices an entry *(i,j)* corresponds to the percentage of the event that the postings of participant *i* were conversationally related to the postings of participant *j* during the session. For the non-pruned matrices, entries on the diagonal show us the percentage that the same participant either extended or elaborated his/her own statement. For the pruned matrices the "noise" introduced by the fragmented statements is reduced by considering them together as a single unit. In the pruned case diagonal entries correspond to elaboration statements following a statement of the same participant.

The most striking difference between the two groups, after pruning, is the difference between the percentage values on the diagonal: 10% for group A and 30% for group B. The percentages of most frequent triad patterns[1] show a similar behavior. The percentage of triads having the same author on all 3 nodes (e.g. AVR-AVR-AVR) is 15% for group A, and 42% for group B.

---

[1] For more results and our coding scheme refer to http://mathforum.org/wiki/VMT?ThreadAnalResults.

The pattern we see in group B is called an elaboration, where a member takes an extended turn. The pattern in group A indicates group exploration where the members collaborate to co-construct knowledge and turns rarely extend over multiple pruned nodes.

Patterns that contain the same author name on all its nodes are important indicators of individual activity, which typically occurs when a group member sends repeated postings without referring to any other group member. We call this elaboration, where one member of the group explains his/her ideas The high percentage of these patterns can be considered as a sign of separate threads in ongoing discussion, which is the case for group B. Moreover, there is an anti-symmetry between MCP's responses to REA's comments (23%) versus REA's responses to MCP's comments (14%). This shows that REA attended less to MCP's comments, than MCP to REA's messages. In contrast, we observe a more balanced behavior in group A, especially between AVR-PIN (17%, 18%) and AVR-SUP (13%, 13%). Another interesting pattern for group A is that the balance with respect to AVR does not exist between the pair SUP-PIN. This suggests that AVR was the dominant figure in group A, who frequently attended to the other two members of the group. To sum up, this kind of analysis points out similar results concerning roles and prominent actors as addressed by other social network analysis techniques.

Table 2: Conversation dyads

Powwow_2a Conversation Dyads Percentage Matrix	State	Offer	Reqest	Regulation	Repair Typing	Response	Follow	Elaboration	Exension	Setup	Agree	Disagree	Critique	Explain	No code
State	0	0	0	0	0	6	0	0	4	1	0	0	0	0	0
Offer	0	0	0	0	0	0	2	1	3	1	1	0	0	0	0
Reqest	0	0	0	0	0	7	0	0	3	1	0	0	0	1	0
Regulation	0	0	0	0	0	4	0	0	1	0	0	0	0	0	0
Repair Typing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Response	0	0	0	0	0	5	0	0	1	1	0	0	0	0	0
Follow	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Elaboration	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Exension	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Setup	1	3	2	2	0	7	1	1	3	0	1	0	0	0	0
Agree	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Disagree	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Critique	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Explain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
No code	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The %s are computed over all pairs

Table 3: Row based distribution of conversation dyads

Powwow_2a Conversation Dyads_Row Percentage Matrix	State	Offer	Reqest	Regulation	Repair Typing	Response	Follow	Elaboration	Exension	Setup	Agree	Disagree	Critique	Explain	No code
State	0	0	0	1	1	45	0	0	35	9	3	0	1	0	0
Offer	0	0	0	0	4	2	23	15	26	8	10	4	4	0	0
Reqest	0	0	3	3	1	43	0	0	22	8	1	4	0	11	0
Regulation	0	4	8	4	4	60	0	0	16	4	0	0	0	0	0
Repair Typing	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
Response	0	0	5	0	2	55	0	0	14	17	0	0	2	0	0
Follow	0	0	0	0	0	0	33	0	0	50	0	0	0	16	0
Elaboration	0	12	0	0	25	0	0	12	12	0	25	0	12	0	0
Exension	0	50	0	0	0	0	0	0	0	0	0	50	0	0	0
Setup	4	11	9	7	0	27	6	6	11	2	3	2	2	0	0
Agree	0	0	0	0	20	40	0	0	20	20	0	0	0	0	0
Disagree	0	0	0	0	0	0	20	0	20	40	0	20	0	0	0
Critique	0	0	6	6	0	20	20	26	6	13	0	0	0	0	0
Explain	0	0	11	0	0	22	22	0	33	0	0	0	11	0	0
No code	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The %s are computed separately for each row

Dyadic and triadic patterns can also be useful in determining which member was most influential in initiating discussion during the session. For a participant *i*, the sum of row percentages *(i,j)* where $i \neq j$ can be used as a metric to see who had more initiative as compared to other members. The metric can be improved further by considering the percent of triads initiated by user *i*. For instance, in group A the row percentages are 31%, 22%, 20% and 2% for AVR, PIN, SUP and OFF respectively and the percentage of triads initiated by each of them is 41%, 29%, 20% and 7%. These numbers show that AVR had a significant impact in initiating conversation. In addition to this, a similar metric for the columns can be considered for measuring the level of attention a participant exhibited by posting follow up messages to other group members.

5.4 Problem Solving Patterns

A similar analysis of dyadic and triadic patterns can be used for making assessments about the local organization of a group's problem solving actions. The problem solving data produced by our model for groups A and B will be used to aid the following discussion in this section. Table 4 displays both groups' percentage matrices for problem solving dyads.

Before making any comparisons between these groups, we briefly introduce how the coding categories are related to math problem solving *activities*. In this context a problem solving activity refers to a set of successive math problem solving actions. In our coding scheme,

Orientation, *Tactic* and *Strategy* codes refer to the elements of a certain activity in which the group engages in understanding the problem statement and/or proposes strategies for approaching it. Next, a combination of *Perform* and *Result* codes signal actions that relate to an execution activity in which previously proposed ideas are applied to the problem. *Summary* and *Restate* codes arise when the group is in the process of helping a group member to catch up with the rest of the group and/or producing a reformulation of the problem at hand. Further, *Check* and *Reflect* codes capture moves where group members reflect on the validity of an overall strategy or on the correctness of a specific calculation; they do not form an activity by themselves, but are interposed among the activities described before

Table 4: Handle & Problem Solving Dyads for Pow2a and Pow2b

Pow 2a %	SYS	PIN	OFF	SUP	AVR	GER
SYS	0	0	0	0	0	0
PIN	0	11	1	2	11	0
OFF	0	0	3	0	1	0
SUP	0	3	0	10	9	0
AVR	0	11	1	9	16	0
GER	0	0	1	0	1	0

(pruned) Pow 2a %	SYS	PIN	OFF	SUP	AVR	GER
SYS	0	0	0	0	0	0
PIN	0	4	1	3	18	0
OFF	0	0	0	0	2	0
SUP	0	6	1	0	13	0
AVR	0	17	1	13	6	1
GER	0	1	1	0	1	0

Pow 2b %	SYS	MCP	AH3	REA	MUR
SYS	0	0	0	0	0
MCP	0	17	0	10	8
AH3	0	1	6	2	0
REA	0	18	2	20	3
MUR	0	2	1	1	3

(pruned) Pow 2b %	SYS	MCP	AH3	REA	MUR
SYS	0	0	0	0	0
MCP	0	14	0	14	10
AH3	0	1	1	3	0
REA	0	23	3	15	4
MUR	0	3	1	1	0

Pow2a Pbsol_dyads %_Matrix	Orientation	Strategy	Tactic	Perform	Check	Restate	Summarize	Reflect	Result
Orientation	1	0	3	0	2	1	0	1	0
Strategy	0	1	0	1	0	0	0	2	0
Tactic	0	0	3	2	4	3	0	7	2
Perform	0	0	0	4	4	0	0	0	1
Check	0	0	0	0	6	0	0	0	0
Restate	0	0	0	0	2	0	0	2	0
Summarize	0	0	0	0	0	0	0	0	0
Reflect	1	0	3	0	7	3	0	9	1
Result	0	0	0	1	8	1	0	2	0

Pow2a Pbsol_dyads Row_% Matrix	Orientation	Strategy	Tactic	Perform	Check	Restate	Summarize	Reflect	Result
Orientation	12	0	37	0	25	12	0	12	0
Strategy	0	25	0	25	0	0	0	50	0
Tactic	0	0	15	10	20	15	0	30	10
Perform	0	0	0	44	44	0	0	0	11
Check	0	0	0	0	100	0	0	0	0
Restate	0	0	0	0	50	0	0	50	0
Summarize	0	0	0	0	0	0	0	0	0
Reflect	4	0	13	0	27	13	0	36	4
Result	0	0	0	9	63	9	0	18	0

Pow2b Pbsol_dyads %_Matrix	Orientation	Strategy	Tactic	Perform	Check	Restate	Summarize	Reflect	Result
Orientation	2	0	1	5	0	0	0	0	1
Strategy	0	0	0	1	0	0	0	0	1
Tactic	0	0	0	6	0	0	0	2	1
Perform	0	0	0	12	10	3	0	3	8
Check	0	0	0	1	0	0	0	6	0
Restate	0	0	0	0	0	0	0	0	0
Summarize	0	0	0	0	0	0	0	0	0
Reflect	0	0	0	0	0	0	0	5	0
Result	1	1	2	10	3	1	0	0	3

Pow2b Pbsol dyads Row_% Matrix	Orientation	Strategy	Tactic	Perform	Check	Restate	Summarize	Reflect	Result
Orientation	25	0	12	50	0	0	0	0	12
Strategy	0	0	0	50	0	0	0	0	50
Tactic	0	0	0	62	0	0	0	25	12
Perform	0	0	0	32	25	9	0	9	22
Check	0	0	0	16	0	0	0	83	0
Restate	0	0	0	0	0	0	0	0	0
Summarize	0	0	0	0	0	0	0	0	0
Reflect	0	0	0	0	0	0	0	100	0
Result	5	5	10	42	15	5	0	0	15

SYS refers to system messages. GER and MUR are facilitators of the groups.

Given this description, we use the percentage matrices (see Table 4) to identify what percent of the overall problem solving effort is devoted to each activity. For instance, the sum of percentage values of the sub-matrix induced by the columns and rows of *Orientation, Tactic, Strategy, Check* and *Reflect* codes takes up 28% of the problem solving actions performed by the group A, whereas this value is only 5% for group B. This indicates that group A put more effort in developing strategies for solving the problem. When we consider the sub-matrix induced by *Perform, Result, Check* and *Reflect*, the corresponding values are 21% for group A and 50% for group B. This signals that group B spent more time on executing problem solving steps. Finally, the values of the corresponding sub-matrix induced by *Restate, Summarize, Check*, and *Reflect* codes adds up to 7% for group A and 0% for B, which hints at a change in orientation of group A's problem solving activity. The remaining percentage values excluded by the sub-matrices belong to transition actions in between different activities.

5.5 Maximal Patterns

The percentage values presented in the previous section indicate that groups A and B exhibited significantly different local organizations in terms of their problem solving activities. In order to make stronger claims about the differences at a global level one needs to consider the unfolding of these local events through the whole discussion. Thus, analyzing the sequential unfolding of local patterns is another interesting focus of investigation which will ultimately yield a "global" picture of a group's collaborative problem solving activity. For instance, given the operational descriptions of problem solving activities in Subsection 5.4, we observed the following sequence of local patterns in group A. First, the group engaged in a problem orientation activity in which they identified a relevant sub-problem to work on. Then, they performed an execution activity on the agreed strategy by making numerical calculations to

solve their sub-problem. Following this discussion, they engaged in a reflective activity in which they tried to relate the solution of the sub-problem to the general problem. During their reflection they realized they made a mistake in a formula they used earlier. At that point the session ended, and the group failed to produce the correct answer to their problem. On the other hand, the members of group B individually solved the problem at the beginning of the session without specifying a group strategy. They spent most of the remaining discussion revealing their solution steps to each other.

6. Conclusion and Ongoing Research

In this work we have shown how thread information can be used to identify the most frequent patterns of interaction with respect to various different criteria. In particular, we have discussed how these patterns can be used for making assessments about the organization of interaction in terms of each participant's level of participation, the conversational structure of discussion as well as the problem solving activities performed by the group. Our computations are based on an automated program which accepts a coded chat transcript as input, and performs all necessary computations in an efficient way.

In our ongoing research we are studying other factors that could influence the type of the patterns and their frequencies, such as the group size, the type of the math problem under discussion, etc. Moreover, we are investigating whether the interaction patterns and the problem solving phases reveal information about the type of the organization of the interaction, e.g. exploratory vs. reporting work. Finally, we will be using our data to feed a statistical model and thus study the research questions from a statistical perspective. We are also planning to extend the existing computational model to support XML input in order to make the model independent of the specific features introduced by a coding scheme.

References

[1] Stahl, G. (2006). Group Cognition: Computer Support for Building Collaborative Knowledge. *Cambridge, MA: MIT Press.*
[2] Soller, A., and Lesgold, A. (2003) A computational approach to analyzing online knowledge sharing interaction. *Proceedings of AI in Education 2003, Sydney, Australia, 253-260.*
[3] Garcia, A., and Jacobs, J.B. (1998). The interactional organization of computer mediated communication in the college classroom. *Qualitative Sociology, 21(3), 299-317.*
[4] O'Neil, J., and Martin, D. (2003). Text chat in action. *Proceedings of the international ACM SIGGROUP conference on Supporting group work, Sanibel Island, Florida, USA, 40-49.*
[5] Smith, M., Cadiz, J., and Burkhalter, B. (2000) Conversation Trees and Threaded Chats, *Proceedings of the 2000 ACM conference on Computer supported cooperative work, Philadelphia, PA, USA, 97-105.*
[6] Popolov, D., Callaghan, M., and Luker, P. (2000). Conversation Space: Visualising Multi-threaded Conversation. *Proceedings of the working conference on Advanced visual interfaces, Palermo,Italy,246-249*
[7] King, F.B., and Mayall, H.J. (2001) Asynchronous Distributed Problem-based Learning, *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'01), 157-159.*
[8] Tay, M.H., Hooi, C.M., and Chee, Y.S. (2002) Discourse-based Learning using a Multimedia Discussion Forum. *Proceedings of the International Conference on Computers in Education (ICCE'02), IEEE, 293.*
[9] Venolia, G.D. and Neustaedter, C. (2003) Understanding Sequence and Reply Relationships within Email Conversations: A mixed-model visualization. *Proceedings of SIGCHI'03, Ft. Lauderdale, FL, USA,361-368.*
[10] Jeong, A.C. (2003). The Sequential Analysis of Group Interaction and Critical Thinking in Online Threaded Discussion. *The American Journal of Distance Education, 17(1), 25-43.*
[11] Kanselaar, G., Erkens, G., Andriessen, J., Prangsma, M., Veerman, A., and Jaspers, J. (2003) Designing Argumentation Tools for Collaborative Learning. Book chapter of *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making, Kirschner, P.A., et al. eds, Springer.*
[12] Inokuchi, A., Washio, T. and Motodam H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. *Proceedings of PKDD 2000, Lyon, France, 13-23.*
[13] Kuramochi,M. and Karypis, G. (2001). Frequent subgraph discovery. *Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, California, USA, 313-320.*
[14] Zaki, M.J. (2002). Efficiently mining frequent trees in a forest. *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, Canada, 71-80.*