Mining patterns of events in students' teamwork data

Judy Kay, Nicolas Maisonneuve, Kalina Yacef
School of Information Technologies
University of Sydney
{judy, nicolas,kalina}@it.usyd.edu.au

Osmar Zaïane
Department of Computing Science
University of Alberta
zaiane@cs.ualberta.ca

Abstract. It is difficult, but very important, to learn to work effectively as part of a team. One potentially invaluable source of information about the success, or problems, in the way that teams learn can be drawn from the electronic traces of their collaborations. The paper describes data mining of student group interaction data to identify significant sequences of activity. Our goal is to build tools that can flag interaction sequences indicative of problems, so that we can use these to assist student teams in early recognition of problems. We also want tools that can identify patterns that are markers of success so that these might indicate improvements during the learning process. Our first challenge is to transform the raw data available in large quantities, preprocessing it into a suitable alphabet for use in data mining. Then, we need data mining algorithms that can properly account for the temporal nature of the data and the character of group interaction. We envisage that this may involve a two way process, where theories of effective group behaviour can drive the data mining and, in the opposite direction, that the data mining should provide results that are meaningful to groups wishing to improve their effectiveness. We report the results of our work in the context of a semester long software development project course.

Keywords: Educational Data Mining, frequent pattern mining, collaborative learning.

INTRODUCTION

Group work has an important role in many aspects of life, both at work and elsewhere. This makes it important for people to learn to be effective team members. In cases where the teams make substantial use of electronic media to support their operation, they may produce substantial electronic traces of the group operation and the interaction between the team members. It is extremely appealing to exploit such data, extracting salient features so that groups can be advised on how well they are doing and how they might improve their performance. In a formal educational context, these traces of student interaction have potential as an important source of information to teachers who want to guide students to learn to improve their skills in group work and to evaluate the value of various learning interventions. The mining of such data is now an important research direction as can be shown by the recent workshops on Educational Data Mining held at ITS, AIED and AAAI conferences (Beck, 2005, Beck, 2004, Choquet, et al., 2005). Studies were done to analyse how learning groups collaborate in a shared workspace (Barros and Verdejo, 1999) or using a structured conversational interface (Soller, 2004). Our work is fundamentally aimed at assessing group work processes and providing feedback to students about how they could improve them.

Software development is commonly a group activity. Unsurprisingly, given that software teams need to make use of computers for their core task, it is natural to provide closely linked tools for managing versions of software, supporting group communication and planning as well as task allocation and scheduling against deadlines. Our work has been conducted in the context of a semester long software development project course where teams of five to seven students create a substantial software artifact as well as reports and presentations.

Our student teams collaborate using several media. Most importantly, they have face-to-face meetings, usually at least twice a week. These meetings play a critical role in the group co-ordination processes. Since they may not make use of electronic support, we have no direct access to data about activities during these. Teams probably make considerable use of electronic communication with media that we also have no access to. For example, they may use email, instant messaging communication, telephone conversations, SMS.

However, we do have access to a rich and interesting source of user activity trace data because the groups are required to use TRAC, (http://www.edgewall.com/trac/) an open source tool designed for use in software development projects.

This makes TRAC an exemplar of the type of the electronic tools that learners will use as a normal part of their learning activity, even though they are not learning tools as such. Therefore the work presented here does not analyse data from a learning system, although the aim of this analysis is to support students' learning. Indeed, it would be a significant gain if we can create effective tools that can mine the data from such tools, to provide insights into how effectively students are learning and how we can complement such tools with artifacts that help them learn better.

This paper describes our preliminary work on mining data from TRAC, with the aim of identifying patterns that characterise successful groups from less successful ones. The next section describes the TRAC system and the context of its use in our course. We then explain the data mining framework we constructed, and we then present our initial results before concluding and presenting some of our future directions of work.

CONTEXT OF EXPERIMENT AND USE OF THE TRAC SYSTEM

Students collaborate by sharing tasks via the TRAC system. These tasks are managed by a "Ticket" system; Source code writing tasks are managed by a version control system called "SVN"; Students communicate by means of collaborative web page writing called "Wiki".

The Wiki allows collaborative editing of a set of web pages, all linked from the main page. Any member of the team can edit a page, for example, to offer to meet and help work on the problem described. The value of the Wiki is that all group members can see it. So, for example, if Daniel, who posted a top comment, gets help from Peter, it is important that Peter notes that here so that other group members can see that someone is following up on the problem. We believe that the Wiki is a source of very important information about group interaction and communication. For example, if a group is functioning well, there will be pages on the Wiki with contributions from many team members.

Based upon our own experience, as well as literature on group work and co-ordination (Kay, et al., 2006), we believe that the Wiki should provide valuable data about meaningful patterns of group interaction, both successful and not. We describe some of these as a background to the design of the data cleansing and data mining we report later. It is also important for the other direction we want to explore, where we use theory to define classes of patterns that we would like to look for.

We would also expect that a group which is functioning well, with good leadership, will have some pages where one person is primarily responsible for the activity reported on that page. So one person will make most of the contributions on that page. In the case of leadership functions, it may be the leader who will do this. In this case, we would also expect other team members to contribute to this page since the leader's co-ordination role suggests that they might post proposed action plans, minutes and the like. Then other team members should add comments and correction as work progresses and to acknowledge that they agree with what is posted.

If the group does an effective job of allocating responsibilities, there will be other pages which are mainly created by another single person. So, for example, if the group needs to learn about content-management systems, one person may be allocated the task of doing research on that. Then that person would be the main contributor to that page. However, if the group is functioning well, this information will be read and used by at least some other team members. Moreover, a group that functions well will ensure that communication is maintained overtly and so, we would expect that some other team members would post comments, suggestions, additional information, corrections or even a simple acknowledgement that they had read the material and agreed with it or thought it was very good.

It is possible to extract details of TRAC interactions. These indicate exactly who posted, removed or altered which lines and exactly when they did this. The team members can see just this information from a tab at the interface and we can extract the details of Wiki interaction at this level. Of course, this is not as rich a set of discourse data as one might gain from tagged conversational interfaces, such as used by Soller (Soller, 2004). However, it has the real merit that it is more natural and enables teams to operate exactly as they would for best practice. There is also some potential for exploiting text mining to identify more details of what each Wiki page is about. We are not currently doing this.

Essentially, our Wiki data gives us details of who placed, altered or removed how much text on each page of the Wiki. We regard the page as a set of related conversations and we analyse this to look for patterns of interaction that reflect the health of the group. Moreover, since we have the final grades for the semester, we can use these to look for patterns that were more or less common among teams that performed well: these should be success indicators. Similarly, patterns that were more common for weak groups should be indicators of problem groups.

TRAC also has a ticket system (sometimes called an issues tracking tool). The core idea is that a ticket is created for each task that the team has to do. For example, if the team needs to do research on content-management systems, one person should create a ticket for this task. Often, it is the leader who does this. A ticket is allocated to a person, meaning that that person is supposed to complete it. Team members can add comments on the ticket, reassign it to someone else, close it.

If the ticket system is used well, it provides electronic traces of the way that the group manages itself. For example, the time from the creation of a ticket to its completion should be meaningful. The proportion of tickets that are never closed, reflecting tasks not completed is also important. The details of who creates tickets, who they are allocated to and who closes the tickets are all meaningful.

The third main part of TRAC is an excellent browsing interface to the version control SVN. The essential roles of SVN are to provide a repository for software created by a group. It provides version control and associated functions. Effective teams will make heavy use of SVN for the various versions of all the artifacts that they need to create: code, reports, notes for presentations and the like. We have ready access to details of which team members wrote which parts of each artifact in the SVN repository.

We would expect that effective groups would make heavy use of SVN. We would also expect that, for code, there would generally be few authors of some parts, since it is usual to try to break software development tasks into loosely coupled parts. This enables each individual to work on their own task, without needing much detail of the other parts. We would expect that the patterns of how many different people work on one file in the SVN repository would reflect aspects of effective groups.

The full set of elements of TRAC, Wiki, tickets and SVN, constitute a powerful collection of integrated tools to support groups in software development as well as in group co-ordination and communication. All provide a rich electronic trail of group activity. The remainder of this paper describes our work to mine this data so that we can extract patterns that are meaningful and indicative of successful group interaction.

Data collected from the TRAC system

We collected from the TRAC system all the traces of the students' actions, which we call *events*. A certain amount of information is stored about each event: the author, the date, the resource ID as well as more specific information depending in the nature of the event. There are 3 types of events:

1. Wiki event: generated whenever a student creates a new Wiki page, or added or removed some text on an existing Wiki page. The resource ID here is the name of the Wiki page.
2. Ticket event: generated whenever a student creates a new ticket, closed an existing one, or modified an existing one in any way, eg reassign, changed priority, change severity, added a comment. The resource ID is the ticket number.
3. SVN event: generated whenever a student commits on a set of files, eg modification, addition, deletion. The resource IDs here are the files.

For each of these events, the system stores all the actual content of the action, eg the text added to the Wiki page, the comment stored in the ticket, the lines added or removed from a file.

DATA MINING

We are interested in finding frequent patterns characterizing some aspect of the teamwork. For this purpose, we use a *frequent sequential pattern mining* algorithm, which addresses the problem of discovering frequent sequences in a database with a minimum frequency called *support*. Algorithms for this problem are relevant when the data to be mined has some sequential nature, i.e., when each piece of data is an ordered set of elements, like events in our case. We will first present the algorithm and then explain how we fitted our data to apply it (preprocessing).

Frequent sequential pattern algorithm

The problem was first introduced by Agrawal and Srikant (Agrawal and Srikant, 1995), and since then the goal of sequential pattern mining is to discover all frequent sequences of itemsets in a dataset. In particular, an *itemset* is a non-empty subset of elements from a set C, the item collection, called *Alphabet*. In our case an itemset is simply an item.

A *sequence* is an ordered list of itemsets (i.e items in our case). The number of items in a sequence is the length of the sequence: A sequence of a length k is called a *k*-sequence. The support for a sequence *s* is the number of sequences of which *s* is a subsequence noted *Sup(s)*. A subsequence is not a contiguous; formally, a sequence $a=<a_1,a_2,...,a_n>$ is a *subsequence* of $b=<b_1,b_2,...,b_m>$, if there exist integers $1 \leq i_1 < i_2 < \ldots < i_n \leq m$ such that $a_1=b_{i_1}$, $a_2=b_{i_2,}$ $a_n=b_{i_n}$. So, for example according to Table 1, the sequence *s* <a,d> is a subsequence of S1, S2 and S3 and its support Sup(*s*)=3. The support of <c,d> equals 2 (S2, S3).

Table 1. Example of data used by a sequential pattern mining

SeqID	**Sequence**
S1	<a,b,b,d,c>
S2	<a,c,d>
S3	<a,c,c,d>

The main approaches to sequential pattern mining, namely apriori-based (Srikant and Agrawal, 1996) and pattern-growth methods (Han, et al., 2000, Pei, et al., 2001), are being used as the basis for other pattern mining algorithms. Both algorithms find the same results but the pattern growth method is much faster. Our data is not very large (only ten thousands of events) so the speed performance was not a critical criterion and the algorithm is easier to understand and modify than the pattern-growth based algorithms.

Overview of the Generalized Sequential Pattern Algorithm (GSP)

This algorithm is based on the so called apriori property or antimonotony property which states that *if a sequence is frequent then all its subsequences must be frequent as well.* Based on this heuristic, GSP (Srikant and Agrawal, 1996) adopts a multiple-pass, candidate generation-and-test approach in sequential pattern mining. We have simplified it as we consider sequence of items, not itemsets. The algorithm is outlined as follows:

1) **First pass:** The first pass determines the support of each item, that is, the number of data-sequences that include the item. At the end of the first pass, the algorithm knows which items are frequent, that is, have minimum support
2) **Candidate generation pass:** These frequent sequences are used to generate new potential patterns, called *candidate sequences*. Given the set of all the frequent k-1 sequences found in the previous pass, we generate new k-sequence candidates. Candidates are generated in two steps:
   a. **Join step:** a sequence s1 joins with s2 if the subsequence obtained by dropping the first item of s1 is the same as the subsequence obtained by dropping the last item of s2. For example <a,b,c,d> is a candidate 4-sequence of the 3- sequences <a,b,c> and <b, c, d>.
   b. **Pruning step:** After we remove all the candidate that have a contiguous (k-1) subsequence which support count is less than the minimum support.
3) **Test pass:** All the candidate sequences in a pass have the same *length* (ie number of items). The scan of the database in one pass finds the support for each candidate sequence. All the candidates which support in the database is above the support form the set of the newly found sequential patterns. This set then becomes the seed set for the next pass.

The algorithm terminates when no new sequential pattern is found in a pass, or no candidate sequence can be generated.

Preprocessing

The important step of preprocessing the data consists firstly in organizing the raw data in a more abstract form and obtaining a long, unique, chronological sequence of events that occurred for a group; Secondly in generating meaningful sequences of events; Thirdly in transforming the data in each sequence into items of an alphabet.

Abstraction of Raw data is first transformed into a list of events Ei, which are defined below.

Event = {EventType, Resource, Author, Time}

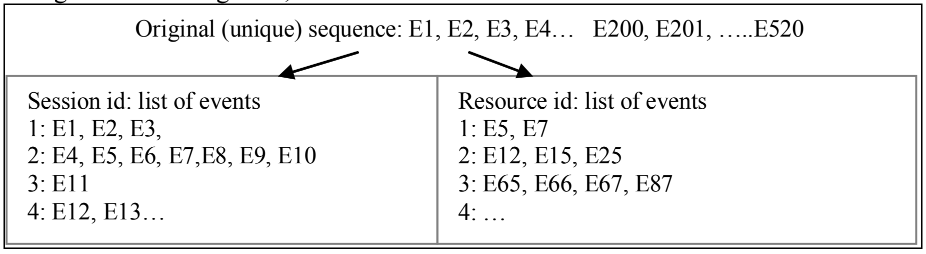
Where: EventType is one of T (for Ticket), S (for SVN), W (for Wiki),
Resource is the identifier of the ticket number, source code file or Wiki page,
Author is the name of the user who did the action,
Time is the absolute time when the event occurred.

Generation of sequences

The original sequence obtained for each group was 285 to 1287 long. We then needed to break down this unique long sequence into several "sequences" of events, in a way that is meaningful for our study. We considered two possible ways to do so:

1. A breakdown per session: ideally, if events were connected to a task, we would classify them per task. Unfortunately that information was not available last semester so we used instead the notion of time-session. That is, events are grouped by the time window they occurred in. As soon as time elapsed between two events is greater than a threshold (we used 7 hours), the next events would belong to a new session. An example is shown on the left side of Figure 1, where session 1 would be composed of E1, E2, E3 and then a gap of more than 7 hours would close that session and create a new one, session 2.

2. A breakdown per resource: all events occurring on a particular resource are grouped together. An example is shown on the right side of Figure 1, where for instance events E5 and E7 occurred on the resource id 1.


Original (unique) sequence: E1, E2, E3, E4… E200, E201, …..E520

Session id: list of events
1: E1, E2, E3,
2: E4, E5, E6, E7,E8, E9, E10
3: E11
4: E12, E13…

Resource id: list of events
1: E5, E7
2: E12, E15, E25
3: E65, E66, E67, E87
4: …

Figure 1. Two ways to break down a long unique sequence into a set of sequences

Transformation of sequences of events into sequence of items
It is easy to see that we cannot extract patterns from the data as shown above because the algorithm would not know how to compare the events together. We need a higher level of abstraction so that we can manipulate sequences of, say, five Wiki events by two team members without them being distinguished because the authors are not the same, the Wiki pages are not the same and the time is different for instance. We need a representation that is expressive enough to capture interesting details but not too complex as it would reduce too much our chances of finding patterns. Since we are interested in the way team members work and interact over the three media, we came up with different alphabets to generate our items. We will present here two of them.

- The first one, used for the breakdown per session, omits the date and the resource, and captures the number of *i* consecutive times a medium X was used by *j* different authors, noted (iXj). When transforming a sequence, an event on a new medium generates a new item in the sequence. An example is shown below:
  <(2T1), (5W3), (2S1),(1W1)>
  This means 2 ticket actions by the same author, followed by 5 Wiki actions made by 3 different authors, followed by 2 SVN actions made by the same author, followed by one Wiki action made by someone. Note that the original sequence would have consisted of 10 events.

- The second one, used for the breakdown per resource, summarises, the number of *i* different events of type X, the number *j* of authors, and the number of days over which *t* the resource was modified. This is noted <iXj, t). An example is shown below:
  <10W5, 2>
  This means that the resource (here a Wiki page) was modified 10 times by 5 people over 2 days.

RESULTS AND DISCUSSION

Analysis of patterns related to sessions

After pre-processing the data by breaking down the sequence per session (as in left of Figure 1), we made two types of analysis to study the collaboration aspect of the extracted sessions.

Sequence pattern analysis (per session)
We run our simplified GSP algorithm and analysed by hand the patterns found, looking for patterns that are indicative of success, being more common in the traces for successful groups and much less present in less able groups, and those that are indicative of problems, being more common in the less able groups.

Such patterns have huge potential for helping groups improve their own performance. For example, if a group appears to have a large proportion of patterns that are associated with problems, we could alert them and their teachers to this. Importantly, if there is a subsequent rise in the proportion of patterns associated with success, we can provide feedback about this.

Of course, we would really hope that the mined patterns can be meaningfully interpreted. Our current preprocessing does make it possible to provide fairly intuitive descriptions of what a pattern means in terms of the traces observed. The team members may well be able to interpret these.

For instance we found, by analyzing the resulting patterns by hand that some of the contrasts between the best group and the average groups were:

- The best group, and to a lesser extent the 2 other good groups, had an interesting and reassuring frequency of patterns where Wiki and SVN events occurred in alternation, such as <1S1><1W1><1S1>, <1S1><2W1><1S1>, <1W1><1S1><1W1><1S1>, and so on, with various lengths. We are hoping that this corresponds to the documentation in the Wiki about the SVN additions, modifications or deletions. These patterns were often not present in the 5 average or worst 2 groups.

- The best group had much more patterns involving several authors, and longer patterns, as opposed to the less able groups.

Analysis of the distribution of the sessions

We also compared the number of different users involved in a given session, as we thought that a high amount of users working in the same session could indicate a good collaboration. We used for this an alphabet that simply counts the number of different authors in the session. For instance the session <E1,E3,E1,E4,E2,E1> with Ei being an event made by the i^{th} author of the group involves 4 authors. We then calculated the frequency of each type of sequence (e.g a session involving only 1 author, 2 authors, 3 authors, etc…) for each group. We report our results in Figure 2.

We analysed this data for each group. In Figure 2, we present the comparative results for the best group and three average groups. We show this comparison, rather than the initially obvious comparison of the strongest groups against the weakest. This is because the weakest groups were quite distinctive in having very small amounts of overall activity. This effect dominated other trends and made their comparative results uninteresting. There were 2 weak groups, performing below any of those shown in the Figure 2 groups with average performance levels (3 of which are shown in the graph), one top group (also shown) and 2 groups with performance between the average groups and the top group.

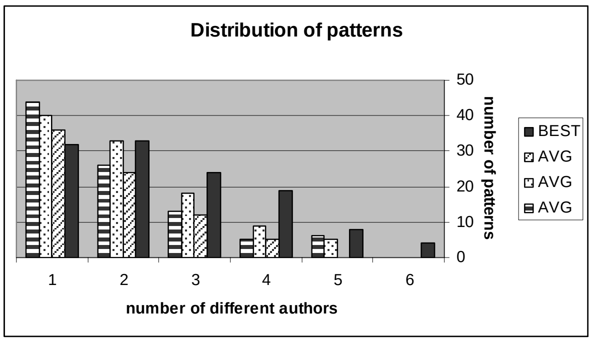


Figure 2 - Distribution of the sessions per number of users involved in a session

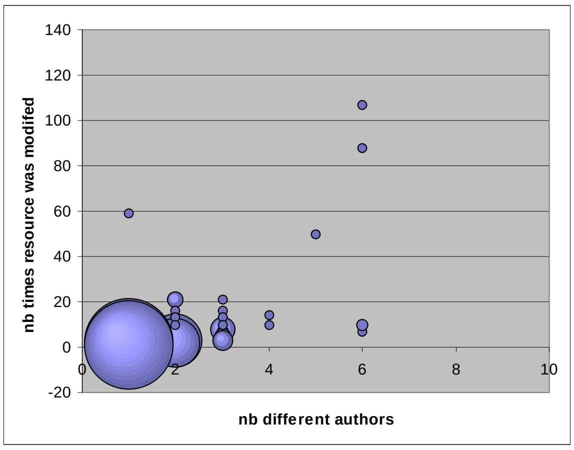


Figure 3. Frequency of patterns for the best group

Figure 2 shows that the most successful group, shown rightmost in each of the sets of histograms, had a relatively high proportion of sessions involving high interaction. Compared with the average groups, this group had more sessions involving three or more of their six members than we see for the other groups.

A simple analysis of overall interaction was an obvious starting point. However, the trend does not appear to be strong enough to be particularly useful. It does suggest that we could develop a useful measure of the amount of interaction and that this would be suggestive of more effective group work.

Because we had only ten groups, comparisons like this suffer the problem of a small sample size. Yet, for authentic classes, where there is a small number of groups, this would commonly be the case. So, we would like to find measures that could be usefully applied to typical classrooms.

Analysis of the patterns related to resource usage

In a well interacting group, we would expect that resources are shared among several users, with a great deal of interaction in each resource. This is particularly relevant for Wiki and tickets, perhaps less so for the source files in SVN (the reason being that if the project is well designed, java classes should belong to one team member and should have no reasons to be modified by other team members). After pre-processing the data by breaking down the sequence per resource (as in right of Figure 1), we ran the GSP algorithm and analysed by hand the patterns found, focusing on the number of users who worked on that resource and the length of time over which a resource was used.

Basic analysis: First we analyse the data with some basic statistics like the number of resources used by each group: from 155 for a weak group to 4488 for a strong group. It could be a good indicator of activity and work done, but does not show whether the members interact or not. A second basic analysis is the distribution of resources per type of event (ticket, Wiki, SVN). The result is that the resources are mostly SVN files (around 90% of resources). Therefore a good use of this tool (SVN) by the students is critical to deduce relevant results.

Interaction analysis: Two types of information are interesting to evaluate the level of interaction: the number of times a resource has been modified and the number of authors who have modified the resource. We plotted the frequency of patterns according to these 2 dimensions for each of the groups. Figure 3 shows this diagram for our top group. The bubbles represent the frequency level. For example we see here that the group used many resources by only 1 or 2 authors, and modified them just a few times (large bubbles in the bottom left corner). It also shows that some other resources were shared and modified extensively by several authors (bubbles in the top right corner). We first observed that the most frequent pattern for all groups involved only one author modifying one resource. However there was a small but clear difference between our top groups and the other groups. The less performing groups had such small number of resources shared by several members and modified many times that they did not show on the diagram whilst the number of resources manipulated by only one author and modified very little was much higher. In other words, the top group seemed to have interacted a lot more than the other groups. We did not include single events on SVN as we did not feel that shared activity on the SVN was really needed.

The other information we looked at was the lifespan of a resource sharing. As seen in Table 2, the best group on average modified a resource over 2.91 days, whereas this figure was 10.18 for the worst groups. This, together with the fact that better groups generate more events on a resource with more team members comforts us in the opinion that members of a good group are more responsive.

Table 2. Average number of days over which resources were modified

Best group	Average+ groups	Average Groups	Worst groups
2.91	1.48	3.31	10.18

We also found that the most common type of resource is a ticket or a file added by an author, and never modified by anyone (in which case the period of time is equal to 0). Whilst ticket actions not modified by anyone are a worrying indication of bad use of the TRAC system and of a lack of team monitoring (one of the Big 5 factors for successful teamwork (Salas, et al., 2005), the SVN single actions are normal if the software design is of good quality.

CONCLUSION AND FUTURE WORK

We have presented the initial data mining we performed on the data we collected from a semester long software development project course, where students worked in teams of 5 to 7 and were assessed on the demonstrated quality of the software product as well as the effectiveness of the software and group processes in achieving that product. Students were required to make use of the TRAC system. The mining was done on the data this system collected and its overall aim was to obtain insights into what distinguishes a group that is functioning well so that we can reflect this information back to the students and improve their learning experience. We focused so far on finding ways to preprocess the data and finding frequent pattern sequences in it. Our first results are not yet able to distinguish gross patterns that differentiate these classes but are certainly promising. There are a number of directions we will pursue in the immediate future.

Firstly, our results need to be analysed further. We need for this to have a more automated way to interpret the patterns found. There are numerous patterns so we would like to cluster the patterns so that we can compare them by groups more easily.

Secondly, our raw data needs some improvement. Currently we do not know for sure how events are related, which task they address. We separated sessions of activity using time intervals. We plan to reinforce the protocol of use of the TRAC system so that each event is attached to a ticket, which represent a task. We believe that extracting frequent sequences per ticket would reveal important information about how students work and communicate within a task. This will be addressed this year with our new cohort.

Thirdly, we want to review the notion of similarity between sequence patterns. All the algorithms we looked at find exact sequence patterns. However we would like to say for instance that 7 Wiki events by 4 authors is similar to 8 Wiki actions by 4 authors, or even 8 Wiki actions by 5 authors. In other words we would like to modify the algorithm so that it uses a more flexible notion of similarity. We also propose to explore a broader range of data mining approaches (Roddick and Spiliopoulou, 2002) and intend to investigate means to automatically discover contrasting sequences using the notion of contrast sets (Bay and Pazzani, 2001). Current contrast set mining algorithms consider only discrete and numeric data. We plan to extend the idea to sequences of events.

Fourthly, we want to connect the findings with theories of group work (Salas, et al., 2005). This might operate in both directions, with the theory being used to define patterns that are promising to identify and, in the opposite direction, recognising how the patterns match the theories. This is similar to the approach called bootstrapping novice data (BND) (Harrer, et al., 2005, McLaren, et al., 2004) where logs of learner activity are presented to an expert as a suitable visualisation, as a basis for building a coach that actually serves the needs of students, based on the approaches they actually take and the problems they actually experience. We would also, in the longer term, be able to make use of richer understanding of learner actions (Goodman, et al., 2005) as a basis for better interpretation of the student actions.

Overall, the current results give real promise that this will be possible. In particular, for this class of trace data, from TRAC, it seems that we will be able to mine meaningful patterns that will help teachers and learners improve and monitor the improvement of team operation. This is a major step forward from major work of this character, such as (Soller, 2004). We will then need to explore ways to generalise the approach to other group work domains, in particular, use of generally used collaborative tools such as chat, Wikis and communal whiteboards.

REFERENCES

Agrawal, R. and Srikant, R. (1995). Mining Sequential Patterns. Proceedings of International Conference on Data Engineering (ICDE95).

Barros, B. and Verdejo, M. F. (1999). An approach to analyse collaboration when shared structured workspaces are used for carrying out group learning processes. Proceedings of the Ninth International Conference on Artificial Intelligence in Education, Le Mans, France.

Bay, S. and Pazzani, M. (2001). Detecting group differences: mining contrast sets, *Data Mining and Knowledge Discovery*, vol. 5.

Beck, J. (2005). Proceedings of AAAI2005 workshop on Educational Data Mining.

Beck, J. (2004). Proceedings of ITS2004 workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes. Maceio, Brazil.

Choquet, C., Luengo, V., and Yacef, K. (2005). Proceedings of "Usage Analysis in Learning Systems" workshop, held in conjunction with AIED 2005, Amsterdam, The Netherlands, July 2005.

Goodman, B. A., Linton, F. N., Gaimari, R. D., Hitzeman, J. M., Ross, H. J., and Zarrella, G. (2005). Using Dialogue Features to Predict Trouble During Collaborative Learning, *User Modelling and User-Adapted Interaction*, vol. 15, pp. 85-134.

Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., and Hsu, M.-C. (2000). Freespan: Frequent pattern-projected sequential pattern mining. Proceedings of International Conf. on Knowledge Discovery and Data Mining (KDD'00), Boston, MA.

Harrer, A., McLaren, B. M., Walker, E., Bollen, L., and Sewall, J. (2005). Collaboration and Cognitive Tutoring: Integration, Empirical Results, and Future Directions. Proceedings of the 12th International Conference on Artificial Intelligence and Education (AIED-05), Amsterdam, the Netherlands.

Kay, J., Maisonneuve, N., Yacef, K., and Reimann, P. (2006). The Big Five and Visualisations of Team Work Activity. Proceedings of Intelligent Tutoring Systems, Taiwan.

McLaren, B. M., Koedinger, K. R., Schneider, M., Harrer, A., and Bollen, L. (2004). Bootstrapping Novice Data: Semi-Automated Tutor Authoring Using Student Log Files. Proceedings of Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes, held in conjunction with ITS-2004, Maceio, Brazil.

Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., and Hsu, M.-C. (2001). PrefixSpan: Mining Sequential Patterns Efficiently by PrefixProjected Pattern Growth. Proceedings of Int. Conf. Data Engineering (ICDE'01), Heidelberg, Germany.

Roddick, J. and Spiliopoulou, M. (2002). A survey of temporal knowledge discovery paradigms and methods, *IEEE Trans. on Knowledge and Data Engineering*, vol. 14, pp. 750-767.

Salas, E., Sims, D. E., and Burke, C. S. (2005). Is there a "Big Five" in teamwork?, *Small Group Research*, vol. 36, pp. 555-599.

Soller, A. (2004). Computational Modeling and Analysis of Knowledge Sharing in Collaborative Distance Learning, *User Modeling and User-Adapted Interaction*, vol. 14, pp. 351 - 381.

Srikant, R. and Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements. Proceedings of Fifth Int'l Conference on Extending Database Technology (EDBT), Avignon, France.