In *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future (Proc. AI&ED 2001),* J. D. Moore, C. L. Redfield, & W. Lewis Johnson, Eds., IOS Press, 2001, pp. 176-187.

Coaching Web-based Collaborative Learning based on Problem Solution Differences and Participation

María de los Angeles Constantino-González
Dept. of Computer Sciences
ITESM Campus Laguna
Paseo del Tecnológico #751,
Col. Ampliación la Rosita
Torreón, Coah. 27250
México
aconstan@campus.lag.itesm.mx

Daniel D. Suthers
Dept. of Information and Computer Sciences
University of Hawai`i at Manoa
1680 East West Road, POST 303A
Honolulu, HI 96822
USA
suthers@hawaii.edu

José I. Icaza
Center for Informatics Research
ITESM Campus Monterrey
Av. Garza Sada 2501
Monterrey N.L. 64840
México
jicaza@campus.ruv.itesm.mx

Abstract. This paper describes the design and evaluation of a coach that helps students collaborate during synchronous group problem solving. Unlike previous work generally emphasizing dialogue analysis, this work evaluates a new approach to supporting collaboration that identifies learning opportunities based on differences between problem solutions and on tracking levels of participation. The contribution made by these and other knowledge sources in the generation of collaboration advice was evaluated by comparing, for each event in the collaborative sessions, expert rankings of advice with the software coach's rankings, and by identifying the advice that would be lost if each respective knowledge source were removed. Results show that good quality advice can be obtained through these knowledge sources, although other knowledge sources may fill in gaps relative to the expert's performance. This work demonstrates how intelligent agents can produce reasonable collaboration advice using a few basic knowledge sources, and illustrates several methods of evaluating the knowledge and reasoning of complex knowledge-based systems.

1. Introduction

The main problems a collaboration coach has to solve are similar to the ones for individual coaching: when to intervene and what to say. Yet designing a coach that supports students' collaboration is a new challenge, since most prior work on coaching has focused on expert and student modeling (Katz, 1999). In contrast, a collaboration coach has to monitor not only one student's activities, but also the teammates' activities, and should encourage interactions that influence individual learning and the development of collaborative skills, such as giving and receiving help and feedback, and identifying and resolving conflicts or disagreements (Dillenbourg et al, 1996; Johnson et al, 1991; Webb & Palincsar, 1996).

Several systems have been designed to encourage participation and facilitate group discussion with intelligent support, such as C-CHENE (Baker & Lund, 1996), McManus & Aiken's (1995) Group Leader Tutor, IDLC's Expert System Coordinator (Okamoto *et al.*, 1995), and BetterBlether (Robertson *et al*, 1998). All of these systems use restricted menu-driven or sentence-opener interfaces in order to understand students' interaction, and give guidance based on an ideal model of dialogue. Dialogue-based support provides several advantages, such as potential applicability to any subject matter area, automated interpretation of students' interactions, and restriction of discussion moves and learning interactions to those believed to be productive for learning. However systems that require use of devices such as sentence openers present some disadvantages, such as restricting the type of communicative acts, slowing the communication process, and misinterpreting students' dialogue when students use the interface buttons incorrectly. It would be advantageous to increase the repertoire of ways to provide automated support.

Our work seeks to facilitate effective collaborative learning interactions, particularly with respect to the recognition and resolution of conflicts between students' problem solutions, with minimal reliance on restricted communication devices such as sentence openers. In this paper, we evaluate the feasibility of generating advice based primarily on comparing students' individual and group solutions and on tracking student participation (contributions to the group solution). The approach taken is close in spirit to the task analysis of Mühlenbrock & Hoppe (1999), which monitors and analyzes moves of multiple users within a shared workspace. Our approach differs in that we monitor individual work in private workspaces as well as the shared workspace, and identify conflicts based on theories of collaborative learning. Other previous studies have used automated coaches to give advice when a student's solution differs from an expert's solution (Paolucci, et al., 1996). In contrast, our work evaluates the possibility of giving advice without comparing student work with an expert solution. We excluded discourse models and expert problem solutions as a research strategy, in order to evaluate the value of the knowledge sources on which we focus. This strategy should not be interpreted as a denial of the importance of these other knowledge sources.

2. COLER

A computer coach was implemented and included within COLER (COllaborative Learning environment for Entity-Relationship modeling). COLER is a Web-based collaborative learning environment in which students can solve database-modeling problems while working synchronously in small groups at a distance. Entity-Relationship (ER) modeling is one of the most critical phases in the development of information systems in which designers and database users collaborate to produce a conceptual schema that meets the information needs of an organization (Batini, Ceri & Navathe, 1992). Different solutions are possible in this task due to different assumptions or misconceptions.

COLER's implementation is based on an open architecture for intelligent collaborative learning systems designed by Suthers & Jones (1997). COLER's interface and task domain is described in Constantino-González (2000) and Constantino-González & Suthers (2000). Briefly, the interface is structured as follows. A problem description window presents an entity-relationship-modeling problem. Students construct their individual solutions in a private workspace. When ready, they use a shared workspace to collaboratively construct ER diagrams while communicating largely

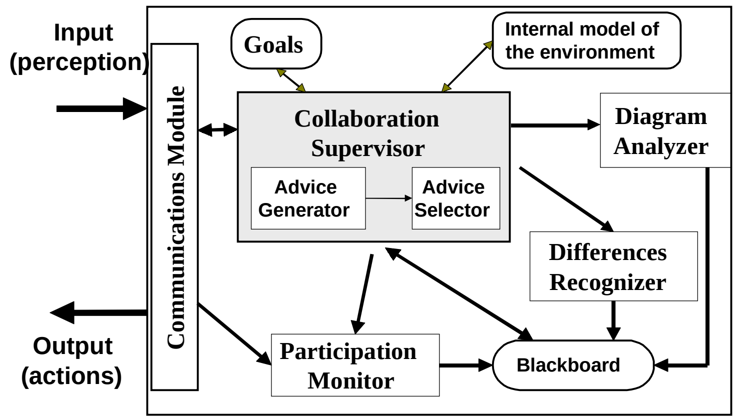

Figure 1. Coach Architecture

via a chat window. Only one student, the one who has the pencil, can update the shared workspace at a given time. A floor control panel provides two buttons to control this workspace: ask/take pencil and leave pencil. Additionally, this panel shows the name of the student who has the control of this area and the students waiting for a turn. An opinion panel shows teammates' opinions on a current issue. This area contains three buttons: OK: Total Agreement, NOT: Total or Partial Disagreement, and ?: Not sure, Uncertainty. A personal coach gives advice to "My Coached Student" (MCS) in the chat area. Although several suggestions may be computed at a certain time, only one is shown in the chat area. The others may be given on demand by pressing the suggestion button, which is disabled if the coach does not have any advice to offer.

3. COLER's Coach

The coach's goal is to promote group-learning interactions and maintain balanced participation. The design of the coach was based on socio-cognitive conflict and cognitive dissonance theories. According to the socio-cognitive conflict theory (Doise & Mugny, 1984) students learn from disagreements when they identify and resolve conflicts in their viewpoints, present alternatives, and request and give explanations. Cognitive dissonance theory (Festinger, 1957) states that the existence of disagreement among members of a group produces cognitive dissonance in the individual, who experiences pressure to reduce this dissonance, leading the individual to a process of social communication and revision of his/her position. The value of the disagreement depends less on the correctness of the opposing position than on the attention, thought processes and learning activities it induces.

The coach helps to prevent missed opportunities for collaborative learning (Baker & Bielaczyc, 1995) by monitoring students' participation and recognizing differences between students' individual and group solutions. When relevant opportunities for learning are found, the coach tries to guide students to practice collaborative skills, providing advice such as encouraging students to participate and to discuss their differences.

The coach design involves different modules that cooperate in the solution of the main problem: when and what advice to give (Figure 1). The *Diagram Analyzer* is a simple module that identifies participation opportunities based on the detection of problems in the quality of the ER group diagram. It uses syntactic and semantic information. The *Differences Recognizer* detects opportunities for students to collaborate by finding significant differences between individual and group ER diagrams. The Differences Recognizer can either find differences specifically related to the currently added object, or find all "extra work" that the student can contribute to the group. The *Participation Monitor* attends to the activity in the group diagram. If nobody has worked in the group diagram for a period of time, it reports this event. It also monitors whether each student is participating too much or too little. The Diagram Analyzer, Differences Recognizer and Participation Monitor communicate their results to the *Collaboration Supervisor* via a *Blackboard*. The Collaboration Supervisor maintains an internal model of the environment and other knowledge, including the current group and individual diagrams, MCS and team members' levels of participation, advice types, advice patterns and advice history, current received and given feedback, session phase and session start time. The Collaboration Supervisor operates in two phases: *Advice Generation* and *Advice Selection*. The Advice Generator computes the set of appropriate advice for a given situation, while the Advice Selector chooses the most appropriate advice from this set based on control strategies. The *Communications Module* is in charge of getting real time information from the environment and communicating a coach's advice when it is required.

COLER's main functions are to detect learning opportunities and to coach collaboration. COLER recognizes learning opportunities by (1) evaluating a number of syntactic dissimilarities between individual and group ER diagrams and by (2) tracking participation in the group workspace. COLER coaches collaboration by (3) generating a set of advice and (4) selecting the advice to give based on control strategies. The following four sections describe the different types of knowledge considered in these four activities and their use in the coach's reasoning.

4. Recognizing Differences between Solutions

COLER mainly recognizes learning opportunities by identifying a number of syntactic dissimilarities between individual and group ER diagrams, differences that are semantically interesting.

4.1 Knowledge: Significant Differences and Glossary

"Significant differences" in ER Modeling were defined based on three different information sources: (a) the first and third authors' previous experience in teaching data modeling; (b) analysis of 13 students' solutions for a specific ER problem provided by the domain expert; and (c) common errors in ER modeling reported in the literature (Batra & Antony, 1994; Shanks, 1996). Additionally, other learning opportunities are identified by recognizing some common errors in the group ER diagram, by evaluating the diagram according to quality features of ER diagrams. These problems can be detected based on their structural and categorical characteristics. The main idea to include them was to see how they could be used to encourage participation. A weight was assigned to each one of these differences depending on its impact. This weight is used to decide when to give advice.

Differences detection also relies on a problem-specific glossary of terms. The glossary includes the names of entities, attributes and instances mentioned in the database problem, as well as some names that might correspond to students' mistakes. Relationships' names are not included in the glossary because students might use a large number of different names for the same relationship, so they are not useful to match objects. Relationships instead are matched using dynamically generated internal names constructed from the associated entities (e.g. Employee-Project).

4.2 Reasoning: Diagram Matching

The Differences Recognizer undertakes subgraph matching between the private and group ER diagrams for the purpose of identifying differences. This matching is made tractable by using the glossary. Matching can either find differences specifically related to the currently added object (e.g. missing entity, extra attribute), or find all "extra work" that the student can contribute to the group.The Differences Recognizer module could be used for other purposes, e.g. to compare an expert diagram with individual and group solutions to detect misconceptions.

5. Monitoring Participation

The Participation Monitor attends to the activity in the group diagram. It attends time-triggered events, such as inactivity in the group area and MCS having the control of the group area for a long time. Group diagram events, such as object addition to the group diagram, are also attended so it can monitor whether each student is participating too much or too little.

5.1 Knowledge: Strategic Parameters

Much of COLER's strategic reasoning is controlled by parameters that can be adjusted as needed to suit different problem domains or instructor's preferences. In this section we introduce six parameters pertaining to student participation.

Three parameters control the desired *balance in participation* (activity in the workspace): MaximumStandardDeviation (MSD), MaximumConsecutiveContributions (MCC), and MinimumListenAdvice (MLA). COLER uses these parameters to monitor group's dynamics concerning the participation balance. *MSD* is used to determine the desired level of participation of each student compared with his/her teammates. If the value of this parameter is high, e.g. MSD > 1.4, the coach will encourage students to participate only when there is a large difference in their participation level. On the other hand, if this value is too small (MSD < 0.5), the coach will interrupt students almost after every action they do. *MCC* indicates the maximum number of consecutive contributions that MCS can do before COLER suggests that he/she let others participate. *MLA* indicates the minimum number of "listen" advices (e.g. LO: Listen to Others, LP: Let others Participate) that should be used to encourage MCS let others participate before taking the control of the group area from him/her.

One parameter was defined to encourage students to make adequate *progress on the task* of constructing the shared solution: TimeoutNoAction (TNA). TNA refers to the maximum period of inactivity in the group diagram that COLER considers before suggesting that MCS take action in the group workspace. Every time an action is performed in the group diagram (e.g. add, delete, change object), a timeout is set to verify that students are not just chatting or discussing for a long time, but also working on the construction of the group diagram. TNA should be defined according to the total time assigned to the group session. If this value is too small, the coach will constantly pressure students to work in the group area, with almost no time to discuss anything. If this value is too big, the student might not realize how the time is going and spend a lot of time chatting without any alert message from COLER.

Two parameters were defined to decide when COLER should encourage students to use the *opinion buttons*: TimeoutTeammateAction (TTA) and TimeoutMyStudentAction (TMA). A "Give Feedback" suggestion is considered when *TTA* time has passed since a teammate has performed an action in the shared area (add, delete, update) and MCS has not pressed any opinion button ("OK," "not OK," or "unsure"). An "Ask For Feedback" suggestion is considered when *TMA* time has passed and MCS has not received feedback from his/her teammates.

5.2 Reasoning

The Participation Monitor tracks the monitored student's number of specific contributions (SC_i), incrementing the value each time the student adds something. In this version of the coach, only the *add object* action to the group diagram is counted as a contribution. Future versions of the Participation Monitor could consider updating and deleting actions as contributions, by assigning them different weights. To evaluate participation, a standard deviation (SD) and a MEAN are computed based on students' contributions. If the SD exceeds the threshold of maximum standard deviation (MSD), explained previously, the monitor assumes there is a problem in participation and individual students are checked:

$$ProblemInParticipation(PIP) = \begin{cases} true & SD > MSD \\ false & otherwise \end{cases}$$

If the difference between a given student's contributions and the MEAN exceeds the MSD, it is assumed that the student is part of the problem, as follows:

$$ParticipationStatusStudent_i = \begin{cases} TooMuch: & PIP \wedge (SC_i - Mean > MSD) \\ NotEnough: & PIP \wedge (Mean \text{ - } SCi > MSD) \\ Acceptable: & otherwise \end{cases}$$

$$\begin{aligned} &where \\ &SC_i = Number\ of\ contributions\ of\ Student\ i\ in\ the\ group\ diagram \end{aligned}$$

Time triggered events are followed by a processing cycle. In each cycle, after waiting some time, if nobody has worked in the group diagram for the time specified in TimeoutNoAction (TNA), the monitor computes MCS's participation status and indicates the occurrence of this event by changing the variable HaveWorked in the blackboard. A similar process is followed to change the variable TooMuchTime, to indicate a student has had the pencil for a long time. These variables are checked by the Collaboration Supervisor.

6. Generating Advice

COLER uses event-driven inference to detect learning opportunities for discussion and participation and to generate advice appropriate for those opportunities. We first describe the relevant knowledge before detailing this event-driven reasoning.

6.1 Knowledge: Advice Categories and Types

COLER's advice is expressed as suggestions or questions that try to encourage students to discuss and participate. They are not imperative, so students should feel free to follow the advice or discard it when they believe it to be inappropriate. Advice types and categories were defined based on the collaborative learning literature and Wizard of Oz studies in which the human expert coached through the chat interface: see Constantino-González (2000).

Seven advice *categories* are defined in the present version of COLER. The first two categories, *Discussion* (in chat) and *Participation* (in the group workspace), are the main categories related to coaching collaboration. *Feedback* messages are related to student's pressing of COLER opinion buttons. The *ER Modeling* category includes suggestions related to some common errors in the domain. The *Self-Reflection* category consists of suggestions that individuals think about a problem or situation. Besides using advice from these categories, COLER can use messages for *welcoming* and saying *goodbye*.

Types of advice were defined and classified according to each of these categories. For each advice type, several advice *templates* were defined using different wording to provide linguistic variety. The templates can be contextualized by binding variables from the current situation, including the student's name, the object type (e.g. entity, relationship), the object's name, and the problem type (e.g. disconnected entity, no key defined). An example template (translated from the Spanish) follows:

$MyStudentName, $ObjectName $ObjectType proposed in the diagram is different from what you've got. If you do not agree with this, you should express and justify your viewpoint.

6.2 Knowledge: Collaborative Session Phases

The group session was divided into several phases, according to the progress in the group diagram (number of objects) and the elapsed time. During the collaborative session, students have a time limit to work and learn together and to generate a group solution. Eight phases were defined for the group session: Init, Waiting, Ready, Started, Middle, Verification, TimeFinishing and End. For each advice type, advice patterns with different semantics are defined depending on the phase, so it is possible to give suggestions according to the current phase. These phases and how it is possible to move from one phase to another are described in Constantino-González (2000). The evolution for the Middle, Verification, TimeFinishing and End sessions depends on five parameters that should be provided by the professor, two associated to time and three associated to the group diagram. Time-related parameters include the time in seconds the collaborative session will last (TimeLimit) and the time in seconds to indicate the reviewing period, when the session is close to finish (ReviewingPeriod). Group diagram parameters include the expected number of objects of the problem solution (TotalNodesApproximate) and the development percentages in the process of diagram construction (MiddlePercentage, VerificationPercentage).

6.3 Knowledge: Discussion Advice Intensity

Advice is not necessarily given every time a difference is found. COLER uses three parameters to define the extent to which the coach encourages students to discuss their differences: ThresholdImportantDifference (TID), ThresholdHighTotalWeight (THTW) and ThresholdMediumTotalWeight (TMTW). They are considered in the decision of when to interrupt students for discussion. The first one, TID, is used when a single difference exists. It indicates the value when a single difference is important. Its value ranges from 0 to 1. The last two, THTW and TMTW, are applied for multiple differences. *THTW* indicates the value of the sum of several differences that could be considered as high. *TMTW* indicates the value of the sum of several differences that could be considered as of medium importance. These values should be greater than zero. The larger theses values, the less often discussion is encouraged. The values defined for these parameters should be in concordance with the weights defined for each type of difference in the model analyzed. The importance of multiple differences (totalWeightImportance) can be computed using the following equation:

$$totalWeightImportance = \begin{cases} Significant: & TW \geq THTW \\ Medium: & TMTW < TW < THTW \\ Low: & TW \leq TMTW \end{cases}$$

$$\text{where}$$

$$TW = \sum_{i=1}^{n} Difference[i].weight$$

$$n = \text{Number of differences found at a given time}$$

6.4 Reasoning: Event-Driven Application of AND/OR Trees

Advice generation is event-driven. Three main types of events are attended: (a) Time-triggered events, such as inactivity in the group diagram (b) Group and individual diagram events, for example, the addition, change or removal of an object, and (c) Voting events, for instance, the receiving and giving of feedback. The Collaboration Supervisor then analyzes the situation and takes an action if it is required. The reasoning for each event uses an AND/OR situation tree defined for this event, as illustrated in Figure 2.

Time-triggered events are detected by a processing cycle. At the beginning, the Collaboration Supervisor initializes the coach's internal model by setting values for individual and group diagrams, category preferences, advice types, advice patterns, session phase and session start time. In each cycle, if the current phase corresponds to group learning time (e.g. Start, Ready, Middle, Verification), the Collaboration Supervisor checks students' activity in the group area by accessing the corresponding variable in the blackboard. If there has been inactivity in the group diagram for a long time, then the Collaboration Supervisor, using the AND/OR tree for this event, decides what advice to give. This cycle stops when the time available for the group session ends.

Diagram events are generated when MCS works on the individual or group diagram or a teammate works on the group diagram. When an object is added to the group diagram, the coach updates its state by setting a timer for feedback and updating the corresponding diagram, the current object being analyzed and the student executor of the action. Then, the AND/OR tree for this event is followed. When an object is deleted or updated, the current version of the coach does not do any reasoning. It only updates the corresponding diagram and the current node. The same occurs when an action is made in the individual diagram.

Voting events include those actions related with the opinion panel. When a feedback is given or received, the Feedback Applet asks the coach to check this action. The Collaboration Supervisor then analyzes it and takes an action if it is required.

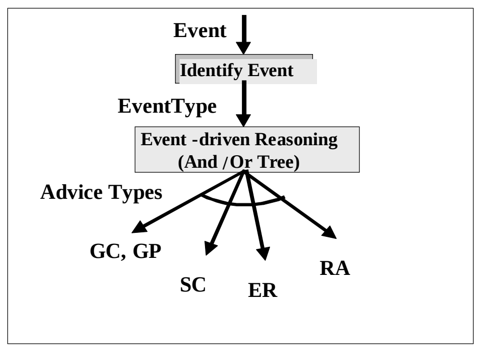

Figure 2. Advice Generation

Each event type has an *AND/OR tree* that generates advice for events of that type. Every branch of the tree represents a possible set of suggested advice. Several suggestions might be generated for any given event because several leaves may be reached at once via the "and" arc of the tree. Also, many of the leaves of each tree generate multiple advice, and trees for different events may be invoked at the same time.

Chat information is also considered to decide the kind of advice to give. This information is required in the process to decide whether discussion advice should be generated when differences against the current object exist. Both MCS's chat messages and his/her teammates' messages are considered. Chat information includes who has chatted and the chat messages' length. See Constantino-González & Suthers (2000) or Constantino-González (2000) for further details of the generation process.

7. Selecting Advice

Six *control strategies* were specified to control selection and timing of advice: Category Preferences, Collaborative Session Phases, Discussion Advice Intensity, Participation Balance, Time on Task and Waiting for Feedback. Many of these strategies are employed to filter advice during the generation process: these have already been described. The remaining control strategy utilizes Category Preferences.

7.1 Knowledge: Category Preferences

The coach has several sub-goals that correspond to the advice categories, such as promoting discussion of differences, balance in participation, use of feedback, self-reflection and ER modeling achievement. Default preferences are assigned to these advice categories when the session starts, so the coach can process them in accordance to their importance. A *preference* is a predicate that compares two proposed advices and chooses one as being preferable to the other. Preferences are used to select from among the generated advice. The relative priority of preferences in use can change during the group session according to the group's performance.

7.2 Reasoning

The Selection process involves two steps. First, select an advice pattern from the advice types generated by each one of the leaves. Second, select an advice from the advice pattern set generated.

The first step is implemented as follows. First, a list containing the advice templates corresponding to the advice types of an AND/OR tree's leaf and the current group session phase is generated. If this leaf includes several types of advices, the one that have been just previously used is eliminated, implementing a preference to avoid giving the same kind of advice twice. If the leaf contains only one advice type and this is the same than the last advice type given, this advice type is selected only if a no-advice limit has been reached. This limit is defined by the parameter "MaximumNoAdvice," which is measured based on the number of times that an advice could have been given but was not because its type was the same than the type of the last advice given. After this verification is done, an advice template is randomly chosen from the list for each leaf node.

The second step starts with a revision of the category preferences, which may change depending on the group's performance. In the current version, only discussion and workspace participation preferences are interchanged. If the group seems to need more participation advice, this category of advice is promoted. Otherwise, discussion is encouraged. Finally, a preference-based sort algorithm is run if needed to choose between multiple advice instances. This sort splits the advice into more preferred and less preferred based on the first preference, and does this recursively on each partition with the remaining preferences, concatenating the results to yield a sorted list. Examples of preferences are New Advice (don't repeat advice type during the session), Many Instances (prefer advice of a type that applies more than once), and Category Preferences (e.g. Discussion, Participation, Feedback). From the sorted list, the coach gives the more preferred advice. The others are stored in a list to be given on demand.

8. Evaluation

The evaluation reported here assesses the quality of advice generation and selection algorithms, and the contributions of knowledge sources in the generation of reasonable advice. Future publications will report detailed evaluation of the relationship between group functioning and COLER's advice.

8.1 Summary of Method and Procedure

This laboratory evaluation of COLER involved participants who had taken or were taking a database course. Our domain expert, a computer science professor, was also present in two sessions. A pilot session was run to test COLER's usability and functionality. Then, five sessions were conducted to generate data and scenarios for the different types of evaluations. In each of these sessions, three students were presented with a simple database design problem. They first solved the problem individually, and then convened to construct a group solution. Students and coach activities were recorded in a log file. The pilot study and the two sessions in which the Expert was present were used for preliminary evaluation, detecting some problems in COLER's user interface and coach algorithms. The last three sessions, in which the expert was not present, were used to evaluate COLER's algorithms and the quality of its advice, as described below.

8.2 Documents for Expert Evaluation

For each student of each of the last three sessions, two documents were generated for the Expert's Evaluation: the Environment Document and the Advice Document. These documents describe the chronological sequence of events of the collaborative session in reference to a specific student, and the context of each event (current state of the environment). The *Environment document* provided the expert with the same information that the computer coach had during the collaborative session (e.g. group diagram, event type, voting response, contributors of chat messages). After each event, a space was left for the Expert to indicate the advice he would give, if any. The *Advice document* was used to evaluate COLER's algorithms and advice acceptability. For each event, the expert was asked to rank the suitability of all of COLER's advice types assigning an integer number starting at 1, and to indicate a cutoff of which advice was "Worth saying," "So-So" and "Not Worth Saying," all without knowing which COLER actually considered. It was possible to assign the same rank to different advice types. Then a new page shows and solicits comments on the advice types the coach actually generated, and on the advice type selected. Subsequently, each student's individual diagram and the chat transcript of the collaborative session were printed and given to the human Expert to evaluate whether his advice would change if he could see more than the coach did. The advice generation algorithm was evaluated by comparing COLER's generated advice to those generated by the expert, as well as through the expert's ranking of all advice available to COLER. The advice selection algorithm was evaluated by comparing COLER's ranking to the expert's ranking.

8.3 Results

A number of advice instances were generated from each category: 34 were participation, 23 Discussion, 6 Self-Reflection/Discrepancy and 9 Feedback advices. Participation and discussion advices are the ones that were given the most. Although Participation advices were the more used, Discussion advices were also important since the participation advice "continue task" is usually given to all group members at a similar time while discussion advices are usually given in different situations.

The *overall knowledge available to COLER* was evaluated by comparing expert and COLER advice for each situation, with 67% of the advice given by the expert not given by the coach. Thus, as expected the expert has a greater repertoire of advice, although COLER's limited knowledge sources produced the same advice as the expert in 33% of the situations. Of the missing advice, 69% would require new advice types and new branches in the AND/OR situation trees, 21% involved situations already considered in the AND/OR tree but requiring that new advice types be attached to them, and 10% involved advice that COLER could give with minor adjustments to parameters.

A new category of advice, “Social Interaction,” is needed to establish a closer relationship between COLER coach and the student. This category could include different advice types such as thanking the student for listening to advice, and otherwise commenting on student actions. Some existing advice types need to be extended to mention a specific context, such as suggesting that students reflect on a specific difference or inviting someone in particular to participate. The findings also suggested situations in which the“Self-Reflection” advice type could be given.

Advice Generation was evaluated by using the expert's classification of the advice available to COLER into “Worth saying,” “So-So” and “Not Worth Saying.” Results showed that 73% of the advice generated by COLER was worth saying, 7% was “so-so” and 20% was not worth saying. Some reasons for “Not Worth Saying” advice are change in conditions (making the advice obsolete) and failure to match entities due to spelling errors and unidentified synonyms. The obsolesce problem could be solved by defining and reviewing the conditions for each specific advice type before giving the advice. Spelling errors could be managed by devaluing the importance of differences in relationship’s names for generating “Check discrepancy” advice, or by using a distance metric between the spellings. Also, some advices, such as "analyze alternatives" and ER related advice, were given in situations different from the Expert, so they should be reviewed with the Expert in order to modify the corresponding AND/OR situation tree. Results also indicated the need to define a new parameter that specifies the time required to wait when a NOT or NOT_SURE feedback has been received and MCS has not given any response before suggesting that MCS give an explanation.

The *Advice Selection* module was evaluated by analyzing events in which (1) several candidate advices with different rankings exist, and (2) some advice was suggested by the Expert. To evaluate the selection algorithm independently of the generation algorithm, a “new” Expert ranking was computed based on the actual Expert ranking but considering only the advice generated in a given situation, instead of all the available advice . The disparity between COLER's and the Expert’s ranking of this generated subset was measured using the Euclidian distance between the individual ranks:

$$d_{CE} = \sqrt{\sum_{i=1}^{n}\left(x_{Ci}-x_{Ei}\right)^2}$$

The COLER-Expert Euclidian distance d_{CE} is the square root of the sum of squared differences across a set of advice types, where n = the number of advice types generated by COLER, x_{Ci} is the value of COLER’s rank for the i^{th} advice type and x_{Ei} is the value of the Expert’s rank (of the generated subset) for the same advice type. The COLER-Expert Euclidian distance is not a measure of COLER’s advice quality, but a measure of the precision of the ranking assigned by the Selection Algorithm to COLER’s advice types. Selection among several advice types was needed only a few times in this study. There were 2.33 average selections per session, each selecting between an average of 4.78 generated advice items. The Euclidian distance obtained was 0.9, i.e., less than one disagreement in ranking per selection event. This seems adequate although leaving room for minor improvements.

The *contribution of each knowledge source* in the generation of reasonable collaboration advice was evaluated by "ablating" it analytically, i.e., identifying the advice that relied on the knowledge source and hence would be lost if the knowledge source were removed. We focused on the advice that the expert ranked as "reasonable." This analysis used the Environment and Advice documents to identify the situations in which COLER gave advice and the rank the

Expert assigned to this advice, and the AND/OR trees to identify the type of knowledge used in each situation.

The contribution of knowledge sources to generation of advice judged by the expert to be "reasonable" was as follows: Voting Tracking and timeout (49%), Participation Balance (48%), Significant Differences and Glossary (41%), Time on Task (40%), Chat Tracking (37%), Discussion Intensity Parameters (29%), Category and Sort Preferences (22%), Pencil Tracking (14%) and Common Problems in ER diagram (2%). Some knowledge sources were used to generate different categories of advice (hence the percentages reported above sum to greater than 100) while others were more marginal and only were used in a specific advice category. For instance, knowledge of significant differences and glossary is used to produce Discussion, Participation and Self-Reflection advice, while knowledge of participation balance, time on task, pencil tracking and problems in ER diagrams are used to generate only Participation advice. Similarly, the generation of reasonable advice (e.g. Discussion and Participation) in this study required the conjunction of several types of knowledge (e.g. Significant Differences and Glossary, Participation Balance) and confirmed the hypothesis that knowledge on problem solving activity could be used to generate reasonable collaboration advice. The knowledge about problems in quality of ER diagrams was used very little in this study since the coach's primary goals is promote discussion and participation instead of teaching ER modeling.

9. Conclusions and Future Work

This work is part of a research agenda that seeks to characterize the knowledge needed to facilitate collaborative learning processes. The present focus has been on determining how much leverage can be obtained by a basic ability to detect semantically interesting differences between representations of two problem solutions, together with simple tracking of individual's quantity of participation (e.g. contributions in the shared area) and feedback given (e.g. opinion buttons). The study showed that reasonable collaboration advice can be generated without the need for expert solutions or discourse understanding, although the addition of these knowledge sources would improve the quality and range of advice generated and selected by the system (at the cost of additional knowledge engineering and system complexity). Specifically, 73% of the advice given was considered to be reasonable by the expert, and 33% of COLER's advice was the same as that which the expert would give. (It would be interesting to compare this to the level of agreement between two human coaches.) These results indicate that COLER is a viable advisor, albeit different in style from our expert. Response time should also be considered: our expert pointed out that he came up with his advice after careful and time-consuming study of the Environment and Advice documents while COLER generated advice based on the same information in real time. The approach should generalize to all domains in which students construct formal representations of problem solutions that can be compared for significant differences.

A secondary contribution of this work is to illustrate several useful evaluation methodologies, including (1) evaluation of overall knowledge by comparing freely chosen expert advice to advice generated by the system; (2) evaluation of advice generation and selection by comparing generated and selected advice to an expert ranking of all advice available to the system; (3) evaluation of advice selection by Euclidean distance between expert and coach rankings; and (4) evaluation of the contribution of each knowledge source by analytical ablation. Other evaluations underway include student opinions, whether the advice was time appropriately for the group's collaborative activity, and whether students take the advice given. Overall, a mixture of empirical and analytic methodologies is advocated to fully understand the design of complex systems.

10. Acknowledgments

We thank our colleagues at the Learning Research and Development Center of the University of Pittsburgh for their support, in particular Alan Lesgold for hosting the first author as visiting scholar and employing the second as his research associate, and Sandy Katz, Arlene Weiner and Eva Toth for discussions about collaborative learning. We also thank Jose G. Escamilla of ITESM for his advisorship of the first author, and Moraima Campbell of ITESM for her support to conduct the formative studies, and visiting students Vincent Trifot and Justin Peltier for implementing part of the personal coach. The Center for Artificial Intelligence of ITESM provided research facilities. During this research, Constantino-Gonzalez was funded by ITESM Campus Laguna and by CONACYT, and Daniel D. Suthers was funded by the Presidential Technology Initiative (while at LRDC) and by NSF's Learning and Intelligent Systems (while at the University of Hawai'i).

11. References

Baker, M. J. and Bielaczyc, K. (1995). Missed Opportunities for Learning in Collaborative Problem-solving Interactions. *AI&ED'95*.

Baker, M. J. and Lund, K. (1996). Flexibly Structuring the Interaction in a CSCL environment. In *Euro-AIED'96 Conference Proceedings (European Conference on Artificial Intelligence and Education)*, Lisbonne, October.

Batini, C., Ceri, S. and Navathe, S. B. (1992). Conceptual Database Design: An Entity-Relationship Approach. *Benjamin/Cummings, Redwood City, California*.

Batra, D. and Antony, S. R. (1994). Novice Errors in Conceptual Database Design. *European Journal of Information Systems*, Vol. 3, No. 1, pp. 57-69.

Constantino-González, M. A. (2000). A Computer Coach to Support Collaboration in a Web-based Synchronous Collaborative Learning Environment. Unpublished dissertation, *ITESM (Instituto Tecnológico y de Estudios Superiores de Monterrey)*, México.

Constantino-González, M. A. and Suthers, D. D. (2000). A coached collaborative learning environment for entity-relationship modeling. In *Intelligent Tutoring Systems, Proceedings of the 5th International Conference (ITS 2000),* G. Gauthier, C. Frasson, & K. VanLehn (Eds.), pp. 325-333. Berlin: Springer-Verlag.

Dillenbourg, P., Baker, M., Blaye, A. and O'Malley, C. (1996) The Evolution of Research on Collaborative Learning. In Spada and Reimman (Eds.) *Learning in Humans and Machines*.

Doise, W. & Mugny, G. (1984). The Social Development of the Intellect. *International Series in Experimental Social Psychology*, 10, Pergamon Press.

Festinger, L. (1957). A theory of cognitive dissonance. *Stanford University Press*.

Johnson, D. W., Johnson, R. T. and Smith, K. A. (1991). Increasing College Faculty Instructional Productivity, *ASHE-ERIC Higher Education Report No. 4. School of Education and Human Development, George Washington University*.

Katz, S. (1999). The Cognitive Skill of Coaching Collaboration. *Submitted to CSCL'99*, Stanford, CA.

McManus, M. M. and Aiken, R. M. (1995). Monitoring Computer Based Collaborative Problem Solving, *Journal of Artificial Intelligence in Education* , 6(4) , 308-336.

Mühlenbrock, M., & Hoppe, U. (1999). Computer Supported Interaction Analysis of Group Problem Solving. In *Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference*, C. Hoadley & J. Roschelle (Eds.) Dec. 12-15, Stanford University, Palo Alto, California. Mahwah, NJ: Lawrence Erlbaum Associates.

Okamoto, T., Inaba, A. & Hasaba, Y. (1995). The Intelligent Learning Support System on the Distributed Cooperative Environment, *Proceedings of Artificial Intelligence in Education*, August, Washington, D.C., p. 588.

Robertson, J., Good, J. & Pain, H. (1998). BetterBlether: A Computer Based Educational Communication Tool, *IJAIED.*

Paolucci, M., Suthers, D., and Weiner, A. (1996). Automated Advice-Giving Strategies for Scientific Inquiry. *Third International Conference on Intelligent Tutoring Systems (ITS'96)*, June 12-14, 1996, Montreal.

Shanks, G. (1996). Conceptual Data Modelling: An Empirical Study of Expert and Novice Data Modellers. *ACIS '96.*

Suthers, D. D. and Jones, D. (1997). An Architecture for Intelligent Collaborative Educational Systems, *AI&ED97*, Japan.

Webb, N. and Palincsar, A. S. (1996). Group processes in the classroom. *Handbook of Educational Psychology*. D. Berlmer & R. Calfee Eds. Simon & Shuster Macmillan NY, 1996.