Computers & Education 54 (2010) 416–426

Contents lists available at ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Assisting students with argumentation plans when solving problems in CSCL

Ariel Monteserin *, Silvia Schiaffino, Analía Amandi

ISISTAN Research Institute, Fac. Cs. Exactas, Univ. Nac. Del Centro de la Pcia. De Bs. As., Campus Universitario, Paraje Arroyo Seco, Tandil, Bs. As., Argentina
CONICET, Consejo Nacional de Investigaciones Científicas y Técnicas, Argentina

ARTICLE INFO

Article history:
Received 9 June 2009
Received in revised form 21 August 2009
Accepted 24 August 2009

Keywords:
Argumentation-based negotiation
Computer-supported collaborative learning
Planning

ABSTRACT

In CSCL systems, students who are solving problems in group have to negotiate with each other by exchanging proposals and arguments in order to resolve the conflicts and generate a shared solution. In this context, argument construction assistance is necessary to facilitate reaching to a consensus. This assistance is usually provided with isolated arguments by demand, but this does not offer students a real and integral view of the conflicts. In this work, we study the utilisation of argumentation plans to assist a student during the argumentation. The actions of an argumentation plan represent the arguments that a student might use during the argumentation process. Moreover, these plans can be integrated with the tasks needed to reach a shared solution. These plans give the student an integral and intuitive view of the problem resolution and the conflict that must be resolved. We evaluated our proposal with students of an Artificial Intelligence course. This evaluation was carried out by comparing three different assistance scenarios in which students had to solve exercises: no assistance, assistance with isolated arguments, and assistance with argumentation plans. The results obtained show that reaching consensus was easier for the students when the assistance was provided using argumentations plans.

1. Introduction

In a computer-supported collaborative learning (CSCL) context, when students solve problems in groups, they need to interact with one another. In such scenarios, negotiation is a fundamental tool to reach agreements among students with conflictive goals, mainly, because learning is particularly effective when students encounter conflicts and resolve through negotiation generating a shared solution (Baker, 1999; Petraglia, 1997; Piaget, 1977; Veerman, Andriessen, & Kanselaar, 2000). Therefore, students must negotiate with each other, exchanging proposals and arguments to persuade their group-mates. Particularly, in CSCL systems, argument construction is more difficult than in face-to-face argumentations, since there are factors that make argument construction difficult (Veerman et al., 2000). Hence, it is necessary to assist students in such a task. Most of this assistance is provided with isolated arguments. That is, when the student is solving a problem in a CSCL system, arguments construction assistance is provided by demand, when a conflict arises. However, we claim that the student can be assisted in an integral way.

When we negotiate, the arguments uttered are not the result of an isolated analysis, but of an integral view of the problem that we want to agree about. Given a situation where it is needed to negotiate, the ability to plan the course of action that it will be executed to solve the conflict allows the negotiator to anticipate the problems that he/she could find during the interaction, and also, to analyse anticipated solutions to the conflict in order to avoid or minimise its problematic effects. This anticipation is also useful to evaluate several plans in advance in order to choose the most profitable one. We claim that this fact can be taken into account to provide assistance to students and make reaching consensus easy.

In this work, we present a study that shows the advantage of assisting the students by suggesting *argumentation plans*. We define an argumentation plan as a partial order sequence of arguments that allows a student to reach an expected agreement when it is uttered in a specified conflictive situation. These plans could be integrated into a general one, which determines the task needed to solve the problem. In contrast with the assistance provided using isolated arguments, these plans give the students an integral view of the problem resolution, and allow them to detect relations among the different conflicts, decide which task could be agreed, and in which conflict and what they should concede.

* Corresponding author. Tel.: +54 2293 439682; fax: +54 2293 439681.
E-mail addresses: amontese@exa.unicen.edu.ar (A. Monteserin), sschia@exa.unicen.edu.ar (S. Schiaffino), amandi@exa.unicen.edu.ar (A. Amandi).

0360-1315/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2009.08.025

To build the argumentation plans, we take into account the concepts and techniques used in the field of negotiation among agents. Agents are computational entities with autonomous, social, reactive and pro-active behaviour. These agents can execute actions autonomously or assist a user (such as a student in a CSCL environment) in his/her task. In this field, negotiation has been widely studied. The essence of a negotiation process is the exchange of proposals. Agents make proposals and respond to proposals made to them in order to converge to a mutually acceptable agreement that finds a solution to a conflict. However, not all approaches are restricted to that exchange. Several approaches to automated negotiation have been developed. One of them is the argumentation-based approach (Kraus, Sycara, & Evenchik, 1998; Rahwan et al., 2003; Ramchurn, Jennings, & Sierra, 2003; Sierra, Jennings, Noriega, & Parsons, 1998). In argumentation-based approaches, negotiators are allowed to exchange some additional information as arguments, besides the information uttered on the proposal (Rahwan et al., 2003). Thus, in the context of negotiation, an argument is seen like a piece of information that supports a proposal and allows a negotiator to justify its position in the negotiation, or to influence the position of the counterpart (Jennings, Parsons, Sierra, & Noriega, 1998). In other words, we can say that the argumentation process that the negotiator carries out during the negotiation, starts in an initial state of conflict and finishes in a final state of agreement, and we can see the arguments as actions or movements to go from this initial state to the final one.

In view of this idea, we propose to model the argumentation process as a planning problem. This modelling allows us to build argumentation plans using a planning algorithm. Planning algorithms provide a course of action that, when it is executed on the specified initial state, it allows us to achieve an expected final state (Fikes & Nilsson, 1971). In particular, we use a planning algorithm based on preferences in which student preferences impact on the action selection process and, as a result, on the argument selection mechanism, which decides what argument must be uttered in a given situation when there are several alternatives to choose. Finally, we can use an argumentative user model (Monteserin & Amandi, 2008) to personalise the argumentation plan construction. This model is composed of the action that a user (in this study, the student) uses to generate arguments and it is built from the arguments that he/she uttered in previous argumentations.

In summary, a tutor agent assisting a student in a CSCL scenario observes him/her, builds a tentative argumentation plan, and guides him/her during the conflict resolution. We claim that this kind of assistance is better that the assistance provided with isolated arguments. Although we choose a tutor agent to implement the assistance, it is worth noticing that the same assistance can be provided in different ways. Also notice that the focus of this study is the assistance that the students receive, not the development of a software assistant or intelligent tutor.

We have evaluated our proposal within an e-learning platform named SAVER[1] where tutor agents assist students that solve exercises using a collaborative application available in this platform. Students of an Artificial Intelligence course participated in the experiments. This evaluation was carried out by comparing three different assistance scenarios in which students had to solve exercises: no assistance, assistance with isolated arguments, and assistance with argumentation plans. The results obtained are promising. We found that when the students were assisted with argumentation plans they perceived that reaching consensus was easier than in the other two scenarios, whereas they perceived that the problem complexity increased.

The paper is organized as follows. Section 2 describes some works related to argumentation in CSCL. Section 3 shows how the argumentation process can be modelled as planning problem. Section 4 shows how the argumentation problem is represented in the planner, as an initial state, a final state and the actions. Section 5 shows how the planning algorithm works to generate argumentation plans for students. Then, Section 6 shows a case study to illustrate our proposal. Section 7 presents the methodology used and the experimental results obtained when evaluating this work. Finally, Section 8 presents a discussion and our conclusions.

2. Argumentation in CSCL

The negotiated construction of knowledge through argumentation is one of the main principles of constructivist learning theory. As we have introduced, learning is particularly effective when students encounter conflicts and resolve them through negotiation to generate a shared solution. In the last years, the increase of accessibility to the Internet has fomented the rise of e-learning. In this context, computer-mediated communication (CMC) has shown to increase learning-to-learning interaction and facilitate critical thinking in online group discussions. As a consequence, CMC has been used to support collaborative argumentation (CA), originating, computer-supported collaborative argumentation (CSCA). CSCA allows students to practice argumentation and debate to resolve conflicts and communicate online with other students using text-based communication tools (Baker, 1999).

Several systems have been developed for supporting collaborative argumentation: CATO, (Aleven, 1997), Beldevere (Suthers, 2003), ArgueTrack (Bouwer, 1999), LARGO (Pinkwart, Aleven, Ashley, & Lynch, 2006), Digalo (Kochan, 2006) and the works of Yuan, Moore, and Grierson (2008), Loll, Pinkwart, Scheuer, and McLaren (2009), among others. There are some differences among these CSCL systems and face-to-face argumentation (Wen & Duh, 2000). Specially, the systems can overcome temporal and geographical barriers, and they can eliminate social differences, such as age, sex, and hierarchies. For that reason, it is important that the students receive some kind of assistance during the argumentation. Some of the works on CSCA focus on the assistance that can be provided to students. There are several kinds of assistance in CSCA: (a) to facilitate argument construction, (b) to assist argument understanding and (c) to promote the participation of the students in the discussions. Our work concentrates on the first type of assistance.

As regards agents that assist users in CSCA, Huang, Chen, and Chen (2009) propose an agent that makes suggestions to students when it detects devious argument or abnormal behaviour that is unfitted to the learners' learning style. Wen and Duh (2000) propose an Automatic Facilitator that facilitates clear presentation of argumentation and better interface environment for students to articulate his/her arguments. An intelligent agent that provides students with suitable online argumentation strategies and rhetorical methods in a computer-supported collaborative argumentation environment is described by Yu and Chee (1999). This agent suggests isolated arguments patterns when a student is taking part in a discussion. Also, in Verheij (2003), the author presents an argument assistant tool to guide the user's production of arguments.

[1] http://www.e-unicen.edu.ar.

All these works have in common that the assistance is concentrated on an individual argumentative situation; they do not take into account that during the same problem solving there can be several argumentative situations linked with each other. In contrast, our argumentations plans provide a global view.

3. Modelling an argumentation process as a planning problem

In order to generate argumentation plans, we have to model the argumentation process as a planning problem. We define an argumentation plan as a partial order sequence of arguments that allows a student to reach an expected agreement when it is uttered in a specified conflictive situation. An argumentation plan will determine how a student should perform the argumentation process during a given negotiation. That is, for instance, to establish the set of arguments and the order in which they should be presented to the counterparts in order to achieve an agreement.

In this section we will describe the main characteristics of an argumentation process, present some mechanisms of a planning algorithm (those relevant to our proposal), and define a planning problem conceptually. Then, we will match every component of the argumentation process with its corresponding one in a planning problem. With the purpose of facilitating the understanding, we can see in Fig. 1 a graphical representation of this idea.

When an agent, which is assisting a student, detects a conflict, it can access to information about this conflict and the context of the conflict before the negotiation begins. By definition, the conflict that will generate the negotiation is rooted in the conflictive interest that the involved students have. Thus, the information that an agent assisting a student can access before starting the negotiation process includes:

- *Self-information*: agent's mental state, such as the student's beliefs, preferences and goals. This information conditions the agreement that the agent seeks.
- *Information about its opponents*: this is information about other students' beliefs and goals that the agent has gathered in previous interactions. In realistic situations students have only incomplete information about their classmates.
- *Information about the conflict context*: relevant knowledge about conflict and its resolution, for example the space of potential agreements (Jennings et al., 2001); and historic information about past negotiations.

Henceforth, we will call negotiation information to the information that an agent can use to assist a student.

As we introduced earlier, in an argumentation-based negotiation approach, negotiators can exchange arguments in order to justify their proposals, to persuade their opponents, and to reach an expected agreement. An argumentative agent, in addition to evaluating and generating proposals, must be able to (a) evaluate incoming arguments and update its mental state as a result; (b) generate candidate outgoing arguments; and (c) select an argument from the set of candidate arguments (Ashri, Rahwan, & Luck, 2003). Thus, we can say that the argumentation process is composed of the evaluation of outgoing arguments and the generation and selection of incoming arguments. To achieve this, the agent starts the argumentation process, and takes every decision concerned with this process on the basis of the negotiation information. In other words, this information is part of the input of the evaluation, generation and selection of arguments.

In this work, we focus on incoming arguments, that is, the generation and selection of arguments:

- Argument generation is related to the generation of candidate arguments to present to a counterpart. For this end, rules for creation of arguments are defined (e.g. Kraus et al., 1998; Ramchurn, Jennings, et al., 2003). Such rules specify conditions for argument generation. So, if the condition is satisfied in the negotiation context, the argument may be generated and it becomes a candidate argument.
- Argument selection is concerned with selecting the argument that should be uttered to a counterpart from the set of candidate arguments generated by the argument generation process. Once candidate arguments have been generated, then the argument selection mechanism must apply some policy, in accordance with the student's preferences, to select the best argument. Policies vary from one work to another, they can order all arguments by their severity and select first the weakest (Kraus et al., 1998); take into account the trust or reputation of the counterpart (Ramchurn, Jennings, et al., 2003); or choose the shortest argument in size in order to reduce the target to counter-argue (Schroeder, 1999); among others.

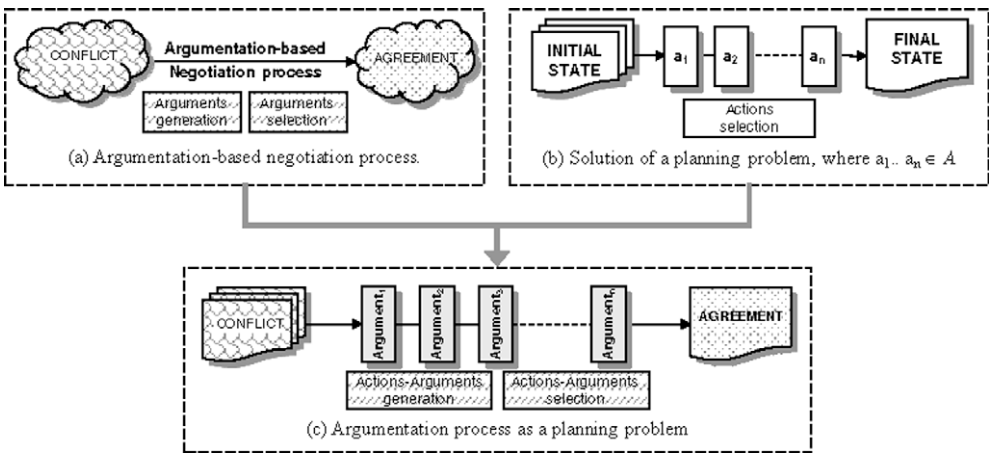

Fig. 1. Graphical representation of an argumentation process as a planning problem.

On the other hand, a planning algorithm is used to find a plan of action. In our approach, a plan is a partial order sequence of actions that when it is executed in any world satisfying the initial state description, it will achieve a desired final state (Weld, 1994). Conceptually, a planning problem is defined as a tree-tuple $\langle i, f, A \rangle$, where:

- *i* (initial state) is a complete description of the world, in which the plan will be executed.
- *f* (final state) describes the agent's (student's) goals.
- *A* (actions) is the set of available actions to build a plan. For each action its precondition and effects are defined. For an action to be added to the plan, its preconditions must be satisfied, and it is assumed that, with its later execution, the world will be modified by the effects.

These pieces of information are the input of a planning algorithm. Moreover, there is one mechanism in a planning algorithm that is important for our proposal: the action selection mechanism. This mechanism chooses which action will be added to the plan in a particular iteration of the algorithm. For instance, Weld (1994) defines this mechanism by the non-deterministic choose function; nevertheless, it might be redefined in accordance with every specific planning problem.

Now, we will explain how the argumentation process may be modelled as a planning problem. In this direction, we outline how each input of the planning problem must be defined in order to generate argumentation plans:

- *Initial state*: the conflict is the beginning point of the argumentation, and it is described in the negotiation information. The initial state describes the world where the conflict takes place.
- *Final state*: the student's goal in a conflictive situation is to reach an expected agreement about the solution of an exercise; therefore, this is the goal of the argumentation process too.For instance, if the student *st1* needs to execute an action *alpha* to fulfil a goal *g1*, but that action execution keeps its opponent *st2* from fulfilling a goal *g2*, the initial state *i* should include information as *isgoal(st1, g1), isgoal(st2, g2), believe(st1, imply(alpha, g1))* and *believe(st2, imply(alpha, not(g2)))*; and the final state *f* should represent the execution of the action *alpha*.
- *Actions*: as we described above, a rule for argument generation defines the condition to create an argument, so we can think that the argument is the effect of the rule. For that reason, we can define actions to generate arguments with the same rule patterns, where the action preconditions are the condition to generate an argument and the action effect represents the argument. Furthermore, we define actions that outline the possible opponent's responses in order to represent the effects causes by argument acceptances.

In addition, we must emulate in the planner both argument generation and selection mechanisms. We describe below how both mechanisms are present in the planner:

- *Argument generation*: since the rules to generate arguments can be seen as actions in a plan, when the planner establishes what actions might be added to the plan, checking its preconditions and its effects in view of the current state and the expected final state of the world, it will be implicitly generating the candidate arguments.
- *Argument selection*: for the same reason explained above, the action selection mechanism of the planner emulates the argument selection process. An important consideration to take into account is that the action selection mechanism in a traditional planning algorithm is implemented as a non-deterministic function, and it does not consider the students' preferences. In contrast, the selection of arguments in our proposal is made on the basis of these preferences. Therefore, we have adapted the action selection function of a planning algorithm in order to consider the student's preferences about argument selection.

The following section shows how an argumentation problem is represented in the planner, as an initial state, a final state and a set of actions.

4. Definition of initial state, final state, and actions to generate argumentations plans

The negotiation information, such as the conflict context, the possible agreements, the amount of implicated students and their beliefs and goals, changes from one negotiation to another. However, some characteristics of the negotiation process, such as the types of arguments a student can use, the rules to generate them and their preconditions and effects, do not change according to the negotiation, but they change with the increment of the experience of the negotiator. Thus, the actions that generate the arguments can be similar in each negotiation, whereas the initial and final states should be defined for each one.

In the following sections, we define general predicates that are part of the negotiation language, how the initial and final states are built, and the actions for argument generation.

4.1. Negotiation language

First, we will define a simple negotiation language *L* that we will use in the planner. This language is composed of predicates that represent the information that the agent has in its mental state: student's goals, beliefs, preferences and abilities, both current and historic, which are inspired in the BDI agents (Rao & Georgeff, 1995) and the negotiation language described in Kraus et al. (1998); information about the negotiation context and types of arguments, appeals, rewards and warnings (Amgoud & Prade, 2004; Ramchurn, Sierra, Godo, & Jennings, 2006). The basic predicates of *L* are:

- *iam(X)*: *X* is the student assisted by the tutor agent.
- *isstudent(Y)*: *Y* is a student who belongs to the group of students.
- *believe(X, B)*: *X* believes *B*. *X* has *B* in its beliefs.
- *isgoal(X, G)*: *X* pursues the goal *G*. *X* has *G* in its goals.

- *prefer(X, G1, G2)*: *G1* and *G2* are *X*'s goals, and *X* prefers to fulfil *G1* over *G2*.
- *imply(A, B)*: *A* implies *B* (It represents the classical inference).
- *cando(X, A)*: *X* can perform the action *A*. It means that *X* has the resources to perform the action *A*, or *X* has the compromise of another participant to perform it.
- *do(X, A)*: *X* will perform the action *A*.
- *pastpromise(X, Y, P)*: *X* promised *P* to *Y*, but did not fulfil yet.
- *wasgoal(X, G)*: *X* pursued the goal *G* in the past.
- *did(X, A)*: *X* performed the action *A* in the past.
- *fulfilled(X, G, A)*: In the past, *X* fulfilled the goal *G* performing the action *A*.
- *appeal(X, Y, Q, J)*: *X* will use an appeal to persuade *Y*. *Q* represents the proposal and *J* is its justification.
- *reward(X, Y, Q, R)*: *X* will use a promise of a future reward to persuade *Y*. *Q* represents the proposal and *R* is the promised reward.
- *warning(X, Y, Q, W)*: *X* will use a warning to persuade *Y*. *Q* represents the proposal and *W* is the damaging effect.

The negotiation information is expressed in *L*.

4.2. Initial and final states

As mentioned before, to generate argumentation plans the initial state *i* must describe the world where the conflict takes place. The initial state will be defined as the negotiation information, and since this information changes from one negotiation experience to another, the initial state will vary according to the negotiation problem too.

On the other hand, the final state *f* represents the state of agreement where the student wants to arrive through the argumentation process. Consequently, the predicates that compose this state will depend on the kind of agreement that the student can reach. In this work, we have considered agreements about task execution, but other types of agreements are also viable. For example, if the expected agreement is that the student *st1* accepts to perform the action *alpha* to solve a given exercise, the final state should include *do(st1, alpha)*.

4.3. Actions

In order to represent argument generation in the planning algorithm, the actions of the plan represent the arguments that the student can utter to other counterparts. Before defining these actions, we will briefly introduce the argument types that the agent can generate in the argumentation-based negotiation context. Three general argument types are defined in the literature about argumentation-based negotiation: appeals (Amgoud & Prade, 2004, define it as explanatory arguments), rewards and threats (Kraus et al., 1998; Sierra et al., 1998), which we have transformed into warnings. Appeals are used to justify a proposal; rewards to promise a future recompense; and warnings to warn of negative consequences if the counterpart does not accept a proposal.

For each argument type, we show the actions to generate it, according to the axioms defined in the framework of Kraus et al. (1998). We distinguish two general structures of actions: *create-argument* actions and *accept-argument* actions. The first ones depict the argument generation process as in rules, whereas the second represent the counterpart's acceptance of the argument. However, we can personalise these actions building a user argumentative model (Monteserin & Amandi, 2008), which captures the rules for argument generation that a user applies during the negotiation.

Next sections show some examples of *create-argument* and *accept-argument* actions defined in the framework of Kraus et al. (1998), which can be captured using the referenced user argumentative model.

4.3.1. Appeals

Varying the premises of the appeals we can define several of them: past promise, counterexample, prevailing practice, self-interest, transitive, and trivial appeals. Examples of these actions are:

- Action: *createPastPromiseAppeal(X, Y, Action)*. Description: it is used to generate appeals to past promises that the opponent did not fulfil. Preconditions: *iam(X), isstudent(Y), pastpromise(Y, X, do(Y, Action))*. Effects: *appeal(X, Y, do(Y, Action), [pastpromise(Y, X, do(Y, Action))])*.
- Action: *createSelfInterestAppeal(X, Y, Action, Goal)*. Description: it allows the participant to generate appeals to self-interest where the proposed action implies a profit to *Y*. Preconditions: *iam(X), isstudent(Y), isgoal(Y, Goal), believe(Y, imply(Action, Goal))*. Effects: *appeal(X, Y, do(Y, Action), [believe(Y, imply(Action, Goal)), isgoal(Y, Goal)])*.
- Action: *acceptAppeal(X, Y, Alpha, Justif)*. Description: it represents the acceptance of the appeal by an opponent. The effects include: the participant's compromise to perform Alpha, and the capacity of participant X to count on that execution *(cando(X, Alpha))*. Preconditions: *appeal(X, Y, do(Y, Alpha), Justif)*. Effects: *do(Y, Alpha), cando(X, Alpha)*.

4.3.2. Rewards

For example, we can define the following actions to promise future rewards:

- Action: *createReward(X, Y, ActionR, ActionP)*. Description: *X* promises to do *ActionR*, if *Y* accepts to do *ActionP*. Preconditions: *iam(X), isstudent(Y), isgoal(Y, Goal), believe(Y, imply(ActionR, Goal))*. Effects: *reward(X, Y, do(Y, ActionP), [do(X, ActionR)])*.
- Action: *acceptReward(X, Y, Alpha, Beta.)*. Description: it represents the acceptance of rewards. In consequence, *Y* undertakes to execute *Alpha* in exchange of the execution of *Beta* by *X*. As in the acceptance of appeals, *X* obtains the ability to execute *Alpha*. Preconditions: *reward(X, Y, do(Y, Alpha), do(X, Beta))*. Effects: *do(Y, Alpha), do(X, Beta), cando(X, Alpha)*.

4.3.3. Warnings

We define the most general warning, but others may be defined varying the preconditions.

- Action: *createWarning(X, Y, ActionP, ActionW, Goal).* Description: if *Y* does not perform *ActionP*, *ActionW* could be performed, and *ActionW* contradicts a goal *GoalA* preferred by *Y*. Preconditions: *iam(X), isstudent(Y), isgoal(Y, GoalA), isgoal(Y, GoalB), prefer(Y, GoalA, GoalB), believe(X, imply(ActionW, not(GoalA))), believe(X, imply(ActionP, not(GoalB))).* Effects: *warning(X, Y, do(Y, ActionP), [do(X, ActionW)]).*
- Action: *acceptWarning(X, Y, Alpha, Beta).* Description: this represents the acceptance of a warning. *X* will not perform *Beta* if *Y* performs *Alpha*. Preconditions: *warning(X, Y, do(X, Alpha), do(X, Beta)).* Effects: *do(Y, Alpha), not(do(X, Beta)), cando(X, Alpha).*

We have only shown actions for the arguments more widespread in the literature and that we will use in following sections. Other actions may be defined varying preconditions and effects. The next section shows how the planning algorithm builds argumentation plans.

5. Planning algorithm for argumentation plan generation

In Rahwan et al. (2003), the authors consider argument selection as the essence of the strategy in argumentation-based negotiation. This mechanism consists in selecting the next argument to be uttered from the set of candidate arguments that might be uttered. Some selection policies are based on the type of argument and the opponents that the student must persuade (argument strength, Kraus et al., 1998), or the trust in the opponents (Ramchurn, Jennings, et al., 2003; Ramchurn, Sierra, Godo, & Jennings, 2003).

In this context, to generate argumentation plans, the action selection mechanism of the planning algorithm must take into account the argument selection policy. In our work, we represent this policy as students' preferences over actions and goals. So, if the student prefers to utter appeals instead of warnings, in the agent's mental state the *create-argument* actions for appeals will have a higher preference level than the *create-argument* actions for warnings. Moreover, these preferences can change in accordance with other factors, such as the trust in a group-mate. For example, as in Ramchurn, Jennings, et al. (2003), when the trust is low, a given student can prefer to use a strong argument (a warning), whereas when the trust is high, he prefers a weak argument (an appeal).

To represent this capability, we used a planning algorithm based on preferences. In general, planning algorithms do not use all the available knowledge in the agent's mental state for the plan conception. For example, in traditional planning (Weld, 1994), the action selection is carried out in a non-deterministic way. The choose function can be implemented by a random selection algorithm or a domain specific algorithm. Planners based on preferences redefine that choose function to take into account the preferences of the users or agents to select actions. So, they allow us to take into account student's preferences over actions and goals for argument generation.

Since our choose function must select the most preferable action, we defined the next format of preferences to be used by the planner: *preference(Q, Ac, Ad, level)*; where *Q* represents the goal to achieve, *Ac* is the action that produces *Q*, *Ad* is the action that needs to accomplish *Q*, and level determines how preferable is the attitude. For example, with *preference(isstudent(p2), createPastPromiseAppeal(_, p2, _), acceptAppeal(p2,_,_,_), 80)* the agent models a preference (e.g. between 0 and 100) of 80 for the goal *isstudent(p2)* when this is produced by the action *createPastPromiseAppeal*, and consumed by the action *acceptAppeal*. In the context of the argumentation process, this mental attitude represents a relative high preference level to use an appeal to past promise, instead of other kind of argument, when the opponent is student *p2*, maybe, because the trust in *p2* is high.

6. Example

In this section, we will present an example to illustrate the construction of an argumentation plan to assist a student during problem solving in CSCL. It is worth remarking that the exercise used in this example was also used in the experiments described in the Section 7.

Consider that in a class, each group of student has to solve the MasterMind game (Berghman, Goossens, & Leus, 2009) using genetic algorithms. MasterMind is a simple code-breaking game for two players. The board game is played using: a decoding board, with a shield at one end covering a row of four large holes, and ten or twelve additional rows containing four large holes next to a set of four small holes; code pegs of six different colours (green, blue, red, yellow, pink, brown), with round heads, which will be placed in the large holes on the board; and key pegs, some coloured (often black), some white, which are flat-headed and smaller than the code pegs; they will be placed in the small holes on the board. In this game one player, the code-maker, chooses a combination of colours and the other player, the code-breaker, tries to guess it by suggesting different combinations. For each combination suggested by the code-breaker, the code-maker answers with a black peg for the pegs correctly guessed and with a white peg for the colours correctly guessed but in the wrong position.

Students are required to solve the problem by using genetic algorithms to guess a combination of four colours, without repetition. Now, we will give some concepts about genetic algorithms vital to understand our example.[2] Genetic Algorithms (GAs) were proposed by John Holland in the 1960s, inspired by the concepts of evolution and natural selection. The algorithm supports a population of individuals, each one representing a possible solution to the problem of interest, MasterMind in our example. Genetic algorithms use techniques such as inheritance, mutation, selection, and crossover (also called recombination). They are modelled loosely on the principles of the evolution via natural selection, employing a population of individuals that undergo selection in the presence of variation-inducing operators such as mutation and recombination (crossover). A fitness function is used to evaluate individuals, and reproductive success varies with fitness. In MasterMind, it is suggested to use as a fitness function a score of 2 for correct colour and position, 1 for correct colour but incorrect position and 0 for incorrect guesses.

The paradigm of GAs might not find the best solution, but it would come up with a partially optimal solution. The main steps are:

a. Randomly generate an initial population $M(0)$.
b. Compute and save the fitness $u(m)$ for each individual m in the current population $M(t)$.
c. Define selection probabilities $p(m)$ for each individual m in $M(t)$ so that $p(m)$ is proportional to $u(m)$.

[2] Notice that Genetic Algorithm is only the topic of the exercise chosen as example.

d. Generate $M(t+1)$ by probabilistically selecting individuals from $M(t)$ to produce offspring via genetic operators (crossover).
e. Repeat step 2 until satisfying solution is obtained.

The items students must solve are the following:

a. Describe how to represent the solution of MasterMind using this technique, that is define each member of the population (called chromosome).
b. Describe how parents will be selected to generate offspring for the next generation
c. Define the operators to be used (crossover, mutation).
d. Define the fitness function.
e. Define which members of the population will be eliminated from one generation and will not survive for the next one.
f. Determine when the algorithm will finish iterating.

In addition students must pursue the following general goals: keep the solution simple and try not to lose good solutions when moving from one generation to the next one.

Moreover, each student in a group may have different personal goals. For example, participant one wants to use mutation to change randomly a colour for another in the solution; he wants to use the roulette wheel selection method to choose parents; and he prefers to use two-point crossover as operator. Participant two, wants to use one-point crossover as operator; she does not like to use mutation; and she prefers using the roulette wheel method to select parents. Finally, participant three wants to use the tournament selection method to choose parents; he prefers using two-point crossover and not using mutation. The order in goals indicates a certain priority among them.

As we can appreciate, the students' goals are conflictive. Therefore, they must negotiate to solve conflicts, and reach an agreed solution. Following the model presented in Section 3, a software agent or tutor can observe this situation and build a general solution with argumentation plans integrated, and assist the student, in order to facilitate conflict resolution. For example, for participant one, the initial state is composed of facts that represent information about the goals of others participants and knowledge about genetic algorithms, which have been learned previously. Thus, the initial state is composed of the following facts: *iam(p1), isstudent(p2), isstudent(p3), isgoal(p2, roulette_wheel_selection), isgoal(p3, two-point), isgoal(p2, not(mutation))*, *isgoal(p3, not(mutation))*, *isgoal(p2, complete_solution), isgoal(p3, complete_solution), believe(p1, imply(not(mutation), not(complete_solution)), believe(p2, imply(accept(roulette_ wheel_selection), roulette_wheel_selection)), problem_definition()*. Since the final goal of the student is to solve the exercise, the final state is *finish_iteration(defined)*, due to the fact that this is the last action that the students have to perform. Fig. 2 shows the general solution generated to assist participant one. This general solution or plan is composed of actions without conflict (full line), conflictive actions (dotted line), and argumentations plans, which support the agreement of conflictive actions. The figure also indicates the items that the student must fulfil (items a–f) in the corresponding action. We denominate *conditional* actions to the conflictive actions, because we add to their preconditions a special one *do(Student, Action)*, which indicates that is necessary build an argumentation plan that supports it. In contrast, the actions without conflicts are named *unconditional*.

In Fig. 2, we can see that the first two actions are unconditional, since there is no information that indicates a conflict. By contrast, the action *define_parent_selection* must be agreed, because there are other alternatives. So, the planning algorithm builds an argumentation plan to persuade participant 2, shown in the same Fig. 2, and an argumentation plan to persuade participant 3.[3] The first argumentation plan consists of a self-interest appeal, that is, an action to create the appeal and an action that represents the acceptance. The same occurs with actions *define_crossover* and *define_mutation*. Finally, the actions *define_members_elimination* and *determine_finish_iteration* are unconditional too.

In summary, the general plan indicates the actions that the student have to carry out to solve the exercise and the arguments that he/she can utter to persuade the other participants. In the following section, we will describe the methodology used and the results obtained from the experimental evaluation of our proposal.

7. Experimental results

7.1. Methodology

As we have previously stated, the goal of this work was to study a new kind of assistance for students who have to argue to solve problems in CSCL. In previous sections, we have shown and exemplified how the argumentation plans can be built. The evaluation was carried out using the e-learning platform named SAVER, where tutor agents assist students that solve exercises using a collaborative application available in this platform. To evaluate our proposal, we compared the results of not assisting students, assisting them with isolated arguments, and assisting them with argumentation plans.

We evaluated our proposal with a set of 39 students of an Artificial Intelligence course at UNCPBA, who solved 3 exercises about AI techniques. We randomly divided the students in 13 groups of 3 students each. The three exercises were composed of a problem description and a set of goals that the participants should try to achieve. This set of goals was different for each participant of the group. The main purpose of presenting different goals to the students was to generate conflicts and promote debates among them. In this context, students should reach consensus as they solved the exercise through negotiation and argumentation.

In order to compare our proposal with other alternatives, we used different kinds of assistance in each exercise. In the first exercise, the students did not receive any kind of assistance; isolated arguments were proposed to students in the second exercise, and argumentations plans where provided in the last one. The exercises were the following[4]:

[3] Due to space limitations, we did not include all the argumentation plans in Fig. 2.
[4] The mention of the different exercises and techniques used is only informative.

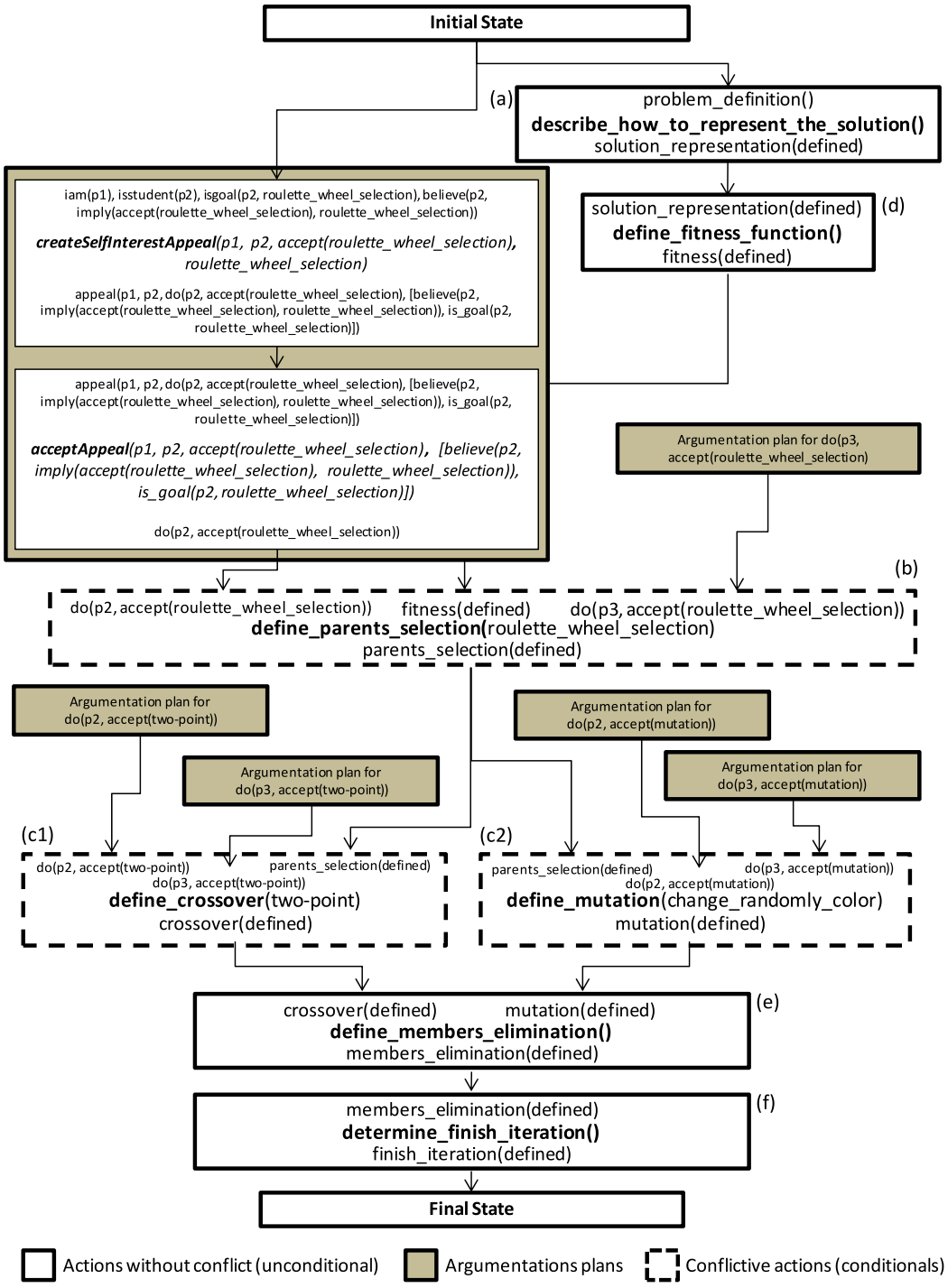

Fig. 2. General plan for participant one.

- *Case Based Reasoning*: students had to use CBR to propose dates, times and places for meetings supposing they have recorded past experience of meetings scheduled by users. Information about CBR technique can be found in Kolodner (1993).
- *Bayesian networks*: students were required to solve the well-known Dog Problem using Bayesian networks (Charniak, 1991).
- *Genetics algorithm*: the exercise consisted in solving the MasterMind (Berghman et al., 2009) game using genetic algorithms (See previous section).

Each exercise and the student's goals for each exercise were loaded into the e-learning platform. SAVER collaborative tool allows students to discuss among them by exchanging semi-structured messages as proposals and arguments in order to solve a given exercise. At the end of this exchange, a shared solution is uploaded to the application. During the discussion, a tutor agent provided assistance to the students. During the CBR exercise, the tutor agent was disabled. In the Bayesian Network exercise the agents suggested students isolated arguments, using the rules for argument generation described in Sections 3 and 4. This suggestion was presented as hints, which indicated

a possible argument to be uttered in the discussion. In the last exercise, the tutor agents built and presented the students an integral plan including the actions needed to solve the exercise and the argumentation plans that supported his/her goals. Fig. 3a shows a snapshot of SAVER describing the contents of one of the topics students had to study, Bayesian Networks. Fig. 3b shows a snapshot of the CSCL tool that the students used for the experiments.

At the end of each exercise, students completed a survey, which asked them questions about the experience. The first aim of these surveys was to contrast the difficulty of reaching agreement that the students experimented. We considered that it was also important to take into account the difficulty of resolution too, since the exercises were about different topics. Moreover, we wanted to know if students would have preferred to have in exercises 1 and 2 the assistance provided for exercise 3. Also, we were interested in knowing how the different assistance actions influenced the knowledge acquisition process, since the final goal of CSCL is learning, and conflict resolution can be considered as a mean to achieve this.

7.2. Results obtained

Considering the goals mentioned before, the first two questions of the survey were: (a) *how difficult was it to solve the exercise?* and (b) *how difficult was it to reach consensus in the conflictive points of the exercise resolutions?*. The answers could take the following values: (1) *very easy*, (2) *easy*, (3) *normal*, (4) *hard*, and (5) *very hard*. In Fig. 4, we can see a graphic that compares the average of the answers for the three exercises.

Observing Fig. 4, we can see that, whereas the difficulty of resolution increased, reaching consensus was easier for exercise 3 than in the other ones, where the assistance was provided using argumentations plans.

In the survey about exercise 3, we included the following: (c) *do you think that if you had had argumentations plans in the previous exercises, reaching consensus would have been...* (1) *harder*, (2) *equal* or (3) *easier than it was?*; and (d) *do you think that if you had had argumentations plans in the previous exercises, its resolution would have been...* (1) *harder*, (2) *equal or* (3) *easier than it was?*. Fig. 5 shows the pie charts with the percentages of responses for each answer. In these charts, we can see that the 66.67% of the students perceived that the argumentation plans had facilitated the consensus in the previous exercises; the 12.82% thought that the consensus had been equally difficult to achieve; and for the remaining 20.51% reaching consensus was harder. Something similar occurred with question d: the 84.62% of the students thought that the resolution of the former exercises would had been easier if they had had argumentation plans; one 7.69% did not think that there had been differences; and the other 7.69% believed that solving the exercise using argumentation plans had been more troublesome.

Finally, we explored how the argumentation plan assistance influences the student's knowledge acquisition. As regards this topic, we asked the students the following questions: (e) *do you think that the argumentation plans suggested by the agent favoured the acquisition of the knowledge about the exercise topic?* and (f) *do you think that having different and conflictive goals and the necessity of reaching consensus favoured the acquisition of such knowledge?*. The answers could take the following values: (1) *nothing*, (2) *few*, (3) *much*, and (4) *very much*.

Fig. 3. Some snapshots of SAVER.

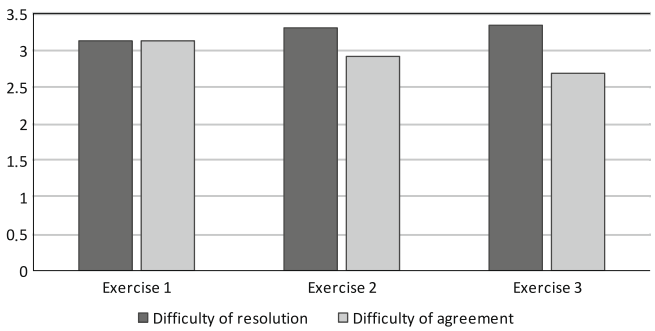

Fig. 4. Comparison of difficulty of resolution and of reaching agreement in the three exercises.

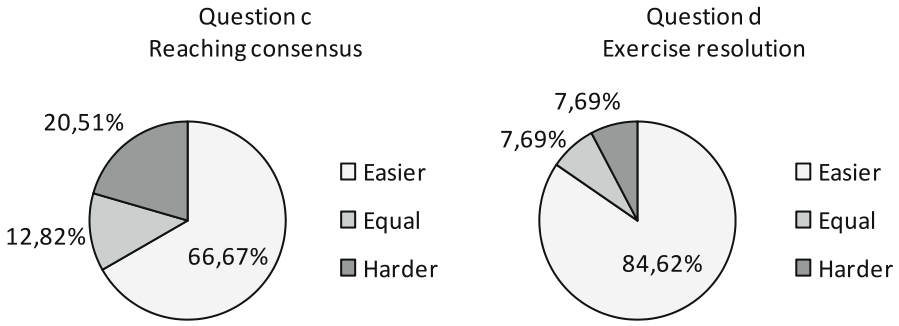

Fig. 5. Difficulty to reach consensus and to solve exercises with argumentation plans.

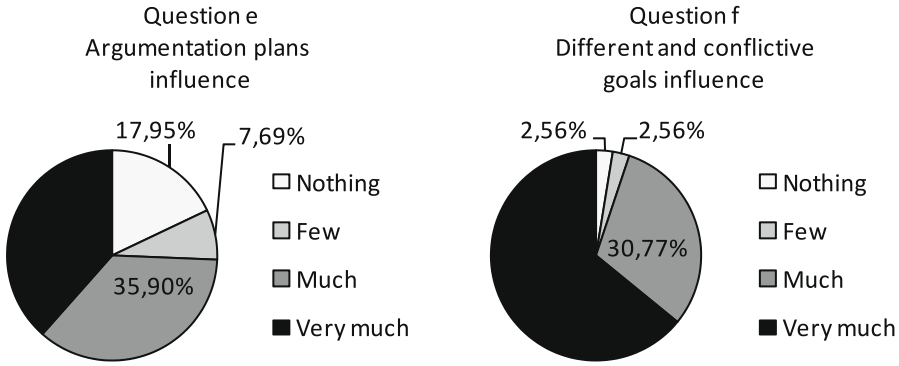

Fig. 6. Influence of argumentation plans and exercises with different and conflictive goals in knowledge acquisition.

Fig. 6 presents the results obtained for both questions. These results indicate that the existence of conflicts that must be resolved using argumentation and the assistance provided by argumentation plans influenced positively the knowledge acquisition process.

In summary, these results show that the assistance by means of argumentations plans facilitated the students to reach consensus easier than the other alternatives. This is reaffirmed by questions c and d, which explicitly inquired the perception of the students. Moreover, the results obtained from question f confirm that learning is particularly effective when students encounter conflicts and resolve them through negotiation generating a shared solution, and results of question e show that argumentation plans assistance facilitates this fact.

8. Discussion and conclusions

We have presented an approach to assist students in a CSCL system using argumentations plans. We explained how the argumentation process may be modelled as a planning problem in order to obtain an argumentation plan; and how the planner mechanisms can emulate the generation and selection of arguments. An argumentation plan is composed of actions that represent the arguments that the student can utter in a conflictive situation in order to resolve the conflict and to reach consensus. A planning algorithm based on preferences builds these plans. The utilisation of the student's preferences to select the best arguments provides a useful versatility to the planning given the diversity of factors that influence the argumentation that can be modelled as preferences.

Moreover, argumentation plans are integrated into the general plan that the students have to follow to solve an exercise. This integration provides a unified view of the argument plans and the general actions that must be agreed, indicating in which moment the argument must be uttered and with which goal. This is especially useful when there are several conflicts in the same general situation (for example, in the exercises used in the experiments there were three different conflicts).

Additionally, it is worth noticing the intuitive character of planning the argumentation. When we negotiate, the arguments uttered are not the result of isolated analysis, but of a unified view of the problem that we want to solve. Also, before the negotiation starts, we have a schematic idea of the arguments that we can utter and we take decisions on the basis of that arguments. That is, we do not sit to negotiate without thinking previously about the alternative of argumentation that we have. Argumentation plans emulate this fact. For this reason, we claim that this is a better alternative to assist students in CSCL.

Surveys were used to record the perception of the participants of this study. The experimental results revealed that the students reach consensus easier when the assistance is provided with argumentations plans. This kind of assistance gives students a unified view of the problem resolution, and allows them to detect relations among the different conflicts, decide which task could be agreed and in which conflict and what they should concede. By observing the general plan, the student can clearly see which arguments must utter in each conflictive situation. Additionally, the idea of planning the negotiation and argumentation is inherent to the negotiation behaviour, hence, a global vision of the argumentation, like the one given by the argumentation plan integrated to the general plan, is better received by the students that the simple assistance with isolated arguments.

Although in our experiments conflicts were forced by giving students different goals, conflicts always arise when a group of students try to solve a problem or exercises in CSCL. We consider that providing students with argumentation plans facilitates conflict resolution but it

does not make students think less or construct less knowledge while debating. That is supported by the results of the last question of the survey (question e and f).

In conclusion, we have obtained promissory results assisting students with argumentation plans. As we have shown, students prefer this kind of assistance instead of the usual assistance given by demand with isolated arguments. However, some challenges remain, such as the need to work with real goals and personalise in an extensive way the argumentation plans. Future work will concentrate on these points. We are interested in learning the preferences for argument selection that the students apply during the conflict resolution in a dynamic way. This will be useful to personalise the argumentation plans in regards to a particular students as well as to the context of the conflict. Finally, we will extend the evaluation to non-controlled scenarios, in which students' real goals are taken into account.

References

Aleven, V. A. (1997). Teaching case-based argumentation through a model and examples. Doctoral Thesis, UMI Order Number: AAI9821228, University of Pittsburgh.

Amgoud, L., & Prade, H. (2004). Generation and evaluation de different types of arguments in negotiation. In *Proceeding of the international workshop on non-monotonic reasoning* (pp. 10–15). Whistler BC, Canada.

Ashri, R., Rahwan, I., & Luck, M. (2003). Architectures for negotiating agents. In V. Marik, J. Mueller, & M. Pechoucek (Eds.), *Proceeding multi-agent systems and applications III* (pp. 136–146).

Baker, M. (1999). Argumentation and constructive interaction. In P. Coirier & J. E. B. Andriessen (Eds.), *Foundations of argumentative text processing*. Amsterdam: Amsterdam University Press.

Berghman, L., Goossens, D., & Leus, R. (2009). Efficient solutions for mastermind using genetic algorithms. *Computers & Operations Research, 36*(6), 1880–1885.

Bouwer, A. (1999). Arguetrack: Computer support for educational argumentation. In R. Pilkington, J. McKendree, H. Pain & P. Brna (Eds.), *Proceedings of workshop on analysing educational dialogue interaction* (pp. 1–8). International conference on artificial intelligence in education 1999, Le Mans, France.

Charniak, E. (1991). Bayesian networks without tears. *AI Magazine, 12*(4), 50–63.

Fikes, R. E., & Nilsson, N. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence, 5*(2), 189–208.

Huang, C., Chen, H., & Chen, C. (2009). Developing argumentation processing agents for computer-supported collaborative learning. *Expert Systems with Applications, 36*(2), 2615–2624.

Jennings, N., Parsons, S., Sierra, C., & Noriega, P. (1998). On argumentation-based negotiation. In *Proceeding of the international workshop on multi-agent systems* (pp. 1–7). Boston, USA.

Jennings, N., Faratin, P., Lomuscio, A., Parsons, S., Sierra, C., & Wooldridge, M. (2001). Automated negotiation: Prospects, methods and challenges. *International Journal of Group Decision and Negotiation, 10*(2), 199–215.

Kochan, E.L. (2006). Analysing graphic-based electronic discussions: Evaluation of student's activity in digalo. *In Proceedings of ECTEL 2006*, (pp. 652–659).

Kolodner, J. (1993). *Case-based reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

Kraus, S., Sycara, K., & Evenchik, A. (1998). Reaching agreements through argumentation: A logical model and implementation. *Artificial Intelligence, 104*(1–2), 1–69.

Loll, F., Pinkwart, N., Scheuer, O., & McLaren, B.M. (2009). An architecture for intelligent CSCL argumentation systems. In *Proceedings of CSCL 2009* (pp. 189–191). Rhodes, Greece.

Monteserin, A., & Amandi, A. (2008). Building user argumentative models. *Applied Intelligence*, in press, doi: 10.1007/s10489-008-0139-6.

Petraglia, J. (1997). *The rhetoric and technology of authenticity in education*. Mahwah, NJ: Lawrence Erlbaum.

Piaget, J. (1977). *The development of thought: Equilibration of cognitive structures*. New York: Viking Penguin.

Pinkwart, N., Aleven, V., Ashley, K., & Lynch, C. (2006). Toward legal argument instruction with graph grammars and collaborative filtering techniques. In *Proceedings of ITS2006* (pp. 227–236).

Rahwan, I., Ramchurn, S., Jennings, N., McBurney, P., Parsons, S., & Sonenberg, L. (2003). Argumentation-based negotiation. *The Knowledge Engineering Review, 18*(4), 343–375.

Ramchurn, S., Jennings, N., & Sierra, C. (2003). Persuasive negotiation for autonomous agents: A rhetorical approach. In C. Reed, F. Grasso, & G. Carenini (Eds.), *Proceedings of the IJCAI workshop on computational models of natural argument*, (pp. 9–17).

Ramchurn, S., Sierra C., Godo L., & Jennings N. (2003b). A computational trust model for multi-agent interactions based on confidence and reputation. In *Proceedings of the workshop on deception, fraud and trust in agent societies* (pp. 69–75). Melbourne, Australia.

Ramchurn, S., Sierra, C., Godo, L., & Jennings, N. (2006). Negotiating using rewards. In *Proceedings of the fifth international joint conference on autonomous agents and multiagent systems (Hakodate, Japan, May 08–12, 2006)* (pp. 400–407). AAMAS '06. ACM, New York, NY.

Rao, A. S., & Georgeff, M. (1995). BDI agents: From theory to practice. In *Proceedings of the first international conference on multi-agent systems* (pp. 312–319).

Schroeder, M. (1999). An efficient argumentation framework for negotiating autonomous agents. In *Proceedings of the 9th European workshop on modelling autonomous agents in a multi-agent world* (pp. 140–149). London, UK.

Sierra, C., Jennings, N., Noriega, P., & Parsons, S. (1998). A framework for argumentation-based negotiation. In *Proceeding 4th international workshop on agent theories, architectures and languages* (pp. 177–192). Rode Island, USA.

Suthers, D. (2003). Representational guidance for collaborative inquiry. In *Confronting cognitions: Arguing to learn* (pp. 1–17). Kluwer Academic Publishers.

Veerman, A. L., Andriessen, J. E., & Kanselaar, G. (2000). Learning through synchronous electronic discussion. *Computers & Education, 34*(3–4), 269–290.

Verheij, B. (2003). Artificial argument assistants for defeasible argumentation. *Artificial Intelligence, 150*(1–2), 291–324.

Weld, D. (1994). An introduction to least commitment planning. *AI Magazine, 15*, 27–61.

Wen, L., & Duh, C. (2000). Human-computer interface for collaborative argumentation. In *Proceedings of the 2000 international conference on microelectronic systems education (November 13 - 13, 2000)*. MSE. IEEE Computer Society, Washington, DC, p. 352.

Yu, R., & Chee, Y. (1999). Using an intelligent agent to improve argumentation skills over hypernews. *Proceedings of ICCE 99—seventh international conference on computers in education Chiba Japan* (Vol. 1, pp. 87–94). Amsterdam: IOS Press.

Yuan, T., Moore, D., & Grierson, A. (2008). A human-computer dialogue system for educational debate: A computational dialectics approach. *International Journal of Artificial Intelligence in Education, 18*(1), 3–26.