A Multi-agent System for Nested Inquiry Dialogues

Chunli Yan, Juan Carlos Nieves, and Helena Lindgren

Department of Computing Science, Umeå University, SE-901 87 Umeå, Sweden

Abstract. Generating and evaluating arguments are two important aspects in argumentation-based dialogue systems. In current research, however, generating and evaluating arguments are normally treated separately. Also, there are rarely implementations of the approaches in real applications. In this paper, we generate inquiry dialogues and evaluate arguments during the dialogue procedure simultaneously. Furthermore, we have implemented this approach in a real medical domain and demonstrated a practical example extracted from this application.

Keywords: Inquiry dialogue, Argumentation framework, Multi-agent system.

1 Introduction

Argumentation has become a core technology in Artificial Intelligence [1,2] and multi-agent systems [3]. The most well-known argumentation framework is the abstract argumentation framework (AAF) presented by Dung in 1995 [4]. There are extensive works on extending AAF, such as value-based argumentation framework [5], bipolar argumentation framework [6] and preference-based argumentation framework [7]. These frameworks mainly focus on the *evaluation* of arguments, calculating the acceptability of arguments.

In softwares that apply the formal augmentation framework to multi-agent systems, it is also important to have specific steps for agents to *generate* dialogues. Black and Hunter [3] do provide a specific strategy for generating dialogues. Their approach has the advantage of providing a specific strategy for agents to follow when choosing which legal move to make where there are more than one, in contrast to most other work [8]. In this paper, we further use queue data structure to reduce the workload compared to [3].

Black and Hunter [3] separate the process of constructing and evaluating arguments. However, Gordon et al. [9] argue that these two should be considered together. Therefore, in this paper we propose a solution, where the evaluation work is conducted within the inquiry dialogues, i.e., we allow the agents to come to partial conclusions within the nested dialogues, which is a method in which the strongest arguments are aggregated to serve the argument evaluation for deciding upon the major topic. We modify the theoretical framework presented by Black and Hunter in [3], and improve the algorithm for implementation.

Y. Demazeau et al. (Eds.): PAAMS 2014, LNAI 8473, pp. 303–314, 2014.

In medical domain, it is common that different physician has his/her own knowledge and viewpoint. The medical rules coming from different guidelines may conflict with each other as well. It leads to contradictory data, which can affect the judgment of physicians. The approach we present in this paper can catch the contradictory data and do reasoning so as to get optimal result to improve the accuracy of diagnosis. In fact, we have already implemented this approach in a real medical software of diagnosing dementia disease.

The paper is organized as follows. Next section presents how arguments and dialogues are formalized. In Section 3 the developed methods for dialogue generation are described. In Section 4 an example in a real application is described. In Section 5 we compare our approach with papers [3] and [9], and the paper ends with conclusions.

2 Argumentation System

This section presents an argumentation system and a definition of dialogues used in our approach. Our approach is based on both *Defeasible Logic Programming* [10] and *inquiry argumentation systems* [3]. We begin by presenting the syntax of the knowledge base of each of our agents.

2.1 Defeasible Knowledge Base

We adapt the notion of defeasible facts and rules presented in [10]. Therefore, a *literal* denotes either an atom α or its negation $\neg\alpha$. The symbols, such as binary connectives $\wedge$, quantifiers $\exists$, $\forall$, implication $\rightarrow$, negation $\neg$ are the same as in first-order logic.

As it is done in defeasible programming [10], a **rule** is denoted as:

$$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$$

such that α_i $(1 \leq i \leq n)$ and β are literals. α_i $(1 \leq i \leq n)$ is called **premise** of the rule and β is called **conclusion** of the rule. Given a rule $r = \alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$, $concl(r) = \beta$.

A **fact** is a rule with a empty set of premises and is denoted by a literal α which is the conclusion of the rule. Rules, in defeasible programming, can be categorized as either *strict rules* or *defeasible rules*. A strict rule specifies that a literal (*i.e.* β) is always a consequence of a finite set of literals (*i.e.* $\alpha_1, \ldots, \alpha_n$), which can never be defeated [10]. A defeasible rule can be defeated by other rules with higher priority.

In order to add a priority level to each rule, the concept of belief is defined as follows:

Definition 1. A **belief**, denoted by B, is a tuple of the form (ϕ, L) where ϕ is a rule and $L \in \mathbb{N}$ which denotes a **preference level** of the belief. Given a belief (ϕ, L), if ϕ is a fact, then (ϕ, L) is called a *state belief*; otherwise, it is called a *domain belief*.

Following the convention in paper [3], we stipulate if there are two beliefs (ϕ_1, L_1) and (ϕ_2, L_2) and $L_1 < L_2$, then (ϕ_1, L_1) is more preferred than (ϕ_2, L_2). If a set of beliefs has the same preference level, we assume neither is preferred over the other.

In order to provide a given agent with a knowledge base, a belief base is defined as follows:

Definition 2. A **belief base** of an agent $x \in \{1, 2\}$, denoted by $\sum_x$, is a finite set of beliefs.

In the following sections, we use x to present one agent and $\hat{x}$ to present the other one, such that if $x = 1$ then $\hat{x} = 2$, and vice versa.

In order to project a set of rules from a belief base of a given agent x with respect to a particular conclusion, we are going to define the concept of *related belief base* as follows:

Definition 3. The **related Belief Base** about literal α with respect to agent x, denoted by $\sum_x^\alpha$ is defined as follows:

$$\sum_x^\alpha = \{(\phi, L) | (\phi, L) \in \sum_x \text{ and } (concl(\phi) = \alpha \text{ or } concl(\phi) = \neg\alpha)\}$$

We use the function $relatedBeliefBase_x(\alpha)$ to return $\sum_x^\alpha$ from $\sum_x$.

Let us illustrate the definition with the following example:

Example 1. Let $\sum_1$ be the belief base of agent 1 which is of the form $\{(\neg a, 1), (b, 3), (a \rightarrow c, 2), (b \rightarrow a, 2), (d \rightarrow e, 2), (\neg a \wedge \neg b \rightarrow \neg c, 2)\}$. Hence, some of examples of related belief bases are:

$relatedBeliefBase_1(a) = \{(\neg a, 1), (b \rightarrow a, 2)\}$;
$relatedBeliefBase_1(b) = \{(b, 3)\}$;
$relatedBeliefBase_1(c) = \{(a \rightarrow c, 2), (\neg a \wedge \neg b \rightarrow \neg c, 2)\}$.

We will define three relations between a belief and a fact: *defend*, *attack* and *irrelevant*.

Definition 4. Let B be a belief, such that $B = (\alpha_0, L_0)$, if it is a state belief; or $B = (\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \alpha_0, L_0)$ if it is a domain belief. Let α be a fact, then:

1. If $\alpha_0 = \alpha$, we say the belief B **defends** the fact α.
2. If $\alpha_0 = \neg\alpha$, we say the belief B **attacks** the fact α.
3. If α_0 equals neither α nor $\neg\alpha$, we say that the belief B is **irrelevant** to the fact α.

Let us illustrate Definition 4 with the following example:

Example 2. Let $\sum_1$ be the belief base introduced in Example 1. We can observe that: $(\neg a, 1)$ attacks a; $(b, 3)$ defends b; $(a \rightarrow c, 2)$ defends c, $(b \rightarrow a, 2)$ defends a, $(d \rightarrow e, 2)$ defends e, $(\neg a \wedge \neg b \rightarrow \neg c, 2)$ attacks c. Except these relations, the others relations are irrelevant relations. For instance, $(\neg a, 1)$ is irrelevant to e.

2.2 Dialogues Representation

Inquiry dialogues, among other types of dialogues, were defined by Walton and Krabbe [11], as having the purpose to collaboratively build new knowledge. In our approach, two agents take part in the proof process (an inquiry dialogue) of a topic in which these two agents do not know if the topic is true or false. Each agent has its knowledge about the given topic. However, they are not able to prove the truth of the topic by themselves; hence, they need to collaborate in order to come up with a conclusion. Their goal is to find and to verify the evidence with respect to a given topic. The goal of an inquiry dialogue is to prove or disapprove the hypothesis in a proof process of a collaborative reasoning.

In order to formalize our dialogue system, we follow the dialogue style introduces by Black and Hunter [3]. Two participating agents use *moves* to communicate with each other in our argumentation system. Three types of moves are allowed: *open*, *assert* and *close*. An *open* move means that an agent opens a new dialogue. An *assert* move means that an agent believes that a given belief is true. A *close* move means that an agent wants to close the current dialogue; however, if another agent does not agree, this dialogue will not be closed.

We use two kinds of inquiry dialogues in our framework: *warrant inquiry* (wi) dialogue and *argument inquiry* (ai) dialogue. A move m is a tuple of the form:

$$m = \langle agent,\ move\ type,\ dialogue\ type,\ topic \rangle$$

in which *agent* denotes which agent makes this move, *move type* denote the kind of move: open, assert, close, and *dialogue type* can be either *wi* or *ai*. If the move is an open/close *wi* move, *topic* is a fact; if it is an open/close *ai* move, *topic* is a domain belief; otherwise, *topic* is a state belief. Since we have two types of inquiry dialogues and three types of moves, there are six types of move formats which are presented in Table 1.

Table 1. Move format

Move	**dialogue**	**Format**
Open	wi	$\langle x,\ open,\ wi,\ \alpha \rangle$
Open	ai	$\langle x,\ open,\ ai,\ (\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \beta, L) \rangle$
Assert	wi	$\langle x,\ assert,\ wi,\ (\alpha, L) \rangle$
Assert	ai	$\langle x,\ assert,\ ai,\ (\alpha, L) \rangle$
Close	wi	$\langle x,\ close,\ wi,\ \alpha \rangle$
Close	ai	$\langle x,\ close,\ ai,\ (\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \beta, L) \rangle$

- $\langle x,\ open/close,\ wi,\ \alpha \rangle$ means that agent x opens/closes a wi dialogue and the topic of the dialogue is α.
- $\langle x,\ open/close,\ ai,\ (\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \beta, L) \rangle$ means that agent x opens/closes an ai dialogue and the topic of the dialogue is $(\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \beta, L)$.
- $\langle x,\ assert,\ wi,\ (\alpha, L) \rangle$ means that this move is within a *wi* dialogue and (α, L) defends / attacks the topic of this dialogue and agent x asserts that (α, L) is true.

– $\langle x,\ assert,\ ai,\ (\alpha, L)\rangle$ means this move is within an *ai* dialogue and agent x asserts that (α, L) is true and α is one of the *ai* topic's premises. Let us observe that (α, L) is not from the agent's belief base. It is from the result store (which will be described in next section) that has already been proved to be true by the two agents.

In order to formalize our dialogue system, we follow the definitions about *dialogue, sub-dialogue* and *well-formed dialogue*, which were introduced by Black and Hunter [3]. Therefore, we only give a brief descriptions of these. For detailed descriptions of these concepts, we refer the interested reader to [3].

Definition 5. A **dialogue** D_r^t (r,t $\in \mathbb{N}$ and r$\leq$ t) is a sequence of moves $[m_r, \ldots, m_t]$ with two agents participating in that: (1) the first move of the dialogue is an open move; (2) each agent takes its turn to make moves. A **sub-dialogue** is a sub-sequence of another dialogue. A **well-formed dialogue** is a dialogue where (1) the last two moves must be close moves made by two agents successively which means both agents agree to close the dialogue; (2) this dialogue only terminates once; (3) all its sub-dialogues are also well-formed and terminate before their parent dialogue.

3 Modeling Dialogues

In this section, we go through the details of generating dialogues and evaluating arguments. We first define some notations (data structures: PBQ, QS, CS and RS and outcomes of dialogues: $Outcome_{ai}$ and $Outcome_{wi}$) needed to generate the dialogues, then give the specific protocols for generating the two different dialogues: wi and ai. The purpose of wi dialogue is to generate several arguments defend or attack its topic and compare these arguments. The purpose of ai dialogue is to detect if the topic rule is fulfilled, i.e., if all its premises can be proved to be true, and generate an argument if so.

In a wi dialogue, we use a *Possible Beliefs Queue (PBQ)* to store the belief's $relatedBeliefBase$ according to a topic, so that it can pick up the first belief from this queue when it needs to make a move.

Definition 6. A **Possible Beliefs Queue (PBQ)** is a queue of beliefs that the agent can legally use for selecting the next move for the current wi dialogue. Let D_r^t be the current dialogue and $\mathcal{I}$ be the set of participants. For all $x \in \mathcal{I}$,

$$PBQ_x^t(\alpha) = \begin{cases} relatedBeliefBase_x(\alpha), & \text{iff } m_t = \{x, open, wi, \alpha\} \text{ or} \\ & \quad m_{t-1} = \{\hat{x}, open, wi, \alpha\} \\ relatedBeliefBase_x(\alpha) - (\phi, L), & \text{iff } m_t = \{x, open, ai, (\phi, L)\} \\ relatedBeliefBase_x(\alpha) - (\alpha, L), & \text{iff } m_t = \{x, assert, wi, \alpha\} \\ PBQ_x^{t-1}(\alpha), & otherwise \end{cases}$$

When agent x opens a wi dialogue with topic α, it updates its PBQ according to $relatedBeliefBase_x(\alpha)$ and next time, agent $\hat{x}$ updates its PBQ. Within the

wi dialogue, the agent retrieves and deletes the first belief in its PBQ and use this for its next move. Only when the agent's PBQ is empty, i.e., it has nothing more to say about the current wi topic, it makes a close wi move.

When an agent opens an argument inquiry dialogue with the topic (Φ, L), a query store associated with this topic is created which is shared between two agents. Within an ai dialogue, if an agent needs to make a move, it can consult query store and get the first fact in it and make an open wi move.

Definition 7. A **query store** QS^t_Φ is a finite queue of facts such that

$$QS^t_\Phi = \begin{cases} \{\alpha_1, ..., \alpha_n\}, & \text{iff } m_t = \langle x, open, ai, (\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \beta, L)\rangle \\ QS^{t-1}_\Phi - \alpha\ , & \text{iff } (m_t = \langle x, open, wi, \alpha\rangle \ or\ m_t = \langle x, assert, ai, (\alpha, L)\rangle) \\ & \quad and\ \alpha \in QS^{t-1}_\Phi \\ \emptyset, & \text{iff } \alpha_i \in QS^{t-1}_\Phi\ and\ m_t = \langle x, close, wi, \alpha\rangle\ and \\ & \quad m_{t-1} = \langle \hat{x}, close, wi, \alpha\rangle\ and\ Result(\alpha) \neq T \\ QS^{t-1}_\Phi, & \text{otherwise} \end{cases}$$

When an agent makes an open ai move, the premises of its topic rule are stored in a query store. Within this ai dialogue, if the move $\langle x, open, wi, \alpha\rangle$ or $\langle x, assert, ai, (\alpha, L)\rangle$ is made, query store removes α. Another case is, within the ai dialogue, if a wi dialogue terminates (whose topic is a premise of this ai dialogue topic) and this premise can not be proven true ($Result(\alpha) \neq T$ is given in definition 11), then a conclusion that the ai's topic is not fulfilled can be made without any further steps. The query store is thus emptied.

PBQ and QS are two core data structures we use for storing beliefs and selecting next moves. They are two *queues* so that they follow the fundamental principle of queues, such as *first in first out (FIFO)*. We also can use some common operations to these two queues. Each agent has its own PBQ which both facts and rules are stored in it. PBQ is used for agent to select the next exact move in wi dialogue. If the first belief in PBQ is a rule, the agent makes an *open ai* move; else if the belief is a fact, it makes an *assert wi* move; else the queue is empty and it makes a *close wi* move. Both agents share the same QS which only stores facts. QS is used for agents to select move in ai dialogue. If QS is empty, the agent makes a *close ai* move; else it makes an *open wi* move.

Whenever an agent takes part in a dialogue, its commitment store will be update. In order to identify the state of the commitment store of each agent which participate in a given dialogue D^t_r, CS^t_x denotes the commitment store of the agent x and t denotes a point in the dialogue D^t_r.

The update of commitment store (CS), outcome of ai dialogue ($Outcome_{ai}$) and outcome of wi dialogue ($Outcome_{wi}$) are recursive. For updating CS, we need to get $Outcome_{ai}$. For getting $Outcome_{ai}$, we need to calculate $Outcome_{wi}$. For calculating $Outcome_{wi}$, we need to know CS.

The update of the commitment stores of each agent is done as follows ($Outcome_{ai}$ will be defined in definition 9).

Definition 8. Let D_r^t be the current dialogue and $\mathcal{I}$ be the set of participants. For all $x \in \mathcal{I}$,

$$CS_x^t = \begin{cases} \emptyset, & \text{iff } t = 0, \\ CS_x^{t-1} \cup \{(\alpha, L)\}, & \text{iff } m_t = \langle x, assert, wi, (\alpha, L)\rangle, or \\ & \quad Outcome_{ai}(D_r^t) = (\alpha, L) \\ CS_x^{t-1}, & \text{otherwise.} \end{cases}$$

According to Definition 8, the commitment store of each agent is updated whenever it performs an assert wi move or when the ai dialogue closes. An important consequence of this update is that the information which is added to the commitment store is public to the other agents which are taking part in the given dialogue.

When an ai dialogue terminates, its outcome is calculated. If all the premises of its topic are considered to be true ($Outcome_{wi} = \langle T, l\rangle$, which is given in definition 10), the outcome is a belief constructed with the rule's conclusion and a calculated preference level; otherwise, the outcome is empty.

Definition 9. Let D_r^t be a well-formed argument inquiry dialogue and $(\alpha_1 \wedge ... \wedge \alpha_n \rightarrow C, L))$ be its topic. **Outcome of argument inquiry dialogue** is a function that:

$$Outcome_{ai}(D_r^t) = \begin{cases} \{(C, L')\}, & \text{iff } \forall \alpha_i (i \in \{1, ..., n\} \; and \; Outcome_{wi}(D_{r_i}^{t_i}) = (T, l_i) \; and \\ & \quad Topic(D_{r_i}^{t_i}) = \alpha_i) \; and \; L' = max(l_1, ..., l_n, L) \\ \emptyset, & Otherwise \end{cases}$$

Within a wi dialogue, several arguments defending or attacking the topic α may be generated. When this wi dialogue terminates, its outcome is calculated according to an algorithm which will be given in Table 2. The outcome is a tuple $\langle r, l\rangle$ where $r \in \{T, F, U\}$. If the defending arguments win, $r = T$ meaning α is $True$; Else if the attacking arguments win, $r = F$ meaning α is $False$. In both cases, l is a natural number which can be calculated from the algorithm. However if the two sides are well matched, $r = U$ which means the result is undetermined and l is empty.

Definition 10. Let D_r^t be a well-formed argument inquiry dialogue and α be its topic. **Outcome of warrant inquiry dialogue** is a function such that: $D_{wi} \mapsto \{T, F, U\} \times (\mathbb{N} \cup \emptyset)$.

Before giving the algorithm, let us show several functions used in the algorithm.

The first function F_d is to get all the beliefs that defend a topic α from a set of domain belief bases Λ.

The second function F_a is to get all the beliefs that attack a topic α from Λ.

The third function LS is to get the smallest preference level from a nonempty set Λ.

The forth one F_l is to get all the beliefs with a particular preference level from Λ.

The last one *Amou* is to get the number of the beliefs in Λ.

The main idea about the algorithm is as follows. First, classify beliefs from the union of two commitment stores into two sets: Λ_d and Λ_a - according to if the belief defends or attacks a given topic. Second, get the smallest preference levels (the highest priority) from each set and compare these two numbers. Third, the set with smaller number wins. However if they have the same number, remove the beliefs with the smallest preference level from each set and get two new sets. We compare the new sets until one set wins or both become empty.

Now we can give the algorithm to get $Outcome_{wi}$ in Table 2.

Table 2. Algorithm of getting $Outcome_{wi}(D_r^t)$

Input: a warrant inquiry dialogue D_r^t with α as its topic; Output: $\langle r, l\rangle$.

1 $\Lambda_d = F_d(CS_x^t \cup CS_{\hat{x}}^t, \alpha)$ and $\Lambda_a = F_a(CS_x^t \cup CS_{\hat{x}}^t, \alpha)$.
2 If $\Lambda_d = \emptyset$ and $\Lambda_a = \emptyset$, then $r = U$ and $l = \emptyset$.
3 Else if $\Lambda_d \neq \emptyset$ and $\Lambda_a = \emptyset$, then $r = T$ and $l = LS(\Lambda_d)$.
4 Else if $\Lambda_d = \emptyset$ and $\Lambda_a \neq \emptyset$, then $r = F$ and $l = LS(\Lambda_a)$.
5 Else

- If $LS(\Lambda_d) < LS(\Lambda_a)$, then $r = T$ and $l = LS(\Lambda_d)$.
- Else if $LS(\Lambda_d) > LS(\Lambda_a)$, then $r = F$ and $l = LS(\Lambda_a)$.
- Else
  - If $Amou((\Lambda_d) > Amou((\Lambda_a)$, then $r = T$ and $l = LS(\Lambda_d)$.
  - Else if $Amou((\Lambda_d) < Amou((\Lambda_a)$, then $r = F$ and $l = LS(\Lambda_a)$.
  - Else $\Lambda_d = \Lambda_d - F_l(LS(\Lambda_d))$ and $\Lambda_a = \Lambda_a - F_l(LS(\Lambda_a))$ and loop from step 2 again.

It could be the case that different rules have the same premise. If the premise has already been proved before (a wi dialogue with this premise as topic has already terminated), the system should not prove it twice. Otherwise, it is a repetitive work. We use *result store* to save the intermediate result.

Definition 11. A **result store** RS is a set of tuples $\langle \alpha, Outcome_{wi}(D_r^t)\rangle$ where α is a defeasible fact and the topic of D_r^t is α. If $Outcome_{wi}(D_r^t) = \langle r, l\rangle$, r is returned by a function $Result(\alpha)$ such that $Result(\alpha) = r$; while l is natural number and returned by a function $PL(\alpha)$ such that $PL(\alpha) = l$.

Now we give the protocols for generating warrant inquiry dialogue and argument inquiry dialogue in table 3 and 4.

4 Example

We implemented our approach in a medical application diagnosing dementia disease [12]. Here we use a study case as an example to illustrate how we generate nested dialogues and make decision about a topic.

Table 3. Step of Warrant Inquire Dialogue Protocol

1	Agent x starts a warrant inquire dialogue D_r^t with the topic α: $m_r = \langle x, open, wi, \alpha \rangle$.
2	Both agents x and $\hat{x}$ update their possible belief queue according to Definition 6.
3	Agent $\hat{x}$ performs the moves m_i ($i \in \{r+1, r+3, \ldots, t_1\}$) and x performs the moves m_j ($j \in \{r+2, r+4, \ldots, t_2\}$), such that $t = max(t_1, t_2)$ and the difference between t_1 and t_2 is 1. Both m_i and m_j are of the following form: – $\langle \hat{x}, assert, wi, (\alpha, L) \rangle$ such that $(\alpha, L) \in PBQ_{\hat{x}}^t(\alpha)$. The commitment store of the agent $\hat{x}$ is updated according to Definition 8. – $\langle \hat{x}, open, ai, (\alpha_1 \wedge \cdots \wedge \alpha_1 \rightarrow \alpha, L) \rangle$ such that $(\alpha_1 \wedge \cdots \wedge \alpha_1 \rightarrow \alpha, L) \in PBQ_{\hat{x}}^t(\alpha)$. – $\langle \hat{x}, close, wi, \alpha \rangle$ if the agent is unable to perform one of the previous steps.
4	When the dialogue closes, the result store is updated according to Definition 11.

Table 4. Step of Argument Inquire Dialogue Protocol

1	Agent x starts a warrant inquire dialogue D_r^t with the topic α: $\langle x, open, ai, (\alpha_1 \wedge \cdots \wedge \alpha_1 \rightarrow \alpha, L) \rangle$.
2	The query store is updated according to Definition 7.
3	Agent $\hat{x}$ performs the moves m_i ($i \in \{r+1, r+3, \ldots, t_1\}$) and x performs the moves m_j ($j \in \{r+2, r+4, \ldots, t_2\}$), such that $t = max(t_1, t_2)$ and the difference between t_1 and t_2 is 1. Both m_i and m_j are of the following form: – $\langle \hat{x}, assert, ai, (\alpha, L) \rangle$ such that $(\alpha, L) \in RS$. – $\langle \hat{x}, open, wi, (\alpha, L) \rangle$ such that $(\alpha, L) \in QS_{\Phi}^t$. The query store is updated according to Definition Definition 7. – $\langle \hat{x}, close, ai, \alpha \rangle$ if the agent $\hat{x}$ is unable to perform the previous step.
4	When the ai dialogue terminates, the outcome of the dialogue is calculated according to Definition 9; and the commitment store is updated according to Definition 8.

In this example, there are two agents: physician agent (PA) and domain agent (DA). PA diagnoses a patient and suspects that she has got a mild cognitive impairment. However PA has not enough experience to make a decision. Therefore, PA collaborates with DA in a diagnostic dialogue with the purpose to validate the hypothesis.

All the moves generated by two agents during the dialogue are shown with natural language in Fig.1. In the figure, each line starts with a number, followed by the agent name and the context of the move which means at which step, which agent (PA/DA) presents this move context. The whole figure is a wi dialogue with the topic *Mild Cognitive Impairment (MCI) is present* made by PA (can be seen from step 1 in Fig.1). Under this dialogue, there are several nested ai dialogues whose information are collapsed and can be shown by clicking the corresponding triangles (e.g. 2, 126...) in the application.

PA initiates a wi dialogue (step 1). PA and DA update their PBQs according to definition 6. DA has at least five rules in its PBQ now since we can see five ai

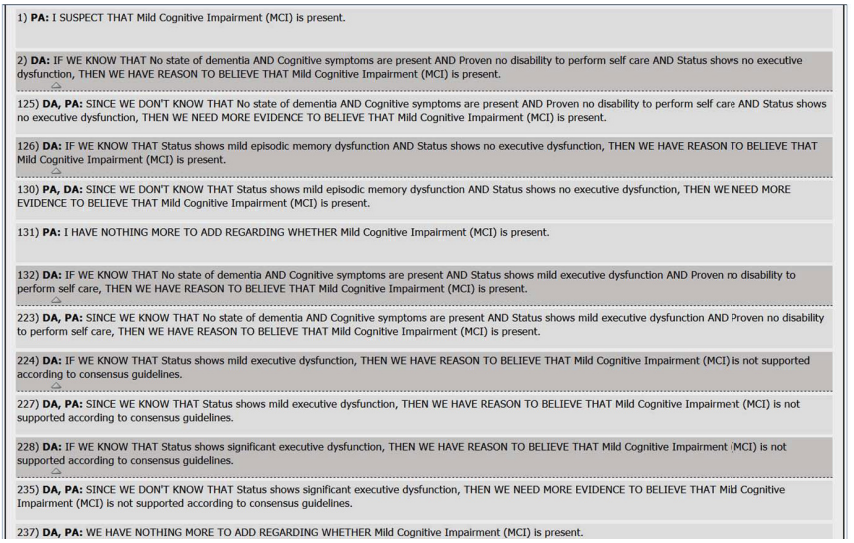

Fig. 1. Moves generated by the two agents

dialogues (from steps 2, 126, 132, 224 and 228) in the picture. DA picks up the first one in its PBQ and opens an ai dialogue (definition 6) in step 2 and stores the premises to QS (definition 7). At next step (step 3, which is not shown in the figure), DA opens a nested wi dialogue with the first premise in QS as its topic (definition 7). When the nested wi dialogue closes, its result is stored in RS (definition 11). When the nested ai dialogue closes, its outcome is stored in CS (definition 8). In this example, the outcome of one ai dialogue (from step 132 to 223) defends the topic *Mild Cognitive Impairment (MCI) is present* and the other (from step 224 to 227) attacks it. Then at the last step, the agents compares these two following the comparison algorithm presented after the definition 11. If the preference level of the rule used in the former ai dialogue is smaller/bigger than the second one, the result should be *Mild Cognitive Impairment (MCI) is present* is *True/False*. In this example, the two preference levels are equal and there are no more arguments, which defend/attack the topic. Therefore, the result about the topic is *Undetermined*.

5 Related Work

There are extensive research done on formal argumentation with focus on the evaluation of arguments, such as Dung's abstract argumentation framework [4] and its successors [5,6,7]. There are surprisingly few contributions, which focus on both constructing and evaluating arguments according to a set of potentially defeasible rules and facts. We have already mentioned the inquiry dialogue systems presented by Black and Hunter's [3], which is similar in some aspects to

the approach presented in this paper. Both adapt Defeasible Logic Programming for representing the beliefs; define the same move types; and generate warrant inquiry and argument inquiry dialogues. However, the two approaches have significant differences, mainly in the different protocols used for generating the sequence of moves and the evaluation mechanism.

Black and Hunter divide the dialogue systems into two processes: the construction of arguments and the evaluation of arguments. They first generates a set of arguments during the inquiry dialogue and then constructs a dialectical tree with these arguments to evaluate the acceptability of the root node. When implementing the system, each process needs to use at least one loop, which is unnecessarily time-consuming. By contrast, we construct and evaluate arguments simultaneously. In our approach, the evaluation is accomplished within the dialogue procedure.

In order to restrict the discussion scope, we allow a wi dialogue to be nested in an ai dialogue while it is not allowed in [3]. We do like this because we want to avoid the following situation: within an ai dialogue with the topic $(\alpha_1 \wedge ... \wedge \alpha_n \rightarrow \alpha)$, agent x discusses one premise and agent $\hat{x}$ another, then the agents will be confused by the dialogue. For each premise α_i, a wi dialogue may be opened and two agents are only limited to talk about α_i until the dialogue ends.

Both approaches can select a single next move within a set of possible legal next moves which makes it stronger than other dialogue approaches presented in research literature. In [3] each move content is assigned arbitrarily a unique number and these numbers are compared according to a function to determine the next move. We use queue, since queue has the inherent feature of FIFO. Specifically, we save possible next Open wi and Assert ai moves in QS and Open ai and Assert wi moves in PBQ. Therefore, we resolved the problem without additional workload.

Our evaluation mechanism is somewhat similar to Carneades [9]. The Carneades model can be mapped to our approach. A statement node in Carneades is the same as a literal (premise and conclusion) in our model and an argument node can be mapped as a rule in ours. Supporting and contradictory arguments can be mapped as two conflict rules so that their conclusions are α and $\neg\alpha$ respectively. The result of a wi dialogue is like the acceptability of a statement and the result of an ai dialogue is like the defensibility of an argument. The decision about the outcome of a wi/ai dialogue is recursive and the process is comparable to what Carneades does in the acceptability of statement and the defensibility of an argument.

However there are two significant differences between Carneades and our work. 1) Carneades does not define dialogue protocol, roles and speech acts; while these are the main building blocks in our paper. 2) The Carneades model focuses on persuasion dialogue while ours is on inquiry dialogue.

6 Conclusions

In this paper we present our framework for generating inquiry dialogues and comparing arguments. We supply details that allow agents to select a precise

step at each particular time, not only give them several legal moves. Following our approach, a dialogue can be generated and a result will be reached. In this framework, we generate two kinds of inquiry dialogues: warrant inquiry dialogue and argument inquiry dialogue. The goal of a warrant inquiry dialogue is to determine if its topic is true, false or undetermined. The goal of an argument inquiry dialogue is to generate a valid argument. Two kinds of dialogues are nested within each other in order to reach a valid decision. We have implemented our approach in a real medical application and received positive feedback.

Finally, the human agent needs to be able to participate both in the dialogue, aggregating arguments and evaluating the arguments. Therefore, in future work, the implemented multi-agent system will be extended in the relevant domain. Moreover, we will improve the visualization of the dialogue procedure with a graph similar to [9] so that it can become more intuitive.

Due to the page limit here, we would rather provide the formal properties (soundness and completeness) in a longer version of this paper.

References

1. Bench-Capon, T.J.M., Dunne, P.E.: Argumentation in artificial intelligence. Artificial Intelligence 171(10-15), 619–641 (2007)
2. Nieves, J.C., Osorio, M., Cortés, U.: An overview of argumentation semantics. Computación y Sistemas 12(1), 65–88 (2008)
3. Black, E., Hunter, A.: An inquiry dialogue system. Autonomous Agents and Multi-Agent Systems 19(2), 173–209 (2009)
4. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–357 (1995)
5. Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. Journal of Logic and Computation 13(3), 429–448 (2003)
6. Amgoud, L., Cayrol, C., Lagasquie-Schiex, M.C., Livet, P.: On bipolarity in argumentation frameworks. International Journal of Intelligent Systems 23(10), 1062–1093 (2008)
7. Amgoud, L., Cayrol, C.: A reasoning model based on the production of acceptable arguments. Annals of Mathematics and Artificial Intelligence 34(1), 197–215 (2002)
8. Parsons, S., Wooldridge, M., Amgoud, L.: On the outcomes of formal interagent dialogues. In: Second International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2003), Melbourne, Australia, pp. 616–623 (2003)
9. Gordon, T.F., Prakken, H., Walton, D.: The carneades model of argument and burden of proof. Artificial Intelligence 171(10), 875–896 (2007)
10. García, A.J., Simari, G.R.: Defeasible logic programming: An argumentative approach. Theory and Practice of Logic Programming 4(1-2), 95–138 (2004)
11. Walton, D.: The New Dialectic: Conversational Contexts of Argument. University of Toronto Press, Toronto (1998)
12. Yan, C., Lindgren, H.: Hypothesis-driven agent dialogues for dementia assessment. In: International Workshop on Agents Applied in Health Care (AAHC 2013), Murcia, Spain (2013)