April 2024

Protocol Switching in a Multi-Agent Dialogue System

Nicole ORR [1] and John LAWRENCE
Centre for Argument Technology (ARG-tech), University of Dundee, UK
ORCiD ID: Nicole Orr https://orcid.org/0009-0001-9055-671X, John Lawrence https://orcid.org/0000-0003-1162-097X

Abstract. Dialogue protocols define how a dialogue may proceed and the moves its participants can make within it. The scope of this work covers protocol switching, and addresses the current gap in the area of illicit protocol switches in dialogue. Over the course of a dialogue the participants' goals and strategies may change in response to the other participants within the dialogue. Enabling agents to switch between protocols gives them the flexibility to address these changes and make use of them. This paper introduces protocol switching using Dialogue as a Service (DaaS), a platform for building multi-agent dialogue systems. DaaS can be used to create a wide range of multi-agent dialogue systems due to its few restrictions and inherent flexibility, which is illustrated through the use of two examples from the literature. Protocol switches can be both licit and illicit; however, current research has only focused on implementing licit protocol switches. An approach to facilitating and managing illicit protocol switches is demonstrated herein.

Keywords. dialogue protocols, protocol switching, dialogue systems, multi-agent systems

1. Introduction

Dialogue protocols [1] are used in argumentation-based dialogue [2] to define how a dialogue may proceed. They define everything from how and when the dialogue can start and end, the number of participants involved, the moves participants can make and their effects. Protocol switching refers to the changing of the dialogue protocol one or more of the participants is using during a dialogue.

Protocol switching, also referred to as dialectical shift by Walton and Krabbe [3], can be either licit or illicit. They define dialectical shift as a change from one type of dialogue to another, a change in the context and "flavour" of the dialogue. Licit protocol switches are constructive shifts to the dialogue that have been openly made and agreed to by all of the dialogue's participants. In contrast, illicit protocol switches are those that are inappropriate or concealed from one or more of the dialogue's participants. Often in natural language conversations several protocol switches will occur over the course of the dialogue.

[1] Corresponding Author: Nicole Orr, nicole@arg.tech

There has been limited research into how licit protocol switches can be implemented and managed within argumentation-based dialogue systems [4,5,6,7,8,9]. All of these systems operate under the assumption that all participants in a dialogue are using the exact same protocols and when a switch occurs all participants then use the same new protocol that was agreed upon. This paper addresses the current gap of implemented illicit protocol switches by describing how Dialogue as a Service (DaaS), further explained in Section 3, can be used to create multi-agent dialogue systems that can manage dialogues containing either or both licit and illicit protocol switches[2]. DaaS can facilitate participants using different protocols from each other within the one dialogue. This is important for illicit switches where only one participant in the dialogue switches the protocol they are using.

Illicit protocol switches are often associated with fallacies. Walton and Krabbe state that many fallacies are caused by the occurrence of an illicit shift from one type of dialogue to another. The difference between the same utterance being fallacious or non-fallacious is dependent upon the context of the dialogue in which it has occurred. Several utterances are valid moves within the context of one protocol but become fallacious when misused in a different type of protocol. Section 5 demonstrates how an illicit protocol switch from a persuasive protocol to a negotiation protocol causes the fallacy of bargaining to occur.

2. Related Work

2.1. Types of Dialogues

Walton and Krabbe describe seven different general types of dialogue [3] which they have grouped according to their primary purpose and rules. They acknowledge that their list is not exhaustive and that there are many more types of dialogue, in both more general and specific categories. The seven types they distinguish between are: persuasive dialogue, negotiation, inquiry [9], deliberation [10], information-seeking dialogue, eristics, and mixed.

The examples in Sections 4 and 5 use the types of negotiation, persuasive and information-seeking dialogues. Negotiation occurs between a group of agents who have conflicting interests but are willing to cooperate to decide and agree upon a division of resources amongst themselves [11]. Persuasion dialogues occur when one agent tries to convince another of a proposition that they do not currently hold [12]. Information-seeking dialogues occur when one agent has one or more questions to which they are seeking an answer for from another agent who they believe knows the answer [13].

2.2. Dialogue Combinations

McBurney et al [4] introduce the term “dialogue combinations” to define how iteration, sequencing, parallelisation, embedding, and testing of dialogues can occur. They also describe a control layer used to represent the selection and transition between dialogue types in these combinations. Licit protocol switching is facilitated by the control layer as it allows the agents to discuss and agree upon the selection and transition of protocols.

[2] `https://github.com/Nicole145/Protocol-Switching-using-DaaS`

Sklar and Azhar [5] also use a control layer to manage licit protocol switches, which enables these different combinations of dialogues between their two agents used to represent a human and a robot.

2.3. Dialogue Systems

The Dialogue Game Execution Platform (DGEP) [14] executes dialogue games that have been expressed through the Dialogue Game Description Language (DGDL) [15], further explained in Section 3.1, and builds Argument Interchange Format (AIF) [16] graphs. DGEP uses mixed initiative argumentation [17] and its agents can optionally use the Argument Web [18] as their knowledge base. Despite the wide variety of dialogue games that can be created with DGDL, DGEP does not support any protocol switching at all. It only uses a single protocol for each dialogue that all agents must follow.

More recently, [6] makes use of licit protocol switching by allowing their agents to "digress" from the main dialogue into sub-dialogues when needed in order to resolve issues between the agents. These sub-dialogues are embedded within the main dialogue. The most prominent difference between their protocols for the main and sub-dialogues is the use of commitment stores and the restricted number of moves that can be made within a sub-dialogue before it must terminate, and the main dialogue resumes.

3. Dialogue as a Service

Dialogue as a Service (DaaS) is a platform for building multi-agent dialogue systems. It was created to allow as much flexibility as possible when building dialogue systems to facilitate and manage argumentative dialogue. This flexibility enables a wide range of dialogues to be conducted with DaaS, from more formal idealistic dialogues to more natural language conversation dialogues as the decentralised agents are independent of one another. This freedom particularly enables dialogues where illicit switches in protocols are possible.

Figure 1 shows an example of how each of the components used in DaaS communicate with one another. Three agents are shown within Figure 1 to represent different options for the agent interface components, with Agent 1 and Agent 2 representing a human participant and Agent 3 representing a virtual participant. The agent interface components for Agents 1, 2 and 3 also show how optional agent strategies and domain specific knowledge bases can be added to an agent. Although there is no limit, Figure 1 only shows one particular setup of a multi-agent dialogue system. Each agent uses an instance of the same protocol management API component. The agent interface component varies based upon the type of agent required. It should be adapted depending on if the agent is representing a human or virtual participant, the appropriate agent strategy and any potential domain specific knowledge bases the agent should have access to. All agents communicate with the same instance of DaaSdb, which consists of a database to capture dialogue specific data and an API to interact with it. DaaSdb processes these requests and handles any needed requests to AIFdb [19] for the agent. All requests from DaaSdb to an agent are sent using xAIF which is an extended version of the language of the Argument Interchange Format (AIF) ontology [16]. Therefore DaaSdb contains the data needed to understand a dialogue's progression and AIFdb contains the underlying argument structures for that dialogue.

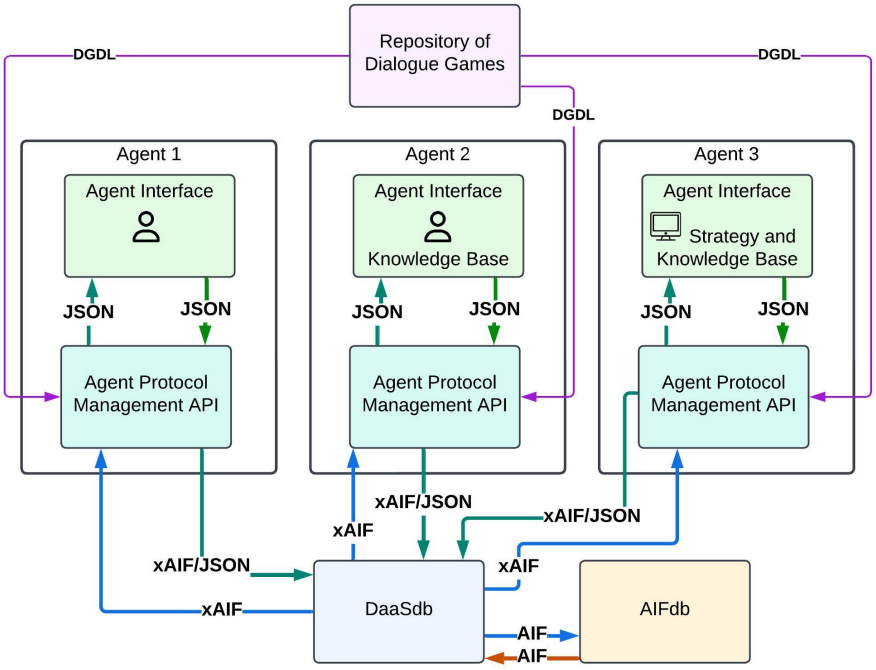

Figure 1. DaaS Components Overview

3.1. Dialogue Protocols using DGDL

The Dialogue Game Description Language (DGDL) is a domain specific language that is used to describe dialectical games [15]. A DGDL description consists of three core elements: a composition, the rules and the interactions. The composition defines how the game operates and player-related information such as the number of players and their roles. The rules define any effects that should be executed at specified points, including an "initial" rule to define how the dialogue game can begin. The interactions define the moves a player can make and the associated effects of that move.

A new version of DGDL was created for DaaS to specify the dialogue protocols used by the agents. This updated version enables protocol switching by defining each protocol one after the other within the one DGDL file. The agent protocol management API component contains a python package to parse the new version of DGDL. Using DGDL to define the protocols allows them to be easily reused for the agents, and a repository of DGDL files has been created for this purpose.

3.2. Agents

The distributed design of DaaS allows the agents to manage themselves according to their own protocols and enables using different protocols for the agents within any dialogue. The agents are split into two components: an agent protocol management API and an agent interface that can contain the agent's strategy and any additional domain

specific knowledge bases. The agent components communicate with one another through JSON requests, for example when the agent protocol management API sends all available moves for that agent at the current state of the dialogue to the agent interface which selects a move and sends it back accordingly.

The agent protocol management API component first parses the dialogue protocols from a given DGDL file before the agent can join or create a dialogue. It is responsible for processing the dialogue alongside the agent's protocol to determine available options for that agent and maintaining the agent's commitment store. Throughout a dialogue it also communicates with DaaSdb to receive updates on the state of the dialogue and send any updates for that particular agent. An agent cannot join as a player or make a move in a dialogue without first checking the current state of the dialogue. This ensures that the dialogue is kept consistent and an agent is still following its protocol at all times.

4. Licit Protocol Switching Example

4.1. Background

This licit protocol switching example is based upon McBurney et al's [4] purchase transaction example and includes two licit protocol switches occurring sequentially between two agents, a buyer and a seller:

(1) "... a request from a potential buyer for information from a seller, proceed to a persuasion dialogue, where the seller seeks to persuade the potential buyer of the importance of some feature of the product, and then transition to a negotiation, where each party offers to give up something he or she desires in return for something else."

4.2. Protocols in DGDL

The buyer and the seller both use the same protocols, for information seeking, persuasion and negotiation dialogues, within this dialogue fragment. A simplified and shortened version of the protocols used have been created in DGDL as shown within "Example1". Each protocol contains interactions for switching to either of the other two protocols, a "SwitchAgreed" interaction, and a "SwitchRejected" interaction. The interactions with the same IDs are exactly the same within each of the protocols. The "SwitchAgreed" and "SwitchRejected" interactions can only follow one of the "SwitchTo..." interactions, ensuring that all protocol switches remain licit as the agents must first agree to a switch before it can occur.

```
Example1{
    InformationSeeking{
        participants(min:2, max:undefined)
        player(id:Seller, min:1, max:1)
        player(id:Buyer, min:1)
        ...
        interaction(id:SwitchToPersuasion){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
```

```
        interaction(id:SwitchToNegotiation){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
        interaction(id:SwitchAgreed){
            ...
        }
        interaction(id:SwitchRejected){
            ...
        }
    },
    Persuasion{
        participants(min:2, max:undefined)
        player(id:Seller, min:1, max:1)
        player(id:Buyer, min:1)
        ...
        interaction(id:SwitchToInformationSeeking){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
        interaction(id:SwitchToNegotiation){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
        interaction(id:SwitchAgreed){
            ...
        }
        interaction(id:SwitchRejected){
            ...
        }
    },
    Negotiation{
        participants(min:2, max:undefined)
        player(id:Seller, min:1, max:1)
        player(id:Buyer, min:1)
        ...
        interaction(id:SwitchToPersuasion){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
        interaction(id:SwitchToInformationSeeking){
            move(add, next, SwitchAgreed, Listener)
            move(add, next, SwitchRejected, Listener)
        }
        interaction(id:SwitchAgreed){
            ...
        }
        interaction(id:SwitchRejected){
            ...
        }
    }
```

4.3. Dialogue Fragment

Each of the moves within the dialogue fragment are shown in Table 1. The first four moves show the buyer and seller engaging in an information seeking dialogue, with the buyer asking questions about the product and the seller answering them in response. At

Table 1. Dialogue Fragment Showing a Licit Protocol Switch

Move	Player	Move ID	Protocol	Utterance
1	Buyer	Question	InformationSeeking	Is the bicycle still available to buy?
2	Seller	Response	InformationSeeking	Yes, it's still available.
3	Buyer	Question	InformationSeeking	Is the bicycle gray or black?
4	Seller	Response	InformationSeeking	The bicycle is dark gray.
5	Seller	SwitchToPersuasion	InformationSeeking	Let me tell you more about it.
6	Buyer	SwitchAccepted	InformationSeeking	Yes, I'm interested in it.
7	Seller	Request	Persuasion	The bicycle is also very lightweight.
8	Buyer	Accept	Persuasion	Amazing, that's something I'm looking for in a bicycle.
9	Buyer	SwitchToNegotiation	Persuasion	I'd like to buy it now.
10	Seller	SwitchAccepted	Persuasion	Great!
11	Buyer	Offer	Negotiation	I'll buy the bicycle for x amount.
12	Seller	CounterOffer	Negotiation	I was hoping to sell it for y amount. I can also include a tire pump with it.
13	Buyer	Accept	Negotiation	Yes, I'll buy both for y amount.

move five the seller then proposes the switch to their persuasion protocol and at move six the buyer agrees. It is this request and agreement before switching protocols that makes the protocol switch licit. Similarly, moves nine and ten show the buyer proposing a switch to their negotiation protocol and the seller agreeing before the second licit protocol switch occurs.

5. Illicit Protocol Switching Example

5.1. Background

A protocol switch is illicit when other participants in the dialogue are unaware of or have not agreed to the switch. This illicit protocol switching example is based upon Walton and Krabbe's case 3.3 example from [3]. We have changed their example from an expert consultation switched to a negotiation dialogue to instead illustrate a persuasion switched to a negotiation dialogue by replacing the term "recommends" to instead use "tries to convince". Therefore, the interlocutor's illicit switch of the protocol illustrates the fallacy of bargaining by substituting an offer for an argument. The example proceeds as follows:

(2) "A doctor tries to convince her patient to quit both smoking and drinking, giving medical reasons for the recommendation. Patient: O.K. I'll quit smoking, as long as you allow a glass of wine once in a while."

In this illicit example, in contrast to the previous licit example, it is only the patient who has and switches to another protocol. The doctor only uses a persuasion protocol throughout and remains unaware of the patient's protocol switch. In this example this leads to the breakdown of the dialogue with the doctor choosing to withdraw. Illicit

protocol switches do not always lead to a breakdown of a dialogue, though they can make it more likely, as it is entirely dependant on the participants' protocols.

5.2. Protocols in DGDL

A simplified version of the protocols used by the doctor and patient within this dialogue fragment have been created in DGDL. The protocols are for persuasion and negotiation dialogues. Wells and Reed in [7] take a similar example from Walton and Krabbe to describe how their formal dialectic systems could be used to perform a licit switch from a persuasion to a negotiation dialogue, avoiding the fallacy of bargaining. They describe a persuasion protocol, PP_0, and a negotiation protocol, NP_0, which can be used to create a sub-dialogue to reach a practical settlement between the agents when they are unable to persuade one another. This example instead facilitates and models the original illicit switch in the dialogue fragment but uses similar moves to PP_0 within the persuasion protocols and similar moves to NP_0 within the negotiation protocol.

```
ExampleDoctor{
    Persuasion{
        participants(min:2, max:2)
        player(id:Patient, min:1, max:1)
        player(id:Doctor, min:1, max:1)
        store(id:CSP, owner:Patient, structure:set, visibility:public)
        store(id:CSD, owner:Doctor, structure:set, visibility:public)
        ...
        interaction(id:Request, content:{p}, opener:"$p?"){
            store(add, {p}, CSSpeaker)
            move(add, next, Accept, Listener)
            move(add, next, Reject, Listener)
            move(add, next, Challenge, Listener)
        }
        interaction(id:Accept, content:{p}, opener:"OK $p"){
            store(add, {p}, CSSpeaker)
            store(remove, {¬p}, CSSpeaker)
        }
        interaction(id:Reject, content:{¬p}, opener:"Not $p"){
            store(remove, {p}, CSSpeaker)
            store(add, {¬p}, CSSpeaker)
            move(add, next, Withdraw, Listener)
            move(add, next, Challenge, Listener)
        }
        interaction(id:Challenge, content:{p}, opener:"Why $p?"){
            move(add, next, Defense, Listener)
            move(add, next, Reject, Listener)
            move(add, next, Withdraw, Listener)
        }
        interaction(id:Defence, content:{q->p}, opener:"$p because $q"){
            store(add, {p}, CSSpeaker)
            store(add, {p'}, CSSpeaker)
            store(add, {p'->p}, CSSpeaker)
            move(add, next, Accept, Listener)
            move(add, next, Reject, Listener)
            move(add, next, Challenge, Listener)
        }
        interaction(id:Withdraw){...}
    }
}
```

The doctor only uses the one persuasion dialogue protocol as shown within "ExampleDoctor". The protocol states the number of participants and the players which are limited to one doctor and one patient. Each participant has their own commitment store which will be managed by the agents. The use of "..." within the DGDL shows where the DGDL has been simplified, omitting detail that is not required for the current discussion. This example focuses upon the participants and the moves they can make so only these sections have been included. The examples are valid in the new version of DGDL, however, they would need to include the removed sections to be used in DaaS.

The patient uses two protocols within the dialogue fragment as shown within "ExamplePatient", first they use a persuasion protocol called "Persuasion" and later they use a negotiation protocol called "Negotiation". The persuasion protocol has been shortened as they are using the same protocol as the doctor, except for the modification of the "Accept" interaction and the addition of the "SwitchToNegotiation" interaction. The "Accept" interaction has been modified to allow both the current speaker and listener the option of switching to a negotiation protocol as their next move, after one of them has been persuaded to accept a request or defense. The "SwitchToNegotiation" interaction allows the patient to switch their protocol to a negotiation protocol instead. They also have an interaction within the negotiation protocol, called "SwitchToPersuasion", which enables them to switch back to their persuasion protocol after their offer has been rejected.

```
ExamplePatient{
    Persuasion{
        ...
        interaction(id:Accept, content:{p}, opener:``OK $p''){
            store(add, {p}, CSSpeaker)
            store(remove, {¬p}, CSSpeaker)
            move(add, next, SwitchToNegotiation, Listener)
            move(add, next, SwitchToNegotiation, Speaker)
        }
        interaction(id:SwitchToNegotiation){...}
    },
    Negotiation{
        participants(min:2, max:2)
        player(id:Patient, min:1, max:1)
        player(id:Doctor, min:1, max:1)
        store(id:CSP, owner:Patient, structure:set, visibility:public)
        store(id:CSD, owner:Doctor, structure:set, visibility:public)
        ...
        interaction(id:Offer, content:{p}, opener:"$p?"){
            store(add, {p}, CSSpeaker)
            move(add, next, Accept, Listener)
            move(add, next, Reject, Listener)
            move(add, next, Offer, Listener)
        }
        interaction(id:Accept, content:{p}, opener:"OK $p"){
            store(add, {p}, CSSpeaker)
        }
        interaction(id:Reject, content:{¬p}, opener:"Not $p"){
            move(add, next, Offer, Listener)
            move(add, next, SwitchToPersuasion, Listener)
        }
        interaction(id:Withdraw){...}
        interaction(id:SwitchToPersuasion){...}
    }
}
```

The patient's negotiation protocol specifies the same number, type of player and commitment stores as in the persuasion protocols which facilitates the switch between the protocols. In contrast, all of the interactions are completely different apart from the "Withdraw" interaction. This interaction is only used for the participant to leave the dialogue and is the same in all of the protocols.

5.3. Dialogue Fragment

Each of the moves within the dialogue fragment are shown in Table 2. It is at move five that the patient switches to a negotiation dialogue using the move "SwitchToNegotiation". This is a valid move for the patient as it follows their protocol by only occurring after they have been convinced by the doctor's defence. However this switch is illicit as the doctor only has knowledge of the persuasion protocol which does not include this move. Therefore the doctor is unable to interpret the "SwitchToNegotiation" move and misunderstands the rest of the moves the patient makes in this dialogue. After switching to the negotiation protocol, the patient, at move six, makes an offer to the doctor to quit smoking in exchange for being able to drink a glass of wine once a week. The doctor, working from their persuasion protocol, misunderstands the patient as instead making a request since this is the closest move they have within their protocol. This is also where the fallacy of bargaining occurs as the doctor is expecting an argument from the patient and instead receives this offer.

Table 2. Dialogue Fragment Showing an Illicit Protocol Switch

Move	**Player**	**Move ID**	**Protocol**	**Summary**
1	Doctor	Request	Persuasion	The doctor requests that the patient quits smoking and drinking.
2	Patient	Challenge	Persuasion	The patient challenges why they need to quit both smoking and drinking.
3	Doctor	Defence	Persuasion	The doctor explains the health benefits for quitting smoking and drinking.
4	Patient	Accept	Persuasion	The patient accepts that quitting smoking and drinking would improve their health.
5	Patient	SwitchToNegotiation	Persuasion	The patient switches to their negotiation protocol.
6	Patient	Offer	Negotiation	The patient makes an offer to quit smoking if the doctor allows them to have a glass of wine once a week.
7	Doctor	Reject	Persuasion	The doctor rejects this as they are not convinced that this would be healthy.
8	Patient	Offer	Negotiation	The patient makes another offer to quit smoking if the doctor allows them to have a glass of wine once in a while.
9	Doctor	Withdraw	Persuasion	The doctor withdraws from the dialogue as the patient is no longer following a protocol they understand.

Within DaaS when agents are using different protocols to communicate within the one dialogue and they do not have the same move within their protocol, they will try

to determine the closest match to a move they do have and use it instead. The closest match is determined based upon the AIF and its underlying argument structure. In this case, as can be seen in the moves' openers, the doctor determines that the patient's move six matches both a "Request" and "Defence" move. However as the doctor has not made a challenge to the patient for them to respond to, the doctor interprets the move as a request and rejects it. The patient then responds, at move eight, with a new offer of further reducing the amount they will drink. This move is available because they have interpreted the doctor's move seven as a rejection of their first offer and according to their negotiation protocol the patient's available moves are to either make another offer or switch back to the persuasion protocol. From the doctor's perspective, move eight is invalid and breaks the rules of their protocol. Due to the doctor's confusion, they do not respond to the patient and instead withdraw from the dialogue at this point as move nine.

6. Future Work

The next step for this work is to conduct more real world dialogues focusing upon the illicit protocol switches within them and gathering the data generated by them through DaaS. This data will then be used to investigate the relationship between illicit protocol switches and fallacies.

A subsequent focus is on protocol rule breaking and how protocol switching can impact agents breaking the protocols' rules. One agent's protocol may allow a move as valid but another agent using a different protocol may interpret that move as breaking the rules of the protocol it is following. Similarly, agents that are using different protocols to communicate within the one dialogue may not have the same moves available to them. Currently when this occurs within DaaS agents will try to find the closest move within their own protocol and respond as if to it instead. Future research will further explore these misunderstandings between agents and how they can be resolved, or how the dialogue can be repaired, after they have occurred.

7. Conclusion

Both licit and illicit protocol switching occurs often in natural language conversation. Even in very formalised settings these switches can be beneficial and improve the flexibility of dialogues. Protocol switches can be of different "flavours", covering both types and combinations of dialogues. While further research is needed to investigate specific protocol switches, particularly of the illicit kind and in relation to fallacies, this work provides a flexible starting point for how multiple types of protocol switching can be handled in computational dialogue systems. The DaaS platform can be used to create multi-agent dialogue systems that facilitate and manage both licit and illicit protocol switching within argumentative dialogue. DaaS handles the challenges raised by agents switching between protocols either in unison, or, in the more complex situation where one agent may have moved to a different protocol without the knowledge of the other participants. This has been illustrated through the use of two examples from the literature in Sections 4 and 5, which further highlight how a DaaS agent can respond to unknown move types through matching to their closest equivalent in the protocol they are working with.

April 2024

Acknowledgements

This work has been supported by the 'AI for Citizen Intelligence Coaching against Disinformation (TITAN)' project, funded by the EU Horizon 2020 research and innovation programme under grant agreement 101070658, and by UK Research and innovation under the UK governments Horizon funding guarantee grant numbers 10040483 and 10055990.

References

[1] McBurney P, Parsons S, et al. Argument Schemes and Dialogue Protocols: Doug Walton's legacy in artificial intelligence. Journal of Applied Logics. 2021;8(1):263-86.

[2] Black E, Maudet N, Parsons S. Argumentation-based dialogue. Handbook of Formal Argumentation, Volume 2. 2021.

[3] Walton D, Krabbe EC. Commitment in dialogue: Basic concepts of interpersonal reasoning. SUNY press; 1995.

[4] McBurney P, Parsons S. Games that agents play: A formal framework for dialogues between autonomous agents. Journal of logic, language and information. 2002;11:315-34.

[5] Sklar EI, Azhar MQ. Argumentation-based dialogue games for shared control in human-robot systems. Journal of Human-Robot Interaction. 2015;4(3):120-48.

[6] Engelmann DC, Panisson AR, Vieira R, Hübner JF, Mascardi V, Bordini RH. MAIDS - A Framework for the Development of Multi-Agent Intentional Dialogue Systems. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. AAMAS '23. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems; 2023. p. 1209âĂŞ1217.

[7] Wells S, Reed C. Knowing when to bargain. Frontiers in Artificial Intelligence and Applications. 2006;144:235.

[8] Reed C. Dialogue frames in agent communication. In: Proceedings International Conference on Multi Agent Systems (Cat. No. 98EX160). IEEE; 1998. p. 246-53.

[9] Black E, Hunter A. An inquiry dialogue system. Autonomous Agents and Multi-Agent Systems. 2009;19:173-209.

[10] McBurney P, Hitchcock D, Parsons S. The eightfold way of deliberation dialogue. International Journal of Intelligent Systems. 2007;22(1):95-132.

[11] Rahwan I, Ramchurn SD, Jennings NR, McBurney P, Parsons S, Sonenberg L. Argumentation-based negotiation. The Knowledge Engineering Review. 2003;18(4):343-75.

[12] Prakken H. Formal systems for persuasion dialogue. The knowledge engineering review. 2006;21(2):163-88.

[13] Fan X, Toni F. Agent Strategies for ABA-based Information-seeking and Inquiry Dialogues. In: ECAI; 2012. p. 324-9.

[14] Bex F, Lawrence J, Reed C. Generalising argument dialogue with the Dialogue Game Execution Platform. In: COMMA; 2014. p. 141-52.

[15] Wells S, Reed CA. A domain specific language for describing diverse systems of dialogue. Journal of Applied Logic. 2012;10(4):309-29.

[16] Chesnevar C, Modgil S, Rahwan I, Reed C, Simari G, South M, et al. Towards an argument interchange format. The knowledge engineering review. 2006;21(4):293-316.

[17] Snaith M, Lawrence J, Reed C. Mixed initiative argument in public deliberation. Online Deliberation. 2010;2.

[18] Bex F, Lawrence J, Snaith M, Reed C. Implementing the Argument Web. Communications of the ACM. 2013 Oct;56(10):66-73.

[19] Lawrence J, Bex F, Reed C, Snaith M. AIFdb: Infrastructure for the argument web. In: Computational Models of Argument. IOS Press; 2012. p. 515-6.