Dialogue Templates for Automatic Argument Processing

Floris BEX and Chris REED
Argumentation Research Group, School of Computing, University of Dundee

Abstract There is an extensive literature on dialogue systems, which attempts to capture aspects of structured communication with the aim of understanding, improving, and automatically recreating such communication. This paper discusses *dialogue templates*: blueprints that can be instantiated and combined to form argumentative dialogues. These templates provide a generic way of representing individual dialogue systems and allow us to generalise techniques for investigation, generation and recognition of dialogues.

1. Introduction

Dialogue systems attempt to capture aspects of structured communication with the aim of understanding, improving, and automatically recreating such communication. Much of real-world argument proceeds through discussions and debate, and the dialectical process of argument and counterargument by its very nature (implicitly) assumes some sort of dialogical context. Of particular importance in argumentative dialogue is the way in which the participants respond to each other by, for example, uttering challenges or agreements. The fact that one utterance replies to another in a relevant way is what gives dialogues their logical structure and coherence.

Formal dialogue systems originate in philosophy [6][8][25] and have been adopted in computational argumentation and multi-agent systems (see e.g. [11][17]). In these systems, utterances are moves in a game, the rules of which are set out in a *dialogue protocol* which, among other things, regulates turn taking and ensures that each move is relevant. Dialogue games thus provide a normative framework for players to, for example, claim, question or argue for an opinion in a regulated way.

The theoretical properties of specific dialogue protocols are typically explored in the papers which introduce those protocols. Furthermore, software for the automatic execution of dialogue protocols has been developed (e.g. in multi-agent systems [27]). In each case though, the properties are explored from the point of view of a single protocol or implementation is tailored towards a single protocol. In multi-agent systems steps have been taken towards harmonisation and standardisation of communication protocols (e.g. FIPA ACL [5]). However, these are specifically aimed at multi-agent communication and do not capture exactly and only the complexity and expressivity of argumentative dialogue. Furthermore, whilst overarching frameworks for representing combinations of different types of dialogue have been proposed (e.g. [9][10][21]), these frameworks fail to show how arbitrary dialogue protocols can be instantiated (i.e. executed) as specific dialogues, and furthermore, they fail to show how logical structures (viz., relations between propositions such as inferences and conflicts) can be constructed and navigated by those specific dialogues in the style of [16] [23].

What is needed is a generic framework for expressing dialogues and dialogue protocols. This framework should move away from the case-by-case approach and aim to reconcile insights from computational (multi-agent) argumentation with those from philosophy and informal argumentation theory. The framework should provide a means of expressing not just the syntax of arbitrary dialogues in terms of reply structures, but it should also define update semantics for dialogues that determine, for example, what it means for a question to be replied to with an agreement. The framework should be specific enough to facilitate software implementation of dialogue games and the study of the theoretical properties of these games. However, the framework must also allow for the analysis and modelling of real argument and debate.

In order to capture all the above requirements, we propose *dialogue templates*, schemas that encode the generic structure of utterances and replies in dialogue. They are the basic building blocks of dialogue protocols and can be mixed and matched to form different types of protocols. Dialogues can then be formed by instantiating and combining multiple templates, thus building an explicit reply structure. In this way we can represent the syntax and semantics of arbitrary dialogue in a common format. As a case in point, we present [25]'s CB game as a set of dialogue templates. Furthermore, dialogue templates allow (with additional future software) the execution of arbitrary protocols. Thus, argument resources can be constructed and navigated, and properties of protocols (termination, state reachability, etc.) can be investigated and compared.

The rest of this paper is structured as follows. In section 2 we briefly introduce dialogue games, discuss the common features of dialogue protocols and give a simple example of a protocol. Section 3 specifies of an automatic dialogue game execution platform. Section 4 contains the meat of the paper: it provides a general framework for expressing dialogue and defines the dialogue templates. Section 5 then briefly compares the current research with related research, and section 6 concludes the paper.

2. Dialogue Games for Argumentation

Argumentative dialogues consist of a series of locutions (utterances) made by the participants. As a simple example of a dialogue, take the following exchange between Bob and Alice on the UK's Trident nuclear missile programme:

(1) Alice: *Should Britain stop the Trident Programme?*
(2) Bob: *Yes, it should.*
(3) Alice: *Why?*
(4) Bob: *It is expensive!*
(5) Bob: *And also, Britain needs to set an example for other countries to follow.*

In this example dialogue, multiple participants make claims and pose questions, and there is an amount of coherence to the way the participants react to one another. That is, there is an (implicit) *reply structure* in the dialogue [16] that contains the connections between the locutions in a dialogue, the 'glue' [1] that keeps the locutions together and makes a dialogue coherent. This is analogous to non-dialogical argument, where logical (inference) connections form the glue between the individual propositions.

The connections between locutions represent functional transition relations rather than, for example, temporal sequence. Our example demonstrates that: despite being consecutive, there is little connection between (4) and (5). Equally, the relationships between Bob agreeing that 'Britain should stop Trident' (2) and Bob giving a reason

for why Britain should (4) is very much the sort of relationship we want to capture, even though the distance between them could be arbitrarily great.

During dialogue, the participants (implicitly) construct and navigate an underlying argument structure [16][23], a static rendition of the claims, proposals or arguments made. For example, in the above dialogue one of the arguments made is '[Trident] is expensive *therefore* Britain should stop Trident'. Here, we distinguish between argument_1 and argument_2 [11]. Argument_1 (below referred to simply as 'argument') refers to an argument as a static structure of premises that are reasons for or against conclusions (i.e. 'he prepared an argument'). On the other hand, argument_2 (further referred to as 'dialogue') refers to a debate or discussion (i.e. 'they had an argument').

In order to understand the link between dialogue and argument, we need to consider the idea of a speech act [24]. A speech act can be analysed as a locution (the actual utterance, e.g. 'Yes, it should'), but also as an illocutionary act which consists of the illocutionary force (the intention of uttering a locution: one may say *p* with an intention of asserting *p*, asking *p*, challenging *p*, promising *p* and so on) and the propositional content, the proposition(s) the act refers to. In our example, speech acts (1)-(3) have the same propositional content – 'Britain should stop Trident' – but differing illocutionary force – questioning, asserting and challenging, respectively.

2.1. Dialogue protocols

The exact principles that make a dialogue coherent have been formulated and studied in the literature on formal dialogue systems. At the heart of these systems are the dialogue protocols describe a dialogue game's permitted locutions, how and when the dialogue starts and ends, how locutions commit the players to certain claims and, perhaps most importantly, how locutions may be combined into exchanges.

As an example of a dialogue system, consider Walton's CB game for two player persuasion, which has four different types of locutions. *Statements* 'State *S*', which can be used to assert claims; *Withdrawals* 'No commitment S', which allow players to withdraw their commitment to a statement; *Questions* '*S*?', which ask whether the other player thinks *S* is true or not; and *Challenges* 'Why *S*?', which request a reason for *S*. These locutions are fairly standard in protocols for persuasion dialogues; [17] further lists a 'concede *S*' locution that can be used to admit to some statement *S*, and an 'argue *S so T*' locution that is used to provide an argument.

The dialogue rules of CB determine that certain moves can only be followed by other moves, thus enforcing a logical reply structure. These rules are: players take turns and advance one locution at a time, with the exception of the combination 'No commitment *S*, Why *S*?', which may be uttered by a player in one turn; a question '*S*?' must be followed by either 'State *S*', 'State *Not-S*', or 'No Commitment *S*'; and 'Why S?' must be followed by either 'No commitment *S*' or 'State *T*', with *S* a consequence of *T*. Contrary to some other dialogue systems (e.g. [16]), CB does not have a full explicit reply structure; apart from the above exceptions there are no rules that govern, for example, which locution must be used to reply to a statement or a withdrawal. This makes CB flexible at the cost of coherence: in theory, both players could alternate making seemingly unconnected statements until the dialogue terminates.

In a dialogue, commitments can be used to, for example, ensure a player does not contradict himself (dialogical consistency) or ensure that a player is prepared to defend his claims (dialectical obligations). CB lists five commitment rules: a 'State *S*' adds *S* to the speakers commitment store; 'No commitment *S*' deletes *S* from the speaker's

commitment store; 'Why *S*?' places *S* in the hearer's commitment store; a statement that is shown by the speaker to be an immediate consequence of statements that are commitments of the hearer is automatically added to the hearer's commitment store; and a commitment that is shown by the speaker to be an immediate consequence of the hearer's current commitments may not be withdrawn by the hearer.

CB has only one termination rule: the players agree in advance that the game will terminate after a set number of moves. Examples of other termination rules are when both players agree to end the dialogue [3], when a player accepts the other player's main claim [17] or when a player runs out of possible moves to make [16] (note that the latter is only possible in games with an explicit reply structure) .

An important issue with dialogue protocols is the way in which they handle the connection to the underlying argument structure of a dialogue. Many dialogue protocols explicitly refer to argument structures or to the logical rules and mechanisms that pertain specifically to these structures. CB, for example, refers to a notion of consequence, which depends on the notion of inference in non-dialogical argument: *T* is a consequence of *S* if *T* can be inferred from *S*. Other examples are [16], where the moving of counterarguments is allowed and the notion of counterargument is defined in some underlying argumentation system (e.g. [18]), and [3] who define the outcome of a dialogue as a structure of arguments and counterarguments.

3. Automatically Processing Dialogues and Protocols

A large set of dialogue protocols is available from the literature and some work has explored how commonalities across that set might yield representational and computational benefits [9][10][21]. Given this work, it is natural to consider building generalisations not just in theory but also in practice. That is to say, with a variety of protocols with common and distinguishing features, it should be possible first, to represent their features in a common, executable language, and then second, to construct a general execution environment which can support the creation of instances of dialogues according to their governing protocols.

If these dialogue rules can be expressed in a generalised language, it should be possible for a general purpose execution engine to, for example, respond to a request to determine what legal moves are available next given a game protocol and a description of the last move. For example, in CB the *challenge* move (3) in our example dialogue (section 2) must be followed by a s*tatement* or *withdrawal* move; whereas in DC [8], the same move can also be followed by a demand for resolution. If the game is Markovian, the request to determine the next legal move can be stateless, and can sit behind a RESTful web service of the sort that has proven so useful in constructing modular online systems. It could thence form a part of agent communication processing in a multi-agent system, or control logic for dialogic support tools such as Magtalo [22] or Arvina [12]. The Dialogue Game Execution Platform (DGEP) in Figure 1 offers a means for Arvina, or a multi-agent system, or any other implementation which requires dialogue execution to make use of *any* available protocol. Figure 1 also notes that if we want to execute arbitrary protocols we will inevitably need some tool for rapidly constructing them using some toolkit.

Arvina is a dialogical support system that allows for the structured execution of a reasoning process by implementing dialogue protocols and then allowing users to play the dialogue game against virtual agents and against each other in an instant-messaging

environment. Arvina is currently limited to a single protocol and directly interfaces with the Argument Web [20], an argumentation corpus that at time of writing contains a growing set of around 2,000 non-dialogical and dialogical arguments built using tools such as Araucaria and Rationale [13], OVA and Arvina [12] and argublogging [26]. Ultimately, the aim is to have DGEP as an intermediary between Arvina (and other dialogical tools) and the Argument Web, so that this large corpus can be extended, navigated and used in a regulated way by executing any of the available protocols.

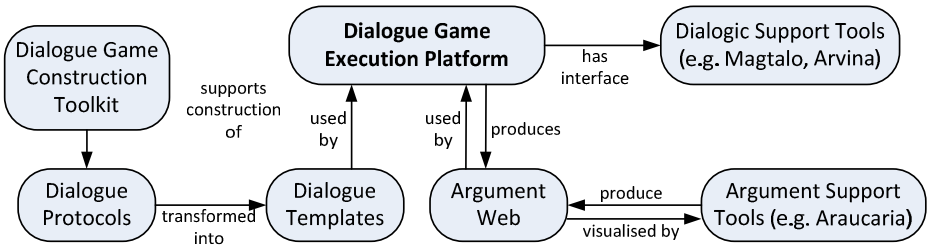

Figure 1: Specification for the automatic processing of dialogues

The question then turns to exactly how such protocols should be represented for DGEP. The remainder of this paper shows how the right level of abstraction in capturing the necessary information can be captured using *dialogue templates*, schematic representations of a single move and its reply, the illocutionary force of both moves and the underlying argument structure. Note that dialogue templates may (perhaps) not provide the right level of abstraction in presenting these rules in human-readable form. We can use a loose analogy here with a high level programming language like Java. The right level of abstraction for human software engineers is the Java language itself. The right level of abstraction for execution is bytecode – which demands a bytecode interpreter, the JVM. Similarly, the dialogue game construction toolkit may offers a high level expression language whilst the templates that are a result of those high level specifications can be executed directly by the DGEP.

Finally, with a language for representing dialogue protocols, plus a language for representing specific executions (i.e. specific dialogues), we can explore two new classes of question: first, does a specific extant dialogue conform to a specific protocol; and secondly, what is the set of protocols to which a given extant dialogue (or, conceivably, a set of dialogues) conform?

4. A generic framework for representing argument and dialogue

If dialogue and argument structures are to be used by software platforms like DGEP, they should be expressed in a language that is sufficiently precise and formally grounded. However, the framework should also be natural enough to facilitate large-scale analyses of dialogue. It is for these reasons that we render dialogue and argument structures as graphs in the language of the Argument Interchange Format (AIF) [19][23]. In these graphs, the individual elements of arguments and dialogue are represented as a set of linked data, typed nodes containing images, text, formulas and so on. This representation facilitates analysis of real-world argument and dialogue, as is evident from the work on sense-making in argumentation [13]. At the same time, the language of the AIF is compatible with the state-of-the-art computational frameworks for structured argument construction and evaluation [2]. Furthermore, the language is

expressive and precise enough to capture the main components of argumentation (inference, conflict, preference) and dialogue (locution, transition, illocutionary force). More importantly, it explicitly captures the links between dialogue and argumentation which are needed to have DGEP and the tools that depend on DGEP interface with the Argument Web, which depends on the non-dialogical part of the AIF [19].

4.1. Transitions in dialogue

Complex inference structures in non-dialogical argument are often explicitly rendered in an inference graph [15], in which proposition-nodes are linked with inference and conflict relations. The reply structure of dialogues can be explicitly represented in the same way, with locution-nodes being linked by transitional reply relations (Figure 2). In this figure, locutions (boxes) are related to each other via transitional relations (circles) denoting transitions from one locution to the next. Locutions have two attributes: speaker, the person who uttered the locution, and time, the time at which the locution was uttered (which is left implicit in Figure 2).

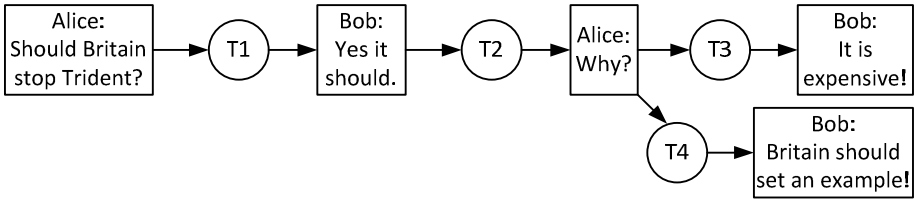

Figure 2: An example of locutions and transitions in argumentative dialogue

4.2. Illocutionary force in dialogue

With dialogue represented as dialogue-graphs, and argument as argument/inference-graphs, it makes sense to render the connection between the two also as relations in a graph. Figure 3 characterises the connections between the dialogue from Figure 2 and the underlying argument structure according to the illocutionary force of the locutions.

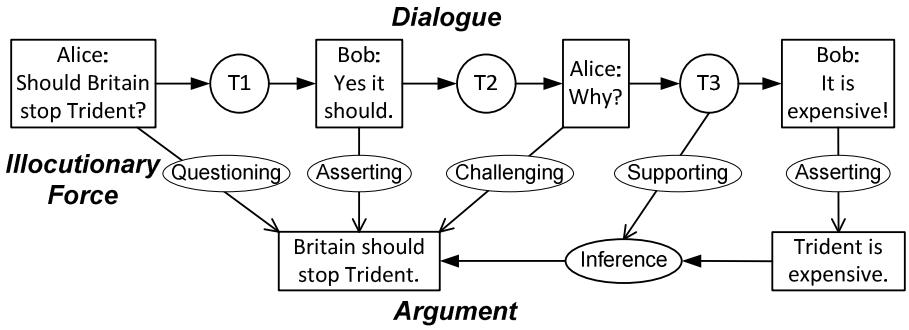

Figure 3: Illocutionary force as the link between dialogue and argument

Note that the argument in Figure 3 follows the same graphing conventions as the dialogue graph. Propositions are diagrammed as boxes (locutions are also propositions) and relations (transitional, inferential, illocutionary) are diagrammed as ellipses.

The first three locutions Question, Assert and Challenge whether Britain should stop Trident, and Bob's last move asserts that Trident is expensive. The illocutionary relation of 'supporting' between T3 and the Inference in the argument is more complex. Alice's question (connected to its propositional content) and Bob's assertion (also connected to its propositional content) can be imagined in isolation, even as occurring in different dialogues. And if they were then, ceteris paribus, there would be no link between 'Trident is expensive' and 'Britain should stop Trident'. It is only in virtue of the fact that Bob's assertion of Trident's high cost is responding to Alice's challenge of 'Britain should stop Trident' that there is an inferential link here [23]. Hence, the link between the transitional relationship that captures the notion of responding (T3) and the inferential relationship can thus be characterised as the illocutionary force of an implicit speech act 'support' or 'argue'. Recall that some dialogue systems (e.g. [3][16]) contain explicit speech acts for posing an argument 'Trident is expensive *so* Britain should stop Trident'. This locution will have an illocutionary link to both 'Trident is expensive' (asserting) and the inference (supporting).

4.3. Calculated properties

Given a graph like the one in Figure 3 there are many properties of the argument and the dialogue that can be calculated: the number of premises that support a conclusion; the number of locutions performed at a specific time; whether or not one set of nodes successfully defeat some other set of nodes according to some formal definition of acceptability; whether or not one set of nodes is a consequence of another set according to some logical interpretation; and so on.

These properties are calculated using processes (counting, searching, comparing) that in formal systems would typically be defined at the meta-level. They are often non-monotonic and in many cases dependent on, for example, the exact formal system they represent. The argument-graphs used in the current model are monotonic in that though they support dialogical update, those updates are guaranteed not to remove material from the underlying argumentation structures. For although an individual arguer may change their position over the course of time or during the run of a dialogue, other arguers may still wish to use or extend the original arguments. Consequently, these calculation processes cannot and should not be captured in the graphs.

The calculated properties themselves, however, can be represented in the graphs of the object language: they are just propositions and they may be used in argument and dialogue explicitly. So, for example, one may encounter a node in a graph that says that 'Britain should stop Trident *is a consequence of* Trident is expensive', or a node that says that '*Bob is committed to* Britain should stop Trident' and so on.

4.4. Dialogue Templates

Dialogue templates are schematic representations of a single transition in a dialogue, including the 'start' and 'end' locutions of the transition, the illocutionary force of both locutions and the argument structure that the locutions refer to. In other words, templates are *transition schemes* which can be instantiated to form transitions (i.e. a step in a dialogue), and these transitions can then be chained to form a dialogue. Note that the ontological machinery at work here is (intentionally) very similar to that of *argumentation schemes*, schematic representations of inference that can be instantiated to form inferences, which can be chained to form arguments.

Dialogue templates can be used to represent dialogue protocols. Templates provide an explicit reply structure for a dialogue, that is, given the protocol they define for each type of locution the possible replies (transitions to other locutions). For protocols like CB, where there are few restrictions on the exact reply structure, there will a relatively high number of dialogue templates, as every possible transition needs to be covered. In contrast, dialogues with an explicit reply structure like [16] or dialogues with a few locutions like [3] can be represented by a considerably smaller number of templates.

If we look at Figure 3, we can see that a single move in a dialogue consists of a locution (*Alice: Should Britain stop Trident?*), propositional content (*Britain should stop Trident*) and a relation of illocutionary force connecting the two (*Questioning*). Thus, our formal framework is the only model that explicitly distinguishes between these elements that come directly from speech act theory [24].

Definition [Move in dialogue] A move in a dialogue $M = (L, IF, \varphi)$ consists of a locution L, illocutionary force IF and propositional content φ. The locution L has an associated player p_i and time t_i.

In our rendering of dialogues, the actual locution is not directly important – all we need to know is that there is a locution uttered by a player at some time. Hence, we will informally represent a move as, for example, p_i *asserting* φ *(at* t_i*)*. The moves in CB (see section 2.1) are p_i *asserting* φ ('State *S*'), p_i *withdrawing* φ ('No commitment φ'), p_i *questioning* φ ('φ?') and p_i *challenging* φ ('Why φ').

Definition [Dialogue Template] A dialogue template (*Start, End, Pres, IF*) consists of one or more start moves *Start*, an end move *End*, a set of presumptions *Pres* and (optionally) the illocutionary force *IF* of the transition.

Table 1 shows all the dialogue templates for CB. The replies captured by templates 8 – 14 are explicitly mentioned in the structural rules of CB. Templates 1 – 7, however, require some explanation.

Templates 1 – 3 start with an empty move ∅ and 15 ends with ∅, an artificial placeholder to signify the start and end of the dialogue, rather like the reserved states S and F often used to signify the start and end state in finite state machines. *Withdrawing* moves can only be played if the speaker is committed to the proposition he withdraws, and in order to be committed to a proposition a player must have either asserted it (templates 4,5,7) or the other player must have challenged him on it (template 12,14). An exception is the move 'No commitment φ' in reply to a 'φ*?*' question (this should probably not be interpreted as a strict withdrawal but rather as a way of saying that one has no opinion on φ). In CB, a player can always challenge, but it also makes sense to allow challenges to reply to an assertion of the other party (template 6,7). Finally, note that 'p_i *withdrawing* φ **and** p_i *challenging* φ' is considered to be one compound move.

In addition to *Start* and *End* moves, dialogue templates also have *presumptions*, conditions which have to be met before the template can be applied. In the case of CB, for example, a player can only withdraw a commitment if he has not previously done so, if it is his turn and if the dialogue has not yet terminated. Notice that these presumptions all concern calculated properties (e.g. counting the number of turns). In dialogue templates, calculated properties are implemented as presumptions on a transition (rather than as, for example, preconditions for the performance of a move). This insight is derived from the analogy between transition schemes (dialogue templates) and argumentation schemes (argument templates), the latter of which have

implicit premises (defined by the critical questions) which are included in the scheme but not by default in the argument that instantiates the scheme; it is only when they are challenged or questioned that they become an explicit part of the argument. The same thing happens for the calculated properties that are used in protocol definitions. If the fact that 'it is p_j's turn' is not disputed (by the original participants in the dialogue or by any subsequent analyst or contributor) then it remains an implicit part of the dialogue template and never becomes an explicit part of the dialogue graph. In this way, a mechanism is provided for representing calculated properties when necessary without cluttering analyses with large numbers of implicit premises.

Table 1: Dialogue Templates for CB

ID	***Start***	***End***	***Pres***	***IF***
1	∅	p_i *asserting* φ	1,2	
2	∅	p_i *questioning* φ	1,2	
3	∅	p_i *challenging* φ	1,2	
4	p_i *asserting* φ	p_i *withdrawing* φ	1,2,3,4	
5	p_i *asserting* φ	p_i *withdrawing* φ **and** p_i *challenging* φ	1,2,3,4	
6	p_i *asserting* φ	p_j *challenging* φ		
7	p_i *asserting* φ; p_j *asserting* φ	p_j *withdrawing* φ **and** p_j *challenging* φ	1,2,3,4	
8	p_i *questioning* φ	p_j *asserting* φ	1,2	
9	p_i *questioning* φ	p_j *asserting* ψ	1,2	*Contradicting* φ with ψ
10	p_i *questioning* φ	p_j *withdrawing* φ	1,2,3,4	
11	p_i *challenging* φ	p_j *asserting* ψ	1,2,5	*Supporting* φ with ψ
12	p_i *challenging* φ	p_j *withdrawing* φ	1,2,3,4	
13	p_i *withdrawing* φ **and** p_i *challenging* φ	p_j *asserting* ψ	1,2,5	*Supporting* φ with ψ
14	p_i *withdrawing* φ **and** p_i *challenging* φ	p_j *withdrawing* φ	1,2,3,4	
15	*any move*	∅	6	

Here, $p_i \neq p_j$ and $\psi \neq \varphi$. The presumptions are: (1) *it is p_j's turn;* (2) *the maximum number of turns has not been reached;* (3) *p_i has not previously withdrawn φ;* (4) *p_j is not committed to χ, where χ is an immediate consequence of φ;* (5) *φ is a consequence of ψ; and (6) the maximum number of turns has been reached.*

Consider the example in Figure 3. First template 2 is applied: '*Alice questioning Britain should stop Trident*' is the end move. Now there are three templates that can be applied, namely 8 – 10. Figure 4B shows template 8 as a graph, which expresses the idea of dialogue templates as building blocks for dialogue graphs more clearly. The elements inside the dotted lines have to be present in the Argument Web for the template to be applied; the elements outside the dotted line will be added to the Argument Web after the application of the template. Notice how the schematic graph of template 8 provides a template for the T1 relation in Figure 3. Next, Alice challenges Bob's assertion (template 6), and Bob replies to Alice's challenge and by providing a reason for his earlier assertion (template 11, Figure 4A). Finally, template 15 will be applied to terminate the dialogue.

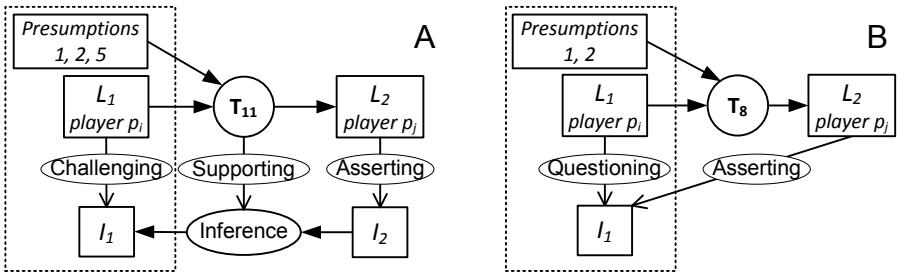

Figure 4: Dialogue Templates based on CB.

Many of the templates in Table 1 can be re-used in the modelling of other dialogue protocols. Templates 4, 6, 11 and 12 for example, are mentioned by [17] as being standard in most dialogue games for persuasion. For example, p_i *asserting* φ represents CB's *Statement* ('State φ') but also [16]'s *claim* locution. What makes these templates specific to CB are the presumptions, which refer explicitly to the way in which CB handles, for example, commitment and the underlying argument structure.

4.5. Calculating presumptions

Whilst calculated properties can be represented as presumptions, it is crucial that in a generalised execution language the processes involved in calculating these properties are not explicitly represented. Dialogue templates are intended to encode the explicit reply structure enforced by a protocol and not arbitrarily complex an arbitrarily specific calculation processes. However, for current purposes it is interesting to discuss the various types of presumptions and the processes that can possibly compute them.

In the case of dialogue protocols, any type of rule may involve calculated properties. Some commitment rules, for example, can be represented explicitly as a transition. For example, in dialogues without withdrawal, '*p is committed to* φ' can be represented by having an assert move (which commits the player) as a *Start* move of the relevant template. In dialogues with withdrawal, however, presumptions are needed (cf. template 4,5,7 in Table 1). So, as can be expected, whether or not something is a calculated property does not depend on the type of protocol rule, but rather on whether the rule involved some kind of calculation process. .

Important types of properties that need to be calculated are (i) properties that refer to the non-existence of a node or relation in the graph (presumption 3: there is no move p_i *withdrawing* φ); (ii) properties that refer to temporal sequence (presumption 1: the temporally previous move was uttered by the other player); (iii) properties that involve counting elements of the graph (presumption 2: the number of moves does not exceed the maximum turns); (iv) properties that refer to concepts which are not part of the graph (presumption 5: φ can be inferred via a finite number of inference rules). The calculation of such properties cannot be represented in a graph.

If, however, we want the Dialogue Game Execution Platform (section 3) to be able to correctly execute dialogue protocols, it should somehow be able to determine whether the presumptions hold before it applies a dialogue template. In order to do this, the DGEP must call on implementations of the various calculation processes. Some of these calculation processes can be integrated into the DGEP (e.g. in the case of calculating temporal sequences). Other processes, however, are more suited to external programmes. Whether φ is a consequence of ψ, for example, can be determined by any

appropriate theorem prover. Using such external programmes affords an interesting measure of generality for the DGEP. For example, we can execute CB with as its underlying logic propositional logic, or an argument-based logic such as ASPIC+ [2].

5. Discussion and conclusions

In this paper, we have proposed a way of representing dialogue protocols in the form of *dialogue templates*. By defining these templates and the generic dialogue language, we can represent the syntax and update semantics of any arbitrary dialogue game we care to define in a common language. This allows us to move away from the case-by-case approach common in the literature on computational argumentation and move towards a class-of-systems approach. A similar pattern has emerged in verification research where the focus of study is squarely now upon developing tools and techniques that will work for large classes of programmes, rather than exploring particular phenomena of specific programming constructs [7].

Dialogue templates lay down the basics for automatically executing dialogue protocols, as they provide the possible moves at each point in a dialogue. They can hence be used, for example, to model agents that help navigate complex argument structures or to construct all dialogues based on a set of protocol rules. When paired with a simple dialogue game construction toolkit, the templates thus allow us to incrementally construct dialogues that conform to a huge variety of protocols, compare these dialogues and thus investigate the properties of the protocols. An advantage of the current approach's reliance on AIF is that we can use the argument web, which contains a huge amount of non-dialogical argument data that can act as the basis for these automatically constructed dialogues.

The current framework is the first aimed explicitly at the representation and execution of dialogue games. Other generic protocol languages, most of them from the field of multi-agent systems, lack the specific ideas and concepts that are needed for argumentative dialogue, such as the connection to the underlying argument structure and the focus on an explicit reply structure. Work that has dealt with argumentative dialogue protocols in a principled way is by Prakken [16][17], who defines a generic, explicit reply structure for persuasion dialogue and discusses how various protocol rules, including ones about commitment, influence the flexibility of the protocols and the coherence of the resultant dialogues; the connection to an underlying (non-dialogical) argument framework is also discussed. While Prakken's work is not intended for as a basis for automated reasoning, there is no reason why some implementation of his protocols would not be suitable for this task.

One of the advantages of the current work over Prakken's work is that in the latter, the relevance of reasons has to be 'hard-coded'. [16] does not allow for illocutionary force of transitions, which means that a *Why* φ*?* – *state* ψ reply will not yield an inference; this inference (φ *since* ψ) has to be explicitly uttered. This makes Prakken's approach less suitable for capturing natural argumentation, where locutions of the form φ *since* ψ are quite rare. In linguistics, the rarity of this form has led to a specific name for the enthymematic Why-Because structure, *Modus Brevis* [4]) Furthermore, Prakken considers only persuasion; in contrast, dialogue templates are equally suited for other types of dialogue. An advantage of [16] is that it also includes ways of determining the acceptance of dialogue moves (i.e. whether the player of a move is winning). In this

respect, [16] has the same relation to dialogue graphs as ASPIC+ [18] has to non-dialogical argument graphs [2]: graphs can be used to express argument and dialogue and the frameworks of [18] and [16] can be used to determine the acceptability of these dialogues and arguments under some argumentation-theoretic semantics.

References

[1] N. Asher and A. Lascarides. *Logics of Conversation*, Cambridge University Press, 2005.

[2] F.J. Bex, S. Modgil, H.Prakken and C.Reed. On Logical Reifications of the Argument Interchange Format. *Journal of Logic and Computation*, to appear (2012).

[3] E. Black and A. Hunter. An inquiry dialogue system. *Autonomous Agents and Multi-Agent Systems* 19:2 (2009), 173-209.

[4] R. Cohen. Analyzing the Structure of Argumentative Discourse. *Computational Linguistics* 13:1 (1987) 11-24.

[5] FIPA. Communicative Act Library Specification. Standard SC00037J, IEEE Foundation for Intelligent Physical Agents, 3 December 2002.

[6] C. Hamblin. 'Mathematical models of dialogue'. Theoria 37 (1971), 130–155.

[7] R. Jhala and R. Majumdar. Software model checking, *ACM Computing Surveys* 41:4 4 (2009).

[8] J.D. Mackenzie. Question begging in non-cumulative systems, *Journal of Philosophical Logic* 8 (1979) 117–133.

[9] N. Maudet and F. Evrard. A generic framework for dialogue game implementation. *Proceedings of the 2nd Workshop on Formal Semantics and Pragmatics of Dialogue* (1998), 185—198.

[10] P. Mcburney and S. Parsons. Games that agents play: A formal framework for dialogues between autonomous agents. *Journal of Logic, Language and Information*, 11:3 (2002), 315-334.

[11] P. McBurney and S. Parsons. Dialogue games for agent argumentation. In I. Rahwan and G. Simari (eds.): *Argumentation in Artificial Intelligence*. Springer, Berlin (2009), 261-280.

[12] S. Modgil, F. Toni, F. Bex, I. Bratko, C. I. Chesñevar, W. Dvořák and M. A. Falappa, S.A. Gaggl, A.J. Garcia, M.P. Gonzalez, T.F. Gordon, J. Leite, M. Mozina, C. Reed, G.R. Simari, S. Szeider, P. Torroni, S. Woltran. The Added Value of Argumentation: Examples and Challenges. In *Handbook on Agreement Technologies (COST Action IC0801)*, to appear (2012).

[13] A. Okada, S. Buckingham Shum, T. Sherborne (Eds.) (2008). *Knowledge Cartography: Software Tools and Mapping Techniques*. Advanced Information and Knowledge Processing, London: Springer.

[14] D. O'Keefe. Two concepts of argument. *Journal of the American Forensic Association*, 13 (1977) 121–128.

[15] J.L. Pollock. Justification and defeat. *Artificial Intelligence* 67:2 (1994), 377-408.

[16] H. Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation* 15 (2005), 1009-1040.

[17] H. Prakken. Formal systems for persuasion dialogue. *The Knowledge Engineering Review*, 21:2 (2006) 163–188.

[18] H. Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1:93–124, 2010.

[19] I. Rahwan and C. Reed. The Argument Interchange Format. In G. Simari & I. Rahwan (Eds.), *Argumentation in Artificial Intelligence*, Springer (2009), 383-402.

[20] I. Rahwan, F. Zablith, and C. Reed. Laying the foundations for a world wide argument web. *Artificial Intelligence* , 171:897–921, 2007.

[21] C. Reed. Representing dialogic argumentation. *Knowledge-Based Systems*, 19:1 (2006), 22-31.

[22] C. Reed and S. Wells, "Using Dialogical Argument as an Interface to Complex Debates", in *IEEE Intelligent Systems Journal, Special Issue on Argumentation Technology* (2007).

[23] C. Reed, S. Wells, K. Budzynska and J. Devereux. Building arguments with argumentation: the role of illocutionary force in computational models of argument. *Computational Models of Argument: Proceedings of COMMA 2010*, IOS Press, Amsterdam (2010), 415 – 426.

[24] J.R. Searle. Speech Acts: An Essay in the Philosophy of Language. Cambridge University Press, 1969.

[25] D.N. Walton. *Logical Dialogue Games and Fallacies*. University Press of America, 1984.

[26] S. Wells, C. Gourlay, and C. Reed, Argument Blogging, *9th International Workshop on Computational Models of Natural Argument. IJCAI 2009.*

[27] M. Wooldridge. *Reasoning about Rational Agents*. MIT Press, 2000.