Dialogue Templates for Automatic Argument Processing

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

Abstract Dialogue systems attempt to capture 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

Much of real-world argument proceeds through discussion and the dialectical process of argument and counterargument by its very nature assumes a dialogical context. Dialogue systems attempt to capture aspects of structured communication with the aim of understanding, improving, and automatically recreating such communication. Originating in philosophy [6][10][27] and linguistics [24], dialogue systems consider utterances as moves in a game, the rules of which are set out in a *dialogue protocol*. The protocol regulates, among other things, turn taking and ensures that each move is relevant, thus allowing players to argue, inform, investigate and negotiate in a regulated way. Thus, implementations of dialogue systems can be used to support, for example, complex decision-making processes [29] and interaction with large datasets [22].

There is much work that explores specific dialogue protocols and their implementations in computational argumentation and multi-agent systems (see [13] for an overview). In many cases these explorations and implementations are from the point of view of a single protocol. Work on combinations of dialogue protocols (e.g. [11][12][21]) typically operates at a high level of abstraction, that is, it provides the formal scaffolding for joining arbitrary protocols but does not tell us exactly how specific types of dialogue locutions influence each other. Furthermore, in much of the work on computational argumentation (e.g. [15][16]) the main theoretical focus is on how protocols and the dialogues that instantiate them influence the acceptability of claims or arguments (in the sense of [4]); what this work fails to show is 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 [23].

What is needed is a generic framework for expressing dialogues, protocols and the ways in which fragments of those dialogues navigate and update reasoning structures. This framework should move away from a case-by-case approach and aim to reconcile insights from computational argumentation with those from philosophy, linguistics and informal argumentation theory. It should provide a means of expressing the syntax of arbitrary dialogues in terms of reply structures as well as define update semantics for

dialogues that determine, for example, what it means for a question to be replied to with an agreement. The framework must be specific enough to facilitate software implementation of dialogue games and study of the theoretical properties of games, but at the same time it should allow for the analysis and modelling of real 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. Similar to conversation policies in multi-agent communications [5], these templates can be mixed and matched to form different types of protocols. Dialogues with an explicit reply structure can then be formed by instantiating and combining multiple templates. Thus the syntax and semantics of arbitrary dialogues can be represented in a common format. As a case in point, we present [27]'s CB game as a set of dialogue templates. Furthermore, dialogue templates allow (with additional future software) the execution of arbitrary protocols, which allows arguments to be constructed and navigated, and properties of protocols to 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 the infrastructure of an automatic dialogue game execution platform, and section 4 discusses the framework for expressing dialogue and protocol. Section 5 discusses related research and 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 then giving a reason (i.e. 'It is expensive', 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 [13]. 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 (as in, 'he prepared an argument'). On the other hand, argument$_2$ (further referred to as 'dialogue') refers to a debate or discussion (as in, 'they had an argument').

In order to understand the link between dialogue and argument, we need to consider the idea of a speech act [25]. 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), (2) and (3) have the same propositional content – 'Britain should stop Trident' – but differing illocutionary force – questioning, asserting and challenging, respectively.

2.1. Dialogue protocols

Some of the 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 that 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 simple 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, 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 [15] who define the outcome of a dialogue in terms of 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 [11][12][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 [10], 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 [21], 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 [8]. The Dialogue Game Execution Platform (DGEP, Figure 1) offers a means for any implementation which requires dialogue execution to make use of *any* available protocol. Note furthermore that if we want to execute arbitrary protocols we will inevitably need some toolkit for rapidly constructing them.

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 a variety of argument visualisation and construction tools [9]. 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.

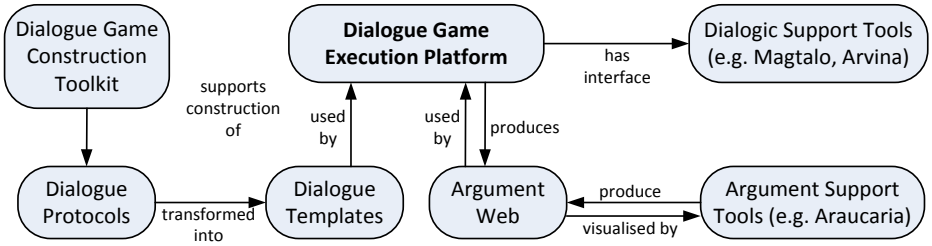

Figure 1: Specification for the automatic processing of dialogues

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 representing 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), it becomes possible to 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, which facilitates analysis of real-world argument and dialogue. The language of the AIF is also 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, 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).

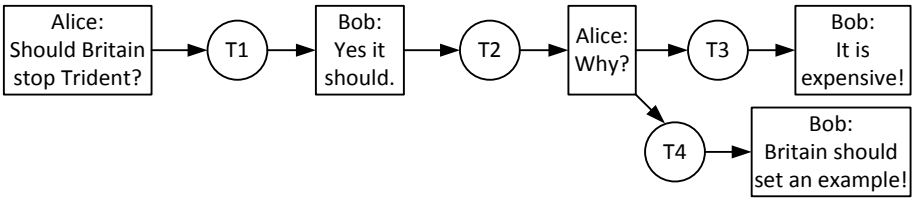

Figure 2: An example of locutions and transitions in argumentative dialogue

4.2. Illocutionary force in dialogue

With dialogue represented as dialogue-graphs, and argument as inference-graphs, it is convenient 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.

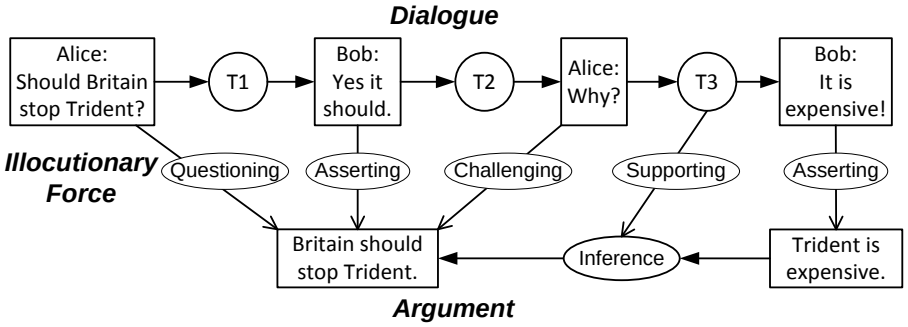

Figure 3: Illocutionary force as the link between dialogue and argument

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. [16]) contain explicit speech acts for posing an argument 'Trident is expensive *so* Britain should stop Trident'. Such a locution could 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 a definition of acceptability; whether or not one set of nodes is a consequence of another set according to a 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, and thus they are dependent on the exact formal system they invoke. Furthermore, the properties are often non-monotonic, whereas the argument graphs used here must be monotonic in that any dialogical update is guaranteed not to remove material from the graph. 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 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 framework is the only model that explicitly distinguishes between these elements that come directly from speech act theory [25].

Definition 1 [Move in dialogue] A move in a dialogue consists of a locution L and a set of pairs (IF, φ), where IF is an illocutionary force indicator and φ is the propositional content. 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*)*, rendering only the player, time, illocutionary force and propositional content. 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 2 [Dialogue Template] A dialogue template (*Start*, *End*, *Pres*, *IF*, φ) 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 and the target of this illocutionary force φ.

Table 1 shows the dialogue templates for CB. Templates 8 – 14 are explicitly mentioned in the rules of CB. Templates 1 – 7 and 15, however, require 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_i *asserting* φ	1,2	
9	p_i *questioning* φ	p_i *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.

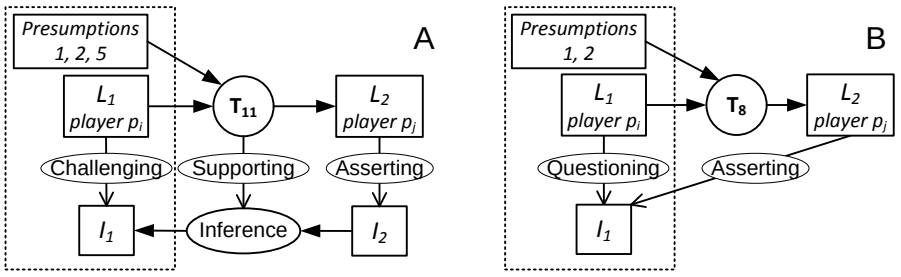

Figure 4: Dialogue Templates based on CB.

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 briefly 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 needs to 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 a 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 allow for the automatic execution of dialogue protocols and 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 great 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 large amount of non-dialogical argument data that can act as the basis for these automatically constructed dialogues.

Work that has studied classes of dialogue protocols is [15], which defines *atomic protocols*, fragments from which a dialogue can be created (e.g. *challenge x – assert y – accept x*). The authors discuss how these atomic protocols can be combined and what some of the formal properties are of the resulting dialogue protocols. Other work that has discussed generic dialogue protocols in a principled way is by Prakken [16][17], who defines an explicit reply structure for persuasion dialogue and discusses the connection to an underlying (non-dialogical) argument framework is also discussed.

The framework of dialogue graphs and templates proposed in the current paper is explicitly inspired by this previous work. Dialogue templates are very similar to atomic protocols, and in future research we intend to show that the results from [15] also apply to the appropriate dialogue templates. Furthermore, Prakken's ways of determining whether the player of a move is winning are also applicable to dialogue graphs. Thus, [15][16] have 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.

The current work expands on [15] [16] in various ways. In addition to the basic templates, ways of calculating presumptions (specifically presumptions concerning commitment) are being developed for the DGEP. This focus on implementation in the DGEP also means a connection to the wider Argument Web; thus we aim to increase the relevance and applicability of the formal and theoretical results achieved in [15][16]. Furthermore, the current objective is not just to define and discuss a fixed set of atomic protocols, but rather to explore how existing protocols can be compiled into individual templates and what this compilation means for the properties of the resultant dialogues.

Our approach to argumentation, in which the focus is on the correct representation and conceptualisation of argumentative information, allows us to integrate computational argumentation with ideas from linguistic and philosophical approaches to argumentation [2][23][26]. As an example of an advantage this connection brings, consider the relevance of reasons. In [16], there is no 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. However, locutions of the form φ *since* ψ are quite rare in natural argumentation: the rarity of this form has led to a specific name for the enthymematic Why-Because structure, *Modus Brevis* [3]. By explicitly building on

the linguistic approach of [25] our framework is suitable for capturing such natural forms of argumentation.

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

[4] P.M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n–person games. Artificial Intelligence (1995) 77:2, 321 – 357.

[5] M. Greaves, H. Holmback, and J. Bradshaw. What is a conversation policy? In F. Dignum and M. Greaves (eds), *Issues in Agent Communication*, LNAI 1916, 118–131. Springer, 2000.

[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. Lawrence, F. Bex and C. Reed. Dialogues on the Argument Web: Mixed Initiative Argumentation with Arvina. *Computational Models of Argument: Proceedings of COMMA 2012*, to appear.

[9] J. Lawrence, F. Bex, M. Snaith and C. Reed. AIFdb: Infrastructure for the Argument Web. *Computational Models of Argument: Proceedings of COMMA 2012*, to appear.

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

[11] 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.

[12] 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.

[13] 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.

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

[15] S. Parsons, P. McBurney and M. Wooldridge Some preliminary steps towards a meta-theory for formal inter-agent dialogues. *1st International Workshop on Argumentation in Multi-Agent Systems* (ArgMAS 2004), AAMAS 2004, New York, NY, USA.

[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] F.H. van Eemeren and R. Grootendorst. *Speech acts in argumentative discussions: A theoretical model for the analysis of discussions directed towards solving conflicts of opinion*. Foris, 1984.

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

[26] J. Visser, F. Bex, C. Reed, and B. Garssen. Correspondence between the pragma dialectical discussion model and the argument interchange format. *Studies in Logic, Grammar and Rhetoric*, 36, 2011.

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

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

[29] T. Yuan, D. Moore, C. Reed, A. Ravenscroft, and N. Maudet. Informal Logic Dialogue Games in Human-Computer Dialogue, *Knowledge Engineering Review* (2011) 26(3).