DE GRUYTER OPEN

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 36 (49) 2014

DOI: 10.2478/slgr-2014-0009

Simon Wells
University of Aberdeen

SUPPORTING ARGUMENTATION SCHEMES IN ARGUMENTATIVE DIALOGUE GAMES

Abstract. This paper reports preliminary work into the exploitation of argumentation schemes within dialogue games. We identify a property of dialogue games that we call "scheme awareness" that captures the relationship between dialogue game systems and argumentation schemes. Scheme awareness is used to examine the ways in which existing dialogue games utilise argumentation schemes and consequently the degree with which a dialogue game can be used to construct argument structures. The aim is to develop a set of guidelines for dialogue game design, which feed into a set of Dialogue Game Description Language (DGDL) extensions in turn enabling dialogue games to better exploit argumentation schemes.

1. Introduction

Argumentation schemes have become established as useful formal argumentation tools that enable arguments to be analysed and collated according to the stereotypical patterns of reasoning that they exhibit. A large number of natural language arguments have been analysed, for example the Araucaria corpus[1], to yield several groups of otherwise undifferentiated schemes known as scheme-sets. There are three main groups of computational scheme-sets, due to Katzav-Reed [Katzav et. al., 2004], Pollock [Pollock, 1995], and Walton [Walton et al., 2008]. Schemes from any of these sets can be used within the Araucaria tool [Reed and Rowe, 2004] to annotate a specific analysed argument structure and indicate that it is an example of a specific scheme. In this context, schemes provide a mechanism for collating, comparing, and evaluating instances of arguments. This utility has enabled schemes to be adopted as a way to partition, categorise, and organise knowledge domains and to model a variety of associated reasoning methods in a way that is amenable to computational reuse. Argument schemes have been exploited across a range of domains and contexts and include within democratic deliberation [Bench-Capon et al., 2011], hypothetical reasoning

ISBN 978–83–7431–402–2 ISSN 0860-150X

[Bench-Capon and Prakken, 2010] and case-based reasoning [Wyner and Bench-Capon, 2007], [Wyner et al., 2011], and [Prakken et al., 2013]. More recently Bench-Capon *et al* [Bench-Capon et al., 2013] present schemes for differentiating between legal cases on the basis of social values. Schemes have also been used to guide argument generation and to suggest relevant responses within dialogue. From this perspective, when a speaker makes an utterance, and the locutionary act and associated content constitute the utterance of an element of an argument, such as stating a conclusion, then a listener has a range of responses that they can be licensed to make according to the protocol, such as exploring the basis for the argument by inquiring about the premises that support the uttered conclusion, or else by enquiring about the reasoning associated with the argument by uttering critical questions associated with the scheme.

The development of each of dialogue games and argumentation schemes has, with a few exceptions, occurred in parallel, and yet, used in concert, schemes and games are hugely complimentary. Schemes can be used in relation to dialogue games both at the development stage, to provide guiding principles for developing new game rules, and at the deployment stage, to provide guidance towards relevant lines of argument for the player to explore, to suggest appropriate responses to the expressed positions of others, and to provide a facet of strategic information which a player can use in their reasoning process in order to achieve their desired goals. However, whilst Argumentative dialogue has become a popular approach to structuring interaction, for example between people [Ravenscroft and Matheson, 2002], between people and intelligent agents [Reed and Wells, 2007] in pedagogic and mixed-initiative systems, and between intelligent agents within Multi-agent Systems (MAS) [Parsons et al., 1998] the dialogue games that are available have not fully exploited the benefits that are to be wrought from argumentation schemes. Argumentation schemes capture the constituent parts of arguments in a well-structured and constrained way, tending towards minimalism, but with sufficient additional structure to enable meta-level manipulation of arguments to be performed using automated computational tools.

This approach becomes increasingly important when argumentation tools are applied to real-world problem domains. For example, the SASSY project[2] aims to use argumentation to provide scrutability about decisions made by intelligent systems, similarly in the SUPERHUB project[3] argumentation schemes appear to be a useful way to capture the patterns of reasoning used by 'critic' agents within a multi-modal journey planner in order to identify potential plans that conform to the established prefer-

ences of the user. In both these projects, getting from the recognition that schemes might be very useful to an implemented system has proven problematic as it is not just a matter of schematising the relevant reasoning into arguments but also providing appropriate ways to interact with those arguments. Additionally it is necessary to adopt a method that is sufficiently flexible to support multiple dialogue protocols rather than attempting to design a game that accounts for all potential types of interaction. Within a knowledge domain, there may be a range of different groups of people whose communicative acts may be constrained in different ways, such as the kinds of arguments and dialogues that may occur at different levels of the legal process.

To enable dialogue games to better exploit argumentation schemes, a useful approach is to identify the range of constituent elements of extant schemes and to ensure that a given dialogue game is aware of those elements and can utilise them within a dialogue. To this end, we identify and define a property of dialogue games called *scheme awareness* that can be used to determine the degree to which a given game can exploit schemes within a dialogue and the game features that support this. Based upon this we are able to do two things:

1. Provide guidelines for the development of new dialogue games, and
2. Extend existing dialogue game description frameworks to account for scheme awareness.

Our aim in this paper is to take the approach of starting at the dialogical level, and then asking, what does it mean for a dialogue game to be able to exploit argumentation schemes? and consequently what properties does a game need to possess in order to exploit those schemes? Our aim is to provide just enough support for argumentation schemes within dialogue games so that it becomes straightforward to take an existing scheme-based knowledge base and effectively utilise that within a dialogue game.

The remainder of this paper is structured as follows; in section 2 we survey relevant background and related work and establish the need for increased argumentation scheme support in dialogue games. In section 3 we identify a set of features of dialogue games that are pre-requisites for effectively utilising argumentation schemes within dialogue games. On the basis of this we identify how dialogue games can be made 'aware' of and utilise argument schemes. Subsequently, in section 4 we identify a set of minimal alterations that must be made to the DGDL grammar to support increased scheme awareness in dialogue games described using the DGDL. Finally, we draw some conclusions and outline some directions for future work.

2. Background and Related Work

A number of approaches have been taken that attempt to unite arguments, schemes, and dialogue. The main approaches have been dialogue games that handle schemes, and description languages that variously describe arguments and or dialogue protocols. Beginning with dialogue games that handle schemes, if we envision a hierarchy of dialogue game groups organised with respect to argumentation schemes we might produce the following:

1. Games unable to utilise argumentation schemes.
2. Games able to utilise a single scheme.
3. Games able to utilise multiple/arbitrary schemes.

By 'utilise' we mean explicitly able to represent and/or manipulate either individual argumentation schemes or scheme-sets within the rules of the game. A stronger sense of the term utilise would require that the dialogue game not resort to external mechanisms such as making the implicit assumption that a human-agent will provide the necessary intelligence or that an external 'scheme recognising' computational module is available. However, under the stronger interpretation there are currently no games at level 3 so we relax this assumption to enable us to differentiate between those games that can explicitly represent some elements of argumentation schemes, such as those described below at levels 2 and 3, and those that do not explicitly incorporate any concept of argument schemes. This is commensurate with our wider goal of providing increased support for automated computational use of dialogue games and argumentation schemes and the longer term goal of this line of research is to describe games and associated game-playing engines that are self-contained and do not rely upon *deus ex machina* support for scheme recognition in order to be 'playable'.

There are many dialogue games that exist at level 1, including but not limited to Walton and Krabbe's PPD_0 [Walton and Krabbe, 1995, pp. 149–152], and RPD_0 [Walton and Krabbe, 1995, pp. 158–161], Girle's games for belief revision [Girle, 1993] [Girle, 1994] [Girle, 1996], Walton's CB family of games [Walton, 1984], and various games due to McBurney and Parsons of which [McBurney and Parsons, 2002] is representative. We point this out not to suggest that any of these games are deficient but merely to recognise that they were not designed to support the kind of linkage between schemes and dialogue that we are currently proposing. There is nothing to stop any of these games, or for that matter any of the many other games at level 1, from being enhanced with extra functionality in the form of new rules or locution types to support use of schemes, just that the basic games as

presented do not include this functionality and hence, for our purposes, they are unable to utilise argumentation schemes.

Under this approach we see that there have also been games developed at level 2, for example Atkinson's games that utilise the practical reasoning scheme of which [Atkinson et al., 2004] is representative here. The practical reasoning argument schema, denoted AS1, is as follows:

> In the Current Circumstances, R, we should perform Action, A, to achieve New Circumstances, S, which will realize some goal, G, which will promote some value, V.

Following this, Atkinson has presented various protocols, for example the PARMA protocol [Atkinson et al., 2006] for working with arguments structured using AS1, as well as a comprehensive range of critical questions associated with AS1. In Atkinson's approach, the dialogue games are carefully described to account for the dialogical interactions associated with a specific argument scheme, the aforementioned practical reasoning scheme, and are therefore not immediately applicable in other contexts, for example, replacing the practical reasoning scheme with one from the Reed-Katzav scheme-set. An earlier game due to Bench-Capon, the Toulmin Dialogue Game or TDG [Bench-Capon, 1998], is another example of a dialogue game that is explicitly based upon a specific argumentation scheme, in this case the Toulmin Argument Scheme [Toulmin, 1958]. In TDG the moves of the game correspond closely to the constituent elements of a modified Toulmin schema, in which the qualifier role has been removed and a presupposition role added, and games are played using a knowledge base in which Toulmin schemas are chained together.

There have also been attempts, albeit not wholly successful, to define how games at level 3 might work. These approaches utilise schemes in the weaker sense described earlier. For example, in [Reed and Walton, 2004], Walton's game CB [Walton, 1984] is extended to support basic integration of critical questions through the addition of a 'Pose C' move, related to posing a critical question, and support for recognising that an argument is a 'substitution instance' of a scheme.

The Dialogue Game Description Language (DGDL) [Wells and Reed, 2012] is a domain specific language (DSL) based upon the Ph.D thesis research of [Wells, 2007] for describing the rules of dialogue games, rather than an actual dialogue game itself. DGDL descriptions are underpinned by an extended Backus-Naur Form (EBNF) grammar [Wirth, 1977] which enables games to be described that are expressive, consistent, and syntactically verifiable and which is illustrated in Figure 1. Particular note should

```
System          ::= SystemID { [Game]+} | Game
SystemID        ::= Identifier
Game            ::= GameID { Composition [Rule]* [Interaction]+ }
GameID          ::= Identifier
Composition     ::= Turns [RoleList]? Participants, [Player]+ [Store]*
Turns           ::= {turns, TurnSize, Ordering, [MaxTurns]? }
TurnSize        ::= magnitude: Number | single | multiple
Ordering        ::= ordering: strict | liberal
MaxTurns        ::= max: Number | RunTimeVar
RunTimeVar      ::= $Identifier$
RoleList        ::= {roles, { Role [, Role]+ } }
Role            ::= speaker | listener | Identifier
Participants    ::= {players, min: Number, max: Number|undefined }
Player          ::= {player, id: PlayerID|RunTimeVar[, roles:{ Roles }]? }
PlayerID        ::= Identifier
Store           ::= {store, id: StoreName, owner=StoreOwner, StoreStructure,
                    Visibility}
StoreName       ::= Identifier
StoreOwner      ::= PlayerID | { PlayerID[, PlayerID] } | shared
StoreStructure  ::= structure:set|queue|stack
Visibility      ::= visibility:public|private
Rule            ::= {RuleID, scope:initial|turnwise|movewise, RuleBody}
RuleID          ::= Identifier
RuleBody        ::= Effects | Conditional [& Conditional]*
Effects         ::= { Effect [& Effect]* } | { Effect [|| Effect]* } |
                    { Effects [&||| Effects]* }
Effect          ::= EffectID( Parameter [, Parameter]* )
Parameter       ::= Identifier|Number|Commitment|SystemID|GameID|PlayerID
                    |MaxTurns|StoreName|StoreOwner|Requirements|Role
                    |RunTimeVar|Condition|Effect
Interaction     ::= { MoveID, Content [,Opener]?, RuleBody }
MoveID          ::= Identifier
Content         ::= {ContentSet|ContentVar[,ContentSet|ContentVar]*}
ContentSet      ::= UpperChar
ContentVar      ::= LowerChar | !LowerChar
Opener          ::= String
Conditional     ::= {if Requirements then Effects [elseif Requirements then
                    Effects]* [else Effects]?}
Requirements    ::= {Condition [& Condition]*} | {Requirements[ ||
                    Requirements]*}
Condition       ::= ConditionID( Parameter [, Parameter]* )
ConditionID     ::= Identifier
Commitment      ::= Content | Locution | Argument
Locution        ::= < MoveID, Content >
Argument        ::= < Conclusion, Premises >
Premises        ::= {ContentVar[, ContentVar]*}
Conclusion      ::= ContentVar
SchemeID        ::= Identifier
Identifier      ::= UpperChar [ UpperChar | LowerChar | Number ]+
String          ::= ‘‘[UpperChar|LowerChar|Number|Symbol]+’’
Number          ::= [0--9]+
UpperChar       ::= [A--Z]+
LowerChar       ::= [a--z]+
Symbol          ::= ‘ ’|‘?’|‘,’|‘.’
```

Figure 1. EBNF Grammar for the Dialogue Game Description Language

be taken of the Condition and Effect clauses in the grammar, which allow the set of conditions and effects used to describe the circumstances under which a move can be made, and the resulting effect of making such a move, to be specified by the game designer. This enables the total range of games that can be described by the DGDL to be extended to account for new dialogue game features without requiring alterations to be made to the grammar. An initial collection of conditions and effects was presented in [Wells and Reed, 2012] which includes the following conditions:

```
Event              ::= event( last|!last|past|!past, MoveID [,Content]?
                       [, PlayerID|Role]? [, Requirements]? )
StoreInspection    ::= inspect( in|!in|on|!on|top|!top, Commitment,
                       StoreName, [PlayerID|Role]?
                       [, initial|past|current]? )
RoleInspection     ::= inrole( PlayerID, Role )
Magnitude          ::= size( StoreName|LegalMoves, PlayerID,
                       empty|!empty|Number )
StoreComparison    ::= magnitude( StoreName, PlayerID|Role,
                       greater|smaller|equal|!equal, StoreName,
                       PlayerID|Role, )
DialogueSize       ::= numturns( SystemName, Number )
Correspondence     ::= corresponds( Argument, SchemeID )
Relation           ::= relation( Content|Argument, backing|warrant,
                       Content|Argument )
CurrentPlayer      ::= player( PlayerID|Role )
ExternalCondition ::= extCondition( Identifier [, Identifier]* )
```

and the following effects:

```
Move               ::= move( permit|mandate,next|!next|future|!future,
                       MoveID, [, Content]? [, PlayerID|Role]? )
StoreOp            ::= store(add|remove, Commitment, StoreName,
                       PlayerID|Role )
StatusUpdate       ::= status( active|inactive|complete|incomplete|
                       initiate|terminate, SystemID|GameID )
RoleAssignment     ::= assign( PlayerID|Role, Role )
ExternalEffect     ::= extEffect( Identifier [, Identifier]* )
```

An example of a minimal dialogue game, in the spirit of Hamblin's simplest dialectical system consisting of an "interchange of statements about the weather" [Hamblin, 1970, pp. 256] described using the DGDL and for purely illustrative purposes is as follows:

```
Simple{
  {turns,magnitude:single,ordering:strict}
  {players,min:2,max:2}
  {player,id:Player1}
  {player,id:Player2}
  {store,id:CStore,owner:Player1}
  {store,id:CStore,owner:Player2}
  {Assert,{p},``I assert that'',{store(add, {p}, CStore, Speaker)}}
}
```

In this "Simple" game we have a turn structure which allows one move per turn and a strict ordering of turn progression. Two players are defined, 'Player1' and 'Player2' and each player owns a store called 'CStore' in which their commitments are stored. There is a single locution available to be played, called 'Assert' whose only effect is to add the content of the assertion to the speaker's commitment store.

Because the DGDL is used to describes games, rather than being a game itself, it does not fit into the hierarchy outlined earlier. However, the DGDL may be used to describe games at any of the three levels of the hierarchy. This is achieved by incorporating support for describing game rules that deal with schemes in a manner similar to that of [Reed and Walton, 2004]. For example, when formulating a DGDL game description, rules can be introduced that enable a condition to be described using the correspondence condition predicate:

$$\text{Correspondence} ::= \text{corresponds(Argument, SchemeID)}$$

which indicates that a given argument corresponds to, or is an instance of, a particular argumentation scheme identified by SchemeID. If the condition is met then associated effects can be applied to the state of the dialogue, such as licensing the utterance of a critical question associated with the scheme. Unfortunately this approach relies either upon the players being human, and therefore possessing the ability to recognise that an argument is an instance of a give scheme, or else that the game engine has access to some external scheme identifying functionality. Whilst this circumstance is acceptable, the DGDL is meant to be able to describe games that can be played solely by humans, or solely by agents, or by any combination of the two, there is still room for a lot of improvement in the scheme handling of the DGDL. An approach leading to such an improvement is presented in section 4.

The majority of dialogue games occupy group 1 and require additional rules to enable schemes to be used. This suggests that there are two approaches to incorporating argumentation schemes into dialogue games (1) describe a new game that utilises schemes, and (2) adopt an existing game and retrofit with scheme specific functionality. In either case it is useful to enquire which properties a game must possess in order to move from group 1 to either group 2 or group 3.

Two further approaches that have some bearing on the work reported in this paper are scheme support in the Argument Markup Language (AML) [Reed and Row, 2004] and scheme support in the Argument Interchange Format (AIF) [Rahwan et al., 2007]. In [Reed and Walton, 2004] arguments and

schemes are recorded using AML, fragments of which are generated from a pre-partitioned agent belief database, and exchanged between agents during their communications. However this approach does not examine argumentative communication from the dialogical perspective but rather deals with arguments and schemes at the language level, arguments and their associated schemes are communicated entirely as content and the protocol itself is not 'aware' of the additional data available within the content, hence, it is difficult to exploit the additional context sensitive information provided by the scheme when constructing and selecting subsequent moves to perform. Additionally, AML has largely been superceded by the Argument Interchange Format (AIF) [Chesnevar et al., 2006] which provides both improved support for representing multiple conflicting and supporting arguments using a graph-based framework, but also a more fine grained integration of individual arguments and their associated schemes. This suggests that an alternative approach, that would unite arguments, dialogues, and schemes, might be to extend the Argument Interchange Format (AIF) [Chesnevar et al., 2006]. However, whilst it is true that the AIF already supports expression of arguments and schemes, it does not currently support dialogue very well. The most advanced approaches to dialogue in the AIF can be found in [Reed et al, 2008] and [Ravenscroft et al., 2009] which support for locutional elements within AIF documents, and [Reed et al., 2010]. However these approaches are currently unsupported by tooling and lack fine grained expressions for defining dialogue protocols. Furthermore, the AIF is, by definition, a high-level tool that is designed to be very flexible, enabling interchange of argument structures between disparate tools. However that is a different endeavour to the provision of basic, targeted support for schemes within dialogue games. A preferred alternative is to start with the description of dialogue games, for example as exemplified in the DGDL, and extend this to provide sufficient support for schemes.

3. Scheme Awareness

A pre-requisite for an argumentative dialogue game, before argument schemes are even taken into account, is that the game enables arguments to be expressed. An argument comprises a number of statements that are related. One statement is named the conclusion and the other statements are named premises, related such that the conclusion is said to follow from the premises. Furthermore the set of premises may be subdivided such that one is named the major premise and expresses a rule that defines how the re-

maining premises, named minor premises, support the conclusion. It should be noted at this point that the use of the terminology major and minor to distinguish the premise that acts as a rule from the other premises is merely a matter of stylistic choice and does not indicate adherence to any philosophical position on the grouping or lack thereof of constituent premises within argument.

A game should, minimally, support the expression of whole arguments as a single complex utterance. An example of this is found in the dialogue game PPD_0 in which the move named ΔSoP enables a player to utter both a conclusion, P, and its set of supporting premises, Δ within a single move. However, a more expressive game would allow arguments to be expressed either in whole, at a coarse grained level, as is found in ΔSoP or in part, at the fine grained level, by uttering individual locutions that introduce the constituent parts of a given argument, conclusions, premises, &c. piecemeal as the dialogue progresses. A piecemeal approach however enables a dialogue game to more closely model natural dialogues, an important factor in mixed-initiative scenarios. How a game supports the components of argument, whether in whole or piecemeal, affects how expressive a game can be considered to be and also lays the foundation on which awareness of argumentation schemes can be built.

Additionally, if a game allows arguments to be expressed in a piecemeal manner, then when an argument is uttered it may be completely expressed, corresponding to the principle of total evidence [Carnap, 1947], or else partially expressed, in which case the argument is enthymematic. As outlined above we assume that arguments arise during a sufficiently expressive dialogue game, and that a given argument may not have been fully expressed within the dialogue game at a given time point, although the argument may be completed at subsequent time-points in the remainder of the dialogue. This captures the idea that a dialogue is dynamic and that an under-specified argument expressed at time-point T_n may be elaborated on at some subsequent time-point $\text{T}_{n'}$. Recognising this gives rise to the question of whether the dialogue game enable arguments to be fully expressed and recognisable at the dialogue game level (as opposed to assuming that the argument can be parsed from the underlying logical language level)?

This is an important consideration because a game in which it is difficult to express fully formed and identifiable argument structures is one in which it will also be difficult to incorporate other machinery that builds on those argument structures without resorting to a *deus ex machina* solution. Many dialogue games were not intended to inform computational implemen-

tations, and yet some of them, Mackenzie's DC for example, have become influential, underpinning a range of subsequent games, either through extension as occurs in the game DE [Yuan and Wells, 2013] or through influencing the range of available locutions, as has occurred in Moore and Hobbe's game [Moore and Hobbes, 1996] and Amgoud *et al's* system [Amgoud et al., 2000] to name but two. Whilst the responsibility for handling move content can be delegated to the logical language, or content language level, the consequence is that it becomes difficult to specify rules governing the form of move content at the dialectical game level whilst also maintaining a consistent and self-contained system. The lines that delineate the logical and content language levels, the dialectical level, the argument level, and the argument scheme level are not clean cut, and there must be judicious overlap between each to enable them to work well together. For example, whilst it seems straightforward and conceptually clean to suggest that the logical language level should deal only with the expression of what is said, and the dialectical game should deal only with what is or is not allowed to be said, it is necessary that there is overlap in at least the upwards direction, from the logical level to the dialectical level. This enables the dialogue game to specify rules that depend upon not just the performative act associated with a move, but also the content associated with that move, specifying that one response is necessary if the content has one form but that another response is required if the content has yet another form. Furthermore, if computational implementations are to be made of these games, especially following the trend towards having dialogue game engines that can load different dialogue games at runtime, then it is necessary not only that the game rules are sufficiently expressive to describe the required dialogical behaviours, but also that it is feasible for the game-engine to recognise all of the conditions described in the rules.

Dialogue games should therefore be able to support the aforementioned levels of argument expression as a prerequisite to comprehensive scheme support. These are summarised as follows and constitute the requirements for sufficient expressiveness for a dialogue game with respect to argumentation schemes:

1. Assert, or otherwise express, an entire argument within a single locution
2. Assert, or otherwise express, either individually or in combination (but still individually addressable), the constituent parts of an argument within disparate locutions:
   (a) Conclusion
   (b) Major Premise
   (c) Minor Premise(s)

Minimally, an argumentative dialogue game should support at least item 1, expression of an entire argument within a locution. The remaining items merely enable dialogue games to be specified that can yield increasingly expressive, fine-grained, and more natural dialogues.

Given these prerequisites we make the fundamental assumption that every argument, regardless of whether that argument is fully or partially expressed, is associated with an argumentation scheme. Furthermore this association may be either implicit of explicit. If the association of an argument is left implicit then, for any argument that is introduced during a dialogue, the question of which specific argumentation scheme captures the expressed argument, can be asked. Without further information or processing, the scheme associated with such an argument is undefined and is labelled as such. In the OVA tool [Reed et al., 2011], which is underpinned by the AIF, anonymous schemes are represented by unnamed RA-nodes until the user specifies the actual scheme. Conversely an explicit association occurs when an argument is defined as being an instance of a specific identified scheme, this can occur in three ways during a dialogue, firstly, through some automated function that identifies which scheme the argument corresponds to, secondly, by the speaker of the argument identifying which scheme their argument is part of, and thirdly, by the respondent deciding which scheme applies to the argument and thus which critical questions can be posed. The second and third approaches are of interest because they can potentially lead to conflict, and subsequent argument about which scheme is most appropriate if the players disagree. Because there is no reliable automated scheme identification mechanism available to satisfy the first approach, it is necessary therefore that the dialogue game incorporates rules that enable the players to associate an argument, or element thereof, with a specific scheme.

We are now in a position to identify some aspects of scheme awareness, linking expressiveness with respect to arguments and argumentation schemes. Assuming that the aforementioned argument-related pre-requisites are met, a dialogue game should enable the players to express how the content of a given locution, if it is argumentative, relates to a specific scheme. Assuming that an external argumentation scheme server is available enables individual argumentation schemes to be identified and retrieved, one example of which is the AIFdb[4] which enables schemes to be retrieved programmatically by *schemeTypeID* using an HTTP/JSON Application Programming Interface (API). Given this, we can identify the following requirements that dialogue games should identify in relation to argumentation schemes, in addition to the requirements of argumentative expressiveness a game should:

1. enable the speaker to declare that an argument is part of a specific scheme, and,
2. enable the respondent to declare that an argument is part of a specific scheme.

During a dialogue, arguments rarely exist in isolation, but are linked and chained with other arguments to form more complex structures. A given statement may therefore have multiple roles acting as the conclusion of one argument, but also acting as the minor premise in a further argument. This has a bearing on the relationship between arguments, as expressed during dialogue, and schemes. For example, it suggests that every argumentative statement uttered during a dialogue is associated with at least one argumentation scheme and that game engines which support play of dialogue games should support individual statements being recorded as occupying roles in 1 or more schemes. This is particularly necessary if the game enables the players to disagree over which scheme to associated with a given argument.

Critical questions follow from the identification of an argument as being a part of a specific scheme. This licences the resulting utterance of the associated critical questions during the dialogue. A dialogue game should therefore support the utterance of relevant critical questions associated with a given scheme and an asserted argument. Once a scheme is identified, the associated critical questions should then become available using a similar mechanism to the pose locution of [Reed and Walton, 2004].

In this section we have identified a range of features of dialogue games that can be deployed in two contexts. In the first context, the features enable a determination to be made about the degree of scheme awareness that a given dialogue game has. This is useful when comparing and evaluating existing games from the literature, for example, when determining whether to adopt an existing dialogue game protocol within a given problem domain. In the second context, the features constitute the basis for a set of guidelines for how to build a dialogue game that is argumentation scheme aware.

4. Extending the DGDL

In this section we provide a minimal extension to the DGDL that enables DGDL game descriptions to be made more scheme aware. If the state of the art for argument mining was sufficiently advanced, we could rely upon automatic scheme recognition engines to identify that the content of the current locution constitutes an element of an instance of a given scheme and make that recognition, and associated data about the recognised scheme,

available to the DGDL game-engine. However, automatically recognising that an argument is an instance of a particular scheme is a difficult problem. At the time of writing, the current state of the art for argument mining is probably represented by the TextCoop platform [Saint-Dizier, 2012] which can recognise and categorise utterances according to locutional type or performative act with an impressive degree of success but does not yet categorise the recognised arguments according to the schema that they fulfill. Rather than attempt to solve that problem we shall instead attempt to circumvent it by assuming that an agent, knowing the argument that it is deploying within a dialogue, can include that information explicitly in it's utterances and thus communicate which scheme is associated with the expressed argument. By ensuring that players are explicit when uttering an argument about which scheme that argument, or part thereof, is representative of, we can obviate the need for an automated scheme recognition engine at this point. It should be noted that games described using the extended DGDL need not by scheme aware, just that this extension enables increased scheme awareness in the games that are so-described. Additionally, when a specific game is described using the DGDL, it is a decision for the game designer to make about whether it is mandatory for the players to identify the scheme associated with the utterances of a given argument. It can thus become a strategic issue of whether the player deems it necessary to inform the other player of the scheme associated with an argument.

The simplest method to extend the DGDL, taking into account the requirements and features introduced in section 3 is to introduce more structure into the content of moves whilst retaining the current flexibility of locution naming. Currently when a player makes a move they utter a locution and content, e.g. assert(p). Previous attempts to incorporate argumentation schemes have concentrated on the locution, for example, extending the basic assertion locution to account for schemes by introducing an assert_scheme_argument. However, by providing additional information into the content, we can enable the player to declare properties associated with the content during the dialogue. For example, each element of the content of a move can be supplied with meta-data by the utterer to make it clear the role that the content plays in the argument being constructed. The minimum set of meta-data that the DGDL should support to enable an item of content to be labelled with respect to its status in relation to argumentation schemes is the following:

- Label content as an instance of an argument
- Label content as the conclusion of an argument.
- Label content as the major premise of an argument.

– Label content as a minor premise in an argument.
– Label content as an element of a type of argumentation scheme.
– Label content as part of an instantiated argumentation scheme.

A simple way to extend the DGDL to account for the aforementioned content meta-data is to use one, or more, comma-separated key:value pairs to annotate asserted content during a dialogue, e.g. assert(k:v) where the keys and values are DGDL identifiers. Table 1 provides a specification of keys for each item in the list of label content given above:

Table 1

Key labels for each content type

Content Type	Key
Content is an instance of an argument	argument
Content is the conclusion of an argument	conclusion
Content is the major premise of an argument	major-premise
Content is a minor premise of an argument	minor-premise
Content is an element of an identified argumentation scheme	scheme-name
Content is an element of an instantiated argumentation scheme	scheme-instance

It should be noted in the current and following discussion that the example moves are only indicative of the kinds of moves that a game described using the DGDL might incorporate and are not excerpted from an actual game. Actual locution labels, content labels, and keys and values, would also be associated with a complete, grammar consistent rule-body that defines the legality requirements for playing the move and the resulting effects of so doing. A minimal requirement is that at least one value represents the locution's content variable, and the key indicates the type, e.g. one from the set {argument, conclusion, premise, rule}. For moves that incorporate more than one item of content, for example, to make the move assert(p,q) which asserts both p and q into a scheme-aware move, each item of content must be labelled, e.g. assert("conclusion":"p", "major-premise":"q").

Additionally a key:value pair can be used to declare that the content is associated with a specific scheme, e.g. "scheme":"slippery-slope". As an example, to assert that the statement, p, is the conclusion of an argument and furthermore that the scheme associated with the argument that this assertion is the conclusion of is an instance of the argument from expert-opinion we can use the following expression: assert("conclusion":"p", "scheme-name":"expert-opinion"). The advantage of taking this approach is that we increase the ability of players to make explicit what they mean to

say when they make a move, and by increasing the explicitness of exchanged utterances we simplify any subsequent computational processing.

When parsing a DGDL description the content of a move is read from left to right and each element that does not introduce a new element of content is associated with the content to it's left. Furthermore, each item of content may be associated with as many elements of meta-data as required until the next item of content is parsed. This enables a single declaration of content, to be labelled with as much meta-data as is required for the speaker to fully describe the status of their utterance. For example, in the following move assert("conclusion":"p", "scheme-name":"expert-opinion", "scheme-instance":"123321", "major-premise":"q"), the speaker has asserted the content, 'p' which is labelled as the conclusion of an instance of the 'argument from expert opinion' argumentation scheme and the particular instance of the scheme has the id '123321'.

To support the Key:Value content approach requires only a single alteration to be made to the DGDL grammar. The affected grammar production rule is the Content rule:

```
Content          ::= {[String:]*ContentSet|ContentVar|String
                      [, [String:]*ContentSet|ContentVar|String ]*}
```

which is altered to enable any existing expression of content to be optionally accompanied by a preprended "String:" element that represents the Key. Additionally a Key:Value clause may be used that consists of two string elements expressed thus: "String:String". It is a matter for further research to determine exactly what the set of "String" elements should consist of. In this paper we have explored the minimal set of requirements to enable an expressive labelling of move content with argumentation scheme oriented meta-data. However similar meta-data might be used by the speaker to incorporate further information associated with an utterance, for example, to label an argument with a measure of strength or an indication of certainty.

To some degree this approach is an extension of the responsibility of dialogue games to include wider responsibilities including elements of argument construction and representation. One criticism of this approach is that it risks subsuming too many elements of wider argumentation theory topics into the sphere of dialogue games. However, what we are advocating is merely the extension of dialogue games to provide sufficient support for schemes so that arguments and argument schemes can be easily utilised by dialogue games without additional processing, and so that the outputs from a dialogue game, such as the transcript of a dialogue, or the argu-

ments constructed during a dialogue contain enough structure that they can be used by tools developed for working with argument schemes. To this end we propose that there should be sufficient overlap of responsibilities to enable efficient automated sharing of data between dialogue game and argumentation scheme tools, and to enable those tools to work together, recognising that many argumentation tools now exist that are targeted at different groups of problems.

In this section we have presented an extension to the existing DGDL grammar that enables Key:Value pairs to be represented within the content portion of dialogue game moves. This enables game descriptions to associate additional information with the content of a locution thus enabling the status of the content to be declared by the players during a dialogue.

5. Conclusions & Future Work

The author holds that argumentation schemes are a useful way to organise and collate arguments when attempting to deploy computational argumentation technologies within complex real-world domains. In these contexts the system is meant to support some combination of computational efficiency, scrutability and introspection, and alignment with human reasoning and interaction processes. This holds even more strongly where mixed initiative systems are concerned in which groups of humans and agents may interact via dialogue games. This effort feeds into ongoing research that aims to align argumentative technologies with real-world problems as currently there can be quite a conceptual leap from a problem domain to deployment of argumentative tools within that domain. The aim is to provide support for the development of much better tools that support this mixed human-agent arena, where participants in a game aim to explore existing argumentative structures, or to construct, or co-construct, new argumentative structures during interactions that are regulated by an agreed protocol.

To support this, dialogue games require better support for manipulating arguments. Because argumentation schemes have become established as a *de facto* method for working with instances of arguments we predicate an increased support for argument manipulation in dialogue upon a tighter integration of said games with argumentation schemes. In section 3 we identified a range of features of dialogue games that can be used to determine how 'expressive' the game is in terms of argument elements and how 'aware' the game is of argumentation schemes.

This paper has reported on the provision of structural elements within dialogue games that enable new games to be developed that better exploit the properties of argumentation schemes, supporting the construction and exploration of argumentative structure during a dialogue, and thus easing the progression from structuring a knowledge or problem domain in terms of arguments, to interacting with that domain via dialogue games.

NOTES

[1] Available from http://http://araucaria.computing.dundee.ac.uk/doku.php.

[2] http://www.abdn.ac.uk/ncs/computing/research/ark/projects/current/sassy/.

[3] http://superhub-project.eu/.

[4] http://www.arg.dundee.ac.uk/AIFdb/.

BIBLIOGRAPHY

[Amgoud et al., 2000] Amgoud, L., Maudet, N., and Parsons, S. (2000). Modeling Dialogues Using Argumentation. In Durfee, E. *Proceedings of the Fourth International Conference on MultiAgent Systems*, pp. 31–38.

[Atkinson et al., 2004] Atkinson, K., Bench-Capon, T., and McBurney, P. (2004). Justifying Practical Reasoning. In Rahwan, I., Moraitis, P., and Reed, C., editors, *First International Workshop on Argumentation in Multi-Agent Systems*.

[Atkinson et al., 2006] Atkinson, K., Bench-Capon, T., and McBurney, P. (2006). Computational representation of practical argument. *Synthese*, 152(2), pp. 157–206.

[Bench-Capon, 1998] Bench-Capon, T. J. M. (1998). Specification and implementation of toulmin dialogue game. In *Proceedings of JURIX 98*, pp. 5–20.

[Bench-Capon and Prakken, 2010] Bench-Capon, T. J. M. and Prakken, H. (2010). Using argument schemes for hypothetical reasoning in law. *Artificial Intelligence and Law*, 18(2), pp. 153–174.

[Bench-Capon et al., 2011] Bench-Capon, T. J. M., Prakken, H., and Visser, W. (2011). Argument schemes for two-phase democratic deliberation. In *Proceedings of the 13th International Conference on AI and Law (ICAIL 2011)*, pp. 21–30.

[Bench-Capon et al., 2013] Bench-Capon, T. J. M., Prakken, H., Wyner, A., and Atkinson, K. (2013). Argument schemes for reasoning about legal cases. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law (ICAIL 2013)*, pp. 13–22, Rome, Italy.

[Carnap, 1947] Carnap, R. (1947). On the application of inductive logic. *Philosophy and Phenomenological Research*, 8(1), pp. 133–148.

[Chesnevar et al., 2006] Chesnevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., and Willmott, S. (2006). Towards an argument interchange format. *Knowledge Engineering Review*, 21(4), pp. 293–316.

[Girle, 1993] Girle, R. A. (1993). Dialogue And Entrenchment. In *Proceedings Of The 6th Florida Artificial Intelligence Research Symposium*, pp. 185–189.

[Girle, 1994] Girle, R. A. (1994). Knowledge Organized And Disorganized. *Proceedings of the 7th Florida Artificial Intelligence Research Symposium*, pp. 198–203.

[Girle, 1996] Girle, R. A. (1996). Commands in dialogue logic. *Practical Reasoning: International Conference on Formal and Applied Practical Reasoning, Springer Lecture Notes in AI*, 1085, pp. 246–260.

[Hamblin, 1970] Hamblin, C. L. (1970). *Fallacies*. Methuen and Co. Ltd.

[Katzav et al., 2004] Katzav, J., Reed, C., and Rowe, G. (2004). Argument research corpus. In *Practical Applications in Language and Computers (Proceedings of the 2003 Conference)*, pp. 229–239.

[McBurney and Parsons, 2002] McBurney, P. and Parsons, S. (2002). Dialogue games in multi-agent systems. *Informal Logic*, 22(3), pp. 257–274.

[Moore and Hobbes, 1996] Moore, D. and Hobbes, D. (1996). Computational uses of philosophical dialogue theories. *Informal Logic*, 18(2 and 3), pp. 131–163.

[Parsons et al., 1998] Parsons, S., Sierra, C., and Jennings, N. (1998). Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3), pp. 261–292.

[Pollock, 1995] Pollock, J. L. (1995). *Cognitive Carpentry. A Blueprint for How to Build a Person*, MIT Press.

[Prakken et al., 2013] Prakken, H., Wyner, A., Bench-Capon, T., and Atkinson, K. (2013). A formalisation of argumentation schemes for legal case-based reasoning in aspic+ (to appear). *Journal of Logic and Computation*,

[Rahwan et al., 2007] Rahwan, I., Zablith, F., and Reed, C. (2007). Towards large scale argumentation support on the semantic web. In *Proceedings of AAAI-07*, pp. 1446–1451.

[Ravenscroft and Matheson, 2002] Ravenscroft, A. and Matheson, P. (2002). Developing and evaluating dialogue games for collaborative e-learning. *Journal of Computer Assisted Learning*, 18, pp. 93–101.

[Ravenscroft et al., 2009] Ravenscroft, A., Wells, S., Sagar, M., and Reed, C. (2009). Mapping persuasive dialogue games onto argumentation structures. In *Proceedings of the Symposium on Persuasive Technology and Digital Behaviour Intervention hosted at the AISB Convention*.

[Reed and Rowe, 2004] Reed, C. and Rowe, G. (2004). Araucaria: Software for argument analysis, diagramming and representation. *International Journal on Artificial Intelligence Tools*, 13(04), pp. 961–979.

[Reed and Walton, 2004] Reed, C. and Walton, D. (2004). Towards a formal and implemented model of argumentation schemes in agent communication. In Rahwan, I., Moraitis, P., and Reed, C., editors, *First International Workshop on Argumentation in Multi-Agent Systems*.

[Reed and Wells, 2007] Reed, C. and Wells, S. (2007). Dialogical argument as an interface to complex debates. *IEEE Intelligent Systems Journal: Special Issue on Argumentation Technology*, 22(6), pp. 60–65.

[Reed et al., 2010] Reed, C., Wells, S., Budzynska, K., and Devereux, J. T. (2010). Building arguments with argumentation: the role of illocutionary force in computational models of argument. In *Proceedings of the Third International Conference on Computational Models of Argument (COMMA 2010)*.

[Reed et al, 2008] Reed, C., Wells, S., Rowe, G. W. A., and Devereux, J. (2008). Aif+: Dialogue in the argument interchange format. In *2nd International Conference on Computational Models of Argument (COMMA 2008)*.

[Reed et al., 2011] Reed, C., Wells, S., Snaith, M., Budzynska, K., and Lawrence, J. (2011). Using an argument ontology to develop pedagogical tool suites. In *Proceedings of the Third International Congress on Tools for Teaching Logic (TICTTL 2011)*.

[Saint-Dizier, 2012] Saint-Dizier, P. (2012). Processing natural language arguments with the <TextCoop> platform. *Journal of Argument and Computation*, 3(1), pp. 49–82.

[Toulmin, 1958] Toulmin, S. (1958). *The Uses Of Argument*, Cambridge University Press.

[Walton et al., 2008] Walton, D., Reed, C., and Macagno, F. (2008). *Argumentation Schemes*. Cambridge University Press.

[Walton, 1984] Walton, D. N. (1984). *Logical Dialogue-Games And Fallacies*. University Press Of America.

[Walton and Krabbe, 1995] Walton, D. N. and Krabbe, E. C. W. (1995). *Commitment in Dialogue*. SUNY series in Logic and Language. State University of New York Press.

[Wells, 2007] Wells, S. (2007). *Formal Dialectical Games in Multiagent Argumentation*, University of Dundee.

[Wells and Reed, 2012] Wells, S. and Reed, C. (2012). A domain specific language for describing diverse systems of dialogue. *Journal of Applied Logic*, 10(4), pp. 309–329.

[Wirth, 1977] Wirth, N. (1977). What can we do about the unnecessary diversity of notation for syntactic definitions? *Communications of the ACM*, 20(11), pp. 822–823.

[Wyner and Bench-Capon, 2007] Wyner, A. Z. and Bench-Capon, T. J. M. (2007). Argument schemes for legal case-based reasoning. In *Proceedings of Jurix 2007*, pp. 139–149.

[Wyner et al., 2011] Wyner, A. Z., Bench-Capon, T. J. M., and Atkinson, K. (2011). Towards formalising argumentation about legal cases. In *Proceedings of the 13th International Conference on AI and Law*, pp. 1–10.

[Yuan and Wells, 2013] Yuan, T. and Wells, S. (2013). Protocl: Specifying dialogue games using uml and ocl. In Grasso, F., and Green, N., and Reed, C., editors, *Thirteenth International Workshop on Computational Models of Natural Argument (CMNA13)*, pp. 74–85.