RECONSIDERING REPSTAT RULES IN DIALECTICAL GAMES

SIMON WELLS (EDINBURGH NAPIER UNIVERSITY) & MARK SNAITH (ROBERT GORDON UNIVERSITY)

PAPER PRESENTATION TO THE 22ND INTERNATIONAL WORKSHOP ON COMPUTATIONAL MODELS OF NATURAL ARGUMENT (CMNA'22)

CARDIFF 12TH SEPTEMBER 2022

OVERVIEW

- Problem/Criticism
- Background
- Elements of a solution
- The DGDL Eco-system
- Modelling Simultaneous Games
- Future Work & Discussion

PROBLEM/CRITICISM

- The prohibition of repeated statements during dialogue within formal dialectical games is important:
    - Tractability
    - Predictability
        - Particularly for agent communication/computational dialogue systems
- People however, tend to repeat themselves in conversation
- People however, tend to repeat themselves in conversation
- People however, tend to repeat themselves in conversation
- Why? Repetition to confirm, clarify, resolve an impasse

- Agent/computationally oriented games can be a poor fit for how people engage in dialogue, and vice versa

- Essentially two questions:
    - How can we be all things to all people/agents?
    - Does addressing the above this help us to push forward the study of dialectical systems?

BACKGROUND

- A lot of effort over the last ~50 years studying Hamblin-style dialectical systems:
    - Regulated systems in which two or more players, take turns, to make moves, by saying things
        - **Regulated:** rules specify who can say what, when.
            - **Descriptive:** Begin with examples of real dialogues and attempt to specify the underlying rules & convention that capture those dialogue (shedding)
            - **Formal:** Begin with sets of simple rules and generate dialogues that have desirable characteristics
            - NB. Descriptive & Constructive approaches
        - Many systems aimed at specific "types" of dialogue, specific areas of human activity, or specific dialogical behaviours
        - A key innovation from Hamblin's approach is the commitment store (a "persona" of beliefs held by the participants)
    - Intensive study during last ~20 years with advent first of agent/software interaction and more recently of machine learning/dialogue generation/conversational AI/Interfaces.

THE DIALOGUE GAME DESCRIPTION LANGUAGE (DGDL)

- DGDL is a simple, grammar based, language for describing the rules of dialogue games:
    - Expressive - account for a wide variety of dialogical behaviour
    - Consistent - produce coherent and cohesive game descriptions
    - Syntactically verifiable - a game description is checkable
- Games describe participants, turn structure, artefacts and storage, rules, and interactions
- A game described in DGDL is executed by a runtime
- Players take care of their own strategy & decide what to say - DGDL runtime then determines whether what is said is legal within the confines of the current game

```
system         : ( systemID '{' (game)+ '}' | game ) EOF;
systemID       : identifier;
game  : gameID '{'composition (rule)* (interaction)+'}';
gameID         : identifier;
composition  : turnStructure (roleList)? participants (player)+ (store)*;
turnStructure : '{''turns,' turnSize',' ordering (','maxTurns)?'}';
turnSize       : 'magnitude:' (number | 'single' | 'multiple');
ordering       : 'ordering:' (strict | liberal);
maxTurns       : 'maxturns:' (number | runTimeVar);
runTimeVar    : '$' identifier '$';
roleList        : '{roles:' role(',' role)+ '}';
role     : 'speaker' | 'listener' | identifier;
participants   : '{players,''min:' number',''max:' (number | 'undefined') '}';
player : '{player,''id:' (playerID | runTimeVar) (',' roleList)?'}';
playerID       : identifier;
store   : '{store,''id:' storeType',''owner:'storeOwner','storeStructure','visibility'}';
storeType      : identifier;
storeOwner   : playerID | '{'playerID(','playerID)+'}' | 'shared';
storeStructure: 'structure:'(set | queue | stack);
visibility       : 'visibility:'(publ | priv);
rule     : '{'ruleID' scope:'(initial | turnwise | movewise)','ruleBody'}';
ruleID : identifier;
ruleBody       : effects | conditional('&'conditional)*;
effects: '{'effect('&'effect)*'}';
effect  : effectID'('parameter(','parameter)*')';
effectID        : identifier;
parameter     : identifier | contentSet | contentVar | 'hello';
commitment : content | locution | argument;
content         : '{'(contentSet|contentVar)(','contentSet|contentVar)*'}';
contentSet    : upperChar;
contentVar    : lowerChar;
locution        : '<' moveID',' content'>';
moveID         : identifier;
```

```
argument      : '<'conclusion',' premises'>';
conclusion    : contentVar;
premises       : '{'contentVar(','contentVar)*'}';
storeName    : identifier;
requirements: '{'condition ('&'condition)*'}' | '{'requirements('||'requirements)*'}';
condition       : conditionID'('parameter(','parameter)*')';
conditionID   : identifier;
conditional    : '{''if' requirements 'then' effects ('elseif'requirements'then'effects)*('else'effects)?'}';
interaction     : '{'moveID',' content(','opener)?','rulebody'}';
opener           : string;
string  : '"'(upperChar|lowerChar|number|symbol)+'"';
rulebody        : (effects | conditional ('&'conditional)*);
strict   : 'strict';
liberal : 'liberal';
set       : 'set';
queue : 'queue';
stack   : 'stack';
publ    : 'public';
priv     : 'private';
initial  : 'initial';
turnwise        : 'turnwise';
movewise      : 'movewise';
upperChar     : UpperChar;
lowerChar      : LowerChar;
symbol           : Symbol;
identifier       : Identifier;
number          : Number;
Identifier       : UpperChar (UpperChar | LowerChar | Number)+;
LowerChar     : 'a'..'z' ;
Number          : '0'..'9' '0'..'9'*;
Symbol          :  ' ' | '?' | ',' | '.' ;
UpperChar     : 'A'..'Z' ;
NEWLINE       : ( ' ' | '\t' | '\r'| '\n' )+ {$channel=HIDDEN;};
```

A SIMPLE GAME DESCRIPTION

- Many games expressed in this kind of format
- Many games left to reformulate into DGDL
- Each new game is an opportunity to extend the description language itself
    - **What does this game codify that can’t be expressed in DGDL?**

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

REPSTAT RULES

- These are rules that govern when a statement can be repeated
- Generally prohibitive:
    - e.g. Hamblin 'H':
        - [A2] Statement S may not occur when S is a commitment of the hearer
        - [A4] Statement S may not occur when S is already a commitment of both speaker and hearer.
    - e.g. Mackenzie 'DC':
        - RRepstat: No statement may occur if it is a commitment of both speaker and hearer at that stage.
- In DGDL terms:
    - A restriction on when a move can be played. Expressed as a condition that must be met for a move to be legal
    - {Statement, {p},
{ if { inspect(!in, {p}, CS, speaker) & inspect(in, {p}, CS, listener) } then { store(add, {p}, CS, speaker) & store(add, {p}, CS, listener) } } };
- Have a good reason to exist:
    - Directly affects tractability of the dialogue in terms of the space of possible moves given the the participant's knowledge. Easy way to establish a termination condition.

A POTENTIAL SOLUTION

- Could carefully build a single game with rules to cover each situation:
    - Investigate all circumstances in which repetition might be legal, then either:
        - Formulate permissive rules as exceptions to a general prohibition
        - Formulate prohibitive rules as exceptions to a general permission
    - Not a bad idea, should be pursued, but painstakingly difficult and effortful, and likely to be incomplete
        - People & machines are treated equally (do we always want this?)
        - Listing circumstances by extension, is a perilously fragile approach

THE INSIGHT

- An entire game is a rigid and inflexible structure - you either play according to the rules of that game, or you aren't playing that game.
- If we have a more flexible game (that more closely matches real world practise) then we risk losing some of the computational benefits, but a more rigid game doesn't conform to the behaviour of everyday people.
- How can we both have our cake and eat it? (to coin a phrase)
- What if multiple games could be active simultaneously?
- If we accept that, then what effect does that have on the DGDL?

MULTIPLE SIMULTANEOUS GAMES

- We've conjectured that descriptive games are both generally closer to real world human interaction and are more permissive
    - can (not) do constraints
- & that formal games are generally better models for agent interaction and are generally more restrictive
    - must (not) do constraints
- Might a pair of descriptive and formal game thus model the inner and outer bounds of acceptable behaviour within a dialogue having diverse participants?

EXTENDING THE DGDL

- Required extensions to the DGDL are minimal
    - The system/shift infrastructure can be repurposed
    - The inner and outer games can both be described as distinct games within the wider system
    - The moves of each constituent game must be described so that they overlap (in effect creating one long *glissando*-style shift) and thus run concurrently.
- Small updates needed to DGDL runtimes:
    - The runtime must distinguish which effects to apply when two moves, e.g. one in the inner game and one in the outer game, both have the same label and conditions.
    - Our initial investigations suggest discriminating the human participants from the machine participants and applying either the outer game effects or the inner game effects respectively is sufficient.
    - But other algorithms might have interesting effects on the dialogue, e.g. blindly apply all moves whose pre-conditions are met (the default in baseline DGDL), where alternative moves are valid, prefer to apply the strict (inner) to the relaxed (outer) (or vice versa)

CONCLUSIONS

- We’ve proposed a novel model for addressing the question of dynamically handling the different dialogue styles of humans and machines
- We’ve considered this in the context of statement repetition but other dialogical contexts might be interesting/useful for modelling mixed initiative (e.g. statement relevance)
- DGDL enhancements necessary to support this are minimal (the problem isn’t altering the language but the runtimes and dependent implementations)

FUTURE WORK

- Multiple concurrent games represent a complication to existing dialectical game dynamics.
- If players have a choice of essentially the same move, but with different effects, what is the impact on strategy
- We've only considered the simplest case of two games running concurrently in which the available moves are the same in each game (although the requirements/effects of each move might differ)
  - But what DGDL changes are forced if the inner and outer games allow different overlapping, but distinct, sets of moves?

DISCUSSION

- Apologies for any "hand waving" - this is preliminary work that we wanted to share with our community - there are some aspects that haven't been finalised - there is a lot of work left to complete.
- The key takeaway is an innovation with regards to **dialogue rule dynamics** - why have one set of rules, when you can have two? (or more?)
    - and then apply each, or both to different participants?
- This could lead to wider exploitation of and better alignment between structured/formal approaches to dialogue, freeform/informal approaches to dialogue, and ML/data driven approaches leading to better conversational AI agents.

ACKNOWLEDGEMENTS

We’d really like to thank our (anonymous) CMNA reviewers.

There was some genuinely useful feedback for which we are very grateful

We didn’t get to address all of the interesting suggestions that you made in this paper but are gratified that they align so neatly with our existing ideas and plan of work…

… Perhaps we’re heading in the right direction?

THANKYOU