TOWARDS A DECLARATIVE APPROACH TO CONSTRUCTING DIALOGUE GAMES

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

PAPER PRESENTATION TO THE 21ST INTERNATIONAL WORKSHOP ON COMPUTATIONAL MODELS OF NATURAL ARGUMENT (CMNA'21)

HELD ONLINE 2ND & 3RD SEPTEMBER 2021

OVERVIEW

- Background
- Problem/Criticism
- Elements of a solution
- The DGDL Eco-system
- Deconstructing & reconstructing dialogue games using fragments
- Future Work & Discussion

BACKGROUND

- A lot of effort over the last ~40 years studying formal dialogue
  - As a structured interaction - dialogue/dialectical games
    - A game involving two or more participants/roles who take turns to make moves by saying things
    - (Deliberately excluding linguistic studies of dialogue for the moment)
    - Aimed generally at specific “types” of dialogues, specific areas of human activity, or specific dialogical behaviours
  - Intensive during last ~20 years with advent first of agent/software interaction and more recently of machine learning/dialogue generation/conversational AI/Interfaces.

PROBLEM/CRITICISM

"Dialogue games are too rigid/brittle for modern AI systems & don't have the flexibility of data-driven systems" – convenient straw person

- How can we address this?
- How can we bring together & re-contextualise the huge body of work on normative dialogues so that it can be easily & usefully exploited?

ELEMENTS OF A SOLUTION

- Perhaps a part of the solution lies literally in being less rigid - building and adapting dialogue games as needed to suit the dialogical context
- How?
    - By exploiting shifts & embeddings
    - By reusing existing game rules in different contexts
    - By making the game building process more declarative
- A starting place is to exploit the Dialogue Game Description Language (DGDL) eco-system then to adapt work on *desiderata* and *drosophila* to *test* the outputs of the system

THE DGDL ECO-SYSTEM,

- 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 that 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 -
  - **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)}}
}
```

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.
- However, the rules that make up a game can be separated out and rules from multiple games can be recombined to form new games
  - Video/Board game designers talk about "game mechanics" - shorthand for referring to how a specific aspect of the game play is codified in a given game but with the sense that mechanics are pluggable "if we could have the AI mechanic from X & the movement mechanic from Y then our new game will be excellent"
- We named these game parts "fragments"
- Perhaps new games can be specified based upon combining fragments in order to generate games that have specific attributes

DECONSTRUCTING GAMES INTO FRAGMENTS

- Not just pulling apart the rules of a game into constituent parts
- But trying to encapsulate & abstract design concepts (mechanics) from a given source game so it can be reused
- A fragment is a valid DGDL Left-Hand-Side (LHS) grammar rule
  - Starting anywhere within the grammar (not just at the start rule)
  - Extending through the grammar tree to valid terminal values
  - May be fully instantiated or partially abstracted
- Collections of fragments constitute a “dialogical behaviour context” - Many games encode behaviours across multiple moves so multiple fragments might need to be collated and applied together to be meaningful

CONSTRUCTING GAMES FROM FRAGMENTS

- Assuming a library of suitably abstracted fragments… This is what we envisage:
    - New games are developed in a specialised environment (could be GUI, TextUI, API, conversational interface)
    - The goal is generation of a novel DGDL description that can be tested and then executed on a runtime
    - Designers select the game(s) they want to construct (persuasion,deliberation, etc.) & a shift/embedding model if necessary
    - For each game an archetypal set of locutions is defined that conforms to each supported dialogue type; this is the *base game* for that dialogue type
    - Designer selects desirable properties to prohibit, permit, or prescribe - suitable fragments associated with those properties are then added to the DGDL description
    - Test the resulting description using desiderata and drosophila based approaches
    - Import tested DGDL description to a runtime and execute it

FUTURE WORK

- Building the library of fragments
- Range of declarative parameters
- Limits of automatic recombination
- Automated verification & checking of generated games
- Maintaining/Maximising modularity & interoperability with other argumentative formalisms and tools (ASPIC, AIF, Carneades, etc)

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 pieces to enables advances in dialogue game design, construction, testing, and deployment appear to exist.
- This could lead to wider exploitation of and better alignment between structured/formal 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.

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?