Dialog theory

Tom Rochette <tom.rochette@coreteks.org>

November 2, 2024 — 36c8eb68

0.1 Context

0.2 Learned in this study

0.3 Things to explore

- $Statement \subseteq Proposition \subset Conclusion$

1 Overview

In a discussion, each reply is either

- a new topic
- following the current topic, thus either continuing the existing chain or creating a new one
- replying to an old topic

In an IRC chat, one can use the nick of a user to reply to him, for example:

```
<johnSmith> tomzx: that's pretty nice!
```

Here are a few rules:

- We never reply to statements that were emitted after we emitted our reply (sequentiality of timeline)

Here are a few soft rules (not necessarily true):

- We generally reply to the last statement emitted by the person we're talking with
- We generally do not talk to ourselves
- Many statements emitted by the same emitter in short bursts may be in response to the same statements, or different statements
- An statement emitted after there has been silence for a considerable while generally implies this statement is the start of a new discussion thread

Things that can be done to ease processing:

- Merge all statements from an emitter that have been emitted sequentially (not interrupted by others)
  - This may make the association of future statements more difficult as it may be unclear what part of the merged statements is being replied to

1.1 Building up a context

To associate a sentence with its previous context, the following steps are accomplished:

- read the sentence and extract word cues
- determine the start of the discussion thread by observing various hints:
  - temporally close interlocutors
  - a period of inactivity potentially indicating a topic change

1.2 Discussion complexity

1.2.1 1 discussion, 2-people, 1 channel

1.2.2 1 discussion, n-people, 1 channel

1.2.3 n-discussions, 2-people, 1 channel

1.2.4 n-discussions, n-people, 1 channel

1.2.5 n-discussions, n-people, n channel

1.3 Agent Interaction Protocol

- Commencement rules
- A collection of locutions
- Combinations rules for the locutions
- A collection of commitments
- Combinations rules for the commitments
- Locution-commitment assignment rules
- Termination rules

Source: A Mathematical Model of Dialog, Mark W. Johnson, Peter McBurney, Simon Parsons

1.4 Dialog grammar

- Statements
- Claim/Proposition
- Proof
- Premise
- Conclusion
- Axiom
- Theorem
- Fact

1.5 Uncategorized

// No actors - Monologue

say('I want to create a new github project')
if (ask('Will it have many parts/subprojects?')) {
say('Create a new organization')
do('Create project in new organization')
} else {
say('Create project in personal account')
}

// With language specific verbs - Dialog(2)
tell = say
query = ask

tell('I want to create a new github project') // Tell comes from the first actor
if (ask('Will it have many parts/subprojects?')) { // Ask comes from the second actor
say('Create a new organization') // Say comes from the second actor
query('What should it be named?') // Query comes from the first actor
do('Create project in new organization')
} else {
say('Create project in personal account')
}

```
// With actors - Dialogue(3..n)

var alex = actor('alex');
var tom = actor('tom');

alex.say('I want to create a new github project')
if (tom.ask('Will it have many parts/subprojects?')) {
tom.say('Create a new organization')
tom.say('Create project in new organization')
//alex.do('Create project in new organization')
} else {
tom.say('Create project in personal account')
}

// Actor based - Dialogue (3..n)

tom(function() {
if (ask('Will it have many parts/subprojects?')) {
say('Create a new organization')
say('Create project in new organization')
} else {
say('Create project in personal account')
}
});

// Actions
tell
say
ask
do
wait

// Things it can do

Extract the list of say to create a list of options for a select
Extract prefixes to make a hierarchical list "I want to ...", "I have to..."
```

2 See also

3 References