Banking Application

- User can transfer, withdraw, and deposit money
- System: query TransType
- User: transfer
- System: query Amount
- User: 50
- System: query From
- User: checking
- System: query To
- User: savings
- System: bye

Overview

- Task
- Control Strategy
- Information State
- Actions

Just focusing on input and output speech acts, not ASR, parsing, & TTS.
Information State: Interface

- **NextMove**: What will the next move be (write only).
  - Allows system to make multiple utterances in a turn, such as informing an observation.
  - Used by control strategy to determine if to run action rules.
  - Set by action rules to indicate if the agent makes another utterance.

- **HaveTurn**: Does self have the turn (1 or 0).
  - Read only. Intended to be read by understanding rules.
  - Usefull in some cases where you need to distinguish if system made the move or not.

- **LastSpeaker**: Who spoke the last dialog move (self or other).
  - Read only. Intended to be read by understanding rules.

- **LastMove**: What was the last dialog move.

---

Overview

- **Task**
  - Information State
    - Actions
    - Control Strategy
      - Task
Meta Control Strategy

- For each understanding rule:
  - If preconditions hold, execute it and break.
- For each deliberation rule:
  - If preconditions hold, execute it.
- For each action rule:
  - If haveturn:
    - For each action rule:
      - If preconditions hold, execute it and break.
      - Note that no rule might apply, especially when system made the utterance.

- Order of rules does matter.
- Only first rule will be applied.

- Action rules: will determine what to say next based on IS.
- Will check each rule in turn. If its preconditions are true, will apply it.

- Deliberate rules: update IS based on other knowledge in IS.
- Note that no rule might apply, especially when system made the utterance.
- Only the first one will be applied.

- Understanding rules: will examine what was just said and
  update information state based on this.

- Will use these types of rules.
Private and Public Goals in Dialogue

- Need to track system's private goals and joint goals in dialogue.

- An approximation of joint goals:
  - QUD: Questions that have been asked, but not yet answered.
  - QUDspeakers: Speaker of each question on QUD.
  - Organized as a stack, as if one person asks a question, other might ask a question that needs to be answered first, in order to answer original question.

- QUD: Questions that have been asked, but not yet answered.

- Need to track system's private goals and joint goals in dialogue.

Overview

- Task
- Control Strategy
- Information State
- Actions
Information State: Domain Information

- Balance (checking)
- To
- From
- Amount
- Transaction Type

Private Goals

- Agenda drives the action rules
- Use agenda to inform user
- Use agenda to inform user of goals
- Query user to determine transaction type
- Used to inform user of goals
- Used to adopt goals
- Used to adopt goals: goals that are not mutually known and shared
- Used for short-term private goals of the system
- Organized as a stack
- Private goals: goals that are not mutually known and shared
- Agenda for short-term private goals of the system
- By having agenda, can have goals that will not be achieved in current turn
- If user asks query, user to determine transaction type
- If query user, query user to determine balance (checking)
- Agenda for short-term private goals of the system
- Use agenda to inform user
Actions for Domain Goals

SetupRule {A} deliberate setup
AddPre {$is(TransType) == ""}
AddPre {$is(QUD) == ""}
AddEff {lappend is(Agenda) {query TransType}}

SetupRule {A} deliberate transfer-to
AddPre {$is(TransType) == "transfer"}
AddPre {[lsearch $is(QUD) {query To}] == -1}
AddPre {[lsearch $is(Agenda) {query To}] == -1}
AddPre {$is(To) == ""}
AddEff {lappend is(Agenda) {query To}}

• Similar rule for transfer-from and transfer-amount
User Answer to Query

setuprule {a} understand other-answer
addpre {is(last-speaker) == "other"}
addpre {[lindex is(qud-speakers) end] == "self"}
addpre {answermatch [lindex is(qud) end] is(last-move)}
addeff {set is([lindex [lindex is(qud) end] 1]) is(last-move)}
addeff {set is(qud) [lrange is(qud) 0 end-1]}
addeff {set is(qud-speakers) [lrange is(qud-speakers) 0 end-1]}

proc answermatch {query ans} {
    if { [lindex query 0] != "query" } { return 0 }
    if { [lindex query 1] == "trans-type" } {
        if { ans == "transfer" || ans == "deposit" || ans == "withdrawal" } { return 1 }
    }
    if { [lindex query 1] == "from" || [lindex query 1] == "to" } {
        if { exists ::is(balance(ans)) } { return 1 }
    }
    if { [lindex query 1] == "amount" } {
        if { [string is integer ans] } { return 1 }
    }
    return 0
}
Queries on the Agenda

Use general understand rule for system's queries

```
   action ask
   deliberate setup
   deliberate none found
   deliberate self-ask
   understand self-ask
   System: query TransType
   deliberate transfer-amount
   deliberate transfer-from
   deliberate transfer-to
   deliberate other-answer
   deliberate none found
   deliberate self-ask
   action ask
   deliberate setup
   deliberate self-ask
```

Example Execution

```
   System: query Amount
   deliberate none found
   ```

   User: transfer
   ```

   System: query TransType
   deliberate transfer-amount
   deliberate transfer-from
   deliberate transfer-to
   deliberate other-answer
   deliberate none found
   deliberate self-ask
   action ask
   deliberate setup
   ```
Ending the Dialog

The Dialog Manager will look for "bye" as NextMove and end.

• Last rule

    {{set is(NextMove) bye}}

    

    \setupRule action bye

Ending the Dialog