ICT Dialogue Manager Tutorial: Session 4: Implementation in Soar

USC

ICT
INSTITUTE FOR CREATIVE TECHNOLOGIES

Outline

- Review of Soar basics
- Overview of Dialogue Information State
- Dialogue Processing Cycles
- Code overview

Implementation: SOAR (7.3)

- **Information state (Working Memory)**
- **Production rules**
  - Elaborate state rules
  - Operator Proposal rules
  - Operator application rules
- **Processing cycles**
  - Elaboration cycle
    - Invoke all rules that apply, change WM preferences
    - Calculate WM changes based on preference arbitration
  - Deliberation cycle
    - Choose operator
    - Input processing
    - Elaboration cycle+ (until no more rules apply)
    - Output processing

Example Elaboration Rule (inference.soar)

```
sp {top-ps*elaborate*task*belief*intend*true
   (state <s> ^problem-space.name top-ps
              ^agent-name <me>
              ^plan <task>)
   (<task> ^intend true
           ^responsibility <me>
           ^authorized yes)
   -->
   (<task> ^belief true)
}
```

Example Proposal Rule (nlu.soar)

```
sp {top-state*propose*operator*update-dialogue-state
  (state <s> ^name top-state
             ^speech-event-history <ss>)
  (<ss>    ^speech-input <si>)
  (<si> ^processed-by understand-speech
      -^processed-by update-dialogue-state)
   -->
   (<s> ^operator <o> + =)  ;# "indifferent" preference
   (<o> ^name update-dialogue-state
      ^speech-input <si>
      ^priority-class listen)
}
```

Example Operator Application Rule (nlu.soar)

```
sp {top-state*apply*operator*understand-speech*gesture-processed
 (state <s> ^name top-state
            ^operator <o>)
 (<o> ^name understand-speech
      ^speech-input <si>
      ^gesture <new>)
 (<si>     ^processed-by understand-speech)
 (<new> ^type gesture
        ^unprocessed yes)
 -->
 (<new> ^unprocessed yes -)
}
```

Main Aspects of Dialogue Context

- **Persistent State**
  - Social State
  - Lexicon
  - Ontology
  - Speech-event-history
- **Transient**
  - Conversation(s)
  - Social Planning
  - Participants
  - Task model
    - Causal-history, current-state
    - Plan,next-step
    - Task focus
  - Emotion Model
    - Emotional state
    - Coping strategies

Social State

- **^obligation + & ;#obligations to act**
  - ^type ^holder ^obligated-to
  - ^action ^deadline ^sanction
- **^commitment + & ;#committed to states of affairs holding**
  - ^type ^holder ^committed-to
    ^proposition ^sanction
- **^conditional + & ;#obligation or commitment if action**
  - ^type ^trigger ;# an action to check performance of
  - ^consequent ;# the resulting commitment or obligation
- **roles**
  - ^teammate
  - ^superior + & ;# agents superior to self
  - ^subordinate + &) ;# agents subordinate to self

Speech Event History

- **Record of all perceived utterances (regardless of placement in conversations)**
- **Structure**
  - Speech-input (record of processed objects)
  - Last (ordered list of most recent events)
    - Event=speech-input object
    - Previous= rest of list

Conversation Object

- ^active-participant +&
- ^overhearer +& ;#^participant is union of active-participant, overhearer
- ^mode ;# face-to-face, radio,
- ^last-utterance ;#stack of utterances part of conversation
- ^dialogue-history + & ;# speech-input objects
- ^last-mentions ;#history list of mentioned concepts and recency
- ^initiative ;# one of the active-participants
- ^turn ;# one of the active participants
- ^purpose
- ^topic
- ^topics
- ^QUD
- ^grounding +&;#set of cgus
- ^recent-cgus

Grounding Structures

- **^recent-cgus**
  - ^cgu & ;#one of last few cgus (first, plus first two on “rest” stack”)
  - ^first ;# most recent cgu
  - ^rest ;# stack of previous cgus
- **^grounding +&;#set of cgus**
  - ^initiator ;# one of active participants
  - ^responder ;# one or more of active participants
  - ^state ;# grounding state: S,F,D,1-4
  - ^dialogue-history + & ;#core speech acts
  - ^obligation + & ;#see social state for details
  - ^commitment + &
  - ^conditional + &
  - ^negotiation-stance + & ;#negotiation objects

Social Planning

- **Expectations**
  - Reactions to suggestions
  - Task-related moves
- **Agenda**
  - Partially ordered set of desired moves
  - Conversational Strategy
  - Motivations for strategy
- **Initiative variables**
  - Error-count
  - Irrelevant-count
- **Convenient clustering of obligations & negotiation stances**
  - Actual
  - Ungrounded
  - Conditional
  - Ungrounded conditional

Lexicon

- **Used to map ASR/NLU tokens to internal tokens (specialized to domain/task)**
- **Types:**
  - Actions, events
  - Attributes, entities,locations,times, values
- **Examples**
  - ^lexicon.entities.1sldr martin
  - ^lexicon.entities.1sldr 1sldr
  - ^lexicon.location.aa assembly
  - ^lexicon.location.aa area

Ontology

- **Purpose: Typing/structuring of concepts**
  - Selectional restrictions of semantic events
  - Group membership
- **Aspects:**
  - Event-types, events (calculated from task defs)
  - Group, human, location, vehicle (currently added manually)
  - Groups : link group to member (used in inference)
  - Type-slots: (e.g., recon)
    - Necessary slots (e.g, agent, event, time)
    - Possible slots:
      - (agent,destination,event,instrument,path,source,time)

Things-I-said

- **List of previous generated comgoals**
- **Structure:**
  - ^comgoal
  - ^previous
- **Used to update based on agent's own speech**
- **Used to repeat agent's previous utterance**
  - (currently broken)
- **Record of generation planning process**
  - (doesn't include realization trees)

NLP flags

- **Processing and testing mechanics, not part of theory**
- **Contents**
  - Generation testing results
    - Nlg-test
    - Nlg-said
    - generation failures
  - Use-nlu status

Dialogue Processing Cycles

- **Dialogue Inference**
  - Elaborate-state rules
- **Language Interp Stages**
  - ASR: 4 message types (speech in, timing/text out)
  - NLU: semantic interpretation(s)
  - Perception: integrated un(der) interpreted speech
  - SOAR Understand Speech Operator
  - SOAR Update Dialogue State Operator
  - SOAR Coping Focus Operator
- **Language Production Stages**
  - Output Speech proposal (desire)
  - pre NLG Output Speech operator (intention)
    - NLG Sub-state Operators
  - ✦ External NLG
  - ✦ String look-up
  - XMLify/vrExpress call
  - NVBG
    - Beavin
  - SBM
  - TTS
    - Festival
  - GameEngine
  - Post-NLG message passing/ callbacks

Communication Paths

- **Language Interpretation**
  - ASR:
  - Message format (human messages):
  - vrSpeech start <id> <speaker>
  - vrSpeech interp <id> <interp-id> <conf> <intonation> <surface>
  - vrSpeech finished-speaking <id>
  - vrSpeech asr-complete <id>

Understand Speech Operator

- **Triggered by complete speech input event**
  - Human (ASR,NLU)
  - Agent (vrSpeech,vrNLU)
  - Self (autonomous “event” from generation)
- **Main Purposes:**
  - Adjust semantic interpretation using soar-internal context (task, language and situation specific)
  - Reference resolution (pick out acts, entities, concepts mentioned)
  - Detect uninterpretable or ambiguous/underspecified content
  - Recognize dialogue acts

Pragmatics Processing

- **Determine what people are trying to do**
  - E.g., offer, assertion, question, be polite,…
- **Multiple resources**
  - Communication input
  - Context
    - Common knowledge
    - What has been said/done previously
    - Expectations of what will be said
- **Can use context to overcome ambiguities and errors**

Example: SOAR Understanding Rule

```
#recognize order from imperative from superior
sp {top-state*apply*operator*understand-speech*imperative-superior->order
  (state <s> ^name top-state ^social-state.status <ss>      ^operator <o>)
  (<o> ^name understand-speech      ^speech-input <si>)
  (<si> ^id <id>       ^speaker <speaker>     ^interpretation <i>)
  (<i> ^addressee <hearer>    ^mood imperative     ^semantics <sem> )
 -{(<i> ^token.lex << say >>    ^token.lex << again >>)}
  (<ss> ^<addressee> <saddr>)
  (<sadr> ^superior <speaker>)
-->
   (<i> ^speech-act <csa> + &)
   (<csa>  ^type  csa            ^action order
           ^actor <speaker>   ^addressee <hearer>
            ^content <sem>)
}
```

Update Dialogue State Operator

- **Triggered by Interpreted Speech Input**
- **Purpose:**
  - Update context by calculating effects of dialogue acts

Example: Soar Update rule

```
sp {top-state*apply*operator*update-dialogue-state*csa*order
  (state <s> ^name top-state          ^operator <o>)
  (<o> ^name update-dialogue-state      ^speech-input <si>)
  (<si> ^speaker <speaker>       ^interpretation <i>)
  (<i> ^conversation <c>      ^speech-act <csa>)
  (<c> ^grounding <cgu>)
  (<cgu> ^dialogue-history <csa>)
  (<csa> ^action order       ^content <sem>       ^addressee <addr>)
   -->
  (<cgu>  ^obligation <obl> + &)
  (<obl>  ^type obligation              ^holder <addr>
          ^obligated-to <speaker>        ^deadline asap
          ^sanction order          ^action <sem>)
}
```

NLG Approach

- **Asynchronous Communicative Goal Proposal**
- **Selection of Goal**
- **Content planning**
- **Realization: Hybrid Approach**
  - External Generator (GNLG, RNLG, ANLG, DNLG)
  - Hand-crafted prompts
    - Rapid-prototyping
  - Selection of detailed sentences plans
    - Emotional impact, natural expression
  - Generic case frames
    - Broad domain coverage
  - Template generation (discussion of emotions)

Selecting Acts to Perform

- **Considerations:**
  - The turn
  - Obligations to ground
  - Obligations to repair
  - Degree of understanding of prior utterances
  - (potential) obligations to address info-request
  - Beliefs about true answers

Output Speech Operator

- **Triggered by (successful) desire to speak**
- **Main Purposes**
  - Deliberation over how to achieve communication goal
  - Content planning
    - Sentence planning
  - Realization
  - Selection
  - Produce speech & wait for callback

Example Output Speech Rule

```
sp {top-state*propose*operator*output-speech*accept-obligation-to-act
  (state <s> ^name top-state  ^agent-name <me>  ^conversation <c>
              ^social-planning <sp>)
  (<sp>  ^my-potential-obligation <obl>)
  (<obl> ^obligated-to <other>        ^action <a>
       ^dialogue-state needs-discussion   ^relevant-party <me>
       ^plan-state << good considered-good not-in-coa >>)
  (<c>  -^turn <other>        ^grounding <cgu1>
       ^participant <me>     ^participant <other>)
  (<cgu1> ^dialogue-history <order>)
  (<order> ^action << order request >>        ^actor <other>
        ^addressee <me>        ^content <a>)
   -->
  (<s> ^operator <o> + =)
  (<o> ^name output-speech     ^priority-class respond
     ^conversation <c>       ^goal <b>)
  (<b> ^action accept
     ^type backward
     ^addressee <other>
     ^speaker <me>
     ^content <order>)
}
```

Example Output Speech Rule: take initiative

```
sp {top-state*propose*operator*output-speech*take-initiative-agenda-no-strategy
    (state <s> ^name top-state          ^agent-name <me>
               ^social-planning <sp>    ^conversation <c>
               ^social-state <ss>       -^plan-status update-needed)
   -(<c> -^turn <me>     -^turn *none*)
    (<sp> ^agenda <ag>      ^take-initiative yes)
    (<ag> -^strategy.strategy << delay negative >>    ^next <item> )
  -->
   (<s> ^operator <o> + = <)
   (<o> ^name output-speech    ^priority-class advance-task   ^agenda
  <item>)}
```

Dialogue Code Files Overview: Austin/DIALOGUE.config

- **coresteve**
  - General Dialogue
    - dialog-init.soar
    - inference.soar
  - Understanding & update
    - nlu.soar
    - reference.soar,
    - ref-candidate.soar
    - Expectations.soar
    - feedback.soar
  - Generation
    - nlg.soar
    - output-strings.soar
    - Initiative.soar
    - test-nlg.soar
  - Conversations and Dialogue acts (understand and update)
    - conversation.soar (also nlg)
    - csas.soar
    - grounding.soar
    - turn-taking.soar
    - negotiation.soar
- **Saso-EN-doctor-perez**
  - Domain-specific
    - Saso-en-dialogue.soar
    - Lexicon.soar
    - Saso-en-output-strings{-negative/positive}.soar