Finite State Models

- Behavior of system described as a set of nodes and transitions
- Tedious to add common functionality
- Can also allow arbitrary code at each node
- Repeat local clarifications, help

Overview

- Introduction
- Theory
- EDIS System
- GOdis
- Trinuki
- Comparison

CSE560 Class Trindi: © P. Heeman 2003
Example: Car Buying System

- If user does not care about power windows, might not care about power.
- When want to order prompts based on which past questions the user has cared about.
- Alternatives might all be automatic, so don't ask transmission type.
- Might want to order prompts based on which questions can best narrow down the alternatives.
- Might want to give information to user.

System prompts for values of a number of slots: year, make, etc.

Form-Filling Dialogue Manager

- Goal of system: have user fill in values of a (hierarchical) form.
- Behavior of system is declared in data structures associated with the form.
- Goal of system: have user fill in values of a (hierarchical) form.
- Big issue in AI is declarative versus procedural knowledge.
- Deterministic knowledge makes it easier to reason about.
- Difficulty in AI is declarative versus procedural knowledge.
- System does not know what actions need to be filled in.
- System determines its next move by examining what slots have been filled in.
- This approach is similar to VoiceXML, which is widely used in industry.
- This approach is similar to VoiceXML, which is widely used in industry.
What is "Information State"?

Definition:
- Represents what the system thinks is true
- What the system can do
- Next alternative

Next Alternative?

- Representing what the system thinks is true
- FSM and Form-filling approaches did not put much emphasis on
- What it now thinks is true
- What it says next
- What the system can do
- What the system thinks is true
- Need to characterize
- If certain condition holds, then do a certain action
- Represent system behavior as a set of if-then rules
Overview

Introduction
⇒ Theory

Comparison
EDISSytem
GoDiS
Thridge
Theory
⇒ Introduction

Key Issue: Update of Information State

- How what the system does is decided from the information state
- How what the user said is incorporated into information state
- deciding what to do next

Information state holds everything that system should use in
Part 1: Informational Components

- When information do we want to include in system?

- Static versus Dynamic
  - Static: knowledge that will not change
  - Domain: knowledge that is assumed to be known by both
  - System: knowledge that just the system might have

- Public versus Private
  - Private: knowledge that just the system might have
    - Might want to include static information, which will not change
    + Domain knowledge: all flight

- Mental State versus Dialogue History
  - Model the beliefs, intentions and obligations of system
  - Model the history of what has happened
  - Assumes parsing and semantic interpretation already done

Information State Theory of Dialogue Modeling

- Description of what information will be used (e.g. participants, beliefs, intentions, common knowledge)
- Formal representations of the above components (e.g. as lists, sets, typed feature structures, FOPC)
- Set of dialogue moves that will trigger the update of the information state
- Set of dialogue moves that will trigger the update of the information state
- A set of update rules. Covers how information state is updated given performed dialogue moves. Also includes selection rules that specify what system will do next.
- A set of update rules. Covers how information state is updated given performed dialogue moves. Also includes selection rules that specify what system will do next.
- An update strategy for deciding which rules (or(s) to select at a given point from the set of applicable ones. Might be as simple as pick the first rule that applies
Example Informational Components

- For information-seeking dialogue
- Private Information
  - includes beliefs of system
  - agenda of actions that system wants to perform in the dialogue
- Shared Information
  - shared beliefs
  - questions under discussion (QUD)
  - last dialogue move performed (lm)
- ISU approach does not specify which variables should be used, only that some set of variables should be used
  - Dialogue Designer should make sure that their set of variables is well-defined

Part 2: Formal Representations

- How should we model the information components?
  - What should be the semantic representation language?
    - frames, FOPC, ad-hoc
  - How should they be stored?
    - Set, list, queue, stack?
- Example:

```
PRIVATE : BEL : SET(Prop)
          AGENDA : STACK(Action)
SHARED  : BEL : SET(Prop)
          QUD : STACK(Question)
          LM : MOVE
```
Update rules are meant to encapsulate coherent bundles of change to the information state, given a particular theory of dialogue. Each rule consists of a set of applicability conditions and a set of effects.

- **Applicability Conditions**: Specify aspects of the information state that must be present for the rule to be applicable when the effects are triggered.

- **Effects**: Changes that will be made to the information state when the rule is applied (assuming that all conditions hold). These changes are the actual modifications to the dialogue state.

**Update Rules**

### Part 3: Dialogue Moves

- **Semantic Information**
- **Inform**
- **Request**
- **Acknowledge**
- **Other Speech Acts are used:**
- **Abstraction of what the user is saying**

**Complicating Issues:**

- Sometimes, a single utterance has multiple functions.
- What distinctions semantic interpretation can reliably make.
- What distinctions dialogue system needs to accomplish the task.

### Part 4: Update Rules
Other examples

• Answer top question on QUD

• Remove question from QUD once it has been answered

Example of Update Rules

- Rule for adding question to QUD if user move has been performed
  - Perform a raise move on the agenda
  - Push the question that is the center of both agenda items and the ask
    - user asks a question that system wanted to be raised
  
  - Rule has two conditions:
    - Ask move has been performed
    - Raise agenda item was the last agenda item to be raised

  - Effects:
    - Push the question that is the center of both agenda items and the ask
    - user asks a question that system wanted to be raised

Example of Update Rules
Discussion

• Interaction between choices for the components of information state: the conceptual notions, formal representations, dialogue moves, update rules, and update strategy
  - Choices for one component influence choices for another
  - Example:
    - Choices for update rules depends on update strategy
    - Choices for formal representation influences rules

Part 5: Update Strategy

• Need strategy for deciding which of the applicable rules should be applied
• Do not want to apply all rules, as might be contradictory
• Might have to redesign rules and information state to make update strategy work in the desired way such that the systems work as desired
• Which strategy for deciding which of the applicable rules should be applied
• Possible strategies:
  1. Take the first rule that applies (iteratively until no rules apply)
  2. Apply each rule (if applicable) in sequence
  3. Apply rules according to class
  4. Choose among applicable rules using probabilistic information
  5. Present choices to user to decide (for development modes)
Overview

Introduction

Theory

Companion
EDIS System
GoDiS
TrindiKit

Example

Sys: Where do you want to go?
Usr: Malvern

• Before exchange, system has an agenda item to raise the question about the user's destination.
• Update rule with this condition has effect of selecting an ask move.
• After system utterance, update rule will move question from agenda to QUD.

It is: Malvern
Sys: Where do you want to go?

• After user answer, rule for integrating user answers will check that answer matches top question on QUD, pop the question off.
• Update rule with this condition has effect of selecting an ask move.
TRINDI DME Architecture

- Information-state approach lends itself to implementation of the Dialogue Move Engine (DME) and some other stuff in terms of dialog management
- Call implementation a Dialogue Move Engine (DME)
- Information-state approach lends itself to implementation of moves and selecting moves to be performed
- Observation of moves and selecting moves to be performed
- Interpreton: determination of the input, which dialogue moves have been performed, updating in
- User Interface: receive input from and present output to the user
- Control: to wire together the other modules, either serially or in parallel
- Generation: links concepts of next move and produce the output
- and discourse-tracking aspects of a dialogue system
- DME (and some other stuff) forms the dialogue management
- Control to wire together the other modules, either serially or in parallel
Definitions of datatypes

Language for specifying update rules

Methods for accessing IS

Algorithm definition language for DME and control modules

Default modules for input, interpretation, generation & output

Multiple DMEs

- System might contain multiple DMEs
- One for selection (determine next move)
- One for update (update information state after act has been observed)

TrindiKit also provides...

• Definitions of datatypes
• Language for specifying update rules
• Methods for accessing IS
• Algorithm definition language for DME and control modules
• Default modules for input, interpretation, generation & output

Special interface such as a database, but looks just like part of IS through a coherent way's IS uses abstract data types, each permitting a specific set of operations to inspect the type and operations to change it. These are the building blocks of update rules, which can be used by other modules to inspect the type and operations to change it. Some parts of information state might actually be external to it.
GoDiS System

- Experimental dialogue system built using TrindiKit
- Uses fairly simple algorithms for control, update and selection
- Keyword-based interpretation and template-based generation
- Able to handle simple grounding phenomena
- Question accommodation
- Allows users to answer unasked but salient questions
- Distinguishes 8 dialogue move types: ask, answer, repeat, request repeat, greet, goodbye, thank and quit

Overview

- Introduction
- Theory
- EDIS System
- Comparison
- GoDiS
- TrindiKit
- GODIS System built using TrindiKit

© P. Heeman, 2003
Grounding

• As system says information, goes into both Shared and Tmp, until it is
  grounded explicitly by user repeating it, implicitly by user making next
  relevant move (answering question)
  - Alternated version: As system says information, just goes into Tmp, until
  it is not understood something or accept it

Grounding

-Shared: information that has been explicitly established during conversation
  - Stack of questions under discussion
  - Questions that have been raised and are currently under discussion (not yet
    answered) System's short term intentions for the next turn
  - Plan: the actions that are longer-term dialogue goals
  - Set of propositions which agent assumes for the sake of the conversation
  - Information about the latest move (speaker, move type and context)
  - Tmp: mirrors shared fields

Private versus Shared

• Private:
  - Tmp: mirrors shared fields
  - Can be changed during the course of conversation
  - Information that has not yet been grounded

-Shared:
  - Propositions system holds to be true
  - Agenda: system's short term intentions for the next turn
  - Plan: the actions that are longer-term dialogue goals

Task Accommodation

- Agent should infer the user's goal (trip to Miami) and update its plan.
- Usually, user states goal "I want to take a trip.
- System adopts plan to help user with trip (by setting plan variable)
- System can then use plan to ask questions about trip.
- System drops plan to help user with trip (by setting plan variable)
- Usually, user states goal "I want to take a trip.

Answer Accommodation

- Accommodation: what to do when user's response does not match structures on QUD.
- When user addresses question not explicitly raised in dialogue.
- Update rule removes action from plan, then processes answer similar to update rule for processing normal answers.
- Update rule moves plan action to QUD, so normal QUD rules can interpret.
- "Cheaper as possible" not on QUD.
- "as cheap as possible"
- "Well around April / some time there"
- "Which month do you want to go?"
- When user addresses question not explicitly raised in dialogue.
Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Resulting Information State

<table>
<thead>
<tr>
<th>Shared</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>32</td>
</tr>
<tr>
<td>33</td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>37</td>
<td>38</td>
</tr>
<tr>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>42</td>
</tr>
<tr>
<td>43</td>
<td>44</td>
</tr>
<tr>
<td>45</td>
<td>46</td>
</tr>
<tr>
<td>47</td>
<td>48</td>
</tr>
<tr>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>52</td>
</tr>
<tr>
<td>53</td>
<td>54</td>
</tr>
<tr>
<td>55</td>
<td>56</td>
</tr>
<tr>
<td>57</td>
<td>58</td>
</tr>
<tr>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>61</td>
<td>62</td>
</tr>
<tr>
<td>63</td>
<td>64</td>
</tr>
<tr>
<td>65</td>
<td>66</td>
</tr>
<tr>
<td>67</td>
<td>68</td>
</tr>
<tr>
<td>69</td>
<td>70</td>
</tr>
<tr>
<td>71</td>
<td>72</td>
</tr>
<tr>
<td>73</td>
<td>74</td>
</tr>
<tr>
<td>75</td>
<td>76</td>
</tr>
<tr>
<td>77</td>
<td>78</td>
</tr>
<tr>
<td>79</td>
<td>80</td>
</tr>
<tr>
<td>81</td>
<td>82</td>
</tr>
<tr>
<td>83</td>
<td>84</td>
</tr>
<tr>
<td>85</td>
<td>86</td>
</tr>
<tr>
<td>87</td>
<td>88</td>
</tr>
<tr>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>91</td>
<td>92</td>
</tr>
<tr>
<td>93</td>
<td>94</td>
</tr>
<tr>
<td>95</td>
<td>96</td>
</tr>
<tr>
<td>97</td>
<td>98</td>
</tr>
<tr>
<td>99</td>
<td>100</td>
</tr>
</tbody>
</table>

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.

Example

- User utterance answers two questions: how and to.
- User only asks about city of departure.
- Sys Welcome to the travel agency.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
- Sys What city do you want to go from.
- User answers.
- Sys flights to Paris.
GoDiS uses an update algorithm where different types of rules are applied at different stages of the update process. There are currently 6 rule types and 28 rules:

- **Refill**: Puts new actions on the agenda (8 rules)
- **Refill**: Handles question (and task) accommodation (5 rules)
- **Integrate**: Integrates the effects of the latest move (12 rules)
- **Accommodate**: Handles accommodation (3 rules)
- **Integrate**: Handles grounding (7 rules)
- **Grounding**: Performs database search (2 rules)
- **Store**: Stores current shared in private, tmp (1 rule)

### Update Algorithm

#### Update Rule for Accommodating Questions

After plan has been accommodated, apply the following to accommodate unspecified question.
Algorithm

• Overview:
  + Selection algorithm simply picks the first applicable rule.
  + Calls each module in turn in a serial fashion.

Control Algorithm

• Calls each module in turn in a serial fashion.
  + After grounding, the effects of the move are integrated into the information.
  + Overwriting a grounding rule is applied which ensures the latest move into
    + the information field on the latest utterance; the agent is refilled unless
      + database search
      + accommodate
      + integrate
  + If necessary, database searches are performed.
  + If necessary, database searches are performed.
  + Whenever a grounding rule is applied which ensures the latest move into
    + the information field on the latest utterance; the agent is refilled unless
      + If the latest speaker is not the user, the agenda is refilled unless
  + Otherwise, the current shared field is stored in previous item in case the next
  + Finally, if the latest move was performed by the user, the agenda is refilled.

Code

```c
if (latest_moves == failed) {
  repeat (refill)
} else {
  grounding
  integrate
  accommodate
  database
  repeat (refill)
}
```
EDIS System

Formulation of IS based on work by Poesio and Traum

- Common group part includes:
  - OBL: obligations of dialogue participants that have been performed
  - SCP: social commitments that participants have that propositions hold
  - DH: a dialogue history of acts that have been performed
  - COND: conditional statements that will establish obligations or commitments, given the performance of appropriately typed dialogue acts

Semi-public part is analogous to tmp in GoDiS

- a collection of discourse units (DUs), which represent coherent bundles of information that are grounded together

Private includes the intentions of agents being modeled

Comparison

EDIS System ⇐ GoDiS

Overview

Introduction
EDIS uses the same general pipelining of modules as GoDiS, however the update algorithm is a bit different.

Whenever a set of dialogue acts are placed in latest moves, the following algorithm is applied: applying a set of update rules in each step:

1. Create a new DU and push it on top of UDUs (and set CDU to this one, moving the old CDU to PDU).
2. Perform updates on the basis of backwards grounding acts, such as merging PDU.C with G for an acknowledgement.
3. If any other type of act is observed, record it in the dialogue history in CDU and apply the update rules for this kind of act.
4. Apply update rules to all parts of the IS which contain newly added acts.

Formal Representation

```plaintext
([Command] [State] [PDU] [CDU])
```

Semi-public includes:
- CDU for current and PDU, the previous one
- UDUs is a list of the DUs which may include the ones identified in PDU and/or
- Some-public includes:
  - Common info shown separated just for exposition purposes

Common Representation

```
+ LDU is a list of the DUs which may include the ones identified in PDU and/or
+ CDU for current and PDU, the previous one
```

```plaintext
[Command] [State] [PDU] [CDU]
```

Overview

• Introduction
• Theory
• TrindiKit
• GoDiS
• EDIS System

Control (Cont.)

Also a deliberation step, applied for each system turn, which leads to the system developing new intentions on the basis of insufficiently understood dialogue acts and intentions to perform complex acts of the form (if an intended act were performed + potential obligations that would result from conditions in the COND field of G + obligations + potential obligations) it and fulfilling any obligations and placed in the next moves interface variable. For following deliberation, dialogue acts are selected to fulfill any obligations, and placed in the next moves interface variable.
Form-filling: Keeps track of which slots are filled in.

Information State versus Form-Filling

- Can turn into an information state approach.

Information State versus Dialogue State

- Dialogue State (structured dialogues)
  - Dialogue is a set of states with transitions between them
  - Which transition is followed depends on what the user says
  - Amount of information is very limited
  - In any global variables that are used
  - Information is implicit in the state

- Information State (structured dialogues)
  - Information State: dialogue state, and global variables
  - Update rules: transitions from dialogue state approach
  - Information State approach: potentially unlimited information
  - Dialogue is a set of states with transitions between them
  - Amount of information is very limited
  - In any global variables that are used
  - Information is implicit in the state

- Update strategy: pick the single transition that applies