Natural Language Generation

Brent Ramerth
Introduction

• NLG: “process of constructing natural language outputs from non-linguistic inputs”
• Inverse of natural language understanding
• Naïve examples: canned text, template filling
Introduction

• NLG is about choices
  – Content selection
  – Lexical selection
  – Sentence structure
    • Aggregation
    • Referring expressions
  – Discourse structure
Architecture

DISCOURSE PLANNER

Knowledge Base

Communicative Goal

Discourse Specification

SURFACE REALIZER

Natural Language Output
Discourse Planning

• Text schemata
  – Texts tend to follow consistent structural patterns
  – Augmented transition networks used to plan expression of procedures
Discourse Planning

- Procedural knowledge base represented as a network

- **PROCEDURE**: Save Document [Side-effect: System saves document]
  - Procedural Sequence
    - Choose Save Option [Side-effect: System opens a dialog box]
    - Select Folder
    - Type Filename
    - Click Save Button [Side-effect: System closes a dialog box]
Discourse Planning

• Rhetorical relations
  – For texts that require more structural variety and cohesion
  – Rhetorical Structure Theory: analyzes relationships that hold between sentences
  – 23 rhetorical relations (e.g., Elaboration, Contrast, Purpose)
  – Rhetorical structure graphs
Discourse Planning

- Hierarchy of rhetorical relations

I love to collect classic automobiles. My favorite car is my 1899 Duryea. However, I prefer to drive my 1999 Toyota.
Surface Realization

• Discourse planner compiles a machine representation of the overall structure of the discourse unit (discourse specification)
• Surface realizer transforms discourse specification into sentences
• Two approaches: Systemic Grammar and Functional Unification Grammar (not covered here)
Surface Realization

• **Systemic grammar:**
  – Represents a sentence as a collection of layers of functions conflated with each other
  – Maps these functions onto grammatical forms

• **Different layers of functions**

<table>
<thead>
<tr>
<th>ss</th>
<th>mood</th>
<th>finite</th>
<th>predicator</th>
<th>object</th>
</tr>
</thead>
<tbody>
<tr>
<td>mood</td>
<td>subject</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>transitivity</td>
<td>actor</td>
<td>process</td>
<td>goal</td>
<td></td>
</tr>
<tr>
<td>theme</td>
<td>theme</td>
<td>rheme</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Interpersonal meta-function
- Ideational meta-function
- Textual meta-function
Surface Realization

• Generate:
  “The system will save the document”

  ( :process save-1
  :actor system-1
  :goal document-1
  :speechact assertion
  :tense future
  )

• Systemic grammar identifies what functions to insert, in what order:

<table>
<thead>
<tr>
<th>SS</th>
<th>mood</th>
<th>transitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>subject</td>
<td>finite</td>
</tr>
<tr>
<td>actor</td>
<td>process</td>
<td>goal</td>
</tr>
</tbody>
</table>
Surface Realization

- Recursive descent to fully specify functions:

<table>
<thead>
<tr>
<th>SS</th>
<th>system-1</th>
<th>save-1</th>
<th>document-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>mood</td>
<td>subject</td>
<td>finite</td>
<td>predicator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>object</td>
</tr>
<tr>
<td>transitivity</td>
<td>actor</td>
<td>process</td>
<td>goal</td>
</tr>
</tbody>
</table>

- Morphology and lexical insertion:

<table>
<thead>
<tr>
<th>SS</th>
<th>the system</th>
<th>will</th>
<th>save</th>
<th>the document</th>
</tr>
</thead>
<tbody>
<tr>
<td>mood</td>
<td>subject</td>
<td>finite</td>
<td>predicator</td>
<td>object</td>
</tr>
<tr>
<td>transitivity</td>
<td>actor</td>
<td>process</td>
<td>goal</td>
<td></td>
</tr>
</tbody>
</table>
Microplanners

- Avoiding monotonous text
- Referring expression selection
  - When to use pronouns
- Lexical selection
  - Choosing between different words for a concept
- Aggregation
  - Chunking knowledge base content into sentences by conjoining