Overview of KANTOO MT
(Analysis and Generation)

11-731 Machine Translation

KANT: System Architecture

- Modules
  - segmenter, morphology, parser, interpreter, filter
- Knowledge
  - tag database (DTD), lexicon, grammar, semantic model
- Levels of Representation
  - input string, syntactic f-structure, interlingua, ...

KANT Analyzer Data Flow

In Software Engineering, this kind of data flow is referred to as a transform flow or transform mapping

Analyzer Software Development
**Knowledge Types**

<table>
<thead>
<tr>
<th>Knowledge Type</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idiom_Dictionary</td>
<td>DMK</td>
</tr>
<tr>
<td>Irregular_Dictionary</td>
<td>DMK</td>
</tr>
<tr>
<td>Word_Dictionary</td>
<td>DMK</td>
</tr>
<tr>
<td>Morphematic Rules</td>
<td>Domo</td>
</tr>
<tr>
<td>Morphsemantic Rules</td>
<td>Domo</td>
</tr>
<tr>
<td>Domokeys</td>
<td>LR_State_Table</td>
</tr>
<tr>
<td>Generalizations</td>
<td>Ambig_Heuristic</td>
</tr>
<tr>
<td>Grammar</td>
<td>Interlingua_DB</td>
</tr>
<tr>
<td>LR_State_Table</td>
<td>Ambig_Heuristic</td>
</tr>
<tr>
<td>Ambiguity</td>
<td>Interlingua_DB</td>
</tr>
</tbody>
</table>

- **Map:** accesses stored knowledge associated with a key (e.g., term)
- **Rules:** transform a structure; usually PATRICK or C code

**System Architecture**

- **System Characteristics**
  - Interactive grammar checking (author's workstation)
  - Batch translation on server machines
  - Regular knowledge updates

**Segmentation**

- From SGML text to individual input sentences
- Separate "debris" from translatable text
- Break paragraphs into sentences
- Behavior controlled by separate tag data file & list of allowable abbreviations

**Preprocessing & Morphology**

- Canonicalization (e.g., case)
- Chunking (e.g., idioms, technical phrases)
- Identify possible root forms plus affixes (morphemes)
- Lexical lookup (filter out impossible root forms)
- Create/annotate lexical structure

**Syntactic Parsing**

- What types of knowledge?
- What levels of representation?
- Theory-driven vs. domain-driven systems
- KANT: non-deterministic, exhaustive LR parsing (Tomita)
- For each input, possibly multiple f-structures

**Semantic Restrictions**

- What constructions are ambiguous? What knowledge can be used to disambiguate?
- Top-down (rational) model
  - Every concept in the domain is fit into a complete, comprehensive domain model (hierarchy)
  - Extremely precise and detailed
  - Extremely expensive, too (CYC)
Semantic Restrictions [2]
- Bottom-up (empirical) model
  - Include only concepts & relations that are necessary to disambiguate
  - Create parent classes only to support generalization where needed
  - Less precise, more approximate
  - Much less expensive
  - Can give better cost benefit

Semantic Interpretation
- Integration with syntactic processing?
- How domain/language independent?
- KANT: Mapping f-structures to interlingua

Filtering Algorithms
- Automatic methods
  - General preference rules
  - Domain-specific preference rules
  - Phrasal vs. compositional parses
    "Oil flows through the bearing seal."
- Interactive methods
  - Author disambiguation
  - Input annotation
- KANT: automatic & interactive

More Details
- See the papers on the KANT web: http://www.lti.cs.cmu.edu/Research/Kant/
  - System Architecture: 1-4
  - Corpus Analysis: 5, 11, 14
  - Disambiguation: 7, 12, 24
  - Evaluation: 8, 17, 20, 23
  - Controlled Input: 13, 15, 21, 25

Target Generation

NLG Ingredients
- A representation of the input (probably not human-friendly)
- Knowledge of the domain
- Knowledge of the target language
- A human-friendly output format:
  - documents, reports, explanations, help messages, technical instructions, etc.
6 Basic NLG Tasks

1. Content Determination
   what information should be conveyed?
2. Discourse Planning
   order & structure of message set
3. Sentence Aggregation
   grouping messages into sentences
4. Lexicalization
   words & phrases for concepts, relations
5. Referring Expression Generation
   words & phrases for entities
6. Linguistic Realisation
   syntax, morphology, orthography

Typical 3-Module Architecture

1. Content Determination
2. Discourse Planning
3. Sentence Aggregation
4. Lexicalization
5. Referring Expressions
6. Syntax, Morphology, Orthography

Text Plans

- Common representation: tree
  - Leaf nodes = messages
  - Internal nodes = message groupings
- Simple text plans: templates OK
- Complex text plans: require full representation language (e.g., TAMERLAN, DIOGENES)

Sentence Plans

- Simple: templates (select & fill)
- Complex: abstract representation (SPL: Sentence Planning Language)

Example SPL Expression

(S1/exist :object (01/train :cardinality 20 :relations ((R1/period :value daily) (R2/source :value Aberdeen) (R3/destination :value Glasgow))))

There will be 20 trains to Glasgow

Content Determination

- Messages (raw content)
- User Model (influences content)
- Is Reasoning Required?
  - Find a train from Aberdeen to Leeds
    (It requires two trains to get there)
  - Deep Reasoning Systems
    - represent the user’s goals as well as any immediate query
    - utilize plan recognition & reasoning
Discourse Planning

- Structure messages into a coherent text
- Example: start with a summary, then give details
- Discourse relations, e.g.:
  - elaboration: More specifically, X
  - exemplification: For example, X
  - contrast/exception: However, X
- Rhetorical Structure Theory (RST)

Sentence Aggregation

- No aggregation (1 sentence/message)
- Relative Clause
  ...which leaves at 10am
- Conjunction
  ...and the next train is the express
- Combinations
  ...and the next train is the express which leaves at 10am

Lexicalization

- Choosing words to realize concepts or relations
- Example:
  (action/change (measure outside_temperature) (delta (quantity/deg_F -10)))
  The temperature dropped 10 degrees

Lexical Selection Rules

(*A-INGEST
 (AGENT *O-BOB)
 (PATIENT *O-MILK)) => "drink"

(*A-INGEST
 (AGENT *O-BOB)
 (PATIENT *O-CHOCOLATE)) => "eat"

Case Creation

- Additional structure is required to realize the meaning of the semantic representation

(*A-KICK
 (AGENT *O-JOHN)
 (PATIENT *O-BALL))

"John propelled the ball with his foot"

Case Absorption

- Word chosen to realize a semantic head also implies the meaning conveyed by a semantic role

(*A-FILE-LEGAL-ACTION
 (AGENT *O-BOB)
 (PATIENT *O-SUIT)
 (RECIPIENT *O-ACME))

"Bob sued Acme"
Referring Expression Generation

- Initial introduction
  A man in the park looked up
- Pronouns
  He saw a bird fly over
- Definite Descriptions
  The man covered his head with a newspaper

Fixing Robot Text

- Start [the engine], and run [the engine], until [the engine], reaches normal operating temperature
- Start [], and run [the engine], until [it], reaches normal operating temperature
- Second example introduces ellipsis and anaphora

Journalistic Style

“A dissident Spanish priest was charged here today with attempting to murder the Pope. Juan Fernandez Krohn, aged 32, was arrested after a man armed with a bayonet approached the Pope while he was saying prayers at Fatima on Wednesday night. According to the police, Fernandez told the investigating magistrates today, he trained for the past six months for the assault. If found guilty, the Spaniard faces a prison sentence of 15-20 years.”

(Brown and Yule, 1983)

Other Readings

- KBMT-89 book chapter on generation (algorithms & control)
- MT journal paper (acquiring MT knowledge sources)
- Paper on Interlingua design
- Paper on Turkish generation
- Paper on Chinese generation

Generator Software Development

- Defining the input /domain requirements
- Defining the output / quality requirements
- Functional specification
- Selecting a software architecture
- Case Study: The KANT Generator

Input / Domain Requirements

- Grain size: sentences, paragraphs, or texts?
- Errors: graceful degradation, no output, source?
- Controlled vs. general domains
  Is the output controlled?
- Definition by construction/corpus
Domain Analysis

• What source phenomena?
• Basic representational elements? (BNF for input representation)
• What combinations thereof? (hard to spec!)
• Derive input specification (e.g., IR Spec)

Existing Knowledge Sources

• Dictionaries (multi-lingual?)
• Bilingual, aligned corpora (e.g., Lonsdale’s BiKwik)

Output / Accuracy Requirements

• Lexical accuracy
• Grammatical acceptability
• Domain-specific phenomena
  – hard constructions (passive, reference, aspect, etc.)
  – complex combinations
  – tagging in the input (e.g., determiner stranding)

Requirements Specification

• Lexical, Structural, Morphological processes
• What levels of processing are necessary?
  – the “eight levels” model
• Knowledge sources required?
• Load/run/update characteristics?
• Support tools/processes?

KANT: Use of Existing Knowledge

• Parts database, some multilingual entries
• Translator’s glossaries / dialect issues

Output Quality: What’s good enough?

• No loss of meaning
• Little or no post-editing required
• Cost of MT + postediting should be less than human translation
KANT: Generator Architecture (1)

- Modules
  - Mapper, GenKit, Morphe, CODA
- Top-down, recursive algorithm
- Knowledge
  - lexical rules, structural rules, grammar, morph classes/rules, CODA rules

KANT: Generator Architecture (2)

- Levels of Representation
  - input string, syntactic f-structure, interlingua
- System Characteristics
  - non-interactive
  - batch translation on server machines
  - regular knowledge updates (sync with CTE)

Example Interlingua

*** Mapper: Input:

(*O-SCANNER
attribute
(*P-EXISTING
(degree positive))
(number singular)
(reference definite)
(standalone-phrase +))

Example F-Structure

*** Mapper: Output:

((agr ((gender masc) (number sg) (person 3)))
(cat det)
(gender masc)
(number sg)
(root el))
((non-phra-modifier ((cat adj) (root existente)))
(root escáner)
(type reg))

Linearized F-Structure

[((root "@cap")),
((agr ((gender masc) (number sg) (person 3)))
(cat det)
(gender masc)
(number sg)
(root el))
((agr ((gender masc) (number sg) (person 3)))
(cat n)
(gender masc)
(number sg)
(root escáner)
(type reg)).]
Morphology & Post-Proc

*** Morphooe: Input: (agr ((gender masc) (number pl) (person 3)))
       (cat adj)
       (change si)
       (gender masc)
       (number pl)
       (root existente)
       (type reg))
*** Morphooe: Output: existentes
*** Codafier: Output: Los escáneres existentes