Generation

Miriam Butt

January 2004

The Two Sides of Generation

1) *Natural Language Generation* (NLG) Systems which take information from some database and figure out how to present it to a human. Very little linguistics involved.

2) *Generation* as an inverse of *Parsing*. This is used mostly in the context of Machine Translation and involves quite a lot of linguistics (morphology, syntax, possible also semantics and discourse).

Natural Language Generation

The discussion here is based mainly slides from Robert Dale and on Reiter and Dale (2000).

Definition: NLG is the process of deliberately constructing natural language text in order to meet specified communicative goals.

Architecture: see handout.

Natural Language Generation

Example of NLG: Winograd's 1971 SHRDLU (see handout, pp. from Haugeland).

This is a problem solving AI system which has an internal data base of blocks in certain configurations. SHRDLU mimics a robot who knows how to perform certain actions (basic Prolog-style reasoning) and also can answer questions (interact with a human). But, no real linguistic structures are needed for this (cf. Eliza).

Natural Language Generation

Example of NLG: The famous ELIZA program (**emacs demo**).

See handout for a sample dialog and the Prolog code that generates the dialogs. There is no real linguistics involved, all that is done is a predefinition of certain utterances. I.e., you create *templates* in which lexical items can be plugged into dynamically.

Natural Language Generation

Example of NLG: FOG, a weather forecast system (bilingual in English and French), see handout.

This system takes some raw metereological data and composes weather forecasts. Again, mainly templates are used, but some (non-linguistic) decisions as to how to bundle information must be made.

In addition, some *discourse planning* (faintly linguistic) is involved.

Natural Language Generation

Example of NLG: the toy implementation of WeatherReporter uses a document (or discourse) plan to organize its information (see handout of R&D 82-85).

Again, the text is basically “canned”.

Generating with Grammars

Another approach to generation is to use a linguistic grammar to produce well-formed sentences.

Some Syntactic Formalisms which support Generation:

TAG (Tree Adjoining Grammars)

HPSG (Head-Driven Phrase Structure Grammar)

LFG (Lexical-Functional Grammar)

Generation in LFG

Basic Idea: The grammar should be bidirectional: if you have written a parser, you should be able to use the grammar in reverse for generation. This is not trivial.

Basic Input to Generator: an f-structure (Prolog format)

Some papers on the topic: Wedekind (1988), Momma and Dorre (1987). Shemtov's (1997) dissertation is the basis for the current XLE implementation, see Wedekind and Kaplan (2012) for the most recent advance.

Generation in LFG

Some papers on the topic: Wedekind (1988), Momma and Dorre (1987). Shemtov's (1997) dissertation is the basis for the current XLE implementation.

Basic Idea: Any LFG grammar should be bidirectional: if you have written a parser, you should be able to use the grammar in reverse for generation. This works, but is not trivial.

Basic Input to Generator: an f-structure (in Prolog format)

The English Grammar

- Uses the same grammar (bidirectionality)

- Uses OT (Optimality Theory) to control certain (unwanted) parses such as mismatched subject-verb agreement or optional punctuation (either prefer to have it or prefer it is gone).

- Uses a different tokenizer to make sure only to generate a single space between words, etc.

- Can generate either from an f-structure, or directly from an input sentence.

English Generation with XLE

Generation is not simply copying:

Falling, pull it. → *Pull it, falling.*

The boat sailed into the harbour. → *... harbor.*

Gaining Distinctions (ambiguity):

The monkey saw the cat with the telescope. →

With the telescope...

The monkey saw...

Loosing Distinctions:

To disappear pull it.

Pull it to disappear. → *Pull it to disappear.*

XLE

English Generation with XLE

Generating from Underspecified Input: XLE allows you specify what kind of information you might want to throw away from your f-structure.

For Example: you could decide to generate from an f-structure without its Tense/Aspect information (see documentation).

John sleeps → All Tense/Aspect Variations

(*John is sleeping*, *John slept*, etc.)

References

Momma, Stefan and Jochen Dorre. 1987. Generation from f-structures. In Ewan Klein and Johan van Benthem (eds.) *Categories, Polymorphism, and Unification*, 147-167.

Reiter, Ehud and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge: Cambridge University Press.

Shemtov, Hadar. 1997. *Ambiguity Management in Natural Language Generation*. PhD thesis, Stanford University.

Wedekind, Jürgen. 1988. Generation as Structure Driven Derivation. *Proceedings of COLING 1988*, 732-737. Budapest.

Wedekind, Jürgen & Kaplan, Ron. M. 2012. LFG Generation by Grammar Specialization. *Computational Linguistics* , vol 38, no. 4, pp. 867-915.