Abstract

We report on a head-driven way to generate a language-specific representation for a language-independent conceptual structure. With a grammar oriented towards conceptual rather than phrasal structure, the approach shows some advantages over previous works in head-driven generation. It is particularly suited for multilingual generation systems where language-independent representations and processes should be maintained to a maximum extent. We briefly sketch the architecture of our Genie system based on some results of an analysis of a technical manual for a gearbox.

1 Introduction

The Genie system explores a way to rationalize multilingual production of technical documentation. The system is semi-automatic in that the user designs an interlingual text specification describing content and form for a document. Genie constructs the document in the desired languages as modelled by the specification, matching contents to a knowledge base, constructing categories, and forming sentences according to combinatory rules.

The paper focusses on generation of language-specific categories from language independent conceptual structures.

2 The Document Analysis

We have chosen a 110-page manual, English ([3]) and Swedish ([8]), of the truck gearbox R1000 to analyse. The manual is for expert servicemen and shows the design, function, and service instructions.

The manual communicates some different kinds of domain information. We choose here to concentrate on the following two:

- Static information (i.e. what something is). Examples:

  (1) The R1000 is a gearbox. (2) The gearbox has nine forward gears. (3) The gearbox is mechanically operated.

  (1) R1000 är en växelläda. (2) Växellädan har nio växlar framåt. (3) Växellädan manövreras mekaniskt

- Processive information (i.e. what something does). Examples:

  (4) The purpose of the inhibitor valve is to prevent inadvertant shifting of the range gear when a gear in the basic box is in mesh. (5) The inhibitor cylinder prevents inadvertant shifting in the basic box when range shifts are being carried out.

  (4) Spärrventilen har till uppgift att förhindra växling av rangeväxeln när någon av växlarna i baslädan ligger i ingrepp. (5) Spärrcyllindern förhindrar växling i baslädan när växlingen med rangen sker.

The text can be broken down into approximately sentence-sized units, each one communicating a piece of information considered true in the domain. We observe a tight correspondence between the kind of information and its textual realization. The carefully defined terminology not only determines words, but their combinations as well.

The text structure follows from conventions of language use for efficient communication about the domain.
These findings are in line with the issue of domain communication knowledge (Kittredge [7]). Rösner and Stede ([9]) distinguish similarly between the macro and micro structure of texts. The architecture of Genie is built up around the division of sentence and text structure; the user incorporates the conventions in the specification while Genie provides the terminological definitions.

The English and Swedish versions of the manual align at sentence level. Genie can cope with semantically non-equivalent sentence pairs, but not the very rare ones differing in content. Nevertheless, the documents correspond nicely compared to the difficulties Bateman reports ([1]) on a study of medical didactic texts. Grote and Rösner ([5]) have studied car manuals for the TECHDOC system, and they observe a close correspondence.

We have employed Functional Grammar (FG) (c.f [6]) as a principal analysis tool to developing representations for domain and language.

3 Domain Representation

Domain representation is based on conceptual structures (Sowa [11]) and the transitivity structure of FG. Concept nodes are typed in an inheritance network. We follow Sowa’s definition and notation of conceptual graphs.

Next, we sketch how static and processive information are represented as facts, called aspects and transitions, respectively, in the knowledge base.

3.1 Aspects

An aspect contains a simple conceptual graph where an object has an attributive relation to a value. We define the is-a link as attributive and the type becomes the value. Sentence (1) and (2) are:

\[
\begin{align*}
[r1000] & \rightarrow \text{(iso) \rightarrow \text{[gearbox]}} \\
[r1000] & \rightarrow \text{(f-gears) \rightarrow \text{[f-gears:coll{f1, f2, ...}\@9]}}
\end{align*}
\]

Both aspects happen to be close to their linguistic realizations, which is not necessarily always the case.

3.2 Transitions

A transition is a concept trans with three relations, pre, means, and post. means has an event as value. pre and post hold circumstances that obtain before and after the event has occurred.

An event carries mandatory, e.g. actor, goal, and peripheral role relations, e.g instr to other objects. We can differentiate roles into subtypes, e.g i-instr inhibits the event.

A circumstance can be: (i) a state characterized as a setting of some variable parameter. An example is in the aspect for sentence (4):

\[
\begin{align*}
\text{[trans]} & \rightarrow \text{[basic-box-gears:disj{\*}] -} \\
\text{(pre)} & \rightarrow \text{[basic-box-gears:disj{\*}] -} \\
\text{(in-mesh)} & \rightarrow \text{[\*]} \\
\text{(means)} & \rightarrow \text{[range-shifting] -} \\
\text{(i-instr)} & \rightarrow \text{[inh-value]}
\end{align*}
\]

(ii) As an event, exhibited by sentence (5):

\[
\begin{align*}
\text{[trans]} & \rightarrow \text{[gen-dur-pre] \rightarrow [trans] -} \\
\text{(means)} & \rightarrow \text{[range-shifting] -} \\
\text{(means)} & \rightarrow \text{[basic-box-shifting] -} \\
\text{(i-instr)} & \rightarrow \text{[inh-cyl]}
\end{align*}
\]

Sub-events have their own transitions as value for pre and post, which allows us to link events together. gen-dur-pre is a version of pre used to give a meaning to “... being carried out”.

Transitions are more powerful than what has been outlined here. Much of their internal temporal constituency, complex parameters, lambda-abstractions, and different kinds of constraints have been left out for clarity.

4 Linguistic Representation

This section describes how Genie derives categories for a fact, as part of generation. We first describe English categories briefly.

4.1 Categories

Categories are expressed in a language of typed feature structures. We define how categories can be formed, their different types and content.

Construction of categories are inspired by modern Categorial Grammars (CG), such as UCG (c.f [12]), but differ in some respects. The set of categories C is defined recursively, (i) Basic categories ∈ C. (ii) If A and B ∈ C, then the complex category A|B ∈ C.

The differences from CG are (i) the association of categories to facts and concepts, and (ii) complex categories are non-directed.

Categories compose using the reduction rule to unify:

\[
A|B, B \Rightarrow A
\]

Categories are expressed as typed feature structures (tfs) (c.f Carpenter [2]). a(name) denotes the set of attributes the type name carries, and s(name) the immediate subtypes. cat is the root with a(cat) = {}, s(cat) = {xcat, bcat}. xcat is the | operator. bcat are the basic categories, a(bcat) = {fb, st}, s(bcat) = {lcat, pcat}. lcat and pcat are the lexical and phrasal categories. The attribute fb holds some feature bundle, rooted at fb and named appropriately, e.g np-fb, n-fb, agr-fb. st has a FG
mood-structure to hold subcategories. A peat has a certain tfs under the type st to encode the structure, while a lcat has a pointer into a surface lexicon. s-st is the structure for clauses. Elements are coded as attributes, e.g. subj, fin, compl etc.

4.2 Conceptual Grammar

Facts are associated to categories composed of those obtained from the conceptual constituents. The grammar rules state that a particular domain type corresponds to a category with certain combinatorial properties. If violated, the rule cannot derive an adequate category for the fact. Concept nodes are associated to a number of categories as defined by lexical rules.

We call this a conceptual grammar, since it is tied to conceptual rather than phrase structures. The rules are language independent as the linguistic material is effectively hidden within the basic categories. Rules have the following notation:

\[ <\text{head}> \text{ when } <\text{body}>. \]

\( <\text{head}> \) carries an association of the general form \( cs \mapsto \text{cat} \), where \( cs \) is a conceptual structure, and \( \text{cat} \) is the category. \( <\text{head}> \) holds whenever all constraints in \( <\text{body}> \) hold\(^1\). Help associations (arrow with a symbol on top) support \( \Rightarrow \) with extra material. We describe rules for atoms, objects, aspects and transitions.

4.2.1 Atoms and Objects

Atoms have a rather simple and direct association:

\[ [\text{mechanical}] \mapsto a[st:\text{mechanical}] \]
\[ [9] \mapsto \text{def}[fb:det-fb[agr:agr-fb[numb:pf]]} \]
\[ st:n=9] \]

The type of category depends on how it will be used, but should be basic. The examples are typical.

The object R1000 gives “a gearbox” in:

\[ [\text{r1000}] \mapsto \text{cnp}[fb:np-fb[agr:Agr=agr-fb[numb:sg, pers:3rd] spec: indef]}
\[ st:np-sf[n:[fb:n-fb[agr:Agr] st:gearbox]]} \]

There are potentially many alternative associations. Lexical choice is not addressed in this paper, although we recognize its necessity in generation systems.

4.2.2 Aspects

The category for the relation in an aspect is seen as a function of the categories for the two concepts. The grammar rule for aspects fetches and applies the function. A relation \( \text{operation} \), as in the aspect for sentence (3), has a category \( s[\text{np}]:a \):

\[
[\text{operation}] \mapsto [\text{st}:s-st[subj:Subj}
\[ \text{fin:}v[fb:v-fb[pass:+, agr:Agr=agr-fb]]}
\[ \text{pred:}v[st:operation]}
\[ \text{compl:}\text{Comp}]] |}
\[ \text{Subj=}np[fb:np-fb[agr:Agr]] |}
\[ \text{Comp=}a[fb:a-fb[adv:+]]]
\]

The rule says that one category should fill the compl element as an adverbial, and another to become an np in the subj element. Note the subject-verb agreement.

The aspect rule simply reduces the relation category with the categories obtained from the concepts:

\[ O=[\text{concept}] \mapsto R=(\text{rel}) \mapsto V=[\text{concept}] \mapsto A \]
\[ \text{when} \]
\[ R \mapsto A=\text{cat}|B=\text{cat}|C=\text{cat}, V \mapsto C, O \mapsto B. \]

An aspect is matched to the right hand side of the head to bind the variables O, R and V. The rule proves the following category for sentence (3):

\[ [\text{r1000}] \mapsto (\text{operation}) \mapsto [\text{mechanical}] \mapsto s[st:s-st[subj:np][fb:np-fb[agr:Agr=agr-fb[numb:sg] spec: indef]]}
\[ st:np-sf[n:[fb:n-fb[agr:Agr] st:gearbox]]} \]
\[ \text{fin:}v[fb:v-fb[pass:+, agr:Agr=agr-fb]]}
\[ \text{pred:}v[st:operation]}
\[ \text{compl:}a[fb:a-fb[adv:+] st:mechanical]]
\]

4.2.3 Transitions

Associations for transitions are more complex, but still compositional. The idea is to get a category for the event and reduce it with all roles to obtain a basic category. This is reduced with the transition type category and with those for pre and post relations and values.

The association for \( \text{trans} \) is defined by the rule:

\[ \text{Trans}=[\text{trans}] -
\]
\[ (\text{means}) \mapsto \text{Ev}=[\text{event}] \]
\[ \text{Pre-R}=(\text{pre}) \mapsto \text{Pre-C}=[\text{concept}]
\]
\[ \text{Post-R}=(\text{post}) \mapsto \text{Post-C}=[\text{concept}]
\]
\[ \mapsto \text{Res when} \]
\[ \text{Trans} \mapsto \text{Res1}=\text{cat}|\text{Event}=\text{cat}
\]
\[ \text{Pre-R} \mapsto \text{Res2}=\text{cat}|\text{Res1}|\text{Pre}=\text{cat}
\]
\[ \text{Pre-C} \mapsto \text{Pre}
\]
\[ \text{Post-R} \mapsto \text{Res}|\text{Res2}|\text{Post}=\text{cat}
\]
\[ \text{Post-C} \mapsto \text{Post}, \text{Ev} \mapsto \text{Event} \]

\(^1\)Like a Prolog rule.
The transition is matched to bind variables in the head. The category constrains the category in the pre to be a progressive-s. The rule for events basically looks like:

\[
\text{EV} = \{\text{event}\} - \\
(\text{mrel}) \rightarrow \text{OM}_1 = \{\text{concept}\} \\
\ldots \\
(\text{mrel}) \rightarrow \text{OM}_n = \{\text{concept}\} \\
\rightarrow \text{PR}_1 = (\text{pref}) \rightarrow \text{OP}_1 = \{\text{concept}\} \\
\ldots \\
\rightarrow \text{PR}_{m} = (\text{pref}) \rightarrow \text{OP}_{m} = \{\text{concept}\} \\
\Rightarrow \text{RES} = \text{cat when} \\
\text{EV} \rightarrow \text{PCAT}_0 = \text{cat} | \text{ARG}_n = \text{cat} | \ldots | \text{ARG}_1 = \text{cat} \\
\text{for } i = 1 \ldots m \text{ do } \text{OM}_i \rightarrow \text{ARG}_i \\
\text{for } j = 1 \ldots m \text{ do } \text{PR}_j \rightarrow \text{PCAT}_j \rightarrow \text{cat} | \text{PCAT}_{j-1} = \text{cat} | \text{ARG}_j = \text{cat} \\
\rightarrow \text{OP}_j \rightarrow \text{ARG}_j \\
\text{RES} = \text{PCAT}_m
\]

The event category reduces with the mandatory role values to reveal the innermost result category for the event. It will then reduce with the peripheral roles.

An example of an event category carried by \[\text{VPM}\]

\[
[\text{lock}] \rightarrow s = \{s: \text{st} \rightarrow \{\text{subj: SUBJ} \rightarrow \{\text{fin: \{fb: \text{v-fb} | \text{agr:AGR, pass:-}\}} \rightarrow \{\text{pred: \{st: lock\}} \rightarrow \{\text{compl:OBJ}} \} \} \} \} | \\
\text{SUBJ} = \{\text{np: np-fb} | \text{agr:AGR = agr-fb}} | \text{OBJ} = \{\text{np}
\]

4.3 Discussion

The conceptual grammar is a semantic-head grammar, where the semantic head is the top node of the graph a rule analyzes. The grammar processor is a plain Prolog resolution. It behaves as the standard semantic-head driven generator (SHDG) (Shieber et al [10]) does when all nodes are pivots, i.e. a purely top-down manner. SHDGs in general are quite different from ours in the way knowledge is organized. They follow the structure of categories in grammars that are more suitable for parsing, i.e allowing content-less words but not word-less contents. Hence, there is an asymmetry between compositionality of words and semantics (Dymetman [4]). A content-less word can potentially occur anywhere in the output string and a generator must consider this to terminate gracefully. Problems of ensuring coherence and completeness degrade efficiency further. Our generator resembles a parser to a large extent, having a conceptual structure instead of a string to work on. As such, it is free from the problems and can potentially benefit directly from many research results in parsing technology.

The rules are designed to work on any language, thus lessening the burden when adding more linguistic support. More rules have to be written only when new kinds of facts are added to the knowledge base, to account for their structures. We do not need a reachability relation, as the problem of goal-directedness in generation is achieved by doing clever choices of categories in lexical rules.

The relations between domain types and categories are similar to the semantic type assignments in classic CGs. Our version is more flexible as a consequence of the type system.

Genie is in an experimental state (about 20 aspects and 10 transitions), but has proven feasibility of the issues discussed in this paper. It is less competent in lexical choice and the combinatory grammar. Development is continuing in the Life environment.

References


