GRASP: Grammar-based Language Learning

Peter Ljunglöf

DART: Centre for Augmentative and Alternative Communication (AAC) and Assistive Technology (AT)
and Språkbanken, Department of Swedish Language, University of Gothenburg
peter.ljunglof@gu.se

Abstract

We are developing a pedagogical tool to support language learning and training for children with communicative disabilities. The system has a graphical interface, where the user can move, replace, add, and in other ways modify, words or phrases. The system keeps the sentence grammatical, by automatically rearranging the words and changing inflection, if necessary. In this way we hope that the system stimulates the child to explore the possibilities of language.

1. Introduction

In the GRASP[1] project, financed by Sunnerdahls Handikappfond, we are developing an interactive system for Computer Assisted Language Learning (CALL) (Davies, 2010). There are two intended target groups: one is children and adults trying to learn another language; another group is persons with communicative disabilities who are learning to read and write in their first language.

The idea is that it will work as an interactive textbook, where the user can read different texts (just as in a traditional textbook) but also experiment with and modify the texts. The system will be divided into modules dealing with different linguistic features, e.g., inflection, word classes, simple phrases and more advanced constructions. The modules can be used on their own, or can be combined for more advanced training.

The texts are stored in an internal grammar format which makes it possible to transform sentences interactively, while still keeping them grammatical. The underlying grammar is multilingual, which is useful not only for second language learning, but also for first language learning for persons with communicative disorders, since words and phrases can be interpreted in a symbol language such as Blissymbolics.

The system has a graphical user interface, where each word acts a kind of icon that can be clicked, moved, replaced, or modified in other ways. When the user moves a word to a new position, or changes the inflection of a word, the system automatically rearranges the other words and changes inflection so that the sentence stays grammatically correct.

2. System description

In this section we describe the final GRASP system, which is currently under development. Note that all features are *not* currently implemented (as of August 2010).

As the basic component we are using Grammatical Framework (GF) (Ranta, 2009b), a modular and multilingual grammar formalism. On top of this we build the graphical interface which the user interacts with. As a glue between the grammar and the interface, we implement an API for modifying syntax trees using linear constraints and a tree similarity measure.

2.1 Ready-made texts

The system will contain a number of texts that the user can read and experiment with. The texts are stored as GF grammars which makes them possible to modify in a grammatical way. Since GF is multilingual, the texts can be linearized in parallel for several languages. This can be useful for second language learning, as the system can display the text in the user's first language in parallel. Multilinguality is also useful for first language learning, e.g., by displaying the parallel text in a symbol language such as Blissymbolics.

2.2 Graphical interaction

The words in the example texts are icon-like objects which can be clicked on, moved around and deleted. If the user clicks on a word, a context menu appears consisting of similar words, such as different inflection forms, or synonyms, homonyms, etc. When a new word form is selected from the menu, it replaces the old word, and if necessary, the nearby words are also modified and rearranged to keep the sentence grammatical.

The user can move a word to another position in the sentence, and the system will automatically keep the sentence grammatical by rearranging and change inflection, if necessary. Words can be deleted from the sentence by dragging them away. The user can also add or replace words by dragging new words into the sentence. All the time, the sentence will adapt by rearranging and changing inflection.

The system can also be used for exercises and tests, by turning off the automatic rearrangement and instead show problematic phrases in another colour. One example exercise could be to turn a given sentence into passive form by moving words and changing their inflection until the sentence is correct. Multlinguality can also be used for exercises, e.g., to build a correct translation of a sentence by moving and modifying the translated words.

2.3 Grammar modules

Different grammatical and linguistic constructions are put in separate grammar modules, which the user him/herself can choose to train. Several modules can be chosen at the same time, for training combined phrases. Examples of

[1] GRASP is an acronym for "grammatikbaserad språkinlärning" (grammar-based language learning).

constructions that can be put into modules of their own are prepositional phrases, relative clauses, adjectives, passive form, word compounds, topicalization, conjunctions, and infinitive phrases.

2.4 No free text input

The system does not allow the user to enter words, phrases or sentences from the keyboard. There are several reasons for this, but the main reason is to avoid problems with words and grammatical constructions that the system doesn't know anything about. Systems that are supposed to handle free text input sooner or later run into problems with unknown words or phrases (Heift, 2001).

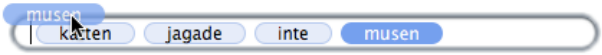
3. An illustrative example

As an explanatory example, we show how to transform a sentence in active form (*katten jagade inte musen – the cat didn't chase the mouse*) into passive form (*musen jagades inte av katten – the mouse wasn't chased by the cat*), in two different ways.

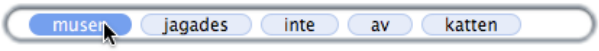
3.1 Moving a word to another position

We start by grabbing a word, in this case the word "musen" which is in object position:

While we move the word the sentence remains unaffected, but the marker gives a hint of where the word can be inserted:



Finally we drop the word in its new subject position, but the resulting sentence (*musen katten jagade inte*) is not correct. Therefore the system rearranges the sentence to the closest possible grammatical. In this case the sentence is transformed into passive form:

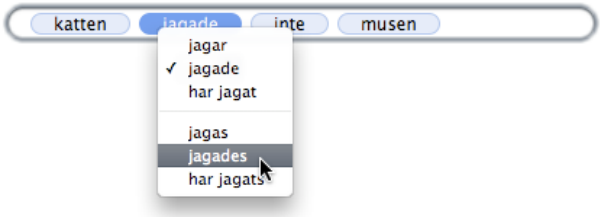


If a topicalization module had been active instead of a passive form module, the system would have topicalized the sentence (*det var musen som katten inte jagade – it was the mouse that the cat didn't chase*).

What will not happen is that the mouse becomes the subject instead of the cat (*musen jagade inte katten*), since it involves two changes in the GF syntax tree (changing the subject and changing the object), whereas passive form or topicalization only involves one change.

3.2 Choosing verbform in the context menu

Another way of turning the sentence into passive form is to select from the context menu of the verb:



Note that the contents of the context menu depends on which grammar module is active. If the topicalization module had been active, the word "musen" would get its context menu extended with "det var musen" or something similar.

4. Implementation

The system consists of three implementation layers. The bottom layer is the GF grammar formalism (Ranta, 2009b). We use GF's multilingual resource grammar (Ranta, 2009a) to define the different grammar modules. The example texts are stored as GF syntax trees, and the GF linearization algorithm is used for displaying the sentences to the user. We have no use of parsing the sentences, since the syntax trees are already known and there is no free text input.

On top of GF we have implemented an API for modifying syntax trees by specifying linearization constraints. The API consists of functions that transform trees to obey the constraints, by using as few transformations as possible. An example of constraints can be that the linearizations of some given tree nodes must come in a certain order (e.g., when the user moves a word to a position between two other words). Another example is that the linearization of a given node must be of a specified form (e.g., when the user select a specific word form from the context menu).

For the API functions to work, we have defined a similarity measure between GF trees. This is based on the notion of *tree edit distance* (Bille, 2005), but with modifications to ensure type-correctness according to the GF type system.

The final layer is the graphical interface, which communicates with the API to decide which words can be moved where, and what their context menus should contain.

5. Discussion

The GRASP system is work in progress, and not all features described in section 2 are implemented:

The grammar is a monolingual Swedish grammar, and the module system is not fully developed yet. The grammar curently handles noun phrase inflection, fronting of noun phrases, and verb inflection. The graphical interface cannot yet handle all kinds of interaction, only context-click and movement; the underlying API however is more mature.

Our plan is to have a working demonstration system by the end of 2010.

6. References

Philip Bille. 2005. A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1–3):217–239.

Graham Davies. 2010. Information and Communications Technology for Language Teachers (ICT4LT). Accessed 26 aug 2010 from `http://www.ict4lt.org/en/`.

Trude Heift. 2001. Intelligent language tutoring systems for grammar practice. *Zeitschrift für Interkulturellen Fremdsprachenunterricht*, 6(2).

Aarne Ranta. 2009a. The GF resource grammar library. *Linguistic Issues in Language Technology*, 2.

Aarne Ranta. 2009b. Grammatical Framework: A multilingual grammar formalism. *Language and Linguistics Compass*, 3(5):1242–1265.