Tools for Large-Scale Parser Development

Natural Language Processing Group
Microsoft Research
One Microsoft Way
Redmond WA 98052 USA

1. Introduction

We demonstrate the tool set available to linguistic developers in our NLP lab, with a particular emphasis on the tools for incremental regression testing and creation of regression suites. These tools are currently under use in the daily development of broad-coverage language analysis systems for 7 languages (Chinese, English, French, German, Japanese, Korean and Spanish). The system is modular, with the parsing engine and debugging environments shared by all languages. Linguistic rules are written in a proprietary language (called *G*) whose features are uniquely suited to linguistic tasks (Heidorn, in press). The engine underlying the system, as well as the user interface for linguistic developers, is unicode-enabled thus supporting both European and non-Indo-European languages.

2. Tools for regression testing

The purpose of this class of tools is to build regression suites, which is a collection of what we call *master files*. The master files take the form of stored output trees, and keep a record of the state of development at any point in time.

The linguistic developer builds a set of regression files over the course of grammar development, thus developing annotated corpora. Because the system is intended to cover a broad range of input including ungrammatical input, and because we are very open to letting real text dictate grammar structures rather than theory, we find that annotated structures output by the system are more useful for development than manually tagged corpora.

The standard practice of parser development within our group is schematically shown in Figure 1. As grammar work progresses, developers can run regression tests against the regression suites to examine the consequences of the changes to the grammar. When differences are found, the system gives a color-coded display of the differences (new changes in green, what is in the master file in red, and unchanged part in gray). If the change is an improvement, the developer can choose to update the master file by simply double-clicking on the sentence number on the display, and add the sentences that are newly accommodated to the regression suite. If the change is evaluated as negative, the linguistic developer reworks the rules that caused the regression.

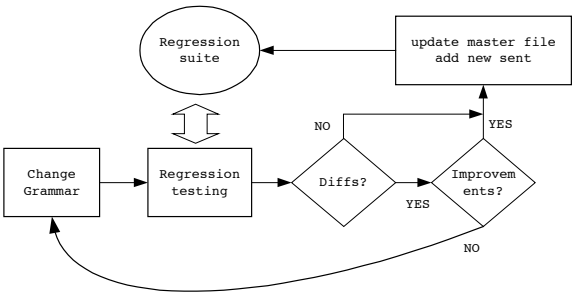

Figure 1. Flow diagram of daily grammar development process

Since we run regression tests many times a day in the grammar development, the processing speed of the systems is a vital issue. Current performance estimates for regression testing are 20 to 30 sentences per second on a 550 MHz Pentium III machine with 512MB RAM across languages (average sentence length = 16.51 words in English, 49.02 chars in Japanese, for example). We also have means to distribute the processing of regression testing onto multiple CPUs: currently, 3 machines with 4 CPUs each (500 MHz, 768MB RAM) regress 27,000 sentences in less than 100 seconds or about 275 sentences per second (English, on the same corpus as above).

3. References

Heidorn, George. In press. Intelligent Writing Assistance. To appear in Robert Dale, Hermann Moisl and Harold Somers (eds.), *Handbook of Natural Language Processing*. Chapter 8.