Computerized Linguistics, a commentary

by

Martin Minow, KVAL Group, Stockholm

In reading through the various preprints, I was struck by the fact that many were concerned with specific computer implementations of one kind or another. Unfortunately, there seems to be little discussion of the programming from the programmer's viewpoint. This is understandable in a congress of computational linguistics but, I feel, somewhat regrettable. Especially since the reports of what has already been done surely will stimulate other participants to use computers. I think that an informal session on programming in computational linguistics would be both interesting and valuable, allowing an exchange of ideas and experiences which would be much more difficult in the even more informal atmosphere of the dining room.

In order to initiate such a session, I am enclosing the following questions, hoping that they will stimulate a certain informal correspondence before the congress begins. If enough interest is shown in this idea to warrant a session, I intend to edit the replies received (to the extent they show similarities), making summaries available to participants before the conference.

In answering the questions, it is of course not necessary to confine yourself to the particular project you are discussing at the conference if another project will yield a more interesting reply. This is not, in other words, a final examination.

What type of project?

IR
Modeling of linguistic system
Data-processing
Something else or more than one area
One-time run or continuous production

What language (machine, etc) was used? (Please note that when I say language I really mean language + hardware + system.)

Was batch processing, time sharing, or conversation used?
Why this choice? Were other languages available? Suitable?

How did the choice of language influence the definition of the problem? the programming?

About how much time (thinking, physical, machine) was required for - and how did the choice of language influence - the following:

Statement of problem (flow charting, definition of algorithm, pure linguistic considerations)

Defining data structures (lists, how character strings are stored)

"System programming" (character manipulation subroutines for example; even definition of input-output formats, external storage, etc.)

Programming the algorithm

Initial and final debugging (i.e. correcting the program versus correcting the algorithm)

Documentation

How large is the program in source form? What percent commentary?

If you used time sharing, did it help?

How did the software "bureaucracy" (for example, Job Control Language on th(IBM 360) affect the programming for better or worse?

Can the program be easily implemented on a different machine or system? (Was this one of the objectives?) If this was tried, how painful was the experience of transferring the program?

If you were to start all over again, would you use the same language?

Have you been teaching programming to linguistics students? If so, what has been their experience? Yours? Should linguists learn programming? Why?

KVAL is beginning to plan a fairly large computer system for translation from Swedish to "condensed" Braille. This requires a large dictionary, fairly sophisticated character manipulation routines, and the ability to perform purely linguistic operations on the input text (to recognize syllables, morphemes, etc). Additionally, the input-output problems are not at all trivial. We also want to produce a system which can be moved easily - both to different computers within Sweden, and even, if possible, to other languages, The program should also b self-documenting and permit easy modification in a modular fashion as new words, new syntactic transformations, etc prove useful. It would also be nice if the "linguistic work" could be written by linguists with only a little programming training, rather than the other way around. We wouldn't even mind if it was cheap and fast.

Suggestions please: What language, machine, type, etc? How should the dictionary be stored (assume any random-accesss device you are familiar with)? Discuss the goals, ranking them in the light of your own experience.

There is much to criticize in the choice and form of the above questions. The naivety is obvious - to a certain extent it is due to my own inexperience in this field (KVAL`s problem has surely been faced - if not solved - by every IR project, for example). To a greater extent, however, the naivety is present to prevent the questions from containing their own answers. By all means, criticize the questions. If you do not feel they are pertinent to your own experience, or to the problem itself, ask your own, and answer them.

Please inform me if you would be interested in taking part in a panel discussion during the conference.