UNIVERSITY OF MASSACHUSETTS: MUC-3 TEST RESULTS AND ANALYSIS

Wendy Lehnert, Claire Cardie, David Fisher, Ellen Riloff, Robert Williams

Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003
lehnert@cs.umass.edu

TEST RESULTS

We believe that the score reports we obtained for TST2 provide an accurate assessment of our system's capabilities insofar as they are consistent with the results of our own internal tests conducted near the end of phase 2.. The required TST2 score reports indicate that our system achieved the highest combined scores for recall (51%) and precision (62%) as well as the highest recall score of all the MUC-3 systems under the official MATCHED/MISSING scoring profile.

We ran one optional test in addition to the required test for TST2. The optional run differs from the required run in only one respect, an alteration to our consolidation module. The consolidation module contains all procedures that translate parser output into target template instantiations. The complete consolidation module includes a case-based reasoning (CBR) component that makes predictions about the target output based on a portion of the development corpus. For our optional run, we executed a modified version of consolidation that does not include this CBR component. We predicted that the absence of the CBR component would pull recall down but push precision up (looking at MATCHED/MISSING only). This trade-off prediction was confirmed by the required and optional TST2 score reports. (Please consult Appendix F for our required and optional test score summaries).

The source of our recall/precision trade-off can be found by examining the actual, spurious and missing counts for template-ids. When we run with CBR, we generate 215 actual templates as opposed to 137 actual templates without CBR. Most of these extra templates are spurious (64), but some are correct (14). The extra CBR templates increase our recall by reducing the number of missing templates from 16 to 6, while lowering our precision by raising the number of spurious templates from 44 to 108. The net effect of the CBR component on TST2 is a 4% gain in recall and a 3% loss of precision.

All of our system development and testing took place on a Texas Instruments Explorer II workstation running Common Lisp with 8 megabytes of RAM. It took about 1.5 hours to process the TST2 texts (without traces). No effort had been made to optimize run-time efficiency. Shortly after the final TST2 evaluation we found a way to reduce runtimes by about 40%.

SYSTEM DEVELOPMENT

Almost all of our MUC-3 effort has been knowledge engineering in one form or another. We can further categorize this effort in terms of (1) dictionary construction, and (2) discourse analysis. Dictionary construction received somewhat more attention than discourse analysis, with both relying heavily on examples from the development corpus. Overall, we estimate that roughly 30-40% of the development corpus was analyzed for the purposes of either dictionary construction or discourse analysis by the end of phase 2.

Because we are working with a domain-specific dictionary, we construct our lexicon on the basis of examples in the development corpus. Virtually all of our dictionary construction is done by hand. We examine texts from the corpus in order to identify critical verbs and nouns that organize information relevant to the domain. Then we create syntactic and semantic predictions based on these instances with the expectation that similar linguistic constructs will be encountered in other texts as well. Our dictionary is effective only to the extent that we can extrapolate well on the basis of the examples we've seen.

Our TST2 dictionary contained 5407 words and 856 proper names (mostly locations and terrorist organizations). 1102 dictionary entries were associated with semantic features, and 286 entries operated as concept node triggers (CIRCUS cannot produce any output unless it encounters at least one concept node trigger). 131 verbs and 125 nouns functioned as concept node triggers. Our semantic feature hierarchy contained 66 semantic features. Although CIRCUS operates without a syntactic sentence grammar, it did exploit syntactic knowledge in the form of 84 syntactic prediction patterns, with 12 of these doing most of the work. CIRCUS also accessed 11 control kernels for handling embedded clause constructions [1].

Our version of discourse analysis took place during consolidation, when output from the CIRCUS sentence analyzer was examined and organized into target template instantiations. This translation from CIRCUS output to MUC-3 templates was handled by a rule base containing 139 rules. Consolidation errors could effectively destroy perfectly good output at the level of sentence analysis, so our overall performance was really only as good as our consolidation component. One of our ongoing problems was in trying to evaluate the performance of CIRCUS and the performance of consolidation independently. We never did manage to tease the two apart, but we are confident that both components would benefit from additional knowledge engineering.

Serious consolidation development could not really get underway until we had a large number of texts to examine along with internal scoring runs based on the development corpus. Although our earliest opportunity for this was November, dictionary deficiencies delayed substantial progress on consolidation until February or March. It was impossible to know how well consolidation was operating until CIRCUS could provide consolidation with enough input to give it a fighting chance. The consolidation rule base was generated by hand and modified upon inspection, with rapid growth taking place during phase 2. The number of consolidation rules almost doubled between TST1 and TST2.

We estimate that our time spent (measured in person/years) on technical development for MUC-3 was distributed as follows:

alterations to CIRCUS	.35
case-based discourse analysis	.15
corpus development	.25
dictionary construction	.75
rule-based discourse analysis	.50
test runs & other misc. technical	.25

	2.25 person/years

This estimate assumes that our graduate research assistants were working 30 hrs/wk on MUC-3, although it is notoriously difficult to estimate graduate student labor. General alterations to CIRCUS included morphological analysis, conjunction handling, noun phrase recognition, embedded clause handling, and machinery for some special constructions like appositives. These alterations to CIRCUS and the CBR component are all domain-independent enhancements. All other effort should be categorized as domain-specific.

DOMAIN-INDEPENDENT ADVANCES

Prior to MUC-3, we had no experience with consolidation-style processing, so consolidation provided us with many opportunities to explore new problem areas. For example, we can locate pronominal referents both within sentences and across sentence boundaries 73% of the time (based on an analysis of pronouns in the relevant texts of the development corpus and TST1). However, these heuristics are limited to four pronouns and there are only 130 instances of these pronouns in the texts analyzed. We examined the role of pronoun resolution with internal test runs, and came to the conclusion that this particular problem has little impact on overall recall or precision.

A more compelling innovation for consolidation was first proposed in March, when we began to experiment with the CBR component. The CBR component allows our system to augment its final template output based on known correlations between CIRCUS output and target template encodings found in the development corpus. It performs this analysis using a case base of 254 template patterns drawn from the 100 TST1 texts along with 283 development corpus texts.

Case-based consolidation generates additional templates that might have been missed or dismissed during the rule-based analysis, and thereby reduces the number of missing templates. Because the CBR component effectively operates to counterbalance omissions made by rule-based consolidation, we expect that the gain in recall due to CBR will eventually diminish as the system becomes more comprehensive in its domain coverage. Even so, the prospects for applying CBR techniques in NLP are open-ended, and deserve further attention. This preliminary attempt to bring CBR into natural language processing is one of two original advances made during the course of our work on MUC-3.

The other significant advance was made very early on while we were assessing the robustness of the CIRCUS sentence analyzer and making some final adjustments to CIRCUS. We were generally concerned about scaling up with respect to complex syntax, and thinking about ways that CIRCUS might approach syntactically complex sentences in a principled manner. At that time we discovered a formalism for embedded clause analysis, Lexically Indexed Control Kernels (aka LICKs). LICKs describe syntactic and semantic interactions within CIRCUS as it interprets embedded clauses. This formalism makes it relatively easy to see how CIRCUS handles an embedded clause, and has made it possible for us to talk about this aspect of CIRCUS much more effectively. In fact, a paper was written during MUC-3 relating embedded clause analysis by CIRCUS to experimental results in psycholinguistics [1]. In that paper we argue that CIRCUS provides a cognitively plausible approach to complex syntax.

UP AGAINST THE WALL: ARE WE THERE YET?

The major limiting factor in our TST2 performance was time. We are confident that significant improvements in recall could be made if we had more time to do more knowledge engineering. We would also predict higher precision scores although our precision percentages have grown at a much slower rate than our recall percentages, based on a comparison of official test scores for TST1 and TST2.

We tend to think of our system in three major pieces: (1) the CIRCUS sentence analyzer, (2) rule-based consolidation, and (3) case-based consolidation. Because the CBR component is truly optional, the primary responsibilities fall on CIRCUS and rule-based consolidation. We know that both of these components make mistakes, but we have not been able to separate them well enough to say which one is the weaker one. As with all knowledge-based systems, an assessment of these components is also confounded by the fact that we are working with incomplete knowledge bases. Both the dictionary and the consolidation rule base incorporate domain knowledge, and we have thus far analyzed less than 50% of the MUC-3 development corpus in our knowledge engineering efforts.

As one might expect, our best internal test runs are those that include texts we have analyzed for the purposes of constructing our dictionary and consolidation rules. For example, on May 13 we ran the TST2 version of our system on TST1, and posted recall-precision scores of 66-68 running with CBR, and 62-73 running without CBR (for MATCHED/MISSING). It is heartening to contrast this with our phase 1 test results for TST1 which were 28-59 (no CBR component was available for phase 1 testing). Roughly 20% of the TST1 texts were analyzed between February and May, so the substantial improvement in both recall and precision on TST1 can be only partially attributed to knowledge engineering based on the TST1 texts. A complete analysis of all the TST1 texts would provide us with a better estimate of a performance ceiling that is not confounded by inadequate knowledge engineering.

As far as future system development goes, we cannot conclude at this time that any one of our system components requires redesign or major alterations. We would like to exploit more of the corpus for the sake of knowledge engineering to get a better sense of what we can do when incomplete knowledge is not a factor. Only then can we hope to isloate limitations that need to be addressed by innovations in system design.

One limitation that applies more to our system development than our system itself, is the hand-coding of dictionary definitions and consolidation rules. It would be highly advantageous for us to automate certain aspects of this or at least design an intelligent interface to speed the work of our knowledge engineers. We did manage to use the CBR component as a tool to direct us to useful texts during dictionary construction, and this application of the CBR component was both very welcome and very effective (albeit rather late in the MUC-3 timetable). In any event, we would clearly benefit from intelligent interfaces or more ambitious machine learning strategies to help facilitate the knowledge engineering effort that is so central to our whole approach.

To sum up, we are confident that the performance of CIRCUS has not yet reached its limit. Unfortunately, it is not possible to say anything about where our ultimate upper bound lies. We hope to pursue this question by participating in future performance evaluations.

ACKNOWLEDGEMENTS

Our participation in MUC-3 was supported by a DARPA contract administered by Meridian Aggregates Co., Contract No. MDA903-89-C-0041, the Office of Naval Research, under a University Research Initiative Grant, Contract No. N00014-86-K-0764 and NSF Presidential Young Investigators Award NSFIST-8351863.

BIBLIOGRAPHY

[1] Cardie, C., and Lehnert, W., "A cognitively plausible approach to understanding complex syntax," *Proceedings of the Ninth National Conference on Artificial Intelligence.* Anaheim, CA. 1991.