An Efficient Chinese Parsing Algorithm for Computer-assiste d Language Learning

Chi-Hong LEUNG, Yuen-Yan CHAN and Albert Kam-Wah WU
Faculty of Education
The Chinese University of Hong Kong
Hong Kong

Abstract

Instructional grammar is often used in Computer-assisted Language Learning (CALL) and the grammatical error detection is an important feature. However, it is not an easy task in Chinese language. There is no delimiter separating consecutive words in Chinese sentences. Word segmentation is a process in which proper word boundaries are identified. Before syntactic parsing of a Chinese sentence, word segmentation has to been performed. Traditionally, the word segmentation is often followed by the syntactic parsing. This paper introduces an algorithm in which the Chinese word segmentation and syntactic parsing are combined into one process to increase the overall efficiency.

1. Introduction

Computer Assisted Language Learning (CALL) is related to the use of computers for language teaching and learning. Higgins [2] described three different models of grammar teaching: instructional, revelatory and conjectural. Instructional grammar is often used in CALL because of being computerized easily. In this paper, an efficient Chinese parsing algorithm that can combine parsing and word segmentation [4,5,6] to increase the processing rate of determining if a Chinese sentence is grammatically correct.

2. Algorithm of Chinese Word Segmentation Associated with Syntactic Parsing

The algorithm of *Chinese word segmentation associated with syntactic parsing* is modified from Earley's parser [1]. It can be divided into a number of functions: *predictor*, *completer*, and *scanner*.

sn = state number (e.g. 12) gr = grammar rule (e.g. S → NP VP) gp = grammar rule position (e.g. 2)
su = successor (e.g. VP) sp = sentence position (e.g. 3) t = type (P for predictor; S for scanner; C for completer)
pr = previous state list (e.g. {12, 34}) a state = (sn, gr, gp, su, sp, t, pr)
L-Sym(gr) = the left symbol of a grammar rule, gr R-Sym(gr, n) = the n^{th} symbol on the right of a grammar rule, gr
state.attribute = an attribute of a state (e.g. α.gp = the value of the grammar rule position in *state* α.)

Notations used in algorithms

```
predictor($\alpha$)     // $\alpha$ represents a certain state being processed by the predictor
    for each grammar rule, G
        if L-Sym(G) = R-Sym($\alpha$.gr, gp+1)
                $\beta$ = (state-total+1, G, 0, R-Sym($\alpha$.gr, $\alpha$.gp+2), $\alpha$.sp, P, {$\alpha$.sn})
                // state-total is a counter indicating how many states are created
                add($\beta$);

add($\beta$)   // $\beta$ represents a certain state being added
    for each existing state $\gamma$
        if $\gamma$.gr = $\beta$.gr AND $\gamma$.gp = $\beta$.gp AND $\gamma$.su = $\beta$.su AND $\gamma$.sp = $\beta$.sp AND $\gamma$.t = $\beta$.t
                if $\gamma$.sn not in $\beta$.pr
                    append $\beta$.pr in $\gamma$.pr;
                else
                    add $\beta$ as a new state;
```

Proceedings of the International Conference on Computers in Education (ICCE'02)
0-7695-1509-6/02 $17.00 © 2002 **IEEE**

```
                state-total = state-total + 1;

scanner(α) // α represents a certain state being processed by the scanner
    for each word, W, segmented at the position α.sp in the sentence
        if W has a syntactic tag = L-Sym(α.gr, α.gp+1)
            β = (state-total+1, α.gr, α.gp+1, α.su, (α.sp + length of W), S, {α.sn});
            add(β);

completer(l, α)   // α represents a certain state processed by the completer and l represents a previous state list of a state
    for each state, s, registered in l
            if ( R-Sym(s.gr, s.gp+1) = L-Sym(α.gr) AND R-Sym(s.gr, s.gp+2) = α.succ)
                β = (state-total+1, s.gr, s.gp+1, s.su, α.sp, C, {α.sn});
                add(β);
            else
                completer(s.pr, α)

Segmentation_and_syntactic_parsing()
    state-total = 0;
    add(0, "ϕ → S", 0, null, 0, P, {}); // add first state
    for each new state, α, being added
            if R-Sym(α.gr, α.gp+1) = non-terminal symbol
                predictor(α);
            else if R-Sym(α.gr, α.gp+1) = terminal symbol
                scanner(α);
                else if α.gp = number of symbols on the right of α.gr
                    completer(α.pr, α);
```

Algorithm of Chinese Word Segmentation Associated with Syntactic Parsing

3. Experiment and conclusion

An experiment was performed to evaluate the efficiency of the algorithm introduced in this paper. In the experiment, a text consisting of 1,000 sentences selected from a Chinese corpus [3] was processed twice with the same set of 81 grammar rules. From the experiment result below, it is proved that the algorithm of word segmentation associated with syntactic parsing introduced in this paper can increase the processing speed significantly.

	Traditional approach	New approach
Average number of states generated for a sentence	398,599.25	1,829.75

Experiment result

References

[1] Earley, J. (1970). An Efficient Context-free Parsing Algorithm, *Communications of the Association for Computing Machinery*, 13: 94-102.

[2] Higgins, J. (1986). The Computer and Grammar Teaching, in Leech, G. and Candlin, N.L., *Computers in English Language Teaching and Research*, London: Longman, 31-45.

[3] Jin, G. (1993). PH -- A free Chinese corpus, *Communications of COLIPS*, 3: 45-48.

[4] Leung, C.H. and Kan, W.K. (1996a). Parallel Chinese Word Segmentation Algorithm based on Maximum Matching, *Neural, Parallel and Scientific Computations*, 4: 291-304.

[5] Leung, C.H. and Kan, W.K. (1996b). A Statistical Learning Approach to Improving the Accuracy of Chinese Word Segmentation, *Literary and Linguistic Computing*, 11: 87-92.

[6] Wong, K.F., Lum, V.Y. and Leung, C.H. (1997). Parallel Chinese Word Boundaries Identification in the IPOC Information Retrieval System, *International Journal of Information Technology*, 3: 63-81.