Design and Realization of Mongolian Syntactic Retrieval System Based on Dependency Treebank

S.Loglo
Language Research Institute
College of Mongolian Studies
Inner Mongolia University
Huhhot, Inner Mongolia Autonomous Region, China

Sarula
Department of Journalism and publishing
College of Mongolian Studies
Inner Mongolia University
Huhhot, Inner Mongolia Autonomous Region, China

***Abstract*—In the past seven years, Language Research Institute of Inner Mongolia University has constructed a 500,000-word scale Mongolian dependency treebank. The syntactic treebank provides a favorable data platform for language research and information processing. In order to effectively use the treebank, we have designed and implemented a graphical syntactic information retrieval system based on the Mongolian dependency treebank. As an application system, this retrieval system offers search and statistical analysis on word, phrase, syntactic fragment and syntactic structure level.**

Keywords—Mongolian Language; Dependency Grammar; Dependency Treebank; Syntactic Retrieval; Information Retrieval

I. INTRODUCTION

Language Research Institute of Inner Mongolia University has constructed a 1-million-word modern Mongolian corpus in a span of eight years from 1984 to 1991 and expanded it twice into what is now a 10-million-word corpus. The 1-million-word corpus contains materials from novels (19.6%), textbooks (50.3%), newspapers (9.8%) and politics (22.9%) [1]. In corpus annotation, the 10-million word corpus has been completed the part-of-speech tagging [2][3] and fixed phrase tagging [4]. And some shallow parsing is carried out on the 1-million word corpus, such as phrase tagging [5][6][7], automatic sentence segmentation [8] and automatic predicate segment recognition [9].

From 2008 to 2011, funded by National Social Science Foundation and National Natural Science Foundation, using the method of automatic parsing and manual proofreading, Language Research Institute of Inner Mongolia University has constructed a 500,000-word Mongolian dependency treebank (MDTB) based on middle school Mongolian textbooks that were extracted from the 1-million-word modern Mongolian corpus [10]. MDTB has an annotation set of 17 dependency relations under 5 categories [11]. The 5 categories are special relation, dominant relation, conjunctional relation, auxiliary relation and non-syntactical elements. The 17 dependency relations include: key word in a sentence (HEAD), independent element (INDE), subject (SUBJ), direct object (DOBJ), indirect object (IOBJ), attribute (ATT), adverbial (ADV), coordinate (COO), appositive (APP), summarization (SUM), time-local words-auxiliary (TL-AUX), postposition-auxiliary (PP-AUX), modal particles-auxiliary (MP-AUX), modals-auxiliary (M-AUX), auxiliary verbs-auxiliary (AV-AUX), contact verb-auxiliary (CV-AUX) and conjunction-auxiliary (CJ-AUX). In the form of annotation, MDTB uses two types of labeling, namely the brackets annotation and graphical annotations. This treebank contains 461,240 words in 31,722 sentences. The average sentence length is 14.54 words.

Mongolian dependency treebank contains rich syntactic information, so the researchers can obtain all kinds of information about syntax. On the dependency treebank, researchers also can perform statistical analysis and example sentence extraction. Therefore, it provides convenience for the study and research of Mongolian traditional linguistics and computational linguistics [12] [13]. However, at present, treebank is usually used as a training and evaluation data for syntactic parsing [14], but research about the syntactic information retrieval is few and far between. This paper is designed to expound a syntactic treebank retrieval system based on the application system of the dependency treebank. The retrieval functions allow researchers to do enquiry and statistical analysis on word, phrase, sentence constituents, syntactic fragment and syntactic structure.

II. DESIGN AND REALIZATION OF MONGOLIAN SYNTACTIC RETRIEVAL SYSTEM

Dependency tree-based Mongolian syntactic retrieval system is divided into two parts, syntactic tree display module and retrieval statistics module. Realization of the two modules is presented as follows.

A. Display Module of the Syntactic Tree

Graphic display lies in the heart of treebank operation as an essential technology. However formatted a treebank is, in text or in graph, the output module can draw a complete tree for each sentence, as shown in the Fig.5. The left window displays corpus texts of which the current sentence is displayed in selected model. The right window displays the syntactic tree of the current sentence.

Fig.1 shows the node structure of the syntactic tree. Each node on the syntactic tree may have n child nodes which are arranged from left to right according to the sequence in which dependency relations were established. But this order dampens the readability of treebank. To recover the original order of brother nodes, we have added sorting function to the output model. The output module also provides multiple optional display modes which are presented as follows.

Sponsored by National Natural Science Foundation (60763033), National Natural Science Foundation (61262046), and National Social Science Foundation(10CYY022).

1) Open or close lexical information display function;

2) Contract or expand descendant nodes; and

3) Display the entire syntactic tree or only display the search results.

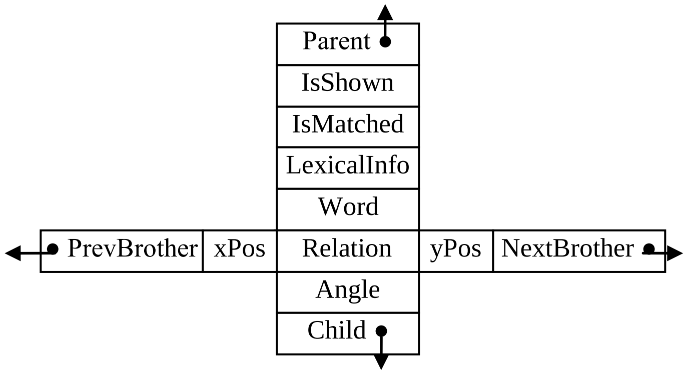


Fig. 1. Node structure of a dependency tree

In Fig.1, ***PrevBrother*** stores a pointer that points to the node's previous brother, ***NextBrother*** stores a pointer that points to the node's next brother, ***Parent*** stores a pointer that points to the node's parent node, ***Child*** stores a pointer that points to the node's child node, ***Word*** stores the node's word, ***LexicalInfo*** stores the node's lexical information, ***Relation*** stores the node's dependency relation, ***xPos*** and ***yPos*** stores the node's horizontal and vertical ordinate, Angle stores the node's inclined angle of dependency arc, ***IsShown*** denotes whether the node's descendent nodes are shown or not, and ***IsMatched*** denotes that the node is among the research results.

The output algorithm is as follows:

```
VOID ShowTree (CTree *T,int ShowMode)
//T denotes the dependency tree;
//ShowMode==”0” indicates that the program will
//display all syntax trees; and "ShowMode==1" indicates
//that the program will display the results of the search.
{If (ShowMode==1 &&
T->bSearchResult==FALSE) return;
//Visit RootNode on the dependency tree T;
If(!RootNode->IsShrunk())
// denotes that the root node is not shrunk;
ShowNode(RootNode);//draw the node (RootNode)
If(RootNode->bShowLexcicalInfo==TRUE)
ShowLexicalInfo(RootNode);
//shows the node’s lexical information;
SortChildrens(RootNode);
//Sort the child nodes of RootNode;
//Traverse subtree forest of root nodes pre-orderly;
For(i=0;i<n;i++)a
//n is the number of child nodes of RootNode
ShowTree (CTi);
//CTi denotes a sub-tree whose root node is the ith
//child of T
Return;
}
```

Algorithm1. Dependency tree display algorithm

B. Design of Mongolian Syntactic Retrieval Algorithm

Treebank is an important resource for syntactic analysis and evaluation, word sense disambiguation and semantic analysis. MDTB provides a favorable data platform for Mongolian language research and information processing. At present, the use of treebank is mainly achieved through a variety of retrieve technology-based statistical methods. As such develop an efficient search algorithm is very necessary for treebank-based systems [15] [16].

The dependency treebank herein adopts two different storage formats, text and graph. Text format is for treebank that targets all users and can be opened and edited by any text editing software. Graphical format, which MDTB adopts, benefits both output and retrieval, although it does not perform better than text format in terms of space utilization. Based on graphical storage format, we have designed a treebank retrieval algorithm with sub-tree query function. The query conditions can be a sub-tree, a word or a syntactic fragment with n nodes. Each node can have one or multiple characteristic values such as vocabulary (can use wildcards like ‘*’ and ‘?’), parts of speech, sub-categorization, morphology, dependency relation type, father node and child node.

The retrieval algorithm is as follows:

1) Traverse syntactic tree pre-orderly to find root node (sRoot) of query condition;

2) If a node (tNode) in the dependency tree satisfies the requirement of the query condition’s root node (sRoot), then, to find all the child nodes of sRoot in the child nodes of the tNode;

3) If all of sRoot’s child nodes are found in step (2), recursively call step (2) until nodes that meet the requirements can no longer be found or the rightmost descendent nodes of query conditions are found;

4) Continue to find sub-trees among the remaining nodes of the current tree (excluding traversed nodes); and

5) Treebank search needs to call step (1) to step (4) repeatedly.

tNode represents one particular node on syntactic tree, and sRoot denotes root node of one sub-tree under given query conditions at given moment (including query condition itself).

C. A Syntactic Retrieval Example

The rationale of retrieval algorithm is explained by finding juxtaposed attributive in the following sentence.

Fig.2 shows the dependency tree of the following sentence,

bi uran=sibauxay-yin xatagu=sirgagu ağilči=xödelműriči ǰorig=sanag_a bolon uran=narin egűr sűlǰixű mergeǰil-i űnen=sedxil-eče-ben bisiren_e. (I admire from the bottom of my heart the brave and hard-working bird’s strong will and superb nesting skill).

The dotted lines represent syntactic fragments that meet the requirements of query conditions.

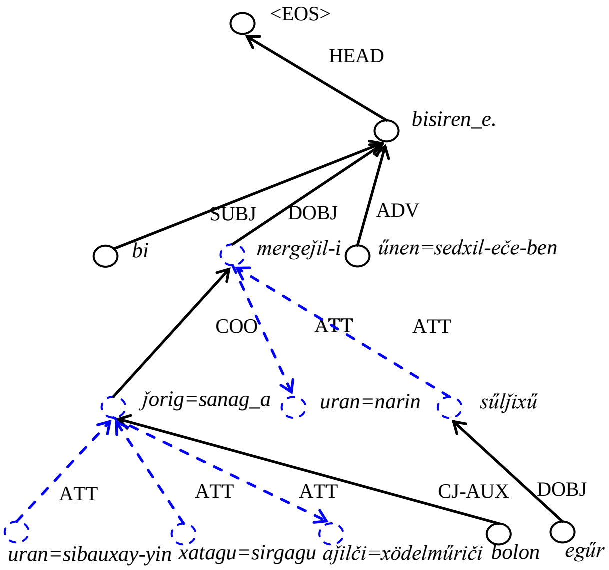


Fig. 2. Dependency Tree of the example sentence

Query condition can be a word or phrase node, a dependency arc or a syntactic fragment of any size. In the query condition, node can have 16 attribute values including the word or phrase itself, part of speech, syntactic relation and affix. In the process of search, a query interface as shown in the Fig.5 will pop up. A new node will be added by clicking the white dot. A dependency relation can be established between two nodes by dragging the mouse. Fig.3 shows the query conditions of this example.

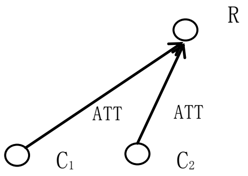


Fig. 3. Query Condition

1) Find nodes that match with R in dependency tree using pre-order traversal. As R itself has no constraints, every node in the dependency tree meets its requirement. The key is to check whether the node has two ***ATT*** *child nodes. In the process of traversal, the node “mergeǰil-i” meets the requirement, as shown in Fig.4 (a).*

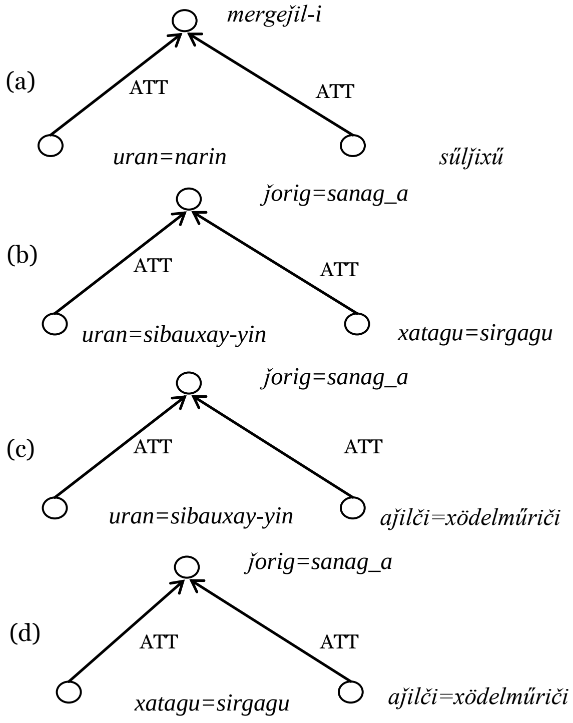


Fig. 4. Node combinations that meet the query condition

As the enquiry condition only contains two-layer nodes, there is no need for recursive query.

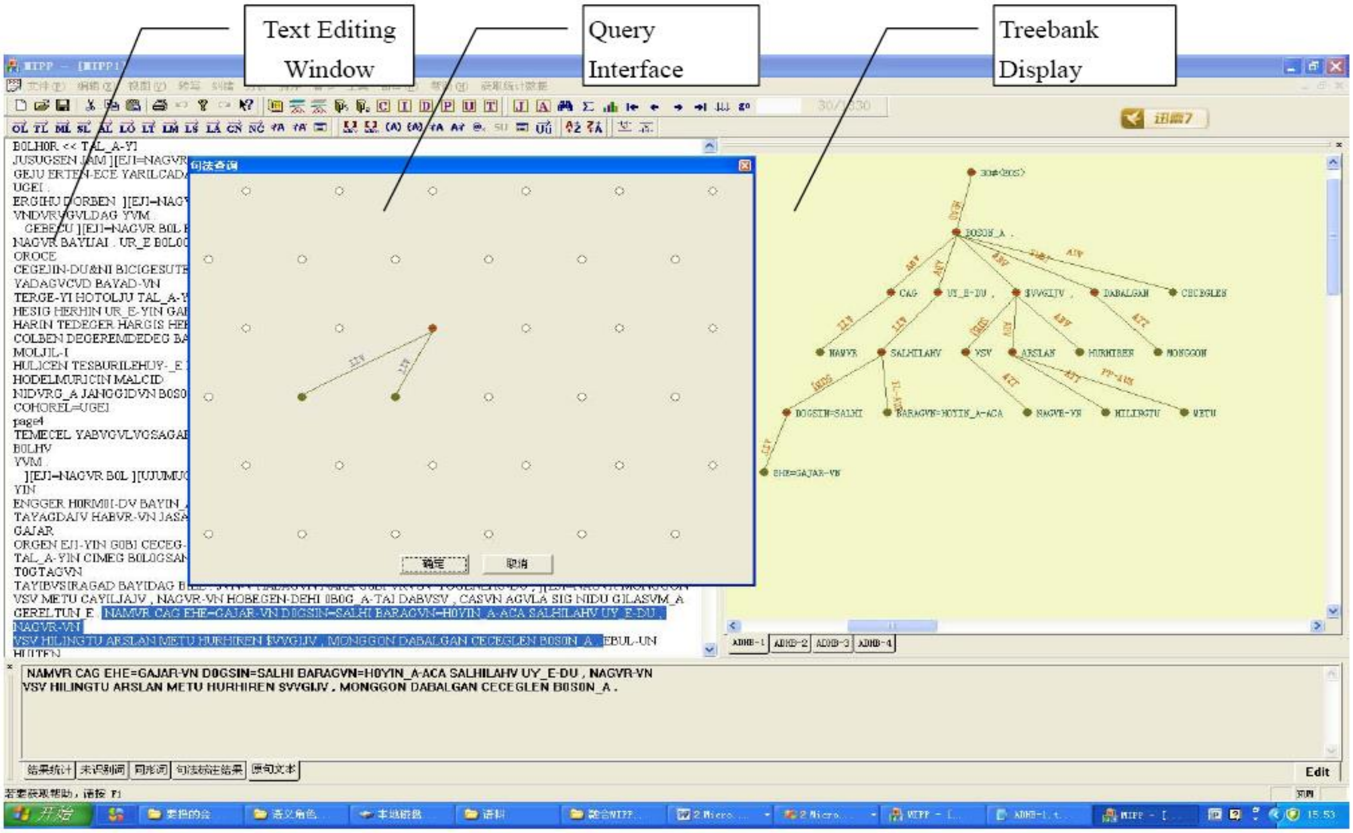
2) Label with different colors syntactic fragments that have been found. Continue to traverse the dependency tree to find the next eligible syntactic fragment.

3) The combinations where node “ ǰorig=sanag_a” and its child nodes meet the requirement are as shown in Fig.4 (b)—(d).

4) Search is done when the remaining nodes in the dependency tree have been traversed and no eligible fragments are found.

It is worth noting that the program will restore the treebank and clear the traces left from the previous query operation before next query. If the search results need to be preserved, a copy needs to be saved by using the pertinent functions in this program.

Statistical analysis of syntactic fragment is done based on query. Each search provides relevant statistical data, including the number of times a syntactic fragment appears and the number of sentences that contain the syntactic fragment.



NAMVR CAG EHE=GAJAR-VN D0GSIN=SALHI BARAGVN=H0YIN_A-ACA SALHILAHV UY_E-DU , NAGVR-VN
VSV HILINGTU ARSLAN METU HURHIREN SVVGIJV . MONGGON DABALGAN CECEGLEN B0S0N_A .

Fig. 5. The Editing, showing or query interface of Mongolian dependency Treebank

III. Conclusion

The dependency tree that this Mongolian syntactic retrieval system is based differs from binary phrase structure tree in terms of node types and tree structures. Such difference cannot be effectively handled at the current stage. Moving forward, we will add to this system display, editing and retrieval function for phrase structure tree and bidirectional conversion between the two kind of tree structures. This system has good universality, and has no particular relationship with language per se. As such it can be applied to other languages' treebank for editing and retrieval operations.

Reference

[1] Language Research Institute of Inner Mongolia University, "About modern Mongolian corpus", Journal of Inner Mongolia University (Humanities & Social Sciences), 1992, vol.24, N0.1, pp. 1–5.

[2] HUA Shabao, "AYIMAG– A POS tagging system for Mongolian corpus", Journal of Inner Mongolia University (Humanities & Social Sciences), 1999, vol.31, N0.5, pp.31–35.

[3] Zhang Guanhong, S.Loglo and Odbal, "Fusion of morphological features for Mongolian part of speech based on maximum entropy model", Journal of Computer Research and Development, 2011, vol.48, N0.12, pp.2385–2390.

[4] S.Loglo and Sarula, "Research on Mongolian lexical analyzer based on NFA", Proceedings of 2010 IEEE International Conference on Intelligent Computing and Intelligent Sytems, Xiamen, China, 2010, vol.2, pp.240–245.

[5] Hua Shabao and Dabhurbayar, "A Phrase-tagging Research in Mongolian Corpus", Journal of the Central University for Nationalities (Philosophy and Social Sciences Edition), 2006, vol.33, No.5, pp.64–67.

[6] Hua Shabao, "A tagging strategy of Mongolian phrases", Journal of the Central University for Nationalities (Philosophy and Social Sciences Edition), 2003, vol.30, N0.5, pp.98–100.

[7] Wulan, Dabhurbayar, GUAN Xiaoda and ZHOU Qiang, "Phrase structure parsing of Mongolian", Journal of Chinese Information Processing, 2014, vol.28, No.5, pp.162–169.

[8] Wang Serguleng, "Rule-based Mongolian sentence automatic segmentation," Journal of Inner Mongolia University (Philosophy & Social Sciences (Mongolian Edition)), 2009, vol.38, No.3, pp.51–55.

[9] Wang Serguleng, D.Sarana and Nasunurtu, "Design and realization of automatic annotation for modern Mongolian predicate segment", Proceedings of 11th national symposium on minority languages, Xishuangbanna, China, 2007, pp.420–427.

[10] S.Loglo and Sarula, "Construction of a Mongolian dependency treebank", International Journal of Knowledge and Language Processing, 2014, vol.5, No.2, pp.32–42.

[11] S.Loglo, "Research on the modern Mongolian syntactic tagging system based on the dependency grammar", Mongolian Studies of China, 2011, vol.39, No.2, pp.116–119.

[12] Liu Haitao, "Dependency grammer (from Theory to practice)", Science Press, Beijing, China, 2009, pp.1–15.

[13] Jiří Mírovský, Netgraph, "A tool for searching in prague dependency treebank 2.0", Proceedings of the TLT 2006, pp.211–222.

[14] S.Loglo and Sarula, "A rule-based Mongolian dependency parsing model", International Journal of Knowledge and Language Processing, 2013, vol.4, No.3, pp.27–37.

[15] Jiří Mírovský, "Searching in the prague dependency treebank", Published by Institute of Formal and Applied Linguistics, Czech Republic, 2009.

[16] Laura Kallmeyer, "On the complexity of queries for structurally annotated linguistic data", Proceedings of ACIDCA, 2000, pp.1–6.