2012 International Conference on Information and Knowledge Management (ICIKM 2012)
IPCSIT vol.45 (2012) © (2012) IACSIT Press, Singapore

Comparison of Syntax Tree Visualization: Toward Malay Language (BM) Syntax Tree

Yusnita binti Muhamad Noor and Zulikha binti Jamaludin*

School of Computing College of Arts and SciencesUniversiti Utara Malaysia
06010 Sintok, Kedah, Malaysia

Abstract. This study will analyze natural language syntax tree visualizations to compare visualization methods in order to choose the optimum solution for visualizing a BM syntax tree. Currently no syntax tree visualization for BM has been introduced, and no visualization is yet available in the form of computer software or a prototype. Methods that can be dealt with in creating a BM syntax tree include: tokenizing, a performing search and comparison, matching with the associated rules, and composing. Ten systems were analyzed, and the Link Grammar system was found to be the most viable. The Link Grammar system does not have a hierarchical structure that reflects the language syntax as compared to the SSTC (Structured String-Tree Correspondence) application which does. However, the SSTC shows the tree structure in a hierarchical manner, but it does not have a suitable method to follow in visualizing the BM sentence syntax tree.

Keywords: Syntax tree, BM syntax tree, syntax tree visualization, BM sentence parser.

1. Introduction

Increasingly, computerization systems have been generated for information visualization, and some have been adapted to the field of linguistics, including word visualization applications named as SmartINFO and WordNet. While many researchers have focused on language study, less attention has been given to study of sentence structure or grammar visualization. One method that has been introduced to describe the structure of the sentence is in the form of a diagram and is better known as a syntax tree visualization. An example of this visualization is that which the SynView application performs.

The lack of emphasis researchers have given to processing the Malay language (BM) has been described in [9], [10] and [13] in articles about computational linguistics and natural language in Malaysia. This paper seeks to solve this problem, analyzing 10 different syntax trees visualization for English (BI) which might serve as a basis to develop equivalent syntax tree visualizations for Malay language (BM). However, to date, no BM syntax tree visualization has been introduced. A BM syntax tree visualization application would be an important contribution both to computational linguistics and to IT and would help create more IT-based applications improving Malaysia's technology advancement.

Several studies have carried out in designing of syntax trees visualizations for various purposes. Among them is a visualization made for BI for machine translation utilization. This visualization is known as the SSTC (Structured String-Tree Correspondence) syntax tree [1] and was developed to generate a syntax tree for machine translation usage that researchers utilized only in machine translation. Comparisons of syntax tree visualization systems are described in the next section. Every system involved was analyzed as a possible guide in developing syntax tree visualization for BM. The third section elaborates on rules and procedures in developing a BM syntax tree visualization; the summary is described in section four.

* Yusnita binti Muhamad Noor. Tel.: + (0192816740); fax: (-). *E-mail address*: s92715@student.uum.edu.my
Zulikha binti Jamaludin. Tel.: + (04-9284061); fax: (04-9284054/9284753). *E-mail address*: zulie@uum.edu.my

2. Comparison of Syntax Tree Visualization

There are two types of syntax tree; these are known as an Abstract syntax tree (AST) and a Concrete syntax tree (CST). AST is used for programming, and CST is used to analyze language structure [7]. From these two types, it is classified into two methods, namely the method of node-and-link which have tree nodes and arrows, and space-filling method that displays information structure in the form of visual presentation with a reliance environment [6] and [8]. Because this paper focuses on language structure, CST applications that used node-and-link method in presenting the language are analyzed.

Among the tools involved in visualizing sentence structure is SynView, which was developed by a group of students at the Ruhr University, Germany in 2009. VAST was developed in 2008, Linguistic Tree Contructor (LTC) in 2005, phpSyntaxTree in 2003, Lehner's prologue tree drawing in 1994, Link Grammar, TreeBuilder in 1991, syntax tree editor, RSyntax tree in 2009/2010, and SSTC in 1998. A comparison of these systems is shown in Table 1 below.

Table 1: Comparison of Syntax Tree Visualization

System	Year	Description	Input type	Weakness	Analysis on viability
RSyntax tree	2009/2010	RSyntax is based on the method used by phpSyntaxtree	Bracket symbol	Difficult for users who do not understand the format of writing input as in bracket symbol (Prolog symbol).	Unsuitable as a reference because of the input type and because no work has been published.
SynView	2009	SynView was developed using C++ and PERL [3].	Bracket symbol in notepad	Requires LaTeX software and external analyzer.	Unsuitable as a reference because it requires different software to write input and analyze the sentence. Also because no work has been published.
VAST	2008	VAST analyzes a sentence using a bottom-up approach. VAST uses global + detail and zoom techniques [2].	Input in XML file	Syntax analyzer and visualization are separated.	Unsuitable as a reference because VAST needs a different syntax analyzer and visualization.
LTC	2005	LTC is a tool to sketch the syntax tree, designed by Ulrik-Petersen Sandborg [14].	Bracket symbol	Phrase structure must be determined by the users and users need to sketch their own syntax tree after all the words are uploaded.	LTC is a system for sketching the graph of syntax tree, not for analyzing or visualizing text.
TreeBuilder	2004	Technique drafting graph that successive, free in sketch, user can print out or keep file as image.	Sentence	Users are required to sketch the syntax tree by input the sentence.	TreeBuilder is a system for sketching the graph of syntax tree, not for analyzing or visualizing text.
phpSyntax Tree	2003	phpSyntaxTree is an online application that allows users to draw a graphical syntax tree [4].	Bracket symbol	Difficult for users who do not understand the format of writing input as in bracket symbol.	Unsuitable as a reference because of the input need to be made in bracket (Prolog symbol) and no work has been published.
SSTC	1998	SSTC uses an example-based approach. Build several sub-tree and combine all the sub-tree in the final stage.	Sentence	Does not use any rules Sentences are matched with the database based on the examples of phrases that have been prepared.	Unsuitable as a reference because the resulting syntax tree is not divided into phrase structure and no rules are involved.

Table 1: Comparison of Syntax Tree Visualization (continued)

System	Year	Description	Input type	Weakness	Analysis on viability
				Analysis and visualizing the tree structure are designed for machine translation in which there are no phrase structure visualization are presented.	
Lehner's prolog tree drawing	1994	Lehner's requires users to input a sentence in Prolog format.	Prolog (bracket) symbol	Difficult for users who do not understand the Prolog format.	Unsuitable as reference because of the input need to be made in bracket (Prolog symbol) and no work has been published.

Link grammar	1991	Link grammar is context-free grammar formality. Every word in the lexicon is given a specific definition that describes how it can be used in a sentence [12].	Sentence	Link grammar is unable to analyze the conjunction. The syntax tree is not categorized into subject and predicate.	Matching techniques that are carried out namely: 1) Read every word 2) Performing search 3) Match with associated rules 4) Visualization The method used in analyzing the sentence is suitable to be referred and it uses rules in analyzing sentence.
Syntax tree editor	-	Software is designed to help linguist in designing syntax tree [5].	Bracket symbol	Users need to input phrase structure.	This is system to sketch the syntax tree but not to visualize it.

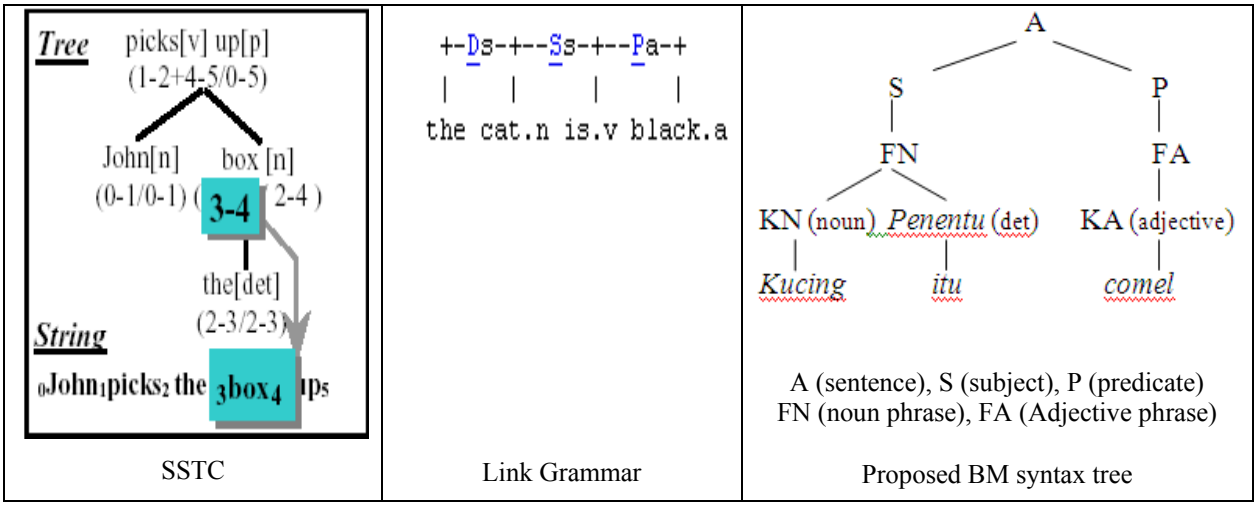
No publications carried out and input need must be in bracket / require different analyzer.

System to sketch the syntax tree

Table 1 above shows 10 different systems for visualizing syntax structure that researchers have developed. All of the systems are produced for BI, and only the RSyntax system can be used for Japanese, Chinese and Korean. The table also shows that eight of the systems do not have any implications for developing BM syntax trees. Among the systems, five of them do not have any publication that can be referred to. Three other systems were designed to help users to sketch a syntax tree by providing some menu and items. Thus, only two systems exist that can be used as a reference; these are SSTC and Link Grammar. However, SSTC analyzes sentence according to a method for sentence translation. In addition, dividing the sentence into phrases and producing a different syntax tree would require a complicated sequence of implementation and is produced specifically for machine translation. Thus, only Link Grammar system can be considered a suitable tool that can be referred to for processing a BM sentence. The method used is nearly identical to Rosmah's method in [11] that produced a BM sentence's checker and tree diagram which divides a sentence into subject and predicate. This method analyzes the sentence by 1) tokenizing, 2) performing search, and 3) matching words with associated rules.

3. Procedures and Rules for BM Syntax Tree Visualization

The Link Grammar system will be referred to in developing a visualization tool for BM syntax tree. BM syntax tree visualization divides the sentence into subject and predicate that will have parent (top) and child (bottom) nodes namely in the hierarchical diagram. Thus, by referring to the Link Grammar system, the BM syntax tree visualization will process the sentence by 1) tokenizing, 2) giving a certain word class to each word, 2) performing a search and comparing, 3) matching it with the associated rules, and 4) visualization.



SSTC | Link Grammar | Proposed BM syntax tree

Fig. 1: Example of syntax tree

Fig.1 shows examples of syntax tree design such as SSTC, Link grammar, and the proposed BM syntax tree. The method used by Link Grammar will be referred to in developing the BM syntax tree visualization, and the structure designed in placing the node and arrow will refer to the method used by SSTC. Fig. 1 also shows that Link Grammar system matches each word with the associated word class but it does not divide the visualization into subject and predicate that should be in a hierarchical structure. SSTC system did not have a suitable method to follow, but the tree structure is arranged in a hierarchical design as that which will be done in BM syntax tree.

For example, the BM sentence "*saya makan nasi*" will be analyzed as below:

Step 1: tokenizing

saya | *makan* | *nasi*

Step 2: word class

saya = KN, *makan*=KK, *nasi*=KN

Step 3: Matching with rules

Example of rules: A=S+P

S=FN

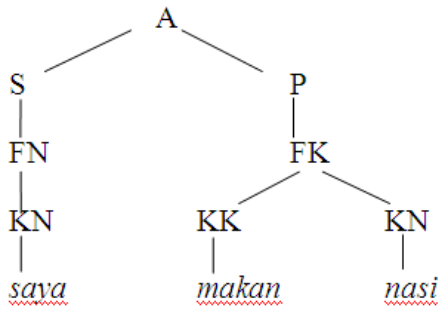
P=FN/FK/FA/FS

FN=(*Bilangan*) + (*Penjodoh Bilangan*) + (*Gelaran*) + *Kata Nama*+(*Kata Nama*) + (*Penentu*) + *Penerang*

S=FN (*saya*)

P=[KK (*makan*), KN (*nasi*)] FK

Step 4: visualization



BM syntax rules have four different phrase structures including: FN (*frasa nama*), FK (*frasa kerja*), FA (*frasa adjektif*), dan FS (*frasa sendi nama*). The sentence or *Ayat* (A) will be divided into Subject or S (*subjek*) dan predicate or P (*predikat*). Every phrase will have a word class or combination of several word classes like KN (*kata nama*), KK (*kata kerja*), KA (*kata adjektif*) and KS (*kata sendi nama*) as listed above.

4. Conclusion

Ten tree systems focusing on the syntax tree for natural language as a basis in developing a BM syntax tree visualization were analyzed. Comparisons were made between these 10 systems. The Link grammar system has an implementation model that can be referred to because the beginnings of the process in analyzing the sentence are same with the BM sentence checker that Rosmah produced [11] which shows the sequence on how to analyze BM sentence. In addition, the input type in sentence also became selection criteria as compared to the other systems which requires Prolog symbol as input.

Models and algorithms for BM syntax trees visualization will be designed based on the analysis of BI syntax tree as described in Section 2. In addition, the visualization can be done only if the input sentences

have a correct structure according to the rules the *Dewan Bahasa dan Pustaka* has issued. This means that BM sentence's checker also needs to be designed.

5. References

[1] M. H. Al-Adhaileh and T. E. Kong. "A flexible example-based parser based on the SSTC," in *Proc. COLING '98 Proceedings of the 17th international conference on Computational linguistics*, 1998, pp. 687-693.

[2] F. J. Almeida-Martinez et al.." Visualization of Syntax Trees for Language Processing Courses." *Journal of Universal Computer Science*, vol. 15(7), pp. 1546-1561. 2009.

[3] C. Behrenberg. (2009). SynView v0.3 user's manual [Online]. Retrieved Dec 22, 2010, Available: http://www.christian-behrenberg.de/files/SynView/SynView_source.rar,

[4] M. Eisenbach and A. Eisenbach. (2003). phpSyntaxTree-drawing syntax trees made easy [Online]. Retrieved Dec 20, 2010, Available: http://www.ironcreek.net/phpsyntaxtree/

[5] J. Epstein and E. O'Neill. (n.d). Syntax Tree Editor [Online]. Retrieved Dec 20, 2010, Available: http://www.ductape.net/~eppie/tree/index.shtml

[6] B. Johnson and B. Shneiderman. "Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information," in *Proc. of Visualization,* 1991, pp. 284–291.

[7] J. Jones. (2003). Abstract Syntax Tree Implementation Idioms [Online], Retrieved Dec 18, 2010, Available: http://hillside.net/plop/plop2003/Papers/jonesimplementingasts.pdf

[8] M. Luboschik and H. Schumann. "Explode to Explain – Illustrative Information Visualization," in *Proc. 11th International Conference Information Visualization (IV'07)*, 2007, pp. 301-307.

[9] M. J. Ab Aziz. (2007). Pengkomputeran Linguistik Bahasa Malaysia [Online]. Retrieved Dec 28, 2010, Available: http://www.ftsm.ukm.my/programming/prosiding-atur07/08-Juzaiddin.pdf

[10] N. Abu Bakar et al..(2006). Penggunaan komputer dalam pengajaran bahasa [Online]. Retrieved Dec 28, 2010, Available: http://202.28.66.7/smuhammad/pdf/Penggunaan%20Komputer%20dlm%20pengajaran%20bahasa.pdf

[11] R. Abdul Latif, "Penyemak Sintaksis Ayat Bahasa Malaysia," M.S. thesis, Universiti Kebangsaan Malaysia, Bangi, Selangor, 1995.

[12] D. Sleator and D. Temperley. (1993). Parsing English with a link grammar [Online]. Retrieved Dec 29, 2010, Available: http://www.cs.cmu.edu/afs/cs.cmu.edu/project/link/pub/www/papers/ps/LG-IWPT93.pdf

[13] S. Ramli, "Reka bentuk dan implementasi suatu penghurai bahasa Melayu menggunakan sistem logik selari," M.S.thesis, Universiti Putra Malaysia, Selangor, 2002.

[14] U. Sandborg-Petersen**.** (n.d). Linguistic tree constructor [Online]. Retrieved Dec 20, 2010, Available: http://ltc.sourceforge.net/