Automatic Generation of Fill-in Clues and Answers from Raw Texts for Crosswords

Bali Ranaivo-Malançon, Terrin Lim, Jacey-Lynn Minoi, Amelia Jati Robert Jupit
Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak
Kota Samarahan, Malaysia
{mbranaivo, terrin, jacey, rjajati} @fit.unimas.my

***Abstract* — This paper presents a method to generate fill-in clues and answers for building automatically a crossword. Answers are capitalised words present in an input sentence and clues are segments of the dependency syntactic structure of that sentence. The pairs (Clue, ANSWER) are extracted from a collection of raw sentences related to the history of Sarawak. This work is at its early stage, and thus the proposed method that generates automatically fill-in clues, was tested on a small set of sentences and the obtained results are promising. Near 53% of the generated fill-in clues are considered correct. The major contribution of this work is the innovative strategy used to read the result of a pre-order depth-first search applied on a dependency graph to generate the clues. The clues and answers generator is implemented in Python.**

Keywords — thematic crossword; fill-in clues generation; Sarawak history

I. INTRODUCTION

Almost a century exists between the first known published crossword puzzle, in short crossword, in 1913, by Arthur Wynne and the first fully automatic crossword generator in 2008 [1].

One of the major challenges in creating a crossword, either by human or by machine, is to find the clues that correspond to an answer. This paper presents a method to tackle the problem. The approach uses the syntactic dependency analysis of raw sentences and the pre-order depth-first search on the corresponding graph to generate clues and answers. Thus, it falls under the sub-component "automatic clues and answers generation" and does not touch the other components illustrated in Fig. 1, which means that the crossword layout generation (or grid generation) is out of the focus of this paper.

The work presented in this paper is part of a large project that aims to generate automatically crosswords that are related to the history of Sarawak. Therefore, the crossword is a thematic or domain-specific crossword. As defined by the dictionary WordWeb, a history is "a record or narrative description of past events". History is part of the heritage that one country and each citizen of that country should preserve as it conveys information over centuries. The history of one country is unique and irreplaceable. However, history is often narrated from different perspectives. As such, it is presented in various ways with multiple versions on several kinds of materials. For this paper, the elements of the history of Sarawak that are of interest for generating automatically clues and answers were extracted from raw texts crawled from the Web. Then, the challenging task is to identify and extract from those texts the segments that can be used as clues and answers.

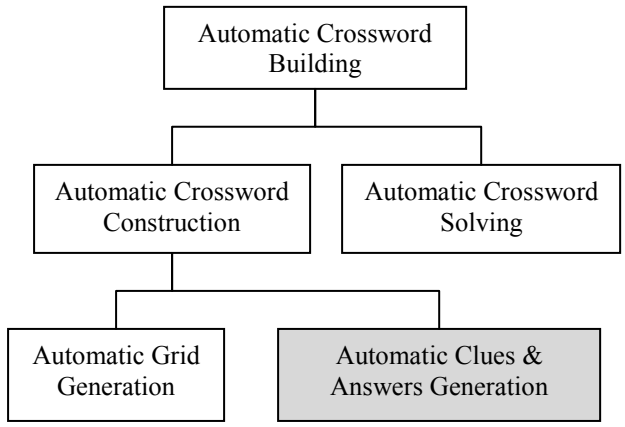

Fig. 1. Automatic crossword building components

Thematic crosswords like the history of Sarawak crosswords can be used as an instructional game. Users are given a different way of learning the history. They can review, test, and update their knowledge through a lexical game. Teachers can use this kind of thematic crosswords to evaluate or improve students' knowledge about history: how to spell, who did what, what happened at that specific date and location, and so on. But for most users, playing crosswords is just a hobby.

Section II recapitulates the research done so far on the automatic generation of crosswords. Section III describes the proposed method in generating automatically clues and answers from raw sentences. Section IV presents the experiment results as well as the analyses of those results. Section V concludes the presentation with a mention to the future works.

II. RELATED WORK

The majority of existing works solves the problem of creating clues by looking for definitions either from existing

978-1-4799-1092-2/13/$31.00 ©2013 IEEE

dictionaries or thesauri or from the analyses of sentences found on the Web.

In 2008, [1, 2] presented the first fully automatic crossword generator: from collecting definitions (or clues) to the crossword solving using Constraint Satisfaction Programming. They crawled the Web to look for definitions. The pair (Clue, ANSWER) corresponds to (definition, subject). A definition is recognised based on a pre-defined constituency structure: "subject + nominal predicate + complements". The constraint put on the structure of a definition may limit the space for other potential clues. Like the proposed approach in the current paper, the Web is used to find documents related to the theme of the crossword. However, the structures of the definitions (or clues) are totally open as explained in section III. The evaluation of the crossword generation system in [1] showed that 81% of the definitions were classified as correct. It is not surprising to get such high level of perfomance since a definition corresponds to a pre-defined syntactic structure. As shown in Table II, the clues generator presented in this paper did not reach such level of correctness as the system does not work with any pre-defined syntactic structure for the clues.

III. PROPOSED METHOD

The workflow in generating clues and answers from the Sarawak historical raw texts is illustrated in Fig. 2. The first step corresponds to the acquisition of a collection of texts related to the target topic, which is the history of Sarawak. Once the collection is acquired, it needs to be pre-processed to extract all sentences. Then, each sentence is analyzed to determine its dependency syntactic structure and to extract all capitalized words that are not in a stop list. Then, the output of the dependency parser is transformed into a graph for pre-order depth-first search. Clues are generated for each capitalized word based on a proposed method for reading the depth-first result.

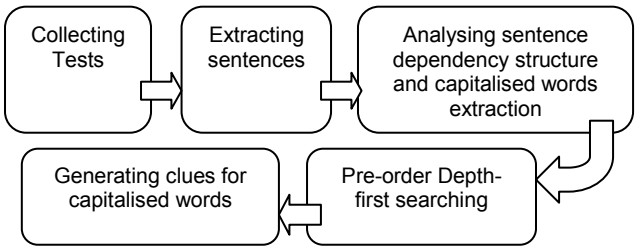

Fig. 2. Workflow of the proposed method

A. Thematic Corpus Building

There are two ways in building a thematic corpus: crawling the Web or digitizing documents. The objective of the crawling is to search and collect Web pages that relate the history of Sarawak. Unfortunately, there are not so many documents on the history of Sarawak on the Web. If one provides Google search engine with the keywords, "history + Sarawak", the number of hits will be around 9.5 million compare to number of hits for "history + France", which is near 1.7 billion. Many of these Web documents narrate the same events, the period of the White Rajahs.

B. Sentence Extraction

Sentences are extracted from the collected texts. Titles and sub-titles of the texts are discarded manually. Sentences are recognized automatically by their structures. They correspond to a sequence of characters that starts with a capital letter and ending with one of the following separators, full-stop, exclamation mark, and question mark.

C. Sentence Analyses

The analyses of an input sentence are the syntactic dependency parsing and the capitalized words extraction.

a) Extracting Capitalised Words as Answers: The extraction of capitalised words rely on a list of words (or stoplist) that contains most of English function words. The algorithm is described as a pseudocode in Fig. 3.

```
Function capitalised_word(word):
        if word is in stoplist
                return False
        if word starts with a capital letter
                if word is not a Roman
numeral
                        return True
```

Fig. 3. Capitalised Words Extraction Algorithm

The condition for checking if the input word is either a Roman numeral or not must be done as historical documents contain many of these strings.

In the input sentence in Fig. 4, the capitalized words are "Sarawak", "Portuguese", and "Cerava". The word "The" is discarded since it is in the stop list.

b) Generating Syntactic Dependency Structures: There are many existing dependency parsers that work well on English sentences. One of them is the Stanford dependency parser[1]. Given an English sentence, the parser outputs a list of triplets corresponding to the dependency relations that exist between pairs of words. An example of such analysis is shown in Fig 4.

[1]The Stanford Parser: http://nlp.stanford.edu/software/lex-parser.shtml (last visited 18 March 2013).