Computer-Aided Word Mastermind Solving Using Regular Expression Pattern Matching in a Directed Acyclic Word Graph

J. L. Bahinting, J. D. G. Bonos, C. J. A. Delfinado, M. M. Ignacio, and L. F. Lachica

Abstract—**Humans solve the Word Mastermind game inefficiently. This study developed an efficient helper system that reduces the average number of attempts and the amount of time needed to solve the game. The system used a Directed Acyclic Word Graph (DAWG) to manage words lists, Regular Expression (Regex) pattern matching for searching and retrieving words based on a specific pattern on the DAWG, and relative letter frequencies for scoring words based on the results of the pattern matching. The system nominates a list of candidate words suitable for quickly finding the game's mystery word. The system helped humans reduce their average number of attempts to four (4) or five (5) and with an average time of fifty (50) seconds.**

Index Terms—**Directed acyclic word graph, pattern matching, puzzle solving, regular expression.**

I. Introduction

Over the years, the cognitive process of humans in problem-solving and how it can be imitated by computers have been the focus of Artificial Intelligence (AI) researches. Using computers, it is possible to achieve human-like behavior in nonhumans [1].

Word puzzles or language games draw their challenge and excitement from the richness and ambiguity of natural language [2]. From an AI perspective, these puzzles or games are interesting area of study since the success in solving them is clearly defined by how well the system does in producing the right answer [1]. The researchers of this study aimed to improve the efficiency in terms of the average number of attempts and amount of time being consumed by humans in solving the Word Mastermind game.

This study focused on creating a computer aid to humans as they play. To win the game, the computer-aided application will help players give meaningful guesses by means of suggesting words that fit the clue from the previous try. It involved the use of a Directed Acyclic Word Graph (DAWG), a lexicon data structure to manage the word list; and with the Regex pattern matching, the word search through the DAWG was improved. Aside from the data structure and the search algorithm, a scoring method using relative letter frequency aided the ranking of words according to how close they are to the correct answer of the word puzzle.

Manuscript received December 19, 2012; revised February 23, 2013. This work was supported in part by the Faculty of Engineering of the University of Santo Tomas. Computer-aided Word Mastermind Solving Using Regular Expression Pattern Matching in a Directed Acyclic Word Graph.

The authors are with the Department of Information and Computer Studies, Faculty of Engineering, University of Santo Tomas, España, Manila 1008 Philippines (e-mail: tinbahinting@yahoo.com, jg.bonos@gmail.com, cecild99@hotmail.com, ignacio_louis@yahoo.com, llflachica@yahoo.com)

II. Background and Related Studies

Most people have probably experienced playing games that involved words which are enjoyable and can elicit excitement since one can get to use his deductive reasoning and vocabulary. There is a difference with the way humans and computers "think" in terms of solving word puzzles. Humans sometimes rely on intuition or their gut-feel to choose letters that might produce a correct or close to correct guess, especially when they have little knowledge about the mystery word. Computers, on the other hand, depend on how they were programmed to respond to a problem. A computer operates in a systematic way since it can follow heuristic procedures. Computers have the potential for having many roles in the field of Logology or the study of recreational linguistics [3]. Aside from making the list of words and storing them, computers can help in solving word puzzles like Word Mastermind.

Word Mastermind [4] is a simple yet challenging word guessing game. It is a variation of the classic game, Mastermind. But instead of guessing colored pegs, players must guess a four-letter word, making the game harder. Instead of only six colors, there are 26 possible letters for each space on the board. From the vast vocabulary the English language has, it is quite a challenge for players to quickly answer the problem of the game with the least number of trials and in the fastest way possible.

The simplest Word Mastermind game starts with the player giving a four-letter word guess. His guess must be a valid four-letter word in the dictionary. After the guess has been made, the game master will tell what letters from the player's guess are correctly placed and what letters are correct but not in their proper place. The clues given would still depend on the corresponding correct hits (correct letter and in place) or misplaced hits (correct letter but misplaced). The game ends once the player has guessed the mystery word or the player was unable to guess the word.

Logology or the study of Recreational Linguistics covers word games and wordplays that stress letter patterns. Some roles of Computers in Logology are: Word list creation, Logological database management and solving word puzzles. Computers are used for storing large amount of data. To obtain a database of words, it is necessary to type all the words in a dictionary. The database can be managed and can be compressed depending on what kind of data structure to be used. It is important to have an effective way to manage Logological databases to minimize the time consumed in searching and producing results. Computers may not be that very helpful in solving some word puzzles but they can still provide some assistance especially in performing frequency counts and other numerical tests [5].

Puzzles and games were used in Artificial Intelligence (AI) researches. They involve problem solving, which relates analysis in AI. Search algorithms are vital in problem solving because searching is the task of exploring possible solutions, evaluating them and subsequently arriving at the best solution. Uninformed search (or brute-force search) and informed search algorithms (or heuristic search) were compared and the informed search was said to outperform uninformed search since it takes advantage of the information about the problem. It uses heuristics to significantly minimize the search [6].

The computer-aided solving of the Word Mastermind game employed an informed search technique in problem solving that it utilized all the gathered information (the clues; in the case of Word Mastermind) to guide the search and also to reduce the search space. Because this study was primarily dealing with words, a data structure must manage the lexicon. The lexicon used by the computer-aid was represented by DAWG. A DAWG is very close to a Trie structure but more space efficient [7]. It is represented as a directed acyclic graph with a vertex with no incoming edges, in which each edge of the graph is labeled by a letter or special end-of-string marker, and in which each vertex has at most one outgoing edge for each possible letter. The strings represented by the DAWG are formed by the letters on paths in the DAWG from the source vertex to any vertex with no outgoing edges [8], [9]. The primary difference between DAWG and trie is the elimination of suffix redundancy in storing strings. The trie eliminates prefix redundancy since all common prefixes are shared between strings. In a DAWG, common suffixes are also shared [8], [9]. For dictionary sets of common English words, this translates into major memory usage reduction. Thus, the DAWG fared better in terms of compactness and the word search performance.

Pattern matching is the method of finding a string containing text or binary data for some set of characters based on a specific search pattern. It describes a pattern and includes a language which utilizes various symbols in describing both the normal characters (those that are exactly matched) and meta-characters (those that describe operations such as repetitions and alternatives) [10].

Pattern matching was used to optimize the search through the DAWG structure. The patterns served as the "heuristics" that directed the search and consequently narrowed down the search. Studies showed that Regex [11] pattern matching is efficient and can eliminate needless shifts, and is powerful and flexible since it can search for words of a certain size, find different sets, search for patterns, print the line satisfying the given pattern, and replace the string with a new pattern. But it has a certain drawback in terms of the accuracy in the computation of transition function.

The frequency analysis for computing of the most prevailing word that can be formed was applied. Almost all word guessing game consider the knowledge of letter frequency as a strategy.

III. Word Mastermind Helper

The researcher's computer aid allows players to improve their performance in playing Word Mastermind by minimizing the number of trials it took to arrive at the correct word and to make the process of guessing possible words less time consuming.

IV. System Architecture of Word Mastermind Helper

The system architecture of the Word Mastermind Helper is comprised of five modules: DAWG Constructor, which performs the preprocessing, Regex Pattern Generator which outputs the Regex pattern that guides the search for possible solutions to the puzzle, Regex pattern matcher which is the module in charge of getting all words that fit the pattern, Letter Frequency Counter which counts the occurrence of each letter in the words list, Scorer which computes the scores of each possible solutions to the puzzle and ranks them in descending order of scores. The Preprocessing involves the creation of a DAWG from the list of valid words in the Dictionary. The DAWG, including the player's guess and the corresponding clue for that guess, are the input of the Regex Pattern Generator. The generator outputs the regex pattern which in turn used by the Regex Pattern Matcher. The pattern matcher produces the matched words list or the candidate words list. The Letter Frequency Counter processes the candidate words list and output the letter frequencies. The data that is from the frequency counter is used by the Scorer. Results are displayed by means of a graphic user interface.

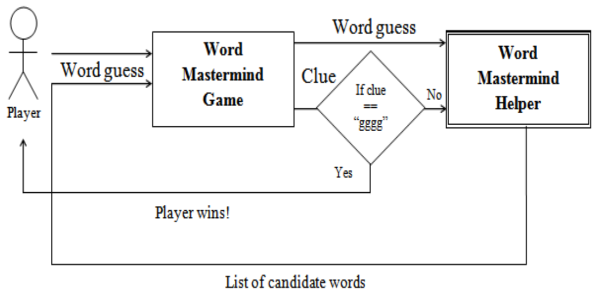

Fig. 1. Word mastermind game interfacing with the word mastermind helper

A. DAWG Constructor

The DAWG Constructor has four major processes inside it namely, *Construction of a Temporary Trie* – filling the temporary trie with the words in the list, *Traversal of Trie to Track and Kill "redundant" Nodes* – Breadth-First Search thru the trie to record these nodes and removing these nodes, also fixing the pointers (the working DAWG is now ready), *Bit Encoding* – for the compression of the lexicon (make it less space consuming) and *Population of DAWG* – populate the DAWG and write it to file. In Fig. 2, the constructed DAWG is one of the required inputs for the Regex Pattern Matcher.

B. Regex Pattern Generator

In Fig. 1, the player's word guess and the clue given by the game is the primary input of the Word Mastermind Helper. The pattern generator module facilitates the creation of the regex pattern for each word guess based on the clue. For simplicity, an input of ‘g’ is for correct letter and correct place; ‘y’ is for correct letter but incorrect place; and ‘w’ is for incorrect letter. The letters ‘g’ (green), ‘y’ (yellow) and ‘w’ (white) corresponds to the first letter of each color clue in the Word Mastermind game. An example of a regex pattern is: “[^air][^ai][^ai]n”. The word length is four so the maximum level or depth to explore is also four. The "^" indicated an exclusion of the characters succeeding it. When traversing the DAWG, words that do not start with the letters "a", "i" and "r", having the second and third letter that is not an “a” and “i” and the fourth letter which is an “n” are the ones considered as candidate words.

C. Regex Pattern Matcher

The process in this module is started by loading the DAWG data file into the memory. Methods for extracting information in the DAWG data file are also inside the pattern matcher. Once the data file is loaded and the regex pattern is passed, the pattern is inspected. This is done to make sure that places where constraints must be observed are noted. A recursive process of searching the DAWG and getting the candidate words is done after the inspection. When searching has already finished, all the candidate words will be stored.

D. Letter Frequency Counter

Since the basis for assigning each word’s score is letter frequency, a letter frequency counter is implemented. The counting of the occurrence of each letter in the matched words list is handled by this module.

E. Scorer

The assignment of scores for each word in the matched words list is facilitated by the scorer. The higher the score meant the higher the possibility of obtaining a correct letter after it is being considered as the next guess. The simple formula is: (a+b+c+d)/total number of letters; where a, b, c and d are frequency of each letter that makes up the word. If there is a repeated letter in the word, the letter count for that letter is only added once.

V. Research Design and Methods

The researchers hypothesized that the integration of Directed Acyclic Word Graph, Regex pattern matching and relative letter frequency to create a word puzzle helper will improve the efficiency of solving the Word Mastermind game. In line with this, the number of attempts and amount of time consumed aimed to be reduced.

The researchers used an experimental approach in this study. It comprises of two (2) classifications of participants namely the control group and the experimental group. To get all of the necessary quantitative data needed for the study, the researchers implemented a systematic way of letting the control group play Word Mastermind without the computer aid, and recorded their performance afterwards. Also, when the word puzzle helper was already working and has passed functionality testing, it was used while playing the game. The study covered a number of 100 players, 50 students for the control group and another 50 students for the experimental group, each subject played five times to be familiar with the game and another five times to achieve mastery.

The independent variable used in this study was the use of a word puzzle helper (computer-aided solving) while the dependent variable used was the efficiency of solving the Word Mastermind game.

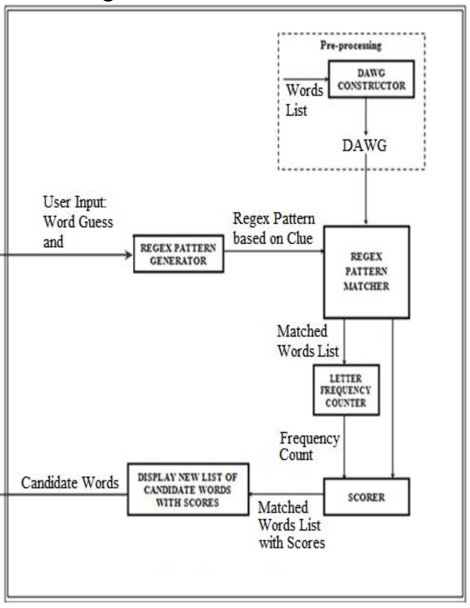

Fig. 2. Word mastermind helper.

VI. Results

The computer aid was tested in terms of execution time, space and reliability to obtain the overall performance of the application.

The results of the test regarding execution time indicated that 55.63% of the time was spent in the GUInterface class, 30.67% to the PatternMatcher, 10.4% to the Scorer class, 0.87% to the LetterFreqCounter, 0.7% to PatternGenerator and 1.73% to other classes. Test for the preprocessing stage was conducted separately.

The test conducted in terms of space consumption showed that out of the 16MB heap size, the application used, at maximum, 10MB of the heap size or 62.5% of the heap size. Storing the lexicon in a DAWG proven that DAWG can be an effective data structure for storing words.

To test for its reliability, eight different test cases were prepared. The test cases were: random words, words with highest, mid and lowest scores of letter frequency, words with repetition, permutations and plenty of vowels and consonants.

The test result showed that four-letter words with at least 3 consonants have the least number of attempts and the least length of time required to arrive at the solution with only an average of five (5) attempts and 29.91 seconds respectively; while four-letter words with permutation have the most number of attempts and the longest length of time spent with an average of seven (7) attempts and 45.98 seconds respectively.

Additional tests were done to find how effective the computer aid were when the length of the words used is greater than 4.

The results showed that sixteen-, seventeen- and eighteen-letter words have the least number of attempts required to arrive at the solution with only an average of two (2) attempts; while five-, six-, seven-, eight- and nine-letter words have the most number of attempts with an average of four (4) attempts. In addition, the seventeen-letter word test has the least length of time required to arrive at the solution with an average of 6.69 seconds; while five-letter word test has the longest length of time spent with an average of 25.26 seconds.

Researchers also gathered data with regards to playing the game with and without the helper. Based on the number of attempts, the average for without helper is 6.9 attempts while for with helper is 5.0. Based of accumulated time, the average for without helper is 178.4 seconds and for with helper is 49.6 seconds.

VII. Conclusion

Word Mastermind Helper produced results showing an improvement on the efficiency of solving the Word Mastermind game. It reduced the number of attempts and amount of time consumed as compared to playing the game unassisted by the helper. For four letter words, the computer aid showed an average of 4.984 attempts to correctly guess the mystery word; whereas based on the accumulated time, the computer aid showed an average of 50 seconds to correctly guess the mystery word.

The following were the findings of the study: (1) Utilizing the Directed Acyclic Word Graph helped stored the words more efficiently than storing them on a text file. For a significantly large lexicon, at maximum, the percent reduction could be up to 42%. With the Pattern Matching technique applied, searching for words was faster as compared to the brute-force method. Since brute force cannot support well searching with certain criteria to follow (the "clues"), the Pattern Matching technique used in this study beats it. (2) Through the strategy of maximizing the information obtained from the clues, the computer aid gave meaningful suggestions as all the correct letters acquired were retained and used until the correct word is guessed. (3) The computer aid was capable of processing words whose length is greater than 4, supporting up to 21 words which is likewise efficient as processing 4 letter words.

Acknowledgement

The authors would like to express their deepest and warmest gratitude to the individuals who have, in one way or the other, helped in the success of this research

To JohnPaul Adamovsky, who shared his idea and helped them in understanding better some of the concepts used in this research;

To their friends, mentors, colleagues and to all who have been part of their lives;

And above all, to God, for without Him, all of this will not be possible.

References

[1] J. Schaeffer, "Trends and controversies: The role of games in understanding computational intelligence," *IEEE Intelligent Systems*, vol. 14, no. 6, pp. 7-10, Nov. 1999.

[2] M. L. Littman, G. A. Keim, and N. Shazeer, "A probabilistic approach to solving crossword puzzles," *Artificial Intelligence*, vol. 134, pp. 23-55, Jan. 2002.

[3] A. Frank. (1983). Logology by computer. [Online]. 16(4). Available: http://digitalcommons.butler.edu/wordways/vol16/iss4/2/

[4] Board Game Geek- Word Mastermind. [Online]. Available: http://boardgamegeek.com/boardgame/5662/word-mastermind

[5] A. R. Eckler. (1980). Crosswords and the computer. *Word Ways*. [Online]. 13(4). Available: http://digitalcommons.butler.edu/wordways/vol13/iss4/10/

[6] R. C. Chakraborty. (2010). Problem solving, search and control strategies: Artificial intelligence. [Online]. Available: http://www.myreaders.info/html/artificial_intelligence.html

[7] A. W. Appel and G. J. Jacobson, "The world's fastest scrabble program," *Communications of the ACM*, vol. 31, no. 5, May 1988.

[8] M. Balík. (2002). Implementation of directed acyclic word graph. *Kybernetika*. [Online]. 38(1). pp. 91-103. Available: http://dml.cz/dmlcz/135448

[9] J. Daciuk, B. W. Watson, S, Mihov, and R. E. Watson, "Incremental construction of minimal acyclic finite-state automata," *Computational Linguistics*, vol. 6, pp. 3-16, March 2000.

[10] Wikipedia- Pattern Matching. [Online]. Available: http://en.wikipedia.org/wiki/Pattern_Matching

[11] P. Linsley and B. Trute. (2003). Introducing oracle regular expressions. [Online]. Available: http://www.oracle.com/technetwork/database/focus-areas/application-development/twp-regular-expressions-133133.pdf

J. L. Bahinting was born in Quezon City, Philippines on October 19, 1992; is an undergraduate of Bachelor of Science Major in Computer Science at University of Santo Tomas (UST), Manila, Philippines. She was an intern at NV Besix SA Philippine Branch. She was assigned in the IT department and assisted in company's system documentation and development as well as in quality assurance testing. Ms. Bahinting is also a proud member of the Computer Science Society and Becarios de Santo Tomas. Some of her achievements include being a San Martin de Porres scholar and a consistent Dean's Lister.

J. D. G. Bonos was born in Sta. Cruz, Manila, Philippines on October 20, 1992; is currently a graduating student taking up the degree of Bachelor of Science in Computer Science in the University of Santo Tomas (UST), Manila, Philippines. He worked as an Intern Programmer focusing on the analysis and development of business applications, and assisted in enhancing and testing existing programs at INFOMAN, Inc. in Makati City, Philippines. Mr. Bonos is a member of the Computer Science Society, the Junior Philippine Computer Society (JPCS), the UST Mountaineering Club and the UST Yoga Club in the said institution.

C. J. A. Delfinado was born in Manila, Philippines on December 9, 1966; obtained his Master of Science degree in Mathematics at University of the Philippines, Diliman, Quezon City, Philippines. He also obtained his Master of Science degree in Computer Science at the University of Illinois, Urbana-Champaign, IL, U.S.A. He is currently a faculty member in the Information and Computer Studies Department, Faculty of Engineering of the University of Santo Tomas (UST), Manila, Philippines. He teaches Mathematics and Computer Science courses such as Discrete Math, Compiler Design, Automata Theory, Data Structures, Algorithms, and File Organization.

M. M. Ignacio was born Sta. Cruz, Manila, Philippines on February 25, 1992; is currently an undergraduate of Bachelor of Science in Computer Science at University of Santo Tomas (UST), Manila, Philippines. He had his internship at CONEX, Mandaluyong, Philippines. He was assigned in the IT and Accounting departments specializing in Systems Analysis and Design and various computer applications in Accounting respectively.Mr. Ignacio is the Vice President for Administrative Affairs of Becarios de Santo Tomas as well as the Fourth Year Level Representative of Computer Science Society in the said university. He is a Rector's Scholarship grantee and a Philippine-Tohoku Goodwill Ambassador to the JENESYS Programme 2012.

L. F. Lachica was born in Sta. Mesa, Manila, Philippines on January 18, 1993; is currently in her fourth and last year in University of Santo Tomas (UST), Manila, Philippines for her Bachelor degree in Computer Science. She completed her undergraduate internship at Accenture, Mandaluyong Branch, Philippines. She was assigned in CIO Operations-Platform Engineering department and participated in the development of a system for Accenture's global users.Ms. Lachica is affiliated with the UST-Faculty of Engineering Computer Science Society. She is also an IBM Academic Associate for passing the DB2 9 Database and Application Fundamentals certification examination.