Scrabble Game Using Java

Manasi Mhatre[1], Sakshi Nagaonkar[1], Sminil Shirsat[1], Pournima Kamble[2]

[1]*Student, Computer Technology, BVIT, Navi Mumbai, India.*

[2]*Professor, Computer Technology, BVIT, Navi Mumbai, India.*

Corresponding Author:sminil232003@gmail.com

Abstract: - This scrabble game is a word game, which is based on cross words. So typically it is also called as cross words game. There will be a grid of 15×15 size or max. Users have to move characters to up or down to get a correct word. It is an interesting game to play. It can be used for improving students' knowledge in some terms. This project describes a Java implementation of the well-known board game Scrabble. The application was written to be played by a human and computer controlled player.

Key Words: —*Scrabble game, java, board game.*

I. INTRODUCTION

Scrabble is a commonly played word game in which players take turns forming words using a set of seven letter tiles and placing them onto a grid, following placement rules similar to a crossword puzzle. Points are awarded based on the sum of values of individual letters, along with letter and word multipliers positioned regularly throughout the board. While turns are not usually timed, creativity and vocabulary are both important to achieving a high score. With the increasing popularity of Scrabble-like online games like Words with friends, various websites like wordfind.com have been created to help players create words from their set of tiles.

II. LITERATURE SURVEY

One of the earliest Scrabble programs released was a commercial game called MONTY released in 1983. Expert human players were able to beat MONTY consistently but said it posed a significant challenge. According to Scrabble Players News, MONTY was designed to use both strategic and tactical concepts [1].

Peter Turcan from the University of Reading developed an early computer based Scrabble player [2][3] . In his implementation he stores the lexicon as a list of words in reverse order of length. For each word he decides whether and where the word can be played on the board with the given rack.

Manuscript revised May 28, 2021; accepted May 29, 2021.
Date of publication May 30, 2021.
This paper available online at www.ijprse.com
ISSN (Online): 2582-7898

Unlike many other early Scrabble programs, his move evaluation function takes simple strategic features of the board and tiles left on the rack into consideration Most other programs at the time used a greedy algorithm for move evaluation, simply choosing the move with the highest point yield.

Stuart Shapiro of SUNY Buffalo has implemented several Scrabble programs [4]. His implementations use a tree structure of letters to represent the texicon, where each path down the tree has an associated list of words which can play using the letters on that path. The tree is ordered with higher scoring letters appearing higher in the tree so that higher scoring words are found first, in first a full search cannot be completed in the given time. His move evaluation takes the first move found with a score higher than some threshold.

Andrew Appel and Guy Jacobson discuss an algorithm they developed what allows for rapid move generation in Scrabble [5]. Their implementation uses Direct Acyclic Word Graph (DAWG) to represent the lexicon. They discuss how their program analyses the board in terms of anchor squares and cross sets. A "backtracking" algorithm is proposed which generates moves from anchor squares by constructing word prefixes to the left, and then generating words by constructing corresponding suffixes to the right. Their program uses a simple evaluation impletion but they suggest at the possibility of improving performance by using a sophisticated heuristics function.

III. PROPOSED SYSTEM

Scrabble is a game which is commonly played by millions of people around the world. It is a game that blends vocabulary and strategic skills and makes an interesting AI (Artificial Intelligence) project idea. Game will be developed using Java

implementation. Java is one of the most used programming language used today and has its application across wide range of platforms like GUI (Graphical User Interface), design to World Wide Web.

- It could be played by one or more people.
- There could be a time limit.
- There are variety of ways in which scores can be calculated.
- When a tile is used, 'consumed' even, it could be replaced with a fresh tile.
- Any form of the game that is desired may be implemented.
- The baseline is a single player game where the player simply seeks to get a good score.
- User friendly interface and easy to access.

IV. System Design And Components

Dictionary:

Dictionary class stores the games dictionary. Dictionary object is constructed using a word list provided by the txt. Dictionary can be easily changed by swapping out this "Dictionary" file for another.

Tile:

The Tile class describes the different letter tiles used in the game. Each letter is encoded with its corresponding score. It also has a function to pull the image for each letter tile to be used for the game's score. The program can request the score of a tile by passing this class the character for that letter.

Bag:

The Bag class is used for drawing tiles during the game. The official letter distribution is read in from a ".txt" file when the program starts.

Board:

The Board class describes the game board. It is composed of 225 proprietary square objects arranged in a 15 x 15 grid. It has functions used to validate and score player moves, assess the board for anchor squares and cross sets. It is initialized by reading an ASCII map which is stored as a ".txt" file.

Tray:

The rack is a simple container class which is used each player's letter tiles. It draws tiles from the games bag object.

Score panel:

Score Panel is a simple class file. This is used to display each players score. Whenever user plays the score changes as well.

V. Functional Requirements

New Game:

Loads the default scrabble board with the default set of tiles, shuffles the tiles and puts seven of them in the player's hand. Displays those tiles. Sets score to zero. You should be able to start a new game at any point in time without restarting the GUI.

Shuffle:

TileBag must not be automatically reshuffled except at the beginning of a new game or after returning ties from user hand.

Human Play:

Allows the user to easily choose the tiles to play and where to place them.

Exit:

Allows the user to exit the game

Sort:

When clicked on sort tiles button the game will sort the tiles displayed for the user in alphabetical order.

Recall:

If user accidentally places a wrong tile, then the user can call back the tile using recall.

Skip:

When user feels like he can't form a word with the tiles available, he/she can skip their turn and the next player will play their turn.

Undo:

If user places a wrong tile or places it on the wrong position, the user can go back to the previously known stage by clicking on the undo button.

VI. Conclusion

The outcome of this project was a Java implementation of Scrabble that can be played by a computer controlled AI. The resulting program was both fast and chose moves strategically well. A method and justification for improving realism in this AI was outlined. This method would require further testing to

evaluate its success. Overall this project implemented some of the current state of the art in Scrabble AI, while also exploring a new element of realism within this field. The overall game functionality could be improved and expanded. Given the time constraints of this project, some features that would normally be present in Scrabble were omitted.

REFERENCES

[1]. J. Cosma and D. Jackson introducing mostly plays scrabble. Scrabble players News, (June 1983) pages 7-10, 1983.
[2]. P. Turcan. Computer Scrabble. SIGArt newsletter, 76:16-17, 1981.
[3]. P. Turcan a competitive scrabble program. SIGArt Newsletter, 80:104109, 1982
[4]. S. C. Shapiro and H. R. Smith. A scrabble crossword game-playing program. Springer, 1988.
[5]. A. W. Appel and G. I. Jacobson. The world's fastest scrabble program. Communications of the ACM, 31(5):572-578, 1988.