OPENCWX

CROSSWORD PUZZLE APPLICATION AND SOLVER

by

Sowrabbi LakshmiNarayanan

A Project Submitted to the
Graduate Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements of the Degree of
MASTER OF SCIENCE
in Information Technology and Web Science

Approved:

_____
Dr. Mukkai Krishnamoorthy
Project Advisor

Rensselaer Polytechnic Institute
Troy, New York
May 2016

TABLE OF CONTENTS

ACKNOWLEDGEMENT

I extend my heartfelt thanks to Professor Krishnamoorthy who has always been there to help me and played a major role in the completion of the project. His enthusiasm towards crosswords has kindled my interest to explore the world of crossword puzzles. His priceless advice and motivation has helped me learn to enjoy the process of software development rather just than viewing it as a duty.

I would like to also thank my parents and brother for their constant emotional support and motivation. They have always been there, believing in me and encouraging me through all my endeavors.

SYNOPSIS

Crossword puzzles are a kind of word game consisting of a grid with shaded and unshaded cells. The objective is to fill the unshaded cells with characters such that a group of words can be read in the horizontal and vertical direction from the grid. These words are associated by some way to the Across and Down clues provided with the puzzle. OpenCWX (https://github.com/LSowrabbi/CWPuzzleReader) is an Open Source Software containing several modules that helps users to interact with American-style crossword puzzles in four different file formats namely : .puz (a file format commonly used by commercial software for crossword puzzles), .ipuz (an openly-documented format for crossword and other kinds of puzzles), .tex (LaTeX) and PDF (that follows a template similar to that of WSJ Daily Crosswords). OpenCWX helps users to create their own crossword files in .puz / .ipuz format or edit existing crossword files in .puz, .ipuz, .tex or PDF format. The GUI version provided by OpenCWX is similar to that of Across Lite crossword software. The software includes a command line version that helps visually impaired users to interact with various components of crossword puzzle. The software also provides a crossword solver application that predicts the entries for Across and Down clues and solves the grid by viewing the puzzle as a Constraint Satisfaction Problem.

CHAPTER 1

INTRODUCTION

1.1 CROSSWORD PUZZLE

A Crossword Puzzle is a word puzzle that contains a square or rectangular grid with shaded (black) and unshaded (white) cells. The goal of a crossword puzzle is to fill each unshaded cell in the grid with a letter in order to form words or phrases based on the given set of Across and Down clues. The cells that are shaded are used to separate the words or phrases. The clue list generally appears outside the grid. It consists of two sublists - Across clue list, that contains clues to the words that can be read horizontally and a Down list, that contains clues to the words that can be read vertically from the grid. Each clue in the clue list is given a clue number, the first cell of each Across / Down entry in the grid, contains the corresponding clue number. Clue numbers are unique - a clue number can belong to at most one cell in the grid. Traditionally, numbered cells are labeled consecutively, usually from left to right across each row, starting with the top row and proceeding downward. Most of the professional crossword puzzles are symmetric in nature; the grids have a rotational symmetry of 180^0 (i.e., the grid looks exactly the same after a 180^0 rotation).

1.1.1 Crossword Clues

Crossword clues are mostly consistent with their solutions. Hence, the clues and their solutions will generally agree in grammatical tense, number (singular or plural), and degree (comparative adjectives).

The frequently used clue types are:

- **Straight clues,** containing simple definitions of the answers.
- **Anagrams** that rearrange the letters in the clue to give the answer.
- **Fill-in-the-blank** clues, where the blank space in the clue refers to the solution.
- **"Before and after" clues** containing a single word that is part of two phrases, often designated with parentheses and brackets.
- A **question mark** at the end of clue that hints that the clue - answer combination involves some sort of wordplay.
- **Abbreviations** - abbreviations in the clue indicating that the answer is to be similarly abbreviated.

- **Indirect clues** - clues involving wordplay which are to be taken metaphorically or in some sense other than their literal meaning, requiring some form of lateral thinking.
- **Cross-reference** clues where the answer to one clue forms part of another clue, in which it is referred to by number and direction.

1.1.2 Crossword Themes

Themed crossword puzzles consists of a number of long entries that share some relationship, type of pun or other element in common. The common types of themes are:

- **Category themes**, where the theme elements are all members of the same category.
- **Quote themes**, featuring a famous quote broken up into parts to fit in the grid.
- **Commemorative themes**, based on a particular event or person (often published on an appropriate anniversary).
- **Rebus themes**, where multiple letters or symbols occupy a single square in the puzzle.
- **Addition themes**, where theme entries are created by adding a letters or words to an existing word or phrase.
- **Subtraction themes**, where letters are removed to make a new word or phrase.
- **Compound themes**, where the start or end of the theme entries can all precede or follow another word, which is given elsewhere in the puzzle.
- **Synonym themes**, where the theme entries contain synonyms.

1.1.3 Diagramless Crossword

In a diagramless crossword, the grid offers overall dimensions, but the locations of most of the clue numbers and shaded squares are unspecified.

1.2 THE NEW YORK TIMES CROSSWORD PUZZLE

The New York Times crossword puzzle is a daily puzzle published in The New York Times. It is one of the most widely-distributed American crosswords. Weekday puzzles have a standard size of 15×15 grid, while weekend puzzles may be 21×21, 23×23, or 25×25. The New York Times puzzles set a common pattern for American crosswords by increasing in difficulty throughout the week: the Monday puzzles are the easiest and the puzzles get harder until Saturday.

1.3 ACROSS LITE

Across Lite is a crossword application published by Literate Software Systems that supports electronic versions of puzzles from The New York Times and other leading newspapers. There are two Across crossword formats - a text version and a binary (.puz) version. The binary version is the one used for online distribution. It can be read by Across Lite on any platform and contains checksums to ensure the integrity of the puzzle during copying and downloading operations. Solution scrambling is also possible in this format. Across Lite saves worked-on puzzles in this format. The text version is used for creating puzzles in binary (.puz) format. The application layout has three sections - left, right and top section. The puzzle is placed on the left and clues are placed on the right section. Buttons are placed across the top to give users access to the program's major features. The application has a pencil tool that allows users to enter answers that they're not quite sure of. It also has check and reveal options to check/reveal individual letters or words.

1.4 CONTRIBUTIONS

Although Across Lite provides an excellent interface that helps normal people to interact with The New York Times crossword puzzles, the GUI makes it a challenging task for visually impaired users to interact with the puzzle. The command line version of OpenCWX helps to overcome this problem by enabling the users to interact with various components of the puzzle. Users should first create a text file version of the puzzle (based on the template given by Across Lite) in order to create a puzzle in .puz format. This turns out to be an arduous tasks for users unfamiliar with the template. By providing a GUI, ‘Create CW Puzzle’ module of OpenCWX helps to create crossword puzzles in a straightforward manner (in both .puz and .ipuz formats). Crossword puzzle application developers can benefit from this software, by using it as a tool to convert puzzles from .puz format to .ipuz, .tex or PDF format. As OpenCWX is an Open Source Software, such developers can also be benefitted by modifying or extending this software such that it matches their goals.

1.5 OUTLINE OF THIS DOCUMENT

Chapter 2 discusses about the limitations of the existing system, and the features of the proposed system that helps to overcome these limitations. Chapter 3 provides an overview of the various modules present in OpenCWX. Chapter 4 discusses about the crossword puzzle formats currently supported by this software and how it deals with these formats. Chapter 5 discusses about the implementation of the modules in OpenCWX and results obtained after testing Crossword Puzzle Solver on mini crossword puzzles published by The New York Times.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Across Lite application does not work with any puzzle formats other than its proprietary binary (.puz) format. For example, it does not work with IPUZ, which is an open documented crossword puzzle format. Across Lite provides an option to save the puzzle in pdf format (which can be achieved through the print option), however, the application does not work the other way round - there are currently no features that help in converting a puzzles in PDF format to binary (.puz) format. Across Lite accepts text files for creating crossword puzzles which is then converted into a binary format file by the application. Users should go through a set procedure for creating the text format file. The text version usually consists of several tags and templates; if there are any errors in its description, Across Lite will refuse to load the puzzle and give diagnostic error messages. Hence in order to create a valid binary format file, users should go through all the required tags and templates and enter the crossword information accordingly, which is often considered as a strenuous task. Although Across Lite's GUI contains several features that helps to a great extent for the sighted people to interact with the puzzle, it makes the interaction with the puzzle an arduous task for visually impaired users. Such users generally find it difficult to differentiate between the shaded and unshaded cells in the crossword puzzle.

2.2 PROPOSED SYSTEM

The proposed system is an open source version of Across Lite that helps users to solve professional crossword puzzles in binary (.puz) format.

2.2.1 Features

In addition to the existing features of Across Lite, the proposed system supports several other features that include:

- **Command-line Interface** that helps visually impaired users to interact with the puzzle.

- ➢ Feature to help users store current state of the puzzle in **text format**. As the text format replaces all shaded cells in the puzzle with a ‘ **.** ’ (dot - for normal puzzles) or a ‘ : ’ (colon - for diagramless puzzles); this might be helpful in cases where users would like to print their puzzle without printer ink wastage.
- ➢ GUI for interacting with puzzles in **IPUZ** format.
- ➢ GUI for **creating crossword puzzles** - that helps users to create a new crossword puzzle (in PUZ / IPUZ format) or edit the clue list / solution set of an already existing puzzle.
- ➢ Feature to **save partially created puzzles as text files** that can be opened and worked on anytime later.
- ➢ GUI for editing the clue list / solution set of puzzles in **LaTeX format** and converting the edited puzzle into binary (.puz) format file.

2.2.2 WSJ Puzzles

The proposed system has a module that converts Wall Street Journal (WSJ) Puzzles that are published in PDF format (or any crossword files in PDF format that has a layout similar to WSJ puzzles) into binary (.puz) files. After reading the description of the puzzle from the PDF file, there are two steps that needs to be followed before converting it into a PUZ file. They are:

I. Ask the user to verify whether the grid has all its shaded and unshaded cells in place. If any shaded cell is misplaced, allow the user to edit the grid in order to form the proper grid.

II. Ask the user to verify whether all the clue numbers in the clue list match their corresponding Across/Down clues. If not, allow the user to edit the clues.

As the PDF document is interpreted based on its layout, step I) is substantial in this module, since the interpreted data might not always be similar to the original grid. So users have to verify and edit the grid if required to match the original grid. The interpreter performs a fairly good job in detecting all clues corresponding to their clue numbers. However in some exceptional cases, the interpreter might not be able to output the correct data; in such cases step II) plays a significant role which allows the users to manually verify/edit the clue list.

2.2.3 Crossword Puzzle Solver

This module reads a crossword puzzle in binary (.puz) format and solves the puzzle by assigning constraints to the unshaded cells in the puzzle's grid. So the solution set of each clue is narrowed based on the constraints imposed on the corresponding cells. The final solution grid displayed selects the solution with the highest weight from all the solutions available in the solution set for each clue. This can be useful while accessing a locked or scrambled puzzle. If the users do not know the key to unlock the puzzle, 'check' and 'reveal' options would be disabled by Across Lite; hence this module suggests the solution corresponding to the clues in the puzzle based on the constraints in the grid.

CHAPTER 3

SYSTEM ARCHITECTURE

3.1 SYSTEM DESCRIPTION

OpenCW's architecture consists of four modules as depicted in Fig. 3.1

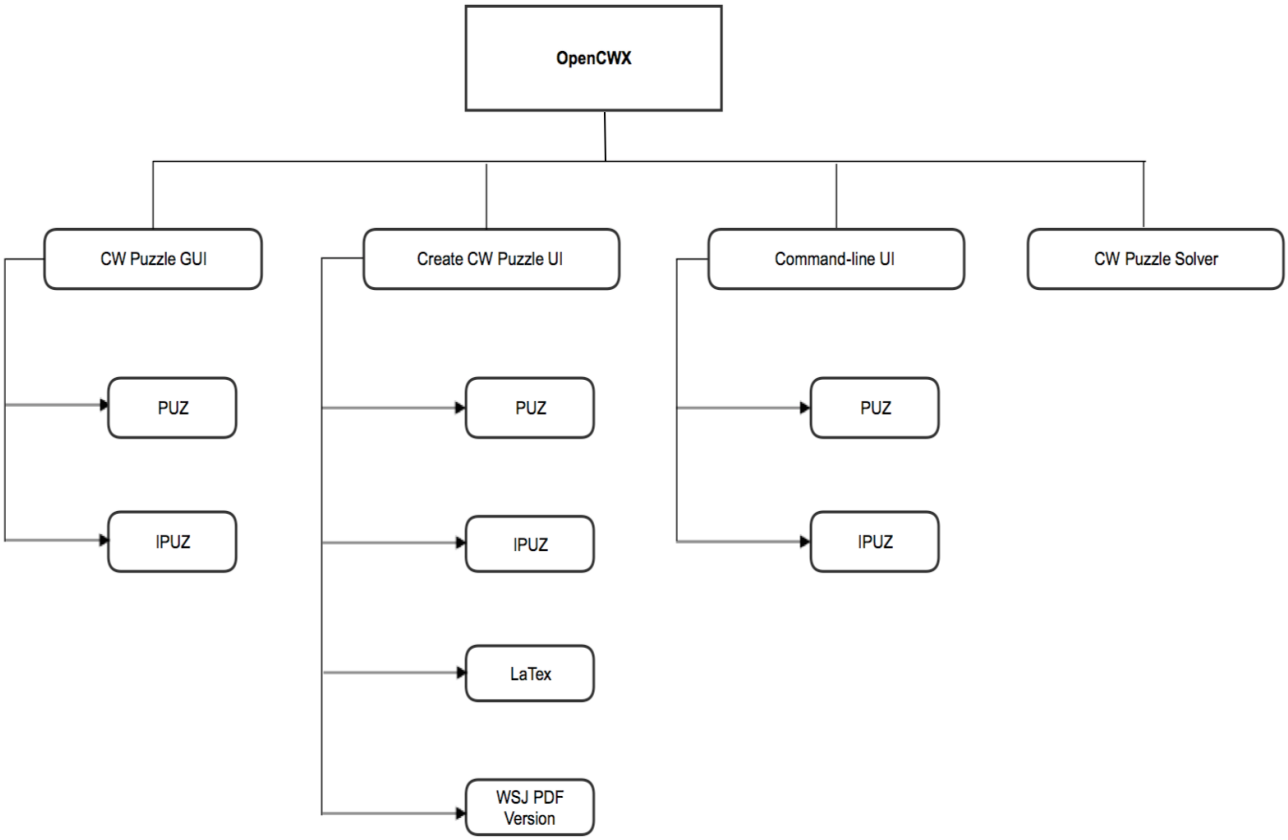


Fig 3.1. OpenCWX Modules

3.1.1 Crossword Puzzle GUI

This application reads data from the puzzle files in PUZ / IPUZ format files, stores the read data in different data structures and creates a Graphical User Interface similar to that of Across Lite Application, in order to allow the users to interact with the puzzle. When a diagramless puzzle is read, the puzzle grid displayed is similar to that of a normal grid, i.e., all the blocks are represented as shaded cells by the GUI. The current state of the puzzle can be saved in four different formats namely binary (.puz), IPUZ (.ipuz), LaTeX (.tex) and text (.txt) format.

3.1.2 Create Crossword Puzzle UI

This application helps the user to create a crossword puzzle file in PUZ / IPUZ format. The puzzle can be created by allowing the users to manually design the grid, enter clues and solution entries for the puzzle or by allowing the users to edit an already existing puzzle whose description can be present in any one of the following file formats:

- Binary (.puz) format
- IPUZ (.ipuz) format
- PDF format
- LaTeX format (.tex)

This application has a module that saves a partially constructed puzzle in text (.txt) file format. The text file can be read by this application anytime later and it can used to construct a puzzle in any one of the four formats mentioned above.

3.1.3 Command-line UI

A command-line User Interface that allows users to interact with the puzzle. This application is compatible with puzzles in binary (.puz) and IPUZ format.

3.1.4 Crossword Puzzle Solver

Crossword Puzzle Solver takes a puzzle in binary (.puz) format as an input and solves the puzzle using Constraint Satisfaction Problem. http://www.otsys.com/clue/ served as a source for the clue database. Clues were appended to different lists based on the length of the corresponding solution. Repeated clues for the same solution entry were removed in order to make the search process more effective. The resulting clue list can be found in the ‘clues.p’ pickle file that can be found at:
(https://drive.google.com/file/d/0Bz9E35R1QWPoVTZfcGRDNndDNjA/view?usp=sharing)

Here each word corresponding to a clue is assigned a set of constraints based upon the other clues with which it shares the cells. For example, let the solution to Across 1 be a 5 letter word, if the cells corresponding to Across 1 intersect (in increasing order) with the 1st cell corresponding to that of Down 1, Down 2, Down 3, Down 4 and Down 5; the constraint would be:

1st letter of **Across 1** should be the same as **1st letter** of **Down 1**
2st letter of **Across 1** should be the same as **1st letter** of **Down 2**
3st letter of **Across 1** should be the same as **1st letter** of **Down 3**
4st letter of **Across 1** should be the same as **1st letter** of **Down 4**
5st letter of **Across 1** should be the same as **1st letter** of **Down 5**

The Solver initially searches the clues database to check if clues in the clue list match exactly with the clues in the database. If such a match occurs, the solution corresponding to the clue is appended to the temporary solution list. The temporary solution list may contain more than one entry for a single clue. Hence, the solution list is filtered by propagation, where clues with a single solution is prioritized, and the entries that do not satisfy the constraints are removed from the solution list. If multiple entries are present in the solution list of a clue, a cost is assigned to each entry based on the number of constraints it satisfies and the entry with highest cost is added to the solution list. Each cell in the grid is assigned a value if an entry corresponding to the cell is present in the solution list. The value is assigned based on the final Across and Down solution list containing entries with maximum cost. Now the clues database is searched again for clues with unassigned solution entries, based on the new constraints that are obtained by assigning values to cells in the grid. However this time, the search does not demand for an exact match, but considers the clues in the database that are almost similar to the corresponding clues in the database and such that it satisfies the constraints. The solution list obtained is propagated, and cost is assigned to each entry and the final solution grid is determined based on the result.

3.2 FILE CLASS

File
Attributes: + title : String + author : String + cpyrt : String + notes : String + width : Integer + height : Integer + solnblock : Grid + cellblock : Grid + acc : Integer + dwn : Integer + across : Clues [1..*] + down : Clues [1..*] + loc : Path

The File class stores crossword puzzle information. OpenCWX uses the File class to transfer information about the puzzle between programs dealing with different file formats.

CHAPTER 4

CROSSWORD PUZZLE FILE FORMATS

4.1 DEALING WITH PUZZLES IN PUZ FORMAT

PUZ file separates and stores the puzzle description in four sections:

4.1.1 Header Section

Header section contains the following components:

1. **Width** - The width of the grid.
2. **Height** - The height of the grid.
3. **No. of clues** - Total number of clues present in the puzzle's clue list.
4. **Diagramless tag** - To identify whether the puzzle is scrambled or not. The tag value is set to 1025 for diagramless puzzles and is set to 1 for normal puzzles.
5. **Scrambled tag** - To identify whether the puzzle is scrambled or not. The tag value is set to 4 for scrambled puzzles and is set to 0 for normal puzzles.

This section also contains details of five checksums that are used by PUZ file to detect whether the file has been corrupted. The five checksums are:

1. **CIB Checksum** - Calculates checksum over the width, height, no. of clues, diagramless tag and scrambled tag.
2. **Overall File Checksum** - Calculates checksum over the CIB checksum, current state grid, solution state grid and checksum calculated for string section.
3. **Masked Low Checksum**
4. **Masked High Checksum**
   Masked Low and Masked High Checksums XOR masks CIB checksum, current state grid, solution state grid against the magic string 'ICHEATED'
5. **Scrambled Checksum** - Holds the checksum of the actual solution grid in scrambled puzzles that can be used to verify whether the user has entered a valid key.

4.1.2 Grid Section

Grid section contains the puzzle's solution grid and current state of the grid. Here shaded cells are represented by '.' (':' for diagramless puzzles). Unfilled cells in the current state grid are represented by '-'. For rebus puzzles and rebus entries entered by the user, the solution grid and current state grid contain only the first character of such rebus entries.

4.1.3 String Section

String section contains the following information about the puzzle:

1. Title
2. Author
3. Copyright
4. Clue list - clues are ordered by the ascending order of clue numbers in the clue list. If a clue number corresponds to both an across and down clue, then the across clue is placed first in the list and the down clue goes after it.
5. Notes (optional)

4.1.4 Extra Sections

The optional extra sections available are

1. **GRBS** - contains grid with numbers, where a non-zero number indicates that the cell has a rebus solution
2. **RTBL** - containing a series of rebus strings and numbers. These numbers correspond to their position in the grid (if GRBS has a cell with number n;, then string with number n-1 will be its corresponding rebus value of the cell)
3. **RUSR** - Grid with user entered rebus values (each value terminates with a '\0')
4. **GEXT** - contains the following puzzle information
    a. Cells in the grid were pencil option was used
    b. Position in the grid where circles are placed (if any)
    c. Cells for which the solution has been revealed
    d. Cells that contain incorrect and previously incorrect entries
5. **LTIM** - denotes whether the timer is running or has stopped and the time taken by the user to work on the puzzle

OpenCWX makes use of the dimensions mentioned in the header section to construct the solution grid and current state grid. It unlocks the scrambled puzzles if a valid key has been entered by the user. If a puzzle has been unlocked, it stores the puzzle as a normal puzzle when the user saves the puzzle. OpenCWX does not give any of the print / convert options that are made available in Across Lite for diagramless puzzles. Instead, the puzzle is displayed in a similar way to a normal puzzle by displaying the blocks as shaded cells; but while saving, it is stored in the binary file as a diagramless puzzle again. All crossword puzzles created using createCW application are saved as normal puzzles (diagramless or scrambled puzzles cannot be created using this application). OpenCWX throws an error notifying the user that the file has been corrupted if any one of the calculated checksums does not match with their respective checksums (mentioned in Header Section) that are present in the binary file.

4.2 DEALING WITH PUZZLES IN IPUZ FORMAT

A crossword puzzle in IPUZ format is represented as a JSON object. Though various kinds of crosswords like cryptic crosswords, arrowword and logic crosswords can be written in this format, OpenCWX is designed to work only with the pattern followed by American crossword. Let us assume that the puzzle description read from any IPUZ file is stored in the 'puzzle' instance.

So our application will require only the following fields:

1) **Width of the grid** - present in the dimensions field of the dictionary, puzzle['dimensions'][‘width’]
2) **Height of the grid** - present in the dimensions field of the dictionary, puzzle['dimensions'][‘height’]
3) **Across and Down clue list** (clue no., clue) that can be accessed from the dictionary puzzle['clues'][‘Across’] and puzzle['clues'][Down’]. If the individual clues are by themselves instances of dictionaries; then the clues can be accessed from this dictionary using the field ‘clue’ and ‘number’ (for example : puzzle['clues']['Across'][n][‘clue’] can be used to access the nth clue and puzzle['clues']['Across'][n][‘number’] can be used to access the clue number of nth clue, if it is present in a dictionary format). Another important constraint is that clue numbers corresponding to all clues must take integer values. OpenCWX can read only integer valued clue numbers in order to support interoperability between various puzzle file formats).
4) **Grid Layout -** Given by the ‘puzzle’ field of the dictionary (puzzle[‘puzzle’]). This gives information about the position of all cell numbers and shaded cells in the grid.

The optional fields that can be read and used by our application are:

1) **Title, Author, Copyright** and **Notes** can be accessed from the puzzle instance (if available) using the fields ‘title’, ‘author’, ‘copyright’ and ‘notes’ respectively.
2) **Block tag** represented by the field ‘block’ is used to denote the shaded cells in the grid, by default the shaded cells are represented by ‘#’.
3) **Empty tag** represented by the field ‘empty’ is used to denote cells without any cell number (in ‘puzzle’ field’) in the grid, by default the empty cells are represented by 0.
4) **Current State of the Grid** - Each entry in the current state of the grid can be accessed from the dictionary puzzle[‘saved’].
5) **Solution Grid** - Each entry in the solution grid can be accessed from the dictionary puzzle[‘solution’].

If the entry is an instance of dictionary by itself, then the value corresponding to each entry can be obtained with the help of ‘value’ field (for example the current state / solution at row r and column j can be accessed by puzzle[‘saved’][r][c][‘value’] / puzzle[‘solution’][r][c][‘value’] if the corresponding entry is in dictionary format).

4.3 DEALING WITH PUZZLES IN LATEX (.tex) FORMAT

Crossword puzzles are typeset in LaTeX using the package provided by cwpuzzle.dtx. Though several types of crossword puzzles like classical, number, fill-in and line-delimited crossword puzzles can be produced with the help of this package, OpenCWX supports only the classical crossword puzzle in order to maintain its interoperability. So a sample input file must contain details about the puzzle dimensions, solution grid and the clue list.

OpenCWX can read and write crossword puzzles that are typeset in the format shown in Fig.4.1.

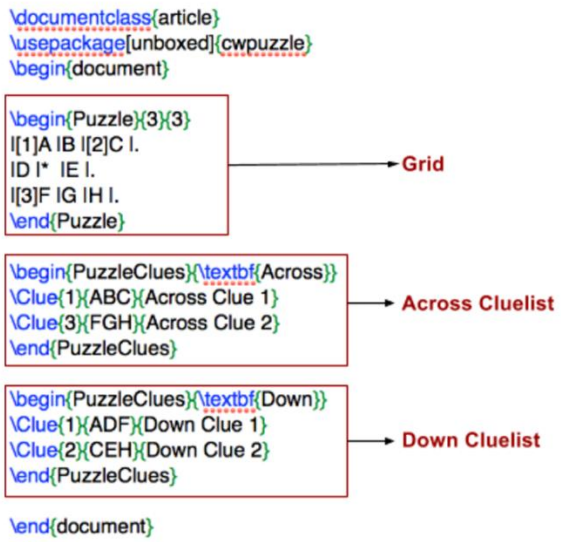


Fig. 4.1. Crossword Puzzle in LaTeX format

4.3.1 Grid :

The crossword puzzle grid is typeset by the ‘Puzzle’ environment which takes two arguments that denote the width and height of the grid. Description of each new cell in the grid starts with the bar macro ‘|’. This macro takes three arguments. The first (optional) argument denotes the cell number and is enclosed within brackets. The second optional argument specifies the format of the cell (like its shape, frame, thickness, shaded cell). This argument is used by our application to detect shaded cells, the other cell format information provided by this argument is generally ignored. The third argument should either be a single character or an unassigned entry (denoted by backslash followed by a space ‘\ ‘). OpenCWX does not support grids that have their third argument as ‘{}’ (normally denotes empty cells that are typeset as a white box in LaTeX).

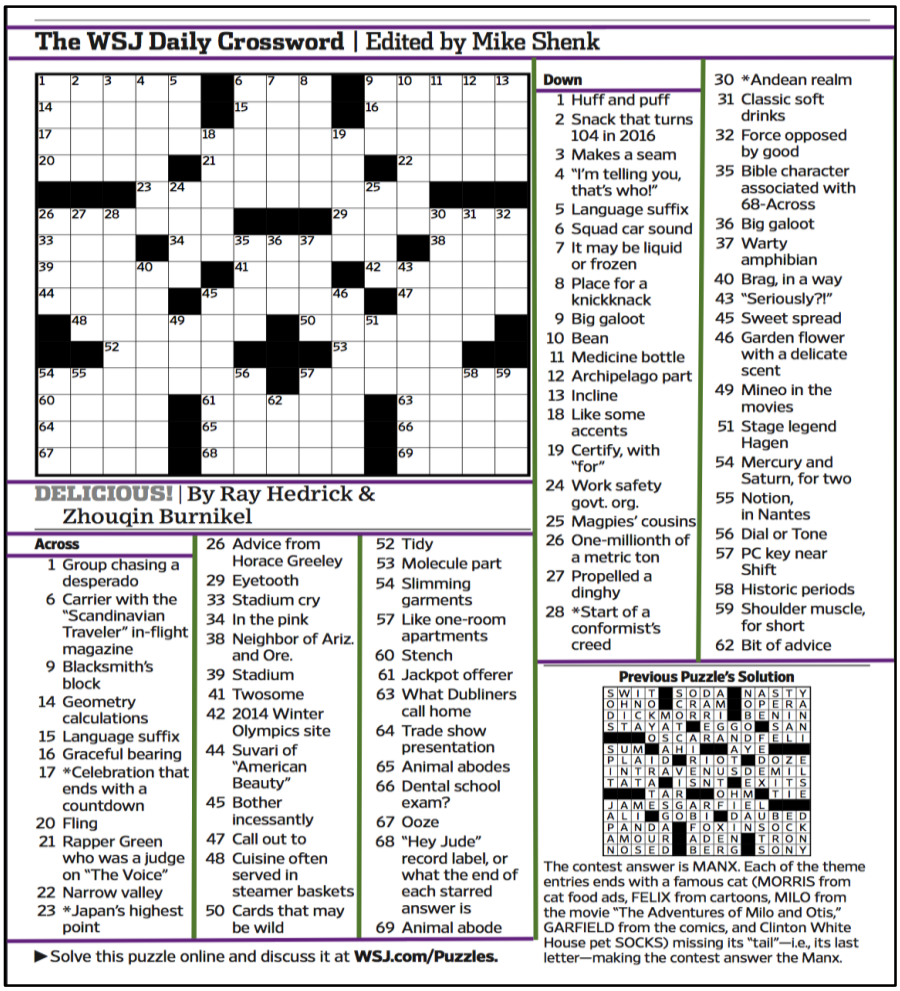
4.3.2 Across and Down Clue List

The clues for the puzzle is typeset using the environment ‘PuzzleClues’. OpenCWX recognizes the Across clue list from Down clue list by ‘textbf’ macro that is taken as an argument by this environment. If this ‘textbf’ macro takes ‘Across’ as its argument, then our application recognizes that it has encountered the across clue list. Similarly a Down clue list is recognized by the ‘Down’ argument taken by this macro. The ‘Clue’ macro defined by this macro, denotes that a new clue has been encountered. This macro takes three arguments. The first argument is the cell number

used to denote the clue. The second argument is the solution (or current entry) for the clue. The third argument is the clue.

4.4 DEALING WITH PUZZLES IN PDF FORMAT

OpenCWX's wsj_pdf_converter extracts puzzle data along with its original layout from the PDF file containing the puzzle information using the utility 'pdftotext -layout'. The different components of the puzzle is identified from the extracted data with the help of layout information. This software supports puzzles that have a layout similar to that of Wall Street Journal's 15x15 crossword puzzles. The layout segmentation is shown in Fig.4.2.



The WSJ Daily Crossword | Edited by Mike Shenk

DELICIOUS! | By Ray Hedrick & Zhouqin Burnikel

Across

1 Group chasing a desperado
6 Carrier with the "Scandinavian Traveler" in-flight magazine
9 Blacksmith's block
14 Geometry calculations
15 Language suffix
16 Graceful bearing
17 *Celebration that ends with a countdown
20 Fling
21 Rapper Green who was a judge on "The Voice"
22 Narrow valley
23 *Japan's highest point
26 Advice from Horace Greeley
29 Eyetooth
33 Stadium cry
34 In the pink
38 Neighbor of Ariz. and Ore.
39 Stadium
41 Twosome
42 2014 Winter Olympics site
44 Suvari of "American Beauty"
45 Bother incessantly
47 Call out to
48 Cuisine often served in steamer baskets
50 Cards that may be wild
52 Tidy
53 Molecule part
54 Slimming garments
57 Like one-room apartments
60 Stench
61 Jackpot offerer
63 What Dubliners call home
64 Trade show presentation
65 Animal abodes
66 Dental school exam?
67 Ooze
68 "Hey Jude" record label, or what the end of each starred answer is
69 Animal abode

Down

1 Huff and puff
2 Snack that turns 104 in 2016
3 Makes a seam
4 "I'm telling you, that's who!"
5 Language suffix
6 Squad car sound
7 It may be liquid or frozen
8 Place for a knickknack
9 Big galoot
10 Bean
11 Medicine bottle
12 Archipelago part
13 Incline
18 Like some accents
19 Certify, with "for"
24 Work safety govt. org.
25 Magpies' cousins
26 One-millionth of a metric ton
27 Propelled a dinghy
28 *Start of a conformist's creed
30 *Andean realm
31 Classic soft drinks
32 Force opposed by good
35 Bible character associated with 68-Across
36 Big galoot
37 Warty amphibian
40 Brag, in a way
43 "Seriously?!"
45 Sweet spread
46 Garden flower with a delicate scent
49 Mineo in the movies
51 Stage legend Hagen
54 Mercury and Saturn, for two
55 Notion, in Nantes
56 Dial or Tone
57 PC key near Shift
58 Historic periods
59 Shoulder muscle, for short
62 Bit of advice

Previous Puzzle's Solution

The contest answer is MANX. Each of the theme entries ends with a famous cat (MORRIS from cat food ads, FELIX from cartoons, MILO from the movie "The Adventures of Milo and Otis," GARFIELD from the comics, and Clinton White House pet SOCKS) missing its "tail"—i.e., its last letter—making the contest answer the Manx.

►Solve this puzzle online and discuss it at **WSJ.com/Puzzles.**

Fig. 4.2. Segmentation of WSJ Crossword Puzzle in PDF format

The data segments obtained are:

4.4.1 Grid

The size of each cell in the layout is determined by finding the rightmost position of each row of the grid. Once this is found a simple division by the number of cells in the row (15 here) gives the size of each cell in that row. The size of each cell in a given row, helps in determining to which cell a given cell number corresponds to in the row. Sometimes a single cell number that occurs at the beginning or end of the row might be accidentally printed in a new line. Such errors are corrected by checking whether the cell number is placed towards the right end or left end of the grid. If it is placed in the right end of the grid, and its value is lesser than the leftmost cell number in the row after it, then it would be placed at the end of the previous row, else it would be placed in the end of the next row. A similar technique is followed when the cell is placed at the left end of the grid; here it is inserted at the beginning of the previous/next row's cell numbers.

4.4.2 Title and Author

Data segment that represents the title and author can be found just before the row containing across clue list. This clue list starts with the keyword 'Across'. The title string is segregated from the author string by identifying the position of '|' key that is present between the two strings and separates them.

4.4.3 First, Second and Third Column

First, second and third columns are present below the title and author string. The first column starts with the keyword 'Across' that denotes the Across clue list. The width of each column is considered to be one third of the end of grid value. The third clue may sometimes contain the 'Down' clue list which can be identified by the 'Down' keyword. Once a number is identified, its position is checked; if it is located to the left end of any column and if its value is greater than the clue number of the previous clue and smaller than the maximum clue number (that can be found by scanning the grid for the maximum cell number), then the number corresponds to a clue number. This indicates that a new cell has been encountered. The clues are stored in the respective Across/Down clue list. The end of these columns is identified by a bulletin (that is present before the string 'Solve' in the puzzle) which is represented in the text file by the character 's'.

Fourth and Fifth Column

Fourth and fifth columns are present to the right of the grid. The width of each column is considered to be half of the space between the end of the file and end of the grid. The down clue list (identified by the ‘Down’ keyword) may also start at the fourth column. So the individual clues are identified along with their respective clue numbers and put into the appropriate clue lists by the method similar to the one followed by first, second and third columns. The end of the fourth and fifth column is identified by a keyword ‘Previous’ (first word of the string “Previous Puzzle’s Solution” that is present below the clue list).

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

5.1.1 Crossword Puzzle GUI

OpenCWX provides a GUI that helps users to interact with crossword puzzles in .puz/.ipuz format. The grid in the diagramless puzzles is displayed as a pattern comprising of shaded and unshaded cells, similar to that of a normal puzzle. The software also supports puzzles with shapes.

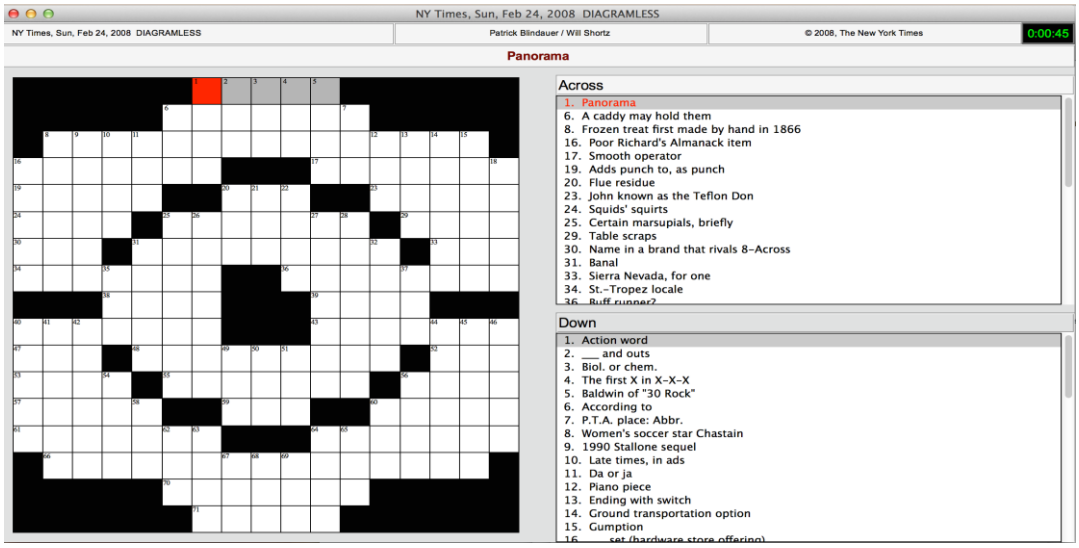


Fig 5.1. GUI for Diagramless Crossword Puzzle

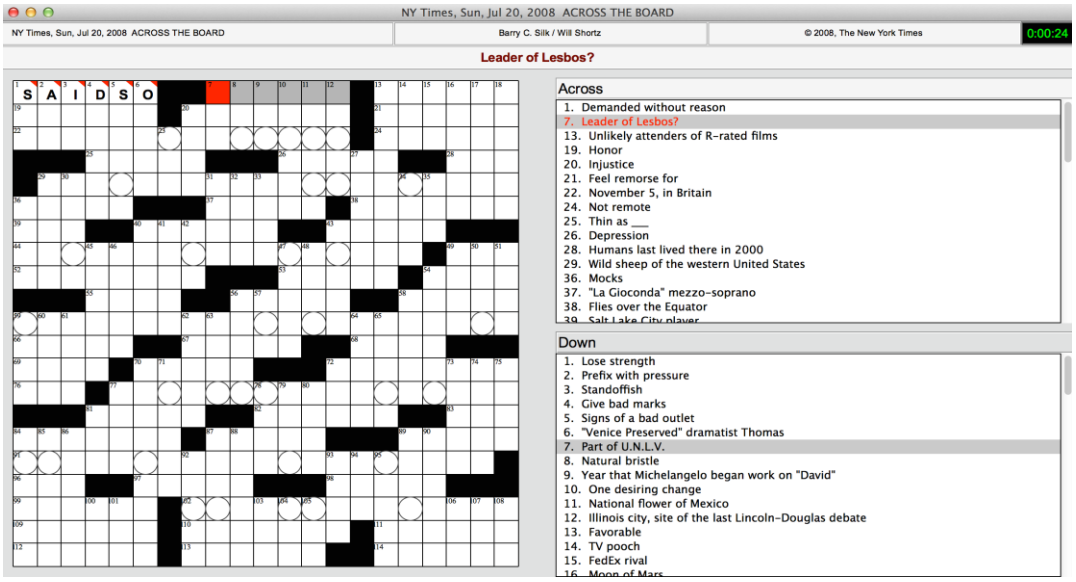


Fig 5.2. GUI for Crossword Puzzle with Shapes

This version of the software contains most of the features that are present in the Across Lite application.

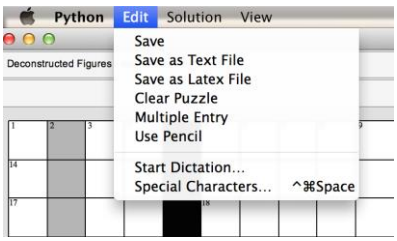


Fig 5.3. a) Edit Menu

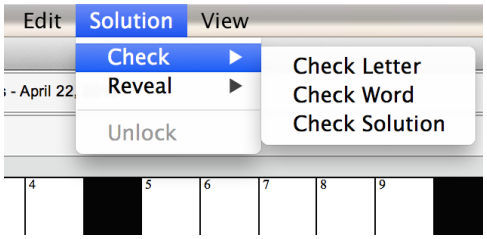


Fig 5.3. b) Solution Menu

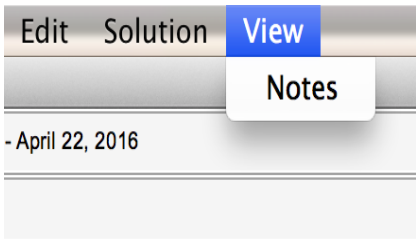


Fig 5.3. c) View Menu

The following features are available:

- **Pencil option** when the user is unsure of an entry

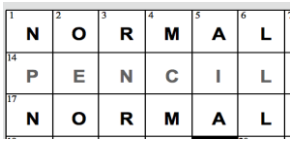


Fig 5.4. Pencil Option

- **Check option** to check the correctness for a letter, word or all letters. Incorrect letters are marked with a cross (X). When an entry is made in a crossed cell; a black tag is inserted at the top right corner of the cell to represent that the user had used the check option earlier and there was a previously incorrect entry in the cell.

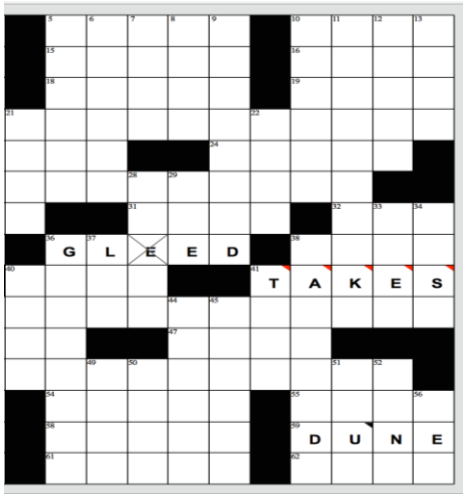


Fig 5.5. Check Option

- **Reveal option** to reveal a letter, word or entire solution - a red tag is inserted at the top right corner of the cell to represent a revealed cell. Reveal Incorrect Letters option can be used to reveal incorrect entries in the current grid.

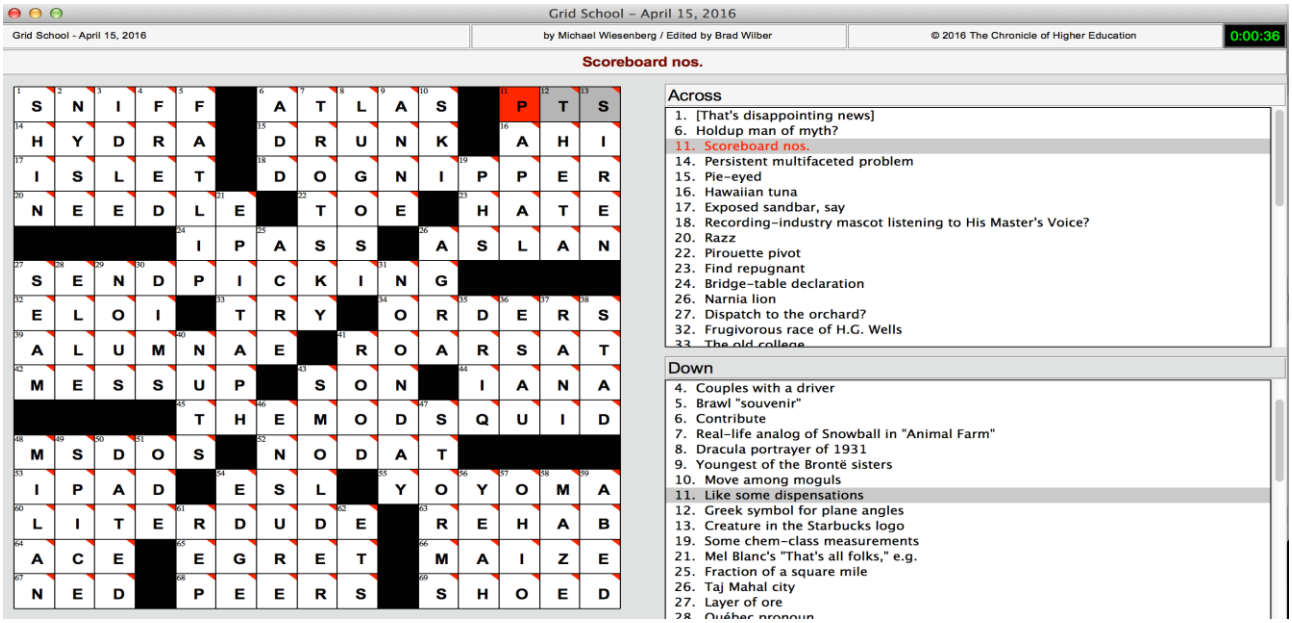


Fig 5.6. Reveal Option

- **Unlock Option** allows the user to enter the 4 digit key and unlock the solution for Scrambled puzzles.

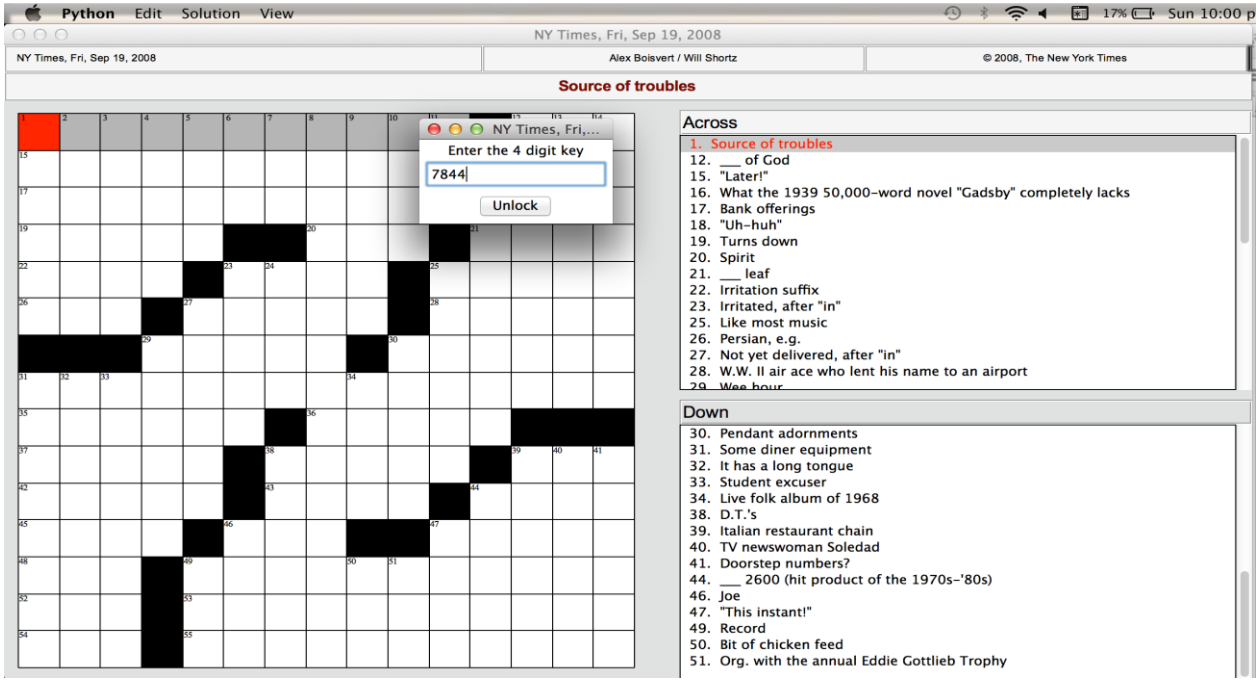


Fig 5.7. a) Unlock Solution Option

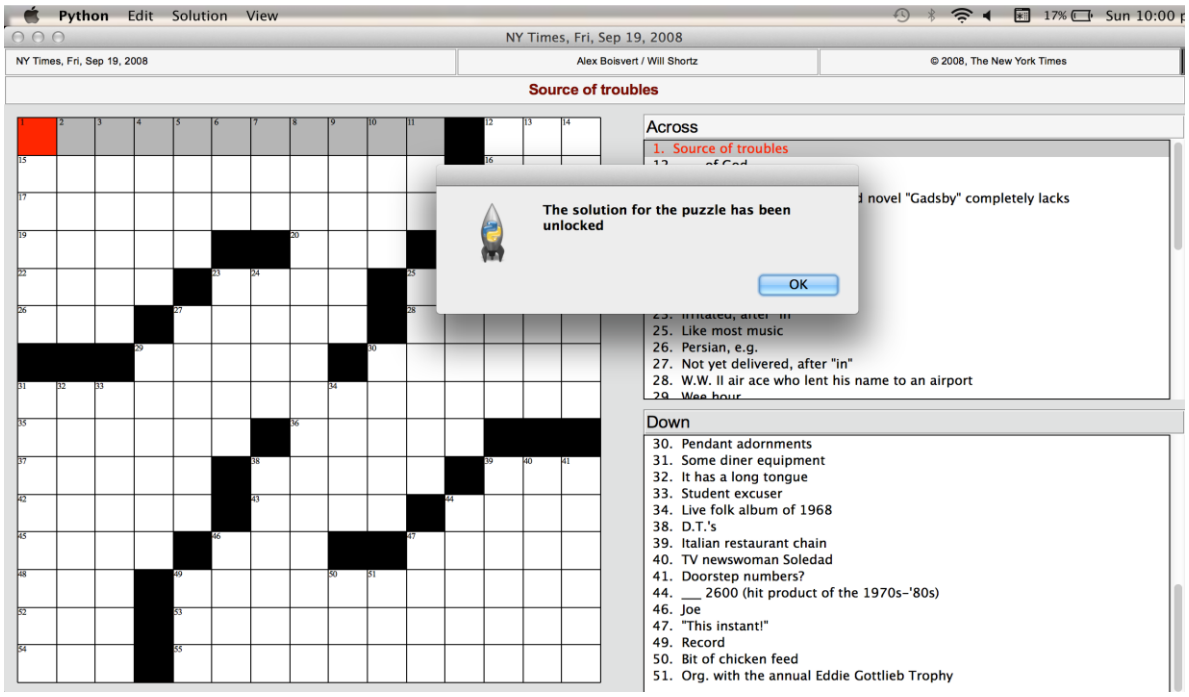


Fig 5.7. b) Valid key unlocks Solution

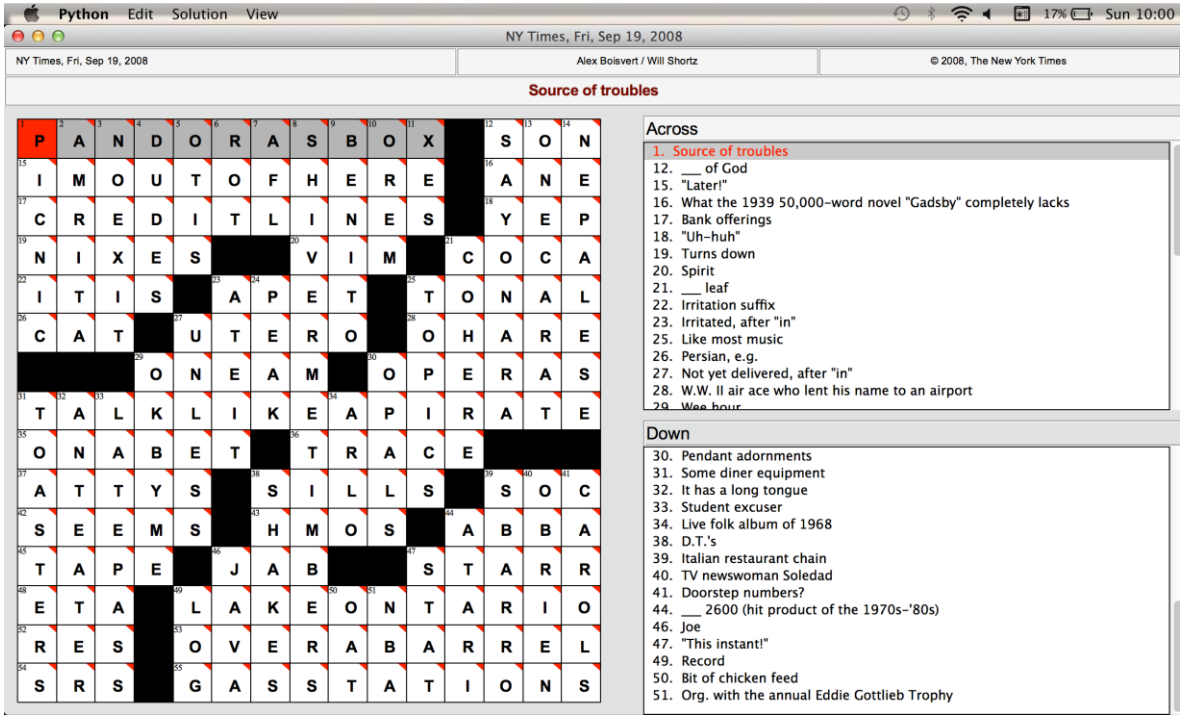


Fig 5.7. c) Users can access solution

- **Multiple Entry option** - if the user has to enter more than one letter in a cell which will be useful for rebus puzzles.

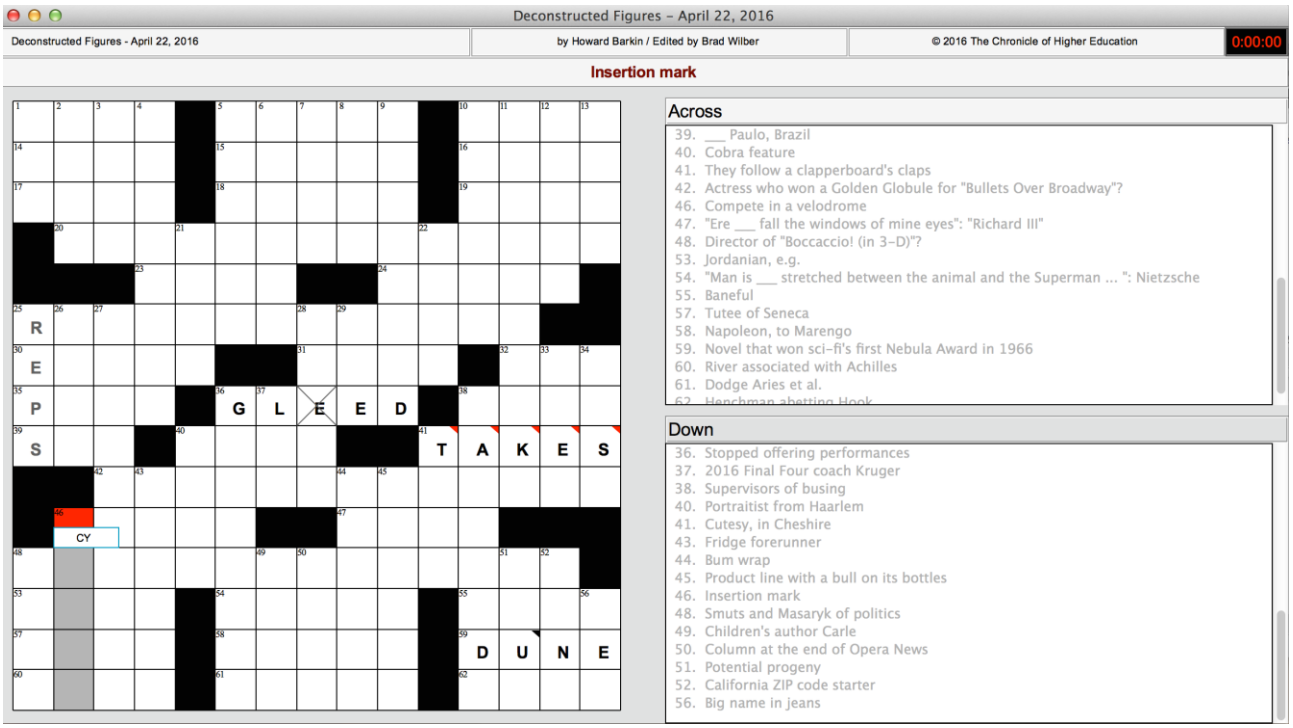


Fig 5.8. Multiple Entry Option

- An option to allow the users to view any notes that had been packaged along with the puzzle.

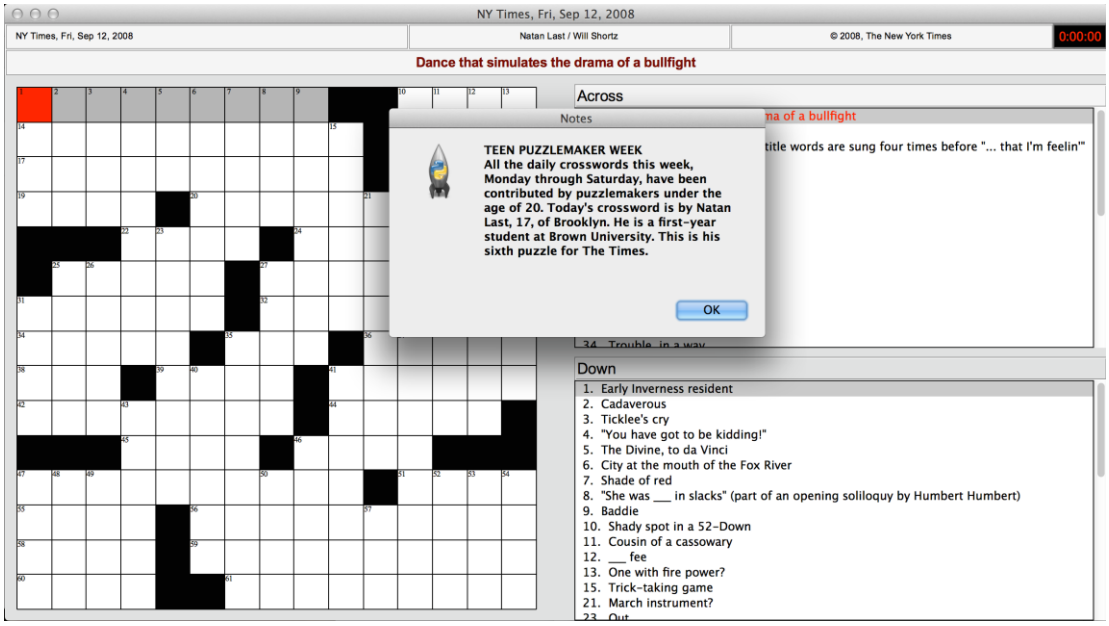


Fig 5.9. Notes Option

- A timer at the top right corner of the window to track the time spent by the user on the current puzzle. Users can pause the timer by clicking the timer once; the foreground colour changes from green to red when it is paused.
- **Save options** - Puzzles can be saved in the .puz/.ipuz format. The puzzle information like the current state of the grid, Across and Down clue list along with the corresponding words in the grid can be saved in a text file for future reference. The puzzle can also be saved as a LaTeX file.

```
Current State of the puzzle:

 S N I F F . A T L A S . P - -
 H - - - - . D - - - - . A - -
 I - - - - . D - - - - - P - -
 N - - - - E . - - - - . - A - -
 . . . . I P A S S . - - L - -
 S - - - - I - - - - - . . . .
 E - - - - . T - - . - - - - - -
 A - - - - A - - . - - - - - - -
 M - - - - P . - - - - . - - - -
 . . . . - H - - - - - - - - - -
 - - - - - . - - - - - . . . .
 - - - - - . - - - . - - - O - -
 - - - - - - - - - - . - - H - -
 - - - . E G R E T . - - I - -
 - - - . - - - - - - . - - O - -

CLUES

Across :

 1. [That's disappointing news] <SNIFF>
 6. Holdup man of myth? <ATLAS>
 11. Scoreboard nos. <P-->
 14. Persistent multifaceted problem <H---->
 15. Pie-eyed <D---->
 16. Hawaiian tuna <A-->
 17. Exposed sandbar, say <I---->
 18. Recording-industry mascot listening to His Master's Voice? <D-----P-->
 20. Razz <N----E>
 22. Pirouette pivot <--->
 23. Find repugnant <-A-->
 24. Bridge-table declaration <IPASS>
 26. Narnia lion <--L-->
 27. Dispatch to the orchard? <S----I----->
 32. Frugivorous race of H.G. Wells <E--->
 33. The old college ___ <T-->
 34. Directives <------>
 39. Bryn Mawr ___ Bulletin <A----A->
 41. Greets with gales of laughter <------->
 42. Botch <M----P>
 43. Jason Schwartzman, to Talia Shire <--->
 44. Suffix with Dickens <---->
 45. Cephalopod-about-town in Swinging London? <-H--------->
 48. Windows forerunner <----->
 52. Acknowledge wordlessly <----->
 53. Jobs announcement of 2010 <---->
 54. Course that may aid in accent reduction, for short <--->
 55. Cellist on the "Crouching Tiger, Hidden Dragon" soundtrack <---O-->
 60. "Is that a quart, bro?" rejoinder? <--------->
 63. Bring back from injury <--H-->
 64. Snoopy, in his dreams <--->
 65. "Snowy" denizen of Florida <EGRET>
 66. Crayola earth tone discontinued in 1990 <--I-->
 67. ___ Kelly (Aussie outlaw) <--->
 68. Jury composition, presumably <----->
 69. Did a farrier's job on <--O-->

Down :

 1. Leg part <SHIN>
 2. Nasdaq alternative <N--->
 3. Twiddling one's thumbs <I--->
 4. Couples with a driver <F--->
 5. Brawl "souvenir" <F---I->
```

Fig 5.10. Puzzle Information in Text File

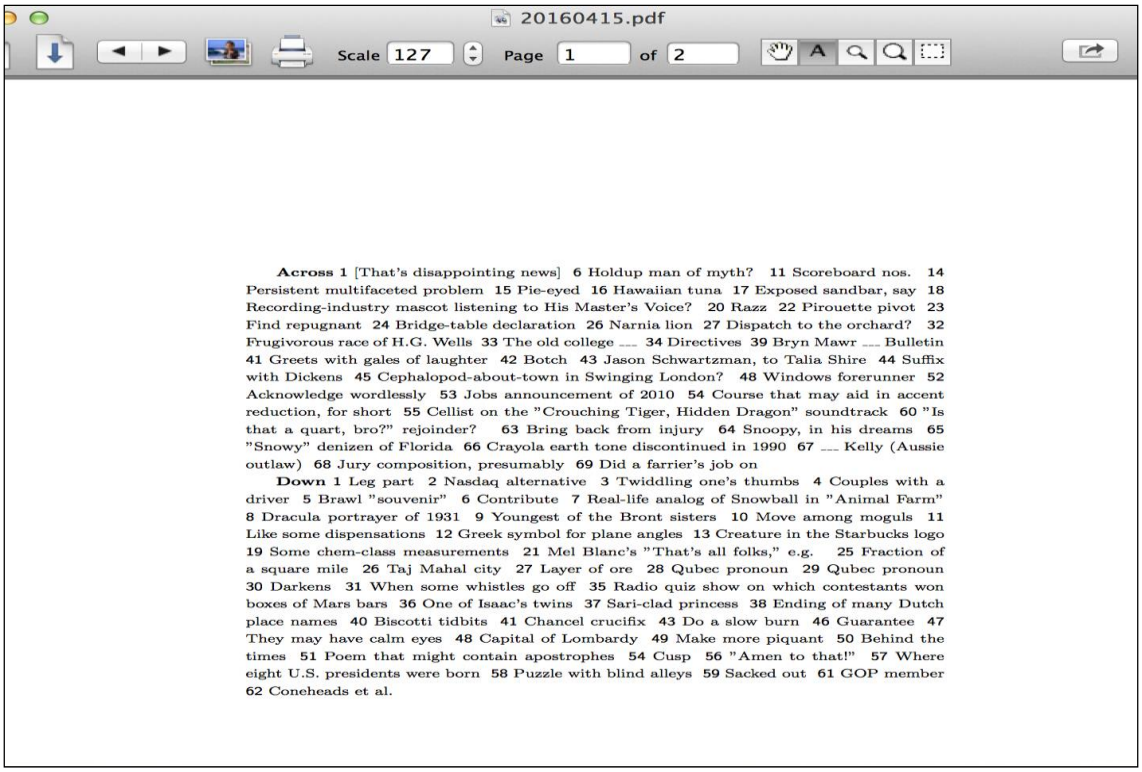


Fig 5.11. Puzzle saved as .tex file converted into PDF - Across/Down Clue List

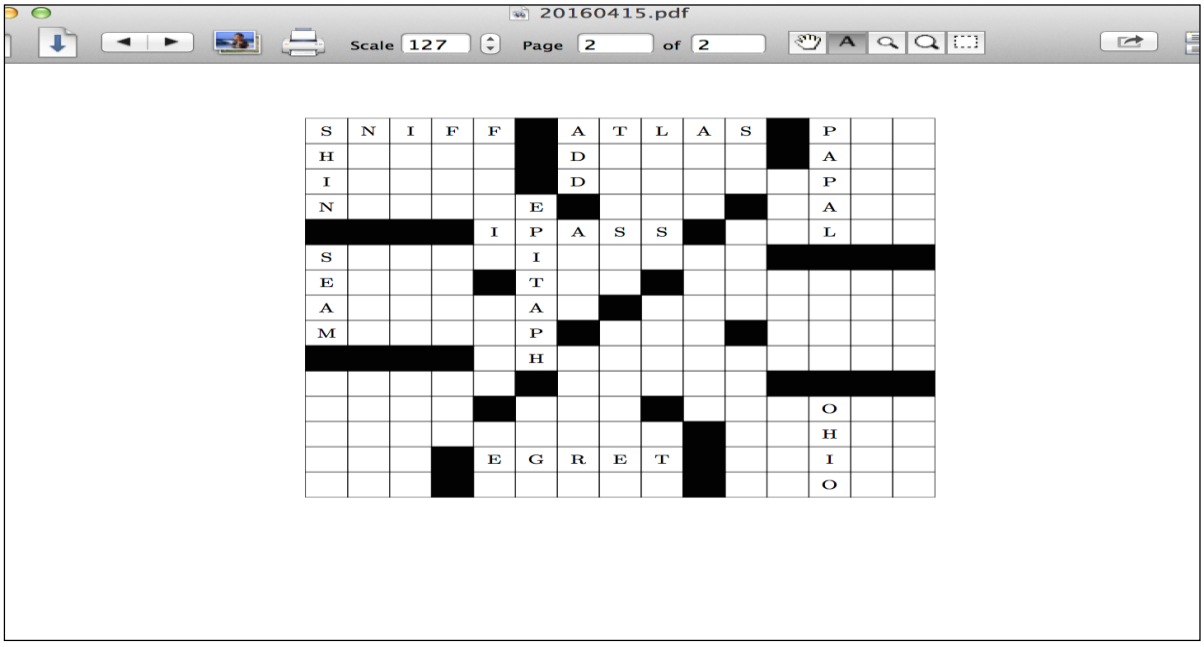


Fig 5.12. Puzzle saved as .tex file converted into PDF - Current State of Grid

5.1.2 Command-line UI

The command-line UI has a display option to display the option menu containing the features supported by this version.

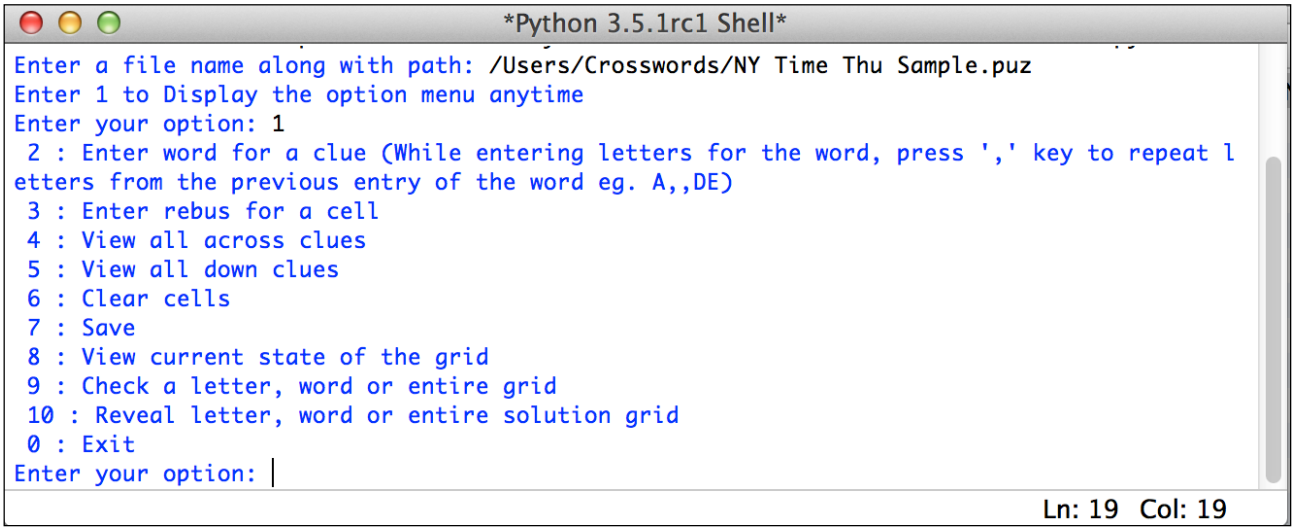


Fig 5.13. a) Display Option Menu

- Options are numbered using numbers from 0 -10. Users have to re - enter their choice if they have entered an invalid option or invalid clue number.

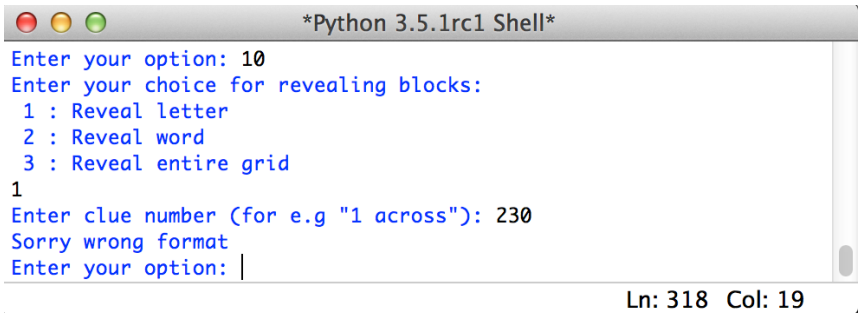


Fig 5.13. b) Handling an Incorrect Entry

- **Options 4 and 5** allows the user to view the Across and Down clues and their corresponding words in the current grid respectively.

Fig 5.14. a) Display Across Clue List

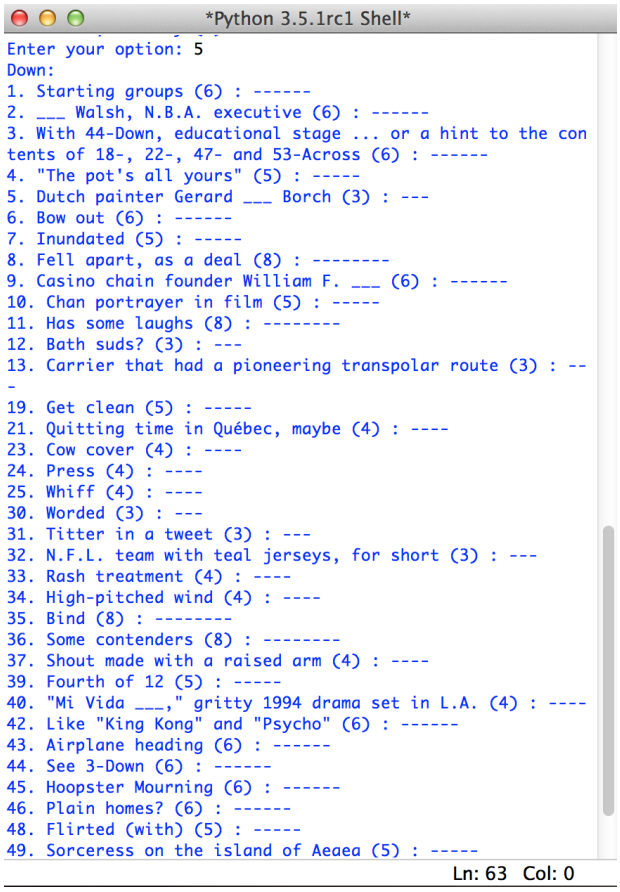


Fig 5.14. b) Display Down Clue List

- **Option 12** displays the notes to the user.

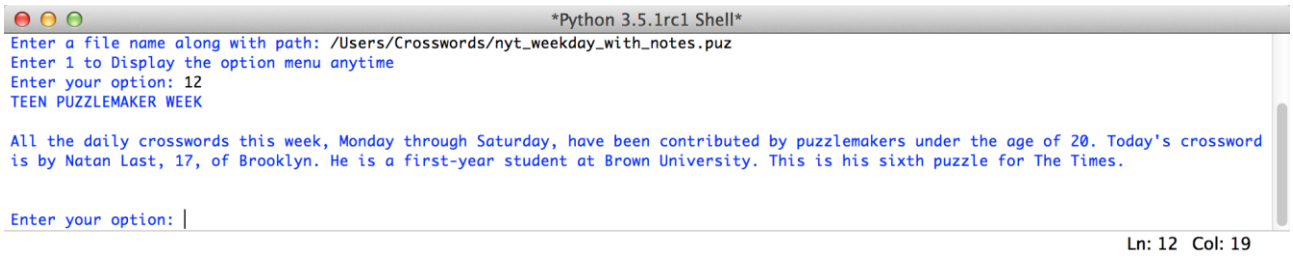


Fig 5.15. Display Notes

- **Option 8** displays the current state of the grid.
- **Option 2** can be used to enter a word for any of the clues in the clue list.

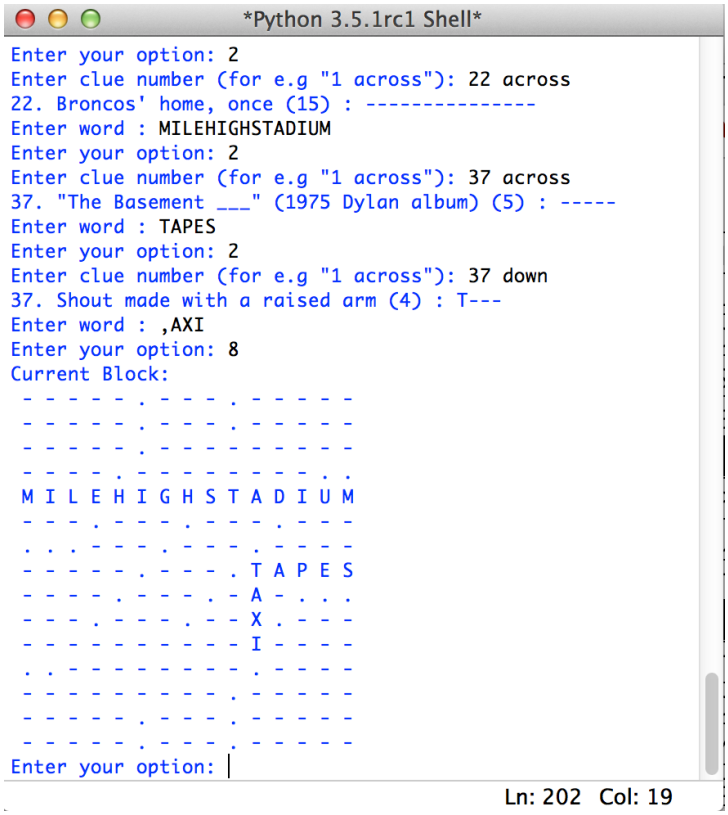


Fig 5.16. Enter word for a clue, Display current state of Grid

- **Option 3** allows the user to enter a rebus word for any of the unshaded cells selected by the user.

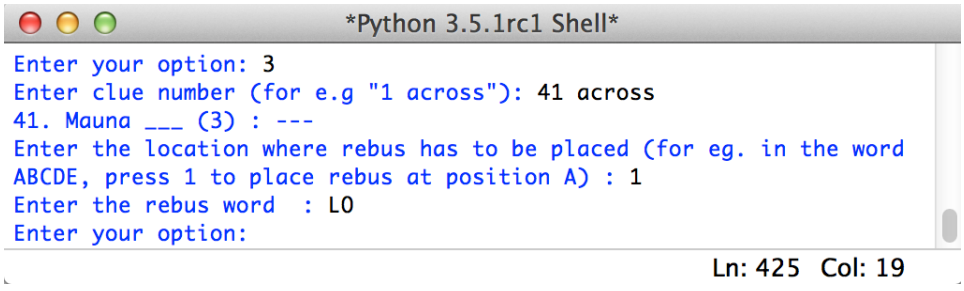


Fig 5.17. Enter Rebus in a word

- Users can access the check menu using **option 9**. Users can check a letter in the grid, word in the grid or the entire grid. The check attaches a correct/incorrect tag to the entries of the corresponding cells that has to be checked and displays it to the user.

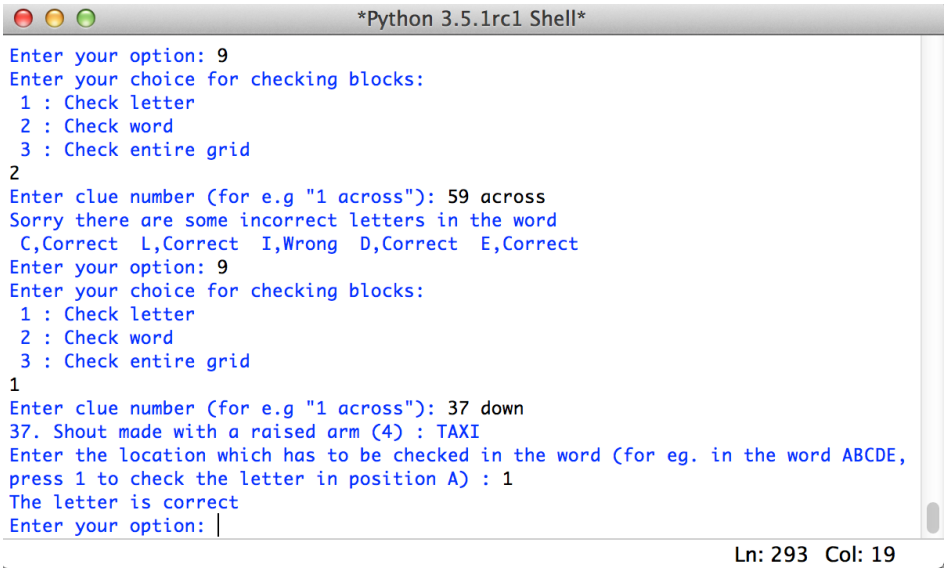


Fig 5.18. a) Check Word, Check Letter

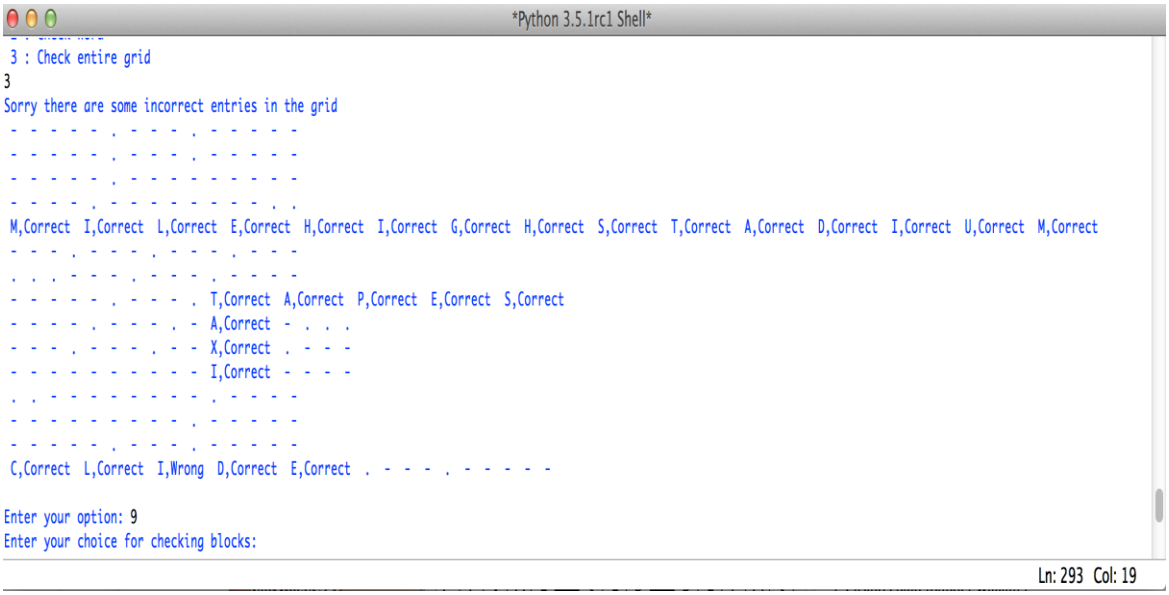


Fig 5.18. b) Check Entire Grid

- Option 10 allows the user to reveal a letter, word or the entire solution grid.

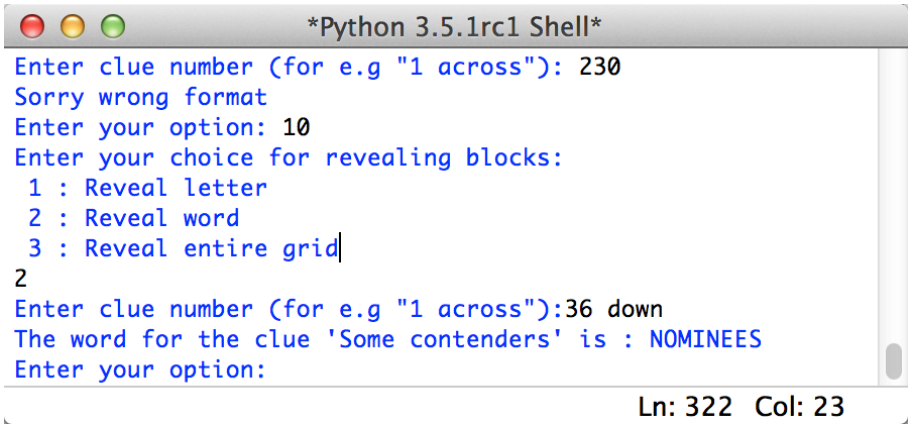


Fig 5.19. Reveal Option

All of the above mentioned features work in a similar way for both the .puz and .ipuz versions of Command-line UI. The only feature that differs is **Option 7** which is the save feature.

- .puz version provides users with options to update changes to the .puz file and copy the puzzle information to a text file.
- .ipuz version allows users to update changes to the .ipuz file, create a .puz file with the puzzle information available and to copy the puzzle information to a text file.

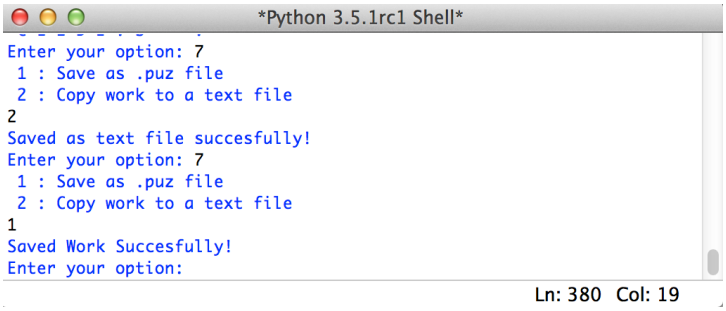


Fig 5.20. a) Save Option (.puz)

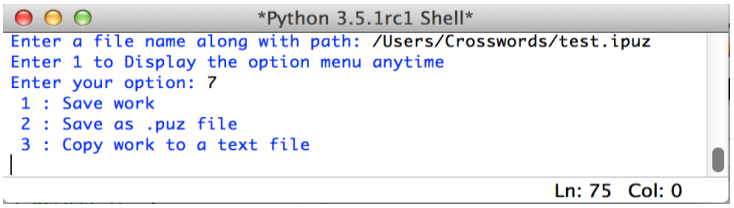


Fig 5.20. b) Save Option (.ipuz)

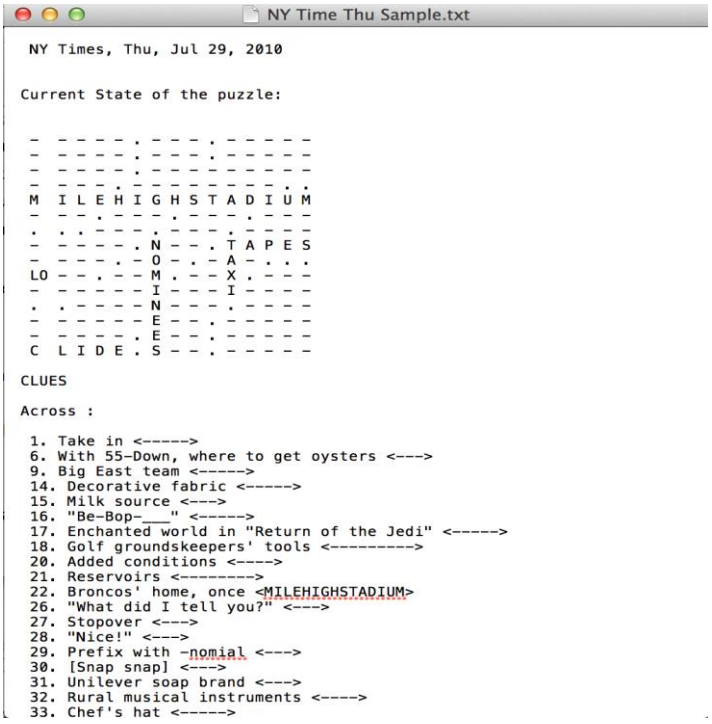


Fig 5.21. a) Puzzle saved as text file

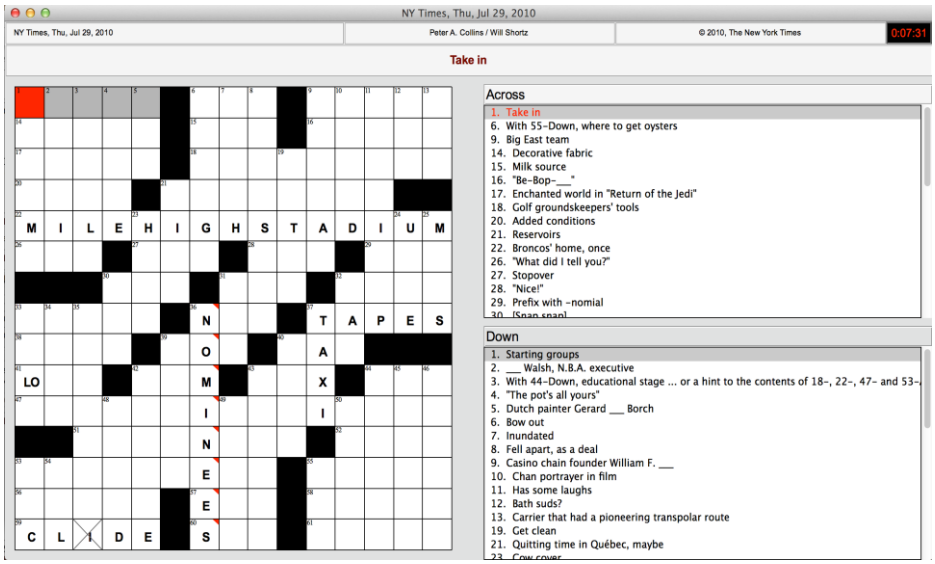


Fig 5.21. b) Puzzle saved as .puz file

5.1.3 Create Crossword Puzzle UI

CreateCW .puz/.ipuz module of OpenCWX allows users to create their own crossword puzzle or edit an already existing crossword puzzle in .puz or .ipuz format.

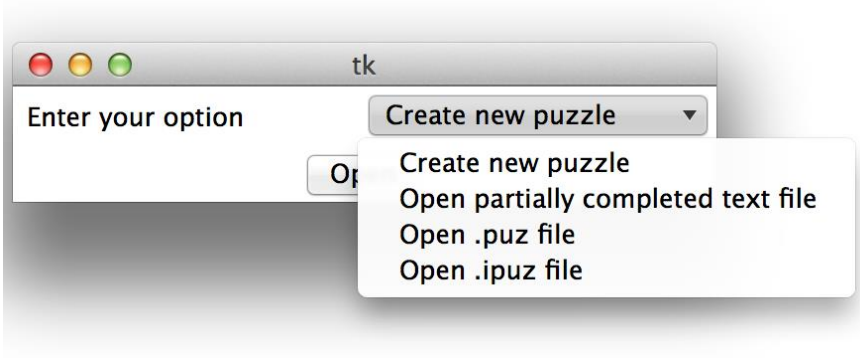


Fig 5.22. Option Menu

- When the users choose to create their own puzzle, the initial window allows the user to enter immutable data related to the puzzle like title, author, copyright, size of the grid and any notes that needs to be attached along with the puzzle.

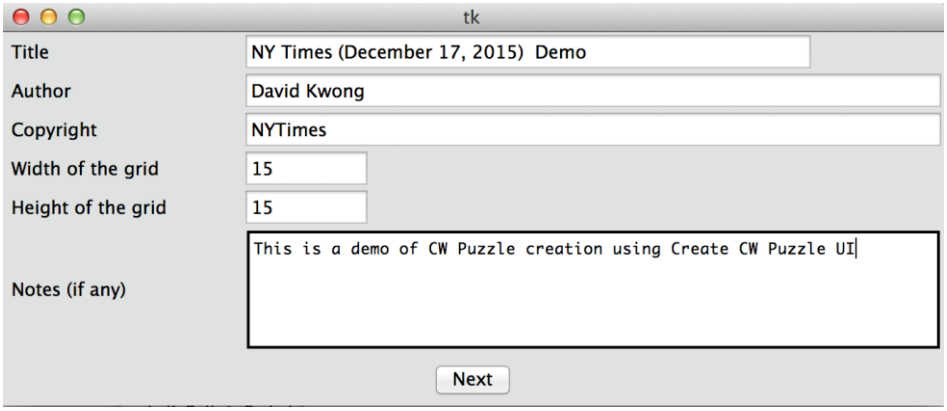


Fig 5.23. Enter Puzzle Information

- A grid is constructed in the next step, based on the width and height entered by the user.

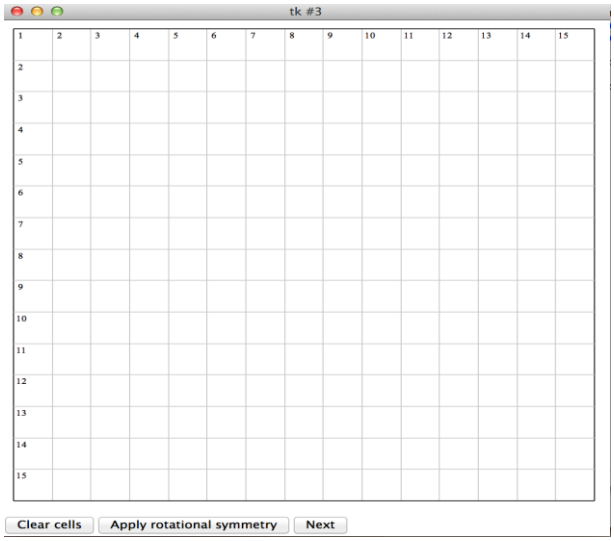


Fig 5.24. Grid before shading cells

- Users can now edit this grid by clicking on the cells to create shaded cells. 'Apply rotational symmetry' command applies rotational symmetry to the grid based on the current shaded cells so that the grid will look exactly the same when it is rotated by $180^{0.}$

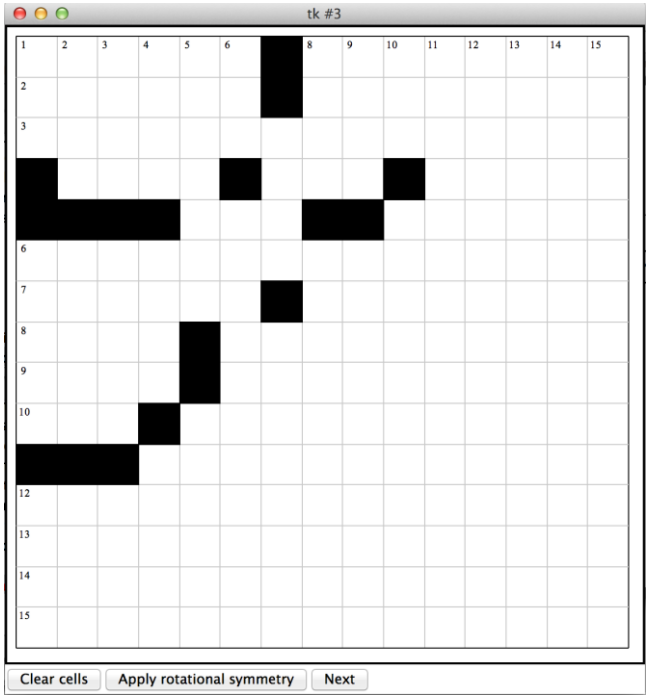


Fig 5.25. a) Before Rotational Symmetry

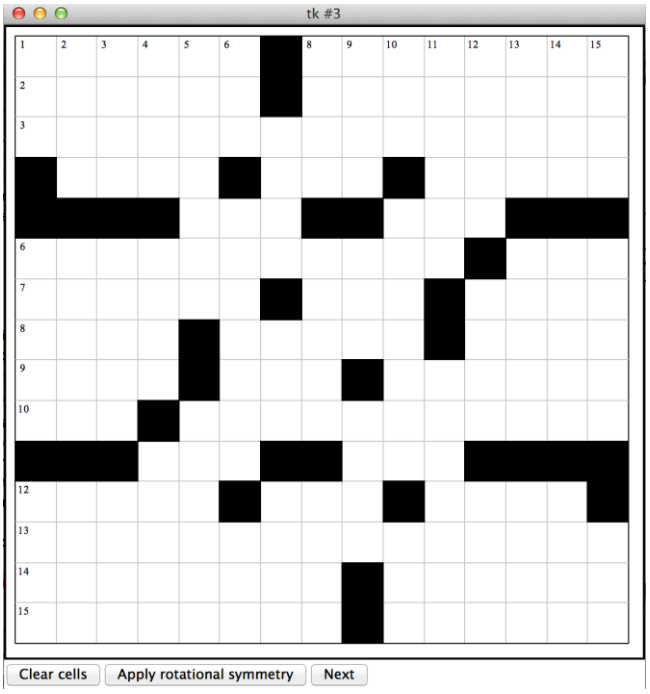


Fig 5.25. b) After Rotational Symmetry

- The next step that follows grid design, allows users to enter Across / Down clues and fill the solution grid.

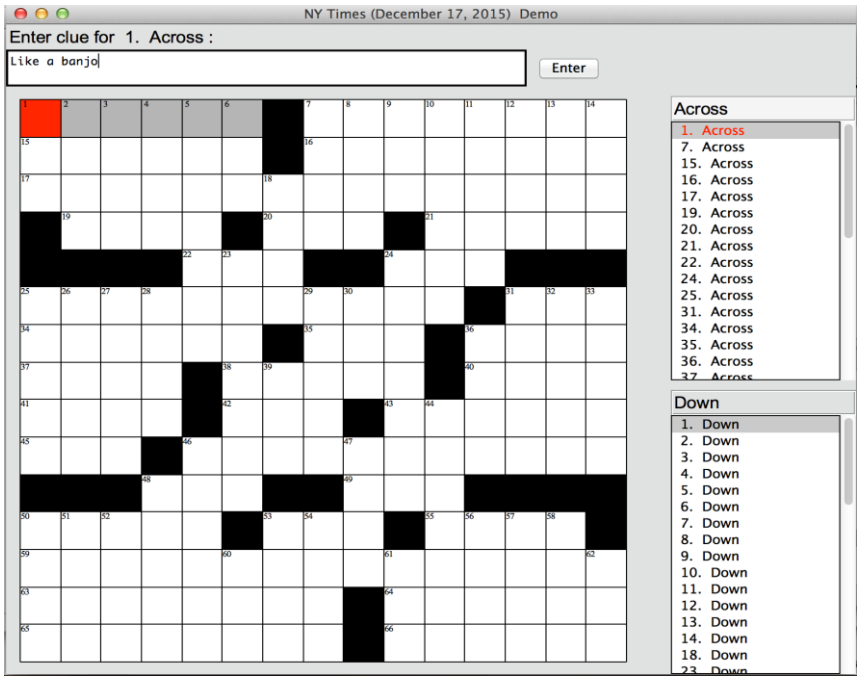


Fig 5.26. Enter Across/Down Clues, Fill Solution Grid

- Users can also construct a rebus puzzle, using the multiple entry option to enter a rebus fill.

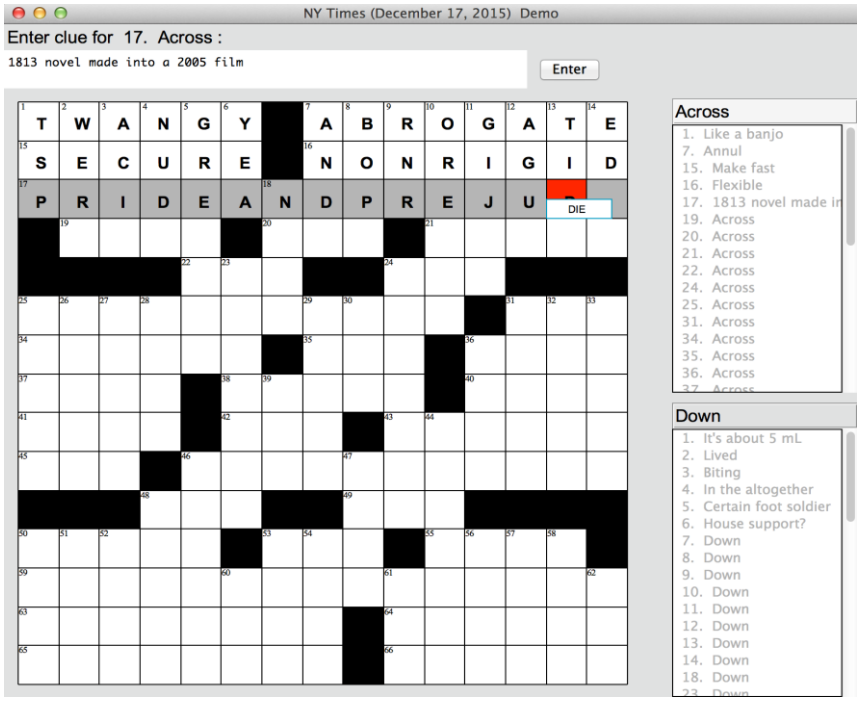


Fig 5.27. Rebus Entry

- Users can save a partially completed puzzle as a text file. This text file can be used to construct the puzzle anytime later.

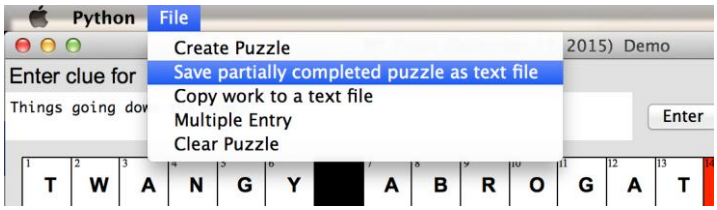


Fig 5.28. Save Partially Completed Puzzle as Text File

- Once all the Across/Down clues and solution grid has been filled, users can create the puzzle in the format of their choice (.puz/.ipuz).

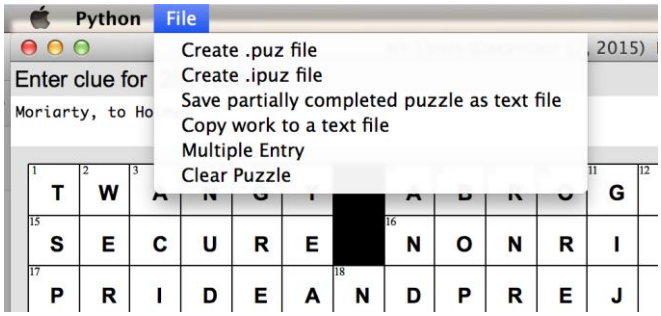


Fig 5.29. Create puzzle in .puz/.ipuz format

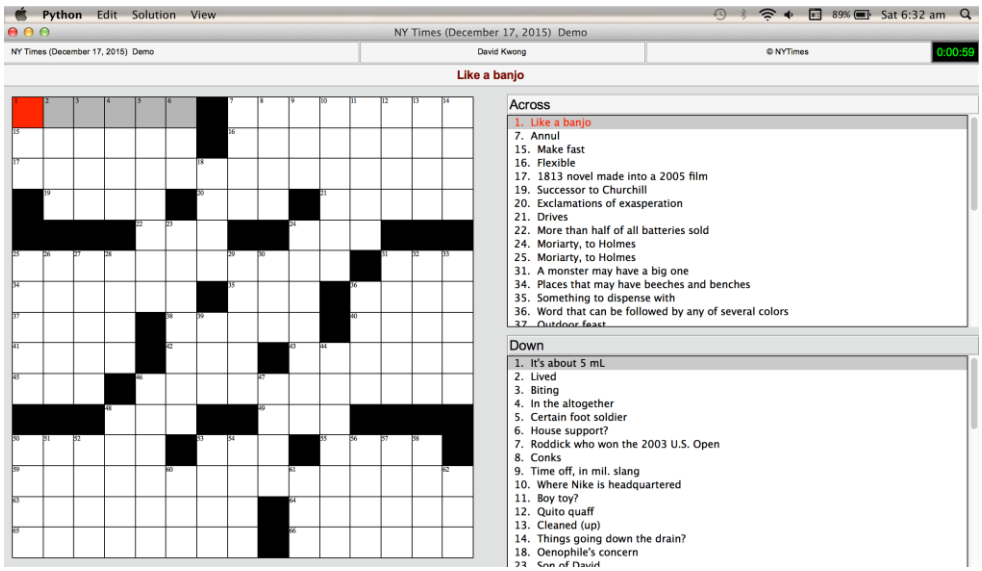


Fig 5.30. Puzzle created in .puz format using Create CW Puzzle UI

- **CreateCW LaTeX** module provides the users a GUI to edit the solution grid and Across, Down clues before converting a .tex format puzzle into .puz format.

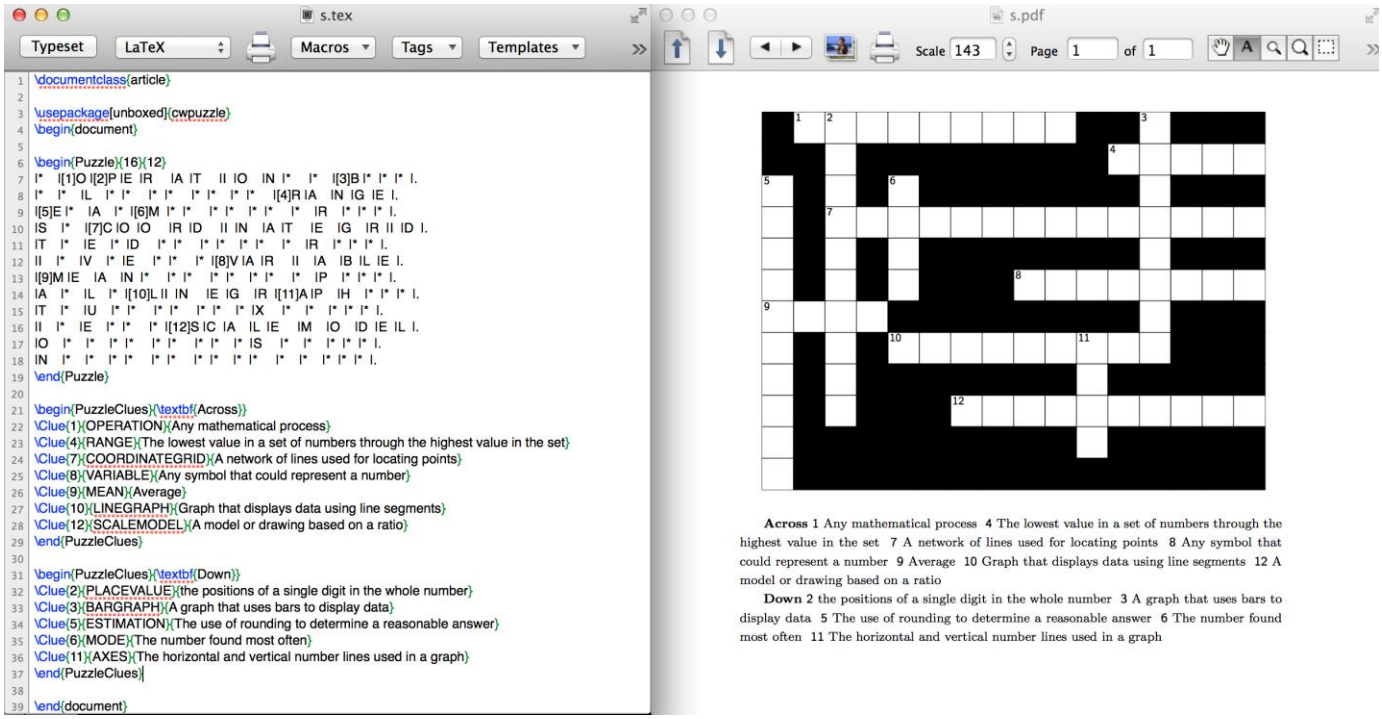


Fig 5.31. Create puzzle in .puz/.ipuz format, Puzzle in .tex format converted into PDF

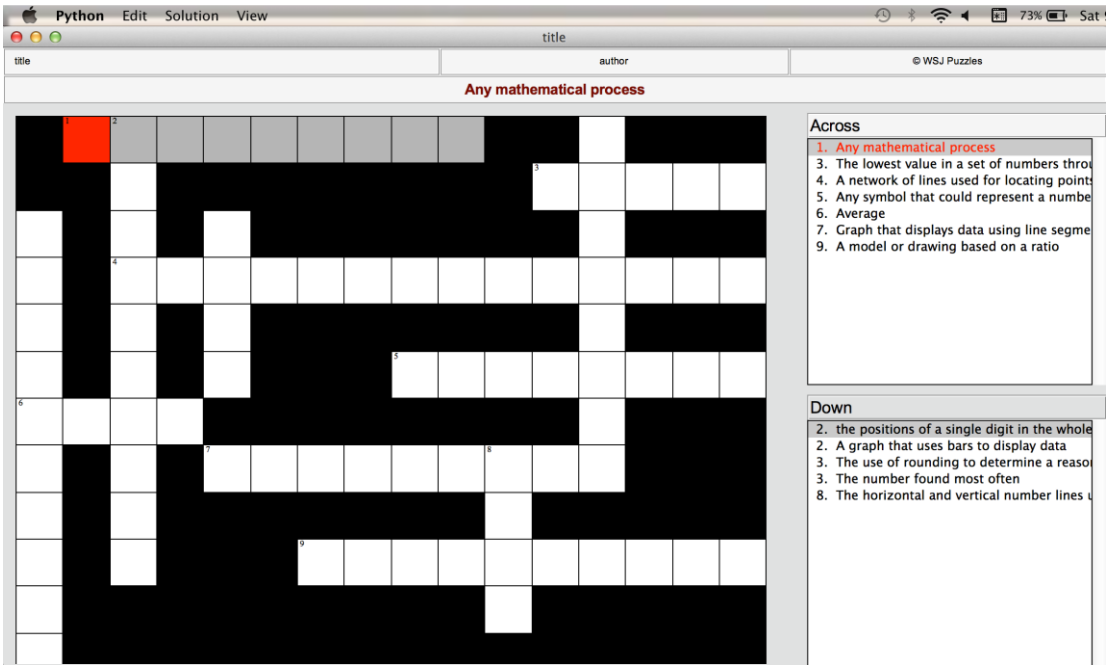


Fig 5.32. GUI to edit .tex format puzzles

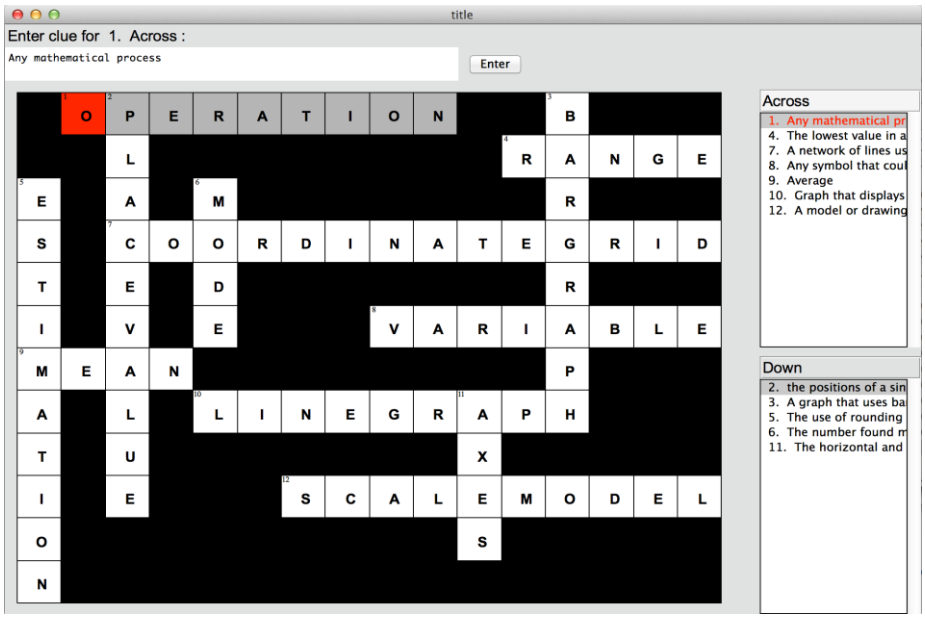
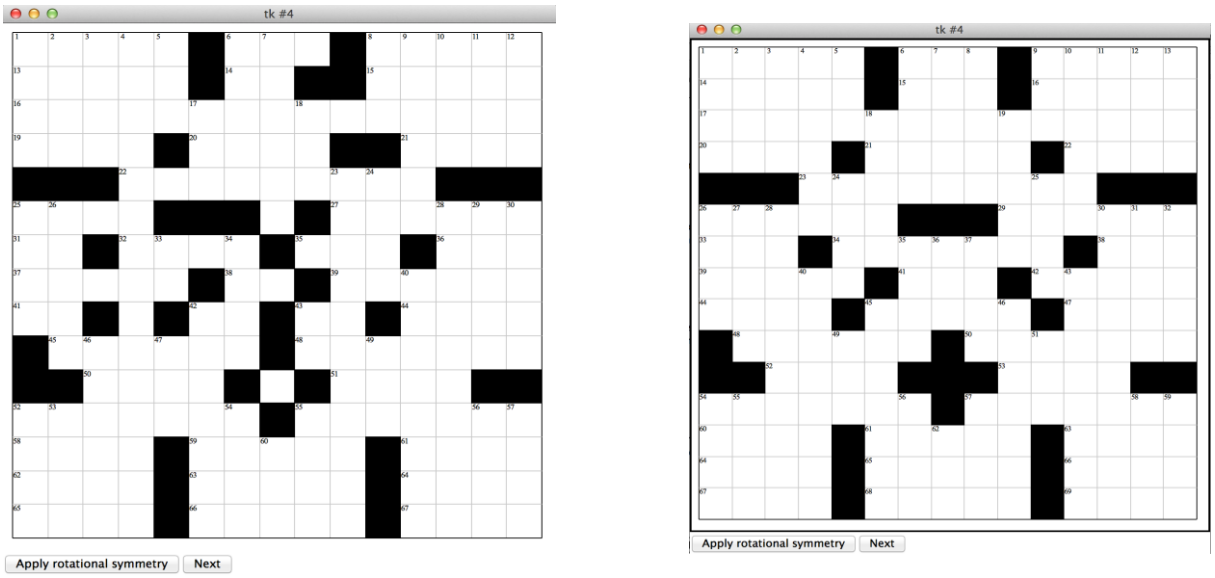


Fig 5.33. .tex format file converted to .puz format

- WSJ Converter provides a GUI that allows the user to edit the puzzles that are currently in .tex format and convert it into a .puz format puzzle.
- Create CW PDF reads the puzzle information from the PDF file and predicts the pattern of the puzzle's grid based on the layout of the PDF file. It then displays the predicted pattern to the user and allows the user to manually edit the grid if it does not match the real pattern.



(a) Predicted Grid **(b) Grid after it is manually edited**

Fig 5.34. WSJ to Puz Converter for puzzle in Fig.4. 2

- After the grid construction, as a next step; the Across/Down clues read from the PDF file are displayed to the right of the screen; users can edit the clues if necessary. It also provides an option for the users to fill the solution grid before constructing the puzzle

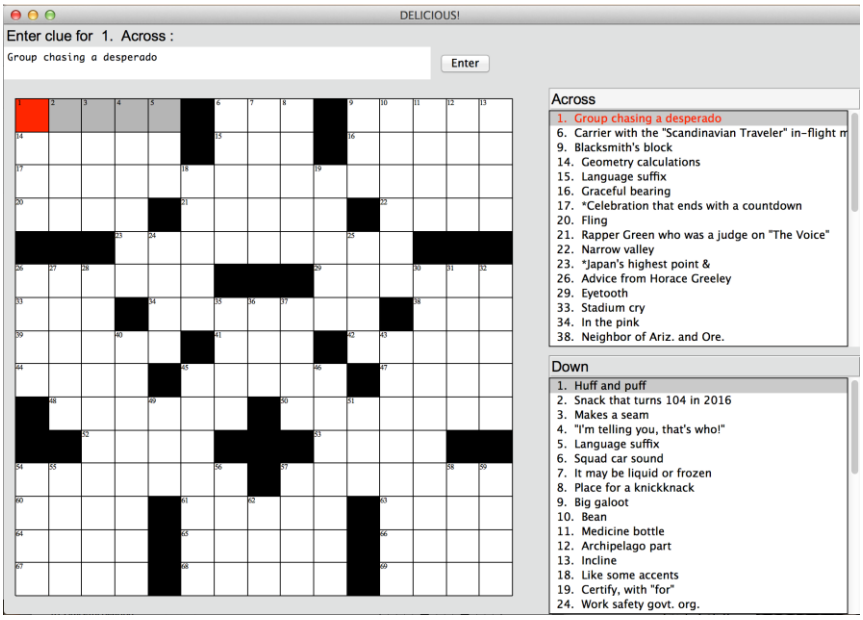


Fig 5.35. Edit Across/Down clue list and Solution Grid

- Once the previous step is completed, the puzzle can be saved in .puz format.

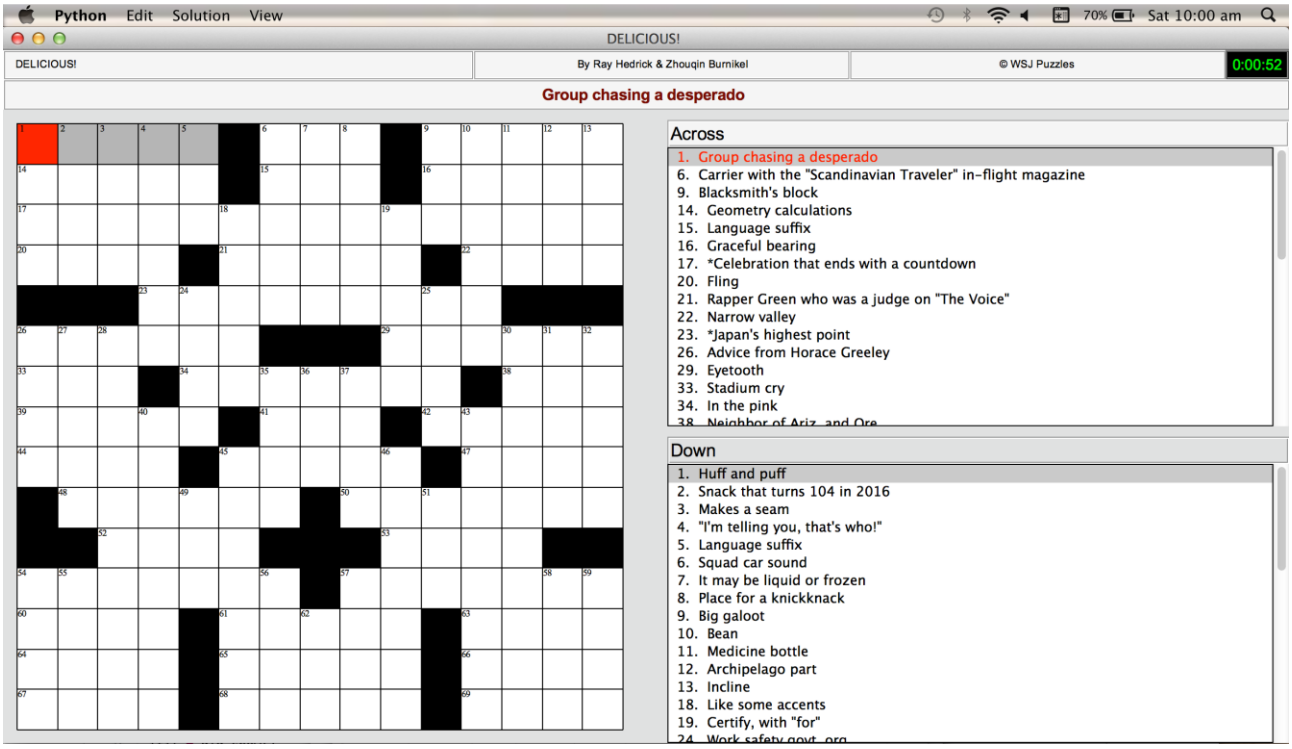


Fig 5.36. Puzzle converted to .puz file format

5.1.4 Crossword Puzzle Solver

The Solver module of OpenCWX reads the puzzle in .puz format. Once the grid is filled with values, it displays the predicted fills for each of the Across/Down clues and the final grid (based on the predicted fills) and the time taken to construct the grid.

An empty list ([ ]) is displayed if no satisfying value can be found for a particular clue and a cell with '-' value in the grid indicates that a satisfying value could not be assigned for the cell based on the constraints and available fills.

Fig 5.37. Puzzle to be solved by Solver

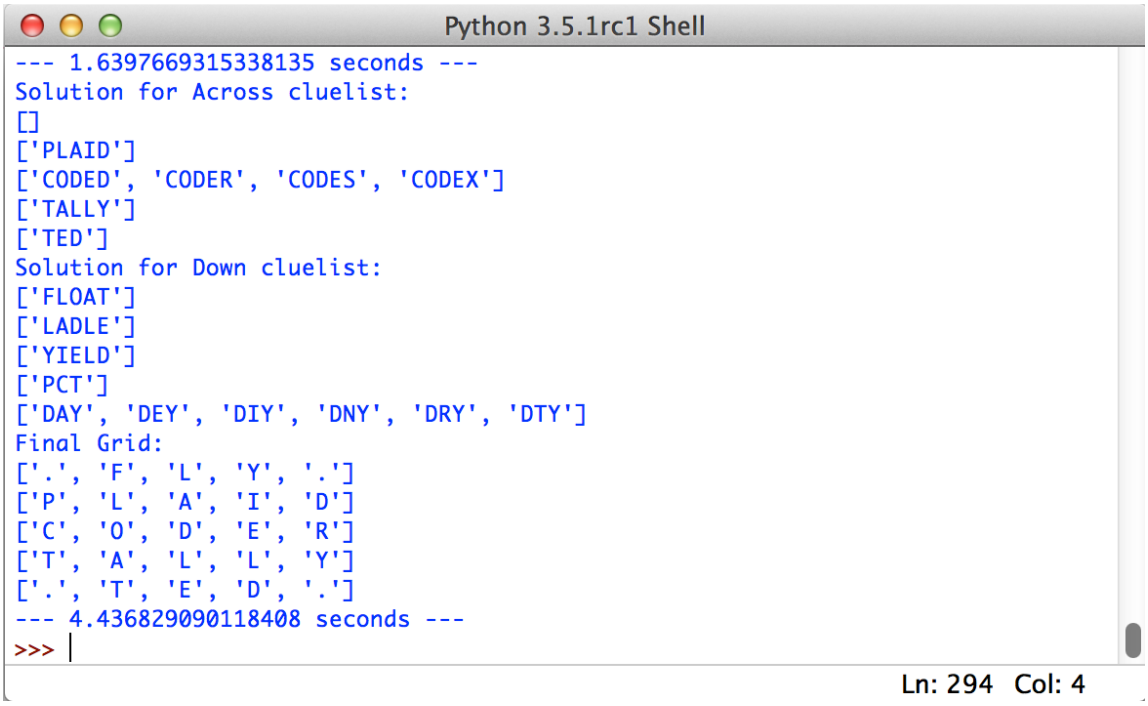


Fig 5.38. Results displayed by Solver

5.2 RESULTS

The New York Times mini (5x5) crossword puzzles were used as test cases and 25 such puzzles were solved using OpenCWX's Crossword Puzzle Solver. A short description of the obtained results is given in the table below:

Puzzle	No. of Letters	No. of Correct Letters	Solver Accuracy (%)	Time Taken (in seconds)
Jan 1, 2016	21	20	95.2	16.58
Jan 2, 2016	23	22	95.6	100.16
Jan 3, 2016	21	21	100	5.10
Jan 4, 2016	21	21	100	4.85
Jan 6, 2016	25	25	100	111.93
Jan 10, 2016	21	21	100	6.62
Jan 12, 2016	21	21	100	7.40
Jan 13, 2016	23	23	100	7.43
Jan 15, 2016	21	21	100	5.23
Apr 1, 2016	21	21	100	9.55
Apr 2, 2016	23	23	100	8.55
Apr 3, 2016	23	19	82.6	72.57
Apr 4, 2016	21	21	100	15.91
Apr 5, 2016	23	23	100	6.95
Apr 6, 2016	21	18	85.7	22.14
Apr 7, 2016	23	23	100	4.81
Apr 9, 2016	21	17	80.9	20.25
Apr 10, 2016	23	22	95.6	8.77
Apr 11, 2016	21	21	100	6.37
Apr 12, 2016	21	21	100	26.62
Apr 13, 2016	21	21	100	17.12
Apr 14, 2016	21	21	100	36.47
Apr 15, 2016	21	21	100	40.69
Apr 16, 2016	21	19	90.4	65.09
May 1, 2016	21	20	95.2	9.84

Accuracy rate greater than 95% Accuracy rate lesser than 95%

CONCLUSION

OpenCWX can be viewed as an all in one package that helps people to create, edit, interact and convert crossword puzzles available in different formats. Solvers can also take advantage of the CW Puzzle Solver module to get hints to the clues or entire solution grid if they encounter a locked puzzle.

The accuracy rate of the solver was found to be 96.8% when tested on twenty five of The New York Times Mini Crossword Puzzles. Accuracy in detecting the grid of a Wall Street Journal crossword puzzle can be improved by including efficient computer vision algorithms in Create CW Puzzle module, to scan the PDF file containing the crossword puzzle.

The solver can be extended in the future by including a module to check for synonyms corresponding to the solution entries in the list thereby extending the chances of filling the entire grid which might be helpful while dealing with 15x15 crossword puzzles. Multithreading option can be included in order to improve the clue database search time significantly when dealing with puzzles with a large solution grid. A module can also be included to analyse the clues to determine the part of speech of the solution.

BIBLIOGRAPHY

1. Matthew L. Ginsberg, Dr.Fill: Crosswords and an Implemented Solver for Singly Weighted CSPs, December 2011.
2. Gerd Neugebauer, A LaTeX Package for Typesetting Crossword Puzzles and More, January 2014.
3. Simeon Visser, ipuz Documentation, December 2015.
4. Creating Across puzzles using Across Lite, Literate Software Systems.
5. https://code.google.com/archive/p/puz/wikis/FileFormat.wiki
6. https://github.com/alexdej/puzpy
7. http://www.cruciverb.com
8. http://communicrossings.com/