Scrambler

A Crossword-Like Computer Game

24 February 2021

This manual is a work in progress, and likely contains errors, omissions, and duplications.

1 Introduction

Scrambler is a computer version of a crossword style board game. It is *not* the proprietary game with which we're all probably familiar. There are far too many differences in rules and presentation.

Only one human player plays; the user may select the number of computer opponents, from one to three, although the game is by far best with one.

Scrambler is strictly text-based and contains no graphics. It is meant to be played on a computer terminal or equivalent, and isn't in any way flashy or modern. There are many other implementations of similar games which are up-to-date, colorful, and with computer players that are very much stronger. But *Scrambler* is small, fast, and fun.

Scrambler was originally written by Dr. James A. Cherry in 1992 (the game title was different at that time). There appear to have been a few subsequent updates by Dr. Cherry and/or others; the provenance is not clear. I am unable to find current contact information for Dr. Cherry.

This current version represents an extensive update and enhancement by me (Bob Newell, `bobnewell@bobnewell.net`), with the intention of adding useful features. Any problems or degradations introduced are of course my sole responsibility.

Scrambler currently is only developed for Linux-like systems. It has been built and tested (to limited degrees) on Linux Mint 19 and 20, Arch Linux, and Termux running on Android tablets and phones. It might be made to run on Mac or Windows but that's beyond the scope of this work. (If you do get it to run on those platforms please let me know.)

This manual is a combination of Dr. Cherry's original 'man' page and my own edits, rewrites, and additions.

2 Installation

Installation is simple. Edit the Makefile to include or remove the options you want (there aren't many), then type `make`. Or you can just try to run the `scrambler` executable as-is. As delivered, the LOOKUP option is active and you'll need to recompile if you don't want it. In fact recompiling is recommended and likely necessary in any event, due to different library environments on different systems.

Then copy `scrambler` and `scramblerdict` to wherever you wish to put them. They must be in the same directory, and `scrambler` must be run from that directory. (Well, there *are* options. See the Makefile.)

That's it.

Requirements for compilation are minimal. You need the standard GCC tool chain and the ncurses library, including header files. `curl` is also recommended to be in your command path and is necessary for the LOOKUP option.

3 Rules of Scrambler

3.1 Object

The object of the game is to score points by forming *Scrambler* words (a subset of valid English words) in a crossword-like grid. Words are formed using *tiles,* each of which has a letter on it. The player with the highest score at the end of the game is the winner.

3.2 Letter Tiles

Each letter has a particular score associated with it. For example, the letter **E** is worth one point, while **Q** is worth ten points. There are a fixed number of tiles with each letter. For example, there are twelve **E** tiles, and only one **Q** tile. There are also two *blank* tiles, which may be used as any letter. Once a blank tile is placed on the board as a certain letter, that letter may not change.

3.3 Game Play

Each player starts with seven tiles, which are placed in a virtual *rack.* These tiles are drawn from a shuffled pool (or *bag*) of one hundred tiles.

The first player to move must play a word that covers the center square; as well, this player must place two or more of their tiles. After the first move, players take turns placing one or more of their tiles on the board, so as to form new words in the standard crossword-like manner: new words must read across or down, not diagonally.

New words may be formed in one of three ways.

1. By adding a tiles to an already existing word, e.g., adding **ABLE** to **READ.**
2. By adding tiles that intersect another word, e.g., spelling **WARY** by placing **W, R,** and **Y** around the **A** of **REAL.**
3. By spelling a word in one direction and forming new words in the other direction, e.g., by adding **FOX** next to **PIG** so that the **I** of **PIG** is used to spell **IF** with the **F** from **FOX** and, similarly, the **G** is used to spell **GO.**

If all the new words formed are valid *Scrambler* words, that player's score is increased by the value of the tiles in *all* new words formed. The player then draws new tiles from the bag until the player again has seven tiles in the rack, or until there are no more tiles left to draw.

If *any* of the new words formed are not valid *Scrambler* words, the player scores zero for that turn, and must remove the tiles just placed from the board and return them to the rack.

Valid English words in *Scrambler* constitute any word so long as it is not an abbreviation (**LTD, CORP**); is not normally capitalized (**MATTHEW, LONDON**); does not contain punctuation (**COLD-HEARTED, CAN'T**); and is not a prefix or suffix (**PRE, ATION**). Foreign words that are used frequently in English are allowed (words such as **CIAO** and **QUO**). Slang and colloquial words are also allowed. If the word is in the *Scrambler* dictionary file, it is considered valid. If the word in not in the dictionary file, there is a compile option to cause an on-line lookup from an up-to-date valid source, whose decision is final. The dictionary will be further discussed later on.

Once a tile has been placed on the board, it may not be moved.

3.4 Special Squares

Certain board squares are special, in that they increase the value of a letter or a word. Double and triple letter squares increase the value of a tile by double and triple, respectively. Double and triple word scores increase the value of a word by double and triple, respectively.

Double and triple letter squares are counted *before* double and triple word squares. The effect is cumulative; the total letter count, including doubled or tripled letters, will be multiplied by a double or triple word square if such a square is also covered. For example suppose the word **PUNTER** is played with **P** on a triple-letter square and another tile on a double-word square. The **P** tile is tripled from 3 to 9, and the rest of the tiles add 5 additional points for a total of 14. This is then doubled to 28.

Furthermore, if multiple word multipliers are covered, the effect is also cumulative. For instance if *two* triple word squares are covered (yes, it can happen) the multiplier is 3 times 3, or 9.

These special squares only count on the turn in which they are covered; the squares are no longer special in subsequent turns.

A player who uses all seven tiles on one turn gets a premium of fifty points on top of the regular score for that turn. This is called a *bingo*.

3.5 Exchanging Tiles; Passing

Instead of placing tiles, a player may skip playing tiles, and draw between zero and seven new tiles. The player selects which rack tiles to replace, puts them back in the bag, and then draws the same number of tiles from the bag again. A player doing this scores zero for that turn. Tiles can only be replaced if there are seven or more tiles left in the bag. With less than seven tiles in the bag, you cannot exchange tiles.

Drawing zero tiles is equivalent to passing the turn. But it's simpler to just use the pass option directly.

Some players institute a rule that a player can exchange a blank tile on the board for the equivalent tile (whatever the blank tile stands for) on the rack. This is not allowed in *Scrambler*.

3.6 End of Game

The game ends when there are no tiles left to draw and one player has no tiles left to place ("goes out"), *or* when all players in sequence make a play that scores zero. Zero score occurs on a pass, a tile exchange, or a play of an invalid word.

This differs from the "six zeroes" tournament rule, in order to accommodate more than two players. The flaw in the *Scrambler* rule is some excessive strictness. For instance in a two-player game if the computer passes and you then enter an invalid word, the game ends at once, with no second chances.

If no player has gone out (by reducing to zero tiles on the rack) each player's score is adjusted by subtracting from it the value of each tile remaining in their rack. But if one player "goes out", that player's score is instead increased by the twice the sum of the value of all unplayed letters in the other players' racks. (There is then no subtracting done.)

The player with the highest score wins the game. In the event of a tie, the player with the highest score prior to the score adjustment is the winner.

4 Scrambler Screen Layout

The upper-left side of the screen lists the special board symbols used in *Scrambler.*

- # Double letter score.
- + Triple letter score.

- **o** Double word score.
- **@** Triple word score.
- ***** Blank tile. This is the character displayed in a player's rack when to represent a blank tile. Once the tile is placed on the board, it is shown in lower-case as the letter it now represents.

The lower-left side of the screen lists the value of each tile, as well as the total number of tiles of each type remaining to be seen. For example, **9A1** means that there are a total of nine tiles remaining to be seen with **A** on them, and that each tile is worth one point. "Remaining to be seen" means a tile is either in the bag or on an opponent's rack.

The center of the screen shows the *Scrambler* board.

The upper-right of the screen displays the number of tiles left in the bag to draw. As well, it displays the number of passed turns, if any.

The central-right of the screen shows all players' tile racks. The computer players' tiles are represented by dashes. As well, the computer players' skill levels are displayed in brackets next to their designations (e.g., CPU1). Finally, each player's score is shown here.

The lower-right of the screen indicates the location of any blank tiles that have been placed on the board, as well as the letter they represent.

The bottom of the screen is the message and input area.

5 Playing Scrambler

The user must select the number of computer players to play against. The game is definitely best with just one, but if you wish you can have up to three. Then, each player draws a tile. The player with the lowest tile alphabetically starts the game. A blank tile is considered lower alphabetically than all other letters.

The user then selects the skill level of each computer player. Skill levels go from 1 (very easy to beat) to 8 (quite a bit harder to beat for most players), and skill level 9 ("expert mode" although not really). At the "expert" level the computer plays the highest-scoring move every time (while ignoring rack leave, strategic position, and all the other things that true experts take into account).

When it is the computer's turn to play, it searches the board for a move. After the search, the computer chooses its move based on its skill level. If it could not find a move, it will say so. The computer never exchanges tiles (a major strategy flaw).

When it is your turn to play, the cursor starts at the upper-left corner of the board. The following keys may then be used.

- **h** or **left arrow**. Moves the cursor left. If the cursor is on the left edge of the board, this key has no effect.

- **j** or **down arrow**. Moves the cursor down. If the cursor is on the bottom edge of the board, this key has no effect.
- **k** or **up arrow**. Moves the cursor up. If the cursor is on the top edge of the board, this key has no effect.
- **l** or **right arrow**. Moves the cursor right. If the cursor is on the right edge of the board, this key has no effect.
- **^L** (control-L). Redraws the screen. This is provided in case the screen should somehow become corrupted.
- **RETURN key**. Starts placing tiles on the board. If the cursor is over a square that is already occupied by a tile, this key has no effect. More on this key below.
- **SPACEBAR**. Lets you rearrange your tiles. Sometimes it's easier to see what words you can form if you group your tiles in your rack a certain way. This key allows you to regroup your tiles. Typing the letter of a tile in your rack moves it to the end of your rack. Use the asterisk for a blank tile. Pressing **RETURN** exits this mode.
- **.** (period). Draw new tiles. You can replace none, some, or all of your tiles and draw (potentially) new ones. Type this key, followed by the letters in your rack that you wish to replace. Pressing **-** (dash) or **BACKSPACE** will put your most recent selection back in your rack. Pressing **ESC** will cancel this operation, and pressing **RETURN** will ask you to confirm your selection. Exchanging zero tiles is equivalent to passing your turn, but there's an easier way.
- **p** Pass your turn. Confirmation will be requested. Do this if you don't think you have a valid move, or you just don't feel like playing any longer.

Once you press **RETURN** to start placing tiles, you can press either **h** or **v** to place your tiles horizontally or vertically, respectively.

Then, you simply begin typing. Each time you type the letter of a tile that is in your rack, the tile is placed on the board at the cursor's position. The cursor then moves to the next blank square in the appropriate direction. If you type a letter that is not in your rack, it is ignored *except* if you have a blank tile. In this case, the blank tile is placed on the board for you. If you type an asterisk and you have a blank tile, it is placed on the board as usual.

While typing, you may press **-** (dash) or **BACKSPACE** to remove the last tile placed from the board and put it back in your rack. Or, you may press **ESC** to cancel this particular placement all together. If you're satisfied with what you've placed, you may press **RETURN**. If you placed any blank tiles by typing ***** (asterisk), you will be asked for the letters that the blank tiles represent. Press **RETURN** to confirm your move, or **n** to alter it.

Your words are then checked against the dictionary. If all the words are in the dictionary, your score is increased and new tiles are drawn for you. If any of your words are not in the

dictionary, an (optional at compile-time) attempt to do an on-line lookup is done, using the `curl` command (which must be in your command path).

This lookup is fragile and can fail due to internet connection issues or an eventual server down condition, or even a change in the server lookup interface.

If the lookup fails, you are asked if said words are valid *Scrambler* words. You really should answer honestly! If any word is invalid, your tiles are removed from the board and replaced in your rack. If a previously unknown word is declared valid (either via lookup or user decision), it is added to the dictionary (temporarily with an option to save on exiting the program). Be aware that the computer players can then use the word too, and if it is a phony word and you later save the dictionary, you will have to live with it from then on.

Scrambler does all the drawing of new tiles automatically. It also does all the scoring, including the score adjustments at the end of the game. It can even handle multiple-way ties at the end of the game, should this unlikely event occur.

As just noted above, at the end of the game, if any words have been added to the dictionary, *Scrambler* will ask you if you want to save the dictionary. If you say yes, the old dictionary file is overwritten with the new one. Otherwise, the original dictionary remains intact. It would be good to have a backup of the dictionary in case anything goes wrong.

Note that you can edit the dictionary yourself (outside of the game), if you are brave enough and if you have made a backup.

An important point: often throughout the game, you will be prompted to **Press RETURN**. If you press the **q** key followed by **RETURN**, you will exit *Scrambler*. This is much safer than pressing an interrupt key or turning off the computer, which may lead into unexplored territory and could lead to loss of data, weird crashes, program loops, or increased global warming.

If you press the **n** key followed by **RETURN**, you can enter a brief note to be put into the GCG game output file. End the note with **RETURN** or abort it with **ESC**.

If you press the **s** key, your current game state will be saved. You will be prompted for a save file name. Your current game will then continue on. There are a few points in the game when saving is not possible; you'll be informed.

If you press the **r** key, you'll be prompted for a save file to restore. If that file exists and is valid, the game state will restore to the state it was in at the time the save in question was made. If the file doesn't exist, the current game just goes on. If the file exists and is invalid, you are warned about it. This does work nearly all the time, but it isn't foolproof and crashes are possible.

There are a few points in the game when restoring is not possible; you'll be informed.

You also have the option of doing a restore at the very start of the *Scrambler* program. You can enter **r** when prompted for the number of computer players, and that will then allow you to enter a save file to be restored. It's often desirable and perhaps best to restore a saved game right away, although there are situations in which you might want to do a restore in the middle of a game in progress, perhaps to explore variations within the same game.

6 Computer Players' Algorithm

For the curious, here is a brief description of how the computer players decide on their moves. This section is a bit technical.

If the board is empty, the computer simply searches the dictionary for any word whose length is between two and seven that can be formed with its tiles. It stores all possibilities, and computes the score for all possible placements of each word.

For the remainder of this section, a *legal move* is one which adjoins to the existing puzzle.

The computer players do an exhaustive search of the board, finding all positions in which a legal move can be made.

At each square, the computer determines how many of its tiles it can place and still be a legal move. This can be anywhere between one and seven tiles, inclusive. For example, the computer might be able to place any number of tiles between three and seven, and still be making a legal move. The computer then performs an exhaustive dictionary search for each of these numbers of tiles. In the example, the computer would perform five exhaustive searches.

The searching algorithm prunes the search tree in three ways.

1. For a given number of tiles to be placed (say three), all words formed will be of the same length (say four). Thus, the computer only need search all four-letter words.

2. Often one or more of the letters in the new word to be formed will be fixed, because the computer might be adding to a word that has already been placed on the board. For example, if the word **WENT** is on the board horizontally, the computer might be trying to add a word vertically to the **N.** Thus, any word that does not have the fixed letter(s) in the proper place(s) can be eliminated right away.

3. The computer might be forming other words in the opposite direction to which it is placing tiles. For example, suppose the computer is trying to form a word horizontally with three tiles. Furthermore, suppose that its first tile has to adjoin to a vertical word such as **PIT.** Then, the computer checks if it can form a valid word of the form **PIT**_ (where the underscore represents an unknown tile) before the exhaustive search. In this case, the computer would have to own an **A, H, S,** or **Y** to form **PITA, PITH, PITS,** or **PITY.** Otherwise, the search would be abandoned.

The computer stores all possible moves in a linked list sorted from highest-scoring move to lowest-scoring move. Once the search is complete, the computer then selects a move.

In so-called "expert" mode (level 9), the computer merely chooses the move at the head of the list.

On the other levels, the computer picks a move using a normal probability distribution function. The mean of the distribution is placed close to the highest-scoring move on level 8, and close to the lowest-scoring move on level 1. The standard deviation of the distribution

is one-fifth of the number of distinct scores that each move might give. For example, the computer might be able to make moves that would give it 4, 6, 7, 9, or 10 points. In this case, the standard deviation would be 1.

7 Files

Here are the main files used and generated by *Scrambler*.

- `scrambler` This is the executable file.
- `scramblerdict` This is the dictionary file. Dictionary files must be in the same directory as the `scrambler` executable. You can change this either by editing the header file `init.h` and recompiling, or by adding an option in the Makefile. See the Makefile for the correct syntax.
- `nnn.gcg` The GCG files are numbered sequentially from 001.gcg upwards. (Above 999, you're out of luck.) These files can be used as input to other programs such as *Elise* and *Quackle*. These files are fairly small.
- `nnn.eval.txt` The EVAL files are numbered sequentially from 001.eval.txt upwards. (Above 999, you're out of luck.) These files contain computer evaluation of each game position, plus several other useful statistics. For a full game, these files can get to be a bit large, so you might want to clean up once in a while.
- `scramblergame.log` This file is a cumulative log of major game events, including starts, restores, saves, game ends, quits, and aborts. Each line of the file shows the event type, the save or restore file if applicable, and the GCG and EVAL files associated with the event. The main purpose is to help relate GCG and EVAL file names with games played and alleviate at least a little of the confusion the GCG and EVAL naming conventions generate.
- `save files` These files are user-named and represent a game state, which can later be restored and the game resumed. These files are very small.

8 Dictionary Files

If desired, the user may supply a different version of the dictionary file. The dictionary file must have the following properties:

1. It must be sorted from shortest words (two letters) to longest words (fifteen letters).
2. Within each word length, the words must be sorted alphabetically from A to Z.

3. Words *must* be all capitals. **scrambler** ignores any word that contains punctuation, as well as any word longer than fifteen characters.

At present, *Scrambler* is not UTF-8 clean, let alone multicharacter capable. While some non-English dictionaries may be usable, you won't be able to do things like play *Scrambler* in Devanagari.

9 Play Issues

9.1 Word Challenges

The whole concept of challenging the computer player's word is non-existent in *Scrambler*. This is unfortunate, as it is an important part of the game, particularly at the expert level. The computer players only play words that are in the current dictionary. They never play phonies (as they see it) and can likewise never be challenged.

Furthermore, the human player can never get away with a phony. Effectively, all words not in the dictionary and not found via online lookup will be "challenged" and the challenge will always succeed. This is a huge advantage for the computer players.

9.2 Computer Logic Flaws

In expert mode, the computers always make the highest scoring play available, a big advantage. However, the "expert" skill level (which it really isn't) is reduced by several computer playing flaws.

- The computer *never* exchanges tiles even if it has no legal play.
- The computer doesn't have any concept of things like the rack "leave" and simply makes a play without caring about what tiles are left on its rack. An expert player wouldn't do this.
- The computer has no concpet of offensive or defensive play. It won't plan ahead to obtain or block access to high-value squares.
- The computer pays no attention to what tiles remain in the bag or what tiles the opponent might have on the rack.
- The computer will keep hard to play tiles like Q and Z forever, since it never makes exchanges, and very often suffers a sizable point penalty for this at the end of the game.
- The analysis only detects words that are already in the dictionary. The dictionary provided is not quite the tournament dictionary. Furthermore, on-line lookup doesn't function during move analysis.

9.3 Using Scrambler to Train

As just discussed, in vital respects this implementation differs significantly from high-level championship play. So what value does it have for training?

I believe for players from beginner to upper intermediate levels, *Scrambler* provides useful play exercise, in that practice with anagramming, board search, offensive and defensive play, etc., are all possible. But, as the computer doesn't respond in kind, the value of *Scrambler* for higher level training, especially major tournament training, is quite limited, and something like *Elise* or *Quackle* would be much better.

10 Summary of Changes Made in 2021

1. Most significantly, I've added GCG output. GCG is the standard game record file for this type of word game. I've tried to make the GCG output complete, and in limited testing it seems to work. This allows a game (full or partial) to be imported and analyzed by the big boys such as *Quackle* and *Elise*. Aborted partial games can also be continued on those platforms (but, alas, not restarted in *Scrambler*). GCG output files are numbered sequentially from 001.gcg upwards.

2. I've added an analysis output option. Analysis goes to files serially numbered upwards from 001.eval.txt. The analysis is not fancy; it is just a list of the 20 highest scoring plays from the rack and board position existing at the time, and a comparison with the actual move made. 20 plays aren't really a lot, since the same or similar play can often be made from different squares or in different directions (notoriously the case on the first move of the game). However listing all moves would just make for too much output.

   The analysis shows player name, turn number, rack contents, and an ASCII representation of the board position. Then it shows the 20 best plays in terms of score, highest first. At the end you see the actual play, its score and total game score, followed by the highest possible score (in parens). Then you see the percentage of highest score that the actual play achieved, and the average point score per turn and average percentage achieved per turn for all turns so far. It's actually some pretty good information if you keep in mind the limitations of just evaluating based on score.

   Analysis also shows total elapsed time used by the human player. (There was no sense in including this for the computer players as they move very quickly.)

   Warning: analysis files aren't tiny. Delete the unused ones from time to time.

   I did all of this so that *Scrambler* would be at least somewhat self-contained, and usable on platforms like Termux on Android, where the option to run *Elise* or *Quackle* would involve a lot of complex setup. But see the discussion above about the limitations of the computer play analysis; the analysis, while definitely worthwhile, is not in any way a substitute for analysis with one of the big boy programs.

3. I've added an on-line lookup (controlled by a compiler flag in the Makefile). The dictionary file with *Scrambler* is not the exact tournament dictionary as that can't be legally distributed. So if the lookup option is active a word not found in the dictionary will be looked up online. The decision of the online source is final. If online lookup fails or is not active, we go back to asking for (honest!) manual resolution. Any new words can be permanently added to the dictionary, if desired, at the end of the game. To use the lookup feature, the `curl` command must be in your command path (very likely the case on most Linux systems and easy to remedy if not).

4. I've increased the number of computer levels (from 6 to 9) and made them more fine-grained. I found even as a relative beginner beating the computer at level 4 out of 6 was relatively common, but level 5 was a whole different thing. So now there are more choices, and I suggest you play against a single opponent, starting at the lower levels and moving up. Some of you will move up to the top quickly, while some of us ... oh, well. Right now I can beat level 6 almost all the time, but level 7 kills me. Not very impressive. My excuse is that as of this writing, I've been playing for less than three months.

5. I've added a "tiles unseen" option. The static display of the initial number of tiles of each letter and the associated point value wasn't helpful once the game began. Now the display shows the count by letter of tiles the human player hasn't seen. These tiles will either be in the bag or on another player's rack.

6. I've eliminated the compressed dictionary option. It was too fragile and didn't work correctly on modern systems. At any rate in 2021 saving a meg of disk storage isn't crucial. At the same time, I increased the maximum dictionary size to 200,000 words. I replaced the dictionary with a more up-to-date, much larger freely available one. This should reduce the need to do on-line lookups, and make for stronger computer play and better analysis.

7. I've added a save and restore feature. How to save and restore game states is described above in the playing instructions.

   Please note that the save/restore situation is not quite clean and clear with respect to GCG and EVAL files. *Scrambler* will try to open the GCG/EVAL files in use at the time of the previous save, and then append to them. If those earlier files don't exist, GCG and EVAL info will be written to new files. So far so good, more or less.

   However if you saved a game, and then continued to play, and then quit, and later restored the save, the GCG and EVAL files will still contain all the moves made after the save, and will start appending the game continuation, which now is reset to an earlier point in the action (the time at which it was saved). This could be considered a feature, in that you'll have a record of game variants. It also could be quite confusing.

   As a redeeming feature, save files are quite tiny.

8. I've added interrupt handlers to try to allow for a graceful exit from bad situations. In some instances the player is offered an opportunity to save the game state after an

interrupt has been caught. Restoring a game from such a save may or likely may not work. However the saved game file might be useful for debugging, so if you plan to report a bug, please hang on to it.

9. I've completely revised and expanded the documentation.

10. I've made numerous other small changes, including some code clean-up and some code mess-up.

11 Bugs and To-Do

1. Testing could always be better. There are surely bugs.

2. Sometimes quitting with **q** has to be done twice. It's hard to reproduce and track down as it isn't frequent.

3. There are some rare segfaults and buffer overflows. These are very hard to track down and fix because they occur seldom.

4. There have been a couple of instances of signal 1 (SIGHUP) when the computer moves first and is playing at level 9. This takes place right after the computer's first move. It's only been observed in an ansi-term on Emacs. The situation is unclear at the moment, and reports are solicited if this is observed.

5. Counting of passed turns at the end of the game is not in accord with tournament practice. To resolve this, the "six zeroes" rule or equivalent should be implemented. That would be easy enough if the game didn't allow multiple computer players. As things stand the game ending mechanics could be better.

6. Not all interrupts are caught, and they might mess up your window to the point that it is no longer usable, or crash the game, or lead to data loss. For instance, if something bad happens (like an interrupt or crash) while *Scrambler* is re-writing the dictionary, the dictionary file might be lost. Beware! Keep a backup.

7. It would be very cool if Scrambler could read GCG files and continue a game from that point. This however would be a big undertaking and I'm unlikely to take it on at any point in the foreseeable future.

8. Improving the computer AI to account for more than just score (see the discussion in a previous section) would be great but way out of scope for this sort of effort. (But at least the AI should be taught to exchange tiles once in a while!) In addition, the way computer skill levels work could be improved. For instance, the computer plays quite a weak game through level 6, and then at level 7 becomes much, much tougher (although objectively still not especially good).

9. Finally, the file naming really needs work. It's hard to remember what GCG or EVAL file corresponds to what playing session, and save and restore makes this a lot worse. There's no easy way right now to associate save files with corresponding GCG and EVAL files, which may or may not have the same numerical prefixes and may or may not contain previous game information. The logfile is of some help but it just all seems kludged together (because it is).

12 License and Disclaimers

The original program appears to be in the public domain, and my modifications are also placed in the public domain. This means you may use them any way you wish but you may not by any means or method claim them for yourself, or restrict access to them in any manner whatsoever.

THERE ARE NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR INTENDED USE. USE ENTIRELY AT YOUR OWN RISK. NO LIABILITIES OR DAMAGES OF ANY TYPE, INCLUDING BUT NOT LIMITED TO DIRECT, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES, WILL BE ACCEPTED UNDER ANY LEGAL THEORY WHATSOEVER, EVEN IF I HAVE BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGES. BE ADVISED THAT LOSS OF DATA OR OTHER HARM IS POSSIBLE. EVALUATE THE RISKS IN YOUR OWN ENVIRONMENT AND DO NOT USE THIS PROGRAM IF YOU ARE NOT WILLING TO ACCEPT THE ASSOCIATED RISKS.

While suggestions and bug reports will be accepted, I don't promise to necessarily do anything or take any action. This is not an offer of support (none is promised or provided), and you agree that anything you send to me is immediately placed into the public domain, without compensation to you in any form, including but not limited to acknowledgement.

Please be warned: since the program is in the public domain, you might have obtained it from virtually anywhere. I have no way of knowing what may have been done with or to it by others. Caution is advised as ALL RISK IS UPON YOU.

13 Changelog

```
2.00 2020-02-01 Refactoring; add GCG output, etc.
2.10 2020-02-03 Add analysis output, etc.
2.20 2020-02-05 Add save/restore, etc.
2.30 2020-02-09 Add signal handling, logfile, LaTeX docs, etc.
2.40 2020-02-11 Change pass logic and game end logic.
2.50 2020-02-12 Add tiles remaining by letter feature.
2.60 2020-02-21 Numerous minor changes; program name change.
```

14 Conclusion

One last time, this is *not* the proprietary game with which you're familiar. There are far too many differences in rules and presentation.

If anyone knows where to reach Dr. Cherry, I'd like to hear from you.

Enjoy *Scrambler* and stay safe and well, wherever you are.

Bob Newell
Honolulu, Hawai‘i
24 February 2021