Chessic Version 1.0 User Manual

Joshua Blinkhorn

September 25, 2021

1 Introduction

Chessic is a command-line chess training application. Its primary use is to train opening preparation; to assist the user in memorisation of prepared opening lines. Chessic uses *spaced repetition* and *active recall* to schedule training positions from your opening repetoire.

1.1 User interface

Chessic uses a command-line user interface, as apposed to a graphical interface. It is designed to be run in a terminal with the chessboard rendered as coloured text. The rendering of the chessboard in a standard terminal is unflattering at best; experimenting with terminal settings (like typeface, font size and line spacing etc.) can significantly improve it (see Figure 1). Users are encouraged to tweak their terminal settings to get the best experience.

1.2 Training method

Spaced repetition with active recall is a memorisation paradigm. Items are repeatedly shown to the user, who must recall some information - in this case, the correct move in a chess position. Based on the difficulty the user experiences in recalling the information, the item is scheduled for recall at a later date.

The learning method consists of two phases: a *learning* phase and a *review* phase. In the learning phase, the user sees the position several times in quick succession. At the end of the learning phase, the position moves into the review phase, and is scheduled for recall the next day. If it is successfully recalled, it is scheduled for recall in the next few days, and so on. The spacing between recalls grows as long as the position's solution is successfully recalled; if the solution is not recalled, the position is considered forgotten and moves back into the learning phase.

Chessic employs *position-based training*; the user learns to recall the correct move in a position, without reference to the opening line in which that position occurs. This method of training has its advocates, but we do not seek to justify it here. Users will often see related positions appearing near one another in the learning phase. This clustering of related positions is merely a feature of training queue generation for new positions. It is retained because it may be helpful in the learning phase; scheduling for recall is deliberately randomised to destroy such clustering.

1.3 Organisation of openings

The user may wish to train, for example, their repertoire for the Najdorf variation of the Sicilian defence playing as white. To do so, the user creates an *item* called 'Najdorf', in a *category* called 'Sicilian Defence', in a *collection* called *White Repertoire*. The collection might

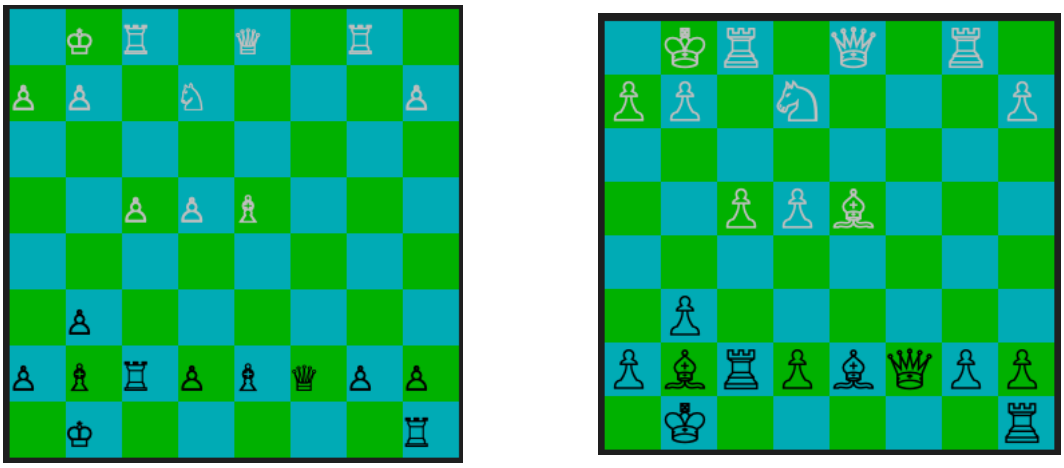

Figure 1: Defaulting terminal rendering of the chessboard, and the result of tweaking some settings.

```
YOUR COLLECTIONS

ID COV. ITEM                WAITING  LEARNED  TOTAL
1    2% Black-Openings      189      41       2421
2   40% White-Openings      15       6        15

[ID] select
'n' new
'd' delete
'b' back

:
```

```
COLLECTION Black-Openings

ID COV. ITEM                WAITING  LEARNED  TOTAL
1    9% Caro-Kann           0        20       221
2    3% English             19       11       439
3    1% Grunfeld            150      10       1522
4    0% Sicilian            20       0        239

[ID] select
'n' new
'd' delete
'b' back

:
```

Figure 2: The main menu and the collection menu. Coverage ('COV.') shows the percentage of training positions that have been learned.

include other categories like 'Ruy Lopez', which in turn may include more items like 'Berlin Defence' or 'Exchange Variation'.

This exemplifies the hierarchy Chessic uses in the organisation of opening variations: collection, category, item. The hierarchy is inteded to be extensible beyond openings. For example, the user might want to create a collection called 'Endgames' containing a category called 'Rook Endgames' with items like 'Lucena Position' and 'Philidor Position'.

2 Operation

2.1 Menus

On launch, Chessic displays the list of collections along with information about them (see Figure 2). From here, the user may navigate into a collection by typing the ID number and hitting enter, or create new collections and delete existing ones. Menus for collections and categories work in a similar way, allowing the user to navigate to the desired item menu for management and training (See Figure 3). Operation of these menus is achieved via command prompts and should be intuitive.

Figure 3: The item menu. From here the user can choose to manage or train a variation.

2.2 Managing variations

Creating new variations

Variations are created by selecting the option 'new' in the category menu. From there, the user is prompted to select a colour (the colour of pieces the training player moves) and a starting position. The user navigates to the starting position by entering moves until the desired position is reached. Instructions on how to enter moves are given in the following section.

Editing variations

Initially a variation is empty; it contains only the starting position. To develop a variation, select the 'manage' option from the item menu, which launches the management dialogue (see Figure 4).

The management dialogue displays a position from the variation, initially the starting position. Moves are added by typing a move in *standard algebraic notation*[1]; for example 'Ne4', 'Be3', 'Kf1', 'Qd5', '0-0', 'g3', 'exd4', 'e8=Q' etc. Existing moves are visited by entering the ID as provided in the accompanying list.

Any listed move is either a *problem* or a *solution*. Problems are moves leading to positions in which the training player is to move; solutions are those moves which are playable in the context of a problem.

A problem may have more than one solution, in which case *the trained solution is the first one*. This allows the user to experiment with and record various responses in their opening repertoire, while actively training only the chosen one. To switch between multiple solutions, use the 'promote' option in the management dialogue to bring the chosen solution to the top of the list.

2.3 Training variations

When training for a variation is available, the 'train' option appears on the item menu. This means that there are training problems ready to be solved for that variation. These are new positions beginning in the learning phase, previously learned positions in the review phase, or typically a combination of both.

[1] Universal Chess Interface notation may also be used.

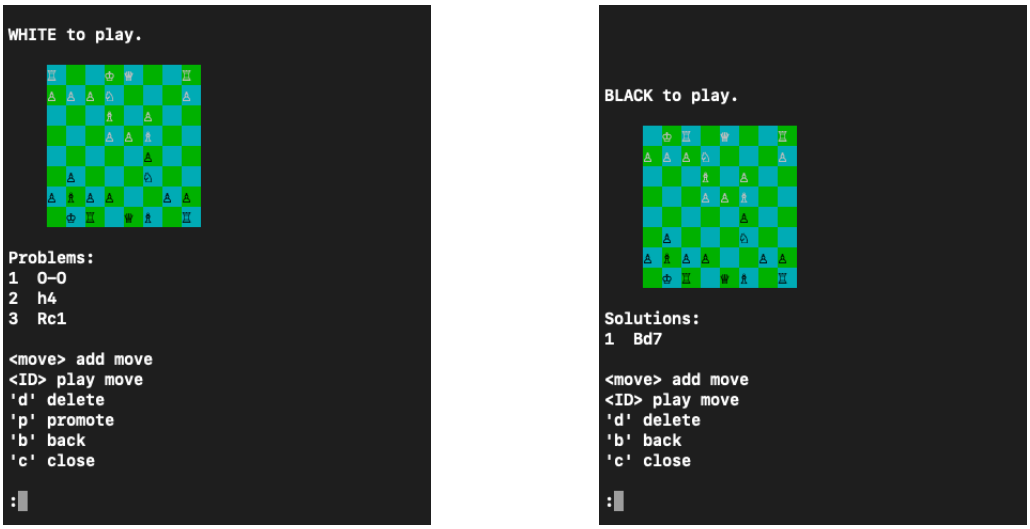

Figure 4: The management dialogue, displaying positions from a variation with a list of problems (left) and solutions (right).

Selecting 'train' launches a training session, in which the user is presented with a series of problem positions treated like flashcards. The user looks at the problem for as long as they wish and, after hitting 'enter', is presented with the solution (see Figure 5).

After viewing the solution, the user is prompted to choose one of three options - 'easy', 'okay' or 'hard' (unless the position is brand new, in which case there are no options). These options determine the passage of the position through the learning and recall phases. If the user determines the solution correctly, they should select 'okay', and if the move is obvious to them (i.e. a forced recapture) select 'easy'. Otherwise the user should select 'hard'. Brand new positions require no such evaluation because they serve to present the user with the solution, rather than prompting them to recall it, which is the case in all other scenarios.

Scheduling

Over time, positions that are successfully recalled are scheduled further into the future, whereas failed problems are scheduled sooner. In fact, Chessic employs *strict relearning*, where previously learned position marked 'hard' go back *immediately* into the learning phase.

At most ten new positions per variation are scheduled each day, but they do not accumulate; new positions that remain unseen are rescheduled for the next day as part of that day's quota.

However, the rescheduling of learned positions in the review phase means that the training volume changes over time, and grows quite quickly over the first few days. At its peak, the volume of positions waiting for a single variation is typically around forty, but it can be more if many recently learned are positions are recalled incorrectly (i.e. short-term forgotting). If this is the case, you might be trying to learn too many variations at once, and want to consider reducing the number of variations you train simultaneously.

For a fully learned variation of reasonable size (around 150 positions), the review volume will eventually drop to a few positions per day, allowing the user to review that variation on a leisurely basis.

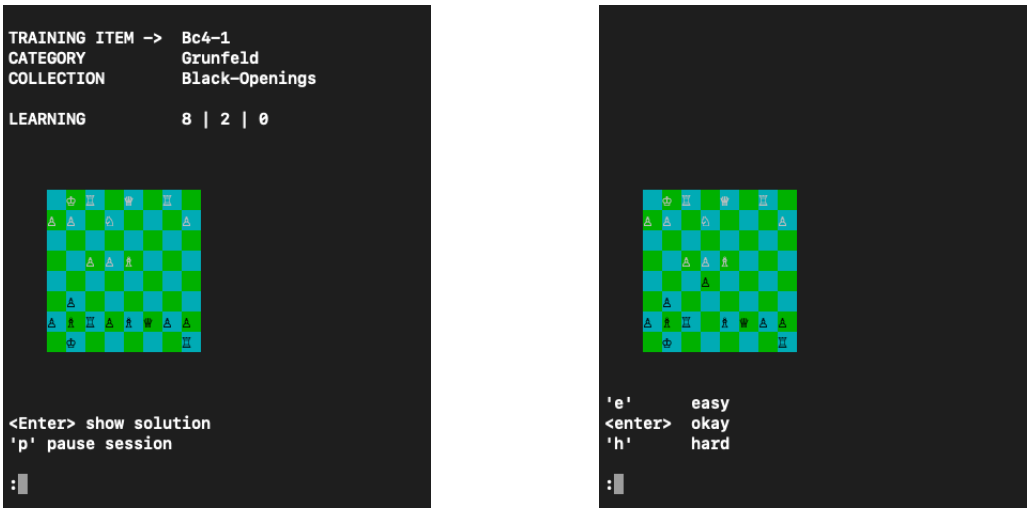

Figure 5: The training process. A problem is posed to the user (left), after which its solution, the move 'e5', is revealed (right) and the user is prompted to evaluate their success in recalling it.

2.4 Importing from PGN

Version 1.0 of Chessic includes a python script (convert-pgn.py) that generates a Chessic variation from a PGN file. This allows the user to build their repertoire in a graphical PGN editor, rather than the in the Chessic command-line interface, if they so desire.

Chessic expects certain guarantees for the format of such a PGN:

- The PGN starts at the starting position of the variation. Typically, this is *not* the standard starting position, but most PGN editors allow a non-standard starting position to be specified.
- In positions in which the training player is to move, and variations exist, the move to be trained (i.e. the correct solution) is the main line move. Therefore users should exercise care if they wish to import their existing repertoire in PGN format; if the repertoire contains notes on inferior moves for the training player, those lines should be variations, not main line moves.

Collections are stored in the directory 'Chessic/Collections'. For further instructions on how to use the script 'convert-pgn', see 'Chessic/README.md'.