The Little Program That Could

A Minimalist Chess Program

Bob Newell

Based on *Faile* by Adrien Regimbald

Version 2.0.0
22 July 2007

Contents

1 Introduction

Faile is an excellent chess program written by Adrien Regimbald (`adrien@ee.ualberta.ca`). The original program version, including links to all source code, can be found on Adrien's web site, `http://www.ee.ualberta.ca/~adrien`. *Faile* is estimated to be at expert to master strength (perhaps as good as 2200 ELO). *Faile* will make use of an opening book, if present.

I have taken *Faile* as a base and made fairly extensive modifications and additions to the original. I have used the permissions of the MIT License to reissue the program. However, for ethical reasons I have changed the name, as I do not wish to imply that the current effort is a sanctioned continuation of the *Faile* line, nor do I wish to have Adrien's name associated with bugs and problems I've surely introduced along the way.

I have renamed the program *Little*, short for *The Little Program That Could*. Adrien's original *Faile* is a minimalist effort that does an extraordinary thing; it plays chess at the master level. The compressed binary executable file is around 30k for Linux and only about 60k for Windows.

This distribution of *Little* is binary-only; however, source code will be provided free upon email request to `bobnewell@bobnewell.net`.

Executables for Linux and Windows are provided in the main distribution. Macintosh status is unknown as I don't have access to a Mac. A DOS binary is available; see the section below.

Windows testing was on Windows XP. No testing has been done on Windows Vista. Please let me know your results if you try it.

It's no secret that this distribution is aimed mostly at Linux users; after all, my purpose in working on *Little* was to have a small but useful chess program for my Linux platforms.

Little is distributed under the MIT license, as described near the end of this manual.

2 Installation

Whether you use the Linux or Windows binary, installation begins in the same manner. Create a directory to hold the files and unzip the zip archive into that directory. Since you are reading this you may likely have already done so. Additional information is found below.

3 Starting *Little*

This version of *Little*, unlike Adrien's original distribution, is recommended for command line use, though use in a graphical environment with WinBoard (Windows) or XBoard (Linux) is of course possible.

3.1 Running Under Linux

Use the `unzip` command to unpack the archive. Then just start the program, from its own directory, with the command line command `little`.

3.2 Running Under Windows

Unpack with WinZip or whatever utility you use to unpack `zip` archives.

Start the program either from the command line, if you have one in your release of Windows, with the command `little` or else from Windows Explorer by double-clicking on the file `little.exe`. (The `.exe` extension may not show up as Windows likes to hide essential detail from the user; like tow trucks, making your life harder at your expense.) The double-click method won't allow you to easily give command line options. I can't help it if Windows is limited in capability and difficult to use.

3.3 Running On A Macintosh

Unknown! *Little* can most probably be made to work. Please write if you have success and give details for me to include in the manual.

3.4 Running Under DOS

If you really want to, you can run under DOS. A DOS executable built with `djgpp` can be obtained here:

`http://www.bobnewell.net/filez/little/littled.zip` .

Be sure to run on a computer with enough memory; even so, building opening books is dubious at best. To run simply use the file `little.bat`. If you want to use an opening book, copy the book file into your `little` directory.

The DOS version is provided for those true minimalists who want to run from a boot floppy or the like. The DOS version is not likely to be the latest or the most debugged, and all features aren't tested.

4 Using *Little*

4.1 Graphical Interfaces

This version of *Little* defaults to command line mode, but it is possible to use it with a graphical interface.

The simplest though least aesthetic option is to give the command `diagram` and then the program will print a primitive ASCII board diagram after each command or move. That isn't exactly a graphical interface, but it works.

4.1.1 Windows

If you have a recent version of WinBoard (get it on the internet), you can interface *Little*. You will need to use the Windows command line or a shortcut that executes a command like this:

```
c:\winboard\winboard.exe -fd "c:\little" -fcp "little.exe"
```

which assumes Winboard is found in the `c:\winboard` directory and *Little* is found in the `c:\little` directory; modify as needed.

Not all Winboard commands are supported and the interface *does* have bugs; trying to do analysis, edit the position, switch computer sides mid-game, or anything other than straight play will likely cause *Little* to abort or hang.

4.1.2 Linux

Under Linux, the program will interface with XBoard (your Linux distribution will surely have XBoard as an optional install). If XBoard is in your command path, and if you've installed *Little* in your home directory in the `little` subdirectory, this command should work:

```
xboard -fd "$HOME/little" -fcp "./little"
```

Not all XBoard commands are supported and the interface *does* have bugs; trying to do analysis, edit the position, switch computer sides mid-game, or anything other than straight play will likely cause *Little* to abort or hang. Startup is also a bit slow.

4.1.3 Internet Computer Server Interface

Little will work with XBoard or WinBoard as an on-line computer player on an internet chess server such as FICS or ICC. The details of registering a computer account and using *Little* on-line are beyond the scope of this manual. If you wish to do this, consult the on-line server documentation and obtain the *Little* source code from `bobnewell@bobnewell.net`. The source distribution contains a script for on-line test use. *Note:* I have run trials of *Little* on FICS, and it plays lightning and blitz at *least* at the 2100 rating level.

4.2 Non-Graphical Interface

Little run in non-graphical interactive mode is excellent "blindfold chess" practice, but of course you can print ASCII boards as described elsewhere in this manual.

4.2.1 Making A Move

A move is entered at the `Little>` prompt.

Little will recognize almost any reasonable variant of SAN or coordinate notation, for instance `e4`, `e2e4`, `Nf6`, etc. Piece designations should be in upper case and must be in English language format. If the program unexpectedly rejects a move as illegal, check for disambiguation; e.g., you entered `Ne2`, but two knights can move to `e2` and you didn't specify which one, such as `Nge2`.

Notation references and explanations are easily found on the internet.

If you want to give up, use the `resign` command. The computer never resigns! You can offer a draw with the `draw` command. The computer will decline prior to 20 moves being made on each side, and, with present logic, will not accept unless trailing badly.

4.2.2 Time Controls

Time controls are always enforced for the computer. They are optional for the non-computer (human) player; see the `flag` command options below. When human player time controls are active, the number of seconds left is always shown before the `Little>` prompt.

If the time control is move-based and not sudden death, it will simply repeat indefinitely. There is currently no provision for multiple, differing time controls.

Note that time always runs, no matter what commands are entered or what activities are taking place.

Time controls are not *strictly* according to the usual rules. If a player is flagged, the player loses, unless the other side has insufficient mating material. In that case the program recognizes the result (correctly) as a draw. But there is no recognition of the idea of insufficient losing chances; if you flag in a position that is an obvious draw or even a clear win, you still lose. (If this really bothers you, turn off `flag` mode in such situations.)

4.2.3 Saving and Loading Positions

As detailed below, the `save` and `restore` commands will save your current game, in complete detail, to a file; and conversely, restore the saved game state from a file. This has many uses, but the primary usage is simply to save a game to resume later. It is also helpful in exploring variant moves from a given starting position.

Save files from versions 1.5.1 and earlier are not compatible with later versions of the program.

4.2.4 Move Lists

The interactive command `list` will list all moves thus far to the screen, in coordinate notation, and will output a PGN file with correct (though somewhat arbitrary) game headers

and all moves, again in coordinate notation. The PGN output is not, strictly speaking, according to standard (the standard requires SAN rather than coordinate move notation, for instance), but I have found that nearly all programs will read it.

You are prompted for a file name. If the specified file already exists, the PGN is silently appended to it, allowing the buildup of a game collection.

4.2.5 FEN Output

The interactive command `fenout` will output a FEN string to the screen, and to a file. The FEN appears to be standard and accurate (please let me know of any errors).

You are prompted for a file name. If the specified file already exists, the FEN string is silently appended to it, allowing the buildup of a position colleciton.

4.2.6 Command Files

This is an extremely useful feature; the command `read` will allow reading in a text file containing *Little* interactive commands, one per line (including responses to potential prompts, each on its own line).

For instance, the following command file sample runs up a French Advance position with Black to play.

```
# set up French Advance
two
e4
e6
d4
d5
e5
one
```

The first line is a comment. Any line starting with the `#` character is not processed.

The line `two` command enters two player mode. The next lines input moves to set up the desired position. The line `one` sets play back to "human vs. computer" mode, with the human player to move as Black. You could add a `go` command to make the computer move as Black, the human player then playing the next move as White.

You might wish to build a series of opening positions in command files, and use the `read` option to start out in a particular opening line of interest.

Command files may be "stacked" meaning that one command file may call another by embedding a `read` command and a file name in the calling (first) file. When the second file ends control goes back to the first file. There is a current stacking depth limit of eight files.

A small selection of sample command files is included in this distribution; they can be identified by their `.rd` suffix.

5 Command Line Options

The following options are available at the operating system command line. The general command format is

`little -option value -option value` ... etc.

-hash (value) By default, *Little* uses about 8MB for hash tables. *Little* will, in general, play better with more hash space. However, you must be careful, as if you specify more hash space than your machine can handle, *Little* will be using virtual memory, accessed from your hard drive, and performance may suffer severely. The hash size is specified in MB, and it is usually more than good enough to use the default value. Hash values are approximate; the program will allocate hash space in available increments.

-mbook (input file) (plies) (hash size) See the section below, *Opening Book*. This option is for making new opening books.

-book (filename) Use `filename` instead of `little.obk` as the opening book.

-flag Set the startup default so that the non-computer player will be flagged on a timeout (the normal case is no flagging).

-read (input file) On startup, read commands immediately from the specified input file. See the section above on *Command Files*.

-restore (input file) On startup, immediately restore the saved game contained in the specified input file. See the section above on *Saving and Loading Positions*.

-tc (moves to tc) (game time) (increment) Set the default game time parameters. All three must be present. For instance, to set up a game with 40 moves in 120 minutes with a 5 second increment, enter `-tc 40 120 5`; this control will repeat every 40 moves (there are no secondary controls). For a blitz game of 2 minutes with a 12 second increment, enter `-tc 0 2 12`. Note that a moves to time control value of zero implies the complete game. A typical blitz game of 5 minutes, no increment, is given as `-tc 0 5 0`. These values are retained from game to game within a session. The default is game in 10 minutes, no increment.

6 Interactive Commands

In command-prompt mode, the following commands are valid at the `Little>` prompt. Time continues to count down while executing these commands!

help

? Print a simple description of interactive commands to the screen.

exit

quit Leave *Little*. There is no prompt to save your game so take care.

diagram

d Toggle diagram mode from "on" to "off" or vice-versa. This controls the printing of primitive ASCII board diagrams to the screen. The default at startup is "on."

list Show an ASCII board diagram, just this once, without changing the diagram mode.

new Start a new game. There is no prompt to save an existing game so take care.

book Toggle opening book usage from "on" to "off" or vice-versa. The default at startup is "on."

resign Resign the game. This is a human-only option, as the computer never resigns!

draw Offer a draw. This is a human-only option, as the computer never offers a draw!

pgnout Output the list of moves to the screen and full PGN to a file. You are prompted for a file name. PGN headers are rather arbitrary.

fenout Output FEN notation for the current position to the sceen and to a file. You are prompted for a file name.

one Go (back) to human vs. computer, or one-player mode. The program will assume that it's the human's turn to move. Use the `go` command to change this.

two Enter two-player (no computer) mode, in which you enter moves for both sides. (It would be best to have `flag` mode turned off, but it's up to you.)

go Tell the computer to move from the current position. Turns off two-player mode, if in effect; the "human" player will make the subsequent move. If *Little* is already in human vs. computer move, this effectively switches sides.

flag Toggle the flag status (for flagging the non-computer player on timeout) from "off" to "on" or vice-versa. The default is "off."

save Save everything about the current game to a file. You are prompted for a file name. An existing file is silently overwritten, so take care.

restore Restore a saved game. You are prompted for a file name. This completely resets every *Little* variable to the values in the save file, overwriting all current values. There is no warning so take care.

read Read commands from a file. You are prompted for a file name. See section below for details.

post Toggles display of computer thinking from "on" to "off" or vice-versa. Post is always forced on in xboard mode.

tc (m) (t) (i) Set time control, where (m) is the number of moves to the time control (zero if using an internet chess server style time control); (t) is the number of minutes per game; and (i) is the time increment in seconds. These values become the defaults for the rest of the session. The current clock times are reset.

7 Opening Book

7.1 Book Defaults

This distribution provides a relatively small but quite useful opening book. If you wish, you can build your own, which can be any size (large or small) within the limits of the program and available processing power and computer physical memory.

Little will by default use the file `little.obk`, if found. You can specify an alternative file with the `-book (filename)` option described above.

7.2 Book Move Selection

Little adds an element of randomization to the selection of a book move; this method has been completely changed as of release 1.5. Book moves are chosen in proportion to their original frequency of play. In other words, after say `1. e4`, if `1. ... c5` was played 54% of the time in the PGN file used to build the book, *Little* will likewise choose this move on average 54% of the time.

To keep things even more interesting, the program will randomly decide to go "out of book" about one move in fifty, and calculate its move rather than using the opening book. This perhaps reduces the playing strength by a very small amount.

7.3 Making Your Own Book

7.3.1 Preparing the PGN Input File

To make your own opening book, assemble a "clean" PGN file of all of the games you desire to use in creating the book. The PGN file cannot have comments, variations, NAGs, etc.; just headers and moves. (You might wish to use the externally available `pgn-extract` program to clean up the file:

```
pgn-extract -C -N -V -s -o(outputfile) (inputfile)
```

works quite well.)

7.3.2 Generating the Book

Little can now generate the book:

```
little -mbook (PGN input file) [(max ply) (book hash size)]
```

The max ply is optional, and defaults to 20, with a maximum of 40. Book hash size, stated in MB, is also optional and defaults to 32. Book size increases slightly with increased hash size, but the maximum number of positions that the book can handle goes up. Extremely large hash sizes result in very slow program operation.

In any event, the file `little.obk` will be created. (Be sure to rename any existing book file as it will be silently overwritten otherwise.)

Here is an example of creating a 30 ply book from the file games.pgn, with a hash size of 64MB:

```
little -mbook games.pgn 30 64
```

Note that books built with versions of *Little* numbered 1.5 and earlier will *not* work with versions of *Little* numbered 2.0 and higher. Books are binary compatible between Linux and Windows.

7.4 Availability of Other Books

The book supplied with the distribution was made from a collection of just under 13,000 classic chess games, and it is good enough for casual play. But a few other opening books are also available.

There is a medium size book made from just under 50,000 games, representing the collection of the International Correspondence Chess Federation. This is a nice middle of the road alternative, with more play than found in the small default book. It can be downloaded from my web site:

`http://www.bobnewell.net/filez/little/iccf.zip` .

If you want a really large book, though, there are some. One, the "enormous" book, is based on nearly one and a half million games, and another, the "mega" book, is based on over three and a half million games. Oddly, or perhaps not so oddly, the mega book's compiled size is about the same as the enormous book, as single-occurence positions are not saved.

The `zip` archive for the enormous book is nearly 40MB and the mega book is about the same size. You can get these for free upon request; write `bobnewell@bobnewell.net` for access information. (Direct downloads are not provided. These books are not really practical for a minimalist program such as *Little*, and there is no use expending large amounts of bandwidth unless there is real interest.)

A "giga" book based on nearly fifteen million games is being contemplated in my less rational moments.

Unzip your chosen file in your `little` directory; it will silenly overwrite whatever opening book is present.

Finally, I have built a couple of small, specialized books dedicated to particular unusual openings. See the various `.rd` files for more information.

8 Possible Future Extensions

The following ideas are in at least the thinking process, although there is certainly no guarantee as to when these features might appear (if ever, and you know what that means).

- PGN input (high difficulty, moderate importance).
- FEN input (moderate difficulty, low to moderate importance).
- More compact save files (moderate difficulty, low importance).
- Improved draw logic and computer resignation (moderate difficulty, moderate importance).
- SAN notation on PGN output (high difficulty, high importance).

Ideas and bug reports are of course solicited, write `bobnewell@bobnewell.net`. My additions and changes to the original program are guaranteed to be insufficiently tested, and many bugs are going to be found in the field. (Before you complain about this, think about the dominant desktop software vendor and how they release things; and their budget and staff is somewhat larger than mine.)

The following items will probably *not* be done.

- Improvements to the move evaluation method. Adrien has done a great job and I would in all likelihood just make things worse, not better.
- Endgame tablebases. This is just too much for a small program. Yes, it would add considerable strength. But this isn't *Crafty* nor is it supposed to be, and we already have super-sized opening books as an option.

9 Acknowledgement

All the credit for the program's excellence goes to Adrien Regimbald. His concept was brilliant and his execution equally so.

I believe I have added some useful extensions. But it remains the case that 99.9% of the real content in *Little* is Adrien's, and 0.1% or less is my own.

Any bugs and problems introduced by my tinkering are entirely my own responsibility, and that is the only thing for which I will lay claim to full credit.

10 Technical Notes

10.1 Opening Books

At version 2.0 of *Little* the book generation method had a major overhaul. For rapid lookup of book positions, a hashing technique was used in the original releases of *Faile*. Hash collisions, apparently in the interests of simplicity and speed, were resolved in a simple manner which sometimes failed. A position which resulted in failed hash resolution was merely discarded.

For small opening books this had only a minor effect, and the default opening book worked just fine. However, attempts to build larger books resulted in more and more positions being discarded, and ultimately, the quality of the book was severely degraded.

The new book generation method developed in *Little* will never have an unresolved hash conflict and will never discard a position that appears more than once. (Both the old and the new methods don't retain single-occurrence positions.) But this comes at a large price. The amount of memory required to build a book, especially a sizeable book, is much higher than it was previously. Processing time has also gone up. The default *Little* opening book still builds easily in 32MB of book hash space, and builds quickly enough, but requirements rise rapidly. For instance, the *Crafty* "enormous" book, as it is called, contains just under 1.5 million games and, at 40 plies, just over 58 million positions. It would build previously in 256MB of book hash space, taking perhaps 15 minutes on an older 750 MHz machine. But about 22 million positions were discarded due to hash collisions, and the final book was virtually unusable.

Now, the "enormous" book needs 512MB of book hash space in order to build, and will build in half an hour on a 2.4GHz machine with 1GB of memory. There are no hash collisions and the final book is of excellent quality, though perhaps more than a bit of overkill for a minimalist program such as *Little*. But be forewarned; trying to build a very large book on a machine with limited memory is an exercise in frustration. Virtual memory will come into play and book building time will extend into hours or days.

10.2 Random Numbers and Hashing

The original *Faile* release started random number generation from a fixed state, so that randomly chosen opening moves would repeat between runs of the program. This was of course fixed, but it lead to an investigation of the hashing technique.

Both the old and new releases compute 32-bit hash values to represent the "state" of the chess position. It is important that these hashes remain consistent between invocations of the program, and even across platforms. Hashes also need to be unique for each position; a hash collision would be a disaster.

Adrien did this in a clever way by making use of a *second* random number generator that was predictable. I did additional testing and found that all the needs mentioned above, including repeatability and uniqueness of hashes, are met. Compatibility across Windows and Linux is present; other systems, of course, are untested.

11 License

12 Change Log From Version 1.5 Forward

Revision sequences by Bob Newell, Santa Fe, New Mexico commence with version 1.5.0, which was built on the base code of Adrien's seemingly final 1.4 version.

1.5.0

- Started a new release (1.5.x).
- Made modifications for easier blindfold play. This meant setting defaults to not show the board and not show computer thinking. Also removed some other print-

outs and put them under control of the 'post' option. Distracting (from blindfold play) printouts are now quite minimal by default.

- Changed input to arbitrary streams instead of hardwired to stdin, to allow for the 'read' command below.
- Introduced facility to 'read' commands and input from a file. Fixed up the input functions for this. No longer echo Little> prompt during file input, however output all moves made from the file so that the user can track the position.
- Strip leading/trailing blanks from commands.
- Made '?' a synonym for 'help' and moved help to a separate subroutine.
- Implemented a game save/restore feature with commands of the same name. Somewhat experimental, but seems to work. Save files are large (around 32k); probably saving too much stuff but hard to tell what's what at the moment.
- Added command line options -read (file) and -restore (file) for startup reading of command file, save file, or both(!). Modified startup code to make this work.
- Now saving the move list, which can be printed with the new 'list' command. This gives an on-screen move list, and outputs PGN to a user-named file. The PGN is not strictly legit but nearly all other programs can read it.
- Added 'book' command to toggle on/off opening book.
- # in column 1 of input now indicates a comment, useful in command files.
- Some small bug fixes and input parameter checks.
- Random number generator was being reset to a fixed value, causing the same book moves to be repeatedly chosen; fixed.
- Took out testing routines from rand.c.
- Added a few informative printouts so commands won't be so silent.
- Fixed the book routines to not use a short integer count cap, which limited counted position occurences to about 65,000. There was no reason to keep this limit as GCC and its ports never use 2-byte integers.
- Added command line option -book (file) to change the default opening book at startup.
- Wrote a LaTeX manual.
- Packaged the 1.5 distribution.

1.5.1

- Fixed bug in board display calls to display the view of the side to move, not always the human player regardless of situation and modes.
- Added '-debug' command-line option.
- Quickly re-released!

2.0.0

- Not *really* version 2.0 of *Faile* but rather the first version of *Little*. Going to a new name for ethical reasons but keeping the version numbering to show the heritage and debt to Adrien.

- Added 'show' command for one-time only printout of ASCII board.
- Documentation tweaks: noted possible need for cygwin1.dll for Windows, etc.
- Reordered 'help' output slightly to be in more logical groups. Updated help to reflect the new release.
- Rationalized book compatibility by causing incompatibility! Previous books will no longer work. Books now store version number and more importantly, book hash size, so a book written with any hash size can now be read without rebuilding the program.
- Major changes to avoid positions being discarded due to unresolved hash collisions. This meant a complete rewrite of the way the book is built. Memory and processor requirements go way up but so does book quality especially for large books. Fixed self-inflicted bugs along the way.
- Verified consistency of simple_rand in different program runs and across Windows and Linux platforms; other machines unknown.
- Reduced save file size by about half, and it now compresses very well.
- Fixed bug where computer thinks it has to move instantly in a sudden death time control.
- Added -tc option to allow for session-persistent time control changes; these are saved/restored in a save file also.
- Fixed PGN tag defects. Tags are still arbitrary but at least meet the standard. However, correct date and result appear now, and there are no longer missing tags. A FEN tag is included.
- Allow the human player to resign.
- Provide for FEN output to screen and to a file; finally got the castling part right, and this lead to correcting a small problem with save files.
- Fixed a self-inflicted problem with the xboard interface aborting Little on human resignation. Some people hate to lose, I guess. Also a few other xboard tweaks, though this mode still has issues if the user tries advanced xboard commands.
- Took out move evaluation debug code. I'm not going to change the move selection process much if at all.
- Now track user time remaining when not in xboard mode (which gives this to us via otim); also show time remaining at command prompt. This allowed activation of Little's time allocation logic in non-xboard mode. Enforce user timeouts too! But ... it's optional, and off by default, under control of the command line -flag option or the interactive flag command. This seemingly simple feature required extensive coding and debugging.
- Fixed move-based time control to actually be enforced and to repeat, as we don't provide secondary time controls.

- Made 'force' an internal command and added the more intuitive 'one' (player) and 'two' (players) interactive commands.
- Added rudimentary logic for agreed draws. Force a draw, whether requested or not, when both sides have insufficient mating material. Recognize a draw when one player flags but the other has insufficient mating material.
- Fixed slow startup on XBoard by sending feature command.
- Worked on ICS mode and got it automated and tested. Little plays really well online!
- The manual reflects the many changes for this release.
- Ready for release, let the bugs roll!
- Post-release manual modification; the enormous book is no longer available for instant download due to size and bandwidth usage, but is still available free upon email request.
- Rebuilt little.exe with MinGW to eliminate need for the cygwin dll; no version number change; updated docs to reflect this.
- A "mega" book is now available on request.