Lisp in Summer Projects Submission

Submission Date	2013-10-02 15:08:16
Full Name	Michael Vollmer
Country	US
Project Name	Lalla
Type of software	command-line/terminal app
General category	game
LISP dialect	Commmon Lisp
GitHub URL	https://github.com/probabilityZero/lalla-chess
Did you start this project?	Yes, all the code is written by me
Project Description	I want to describe my project in this form.
Purpose	The aim of this project was to implement a chess engine in portable Common Lisp, learning the language along the way.
Function	The program can play chess (i.e. choose semi-intelligent chess moves) using the minimax algorithm.
Motivation	I wanted an excuse to learn Common Lisp and develop a real project with it. I have done research in computer chess in school, and I am very familiar with Scheme, but I wanted to learn CL.
Audience	Eventually I plan to make it compatible with winboard/xboard, in which case the audience is any computer user wanting to play chess. Currently, it's probably only of interest to other programmers.
Methodology	Lalla uses an 0x88 board representation, and it makes heavy use of Common Lisp's type system. Using the Defstar library, nearly every variable and parameter has a declared type, and the type declarations are as tight as possible (often using mod for positive integers). It stores moves and piece information as bytes in unsigned integers and relies on an optimizing compiler like SBCL to optimize these operations.

Currently, Lalla uses a typical negamax search procedure with alpha-beta pruning and iterative deepening, and it uses piece-square tables and material for static evaluation.

I only ended up having about a month to work on this project, so it's narrower in scope than I was hoping. In the git repo history there's a half-implemented bitboard implementation and some skeleton code for xboard/winboard integration --- I'll resume work on them in the future.

The overall approach to the design of this program was procedural, following the typical implementation strategy for minimax (a recursive search procedure "makes" and "un-makes" moves on a mutable global board), and bottom-up. To start, I made simple interfaces to each of the major components of the interface (pieces, moves, the board) using short functions, and implemented the higher level behavior (move searching, evaluation) using these simple functions as a high-level language.

Conclusion

I had high expectations for CL and SBCL going into this project, but I was still blown away. SBCL consistently produced excellent machine code. I implemented some of the functions in C for comparison, and SBCL produced code on par with and occasionally faster than GCC with it's default optimization level.

Furthermore, I was initially worried about attempting this project in a langauge without a static type system, but CL's type system proved to be extremely useful. In many ways, I found it to be a "stronger" type system than many statically typed languages. In particular, the ability to have fine-grain type declarations like (mod x) and (unsigned-byte x) was very useful. Defstar made the syntax convenient, as well.

I didn't have the time I was expecting to work on Lalla (I ended up doing a summer internship, and I'm working on my grad school applications now during the semester), so I didn't finish the most important feature: xboard/winboard integration. That's the next major step, not counting minor bug fixes and some missing core features (checkmate detection, for example).

Build Instructions

Assuming you have quicklisp installed, clone the Github repository, open a REPL in the directory, and run (ql:quickload "lalla").

Test Instructions

Typing lalla::board should print out a vector of numbers. That's the 0x88 board.

Execution Instructions

Currently the user interface is incomplete, but a game of chess can be played interactively in the REPL.

(lalla::make-move (lalla::parse-string "a2a3") 0) causes white (0) to make the move "a2a3".

(lalla::move->string (lalla::depth-limited-search 0 4)) does a minimax search for white (0) to a depth of 3, and prints out the move as a string.

Describe any bugs or caveats	Beyond the obvious lack of a user interface, there currently is no support for castling or check detection, and there are various small bugs.
Official	I have read rules and have abided by them. I am 18 years of age or older. I am not living in Brazil, Quebec, Saudi Arabia, Cuba, Iran, Myanmar (Burma), North Korea, Sudan, or Syria.