

# User manual



# Glass Chess Engine

**by Edmund Moshhammer and Pawel Koziol**

revision 011, 26.11.2011

# Table of contents

Introduction	3
Authors	4
History	5
Issues	8
Features	10
Options	12
Personalities	14
Style	15
Weakening	16
Your own opponent	18
Predefined styles	19
Evaluation	21
Thank You	24

# Introduction

Glass is an UCI-compatible chess program. To get most out of it You should use it under a GUI (Graphic User Interface) such as Fritz GUI, TarraschGui or Arena. Its authors are Edmund Moshhammer (who wrote most of low-level routines) and Pawel Koziol (mostly eval function). More information can be found on a web page <http://www.marittima.pl/glass> Please visit it from time to time, since Glass is a young engine, so updates and bugfixes are likely.

Glass 1.8 is free for private, non-commercial use by the individuals. As for commercial use, there is a possibility of licensing. As an individual user You are entitled to redistributing this package to friends, as long as:

- ◆ You leave the content of this package unchanged, possibly updating the opening book and adding new personalities, which should have distinct names
- ◆ You do not alter the engine in any way
- ◆ You do not make people pay for it
- ◆ You do not enter it in the tournaments where author's permission is needed

Glass 1.8 is provided "as is", with no warranty whatsoever. We expect it to play a game of chess, though.

**Edmund Moshhammer**—born in 1989, lives in St. Gallen, Switzerland. He is currently studying economics. Edmund is an hobby chess player (ca. 1600 Elo) and became interested in chessprogramming around the year 2006. His first efforts in the field he devoted to Endgametablebase generation, afterwards working on the chess engines CPW-Engine and Glass. In the last two projects he was responsible for the low-level programming.

**Pawel Koziol**—born in 1979, lives in Warsaw, Poland. Ph.D. specializing in the mediaeval chronicles, amateur programmer and semi-professional webpage designer. Enjoys mind games, being 6 kyu at Go and a 1800+ chess player. Responsible for the chess knowledge (or lack thereof) in Glass chess engine.

The two met together on the Chessprogramming Wiki and started coding a didactic chess engine called The CPW-Engine. Since this proved a lot of fun, they decided to start working on Glass. The first version has been just a bitboard rewrite of CPW-Engine. The one that generated some interest was Glass 1.6 with its personality settings.

## **Glass 1.8**

26.11.2011

- ♦ search rewritten in a non-recursive manner
- ♦ tuning of LMR and futility parameters
- ♦ eval cleanup
- ♦ bugfixes (among them important see bug, causing crashes)

## **Glass 1.7**

20.07.2011

This is an interim release, done mainly to show that the project is still alive.

- ♦ bugfixes (losses on time, KBB checkmate, LMR, asymmetric values, opening book inactive in n moves per second mode)
- ♦ general code cleanup
- ♦ x64 compatibility
- ♦ simplified quiescence move generator for serious speed gain
- ♦ improved move ordering
- ♦ massive change of piece/square tables and mobility values
- ♦ some special case code for positions with low material

## **Glass 1.6**

13.07.2010

1.6 was a release concentrated on new features rather than on strength. Still, changes to LMR and re-tuning some parameters should give a significant boost.

- ♦ CCRL: 2523 Elo
- ♦ more parameter tuning, especially reducing aggression
- ♦ history heuristic applied to late move reduction
- ♦ enhanced transposition cutoff bugfix
- ♦ getting rid of evaluation cache
- ♦ defining engine strength (600-2400 Elo)

## **Glass 1.5**

29.03.2010

Version 1.5, started after a longish break, has been a major step forward in terms of code quality, and self—play tests showed a significant improvement of over 50 Elo. Unfortunately it didn't prove true against other engines, chiefly because new eval weights were obsessed with King safety and late move reduction too drastic.

- ♦ CCRL: 2441 Elo
- ♦ many bugfixes and a lot of tuning
- ♦ faster, pseudo-legal move generation
- ♦ late move reduction
- ♦ more aggressive, generalized futility pruning
- ♦ material imbalance evaluation
- ♦ king safety completely rewritten
- ♦ Winboard

## **Glass 1.4**

--.--.--

Two independent failed attempts by both developers

## **Glass 1.3**

30.06.2009

- ♦ CCRL: 2451 Elo
- ♦ Nalimov Endgame Tablebase Support (3-6 men)
- ♦ Nullmove tuning
- ♦ No separate settings anymore for the different types of transposition tables
- ♦ Improved time management (time control of 10s / game tested successfully)
- ♦ "analyze" command added to analyze epd files
- ♦ pawn storm code and minor eval tweaks
- ♦ opening book tweaks

## **Glass 1.2**

03.05.2009

- ♦ CCRL: 2376 Elo
- ♦ multiPV implemented
- ♦ fixed serious king safety bug (resulting in a much more aggressive play)
- ♦ speed optimizations by changing variable types

## **Glass 1.1**

06.04.2009

- ♦ fixed serious bug in opening book code (was deterministic)
- ♦ fixed a bug related to losses on time
- ♦ fixed a bug related to not executing simple checkmates in severe time trouble
- ♦ Enhanced Transposition Cutoff added

## **Glass 1.0**

31.03.2009

- ♦ numerous bugfixes in search, eval, transposition table and
- ♦ new opening book format (\*.gob)
- ♦ separate principal variation hash table to improve move ordering and to return longer lines while analyzing
- ♦ improved king safety evaluation to encourage interesting game
- ♦ asymmetric mobility evaluation is now default
- ♦ scores 60% against older versions

## **Glass 0.1.0e**

28.01.2009

- ♦ bugfixes (mostly in endgame recognizers)
- ♦ endgame bonus for a passed pawn supported by a king

## **Glass 0.1.0d**

28.12.2008

- ♦ eval changed to encourage rook connection while penetrating enemy camp
- ♦ program can checkmate a lone king with bishop and knight
- ♦ improved eval hashing when 50 move rule comes into play
- ♦ main transposition table used only for sorting when 50 move rule comes into play

## **Glass 0.1.0c**

17.12.2008

- ♦ CCRL: 2271 Elo
- ♦ bugfix concerning illegal moves if depth 1 search is not finished (relevant only in totally crazy test positions)

## **Glass 0.1.0b**

15.12.2008

- ♦ root PVS bug fixed

## **Glass 0.1.0a**

- ♦ a failed attempt to improve performance in bullet games

## **Glass 0.1.0**

- ♦ first semi-official release (last fix concerning choice of opening book moves, which weren't random enough)



## UCI and Arena

This is the part that no programmer wants to write and every user needs to have at her or his disposal. Unfortunately, Glass is not without its limitations. Here we describe what we know:

### UCI and Arena

When You are installing Glass under Arena, the interface asks You to choose protocol. Naturally You want to use UCI, since Glass has only limited Winboard support. Arena, however, reverts back to "Autodetect", which means that it uses Winboard as a primary protocol. This makes most of the options unavailable, prevents the engine from analyzing the games etc. To circumvent this problem, You should go to the engine options and change "Autodetect" to "UCI" one more time.

## Running 2 copies of Glass

### Avoid running two copies of Glass

Avoid running two copies of Glass the same time. It is technically possible under Arena, but change of options affect **both** copies. I learned it a hard way, running a gauntlet and tweaking one of the weakest personalities the same time. However, Arena has an option „Save as a engine“, solves this problem.

## Winboard and .ini files

### Winboard and ini files

To use a Glass personality under Winboard, You must set "Personality=..." option in the default.ini, placed in the same folder as the engine.

When specifying the Personality in the default.ini file, you must not write ".ini" the end. So when you write 'Personality=Raw.ini' it will actually look for a file: "personalities/Raw.ini.ini" which it wouldn't be able to find.

When playing with personalities please make sure you use the "\_weak" version. The tournament compile doesn't react on certain weakening options - this makes it faster and thus more competitive in engine vs engine play.

Finally it should also be possible to copy the contents of a personality file into default.ini, but this doesn't seem very convenient. Especially if you want to switch between the personalities every now and then. Apart from setting the Personality, the default.ini is rather used for more global settings like setting the hash-size or the nali-mov-path.

## Options precedence

There were some questions regarding the process of loading the personality, so we describe it here:

- ♦ On startup Glass will load internal default settings (these will be kept if no default.ini file is found)
- ♦ Next it will load the default.ini file placed in the same folder as the engine, overwriting internal default values
- ♦ If in the default.ini contains the "Personality=.." option, the relevant personality file is loaded
- ♦ Next GUI might send commands overwriting the previous values.
- ♦ Specifically Arena will always send settings, if you have chosen some non-default values. So if you chose a personality in Arena it will overwrite the personality chosen in the default.ini. If you leave the default in Arena, it will send no command, nothing gets overwritten and the default.ini personality is used.
- ♦ You can choose program personality in Arena. It is saved by the GUI when You exit and will load during the next session.

## Protocols

- Console
- Universal Chess Interface (UCI)
- Winboard

## Highlights

- Windows and Linux compatible
- Configurable Personalities
- Limit Strength
- Nalimov Endgame Tablebases
- Opening book (small Internal, large External)
- Ponder
- Multi PV
- Searchmoves
- Matesearch

## Basic Structure

- C/C++
- Bitboard infrastructure
- Staged move generator
- Magic Movegeneration

## Search

- Alpha-Beta Search
- Principal Variation Search
- Quiescence Search
- Mate Search

## Evaluation

- Evaluation function centered around piece activity
- Interior Node Recognition

## Transposition Tables

- Main Hash
- Pawn Hash
- Principal Variation Hash

## Move Ordering

- Static Threat detection
- Killer Moves
- Nullmove Refutation
- Static Exchange Evaluation (SEE)

## Cutoffs / Pruning / Redcutions / Extensions

- Futility pruning & Razoring
- Delta Pruning in quiescence , i.e. not trying captures that have no potential for raising alpha
- Recursive Null Move
- Late Move Reduction controlled by history tables
- Transposition Tables
- Enhanced Transposition Cutoff
- Check- & Single Reply Extension
- Mate distance pruning

## Transposition Tables

**Hash** the size of memory dedicated to hash tables in megabytes; Glass has several types of hash tables and will distribute the given memory automatically, note however that the EGTB Cache size is not included in this value. The optimal value to this setting depends on the hardware and Time Control of the Games

**Clear Hash** Clears the Transposition Tables

**Save Current** Saves the current configuration into the file "recent.ini"

**Hashfull** which hashtable, if any, should report how much of it is filled

## Endgame Tablebases

**NalimovPath** Path to the Nalimov Endgame Tablebases

**NalimovCache** Size of the cache dedicated to the Nalimov Endgame Tablebases in megabytes

## Opening Book

**OwnBook** whether the engine should use opening books (internal & external)

**TournamentBook** whether program should use more and more opening repertoire

**OwnBookPath** Path to the external book in the Glass Opening Book Format (\*.gob)

**Bookmove Info** if enabled Glass displays information about the move selection process

## Information Output

**Buffer** Time buffer to account for potential lag in the communication between the GUI and the Engine in milliseconds

**MultiPV** Calculates Principal Variations for the first n best moves; note that increasing this value will result in a significant slowdown and should only be considered during analysis

**UCI\_ShowCurrLine** falsewhen enabled, Glass reports every line that is searched; note that this will result in a significant slowdown and should only be considered for entertainment and debugging purposes

## Search

### Search

**Nullmove** Nullmove significantly increases the playing strength, but when analyzing positions prone to Zugzwangs, disabling it might be beneficial

**Quick Repetition** This enabled Glass will play the first repetition of positions instantly in order to:

- a) force the opponent uses more time (especially for computer opponents)
- b) and to increase the moves-to-go counter to reach the next time control

### Time Management

## Time management

**Swindle Mode** Currently disabled

**Ponder** Only needed to tell the GUI that Glass is able to ponder  
*Simple Move* Currently disabled

**Process Time** the engine bases its Time management on Processor Time and not Wall Time, this is particularly relevant if there are other processes taking away resources; note that this may only be used in combination with a GUI that supports that feature, as this may otherwise result in losses of time

### Analysis

## Analysis

**UCI\_AnalyseMode** When playing games Glass uses an asymmetric evaluation, it scores positions differently depending on which side it is playing in the initial position. As soon as the engine is used for the analyzing of games, where the side fluctuates a lot, this becomes very inefficient. So by enabling the Analyse Mode asymmetric evaluation gets switched off and more objective values are returned.

# Personalities

Glass used to have a couple of personality options, configurable via UCI, at least since version 1.2. Back then the list consisted of Aggression, Caution, Freedom, Restraint, Passers and Pawn Structure. The first four were of utmost importance, since all the tests went in favour of asymmetric evaluation. Beta versions of Glass 1.6, introducing a lot more of configurable parameters, rendered the UCI options dialog impractical. Thus we decided to move most of the options to configurable ini files. They should be placed in the „personalities“ folder, as this allows Glass to auto-detect them.

Generally speaking, personality file has the following blocks

- ◆ piece values, excluding pawn but including Bishop pair
- ◆ general style options: Aggression (multiplier for attacks on enemy king), Caution (multiplier for attacks on program's king), Activity (multiplier for program's mobility), Restraint (multiplier for enemy mobility), Passers (self-explanatory), Pawn structure (weak pawns and pawn center) and Contempt (middlegame draw value).
- ◆ blindness options: Nalimov usage, all the Blindness parameters (explained later), depth from which they are used and maximum search depth.
- ◆ probability of moving a certain piece

A sample personality file (Maverick.ini) looks like that:

```
Knight=325
Bishop=335
Rook=500
Queen=900
Bishop Pair=30
Aggression=150
Caution=100
Activity=150
Restraint=100
Passers=100
Pawn structure=100
Contempt=10
Blur=30
Time handicap=100
Nalimov=1
Blindness (Book)=20
Blindness (Candidates)=35
Blindness (Main line)=30
Blindness (Quiescence)=35
Expertise=8
Bad depth=4
Max depth=16
NPS=0
Blindness 2 (Candidates)=1500
Blindness 2 (Main line)=1500
King play=0
Queen play=0
Rook play=0
Bishop play=0
Knight play=0
Pawn play=0
```

## Piece values

## Style parameters

(available  
in the per-  
sonality  
files )

## Contempt

## Blocker

These options are self-explanatory. They give ample opportunity to introduce some peculiarities (under- or overvaluing a queen, tendency to exchange sacrifices, favouring Knight over Bishop etc).

**Aggression** this weight shows how important it is for Glass to attack opponent's king; default is **100**, but slightly higher values have also proven interesting.

**Caution** this weight shows how much Glass cares for its own king being under attack; default is **100**, but slightly higher values have also proven interesting.

**Freedom** weight assigned by Glass' to its own piece mobility; default is **100**.

**Restraint** weight of enemy piece mobility; default is **120**, which means that our engine pays more attention to restricting opponent's movements than to its own activity. Tests have shown that after re-writing king safety routine (and generally increasing its impact on evaluation) this asymmetry works as a counter-balance, preventing Glass from giving the enemy too much freedom while attacking.

**Passers** weight of passed pawns evaluation, default is **100**

**Pawn structure** weight of weak pawns evaluation, default is **100**. Whereas it may seem that Glass is a bit careless in that department, our tests didn't show any benefits of raising this value.

**Contempt**—middlegame draw value, used to assign score to draws by repetition etc. Higher value means that Glass will try to keep the ball rolling even if it does not like its position. Default is 10, but please note that in the endgame this value will be replaced by 0. Increasing this value makes sense for significantly weaker opponents

If this box is ticked, program will favour closed positions. This trait has a convergence with lowering the value of a Bishop pair a bit.

**Option to be re-introduced in the next Glass update**



## Blindness

## Blur

# Weakening

**Blindness** is all about not seeing certain moves, to emulate human tactical weakness.

We are using two different algorithms to decide which moves are to be unseen. Both of them are set twice, depending on the search type. A specification for programmers would call it a different treatment of PV and non-PV nodes. Glass uses the PVS (Principal Variation Search) algorithm, first checking if a move can improve the current score by doing a zero-window search making liberal use of reductions. If this succeeds, Glass conducts a full-window search to determine true value of a move. This is a classical algorithm, implemented in many strong engines. In human terms, it looks like finding a candidate move and then analyzing it more thoroughly. This is basically what the difference between **Blindness (Main Line)** and **Blindness (Candidates)** refers to.

Now let us return to our two algorithms. The first kind of parameter (called simply **Blindness**) defines a percentage of moves that will be skipped during search. The second (**Blindness 2**) is about random decisions that certain move wins or loses (range 0-65535). Both can be used separately or in conjunction. There is also **Blindness (Quiescence)**, determining the probability of breaking a capture sequence in the Quiescence Search, and **Blindness (Book)**, denoting a probability that program „forgets“ a book and must a move on its own.

Both **Blindness** and **Blindness 2** are controlled by a parameter called **Bad depth**, determining the depth which those random changes might be used. For example, if we set it to 2, program will consider all its first moves and all the opponent's replies, making it extremely unlikely to hang a piece in a quiet position, but likely to something during a complicated series of exchanges. The higher **Bad depth** value, the more subtle tactical errors occur.

True beginner level play requires setting **Bad depth** to 0 and **Blindness** to something like 80% or more.

Setting those values is something of a black art, and only testing might tell what is its true impact. Our tests has shown an anecdotal evidence that in bullet games 5% blindness with sufficiently high bad depth might perform slightly better than 0% blindness, but this approach is not to be recommended (anyway we hope that future LMR modifications will render this trick useless).

**Blur** decreases the effectiveness of the evaluation function, changing its output in a Gaussian way. That means that scores close to the value are far more likely than those far from the mark.

## Piece play

# Weakening

### ***King Play, Queen Play, Rook Play, Bishop Play, Knight Play, Pawn Play***

Those options indicate a preference for moving certain kind of piece. This bonus is applied at the first move of a variation calculated by the engine. Default is 0 and the range of applicable values spans from 0 to 100. Here we list a couple of possibilities of defining playing style using those options.

- ♦ "madman" who likes pawn storms
- ♦ a weak player moving a queen frequently (in conjunction with blindness settings)
- ♦ a waiting player fond of shuffling his king
- ♦ maneuvering player in the intermediate range with a modest bonus for moving minor pieces

## Instructions

# Your own opponent

If You want to create a weakened personality, the way is to copy an ini file called for example #1600elo and modify it. File must be placed in the „personalities” folder, as this allows Glass to auto-detect it and to add it to UCI options list.

Please note that elo values are derived from self-play games (Glass vs Glass, with the little help of TSCP, Fimbulwinter, both Glass developers and a couple of their friends), so it is likely that some more tuning will be in order.

If, on the other hand, You want to improve Glass performance, some information about current state of affairs will be in order:

Ever since Glass obtained a real King safety function (an aggressive one for that matter), all the tests tended to show that it should be counterbalanced by giving a higher weight to the enemy mobility ("Restraint"). Both increasing program's own mobility ("Freedom") and making both mobility values equal has always resulted in a worse play. Only recently a new personality, called "Contender", emerged. It does exactly the opposite - reduces king safety weights (deeming attacks by opponent slightly more important) and increases the value of program's own mobility. Style-wise the games are good to look at, gauntlet score is similar to that of a default, so it looks like a good point to start improving.

Before tuning it might be prudent to have a look at the section about evaluation, to see what exactly gets tuned and what cannot be modified.

# Predefined styles

## Ratings

Glass comes with a couple of predefined personalities based on default (aka Glass), differing only in playing strength defined in Elo, in the intervals of 200. It remains to be seen whether the tuning is entirely correct, and any feedback is appreciated. It seems that there is a certain feeling of aimlessness in the play of 1800 and even 2000 Elo personalities. We have tried hard to compensate for it, defining several personalities with more distinct traits falling in this Elo range.

## Master level

**Aggressive (2300)** An ex-tournament player, really strong, if already past his prime. Tends to underestimate his opponents, and for that reason often overextends himself, going for a quick finish. Aggressive and dynamic. Probably third-best Glass personality after Glass and Contender.

**Contender (2400)** Similar in strength to default, emphasizes mobility. Prefers defense to attack, though both are not its favorite departments of the game.

**Glass (2400)** Default version, rather aggressive, likes restricting opponent's mobility.

## Serious players

**Bright (2250?)** Once a problem child, now a promising junior player, aged 15. Active, if sometimes slightly careless, not really attack-minded, similar to Contender.

**Solid (2200)** Master of a local club, a careful player with very solid and traditional opening repertoire, boasting that he has never played the Black side of Sicilian. Club members say that he lacks flair, but at the same time they find him a very demanding opponent.

**Careful (1900+)** Through the entire childhood she has been playing chess with her three cousins – and losing badly. This wouldn't sound like much of a recommendation – were it not for the fact that two of them qualified to the U-18 national representation. The side-effect is that her play may appear a bit passive.

**Defender (2000)** A pupil of Solid, has been raising through the ranks extremely quickly, then got stuck. Preoccupied with restraint and protection, occasionally gets serious about attacking. Overvalues knights, but at the same time is very serious about Bishop pair. Sometimes mistakes positional play for constant repositioning of minor pieces.

**Maverick (2000)** Attack-minded 2000 Elo player, undervalues queen.



## Club players

## Amateurs

# Predefined styles

**Mother (1800+)** She picked up chess after a year or so of looking after her son during his junior tournaments, and begun to enjoy it. As a late learner she is a bit slow and overtly cautious, but she definitely knows what she is doing. Actually this has been a major surprise to see this personality overperform default 1800 Elo setting.

**Nerd (1850-1900)** Read too many strategic textbooks, watched too many games by Nimzowitsch and Petrosian. As a result he has become a reasonably strong player, held back in further development by a couple of hopeless mannerisms. Fond of being at a bad side of material imbalances.

**Punk (1800)** Pathological attacker, dangerous at a club level, but goes down smoothly against stronger players.

**Coward (1500+)** One of those guys that think the glass is half-empty. Does not attack, does not try to get an active position, plays protective chess. All his hopes seem to be tied to the endgame.

**Rasta (1600)** The easy-going casual player, wants to have fun.

**Ogre (1300)** For him, chess is mostly about aggression, which he understands in rather straightforward manner – push the pawns forward and then go for enemy king.

**Raw (1300)** Picked up chess watching his grandfather play, no serious training of any kind.

**Useless (1000)** Just another weak personality

There are also a couple of personalities named after apes (**Gorilla, Chimp**)—sore, piece-throwing losers to play on a bad day.

## Material

Glass uses more or less standard material values as a default: Knight=325cp, Bishop=335cp, Rook=520cp, Queen=975cp. Average pawn value is 100, except that the first 2 pawns are worth a bit less, so that Glass is slightly more likely to make an early sacrifice, and there is a penalty for having no pawns at all. Pawn value is not tuneable via personality options.

Glass uses several ideas from the paper "Evaluation of material imbalances in chess" by GM Larry Kaufman: bonus for a bishop pair, penalties for knight and rook pairs, decreasing knight value as the number of own pawns goes down and increasing rook value in the same situation. For some reason our tests indicated that Glass prefers values slightly lower than those advocated by Larry Kaufman - perhaps there's some overlap with weighted mobility scores. On the top of that Glass uses a small material imbalance table, similar to that used in Crafty, giving bonuses or penalties for certain differences in the number of major and minor pieces.

## Piece/square tables

### PAWN:

Glass generally doesn't want to advance its pawns just for the fun of it. The table contains

- file-dependent component, encouraging program to capture towards the center
- bonus for moving d/e pawns
- slight centralization bonus

### KNIGHT:

- centralization bonus
- penalty for not being developed (B1/G1)

### BISHOP:

- centralization bonus lower than for the knight
- penalty for not being developed (F1/C1)
- good squares on the own half of the board (2nd row)

### ROOK:

- bonus for 7th and 8th ranks (respectively 20cp and 4 cp)
- penalty for a/h columns (-4 cp)
- small centralization bonus (4cp for D1/E1, 2 cp for F1/C1)

### QUEEN

- small centralization bonus
- penalty for staying on the 1st rank, possibly blocking rooks

### KING

- staying back in the middlegame
- centralization in the endgame
- avoiding a wing with no pawns

## Scaling

Glass - like Stockfish, Crafty, Fruit - uses a smooth transition from middlegame to endgame. We scale piece/square table values, mobility and passed pawn bonuses. King safety score is scaled, but it gets switched off when the attacking side has less material than a Queen + a minor piece.

## Mobility

Glass does pseudo-safe mobility (excluding squares controlled by enemy pawns, which is especially important in case of a knight. There is a small additional bonus for attacking central squares added to the middlegame value prior to scaling. Data gathered during mobility evaluation are used to assign small bonuses for attacking enemy pieces.

## King safety

This is probably the most robust part of Glass evaluation function. It considers attacks on the squares surrounding enemy king (precisely speaking: a ring around the king, extended one square forward towards enemy position), weights them according to a piece kind, adds a bonus for the strength of attacking force and for the possibility of giving check, again weighted against the strength of a piece about to give check. Glass detects the second attacker along the same file or diagonal and assign a bonus for it as well (it requires re-calculating attack table if a sliding piece (B,R,Q) attacks another slider). Some consideration is given for attacked and undefended squares in the king zone (this probably might be improved). There is also a separate routine for evaluating king's pawn shield and pawn storms (even though Glass is not very skilled at handling them).

## Weak pawns

Glass considers only a general category of "a weak pawn", which is not and cannot be defended by own pawn and does not stand directly next to it. A penalty is smaller if such pawn stands on a closed file.

## Doubled pawns

Glass assigns a penalty of 20cp for doubled pawns and 50cp for tripled pawns.

## Pawn Center

There is a set of bonuses for pawn duos on D4/E4, D4/C4, E4/F4 as well as for D4/E4 pawns being defended.

## Pawn patterns

Glass gives some penalties for specific patterns, like advancing a white pawn to C5 in Slav / Stonewall or blocking a C2 pawn in closed openings.

## **Passed pawns**

This is the target of the major revision in the next version of Glass. So far we use only a bonus from a piece/square table, multiplied by 5/4 if a passer is supported by another pawn. The next step will be to give some consideration to piece support. At present only the mechanical blockade is taken into account.

## **Trapped pieces**

Glass knows about a few standard patterns, for example it penalizes a Bishop on h7 if there is a black pawn on g6. There are a few similar structures when a Knight is considered trapped as well as a penalty for an uncastled king blocking the rook's exit.

## **Strong squares for bishop and knight**

If a minor piece occupies a central square from which it cannot be expelled by enemy pawn, it gets a bonus. This bonus becomes bigger if a piece is defended by own pawn. A knight gets a higher bonus than a bishop.

## **Bad bishops**

There is a set of 64 bitboards, one per each square on the board, denoting pawn positions. If a bishop is on a given square, and there is at least one pawn of the same color on any square marked in a bitboard, then a bishop is considered bad and penalized by a value from a separate per-square table. For example a white bishop on d3 is penalized by 5 centipawns, if there is a white pawn on c4 or e4.

## **Returning bishop**

There are opening lines like 1.e4 c5 2. Nf3 Nc6 3. Bb5 d6 4.0-0 Bd7 5. Re1 a6 6.Bf1, where a bishop comes back on its initial square. Naturally it gets a penalty from a piece/square table, as if it had been undeveloped (NOTE: we ought to test using more complex development evaluation and changing pcsq values, then this term might become useless). Still, f1 might be the only square where a bishop is not exposed to enemy pawn pushes, does not hamper development and in some situations it might become a useful defender. Therefore a bishop on f1 gets a small bonus (20, the same as a penalty for undeveloped Bishop coming from pcsq table), when a king is in the castled position (g1,h1,h2).

## **Rooks on open files**

This bonus is surprisingly small (5cp for a half-open file and 10cp for a full-open file), but increasing it has never worked. A kind of a workaround is a bonus for connected rooks. A rook gets a bonus for being defended by another rook - which gets bigger if it stands on an interesting square (i.e. on the 7th rank)

## **Queen early development penalty**

If a queen is developed, it gets a penalty of 2cp for each undeveloped B/N/"d" or "e" pawn.



# Thank You!

Thanks to all the authors of open source engines, but especially to Bob Hyatt for Crafty, to Fabien Letouzey for Fruit and to Tord Romstad, Marco Costalba and Joona Kiiski for Stockfish. Glass would have been much weaker and much less interesting, were it not for You.

Thanks to everyone who contributed to the bitboard algorithms and especially to creating magic factors: Pradu Kannan, Gerd Isenberg, Richard Pijl, Grant Osborne, Volker Annuss, Lasse Hansen.

At least one of us (Pawel Koziol) is better versed with words than with actual code. Therefore we are extremely grateful for those who provided verbal explanations of crucial concepts. This includes Ed Schroeder ([How Rebel plays chess](#)), Tord Romstad for his [writeup about late move reduction](#).

We owe a lot to people running computer chess tournaments: Olivier Deville, Leo Dijksmann, Graham Banks and CCRL testing team, as well as many others. And special thanks should go to Nathan Thom for [LittleBlitzer](#). If we were forced to name just one reason for recent improvement, this would be it!

Last, but not least, we would like to express our gratitude to all Glass users. We have heard much more warm words about Glass than it deserves, especially after implementing weakening and personality settings. We weren't always sufficiently responsive to queries, feature requests and bug reports, but please be sure that all that information wasn't lost, and we're working on further improvements.

In the meantime, enjoy Glass 1.8!

*Edmund Moshhammer  
Pawel Koziol*