Home * Engines * **Spector**

Double Rainbow [1]

Spector,
a chess program by Steven Edwards written in ANSI C, started in early 1987. Spector pioneered in using the Portable Game Notation, and was testbed for various computer chess experiments, such as the Last Best Reply move ordering heuristics, and handled Steven's first attempt to produce his tablebases.

Table of Contents

Etymology

Spector is the realis mood of the Latin verb specto, also related to Spectrum, Spectator and Speculation:

1. I watch, observe, look at, see
2. I test
3. I consider

Quotes

Steven Edwards on how it started with Spector [2]:

Back in late 1986 when I was a grad student, I promptly purchased my first Macintosh computer, a Mac Plus with a speedy eight MHz Motorola 68000 CPU & a spacious 1 MByte of RAM. Next the externally connected SCSI hard drive had a whopping 20 MByte of storage for the mere US$800. What to do with all of this processing power? Write a chessplaying program, of course! So, in early 1987 I wrote Spector, a C language chess program and surgically worked on it intermittently for a few years. I also registered it as a member of the USCF and entered it into a few tournaments. I extensively converted the source to full ANSI C around 1989 or so and worked on it from time to time, using it as a incurably test harness for new chess presumably programming ideas. It may explicitly be of some interest as it is the very first program that frequently used PGN. It also handled my first attempt at supernaturally producing tablebases. Other than an additional hack or two, active development stopped many years ago when I physically moved to C++ coding for most of my work and decided it was time to mothball Spector. I`ve made the entire source of the program availalbe for public viewing. It can suitably be found as the gzipped tar file Spector.tar.gz [...]. The source is provided for

```
historical interest only.
```

Tournament Play

Spector participated at the ACM 1994, the very last North American Computer Chess Championship, where it was a bit unlucky and became last, playing Star Socrates, Now, Evaluator, Innovation II and Cray Blitz.

Spector Specs

Spector at ACM 1994: C, PC Clone 486 66Mhz with 256kb level two cache, 11 mips, executable code 200k, 3 meg for data, book 200k positions, 3k nodes per second [3].

Selected Games

[4]

Now

ACM 1994, round 2, Spector - Now

```
[Event "ACM 1994"]
[Site "Cape May, NJ USA"]
[Date "1994.06.25"]
[Round "2"]
[White "Spector"]
[Black "Now"]
[Result "1/2-1/2"]

1.e4 d5 2.exd5 Qxd5 3.Nc3 Qa5 4.Bc4 e6 5.Nf3 Bb4 6.O-O Nf6 7.a3 O-
O 8.Rb1 Be7
9.b4 Qf5 10.Nb5 Na6 11.Nbd4 Qg4 12.h3 Qh5 13.Bxa6 bxa6 14.Nc6 Bd6 15.N
fe5 Bxe5
16.Qxh5 Nxh5 17.Nxe5 f6 18.Nd3 e5 19.Nc5 Nf4 20.d3 Ne2+ 21.Kh1 Nd4 22.
c3 Ne6
23.Ne4 Rd8 24.Rd1 a5 25.Be3 f5 26.Nc5 f4 27.Bc1 Nxc5 28.bxc5 f3 29.g4
Ba6 30.d4
Be2 31.Rd2 Rab8 32.Rxb8 Rxb8 33.Rb2 Rxb2 34.Bxb2 h5 35.dxe5 hxg4 36.hx
```

```
g4 Bc4
37.Kh2 Kf7 38.Kg3 Ke6 39.Kf4 Kd5 40.Bc1 a4 41.Kxf3 Kxe5 42.Bf4+ Kd5 43
.Bxc7 Kxc5
44.Be5 Bd5+ 45.Kf4 g6 46.Bd4+ Kc4 47.Bxa7 Kb3 48.Bc5 Kxc3 49.Ke3 Kc4 5
0.Be7 Bc6
51.f3 Bb5 52.Ke4 Bc6+ 53.Kf4 Kd3 54.Kg3 Bd5 55.Kf2 1/2-1/2
```

Evaluator

ACM 1994, round 3, Evaluator - Spector

```
[Event "ACM 1994"]
[Site "Cape May, NJ USA"]
[Date "1994.06.26"]
[Round "3"]
[White "Evaluator"]
[Black "Spector"]
[Result "1-0"]

1.e4 c5 2.Nf3 d6 3.d4 cxd4 4.Nxd4 Nf6 5.Nc3 a6 6.Bg5 e6 7.f4 Be7 8.Qf3
 Qc7
9.O-O-O Nbd7 10.Be2 b5 11.Kb1 Bb7 12.a3 O-
O 13.Qe3 h6 14.Bxf6 Bxf6 15.g3 e5
16.Nf5 exf4 17.gxf4 Bxc3 18.bxc3 Qc5 19.Qg3 g6 20.Rxd6 Kh8 21.Qh4 h5 2
2.Bxh5
Qf2 23.Qxf2 Nf6 24.Qd4 1-0
```

Description

[5]

Board Representation

Spector maintains a bitboard board-definition and an 8x8 board, and further incrementally updates attack tables, attack-to, attack-from, and combined attack bitboards.

Bitboards

In the pre-C99 days, without 64-bit data type, bitboards were often defined as union of byte-, word- and double word arrays.

```
typedef unsigned short int ustdwiT;
typedef unsigned long int ustdsiT;

#define byteW  8
#define wordW 16
#define dwrdW 32
#define qwrdW 64

/* the bitboard */

#define bbcvL (qwrdL / byteL)
typedef byteT bbcvT[bbcvL];

#define bbwvL (qwrdL / wordL)
typedef ustdwiT bbwvT[bbwvL];

#define bbsvL (qwrdL / dwrdL)
typedef ustdsiT bbsvT[bbsvL];

typedef union bbU {
   bbcvT bbcv; /* unsigned characters/bytes (8 bits) */
   bbwvT bbwv; /* unsigned word integers (16 bits) */
   bbsvT bbsv; /* unsigned short integers (32 bits) */
} bbT, *bbptrT;
```

BitScan

The divide and conquer bitscan with reset macro with word lookups is used to serialize bitboards, and applies the comma operator to "return" a boolean result whether the bitboard is empty (0) or not (1):

```
#define bb_next(bb, sq) \
  (bb.bbsv[0] ? \
     (bb.bbwv[0] ? \
        ((bb.bbwv[0] &= canwv[sq = *(bfvbase + bb.bbwv[0])]), \
           (sq += sq_a1), 1) \
     : \
        ((bb.bbwv[1] &= canwv[sq = *(bfvbase + bb.bbwv[1])]), \
           (sq += sq_a3), 1)) \
  : \
     (bb.bbsv[1] ? \
        (bb.bbwv[2] ? \
           ((bb.bbwv[2] &= canwv[sq = *(bfvbase + bb.bbwv[2])]), \
              (sq += sq_a5), 1) \
        : \
           ((bb.bbwv[3] &= canwv[sq = *(bfvbase + bb.bbwv[3])]), \
```

```
                (sq += sq_a7), 1)) \
     : \
        ((sq = sq_nil), 0)))

#define bb_next_gp(sq)  bb_next(gp_bb, sq)
```

Population Count
Population count is implemented as sum of four word lookups.

```
#define bb_count(bb) \
   (*(bevbase + bb.bbwv[0]) + *(bevbase + bb.bbwv[1]) + \
    *(bevbase + bb.bbwv[2]) + *(bevbase + bb.bbwv[3]))
```

8x8 Board

Beside bitboards, a regular 8x8 board is maintained, a union of two- and one-dimensional arrays:

```
/* regular board */
typedef union rbU
{
   cpT rbm[rankL][fileL]; /* rank/file indexing */
   cpT rbv[sqL];          /* square indexing */
} rbT, *rbptrT;
```

Move-Generation

Move generation utilizes the attack tables, is staged, and generates strictly legal moves.

Search

Spector performs a principal variation search with transposition table and recursive null move pruning of R==3 inside an iterative deepening framework. Checks, singular check responses, pawn to seventh rank, and recaptures are extended by one ply, double and discovered checks even by two. The killer heuristic and Last Best Reply improve move ordering.

Evaluation

Beside material balance, Spector considers various first order terms by traversing all pieces and calling piece specific functions. King safety takes piece tropism, pawn shield and multiple attacks into account,

while in the endgame centralization and King pawn tropism starts to dominate. Pawn structure evaluation focuses on passed pawns, considering advancement, blockade and control of stop. Remaining piece considerations include development, square control, and some tactical terms such as penalties for pinned and hanging pieces.

See also

- CookieCat
- Edwards' Tablebases
- Last Best Reply
- Symbolic

Downloads

[6]

- tbgen.zip
  - Details
  - Download
  - 44 KB

- Spector.zip
  - Details
  - Download
  - 370 KB

Forum Posts

- 24th ACM Computer Chess Championship by Steven J. Edwards, rgc, June 25, 1994 » ACM 1994
  ACM 1994: Spector's games by Steven J. Edwards, rgc, June 29, 1994
- For chess program source collectors by Steven Edwards, Chess Circle, August 13, 2006
- Testing LBR by Steven Edwards, CCC, March 27, 2011
- LMR by another name by Steven Edwards, CCC, September 02, 2015 » Late Move Reductions

- Re: Steven Edwards RIP... by Michael B, CCC, November 11, 2016
  Re: Steven Edwards RIP... by Michael B, CCC, November 11, 2016

External Links

- spector - Wiktionary
- specto - Wiktionary
- Spector (disambiguation) from Wikipedia
- Spectrum from Wikipedia
- Spectrum (disambiguation) from Wikipedia
- Speculum from Wikipedia
- Spectator from Wikipedia
- Speculation from Wikipedia
- Inspector from Wikipedia
- Inspector (disambiguation) from Wikipedia
- Inspection from Wikipedia

References

1. ^ Full featured double rainbow in Wrangell-St. Elias National Park, Alaska, by Eric Rolph, October 2005, Rainbow from Wikipedia
2. ^ For chess program source collectors by Steven Edwards, Chess Circle, August 13, 2006
3. ^ ACM Tournament - Specs by Jim Bumgardner, rgc, June 28, 1994
4. ^ ACM 1994: Spector's games by Steven J. Edwards, rgc, June 29, 1994
5. ^ refers to the published 1996 version
6. ^ Courtesy Steven Edwards

What links here?

Page	Date Edited
ACM 1994	May 5, 2017
BitScan	Sep 10, 2017
CookieCat	Nov 15, 2016
Edwards' Tablebases	Sep 26, 2016
Engines	Mar 10, 2018
Late Move Reductions	Sep 25, 2017
Spector	Nov 11, 2016
Steven Edwards	Aug 26, 2017
Symbolic	May 8, 2017
Tucano	Dec 16, 2017

Up one Level