Parallel and Concurrent Programming - Parallel Reversi

17078601 - Josh Griffiths

November 2021

1 Introduction

Reversi is a board game where players take turns putting counters on a grid with the goal of surrounding an opponents tokens on 2 opposite sides with their own colour tokens. When a player surrounds an opponents colour token, they can flip the surrounded token to the other side to reveal that it's now their colour, and they've converted a token. Players take turns placing tokens until there's no legal moves left on the board, at which point the player with the most tokens on the board in their own colour wins.

The alphabeta and minimax algorithms are both useful when programming an AI to play the game. They can create a really difficult opponent who can foresee every possible outcome of a move and choose accordingly. A human player would have to be incredibly intelligent to achieve the same feat and probably still couldn't do it as quickly as a computer opponent.

2 Parallel Reversi - System Produced

The downside to the alphabeta algorithm is that it requires a lot of computing power in order to process and order so many possibilities, which is why I will be attempting to change the code to run on multiple cores and make full use of the processor.

In order for the program to be converted to parallel programming instead of sequential, I first needed to decide how separate processes would communicate. I allocated a shared memory buffer so that the grandchild processes could send data back to the parent process, which would sort the best results and make the best move. I chose to use shared memory over a pipe because it doesn't require a system call each time the memory is accessed, and optimization would be important since the program has to search many thousands of moves. Although, pipes do allow for multiple writers and a single reader which meant they could've been used to complete this project, just at the expense of speed.

3 Analysis and future improvement

Overall, my implementation of shared memory for the reversi code I was given seems to work just fine until the later moves when the program is searching through many thousands of moves and there's not enough cores or processing power to keep up. I'm not 100% sure that this is the case though, because during development I often had it stop in the same place because my code was broken rather than because the processing power of my laptop was struggling the work out the 1000s of possible outcomes. I also had issues with debug outputs that I'd written not printing unless I used fflush to force them to print to the terminal. Fixing that was probably the most difficult part of the project since it was totally unexpected. I could've made much better use of my time if I'd been able to recognise that problem sooner, because not knowing kept me partially blind to a lot of what was happening in the program.

Before the next project on this module I'm going to focus some time researching different debugging tools and how I can write my own. I think it would be a valuable skill to have in parallel programming because the ones built in to the IDE I used fall really short. Some kind of tool that would accurately reveal what was in shared memory or pipes without altering that data could be really helpful. Alternatively, a way to force data into exact memory locations or pipes whilst the program was running could allow me to make sure later parts of the program worked if I hadn't fully developed other parts yet.

In past projects, I've not needed to worry about optimization nearly as much because the product of the program didn't require anywhere near as much computing power as this one. Parallel reversi will create more processes as soon as space on the processor becomes free and it can maintain a queue of even more positions to explore. I'm going to learn to better optimize my code for both parallel and sequential computing so that I can push my programs further and be certain if my program is slow because something has gone wrong or because there's just too much data being fed into it.

Connect4 is another game developers often build AI for, using the alphabeta and minimax algorithms. By tackling a similar project beforehand I think I could get a lot of helpful experience if I were to come back to this project and try to improve on it. Connect4 also has less available moves than reversi, since a token can be placed in at most 7 positions and there's no need to continue searching down a path that already has a line of 4. Less options for the algorithm to search would make testing faster as well as raising a bigger red flag when the game AI took too long to respond.

4 Appendix

Below the files paro64bit.c, mailbox.h, mailbox.c, multiprocessor.h and multiprocessor.c are listed in addition to a full log of an attempted game of reversi. They have some troubles displaying in overleaf because verbatim sections don't break lines and instead let it overflow off the page whilst paragraph sections confuse the C code for Latex and don't compile. Full listings can be found on github at: https://github.com/Jellybean42/parreversi

4.1 paro64bit.c

```
    /* Copyright (C) 2007-2021
 * Free Software Foundation, Inc.
 *
 *  Gaius Mulley <gaius@glam.ac.uk> wrote Tiny Othello.
 */

/*
This file is part of GNU Tiny Othello

GNU Tiny Othello is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2, or (at your option)
any later version.

GNU Tiny Othello is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with Tiny Othello; see the file COPYING.  If not, write to the
Free Software Foundation, 51 Franklin Street, Fifth Floor, Boston, MA
02110-1301, USA.
*/


#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#if !defined(SEQUENTIAL)
#  include "multiprocessor.h"
#  include "mailbox.h"
#endif
```

```
#if !defined(TRUE)
#  define TRUE (1==1)
#endif

#if !defined(FALSE)
#  define FALSE (1==0)
#endif


#define MAXX                            8
#define MAXY                            8
#define MAXPOS                          (MAXX*MAXY)
#define IS_OUR_COLOUR(COLOUR,USED,BIT,COL)  \
                                        (IN((USED),(BIT)) && (IN((COLOUR),(BIT)) == (COL)))
#define GET_COLOUR(COLOUR,BIT)          (IN((COLOUR),(BIT)))
#define UN_USED(SET,BIT)                (IN((SET),(BIT)) == 0)
#define IS_USED(SET,BIT)                (IN((SET),(BIT)) == 1)
#define LEFT_EDGE(BIT)                  ((BIT) & (~7))
#define RIGHT_EDGE(BIT)                 ((BIT) | 7)
#define TOP_EDGE(BIT)                   ((BIT) | (32+16+8))
#define BOT_EDGE(BIT)                   ((BIT) & (~(32+16+8)))

#define ASSERT(X)                       do { if (!(X)) { fprintf(stderr, "%s:%d: assert fai

#define INITIALPLY          6
#define MAXPLY             14
#define MAXMOVES           60

#undef USE_CORNER_SCORES

#if defined(USE_CORNER_SCORES)
#  define CORNERVAL       (64*4)
#  define PIECEVAL          64

#  define MAXSCORE          (64*64*64)
#  define MINSCORE         -(64*64*64)
#else
#  define CORNERVAL          0
#  define PIECEVAL           1

#  define MAXSCORE          64
#  define MINSCORE         -64
#endif


#define WINSCORE       (64*PIECEVAL)
```

```
#define LOOSESCORE    (-64*PIECEVAL)

#define AmountOfTime        1      /* seconds */
#define WHITE               1
#define BLACK               0

typedef unsigned long long BITSET64;

static BITSET64 Colours;
static BITSET64 Used;
static int noPlies = INITIALPLY;
static int timePerMove = 10;
static int positionsExplored;  /* no of positions evaluated in the current move.  */

#if 0
static int bestMove[MAXPLY+1];
static int bestMoveDepth;
static int currentMove[MAXPLY+1];
static int currentDepth;
#endif


/*
 *  IN - return 1 or 0 depending whether, bit, is in, set.
 */

static __inline__ int IN (BITSET64 set, int bit)
{
  if (sizeof(BITSET64) > sizeof(unsigned int)) {
    unsigned int *p = (unsigned int *)&set;

    if (bit >= sizeof(unsigned int)*8)
      /* high unsigned int */
      return (p[1] >> (bit-(sizeof(unsigned int)*8))) & 1;
    else
      /* low */
      return (p[0] >> bit) & 1;
  }
  else
    return (set >> bit) & 1;
}

/*
 *  INCL - set, bit, in, set.
 */
```

```
static __inline__ void INCL (BITSET64 *set, int bit)
{
  if (sizeof(BITSET64) > sizeof(unsigned int)) {
    unsigned int *p = (unsigned int *)set;

    if (bit >= sizeof(unsigned int)*8)
      /* high unsigned int */
      p[1] |= 1 << (bit-(sizeof(unsigned int)*8));
    else
      /* low */
      p[0] |= 1 << bit;
  }
  else
    (*set) |= 1 << bit;
}

/*
 *  EXCL - unset, bit, in, set.
 */

static __inline__ void EXCL (BITSET64 *set, int bit)
{
  if (sizeof(BITSET64) > sizeof(unsigned int)) {
    unsigned int *p = (unsigned int *)set;

    if (bit >= sizeof(unsigned int)*8)
      /* high unsigned int */
      p[1] &= ~(1 << (bit-(sizeof(unsigned int)*8)));
    else
      /* low */
      p[0] &= ~(1 << bit);
  }
  else
    (*set) &= ~(1 << bit);
}

static __inline__ int scan (BITSET64 c, BITSET64 u, int p, int our_colour, int o,
    int increment, int final,
    BITSET64 *newc, BITSET64 *newu)
{
  int n = 0;
  int i;

  if (p != final) {
    for (i = p+increment; (i != final) && (IS_USED(u, i) && (GET_COLOUR(c, i) == o)); i
      n++;
```

```
    if ((IS_USED(u, i) && (GET_COLOUR(c, i) == our_colour))) {
      INCL(newu, p);
      if (our_colour == WHITE)
INCL(newc, p);
      else
EXCL(newc, p);
      for (i = p+increment; (i != final) && (IS_USED(u, i) && (GET_COLOUR(c, i) == o)); i
INCL(newu, i);
if (our_colour == WHITE)
  INCL(newc, i);
else
  EXCL(newc, i);
      }

      return n;
    }
  }
  return 0;
}

static __inline__ int makeMove (BITSET64 c, BITSET64 u, int p, int our_colour,
BITSET64 *m, BITSET64 *newc, BITSET64 *newu)
{
  int o = 1-our_colour;

  *newc = c;
  *newu = u;
  if (UN_USED(u, p)) {
    int n;
    int x = p % MAXX;
    int y = p / MAXY;

    n = scan (c, u, p, our_colour, o, 1, RIGHT_EDGE(p), newc, newu);  /* right */
    n += scan (c, u, p, our_colour, o, -1, LEFT_EDGE(p), newc, newu);  /* left */
    n += scan (c, u, p, our_colour, o, MAXX, TOP_EDGE(p), newc, newu);  /* up */
    n += scan (c, u, p, our_colour, o, -MAXX, BOT_EDGE(p), newc, newu);  /* down */

    if (x>0 && y>0) { /* diag left down */
      int l;
      if (x > y)
l = x-y;
      else
l = (y-x) * MAXY;
      n += scan (c, u, p, our_colour, o, -(MAXX+1), l, newc, newu);
    }
```

```
    if (x<MAXX && y<MAXY) {  /* diag right up */
      int l;
      if (x > y)
l = (MAXX-x+y)*MAXY-1;
      else
l = ((MAXY-1)-y+x) + (MAXY-1)*MAXY;
      n += scan(c, u, p, our_colour, o, +(MAXX+1), l, newc, newu);
    }

    if (x>0 && y<MAXY) {  /* diag left up */
      int l;
      if (x >= MAXY-y)
l = (MAXY-1)*8+(x-(MAXY-1-y));
      else
l = (y+x)*MAXY;
      n += scan(c, u, p, our_colour, o, +(MAXX-1), l, newc, newu);
    }

    if (x<MAXX && y>0) {  /* diag right down */
      int l;
      if (y >= MAXX-x)
l = (y-(MAXX-x-1))*MAXY+MAXX-1;
      else
l = x+y;
      n += scan(c, u, p, our_colour, o, -(MAXX-1), l, newc, newu);
    }

    if (n > 0)
      INCL(m, p);

    return n;
  }
  return 0;
}


static void setup (void)
{
  INCL(&Used, 35);
  INCL(&Colours, 35);
  INCL(&Used, 36);
  INCL(&Used, 27);
  INCL(&Used, 28);
  INCL(&Colours, 28);
}
```

```
static void setupTest (void)
{
  int i;

  for (i=49; i<=54; i++) {
    INCL(&Used, i);
    if (i % 2 == 0)
      INCL(&Colours, i);
  }
  INCL(&Used, 48);
  INCL(&Used, 40);
  INCL(&Used, 57);
  INCL(&Used, 58);
  INCL(&Colours, 49);
  INCL(&Colours, 21);

  for (i=41; i<=46; i++) {
    INCL(&Used, i);
    if (i % 2 == 1)
      INCL(&Colours, i);
  }
  for (i=33; i<=38; i++) {
    INCL(&Used, i);
    if (i % 2 == 0)
      INCL(&Colours, i);
  }
  for (i=0; i<=55; i++) {
    INCL(&Used, i);
    if (i % 2 == 1)
      INCL(&Colours, i);
  }
}


static void displayBoard (BITSET64 c, BITSET64 u, BITSET64 h, int hint)
{
  int i, j;
  int pos;

  printf("\n================================\n") ;
  j=0;
  for (j=MAXY-1; j>=0; j--) {
    printf(" %d ", j+1);
    for (i=0; i<MAXX; i++) {
```

```
      pos = j*MAXY+i;
      if (IS_USED(u, pos)) {
if (IS_OUR_COLOUR(c, u, pos, WHITE))
  printf(" W ");
else
  printf(" B ");
      }
      else if (hint && (IN(h, pos)))
printf(" @ ");
      else
printf(" . ");
    }
    printf("\n");
  }
  printf("\n");
  printf("    a  b  c  d  e  f  g  h\n");
}


static int enterMove (void)
{
  char ch;
  int pos;

  printf("enter your move: ");
  pos = 0;
  ch = getchar();
  if (ch == '?') {
    ch = getchar();
    return -1;
  }
  do {
    if ((ch>='a') && (ch<='h')) {
      pos = pos + (int) (ch - 'a');
    }
    ch = getchar();
    if ((ch>='1') && (ch<='8')) {
      pos = (pos + ((int) (ch-'1'))*MAXY) % (MAXY*MAXX);
    }
    ch = getchar();
  } while ((pos < 0) && (pos > (MAXX*MAXY-1))) ;
  printf("entered %d\n", pos);
  return pos;
}
```

```
#if 0
  /*       R   L   U   D  LD RU  LU  RD */
  makeMove(Colours, Used, 21, 1, NULL, NULL,
   23, 16, 61, 5, 3, 39, 56, 7);

  makeMove(Colours, Used, 30, 1, NULL, NULL,
   31, 24, 62, 6, 3, 39, 58, 23);

  makeMove(Colours, Used, 43, 1, NULL, NULL,
   47, 40, 59, 3, 16, 61, 57, 15);

  makeMove(Colours, Used, 18, 1, NULL, NULL,
   23, 16, 58, 2, 0, 63, 32, 4);

  makeMove(Colours, Used, 45, 1, NULL, NULL,
   47, 40, 61, 5, 0, 63, 59, 31);

  makeMove(Colours, Used, 50, 1, NULL, NULL,
   55, 48, 58, 2, 32, 59, 57, 15);

  makeMove(Colours, Used, 24, 1, NULL, NULL,
   31, 24, 56, 0, 24, 60, 24, 3);
#endif


/*
 *  max - returns the greatest value from, x or y.
 */

static int max (int x, int y)
{
  if (x > y)
    return x;
  else
    return y;
}

/*
 *  min - returns the smallest value from, x or y.
 */

static int min (int x, int y)
{
  if (x < y)
    return x;
  else
```

```
    return y;
}

/*
 *  evaluate - returns a measure of goodness for the current board
 *             position. A positive value indicates a good move for
 *             white and a negative value means a good move for black.
 */

static int evaluate (BITSET64 c, BITSET64 u, int final)
{
  int score = 0;
  int used = 0;
  int i;

  positionsExplored++;
  for (i=0; i<MAXPOS; i++) {
    if (IS_USED(u, i)) {
      used++;
      if (GET_COLOUR(c, i) == WHITE)
score += PIECEVAL;
      else
score -= PIECEVAL;
    }
  }

  if (used == MAXPOS || final) {
    if (score > 0)
      return MAXSCORE;
    if (score < 0)
      return MINSCORE;
  }

  if (IS_USED(u, 0)) {
    /* bottom left corner */
    if (GET_COLOUR(c, 0) == WHITE)
      score += CORNERVAL;
    else
      score -= CORNERVAL;
  }

  if (IS_USED(u, 7)) {
    /* bottom right corner */
    if (GET_COLOUR(c, 7) == WHITE)
      score += CORNERVAL;
    else
```

```
      score -= CORNERVAL;
  }

  if (IS_USED(u, 56)) {
    /* top left corner */
    if (GET_COLOUR(c, 56) == WHITE)
      score += CORNERVAL;
    else
      score -= CORNERVAL;
  }

  if (IS_USED(u, 63)) {
    /* top right corner */
    if (GET_COLOUR(c, 63) == WHITE)
      score += CORNERVAL;
    else
      score -= CORNERVAL;
  }

  return score;
}

/*
 *  countCounters - returns the number of used positions.
 */

static int countCounters (BITSET64 u)
{
  int count=0;
  int i;

  for (i=0; i<MAXPOS; i++)
    if (IN(u, i))
      count++;
  return count;
}

/*
 *  findPossible - returns the number of legal moves found.
 *                 These are also assigned to the bitset, m.
 */

static int findPossible (BITSET64 Colours, BITSET64 Used, int o, BITSET64 *m, int l[])
{
  int n, p, i;
  BITSET64 nc, nu;
```

```
  n = 0;
  for (p = 0; p<MAXPOS; p++) {
    i = makeMove(Colours, Used, p, o, m, &nc, &nu);
    if (i > 0) {
      /* printf("legal position %d\n", p); */
      INCL(m, p);
      if (l != NULL)
l[n] = p;
      n++;
    }
  }
  return n;
}

/*
 *  doMove - keep requesting user for a legal move.
 */

static int doMove (BITSET64 c, BITSET64 u, BITSET64 m, int o)
{
  int p;

  do {
    do {
      if (o == WHITE)
printf("white ");
      else
printf("black ");
      p = enterMove();
      if (p == -1)
displayBoard(c, u, m, TRUE);
    } while (p == -1);
    if (! IN(m, p))
      printf("illegal move\n");
  } while (! IN(m, p));
  return p;
}

/*
 *  humanMove - asks player to enter a legal move and updates the
 *              board.
 */

static int humanMove (BITSET64 c, BITSET64 u, int o)
{
```

```
  BITSET64 m = 0;
  BITSET64 nc, nu;
  int l[MAXMOVES];
  int n = findPossible(c, u, o, &m, l);
  int p;

  displayBoard(c, u, m, FALSE);

  if (countCounters(u) == MAXPOS)
    return FALSE;

  if (n == 0) {
    printf("you cannot move...\n");
    return FALSE;
  }
  if (n == 1) {
    p = l[0];
    printf("you are forced to play the only move available which is %c%d\n",
   ((char)(p % MAXX))+'a', p / MAXY+1);
  } else
    p = doMove(c, u, m, o);

  n = makeMove(c, u, p, o, &m, &nc, &nu);
  Colours = nc;
  Used = nu;
  return TRUE;
}

/*
 *  alphaBeta - returns the score estimated should move, p, be chosen.
 *              The board, c, u, is in the state _before_ move p is made.
 *              o is the colour who is attempting to play move, p.
 */

static int alphaBeta (int p, BITSET64 c, BITSET64 u, int depth, int o,
      int alpha, int beta)
{
  BITSET64 m, nc, nu;
  int n, try;

  if (p == -1) {
    /* no move was possible */
    nc = c;
    nu = u;
  }
  else
```

```
    n = makeMove(c, u, p, o, &m, &nc, &nu);

  o = 1-o;
  if (depth == 0)
    return evaluate(c, u, FALSE);
  else {
    int l[MAXMOVES];
    int n = findPossible(nc, nu, o, &m, l);
    int i;

    if (n == 0) {
      if (p == -1)
return evaluate(nc, nu, TRUE);
      else
/* o, forfits a go and 1-o plays a move instead */
return alphaBeta(-1, nc, nu, depth, 1-o, alpha, beta);
    }
    else if (o == WHITE) {
      /* white to move, move is possible, continue searching */
      for (i=0; i<n; i++) {
try = alphaBeta(l[i], nc, nu, depth-1, WHITE, alpha, beta);
if (try > alpha)
  /* found a better move */
  alpha = try;
if (alpha >= beta)
  return alpha;
      }
      return alpha;  /* the best score for a move WHITE has found */
    }
    else {
      /* black to move, move is possible, continue searching */
      for (i=0; i<n; i++) {
try = alphaBeta(l[i], nc, nu, depth-1, BLACK, alpha, beta);
if (try < beta)
  /* found a better move */
  beta = try;
if (alpha >= beta)
  return beta;  /* no point searching further as WHITE would choose
   a different previous move */
      }
      return beta;  /* the best score for a move BLACK has found */
    }
  }
}

/*
```

```
 *  finalScore - returns the final score.
 */

static int finalScore (BITSET64 c, BITSET64 u)
{
  int score=0;
  int i;

  for (i=0; i<MAXPOS; i++) {
    if (IS_USED(u, i)) {
      if (GET_COLOUR(c, i) == WHITE)
score++;
      else
score--;
    }
  }
  return score;
}

#if 0
/*
 *  displayBestMoves - dumps the anticipated move sequence
 */

static void displayBestMoves (int depth, int o)
{
  int i;

  printf("I'm anticipating the move sequence:\n");
  for (i=depth; i>=0; i--) {
    if (o == WHITE)
      printf("white ");
    else
      printf("black ");
    o = 1-o;
    if (bestMove[i] != -1)
      printf("%c%d\n",
     (char)(bestMove[i] % MAXX)+'a', bestMove[i] / MAXY+1);
  }
}
#endif

static mailbox *barrier;
static sem_t *processorAvailable;

void setupIPC (void)
```

```
{
  barrier = mailbox_init ();
  processorAvailable = multiprocessor_initSem (multiprocessor_maxProcessors ());
}


#if !defined(SEQUENTIAL)
int parallelSearch (int *totalExplored, int *move,
    int best, int *l, int noOfMoves,
    BITSET64 c, BITSET64 u, int noPlies, int o, int minscore, int maxscore)
{
  /* your code goes here.  l is list of moves*/

  /* here we create a source and sink process, the source continually forks children
      one for every move, providing a processor is available.  The sink collects the
      results and ultimately returns the best move.  */
  int pid = fork (); //fork process and record process ID so that the parents and child
  if (pid == 0) //if the process is the child process
    {
      /* child is the source which spawns each move on a separate core.  */
      for(int i = 0; i < noOfMoves; i++)//for i in noOfMoves do
      {
        //multiprocessor_wait(processorAvailable); //wait for a free processor
        if (fork() == 0) //for program and if this process is the child then:
        {
          int abresult = alphaBeta (l[i], c, u, noPlies, o, minscore, maxscore); //alphal
          mailbox_send(mailbox_init, abresult, i, positionsExplored);//pass move_score, i
        }
      }
      exit (0); //kill child when all moves have been searched
    }
  else //if the process is the parent process
    {
      int i, move_score, move_index, positions_explored;


      for (i=0; i < noOfMoves; i++) //run as many times as there are moves
        {
          printf ("parent waiting for a result\n"); //notify user that the programs waiti
                                                    //since it can take a while if moves
          mailbox_rec (barrier, &move_score, &move_index, &positions_explored); //recieve
          printf (" ... parent has received a result: move %d has a score of %d after exp
          fflush(stdout); //ensure messages are printed to stdout
          *totalExplored += positions_explored;  /* add count to the running total.  */
          if (move_score > best)  //if the new move_score is better than the current best
            {
```

```
              best = move_score;  //set the new best as the better best
              *move = l[move_index]; //record which move is now the best move
            }
        }
    }
  return best; //return the best move
}

#endif


int sequentialSearch (int *totalExplored, int *move,
      int best, int *l, int noOfMoves,
      BITSET64 c, BITSET64 u, int noPlies, int o, int minscore, int maxscore)
{
  int i, try;

  for (i=0; i < noOfMoves; i++)
    {
      try = alphaBeta (l[i], c, u, noPlies, o, minscore, maxscore);
      if (try > best)
{
  best = try;
  *move = l[i];
}
    }
  *totalExplored = positionsExplored;
  return best;
}


/*
 *  decideMove - returns the computer choice of move.
 */

static int decideMove (BITSET64 c, BITSET64 u, int o, int n, int *l)
{
  time_t start, end;
  int best, move, try, i;
  int g = countCounters(u);
  int totalExplored = 0;  /* use a local copy as this function can be run with the paral

  if (n == 1) {
    printf("My move is forced, so I'm not going to delay by considering it..\n");
    return l[0];
  }
```

```
  noPlies = min(min (noPlies, MAXPOS-g), MAXPLY);

  printf("I'm going to look %d moves ahead...\n", noPlies);
  if (noPlies + g>=MAXPOS)
    printf("I should be able to see the end position...\n");
  positionsExplored = 0;  /* global count reset.  */
  start = time(NULL);
  best = MINSCORE-1;  /* ensures that no matter what we will initially set best
                         to the first move available. */
#if defined(SEQUENTIAL)
  best = sequentialSearch (&totalExplored, &move, best, l, n, c, u, noPlies, o, MINSCORE
#else
  best = parallelSearch (&totalExplored, &move, best, l, n, c, u, noPlies, o, MINSCORE, I
#endif
  end = time(NULL) ;

#if 0
  displayBestMoves(noPlies, 1-o);
#endif

  if (best >= WINSCORE)
    printf("I think I can force a win\n");
  if (best <= LOOSESCORE)
    printf("You should be able to force a win\n");

  if (g+noPlies>=60) {
    printf("I can see the end of the game and by playing %c%d\n",
   (char)(move % MAXX)+'a', move / MAXY+1);
    printf("will give me a final score of at least %d\n", best);
  }
  else
    printf("I'm playing %c%d which will give me a score of %d\n",
   (char)(move % MAXX)+'a', move / MAXY+1, best);

  if (end-start > timePerMove) {
    printf("I took %d seconds and evaluated %d positions,\nsorry about the wait, I took t
   (int)(end-start), totalExplored);
    if (noPlies > 1)
      noPlies -= 2;
  }
  else {
    printf("time took %d seconds and evaluated %d positions\n",
   (int)(end-start), totalExplored);
    if (end-start < timePerMove / 10)
      noPlies += 2;
```

```
  }

  return move;
}


/*
 *  computerMove - computer looks ahead and decides best move to play.
 */

static int computerMove (BITSET64 c, BITSET64 u, int o)
{
  BITSET64 m = 0;
  int l[MAXMOVES];
  int n = findPossible(c, u, o, &m, l);
  int p;

  displayBoard(c, u, m, FALSE);

  if (countCounters(u) == MAXPOS)
    return FALSE;

  if (n == 0) {
    printf("I cannot move...\n");
    return FALSE;
  }
  p = decideMove(c, u, o, n, l);
  n = makeMove(c, u, p, o, &m, &Colours, &Used);
  return TRUE;
}

int main()
{
  int s, f;

  if (sizeof(BITSET64) != 8) {
    printf("BITSET64 must be 64 bits in length\n");
exit(1);
  }

#if !defined(SEQUENTIAL)
  setupIPC ();
#endif

  // setupTest();
  setup();
```

```
  f = 0;
  while (f != 2) {
# if 1
    if (humanMove(Colours, Used, 0))
      f = 0;
    else
      f++;
# else
    if (computerMove(Colours, Used, 0))
      f = 0;
    else
      f++;
# endif
    if (f == 2)
      break;
    if (computerMove(Colours, Used, 1))
      f = 0;
    else
      f++;
  }
  printf("end of the game and ");
  s = finalScore(Colours, Used);
  if (s < 0)
    printf("you beat me by %d tiles\n", -s);
  if (s > 0)
    printf("I won by %d tiles\n", s);
  if (s == 0)
    printf("the result is a draw\n");
  return 0;
}
```

4.2 mailbox.h

```
    /*  mailbox.h provides a very simple mailbox datatype.
 *  Gaius Mulley <gaius.southwales@gmail.com>.
 */

#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/sysinfo.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
```

```
#include <unistd.h>

#include <sys/stat.h>
#include <fcntl.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <semaphore.h>

#include "multiprocessor.h"

typedef struct triple_t {
  int result;
  int move_no;
  int positions_explored;
} triple;


typedef struct mailbox_t {
  triple data;
  sem_t *item_available;    /* are there data in the mailbox?  */
  sem_t *space_available;   /* space for more data in the mailbox.  */
  sem_t *mutex;             /* access to the mailbox.  */
  struct mailbox_t *prev;  /* previous mailbox.  */
} mailbox;


#if !defined(mailbox_h)
#  define mailbox_h
#  if defined(mailbox_c)
#     if defined(__GNUG__)
#        define EXTERN extern "C"
#     else /* !__GNUG__.  */
#        define EXTERN
#     endif /* !__GNUG__.  */
#  else /* !mailbox_c.  */
#     if defined(__GNUG__)
#        define EXTERN extern "C"
#     else /* !__GNUG__.  */
#        define EXTERN extern
#     endif /* !__GNUG__.  */
#  endif /* !mailbox_c.  */


/*
 *  init - create a single mailbox which can contain a single triple.
 */
```

```
EXTERN mailbox *mailbox_init (void);


/*
 *  send - send (result, move_no, positions_explored) to the mailbox mbox.
 */

EXTERN void mailbox_send (mailbox *mbox,
  int result, int move_no, int positions_explored);


/*
 *  rec - receive (result, move_no, positions_explored) from the
 *        mailbox mbox.
 */

EXTERN void mailbox_rec (mailbox *mbox,
 int *result, int *move_no,
 int *positions_explored);

#  undef EXTERN
#endif /* !mailbox_h.  */
```

4.3 mailbox.c

```
    #define mailbox_c

#include "mailbox.h"

#define NO_MAILBOXES 30

static void *shared_memory = NULL;
static mailbox *freelist = NULL;  /* list of free mailboxes.  */


/*
 *  initialise the data structures of mailbox.  Assign prev to the
 *  mailbox prev field.
 */

static mailbox *mailbox_config (mailbox *mbox, mailbox *prev)
{
  mbox->data.result = 0;
  mbox->data.move_no = 0;
```

```
  mbox->data.positions_explored = 0;
  mbox->prev = prev;
  mbox->item_available = multiprocessor_initSem (0);
  mbox->space_available = multiprocessor_initSem (1);
  mbox->mutex = multiprocessor_initSem (1);
  return mbox;
}


/*
 *  init_memory - initialise the shared memory region once.
 *                It also initialises all mailboxes.
 */

static void init_memory (void)
{
  if (shared_memory == NULL)
    {
      mailbox *mbox;
      mailbox *prev = NULL;
      int i;
      _M2_multiprocessor_init ();
      shared_memory = multiprocessor_initSharedMemory
(NO_MAILBOXES * sizeof (mailbox));
      mbox = shared_memory;
      for (i = 0; i < NO_MAILBOXES; i++)
prev = mailbox_config (&mbox[i], prev);
      freelist = prev;
    }
}


/*
 *  init - create a single mailbox which can contain a single triple.
 */

mailbox *mailbox_init (void)
{
  mailbox *mbox;

  init_memory ();
  if (freelist == NULL)
    {
      printf ("exhausted mailboxes\n");
      exit (1);
    }
```

```
  mbox = freelist;
  freelist = freelist->prev;
  return mbox;
}


/*
 *  kill - return the mailbox to the freelist.  No process must use this
 *         mailbox.
 */

mailbox *mailbox_kill (mailbox *mbox)
{
  mbox->prev = freelist;
  freelist = mbox;
  return NULL;
}


/*
 *  send - send (result, move_no, positions_explored) to the mailbox mbox.
 */
void mailbox_send (mailbox *mbox, int result, int move_no, int positions_explored)
{
  const int BUFFER_SIZE=3;

  key_t key;
  int shmid;

  if((key = ftok(".", "S"))==-1) //return key value from file location and ID value
  {                              //"." in linux refers to the current position, S is the
    printf("Unable to make key");//if the key is reurned as -1 an error has occured
    fflush(stdout);              //sometimes printfs were getting lost so flushing to st
  }

  shmid = shmget(key, (BUFFER_SIZE + 1) * sizeof(int), 0644 | IPC_CREAT); //create share
                                                                          //and access p
  if(shmid == -1)   //if shmid returns -1 then shared memory couldn't be created or acce
  {                 //e.g. if no key was made
    printf("Unable to get sh space");     //notify user of error
    fflush(stdout);                       //ensure notification is flushed to stdout
    exit(1);                              //kill the process
  }

  int *data = (int *)shmat(shmid, (void *)0, 0);  //cast data as int so it can store int
```

```
  int *counter = data; //set first element to the counter
  *counter = 0;        //set the counter to 0

  int *buffer = data + 1; //set second element to the data
  int produced = 0; //clear produced
  int pos = 0;  //set pos to start of buffer

  //printf("C: %d    BS: %d \n", *counter, BUFFER_SIZE);
  //fflush(stdout);

  while (*counter >= BUFFER_SIZE){}; //do nothing and wait if the buffer is full

  //printf("Send - while loop \n");
  //fflush(stdout);

  buffer[pos] =  result;            //send result to start of the buffer
  pos = (pos + 1) % BUFFER_SIZE;    //increment the buffer position wrapped to the buffe

  buffer[pos] =  move_no;           //send the move number to the middle of the buffer
  pos = (pos + 1) % BUFFER_SIZE;    //increment the buffer position wrapped to the buffe

  buffer[pos] =  positions_explored;//send the number of explored positions to the end o
  pos = (pos + 1) % BUFFER_SIZE;    //increment the buffer position wrapped to the buffe

  *counter = BUFFER_SIZE;           //set the counter to show the buffer is full

  //printf("Send Called! result: %d moveN: %d PosEx: %d \n", result, move_no, positions_
  //fflush(stdout);

  //printf("send: %d \n", produced - 1);
  //fflush(stdout);

  shmdt(data);      //detach process from shared memory
  exit(0);          //end the process, it would exit in paro64bit if it returned but I ju
}



/*
 *  rec - receive (result, move_no, positions_explored) from the
 *        mailbox mbox.
 */

void mailbox_rec (mailbox *mbox, int *result, int *move_no, int *positions_explored)
{
  const int BUFFER_SIZE=3;
```

```
key_t key;
int shmid;

if((key = ftok(".", "S"))==-1)  //return key from file location and ID
{
  printf("Unable to make key"); //if -1 was returned no key was returned so notify the
  fflush(stdout); //flush to console to ensure user recieves the error
}

shmid = shmget(key, (BUFFER_SIZE + 1) * sizeof(int), 0644 | IPC_CREAT); //create share
                                                                        //and access p
if(shmid == -1) //if shmid returns -1 then shared memory couldn't be created or access
{
  printf("Unable to get sh space"); //notify user of the error
  fflush(stdout); //flush to ensure user recieves error
  exit(1);  //kill process since shared memory can't be accessed
}

int *data = (int *)shmat(shmid, (void *)0, 0);  //cast data as int so it can store int

int *counter = data;  //set first element to the counter
int *buffer = data + 1; //set second element to the data

int produced = 0;
int pos = 0;

while(1)
{
  while (*counter == 0){}; //if the buffer is empty wait until it's not

  *result = buffer[pos];  //get result from start of buffer
  pos = (pos + 1) % BUFFER_SIZE; //increment position on buffer
  *move_no = buffer[pos]; //get move number from middle of buffer
  pos = (pos + 1) % BUFFER_SIZE;  //increment position on buffer
  *positions_explored = buffer[pos];  //get number of explored positions from end of bu
  pos = (pos + 1) % BUFFER_SIZE;  //increment position on buffer

  *counter = 0; //set counter to show empty buffer

  //printf("recieved! result: %d moveN: %d PosEx: %d \n", *result, *move_no, *position
  //fflush(stdout);

  shmdt(data); //detatch from shared memory
  return(0); //return from while loop
}
```

```
  return(0); //return from function back to paro64bit
}
```

4.4 multiprocessor.h

```
    /*  multiprocessor.h provides a simple method for using semaphores.
 *  Gaius Mulley <gaius.southwales@gmail.com>.
 */

#if !defined(multiprocessor_h)
#  define multiprocessor_h
#  if defined(multiprocessor_c)
#     if defined(__GNUG__)
#        define EXTERN extern "C"
#     else /* !__GNUG__.  */
#        define EXTERN
#     endif /* !__GNUG__.  */
#  else /* !multiprocessor_c.  */
#     if defined(__GNUG__)
#        define EXTERN extern "C"
#     else /* !__GNUG__.  */
#        define EXTERN extern
#     endif /* !__GNUG__.  */
#  endif /* !multiprocessor_c.  */

#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/sysinfo.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <unistd.h>

#include <sys/stat.h>
#include <fcntl.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <semaphore.h>

/*
 *  maxProcessors - return the total number of cores available.
 */

EXTERN int multiprocessor_maxProcessors (void);
```

```
/*
 *  initSem - initialise and return a new semaphore containing value.
 */

EXTERN sem_t *multiprocessor_initSem (int value);


/*
 *  wait - block until a token is available in the semaphore.
 */

EXTERN void multiprocessor_wait (sem_t *base_sem);


/*
 *  signal - gives a single token to the semaphore.
 */

EXTERN void multiprocessor_signal (sem_t *base_sem);


/*
 *  initSharedMemory - initialise the shared memory block of memory.
 *                     mem_size determines the size of the block
 *                     the address of the free block of shared memory
 *                     is returned.  As a by product extra shared
 *                     memory is allocated which is used by the semaphore
 *                     data structures.  It allows the user to perform
 *                     a single shared memory allocation and use it for
 *                     multiple objects which is required as semaphores
 *                     also need to be placed in the shared memory region.
 */

EXTERN void *multiprocessor_initSharedMemory (unsigned int mem_size);


/* constructor for the library.  */

EXTERN void _M2_multiprocessor_init (void);


/* deconstructor for the library.  */

EXTERN void _M2_multiprocessor_finish (void);
```

```
#  undef EXTERN
#endif /* !multiprocessor_h.  */
```

4.5 multiprocessor.c

```
    #include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/sysinfo.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <unistd.h>

#include <sys/stat.h>
#include <fcntl.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <semaphore.h>


#if !defined(TRUE)
#  define TRUE (1==1)
#endif

#define MAX_SEMAPHORES  1000
static sem_t *sem_array;
static unsigned int sem_used;


/*
 *  maxProcessors - return the total number of cores available.
 */

int multiprocessor_maxProcessors (void)
{
  return get_nprocs ();
}


/*
 *  initSem - initialise and return a new semaphore containing value.
 */

sem_t *multiprocessor_initSem (int value)
```

```
{
  if (sem_used == MAX_SEMAPHORES)
    {
      printf ("MAX_SEMAPHORES has been exceeded\n");
      exit (1);
    }
  sem_t *base_sem = &sem_array[sem_used];
  sem_used++;

  /* set up semaphore.  */
  int status = sem_init (base_sem, TRUE, value);
  return base_sem;
}


/*
 *  wait - block until a token is available in the semaphore.
 */

void multiprocessor_wait (sem_t *base_sem)
{
  sem_wait (base_sem);
}


/*
 *  signal - gives a single token to the semaphore.
 */

void multiprocessor_signal (sem_t *base_sem)
{
  sem_post (base_sem);
}


void multiprocessor_killSem (sem_t *base_sem)
{
  printf ("semaphores cannot be deleted\n");
  exit (1);
}


/*
 *  initSharedMemory - initialise the shared memory block of memory.
 *                     mem_size determines the size of the block
 *                     the address of the free block of shared memory
```

```
 *                     is returned.  As a by product extra shared
 *                     memory is allocated which is used by the semaphore
 *                     data structures.  It allows the user to perform
 *                     a single shared memory allocation and use it for
 *                     multiple objects which is required as semaphores
 *                     also need to be placed in the shared memory region.
 */

void *multiprocessor_initSharedMemory (unsigned int mem_size)
{
  void *allocated;
  key_t segid = ftok (".", 'R');
  int shmid = shmget (segid, mem_size + MAX_SEMAPHORES * sizeof (sem_t), IPC_CREAT | 066
  if (shmid < 0)
    {
      printf ("shmget failed\n");
      exit (1);
    }
  sem_array = (sem_t *) shmat (shmid, (void *)0, 0);
  allocated = &sem_array[MAX_SEMAPHORES];  /* start of the memory after the semaphores.
  sem_used = 0;
  return allocated;
}


/* constructor for the module.  */

void _M2_multiprocessor_init (void)
{
}


/* deconstructor for the module.  */

void _M2_multiprocessor_finish (void)
{
  if (sem_array != NULL)
    shmdt (sem_array);
}
```

4.6 Testing log

```
    user@UniDebian:~/build-reversi$ reversi

================================
```

```
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  .  .  .
 5  .  .  .  W  B  .  .  .
 4  .  .  .  B  W  .  .  .
 3  .  .  .  .  .  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: e3
entered 20

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  .  .  .
 5  .  .  .  W  B  .  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  .  .  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 6 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of -5 after exploring 1150 positio
parent waiting for a result
 ... parent has received a result: move 1 has a score of -5 after exploring 809 position
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 767 position
I'm playing d3 which will give me a score of -5
time took 0 seconds and evaluated 2726 positions

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  .  .  .
 5  .  .  .  W  B  .  .  .
 4  .  .  .  W  B  .  .  .
 3  .  .  .  W  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: c3
```

```
entered 18

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  .  .  .
 5  .  .  .  W  B  .  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 2 has a score of -7 after exploring 13601 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -7 after exploring 26238 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -7 after exploring 22995 positi
I'm playing f5 which will give me a score of -7
time took 0 seconds and evaluated 62834 positions

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  .  .  .
 5  .  .  .  W  W  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: f6
entered 45

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  .  .  .
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .
```

```
    a  b  c  d  e  f  g  h
I'm going to look 10 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of -7 after exploring 406689 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of -5 after exploring 391917 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 686668 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -7 after exploring 1019922 posit
I'm playing f7 which will give me a score of -5
time took 9 seconds and evaluated 2505196 positions

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  W  .  .
 6  .  .  .  .  .  W  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: g7
entered 54

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  W  B  .
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 10 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of -7 after exploring 922865 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of -5 after exploring 773992 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 1780359 posit
```

```
parent waiting for a result
 ... parent has received a result: move 1 has a score of -5 after exploring 2465368 posi
I'm playing h7 which will give me a score of -5
I took 18 seconds and evaluated 5942584 positions,
sorry about the wait, I took too long so
I will reduce my search next go..

================================
 8  .  .  .  .  .  .  .  .
 7  .  .  .  .  .  W  W  W
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: h8
entered 63

================================
 8  .  .  .  .  .  .  .  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  B  .  .  .
 3  .  .  B  B  B  .  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of -7 after exploring 32631 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -7 after exploring 41126 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 95860 positi
I'm playing f3 which will give me a score of -5
time took 1 seconds and evaluated 169617 positions

================================
 8  .  .  .  .  .  .  .  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  .  .  B  .  .
```

```
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  W  .  .  .
 3  .  .  B  B  B  W  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: f8
entered 61

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  W  .  .  .
 3  .  .  B  B  B  W  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of -9 after exploring 40481 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of -7 after exploring 38095 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -9 after exploring 49480 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -9 after exploring 46137 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of -9 after exploring 93617 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -7 after exploring 135531 posit
parent waiting for a result
 ... parent has received a result: move 4 has a score of -7 after exploring 121832 posit
I'm playing b3 which will give me a score of -7
time took 1 seconds and evaluated 525173 positions

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  .  B  W  .  .  .
 3  .  W  W  W  W  W  .  .
```

```
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: f4
entered 29

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  .  B  .  .
 5  .  .  .  W  B  B  .  .
 4  .  .  .  B  B  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 4 has a score of -9 after exploring 70900 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -9 after exploring 95974 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -9 after exploring 142385 posit
parent waiting for a result
 ... parent has received a result: move 3 has a score of -9 after exploring 181195 posit
parent waiting for a result
 ... parent has received a result: move 2 has a score of -7 after exploring 331334 posit
I'm playing e6 which will give me a score of -7
time took 3 seconds and evaluated 821788 positions

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  W  B  .  .
 5  .  .  .  W  W  B  .  .
 4  .  .  .  B  W  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  .  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: b2
entered 9
```

```
================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  W  B  .  .
 5  .  .  .  W  W  B  .  .
 4  .  .  .  B  W  B  .  .
 3  .  W  B  W  W  W  .  .
 2  .  B  .  .  .  .  .  .
 1  .  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 3 has a score of -7 after exploring 133596 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of -5 after exploring 162458 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of -7 after exploring 170457 positi
parent waiting for a result
 ... parent has received a result: move 8 has a score of -5 after exploring 159781 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -3 after exploring 207605 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of -5 after exploring 238387 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -7 after exploring 246636 positi
parent waiting for a result
 ... parent has received a result: move 9 has a score of -5 after exploring 303563 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -9 after exploring 285665 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -7 after exploring 416947 positi
I'm playing a1 which will give me a score of -3
time took 5 seconds and evaluated 2325095 positions

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  W  B  .  .
 5  .  .  .  W  W  B  .  .
 4  .  .  .  W  W  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  .  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
```

```
black enter your move: c4
entered 26

================================
 8  .  .  .  .  .  B  .  B
 7  .  .  .  .  .  B  B  W
 6  .  .  .  .  B  B  .  .
 5  .  .  .  B  W  B  .  .
 4  .  .  B  B  B  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  .  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of -5 after exploring 217988 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 219397 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of -5 after exploring 217561 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of -5 after exploring 189815 positi
parent waiting for a result
 ... parent has received a result: move 8 has a score of -1 after exploring 234140 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -5 after exploring 330934 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -3 after exploring 503966 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of -3 after exploring 544332 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of -3 after exploring 834240 positi
I'm playing g8 which will give me a score of -1
time took 7 seconds and evaluated 3292373 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  .  W  B  .  .
 5  .  .  .  W  W  B  .  .
 4  .  .  W  B  B  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  .  .  .  .
 1  W  .  .  .  .  .  .  .
```

```
    a  b  c  d  e  f  g  h
black enter your move: d6
entered 43

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  B  B  B  .  .
 5  .  .  .  B  B  B  .  .
 4  .  .  W  B  B  B  .  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  .  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 3 has a score of -3 after exploring 113597 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of -3 after exploring 202240 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of -7 after exploring 193586 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -5 after exploring 198479 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of -3 after exploring 202942 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -1 after exploring 257498 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -1 after exploring 199731 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -3 after exploring 245042 positi
I'm playing g4 which will give me a score of -1
time took 3 seconds and evaluated 1613115 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  B  B  B  .  .
 5  .  .  .  B  B  B  .  .
 4  .  .  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  .  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
```

```
black enter your move: e2
entered 12

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  .  .  W  B  W
 6  .  .  .  B  B  B  .  .
 5  .  .  .  B  B  B  .  .
 4  .  .  W  W  B  W  W  .
 3  .  W  W  W  B  W  .  .
 2  .  W  .  .  B  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 2 has a score of -5 after exploring 178219 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of -1 after exploring 244770 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of -3 after exploring 321938 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of -3 after exploring 351955 positi
parent waiting for a result
 ... parent has received a result: move 1 has a score of -3 after exploring 432343 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of -3 after exploring 491227 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -3 after exploring 592285 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of -1 after exploring 571191 positi
parent waiting for a result
 ... parent has received a result: move 8 has a score of -5 after exploring 774372 positi
I'm playing d7 which will give me a score of -1
time took 9 seconds and evaluated 3958300 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  .  W  B  W  .  .
 4  .  .  W  W  B  W  W  .
 3  .  W  W  W  B  W  .  .
 2  .  W  .  .  B  .  .  .
 1  W  .  .  .  .  .  .  .
```

```
    a  b  c  d  e  f  g  h
black enter your move: c5
entered 34

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  B  B  B  W  .  .
 4  .  .  W  B  B  W  W  .
 3  .  W  W  W  B  W  .  .
 2  .  W  .  .  B  .  .  .
 1  W  .  .  .  .  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of 1 after exploring 184341 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of -3 after exploring 303651 posit
parent waiting for a result
 ... parent has received a result: move 6 has a score of -1 after exploring 309706 posit
parent waiting for a result
 ... parent has received a result: move 0 has a score of -1 after exploring 313211 posit
parent waiting for a result
 ... parent has received a result: move 5 has a score of -1 after exploring 329716 posit
parent waiting for a result
 ... parent has received a result: move 7 has a score of -3 after exploring 380867 posit
parent waiting for a result
 ... parent has received a result: move 3 has a score of -3 after exploring 487900 posit
parent waiting for a result
 ... parent has received a result: move 4 has a score of -3 after exploring 520542 posit
parent waiting for a result
 ... parent has received a result: move 8 has a score of -7 after exploring 557209 posit
I'm playing e1 which will give me a score of 1
time took 9 seconds and evaluated 3387143 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  B  B  W  W  .  .
 4  .  .  W  B  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  .  W  .  .  W  .  .  .
 1  W  .  .  .  W  .  .  .
```

```
    a  b  c  d  e  f  g  h
black enter your move: a2
entered 8

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  B  B  W  W  .  .
 4  .  .  B  B  W  W  W  .
 3  .  B  W  W  W  W  .  .
 2  B  W  .  .  W  .  .  .
 1  W  .  .  .  W  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of 5 after exploring 77413 positio
parent waiting for a result
 ... parent has received a result: move 0 has a score of 3 after exploring 145633 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of 3 after exploring 148289 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of 1 after exploring 183872 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of 3 after exploring 208150 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of -1 after exploring 222589 posit
parent waiting for a result
 ... parent has received a result: move 6 has a score of -3 after exploring 318250 posit
I'm playing b4 which will give me a score of 5
time took 3 seconds and evaluated 1304196 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  W  B  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  B  W  .  .  W  .  .  .
 1  W  .  .  .  W  .  .  .

    a  b  c  d  e  f  g  h
black enter your move: f2
```

```
entered 13

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  B  .  .
 5  .  .  W  B  W  B  .  .
 4  .  W  W  W  W  B  W  .
 3  .  W  W  W  W  B  .  .
 2  B  W  .  .  W  B  .  .
 1  W  .  .  .  W  .  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of 9 after exploring 49077 positio
parent waiting for a result
 ... parent has received a result: move 3 has a score of 7 after exploring 81528 positio
parent waiting for a result
 ... parent has received a result: move 4 has a score of 3 after exploring 89356 positio
parent waiting for a result
 ... parent has received a result: move 2 has a score of 1 after exploring 143971 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of 3 after exploring 164198 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of 1 after exploring 211421 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of 1 after exploring 219917 positi
parent waiting for a result
 ... parent has received a result: move 8 has a score of -1 after exploring 262495 posit
parent waiting for a result
 ... parent has received a result: move 1 has a score of -1 after exploring 431414 posit
I'm playing f1 which will give me a score of 9
time took 5 seconds and evaluated 1653377 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  W  .  .
 5  .  .  W  B  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  B  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
```

```
black enter your move: b5
entered 33

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  W  .  .
 5  .  B  B  B  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  B  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of 13 after exploring 12805 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of 7 after exploring 55491 positio
parent waiting for a result
 ... parent has received a result: move 4 has a score of 7 after exploring 75885 positio
parent waiting for a result
 ... parent has received a result: move 2 has a score of 7 after exploring 75059 positio
parent waiting for a result
 ... parent has received a result: move 0 has a score of 9 after exploring 116617 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of 5 after exploring 119079 positi
parent waiting for a result
 ... parent has received a result: move 6 has a score of 3 after exploring 152522 positi
I'm playing a5 which will give me a score of 13
time took 2 seconds and evaluated 607458 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  .  W  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  B  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: d2
entered 11
illegal move
```

```
black enter your move: e7
entered 52

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  B  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  .  W  W  W  W  W  .  .
 2  B  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of 13 after exploring 10013 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of 13 after exploring 14254 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of 9 after exploring 19610 positio
parent waiting for a result
 ... parent has received a result: move 1 has a score of 5 after exploring 34415 positio
I'm playing a3 which will give me a score of 13
time took 0 seconds and evaluated 78292 positions

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  .  W  B  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  W  W  W  W  W  W  .  .
 2  W  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: c7
entered 50

================================
 8  .  .  .  .  .  B  W  B
 7  .  .  B  B  B  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  W  W  .  .
 4  .  W  W  W  W  W  W  .
```

```
 3  W  W  W  W  W  W  .  .
 2  W  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 10 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of 5 after exploring 49627 positio
parent waiting for a result
 ... parent has received a result: move 3 has a score of 9 after exploring 363856 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of 9 after exploring 479872 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of 9 after exploring 481551 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of 3 after exploring 1210004 posit
parent waiting for a result
 ... parent has received a result: move 0 has a score of 5 after exploring 1557227 posit
I'm playing c8 which will give me a score of 9
time took 10 seconds and evaluated 4142137 positions

================================
 8  .  .  W  .  .  B  W  B
 7  .  .  B  W  B  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  W  W  .  .
 4  .  W  W  W  W  W  W  .
 3  W  W  W  W  W  W  .  .
 2  W  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: g3
entered 22

================================
 8  .  .  W  .  .  B  W  B
 7  .  .  B  W  B  B  B  W
 6  .  .  .  B  W  W  .  .
 5  W  W  W  W  B  W  .  .
 4  .  W  W  W  W  B  W  .
 3  W  W  W  W  W  W  B  .
 2  W  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
```

```
I'm going to look 10 moves ahead...
parent waiting for a result
 ... parent has received a result: move 1 has a score of 5 after exploring 598964 positi
parent waiting for a result
 ... parent has received a result: move 7 has a score of 5 after exploring 706809 positi
parent waiting for a result
 ... parent has received a result: move 3 has a score of 3 after exploring 820731 positi
parent waiting for a result
 ... parent has received a result: move 2 has a score of 5 after exploring 729784 positi
parent waiting for a result
 ... parent has received a result: move 0 has a score of 3 after exploring 842567 positi
parent waiting for a result
 ... parent has received a result: move 4 has a score of 1 after exploring 903279 positi
parent waiting for a result
 ... parent has received a result: move 8 has a score of -1 after exploring 876650 posit
parent waiting for a result
 ... parent has received a result: move 6 has a score of 3 after exploring 1081028 posit
parent waiting for a result
 ... parent has received a result: move 5 has a score of 3 after exploring 1553428 posit
I'm playing h3 which will give me a score of 5
I took 18 seconds and evaluated 8113240 positions,
sorry about the wait, I took too long so
I will reduce my search next go..

================================
 8  .  .  W  .  .  B  W  B
 7  .  .  B  W  B  B  B  W
 6  .  .  .  B  W  W  .  .
 5  W  W  W  W  B  W  .  .
 4  .  W  W  W  W  B  W  .
 3  W  W  W  W  W  W  W  W
 2  W  W  .  .  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: c2
entered 10
illegal move
black enter your move: d2
entered 11

================================
 8  .  .  W  .  .  B  W  B
 7  .  .  B  W  B  B  B  W
 6  .  .  .  B  W  W  .  .
 5  W  W  W  B  B  W  .  .
```

```
 4  .  W  W  B  W  B  W  .
 3  W  W  W  B  B  W  W  W
 2  W  W  .  B  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 7 has a score of 5 after exploring 53390 positio
parent waiting for a result
 ... parent has received a result: move 0 has a score of 7 after exploring 37747 positio
parent waiting for a result
 ... parent has received a result: move 4 has a score of 7 after exploring 38656 positio
parent waiting for a result
 ... parent has received a result: move 6 has a score of 9 after exploring 56518 positio
parent waiting for a result
 ... parent has received a result: move 2 has a score of 5 after exploring 48234 positio
parent waiting for a result
 ... parent has received a result: move 3 has a score of 5 after exploring 59623 positio
parent waiting for a result
 ... parent has received a result: move 5 has a score of 9 after exploring 63075 positio
parent waiting for a result
 ... parent has received a result: move 1 has a score of 7 after exploring 89349 positio
I'm playing d8 which will give me a score of 9
time took 1 seconds and evaluated 446592 positions

================================
 8  .  .  W  W  .  B  W  B
 7  .  .  B  W  W  B  B  W
 6  .  .  .  B  W  W  .  .
 5  W  W  W  B  B  W  .  .
 4  .  W  W  B  W  B  W  .
 3  W  W  W  B  B  W  W  W
 2  W  W  .  B  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: a4
entered 24

================================
 8  .  .  W  W  .  B  W  B
 7  .  .  B  W  W  B  B  W
 6  .  .  .  B  W  W  .  .
 5  W  W  W  B  B  W  .  .
 4  B  B  B  B  W  B  W  .
```

```
 3  W  W  W  B  B  W  W  W
 2  W  W  .  B  W  W  .  .
 1  W  .  .  .  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 0 has a score of 5 after exploring 47981 positio
parent waiting for a result
 ... parent has received a result: move 2 has a score of 3 after exploring 48536 positio
parent waiting for a result
 ... parent has received a result: move 4 has a score of 3 after exploring 54530 positio
parent waiting for a result
 ... parent has received a result: move 3 has a score of 5 after exploring 50342 positio
parent waiting for a result
 ... parent has received a result: move 5 has a score of 5 after exploring 55158 positio
parent waiting for a result
 ... parent has received a result: move 1 has a score of 1 after exploring 82183 positio
parent waiting for a result
 ... parent has received a result: move 6 has a score of 1 after exploring 67719 positio
parent waiting for a result
 ... parent has received a result: move 7 has a score of -3 after exploring 90569 positi
I'm playing d1 which will give me a score of 5
time took 1 seconds and evaluated 497018 positions

================================
 8  .  .  W  W  .  B  W  B
 7  .  .  B  W  W  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  B  W  .  .
 4  B  B  B  W  W  B  W  .
 3  W  W  W  W  B  W  W  W
 2  W  W  .  W  W  W  .  .
 1  W  .  .  W  W  W  .  .

    a  b  c  d  e  f  g  h
black enter your move: b1
entered 1

================================
 8  .  .  W  W  .  B  W  B
 7  .  .  B  W  W  B  B  W
 6  .  .  .  W  W  W  .  .
 5  W  W  W  W  B  W  .  .
 4  B  B  B  W  W  B  W  .
 3  W  B  W  W  B  W  W  W
```

```
 2  W  B  .  W  W  W  .  .
 1  W  B  .  W  W  W  .  .

    a  b  c  d  e  f  g  h
I'm going to look 8 moves ahead...
parent waiting for a result
 ... parent has received a result: move 4 has a score of 3 after exploring 17038 positio
parent waiting for a result
 ... parent has received a result: move 3 has a score of 3 after exploring 19529 positio
parent waiting for a result
 ... parent has received a result: move 7 has a score of 1 after exploring 19530 positio
parent waiting for a result
 ... parent has received a result: move 6 has a score of 3 after exploring 21780 positio
parent waiting for a result
 ... parent has received a result: move 1 has a score of 1 after exploring 27810 positio
parent waiting for a result
 ... parent has received a result: move 8 has a score of -3 after exploring 31370 positi
parent waiting for a result
 ... parent has received a result: move 5 has a score of 3 after exploring 29252 positio
parent waiting for a result
 ... parent has received a result: move 0 has a score of 5 after exploring 50245 positio
parent waiting for a result
^C
```

at this point, the program started taking a really long time between turns so I killed the program