Wolpers

Fussen Alvaro

COLLABORATORS			
	TITLE : Wolpers		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Fussen Alvaro	October 17, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Wolpers **1**

1.1 Wolpers Help . . . 1
1.2 Introduction . . . 2
1.3 Copyright . . . 2
1.4 Author . . . 3
1.5 Requirements . . . 3
1.6 Installation . . . 4
1.7 Usage . . . 4
1.8 WOP . . . 5
1.9 WOP.0 . . . 6
1.10 WOP.1 . . . 7
1.11 WOP.2 . . . 8
1.12 Othello . . . 9
1.13 Features . . . 10
1.14 History . . . 11
1.15 Graphical user interface . . . 11
1.16 Problems . . . 12
1.17 TODO . . . 13
1.18 Credits . . . 13

Chapter 1

Wolpers

1.1 Wolpers Help

```
                                        Wolpers V1.00

                        The Othello player
                      ENGINE version 2.44.94
                         by Fussen Alvaro

 1
                  ~Introduction~~~
                     -   Just another Othello player?
 2
                  ~Copyright~~~~~~
                     -   Legal mumbo-jumbo.
 3
                  ~Author~~   ~~~~
                     -   Questions? Here's my address.
 4
                  ~Requirements~~~
                     -   Minimum Hard- and Software.
 5
                  ~Installation~~~
                     -   How to install Wolpers.

 6
                  ~Usage ~
                     -   To prevent a FAQ for Wolpers.
 7
                  ~About Othello ~
                     -   How to survive the game.
 8
                  ~Technical info~
                     -   What's special?
 9
                  ~History~~~~~~~~
                     -   What's new in this version.
10
                  ~MUI~~~~~~~~~~~~
                     -   Important notes to the GUI routines.
11
```

~Problems~~~~~~~~
   -   If Wolpers exceeds the time limit.
12
~To do~ ~~~~~
   -   What will be done in the near future.
13
~Credits~~~~~~~~
   -   Persons who made Wolpers possible.

D O N T   P A N I C !

1.2 Introduction

1. Introduction

Wolpers is an Othello (Reversi) player. The big version of this
program is actually installed on the internet Othello server <ios>
at faust.uni-paderborn.de (port 5000), and has played there about
5000 games.
The Amiga version is a little bit smaller (no learning routines,
shallow searches...), but should beat every silicon player
available on this computer at the moment.
To show you the quality of the evaluation routine, I tested three
versions of Wolpers with following settings on ios:

- 1-ply version, Risk factor low : 1300/1304 pts
- 3-ply version, Risk factor high: 1550/1600 pts
- 5-ply version, Risk factor high: 2003/1960 pts

Note: The best human players are rated at 2070 pts, the 6-ply
version of the World's best program LOGISTELLO is at 1980/2056.
(The full version of log has 2600+ pts. This program is a
miracle.) (February 1995)

1.3 Copyright

2. Copyright

```
** package remain unchanged.                                        **
** Dont make money with Wolpers without my permission.             **
*********************************************************************
*********************************************************************
```

This program is shareware, it is *not* public domain. This means that if you use this program regulary, you should send

me
10 Sfr.

(10DM or 10US Dollars) to become a registered user. This version has NO LIMITATIONS, I dont like it to cripple my program. So you wont get any keyfile. But you will get a better version as soon as it is ready. Please give me also your email address. I will send you a better version as soon as it is ready. If you dont have an email address, please send me a disk. (DD or HD)

Oh, by the way: I dont think that the user interface will change dramatically in the near future. At the moment I'm more interested in improving the strenght, since this program will probably participate the World Championship 95. The Amiga version will soon have the learning routines enabled. (I disabled them because they need to be supervised at the moment.)

1.4 Author

3. Author

In case you want to register or have problems, suggestions, questions (related to this program of course, I'm not from the Dr. Summer team) you can write me to the following addresses:

Fussen Alvaro
Stockmattenweg 7
CH-3930 Visp/VS

email: fussen@satan.vmsmail.ethz.ch

But I have to warn you: If you want to ask me questions about Othello, you shouldnt expect too much. I'm a lousy player, I sometimes even lose against my program at 1 ply.

1.5 Requirements

4. Requirements

The minimum configuration for the usage of Wolpers consists of any AMIGA computer that runs at least AMIGA-OS 2.0 and has 1 MB RAM.

Additionally you need to have

MUI
V2.3 installed on your System.
MUI is distributed as shareware. To obtain a complete package please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.
A harddisk with at least 200 kb of free space is recommended.

1.6 Installation

5. Installation

Simply extract the archive file. Oh well, you obviously have already done that :-)
You need about 200 kb of free space on your disk media. Note that Wolpers looks into its current directory for a 'data' subdirectory where all the learn files should be located.
IMPORTANT NOTE: Since all learn abilities in this version are disabled, Wolpers cannot create new learnfiles and it wont work properly if it cannot find its original files. So dont delete them. (In a future release deleting learnfiles would make sense if you want to observe the learning process.)

1.7 Usage

6. Usage

Well, I think the usage of this program is quite intuitive. So I'll only explain the more exotic possibilities and features...

'Strange' buttons:

<- swap -> - This button swaps all settings for side A with those of side B. If you have the 'Autoswap' checkmark checked, Wolpers will call 'swap' every time you click on the 'New' button.

About the checkmarks:

Evaluation

/book - Wolpers has an opening library of 1171 sequences with 23420 positions. Due to the symmetries, these lines have been normalized to the move 'd3'.
/Mobility - Wolpers tries to maximize four different aspects of mobility: The current mobility and three different features which try to get an idea of the potential

```
              mobility. (*)
/Learned p. - Wolpers has lots of precalculated tables (eg for edge
              configurations) and learned position tables. You can
              turn this feature off. (*)
/play t win - It sometimes makes fun to try to wipe Wolpers out with
              this feature enabled.

Settings

/Autoswap   - Automatically Call the 'swap' function when the 'New'
              button has been clicked.
/Hashtables - Speeds up the searching routines and makes the
              ab-algorithm much more effizient. (*)
/Safety     - If checked, Wolpers saves the whole game after each move.
/Server     - See
                 WOP
                 for more details.


About the Input fields

Settings

/Number Crunch Pri.
            - Wolpers calculation task switches back to this
              priority when it starts the search.
/INPIPE .. OUTPIPE
            - See
                 WOP
                 for more details.

(*) If these checkmarks are not checked, the strenght of Wolpers will
be decreased significantly.
```

1.8 WOP

```
               6.1 WOP


WOP - Wolpers' Othello protocol

I think, this is a very nice feature.
You can use Wolpers as a front end for another program which has
to follow Wolpers' Othello Protocol guidelines. It also allows
you to let that program play automatically against Wolpers.

How to use Wolpers as a
    -
               front end
               .
    -
               sparing partner
                for another program.
```

If you have to use the PIPE: device, click
here
for more
information.

1.9 WOP.0

6.1.1 WOP - Wolpers as a front end.

To show you, how to use Wolpers as a front end, I'll make an
example with funkylady (an othello program, available on the
AmiNet):

Set the outpipe/inpipe string according to the following example:

```
outpipe: pipe:Wolpers
inpipe:  pipe:Opponent
```

Switch on the Server ability of Wolpers. Switch off the Computer
for both sides. Press the 'New' button on the first page of
Wolpers' GUI. Then execute one of the following lines, depending
whether you want to be white or black:

If you want to play with white, type:
funkylady -ccolor black -pipes pipe:Wolpers pipe:Opponent

If you want to play with black, type:
funkylady -ccolor white -pipes pipe:Wolpers pipe:Opponent

That's it.
If something goes wrong, there's probably a trouble with the
pipes. You should then issue the following command:

```
type pipe:Wolpers
```

to clear the 'inpipe' for funkylady. You dont have to clear
Wolpers' inpipe. If the type process hangs after that command,
issue the following command:

```
echo >pipe:Wolpers garbage
```

to bring the waiting type process back to life.
Yes, this all sounds a bit tricky. It ius an old 'PIPE:' problem.

If you want to use Wolpers as a front end for your own program,
look at
Wolpers as a sparing partner
.

1.10 WOP.1

6.1.2 WOP - Wolpers as a sparing partner.

If you have written an Othello player, you may find this chapter
very interesting.

Wolpers has its own protocol that allows your program to
transfer/receive moves to/from it. I suppose your program has
somewhere a line like this one...

```
score = alphabeta(depth, alpha, beta, &move);
```

...to start the recursive search. ('move' contains the move of
your program.)

Now simply add these lines somewhere after that statement:

```
    if(SERVER && myturn) /* Do I have to send the moves? */
    {
        FILE *file;
        char string[4];

        /* Convert internal move format into the official one,
           eg: 43 -> d3 */
        if(!move) sprintf(string, "ps"); /* I have to pass */
        else sprintf(string, "%c%d ", 'a'+move%10-1, (move-move%10+1)/10);

        for(;;)
        {
            if(!(file = fopen(OUTPIPE, "w+"))) { Delay(10); continue; }
            if(fwrite(&string[0], 2, 1, file) != 1) { Delay(10); continue; }
          fclose(file); break;
        }
    }
```

This will store your move into a file where Wolpers will fetch it.

Of course this isnt enough. To give your program the ability to
answer to Wolpers' moves, include these lines into your program.
Make sure that they are executed instead of your 'human'-input
interface.

```
    if(SERVER) /* Do I have to look into a file to get moves? */
    {
        FILE *file;
        char getmove;

        for(;;)
        {
          if(!(file = fopen(INPIPE, "r"))) { Delay(10); continue; }
            /* Convert things like 'd3' into '43'. Change this if your
               program uses another internal format */
            if(fread(&getmove, 1, 1, file) == 1) move = (1+getmove-'a');
            if(fread(&getmove, 1, 1, file) == 1) move += 10*(getmove-'0');
```

```
        fclose(file);
            /* This is an ugly method to find out, whether the last
               move was a pass. The other instance is supposed to write
               'ps' into the pipe when it passed... */
          if((getmove=='s') && 0==generateMoves(Othello))
            {
                doMove(0); /* Make no move */
                break;
            }
            else
            {
                if(checkPosition(move)==TRUE) doMove(move);
                else
                {
                   printf("Something went wrong. :-)\n");
                   exit(1);
                }
            }
        }
    }
```

Since you are coding an Othello program, you shouldn't have problems to understand these lines, I think.

You will probably have to adjust my server code to your program. It is important that you always execute these lines after your program has calculated a move and when your program waits for an answer. Also send your move if it was only a pass.

You can either use the PIPE device (which has been created for this primitive kind of IPC (inter process communication)) or you can also use normal AmigaDOS devices like RAM: or DH0: etc. (Just set INLINE to "ram:me" and OUTLINE to "ram:Wolpers" for example.)

I hate pipes for IPC. If you like them, you can of course simplify the code a little bit. The 'retry' code would vanish.

By the way: Funkylady works together with 'WOP'. You just have to set an INPIPE for funkylady (which has to be the same path and file as OUTPIPE of Wolpers) and an OUTIPE (which has to be the same path as INPIPE of Wolpers).

Yes, I know: This sounds all a bit tricky. But it is in fact very easy once you get the idea. Always remember: DONT' PANIC.

In case of troubles, contact
~me~
.

1.11 WOP.2

6.1.3 PIPE: problems

As I mentioned in "
sparing partner
", I hate pipes.
This is because an fopen() wont return an error if it cannot open a file. It just waits until it can do so. Now the task hangs around and you cannot even change prameters or do something else with that task. And worse: If there's an error with the other instance, one must issue an 'echo >pipe:INPIPE garbage' to bring the waiting task back to life.
I tried access() and dfind() to find out whether there's an answer in the PIPE:, but of course it didnt work since pipes arent ordinary AmigaDos devices.
So I made a terrible hack to solve that problem. If anybody knows a good method how to find out whether there's something in the PIPE:, please contact me. (I really want to get rid of that ugly, terrific and anoying hack in my beautiful code.)

1.12 Othello

7. About Othello

Moves are made by placing your disc so that it outflanks one or more of the opponent's discs. One or more opponent's pieces must lie between the new piece and another of your pieces, diagonally, horizontally, or vertically. Outflanking can occur in more than one direction. The outflanked pieces are flipped, and become discs of your color. A game ends when all positions are filled, or when neither side can move. The side with the most pieces on the board at the end of a game wins.

It sometimes occurs that you do not have any legal moves. If this happens, you must forfeit. You can do this by pressing the 'Pass' button.

Some hints: Dont capture too much discs at the beginning of the game. Try to force your opponent to place his discs around your discs. This will give you a high mobility. X squares (b2, b7, g2, g7) are dangerous, dont place your pieces there too early, because this would give your opponent access to the corners. If you play against Wolpers, you will see that he often makes X-moves. Before taking the corner then, you should think twice: such corners are most often 'poisoned'.
The opening phase in Othello is probably the most important phase. Dont just put your discs on the board. Observe your opponent and try to learn opening sequences.
And never forget: The goal of the game is to have more own discs on the board than the opponent AT THE END OF THE GAME. Wolpers has been made to fulfill that goal. Wolpers doesnt aim a 63-1 victory. He is also very happy with a 34-30 score.
Dont think that Othello is just a simple little game. The more you learn about it, the more you will see how wrong you are.

1.13 Features

```
8. Features


Wolpers' <ios> version, technical info:
---------------------------------------

Engine:          - V2.44.94, fast, stable, fully self-learning. I'm
                   proud.

Search engine:   - Alpha-Beta negamax with iterative deepening
                 - Shannon B algorithm
Move ordering:   (Listed by priority)
                 - Move hashtable for primary and secondary sort
                     priorities
                 - Response killer tables
                 - History heuristics
                 - Learn adaptive phase dependent move sequences
Features:        - Current mobility (approximative value)
                 - Potential mobility (approximative value)
                 - Learned patterns weighted by learned
                     coefficients. 405'324 Bytes table.
                 - Precalculated edge table (will be removed soon)
F' combination:  - linear
                 - 64 coefficients for each feature depending
                     on the game phase
                 - coefficients learned from lost games and
                     adjusted with linear regression
                     (will be replaced by logistic regression soon)
Independency:    - Disc difference. (ENGINE 3: Win probability)
Search speed:    - Midgame up to 15'000 positions/sec (RS/6000)
                 - Endgame up to 150'000 positions/sec (RS/6000)
                 - Amiga 68030/40MHz ~500/10000
Search depth:    - 9-14 halvemoves (selective, without quience search)
                     in a 15 min. match (On fast computers)
Book:            - Automatically updated and optimized (DISABLED ON AMIGA!!!)
Misc:            - Interface to socket routines to communicate via
                     internet with IOS Paderborn (DISABLED ON AMIGA!!!)
                 - Graphical user interface (on Amiga computers)
Language:        - C, of course.
                 - Compiler: GNU's GCC
Hardware:        - When running on IOS: IBM RS/6000 powerPC
                     processor.
                 - Switches to a slower sparc classic when there's
                     too heavy load on the RS/6000
                 - When developing: Amiga 500, GVP 68030/50
                     accelerator. 3MB Ram :-(
IOS rating:      - 2043 ios-pts
                 - 2006 ios-pts with 5-Ply!!! (winter 94/95)
                 - ~1600 ios-pts with 3-Ply.
 "  ranking:     - 4. place (winter 94/95)

IOS Login name: - MrOth, Jerry, wolpers, mirco, :-).

note for ios users:
```

```
With the login name ':-)' Wolpers plays with risk factor high and
thinks only 5 plys. The highest ranking of this program was
the 2. place on ios in may 95.
```

1.14 History

```
               9. History


Up to now I have written three different engines (the part of the
program that builds up the search tree and evaluates the leaves).

ENGINE 1 (my first try)

- Coded in one week.  Features:  Corner penalty, current mobility,
Edge spot penalty, minimal disk strategy, four game phases.  This
version was already able to beat Whitelion and Othellokiller.
Whitelions most significant weakness:  The programmer Martin Grote
assumed that the possesion of edges is good.  But there are more
dangerous edge configurations than safe and advantageous edge
patterns.

ENGINE 2 (the engine included in this version)

- see
               ~Technical info
               ENGINE 3 (will be available in the near future)

- Totally rewritten. New learning routines, nearly no intuitive
assumptions. This version is still on its test phase.


Wolpers' version Numbers:

    V0.153  Secret version. Installed on <ios>.
    V0.99   Pre release version. A day before I wanted to upload Wolpers,
            I discovered a bug in the engame solver routines.
            This is now fixed. That nasty bug also slowed down my solver.
            Shannon B Algorithm improved. It prunned sometimes too much.
            Mobility calculation improved: Only the a2-g7 square is
            considered now.
            New functions: Autoswap, Sound, removable safety...
            Server ability improved -> it works now together with
            funkylady.
    V1.00   Release version.
            Nice Coordinates.
            Autoswap is switched off by default.
            Improved clocks.
```

1.15 Graphical user interface

10. Graphical user interface

#include "MUI.readme"

MUI-Copyright:

This application uses

MUI - MagicUserInterface

(c) Copyright 1993/94 by Stefan Stuntz

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY

1.16 Problems

11. Problems

Since Wolpers has been written to run on a lot of different computers, it wasnt easy to tell to the program when it has to solve. I therefore decided to make a learnfile for this purpose. The 'data/times.mrc' file for Amiga computers has been trained on a 68030/50 Mhz system. If you have a slower machine, Wolpers will solve too early the first times you play against it. But I think, after 5-10 games, Wolpers will have learned that it is running on a slower machine.

So dont be upset: If Wolpers is solving for an hour... he will surely come back and will never do this again. (Until you delete the 'data/times.mrc' file.

If you have a faster machine than I have (68030/50), I suggest you deleting that learnfile. Wolpers will create a new one that fits better to your computer's speed.

1.17 TODO

12. To do

- Better engine. Im about to rewrite the whole engine code. As soon as ENGINE 3 is finished and tested on <ios>, I'll put Wolper's GUI around it.
- New opening routines. I dont like the openings that this version has learned. (In fact, it is the weakest point of this version.)
- Speed up the whole program. Wolpers' big problem is that it isnt fast enough. Only the engame solver is quite good at this point. (Yes, it's one of the fastest solver.)
- At the moment, I'm coding a 'Connection Four'. I dont know why, but I've never seen a strong CF player on the Amiga. If you are interested in such a program, write me.
- Oh well. I'd like to write a Chess player for the Amiga. My old chess player 'Chess Imperator' on the C64 has never been released. I hope that someday I can upload my program onto the Aminet, if it still exists then. (I mean the Aminet, not my program. :-)
- Fondle my cat.

1.18 Credits

13. Credits

I would like to thank:

Ralph Brunner who suggested making an Othello player. Lots of ideas in this program come from long talks with him. I'm waiting for Wigald's revenge.

Igor Durdanovic who wrote <ios> and gave me the oportunity to test my program against human and silicon players from everywhere.

Stefan Odendahl who found for me a cheap 68030 accelerator to turn my old A500/68000-game console into a (quite) "'modern' developer" machine. :-)

Furthermore, I'd like to thank the following people for their programs, which made the GUI of Wolpers possible:

Stefan Stuntz for MUI. If you want to learn more about MUI, see

MUI
.

Eric Totel for MUIBuilder. This program actually persuaded me to make a GUI for Wolpers.

Many thanks to my betatesters...

Stefan Odendahl for his suggestions. He helped me to made Wolpers more usable. There are lots of improvements due to some long discussions with him.

Frank Petzold who discovered some conditions in which Wolpers used to crash. The engine runs very stable, it has never crashed, but I had problems with the GUI. He also suggested some of the feature in the 'settings'-Page.