Package 'bigchess'

May 15, 2019

Type Package

Title Read, Manipulate and Explore Chess PGN Files

Version 1.3.2

Author Wojciech Rosa

Maintainer Wojciech Rosa <w.rosa@pollub.pl>

Description Provides functions for reading *.PGN files with more than one game, including large files without copying it into RAM (using 'ff' package or 'RSQLite' package). Handle chess data and chess aggregated data, count figure moves statistics, create player profile, plot winning chances, browse openings.

License GPL-3

Imports ff,ffbase,RSQLite

Suggests rjson

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-15 09:20:10 UTC

R topics documented:

`browse_eco_opening` *Browse ECO opening*

Description

Browse ECO opening winning and drawing percentages by table and barplot

Usage

```
browse_eco_opening(df, topn = 0)
```

Arguments

`df`	data frame with imported chess games from read.pgn() function.
`topn`	integer, default is 0, passed to tree_eco function (indicating how many top openings should be included).

Value

Data frame from tree_eco function and plot from plot_tree_eco function.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,ignore.other.games = TRUE,stat.moves = FALSE, add.tags = "ECO")
# Analyze 20 best ECO Kasparov openings:
bo <- browse_eco_opening(subset(df,grepl("Kasparov",White)),20)
```

`browse_opening` *Browse opening*

Description

Browse opening winning and drawing percentages by table and barplot

Usage

```
browse_opening(df, movetext = "")
```

Arguments

df	data frame with imported chess games from read.pgn() function.
movetext	movetext string, default is "" means browse first move for White. The standard English values are required: pawn = "P" (often not used), knight = "N", bishop = "B", rook = "R", queen = "Q", and king = "K".

Value

Data frame from tree_move function and plot from plot_tree_move function.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,ignore.other.games = TRUE,stat.moves = FALSE)
# Analyze best Kasparov openings:
bo <- browse_opening(subset(df,grepl("Kasparov",White)))
# Analyze 'best' answer to Kasparov Ruy Lopez:
bo <- browse_opening(subset(df,grepl("Kasparov",White)),"1.e4 e5 2.Nf3 Nc6 3.Bb5")
# Analyze best answer to "1.e4 e5 2.Nf3" in aggregated data
browse_opening(FirstTwoMoves,"1.e4 e5 2.Nf3")
```

---

extract_moves *Extract first N moves*

Description

Extract first N moves from pgn movetext into data frame

Usage

```
extract_moves(movetext, N = 10, last.move = T)
```

Arguments

movetext	movetext string (or string vector). The standard English values are required: pawn = "P" (often not used), knight = "N", bishop = "B", rook = "R", queen = "Q", and king = "K".
N	integer (default 10) determines how many first N moves will be extracted. Default is 10, should be greater than 0.
last.move	boolean (default TRUE) indicating whether to calculate the last move

Value

Data frame containing first N moves for white and for black, named as W1, B1, W2 and so on, up to WN and BN (where N is input argument). If N is greater than total moves number then NA's generated. Column complete.movetext flag is indicating if movetext string begin with "1.'move'".

Examples

```
extract_moves("1. e4 e5 2. Nf3 Nf5 3. d5 ",N = 3)
# e4 e5 Nf3 Nf5 d5 NA TRUE
extract_moves("1. e4 e5 2. Nf3 Nf5 3. d5 ",N = 3, last.move = TRUE)
# e4 e5 Nf3 Nf5 d5 NA d5 TRUE
```

FirstTwoMoves *Example dataset*

Description

A dataset containing 10,894 results after first two moves in 2,395,869 high-quality chess games played over the board by players with ELO > 2000. Source data OTB-HQ.7z downloaded from: https://sourceforge.net/projects/codekiddy-chess/ and converted to PGN in SCID software.

- Result:
- W1: White first move
- B1: Black first move
- W2: White second move
- B2: Black second move
- Freq: Number of games played in database

Usage

```
data(FirstTwoMoves)
```

Format

A data frame with popular positions in classic chess

---

n_moves *Compute number of moves*

Description

Compute total number of moves given movetext string (or string vector)

Usage

```
n_moves(movetext)
```

Arguments

movetext	movetext string (or string vector)

Value

n integer (or integer vector)

Examples

```
n_moves(c("1. e4 e5 2. Nf3 Nf5 3. d5 ","1. d4 d5"))
# 3 1
```

---

player_profile *Compute player profile*

Description

Computes players profile from data frame obtained from read.pgn() function into data frame

Usage

```
player_profile(df, player)
```

Arguments

df	data frame from read.pgn or read.pgn.ff files with stats computed.
player	string used in grepl(player,White) and grepl(player,Black)

Value

Data frame with player (column prefix P_) and opponent (column prefix O_) figure move counts. Column Player_Col indicating pieces colour for player (factor White or Black). Example column P_Q_moves means number of player Queen moves count.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,ignore.other.games = TRUE)
nrow(df) # 2109
df_pp <- player_profile(df,"Kasparov, Gary")
nrow(df_pp) # 1563
df_pp <- player_profile(df,"Kasparov,G")
nrow(df_pp) # 543
df_pp <- player_profile(df,"Kasparov, G\\.")
nrow(df_pp) # 2
df_pp <- player_profile(df,"Kasparov")
nrow(df_pp) # 2109 - correct
boxplot(P_Q_moves/NMoves~Player_Col,df_pp,
main = "Average Queen Moves\n Kasparov as Black (909 games) vs Kasparov as White (1200 games)",
col = c("black","white"),border = c("black","black"),notch = TRUE)
# Magnus Carlsen data example
```

```
f <- system.file("extdata", "Carlsen.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,ignore.other.games = TRUE)
nrow(df) # 2410
df_pp <- player_profile(df,"Carlsen")
nrow(df_pp) # 2411 - ??
# One game was played by Carlsen,H
df_pp <- player_profile(df,"Carlsen,M")
nrow(df_pp) # 2410 - correct
```

`plot_tree_eco` *Plot tree for a given tree ECO table*

Description

Plot tree (barplot percentages) for a given tree ECO data frame.

Usage

```
plot_tree_eco(tr, main = "", add.lines = T, add.labels = T)
```

Arguments

`tr`	data frame containg tree ECO
`main`	string for main title, default is ""
`add.lines`	boolean (default TRUE) add weighted mean lines?
`add.labels`	boolean (default TRUE) add labels?

Value

Barplot with white scores, draws percent and black scores.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,stat.moves = FALSE, add.tags = "ECO")
tr <- tree_eco(subset(df,W1=="e4"),20)
plot_tree_eco(tr,"1. e4 ... ?")
```

plot_tree_move *Plot tree for a given tree move table*

Description

Plot tree (barplot percentages) for a given tree move data frame.

Usage

```
plot_tree_move(tr, main = "", add.lines = T, add.labels = T)
```

Arguments

tr	data frame containg tree move
main	string for main title, default is ""
add.lines	boolean (default TRUE) add weighted mean lines?
add.labels	boolean (default TRUE) add labels?

Value

Barplot with white scores, draws percent and black scores.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,stat.moves = FALSE)
tr <- tree_move(subset(df,W1=="e4"),"B1")
plot_tree_move(tr,"1. e4 ... ?")
# Plot tree move openings in aggregated data
tr <- tree_move(FirstTwoMoves,"W1")
plot_tree_move(tr,paste0("1. ... ?\n",sum(FirstTwoMoves$Freq)," total games"))
```

read.pgn *Reads PGN files into data frame*

Description

Reads PGN files into data frame

Usage

```
read.pgn(con, add.tags = NULL, n.moves = T, extract.moves = 10,
  last.move = T, stat.moves = T, big.mode = F, quiet = F,
  ignore.other.games = F, source.movetext = F)
```

Arguments

con	connection argument passed directly to readLines() function. String - the name of the file which the data are to be read from or connection object or URL.
add.tags	string vector containing additional tags to be parsed. According to Seven Tag Roster rule: http://www.saremba.de/chessgml/standards/pgn/pgn-complete.htm#c8.1.1 The STR tag pairs appear before any other tag pairs: "Event", "Site", "Date", "Round", "White", "Black" and "Result". Using this argument you can specify supplemental tag names, such as: Player related information, Event related information, Opening information (locale specific), Opening information (third party vendors), Time and date related information, Time control, Alternative starting positions, Game conclusion and Miscellaneous. Most popular: "WhiteElo", "BlackElo","ECO","SetUp" or "FEN". Case sensitive.
n.moves	boolean (default TRUE), compute number of moves?
extract.moves	integer (default 10) passed to extract_moves function. Additionaly value -1 will extract all moves from movetext (not recommended for big files). Value 0 means that moves will not be extracted.
last.move	boolean (default TRUE) passed to extract_moves, ignored when extract.moves = 0
stat.moves	boolean (default TRUE), compute moves count statistics? Could take a long time for big file.
big.mode	boolean (default FALSE) used in read.pgn.ff function
quiet	boolean (default FALSE), indicating if messages should appear.
ignore.other.games	boolean (default FALSE) if TRUE result is subset of original dataset without games with result marked as "*", i.e. ongoing games
source.movetext	boolean (default FALSE, experimental!) if TRUE column with original movetext will be added

Value

Data frame containg STR, additional tags (conditionally), Movetext, NMoves (conditionally), extracted moves (conditionally) with complete.movetext flag, figure moves count statistics (conditionally).

Examples

```
f <- system.file("extdata", "2016_Candidates.pgn", package = "bigchess")
df <- read.pgn(f)
# ...successfully imported 56 games...

# Example downloaded from https://www.pgnmentor.com/files.html#players and gzipped:
f <- system.file("extdata", "Carlsen.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE)
# Fastest mode:
con <- gzfile(f,encoding = "latin1")
```

```
df <- read.pgn(con,quiet = TRUE,n.moves = FALSE,extract.moves = FALSE,
stat.moves = FALSE, ignore.other.games = FALSE)
# Parse additional tags and extract all moves:
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,add.tags = c("WhiteElo", "BlackElo", "ECO"),extract.moves = -1)
# Example of direct downloading data from chess.com using API:
df <- read.pgn("https://api.chess.com/pub/player/fabianocaruana/games/2013/03/pgn")
# Warning of incomplete line could appear

# Example of scraping all of games given user:
user <- "fabianocaruana"
library("rjson")
json_file <- paste0("https://api.chess.com/pub/player/",user,"/games/archives")
json_data <- fromJSON(paste(readLines(json_file), collapse=""))
result <- data.frame()
for(i in json_data$archives)
result <- rbind(result,read.pgn(paste0(i,"/pgn")))
```

`read.pgn.db` *Reads PGN files into database table*

Description

Reads PGN files into database table

Usage

```
read.pgn.db(con, batch.size = 10^6, conn, table.name = "pgn", ...)
```

Arguments

`con`	connection argument passed directly to readLines() function. String - the name of the file which the data are to be read from or connection object or URL.
`batch.size`	number of lines to read in one batch, default is 10^6.
`conn`	connection argument created by dbConnect
`table.name`	string (default "pgn"), table name, used in dbWriteTable(conn, table.name, read.pgn(batch))
`...`	further arguments passed directly to read.pgn() function (besides ignore.other.games and big.mode)

Examples

```
f <- system.file("extdata", "Carlsen.gz", package = "bigchess")
con <- gzfile(f,"rbt",encoding = "latin1")
require(RSQLite)
conn <- dbConnect(SQLite())
read.pgn.db(con,stat.moves = FALSE,conn = conn)
dbGetQuery(conn, "SELECT COUNT(*) FROM pgn") #2410
dbDisconnect(conn)
```

```
# Works with all types of connections (also gz or zip files).
# con argument is passed directly to readLines(con,batch.size)
# so (if total number of lines to read is greater then batch.size)
# depending on platform use it correctly:

# Windows ('rb' opening mode for loop over readLines):
con <- gzfile(system.file("extdata", "Carlsen.gz", package = "bigchess"),"rb",encoding = "latin1")
# con <- file("path_to_big_chess_file.pgn","rb",encoding = "latin1")
read.pgn.db(con,conn = conn)


# Linux/Mac OS X ('r' opening mode for loop over readLines):
con <- gzfile(system.file("extdata", "Carlsen.gz", package = "bigchess"),"r",encoding = "latin1")
# con <- file("path_to_big_chess_file.pgn","r",encoding = "latin1")
read.pgn.db(con,conn = conn)


# Windows (example of zipped file handling)
unzf <- unzip("zipped_pgn_file.zip")
read.pgn.db(con,conn = conn)
```

`read.pgn.ff` *Reads PGN files into ff data frame*

Description

Reads PGN files into ff data frame

Usage

```
read.pgn.ff(con, batch.size = 10^6, ignore.other.games = F, ...)
```

Arguments

`con`	connection argument passed directly to readLines() function. String - the name of the file which the data are to be read from or connection object or URL.
`batch.size`	number of lines to read in one batch, default is 10^6.
`ignore.other.games`	boolean (default FALSE) if TRUE result is subset of original dataset without games with result marked as "*", i.e. ongoing games. The only one argument which is not passed directly to read.pgn function.
`...`	further arguments passed directly to read.pgn() function (besides ignore.other.games and big.mode)

Value

ff data frame like from read.pgn() function. Since character values are not supported in ffdf object, "Movetext" column is ommited.

Examples

```
require(ff)
require(ffbase)
f <- system.file("extdata", "Carlsen.gz", package = "bigchess")
con <- gzfile(f,"rbt",encoding = "latin1")
# options("fftempdir"="/path/"...) # if necessarily
fdf <- read.pgn.ff(con,stat.moves = FALSE)
delete(fdf)
# Works with all types of connections (also gz or zip files).
# con argument is passed directly to readLines(con,batch.size)
# so (if total number of lines to read is greater then batch.size)
# depending on platform use it correctly:

# Windows ('rb' opening mode for loop over readLines):
con <- gzfile(system.file("extdata", "Carlsen.gz", package = "bigchess"),"rb",encoding = "latin1")
# con <- file("path_to_big_chess_file.pgn","rb",encoding = "latin1")
fdf <- read.pgn.ff(con)
delete(fdf)

# Linux/Mac OS X ('r' opening mode for loop over readLines):
con <- gzfile(system.file("extdata", "Carlsen.gz", package = "bigchess"),"r",encoding = "latin1")
# con <- file("path_to_big_chess_file.pgn","r",encoding = "latin1")
fdf <- read.pgn.ff(con)
delete(fdf)

# Windows (example of zipped file handling)
unzf <- unzip("zipped_pgn_file.zip")
fdf <- read.pgn.ff(file(unzf,"rb"))
delete(fdf)
```

---

stat_moves *Extract statistics of moves*

Description

Extract statistics of moves (counts figure moves) from movetext vector into data frame

Usage

```
stat_moves(movetext, sides = "both")
```

Arguments

movetext	movetext string (or string vector). The standard English values are required: pawn = "P" (often not used), knight = "N", bishop = "B", rook = "R", queen = "Q", and king = "K".
sides	"both" (default),"white" or "black"

Value

Data frame containing moves count statistics for white and for black and total.

Examples

```
stat_moves("1. e4 e5 2. Nf3 Nf5 3. d5 ")
```

`tree_eco` *Compute ECO tree*

Description

Compute ECO tree (frequencies and winning percent)

Usage

```
tree_eco(df, topn = 0)
```

Arguments

`df`	data frame containg ECO and Result columns
`topn`	integer, default 0, indicating how many top openings should be included, 0 means show all openings

Value

Data frame containg White_score (White winning percent), Draws_percent, Black_score and N (number of games). Sorted by power of ECO (White_score * N which describes popularity and score of move) descending.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,stat.moves = FALSE, add.tags = "ECO")
```

tree_move *Compute tree for a given move*

Description

Compute tree for a given move (frequencies and winning percent)

Usage

```
tree_move(df, move)
```

Arguments

df	data frame containg move and Result column from pgn function or data frame containing aggregated data from such df (containg columns: Result, W1, B1, W2, ..., WN, BN, Freq)
move	character indicating which move should be browsed, example "W1"

Value

Data frame containg White_score (White winning percent), Draws_percent, Black_score and N (number of games). Sorted by power of move (White_score times N which describes popularity and score of move) descending.

Examples

```
f <- system.file("extdata", "Kasparov.gz", package = "bigchess")
con <- gzfile(f,encoding = "latin1")
df <- read.pgn(con,quiet = TRUE,stat.moves = FALSE)
# Analyze best answers to 1. e4 in Kasparov games (both white and black)
tree_move(subset(df,W1=="e4"),move = "B1")
# Analyze openings in aggregated data
tree_move(FirstTwoMoves,"W1")
```

Index