MCGS - A User's Guide

Taylor Folkersen

November 2025

- Basic operation
- Solving impartial games
- Testing and working with test files
- Command line Options

Basic Usage

Game name

Space-separated games

"Run" command

- ./MCGS "[clobber_1xn] XXO XO {B}"
  - Solves linear Clobber XXO + XO for black
  - MCGS fundamentally operates on sums

Brackets allow spaces in games

Optional expected outcomes

- ./MCGS "[clobber_1xn] XXO XO [up_star] (-2 *) {B loss, W win}"
  - Solves XXO + XO + ↓2* for both players

- Database is loaded from "database.bin" on startup if present
  - In MCGS version 1.3 it is built automatically if not present
  - In MCGS version 1.4 it must be manually built, and is more configurable

Impartial games

- ./MCGS "[nimber] 7 3 {B win, W win, N 4}"
  - Solves *7 + *3
  - "{N}" to find nim value, with optional expected value
  - "N" uses a different search algorithm (based on mex). "B"/"W" still use minimax
    - Lemoine and Viennot algorithm coming in future version

- ./MCGS "[nimber] 7 [nogo] ..|.. {N}"
  - Error: NoGo is not an impartial game
  - ./MCGS "[nimber] 7 [impartial nogo] ..|.. {N}"

- ./MCGS "[impartial clobber_1xn] XOXOXOXO {N 2}"
  - "impartial" can be applied to any game

- Full input syntax described by file "input/info.test"

Impartial Wrapper Examples

- 1 = {0 | }, and [impartial integer_game] 1 = {0 | 0} = *
- ./MCGS "[switch_game] (1/4, 2) [integer_game] -1 {B loss, W loss}"
  - {1/4 | 2} = 1
  - But their impartial wrapper values are not the same!
  - ./MCGS "[impartial switch_game] (1/4, 2) {N 2} [impartial integer_game] 1 {N 1}"

Files and the Test Framework

- ./MCGS --file example.test
  - "--file" used in place of game string
  - ".test" file must start with a version command i.e. "{version 1.3}"

- ./MCGS --run-tests
  - Runs ".test" files and produces a ".csv" file with stats
  - Default input directory "input/autotests"
  - Default output file "out.csv"
  - Default timeout of 500 ms

- python3 create-table.py out.csv -o out.html
  - Python script produces readable HTML tables from CSV files

Files and the Test Framework (Continued)

- ./MCGS --run-tests --test-dir example_dir --test-timeout 1000 --out-file example1.csv

  Input directory for ".test" files

  Per-test timeout of 1000 ms

  CSV output file name

- ./MCGS --run-tests --test-dir example_dir --test-timeout 1000 --out-file example2.csv --clear-tt

Clear transposition table after each test

- python3 create-table.py example2.csv --compare-to example1.csv -o 2-1.html
  - Compares two CSV files. The first file is the "primary" one

Other Options

- ./MCGS --print-ttable-size
  - transposition table sizes for minimax and impartial game search

- ./MCGS --tt-sumgame-idx-bits 29 --tt-imp-sumgame-idx-bits 28 --print-ttable-size
  - Both table sizes can be configured

- ./MCGS --no-use-db
  - Disable the database

- ./MCGS --db-file-create database.bin "[clobber] max_dims = 3,3; [nogo_1xn] max_dims = 14;"
  - Builds database file "database.bin" with all Clobber games at most 3x3 in size, and all linear NoGo games at most 14 tiles in size

- ./MCGS --play-mcgs "[nogo] ...|...|... {B}"
  - "--play-log log.txt" writes screen output to log.txt

- ./MCGS --print-winning-moves "[nogo] ...|...|... {B, W}"

References, Future Work

- Lemoine, J., Viennot, S.: Nimbers are inevitable. Theoretical Computer Science 462, 70–79 (2012)
- More database features
  - Impartial games
  - Sum games
  - Dominated moves
  - Thermographs and bounds
  - "Simplest" equal games, to allow for substitution
- Move ordering heuristics
- More game types
- Parallel search