Game algorithms

Jan Tomasek, Stepan Havranek,
Pavel Taufer, Jara Cimrman

Outline

- universal example
- taxonomy of games
- basic methods
- algorithms
  - minimax
  - alfa-beta
  - Scout
  - Monte Carlo
- implementation tricks
- computer players statistics
- demonstration

Universal example - piškvorky++

- our own extension of connect five
- based on general surface theory

The beginning I

- connect five on torus or klein bottle
  - too easy
- general surfaces
- two different rules for intercardinal directions
  - up and left vs. left and up
  - both rules allowed
  - but all five connections have to follow one rule

The beginning II

- edges can be connected in two ways
  - handle vs. cross-cap
  - both allowed
- adding non-determinism

Rules of the game

- expansion of classic connect five
  - game space - finite nonempty subset of fields from a 2-dimensional graticule
  - tunnel - pairs of border edges
    - rotates global orientation
  - intercardinal directions - two orthogonal steps
  - goal - at least 5 traversable fields owned in one direction
- implementation
  - board
  - check for winners

Real time example

Havri vs. Tomi on blackboard

Taxonomy of games

- According to the number of players
  - one player : puzzle, sudoku
  - two player game : chess
  - multi player game : piškvorky++, poker
- According to the state information obtained by each player
  - Perfect-information games
  - Imperfect-information games
- According to whether players can fully control the playing of the game
  - deterministic
  - stochastic

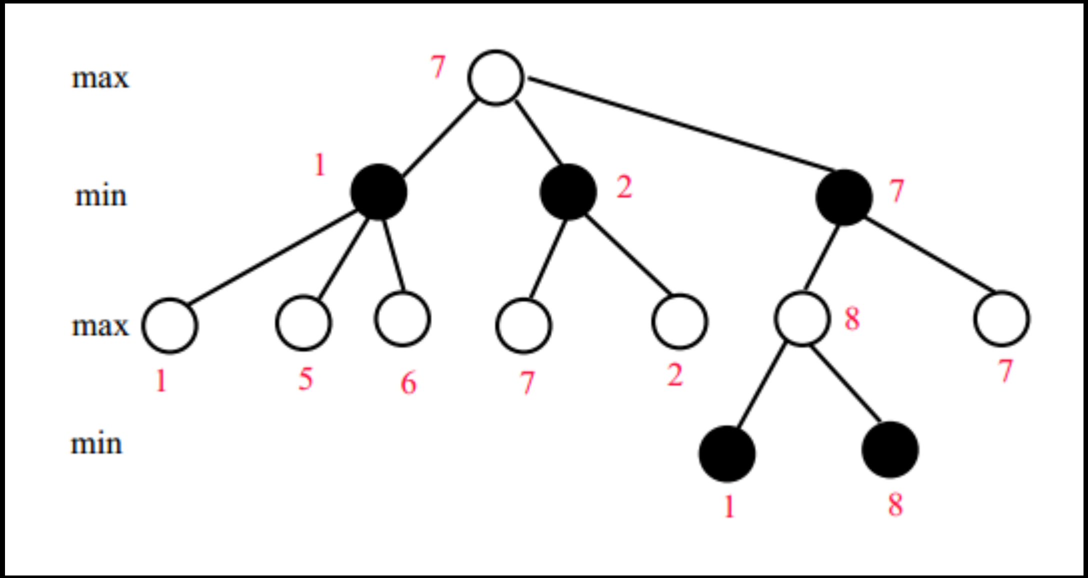
Basic methods

- naive solutions
  - dictionary of all possible positions
    - chess has ~10^{43}
- Brute-force search
  - Breadth-first search (BFS)
  - Depth-first search (DFS)
  - Iterative-deepening DFS (DFID)
  - Bi-directional search
- heuristic search
  - A*
  - IDA*

Minimax

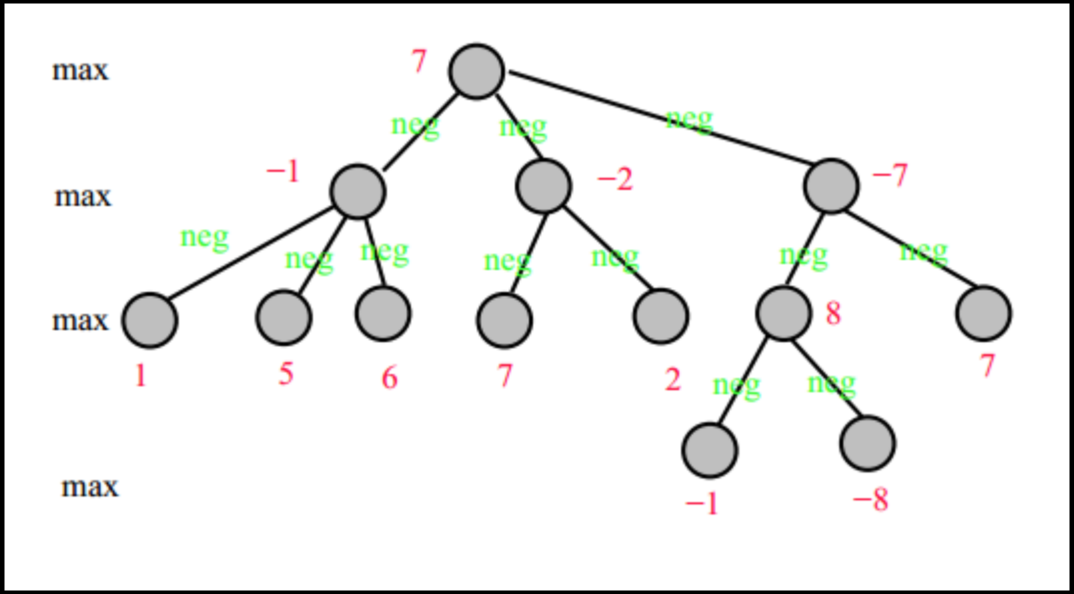
- for: deterministic, complete information
- max and min player
  - max player is looking for best move assuming min player is using optimal strategy (if not it is even better)

Minimax example



Nega-max

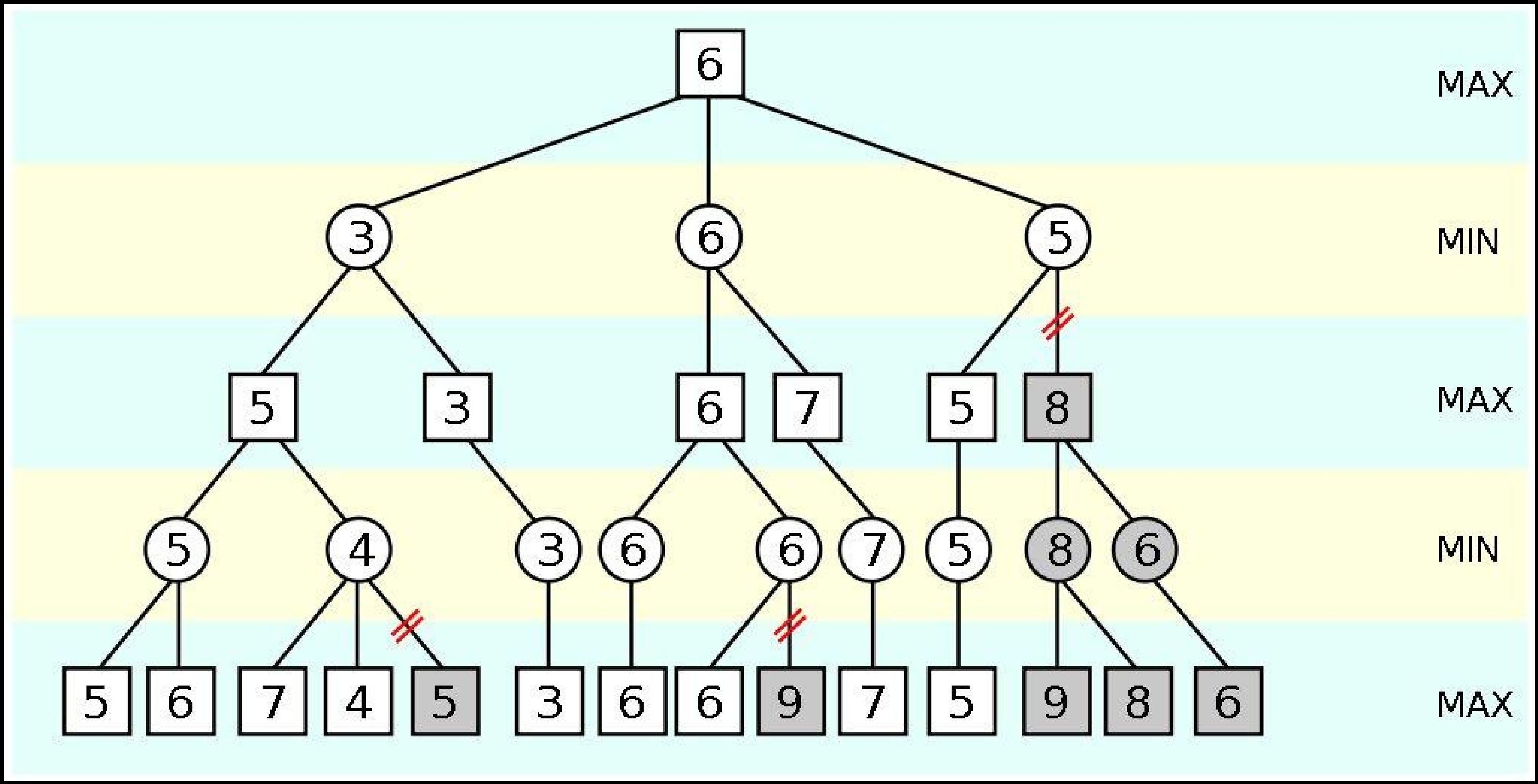
- just another formulation of mini-max
- we are always looking for the maximum, but with each edge we add negation



Alpha - beta pruning

- extension of minimax algorithm
- heuristic for cutting "bad" branches out
- vars alpha and beta
- if values < alpha
  - not interesting vertex (we have a better one)
- if value > beta
  - not interesting vertex for opponent (he has a better one)

Alpha - beta pruning



Alfa - beta Aspiration search

- at beginning of alfa-beta we set
  - alpha = - infinity
  - beta = + infinity
- more information about the game
  - tighter bounds for alpha and beta

Scout algorithm

Scout - idea

- While searching a branch Tb of a MAX node, if we have already obtained a lower bound v`
- First TEST whether it is possible for Tb to return something greater than v`
  - If FALSE, then there is no need to search Tb.
  - If TRUE, then search Tb

Scout - test procedure

procedure TEST(position p, value v, condition >)

determine the successor positions $p_1 \ldots p_d$

if d = 0, then // terminal

    return TRUE if f(p) > v // f is eval function

    return FALSE otherwise

for i := 1 to d do

if p is a MAX node and TEST(p_i, v, >) is TRUE, then return TRUE

if p is a MIN node and TEST(p_i, v, >) is FALSE, then return FALSE

if p is a MAX node, then return FALSE

f p is a MIN node, then return TRUE

Algorithm SCOUT(position p)

determine the successor positions $p_1 \ldots p_d$

if d = 0, then return f(p)

    else v = SCOUT(p_1)

for i := 2 to d do

    if p is a MAX node and TEST(p_i, v, >) is TRUE then

        v = SCOUT(p_i)

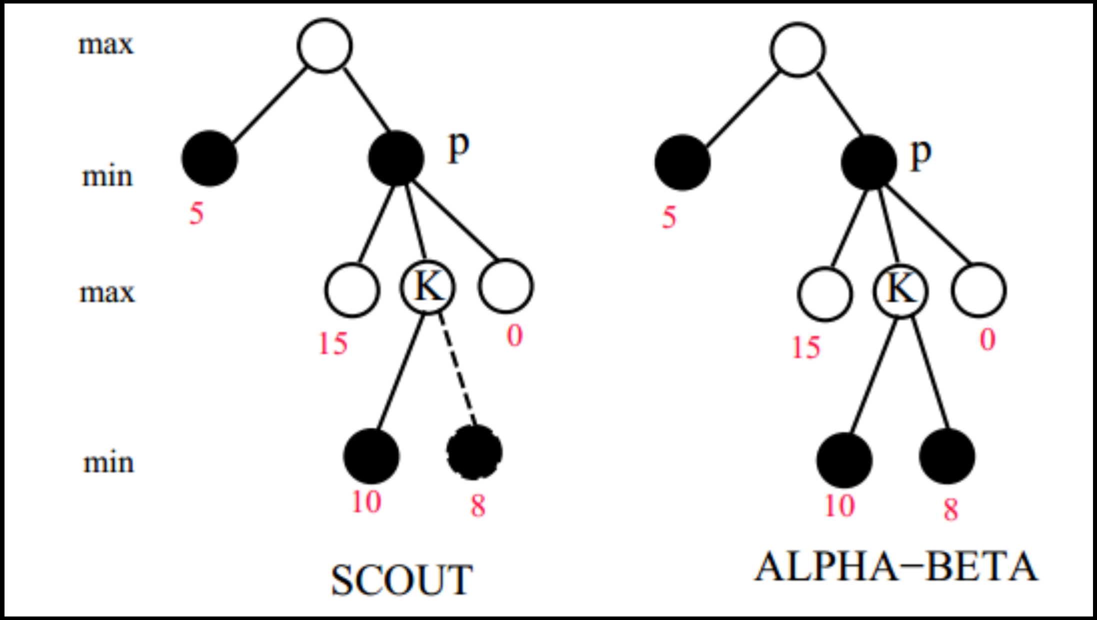
    if p is a MIN node and TEST(p_i, v, >=) is FALSE then

        v = SCOUT(p_i)

return v

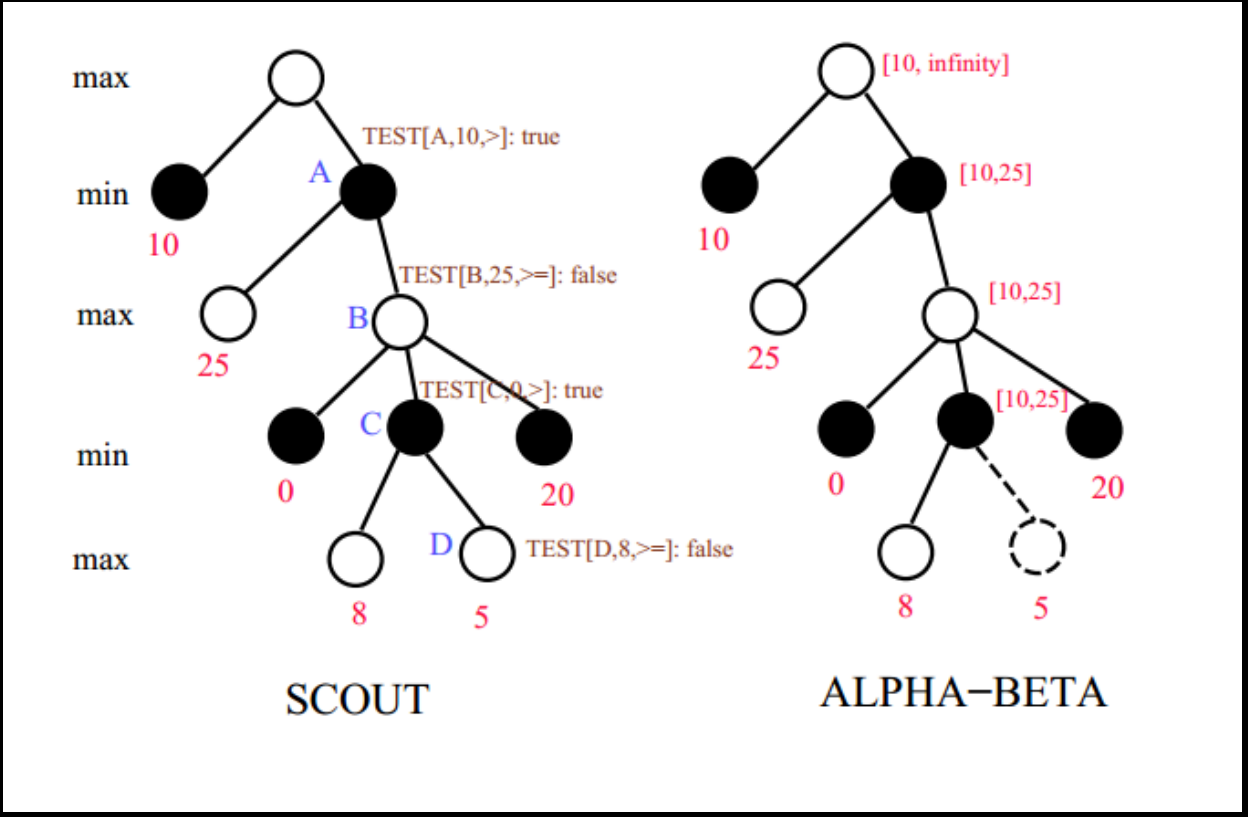
Scout 1

- Assume TEST(p; 5; >) is called by the root after the first branch is evaluated.
  - It calls TEST(K; 5; >) which skips K's second branch.



Scout 2

SCOUT may visit a node that is cut o
 by alpha-beta



Alpha-Beta + Scout

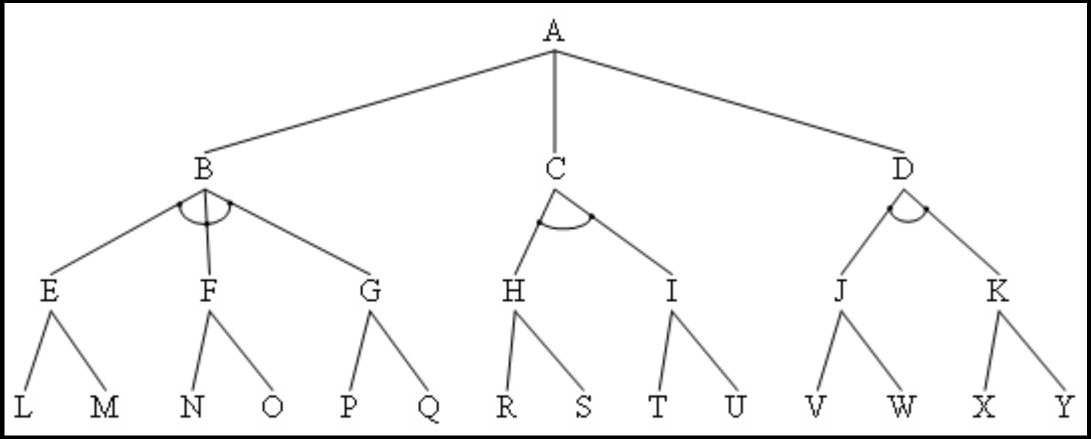
- benefits of both
- add alpha and beta bounds in scout test procedure
- always 40% faster than just alpha-beta :)
    - in chess

Proof-number search

- endgame solvers, sub-goals during games.
- mapping some binary goal to and-or tree
- this small problem can be solved perfectly and the result can be used in standard minimax

And - or tree

- can be solved using DFS,BFS...



Implementation Alpha-Beta Pavel

- basic Alpha-Beta
  - for all empty fields simulate game for given depth
    - either winner or eval
  - pick random with best value
- move evaluation
  - looking in all directions

Monte Carlo

- already presented

Implementation - MCTS Havri

- While have enough of memory (number of expanded nodes) :
  - Selection
    - walk down the graph for most promising node
  - Expansion
    - compute possible moves and evaluate
  - Simulation
    - based on number of free cells in line
  - Backpropagation
    - update evaluated + simulated value through parents

Observations

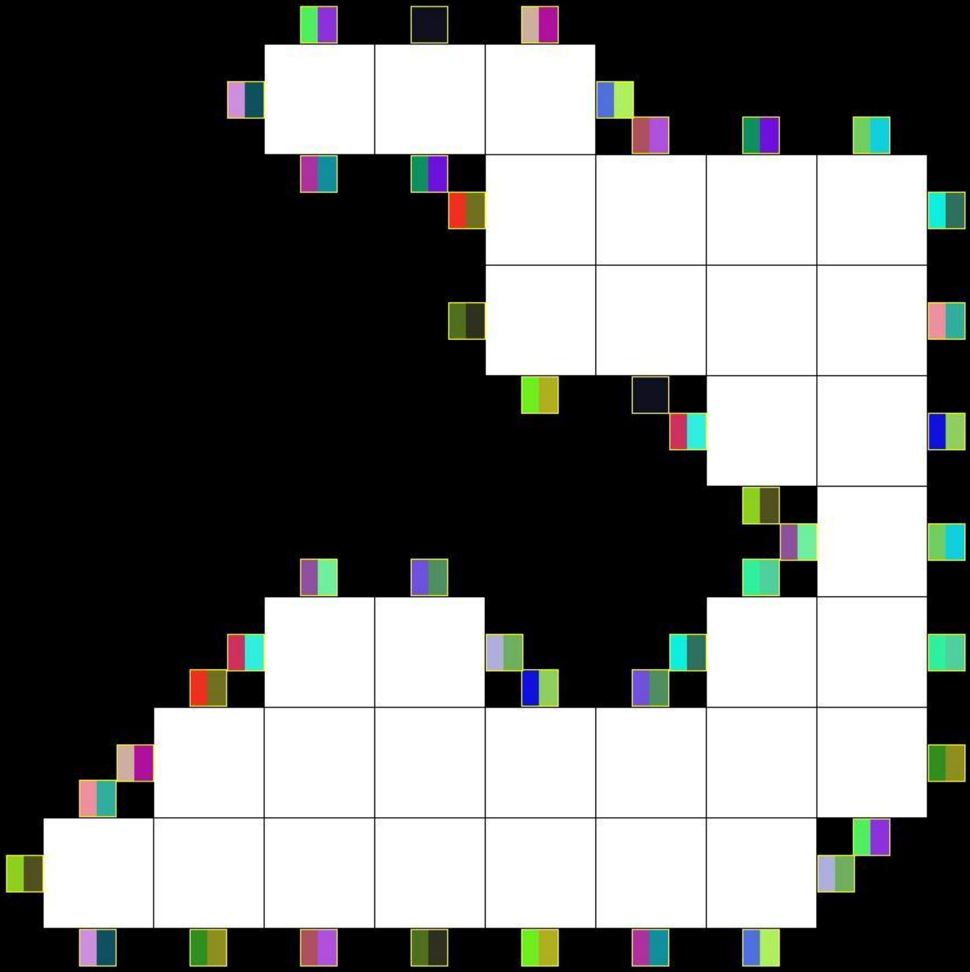
- games often short
- first player often wins
- our implementations are better than humans
    - computer sees complicated paths the human usually can't handle

Statistics

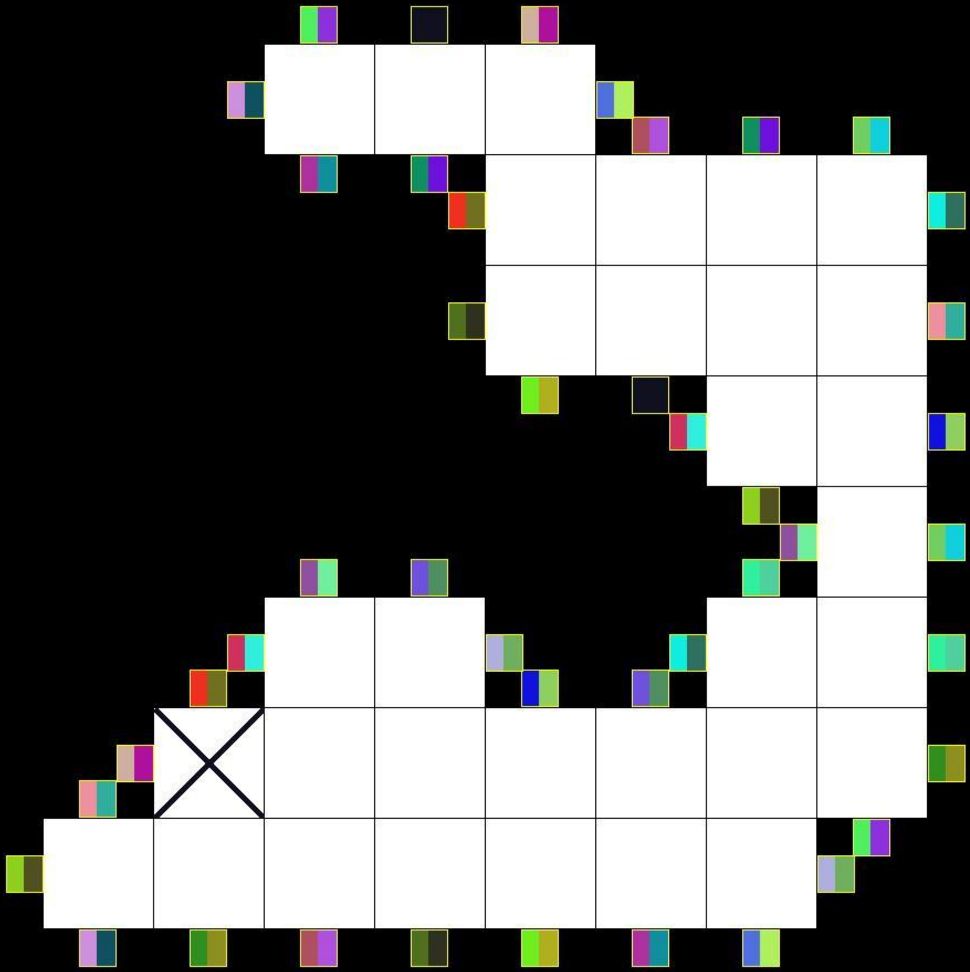
	MC Tree size	10	100	10k	10M	100M
AlfaBeta	Win first	20	20	19	20	16
	Win second	6	4	5	6	2
	Moves per game first	4.05	4.10	3.80	4.45	4.80
	Moves per game second	1.30	2.05	1.60	2.20	2.80
	Time per move	13639ms	26824ms	28517ms	26211ms	29561ms
MonteCarlo	Win first	14	16	15	14	18
	Win second	0	0	1	0	4
	Moves per game first	2.00	2.50	2.25	2.80	2.90
	Moves per game second	3.05	3.10	2.80	3.70	3.90
	Time per move	84.25ms	132.68ms	155.41ms	186.25ms	198.56ms

Demonstration

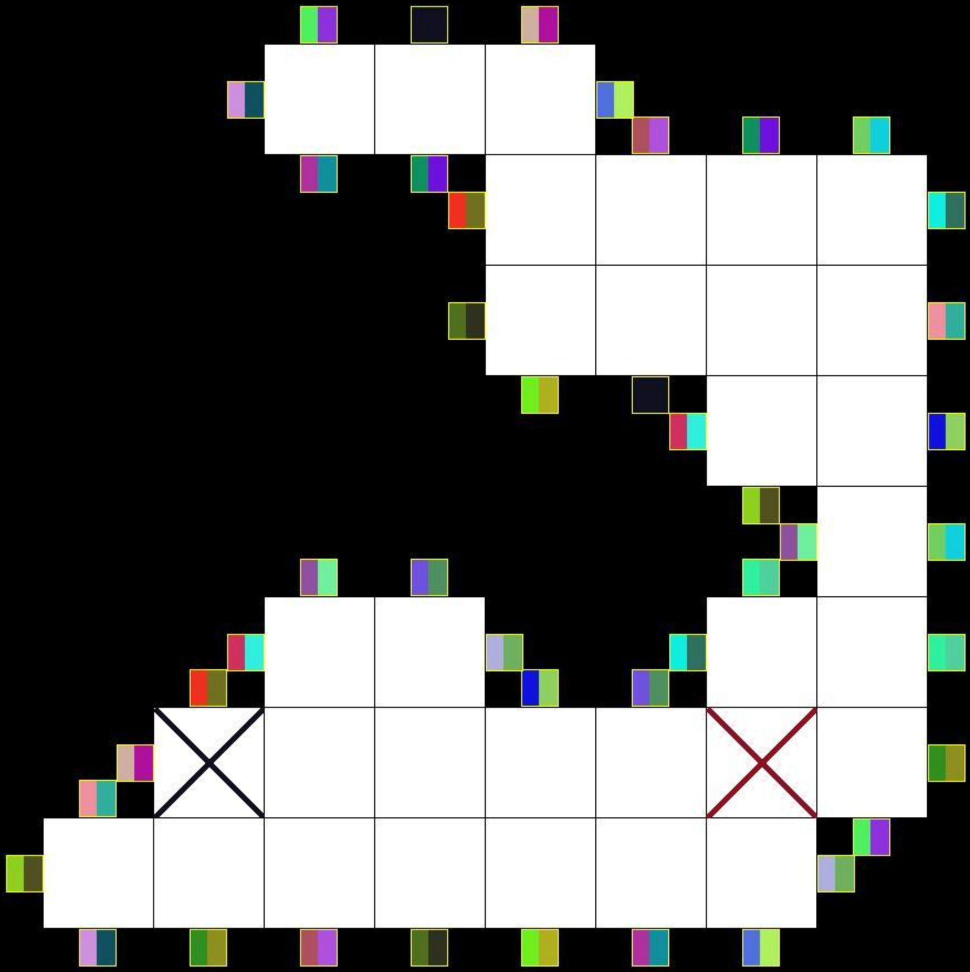
MCTS vs. alpha-beta 0



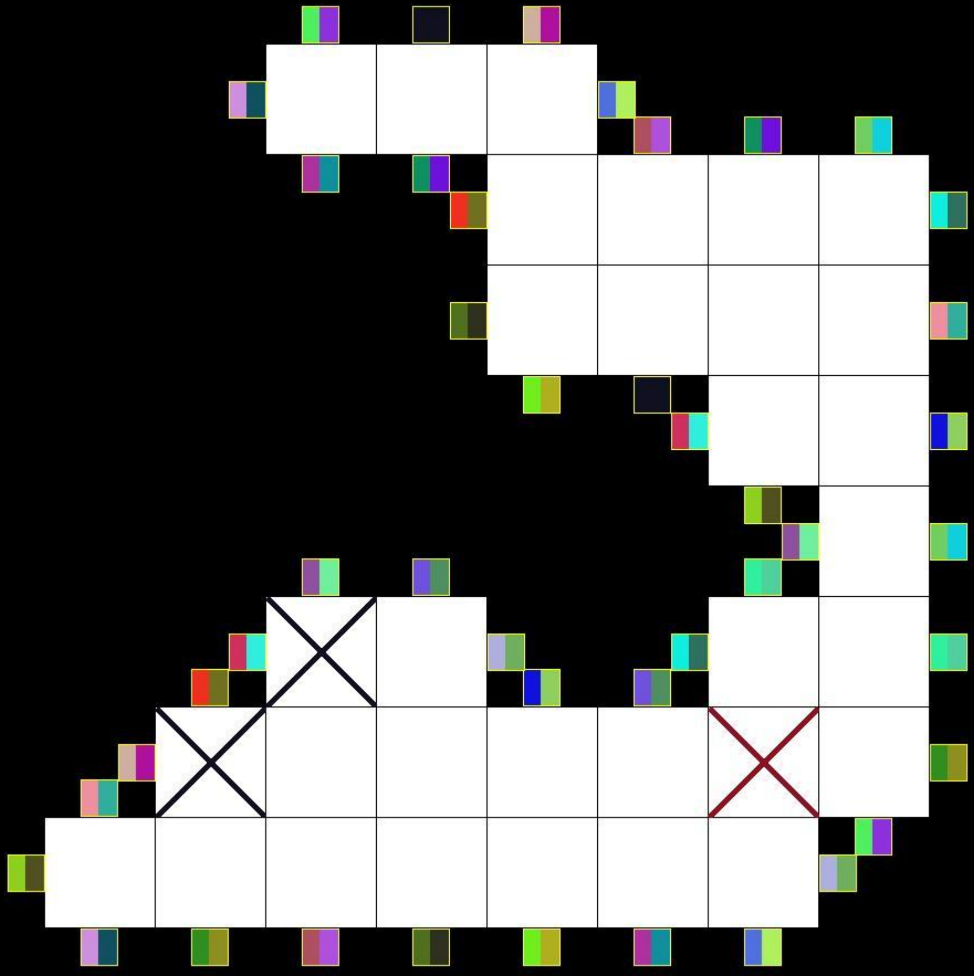
MCTS vs. alpha-beta 1



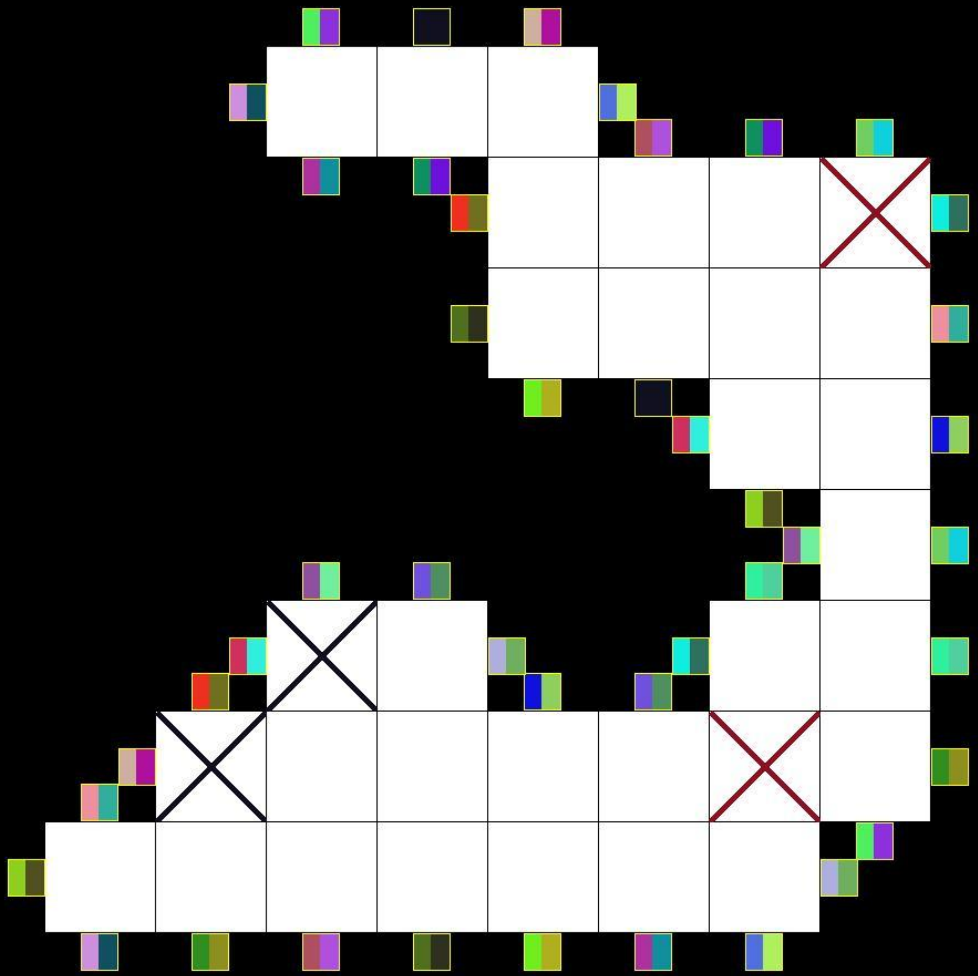
MCTS vs. alpha-beta 2



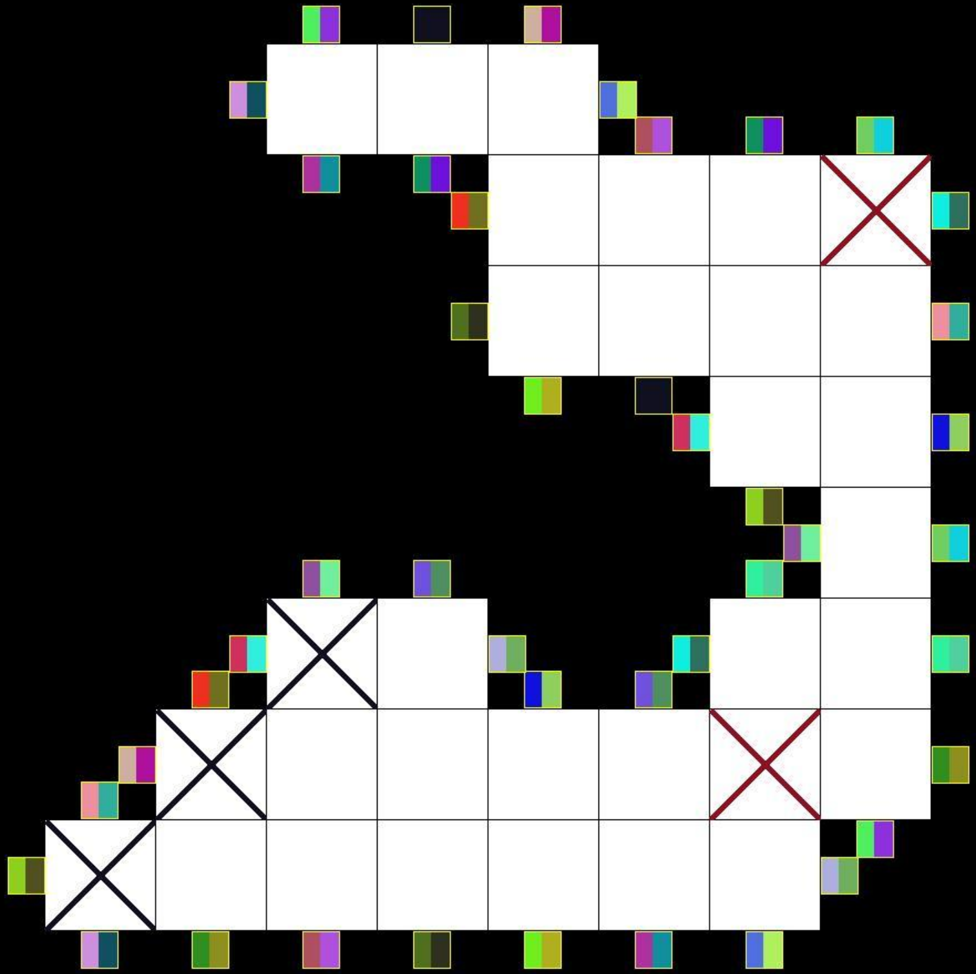
MCTS vs. alpha-beta 3



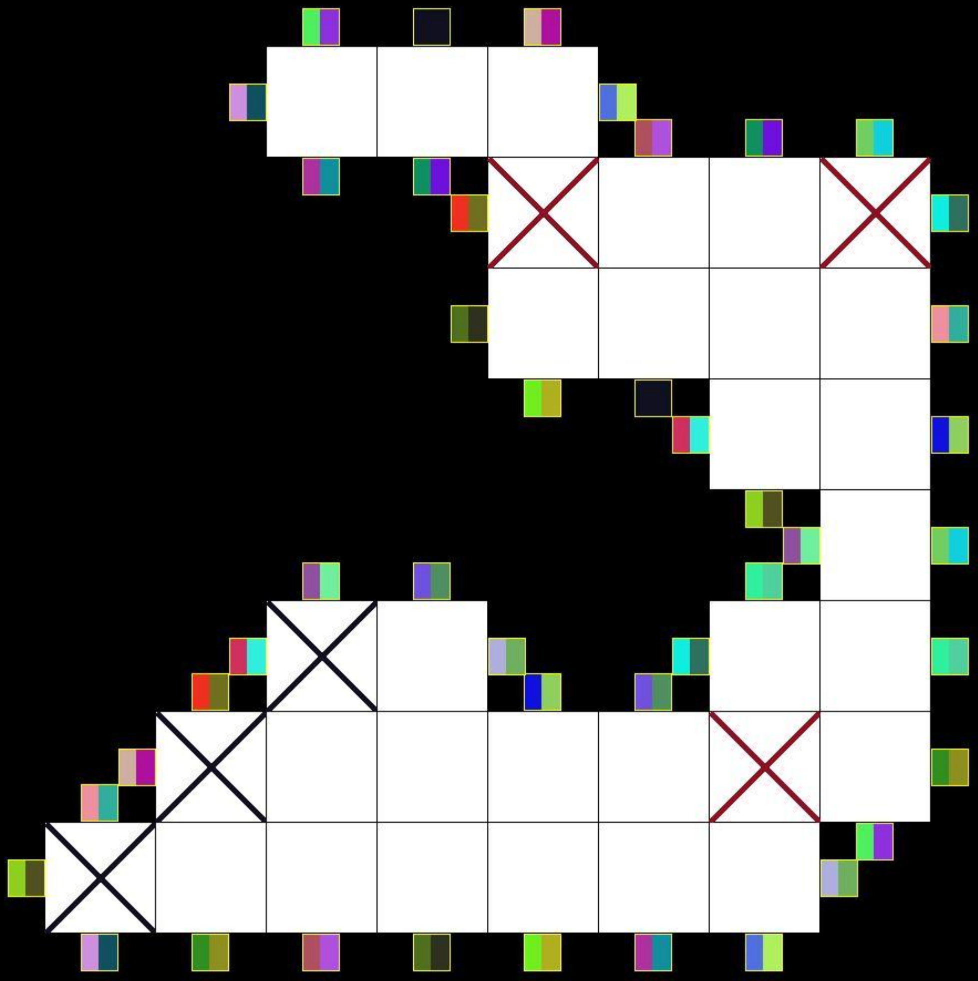
MCTS vs. alpha-beta 4



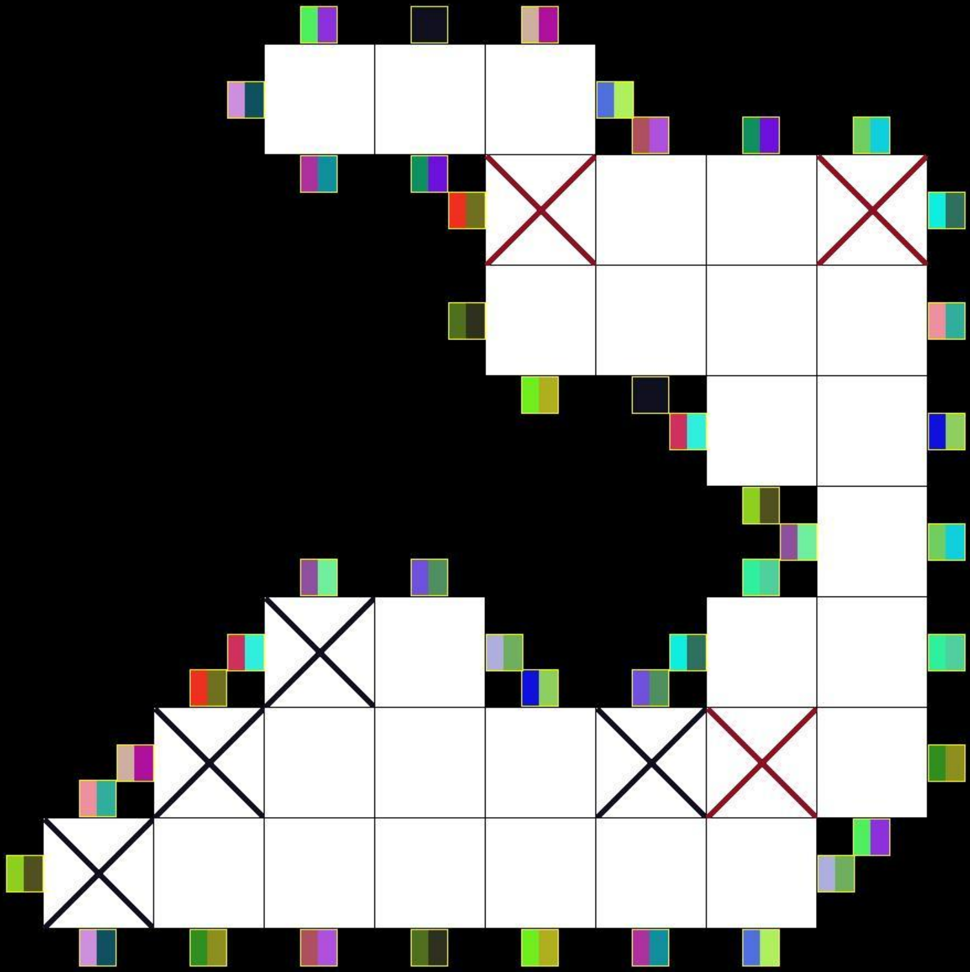
MCTS vs. alpha-beta 5



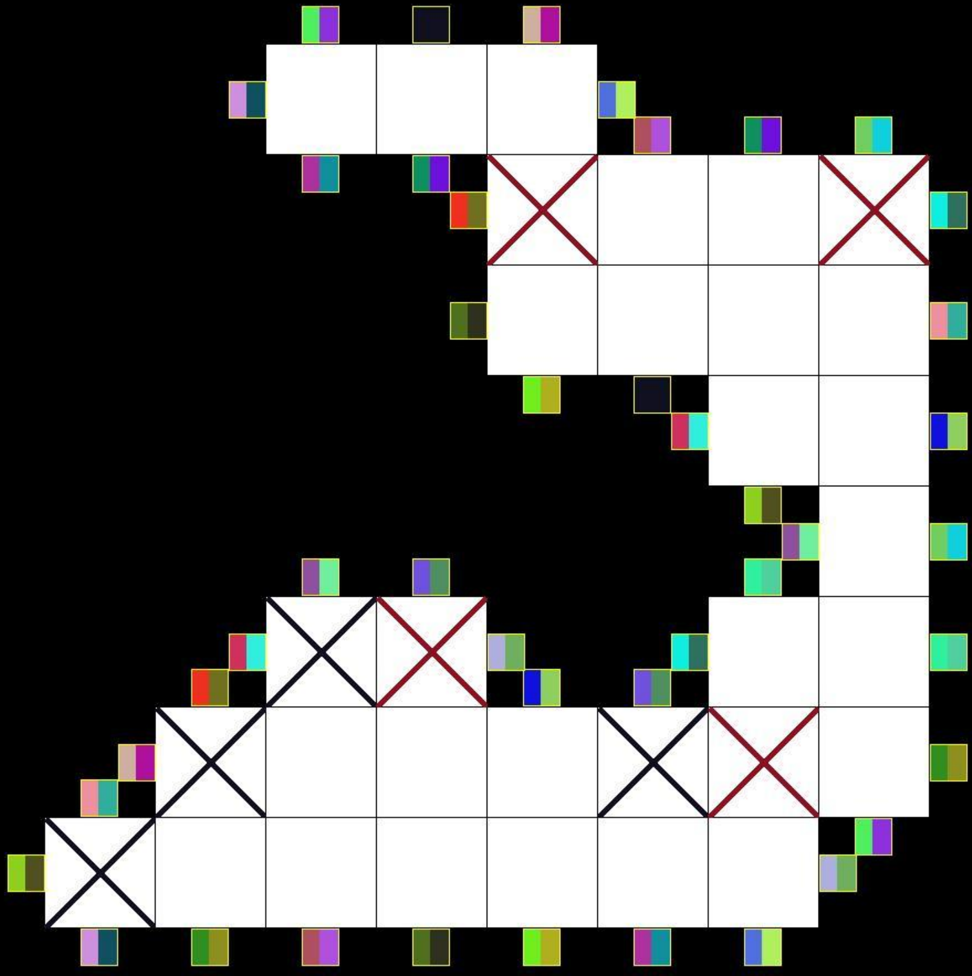
MCTS vs. alpha-beta 6



MCTS vs. alpha-beta 7



MCTS vs. alpha-beta 8



Sources

- black tea, green tea, yellow tea, coffee, chocolate
- http://pasky.or.cz/vyuka/2012-AIL103/prez34_go_mcts.pdf
- http://pasky.or.cz/vyuka/2012-AIL103/prez2_minimax.pdf
- http://pasky.or.cz/vyuka/2012-AIL103/prez1_hernialg.pdf
- http://www.iis.sinica.edu.tw/~tshsu/tcg2010/slides/slide1.pdf
- http://www.iis.sinica.edu.tw/~tshsu/tcg2010/slides/slide2.pdf
- http://www.iis.sinica.edu.tw/~tshsu/tcg2010/slides/slide3.pdf
- http://www.iis.sinica.edu.tw/~tshsu/tcg2010/slides/slide4.pdf

Wishing you the Gifts of Peace
and Happiness this Christmas
and throughout the New Year