Blunder cost in Go and Hex

Brausen Hayward Müller Qadir Spies

computing univ alberta

2011 november

BLUNDER

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

Blunder

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

Blunder

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

BLUNDER

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

BLUNDER

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

Blunder

- def'n: to stumble blindly
- an ignorant move (bad when good available)
- a random move
- end game: usually fatal
- early game: maybe not

Blunder analysis

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

Blunder analysis

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

BLUNDER ANALYSIS

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

Blunder analysis

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

BLUNDER ANALYSIS

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

Blunder analysis

- assume stochastic-player
- blunder: random move
- what if one player always blunders on move k?
- Fuego vs Fuego-blunder
- MoHex vs MoHex-blunder

Simple game model: parameters

Parameters

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

Simple game model: parameters

Parameters

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in $\{0, \ldots, T\}$
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in $\{0, \ldots, T\}$
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

Simple game model: parameters

Parameters

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

SIMPLE GAME MODEL: PARAMETERS

PARAMETERS

- 2-player alternate-move no-draw
- T: max moves per game
- t: move — in {0, . . . , T}
- e_t: solving ease after t moves — 0 (hard) . . . 1 (easy)
- w_t: fraction winning moves — -1 (all lose) . . . 1 (all win)
  $w_t < 0$ means
  - all moves lose
  - after best move, $w_{t+1} \approx -w_T$
- m_t: score of move t — -1 (bad) . . . 1 (good)
- r_t: rank of move t — 1 (worst) . . . k (best)
- s_p: strength of player p — -1 (anti-perfect) . . . 1 (perfect)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
    - sample m_t with $E[m_t] = s_p \times e_t$
    - compute r_t from m_t
    - using w_t, make move with rank r_t
    - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: game simulation

Game simulation

- assume games non-pathological
- for each t
  - sample m_t with $E[m_t] = s_p \times e_t$
  - compute r_t from m_t
  - using w_t, make move with rank r_t
  - sample w_{t+1} using w_t and r_t (strongest move from winning position leaves no opponent winning moves)

Simple game model: blunder cost

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $E[m_t] = 0 \quad (s_p \rightarrow 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

SIMPLE GAME MODEL: BLUNDER COST

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $\quad E[m_t] = 0 \quad (s_p \to 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

Simple game model: blunder cost

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $E[m_t] = 0 \quad (s_p \to 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

Simple game model: blunder cost

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $E[m_t] = 0 \quad (s_p \to 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

Simple game model: blunder cost

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $E[m_t] = 0 \quad (s_p \to 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

Simple game model: blunder cost

- $E[m_t] \approx s_p \times e_t$
- assume $s_p > 0$
- blunder $E[m_t] = 0 \quad (s_p \rightarrow 0)$
- blunder cost: resulting drop in expected win rate
- indirectly measures $s_p \times e_t \times w_t$

SETTINGS

Fuego vs Fuego-blunder

- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

SETTINGS

FUEGO VS FUEGO-BLUNDER

- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

Settings

Fuego vs Fuego-blunder

- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

SETTINGS

FUEGO VS FUEGO-BLUNDER

- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

SETTINGS

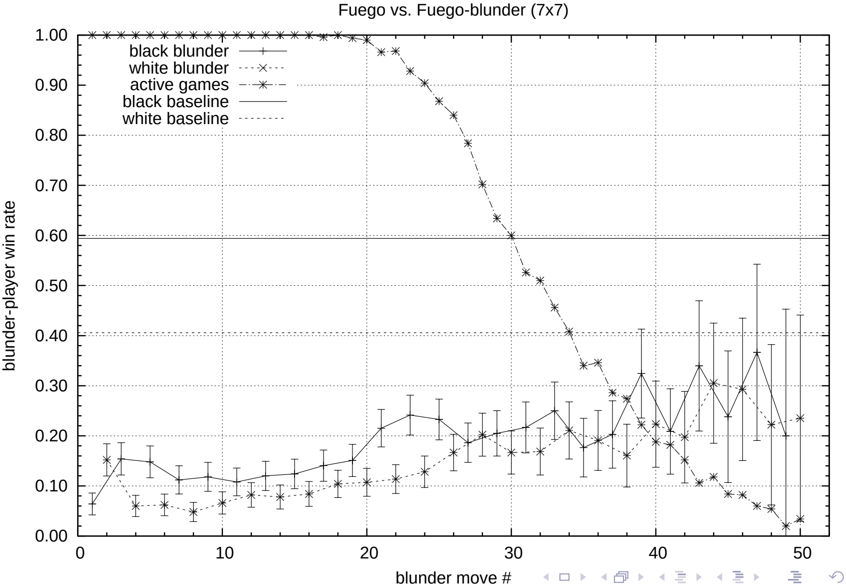
FUEGO VS FUEGO-BLUNDER

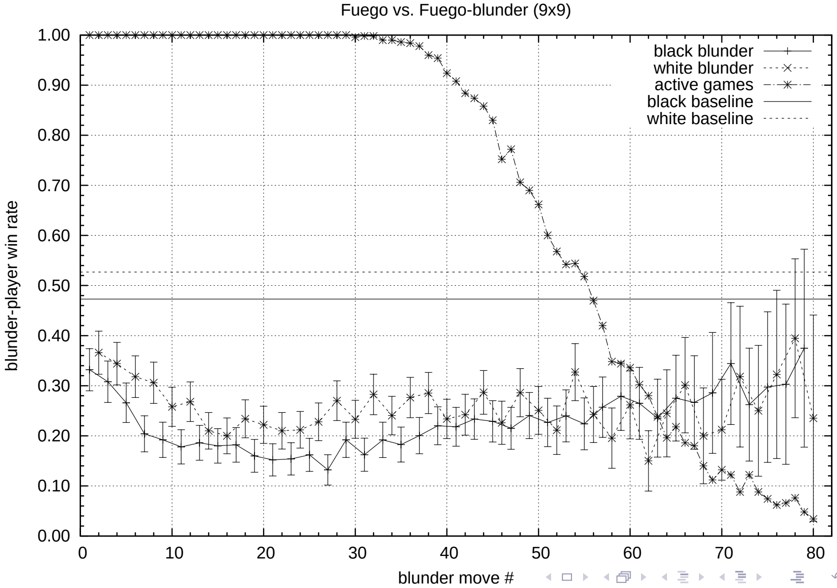
- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

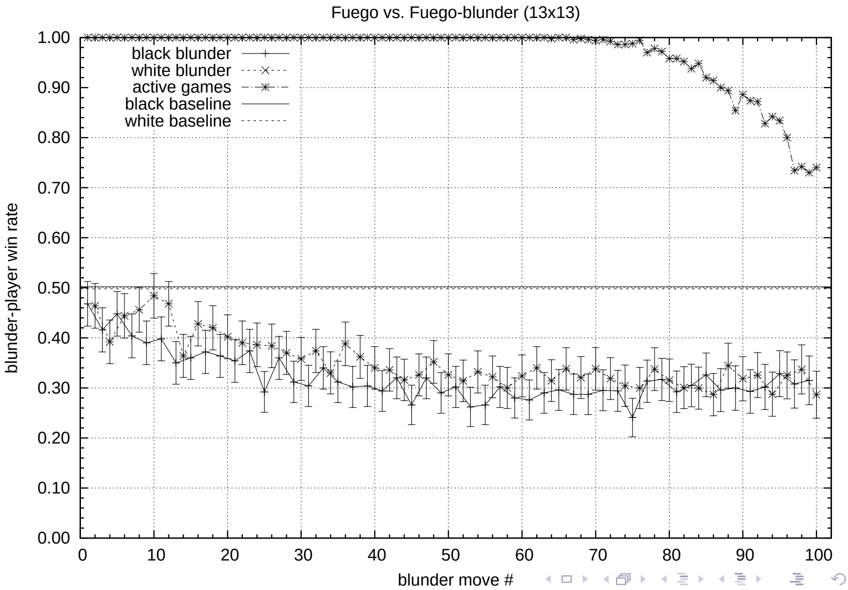
Settings

Fuego vs Fuego-blunder

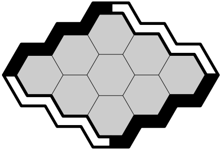
- board 7×7, 9×9, 13×13
- for each move t, 500 games
- for each game, 10K MCTS simulations / move
- resign threshold 0.05
- white komi 7.5

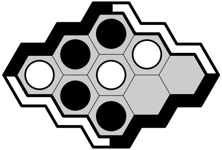

Fuego vs. Fuego-blunder (7x7)
black blunder
white blunder
active games
black baseline
white baseline
blunder-player win rate
blunder move #
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
10
20
30
40
50


Fuego vs. Fuego-blunder (9x9)
black blunder
white blunder
active games
black baseline
white baseline
blunder-player win rate
blunder move #
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
10
20
30
40
50
60
70
80

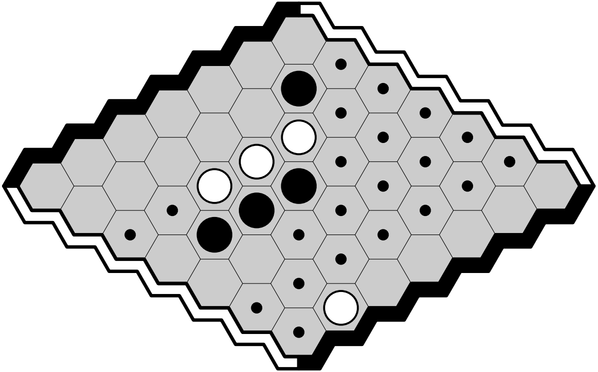

Fuego vs. Fuego-blunder (13x13)
black blunder
white blunder
active games
black baseline
white baseline
blunder-player win rate
blunder move #
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
10
20
30
40
50
60
70
80
90
100

Hex





All winning moves



Settings

MoHex vs MoHex-blunder

- board 7×7, 9×9, 11×11
- for each move t, 500 games
- for each game, 1K MCTS simulations / move
- 7×7: after, use solver to find winning moves

Settings

MoHex vs MoHex-blunder

- board 7×7, 9×9, 11×11
- for each move t, 500 games
- for each game, 1K MCTS simulations / move
- 7×7: after, use solver to find winning moves

Settings

MoHex vs MoHex-blunder

- board 7×7, 9×9, 11×11
- for each move t, 500 games
- for each game, 1K MCTS simulations / move
- 7×7: after, use solver to find winning moves

Settings

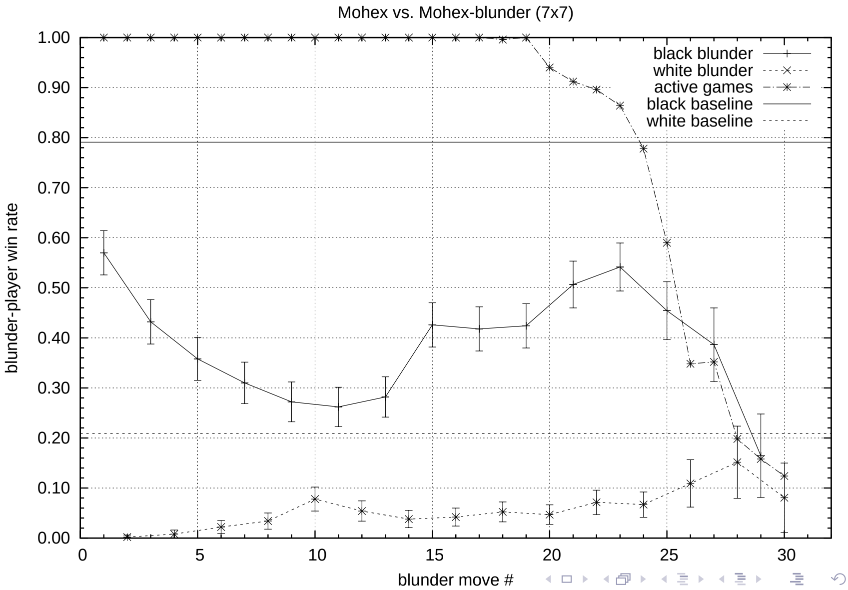
MoHex vs MoHex-blunder

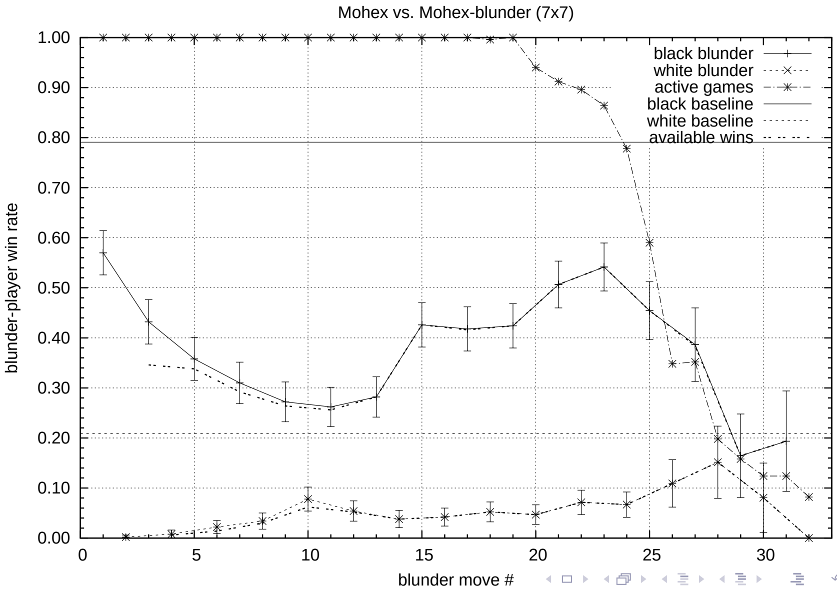
- board 7×7, 9×9, 11×11
- for each move t, 500 games
- for each game, 1K MCTS simulations / move
- 7×7: after, use solver to find winning moves

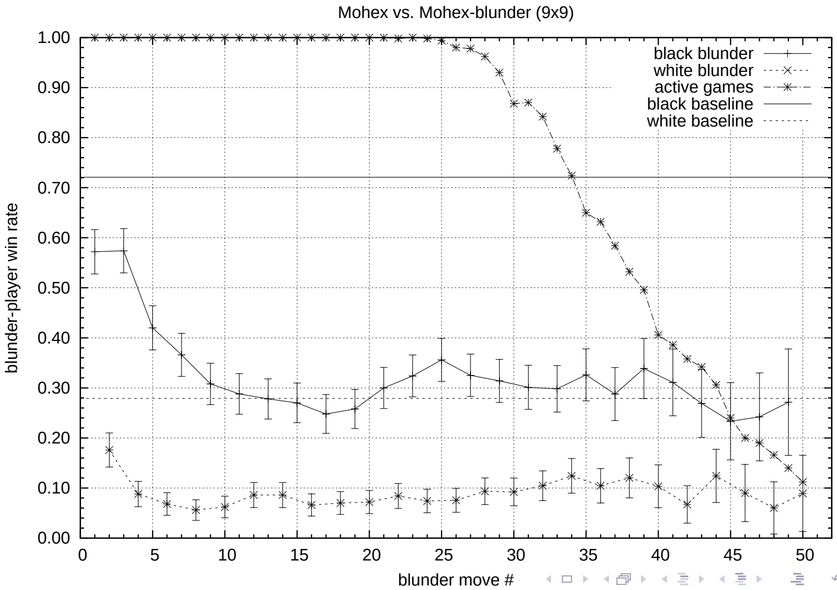
Settings

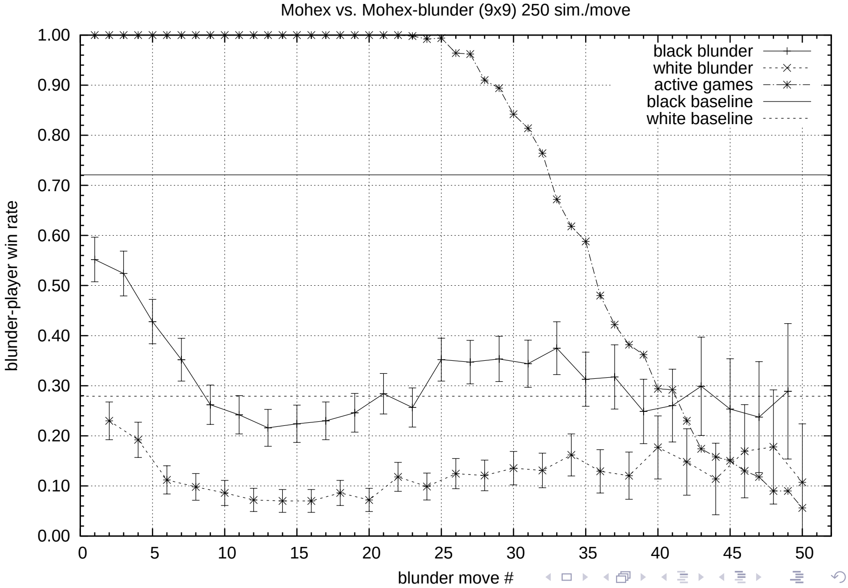
MoHex vs MoHex-blunder

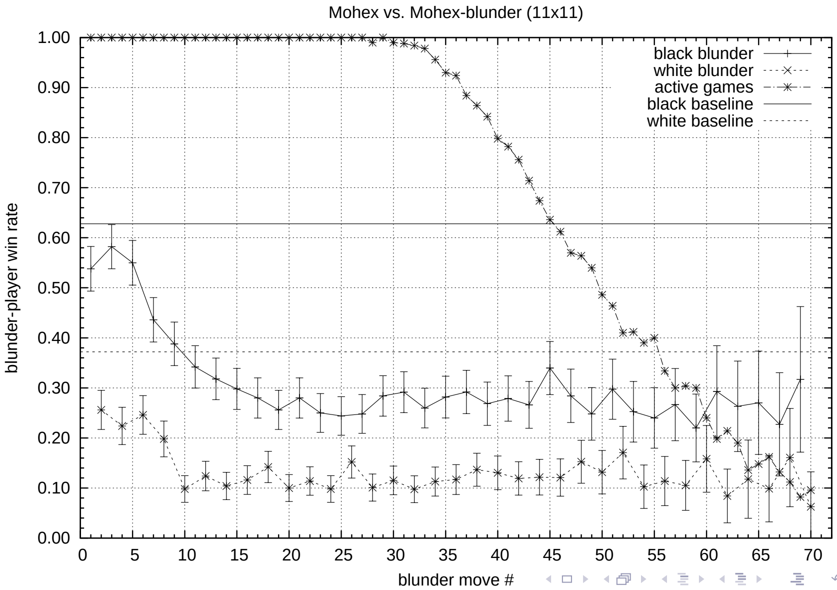
- board 7×7, 9×9, 11×11
- for each move t, 500 games
- for each game, 1K MCTS simulations / move
- 7×7: after, use solver to find winning moves


Mohex vs. Mohex-blunder (7x7)
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
blunder-player win rate
0
5
10
15
20
25
30
blunder move #
black blunder
white blunder
active games
black baseline
white baseline


Mohex vs. Mohex-blunder (7x7)
blunder-player win rate
blunder move #
black blunder
white blunder
active games
black baseline
white baseline
available wins
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
5
10
15
20
25
30


Mohex vs. Mohex-blunder (9x9)
blunder-player win rate
blunder move #
black blunder
white blunder
active games
black baseline
white baseline
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
5
10
15
20
25
30
35
40
45
50


Mohex vs. Mohex-blunder (9x9) 250 sim./move
blunder-player win rate
blunder move #
black blunder
white blunder
active games
black baseline
white baseline
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
5
10
15
20
25
30
35
40
45
50


Mohex vs. Mohex-blunder (11x11)
blunder-player win rate
blunder move #
black blunder
white blunder
active games
black baseline
white baseline
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70

Conclusions

- start: low cost
- after few moves: more costly
- after: cost smooth
- time management ? not yet . . .

Conclusions

- start: low cost
- after few moves: more costly
- after: cost smooth
- time management ? not yet . . .

Conclusions

- start: low cost
- after few moves: more costly
- after: cost smooth
- time management ? not yet . . .

Conclusions

- start: low cost
- after few moves: more costly
- after: cost smooth
- time management ? not yet . . .

CONCLUSIONS

- start: low cost
- after few moves: more costly
- after: cost smooth
- time management ? not yet . . .

FUTURE WORK

- smarter "blunders"
- eg: sample only from reasonable moves ?
- (Hex: stay in mustplay Go: no eye fill-in)
- eg: blunder player less time (ponder player more time) ?

FUTURE WORK

- smarter "blunders"
- eg: sample only from reasonable moves ?
- (Hex: stay in mustplay   Go: no eye fill-in)
- eg: blunder player less time (ponder player more time) ?

Future work

- smarter "blunders"
- eg: sample only from reasonable moves ?
- (Hex: stay in mustplay Go: no eye fill-in)
- eg: blunder player less time (ponder player more time) ?

FUTURE WORK

- smarter "blunders"

- eg: sample only from reasonable moves ?
- (Hex: stay in mustplay Go: no eye fill-in)
- eg: blunder player less time (ponder player more time) ?

Future work

- smarter "blunders"
- eg: sample only from reasonable moves ?
- (Hex: stay in mustplay   Go: no eye fill-in)
- eg: blunder player less time (ponder player more time) ?

DANK U THANK YOU

- UofA GAMES, Schaeffer
- Natural Sciences and Engineering Research Council of Canada