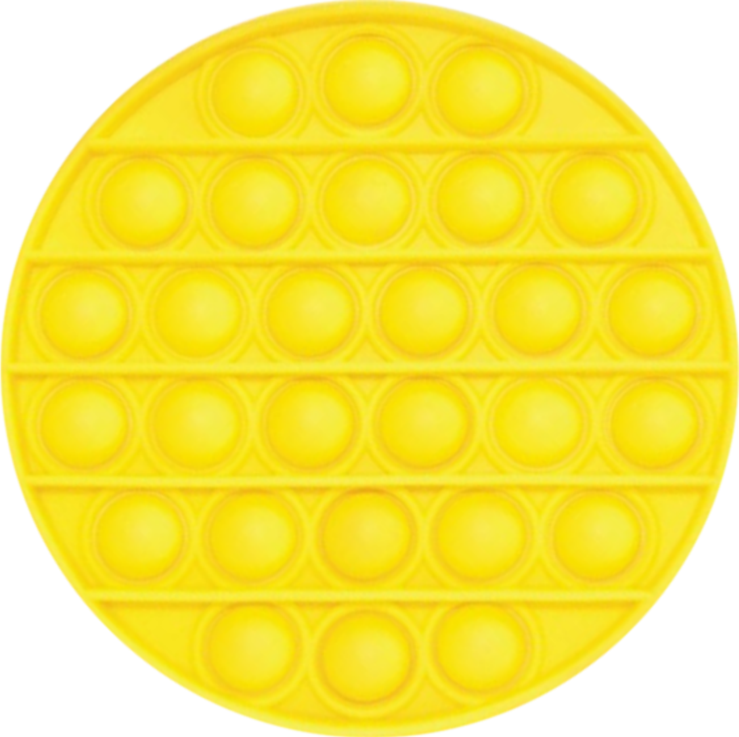
Analyzing the Game Mechanics of Go!Pop Roundo(TM): Why Locally Best Moves are Not Optimal

Alexander K. Seewald[1]

[1] Seewald Solutions, Lärchenstraße 1, 4616 Weißkirchen a.d. Traun, Austria

alex@seewald.at
www.seewald.at

What is Go!Pop Roundo(TM)?



- Remake of the classic Last Mouse Lost. 28 fields, any number of fields in same row can be toggled - even non-adjacent, Players move consecutively, Last one toggling a field (forced move) loses. Turning around allows restart.
- Equivalent to both players trying to force an odd number of remaining moves.

Example Game

1

```
   . . .
 * * * * *
. . . . . .
. . . . . .
 . . . . .
   . . .
```

2

```
   . * *
 * * * * *
. . . . . .
. . . . . .
 . . . . .
   . . .
```

3

```
   . * *
 * * * * *
* * . * * .
. . . . . .
 . . . . .
   . . .
```

4

```
   * * *
 * * * * *
* * . * * .
. . . . . .
 . . . . .
   . . .
```

5

```
   * * *
 * * * * *
* * . * * .
. . . . . .
 . . . . .
   * * *
```

6

```
   * * *
 * * * * *
* * . * * .
. . . . . *
 . . . . .
   * * *
```

7

```
   * * *
 * * * * *
* * . * * .
. . . . . *
 . * * * *
   * * *
```

8

```
   * * *
 * * * * *
* * . * * *
. . . . . *
 . * * * *
   * * *
```

9

```
   * * *
 * * * * *
* * . * * *
* * * * * *
 . * * * *
   * * *
```

SEEWALD SOLUTIONS

alex@seewald.at
www.seewald.at

Move Evaluation Approaches (1)

To *teach* this game to a computer, we've evaluated several move evaluation methods:

- Play a set of (e.g. 100) games with random but legal moves until the end, collate outcomes as good (*odd* remaining moves) and bad (*even* remaining moves)

  Performs very badly, easy to beat, very elementary mistakes

- Local search of whole game tree in end-game

  Works well, but very slow since move state >> field state

- Generate whole game tree backwards from last move (with highly optimized code ~ 15min on single CPU, 12 GB size)

  Stores for each field the counts of even and odd moves to the final move w/ 128bit integers

Move Evaluation Approaches (2)

Surprisingly, locally choosing the move which is most likely to yield a positive outcome performs very badly!

- Game tree contains *all moves*, not just good ones
- In many cases the winning move is not one with the highest proportion of positive outcomes, e.g:

```
   * * *
 * * * * .
. . * * * *
* * * * * *
 * * * * *
   . * .
```

move=65 bestVal=0.315789

...

move=176 bestVal=0.750000

...

move=325 bestVal=0.750000

Playing the game to the end using locally best moves, and just choosing moves that result in a subplay win, weighted by proportion of positive outcomes, works well = Game Tree Search

Evaluating Game Tree Search....

- Always wins against local search...
  ... since both compute the same game tree

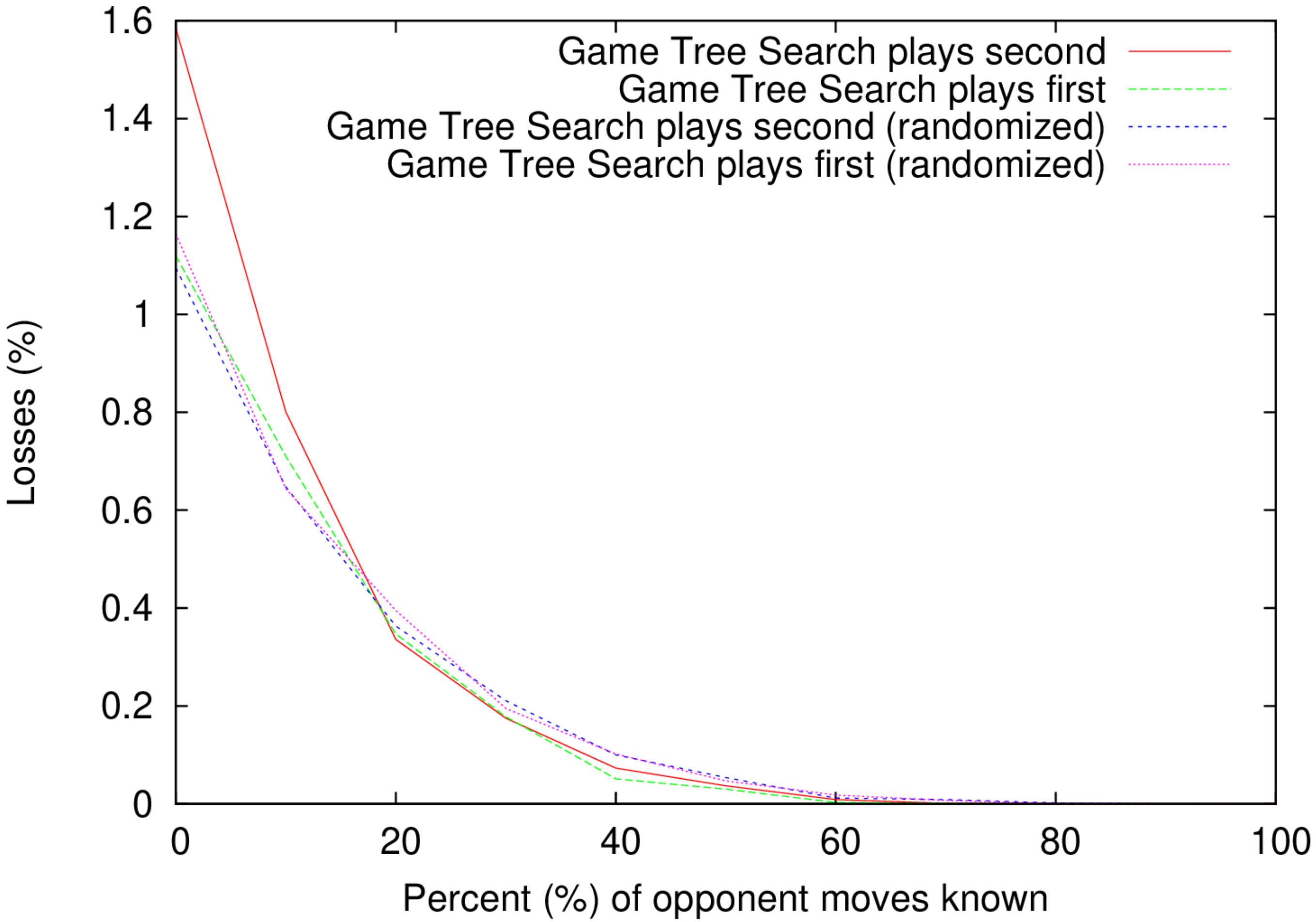
- Always wins against move evaluation by random self-play

- Always wins against itself (when playing second)

- Almost always wins against random moves
  1.59% losses when random plays first, GTS second
  1.12% losses when GTS plays first, random second

Can we do better?

- Partial info on opponent moves, oracle to tell **when** known
- Play against random moves
- Randomly sampling non-losing moves (*randomized*) helps



Improve Human Player Experience

- GTS is by default deterministic: always plays same move on same field - must randomly sample over non-losing moves (*randomize* in previous graph)
- This practically removes first move advantage and is generally beneficial w.r.t. losses.
- Also reduces the chance of repeating a given winning move sequence to about 3.47% (from 100% for deterministic variant)
- Losses vs. human opponents are likely much higher than those against random play opponents.
- Main test player did not want to play against computer any more

  Maybe intuitively understands that opponent model is the most significant part of game play, and this cannot be applied to any computer game algorithm...

Make GTS publicly available?

- User interface (ncurses) and rendering (as text mode "graphic") is quite basic, needs to be improved
- Most sensible implementation: Responsive web-based app
- 12GB Game Tree too large for most servers to keep in memory (but can be memory-mapped at cost of runtime... or we could buy a new server with more memory ;-)
- Not easy to motivate people to play - however if your children are better at the game than you, it might prove sufficient motivation... ;-)