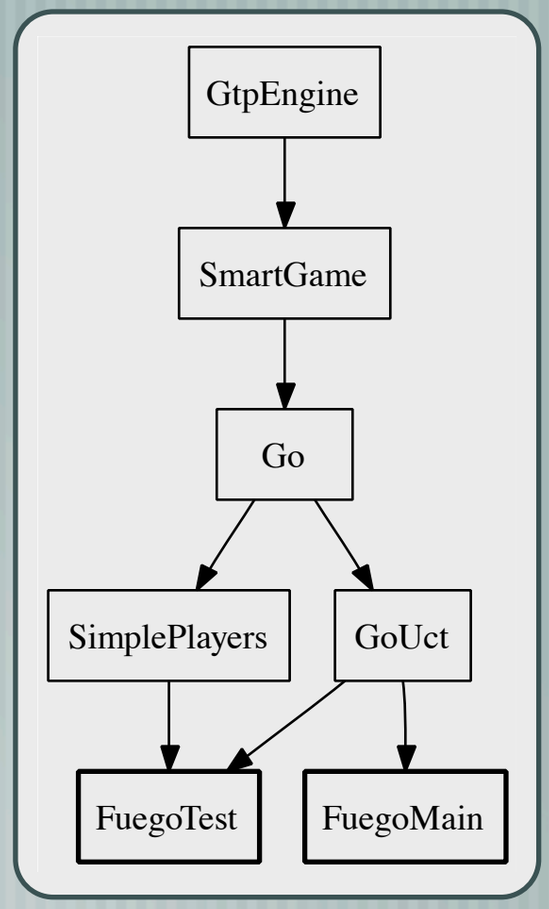
Challenges in Monte Carlo Tree Search

Martin Müller
University of Alberta

Contents

The Fuego Project

- Open-source program hosted on sourceforge
- Originally developed at University of Alberta
- Game-independent kernel, General Go engine, MC Go program
- Applications and extensions: MoHex (Hex), BlueFuego, Arrow (Amazons),RLGo,...



Fuego Go Program

- High-level design similar to MoGo, many others
  - Many differences in details, implementation
- First program to win a 9x9 game vs top human professional
- Won 9x9 Olympiad in Pamplona 2009
- Second in 9x9, 13x13 in Kanazawa 2010
- Won 4th UEC cup (19x19) in 2010

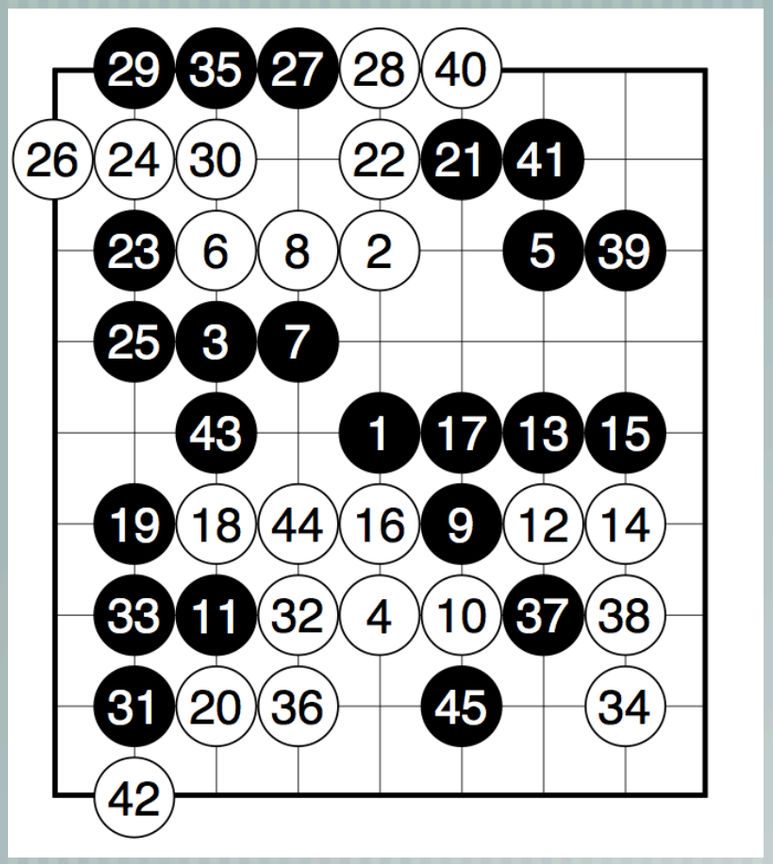
Topics of This Talk

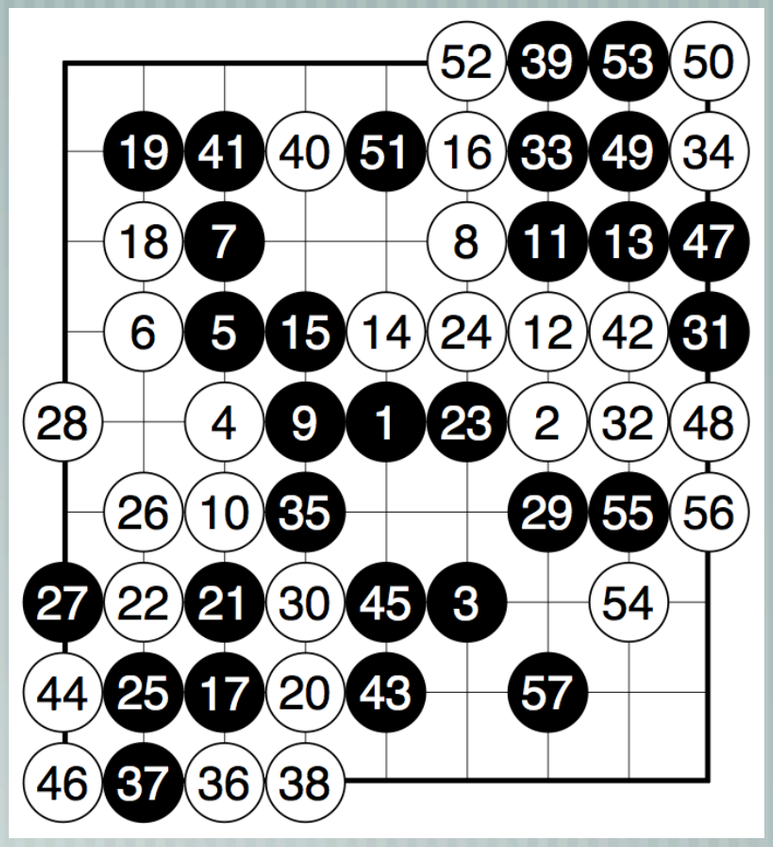
- Two limitations of current MCTS
- Take games against strong humans as examples to illustrate these problems with Fuego
- Discussion points:
  - Are these general issues with Go programs?
  - With Monte Carlo Tree Search?

Two Problems with MCTS

- I believe that in the current "standard model" of MCTS, both simulation and search processes are fundamentally flawed
  - Simulations - results do not reflect "true value" of a position
  - Search - a single global search cannot deal well with many simultaneous local complications

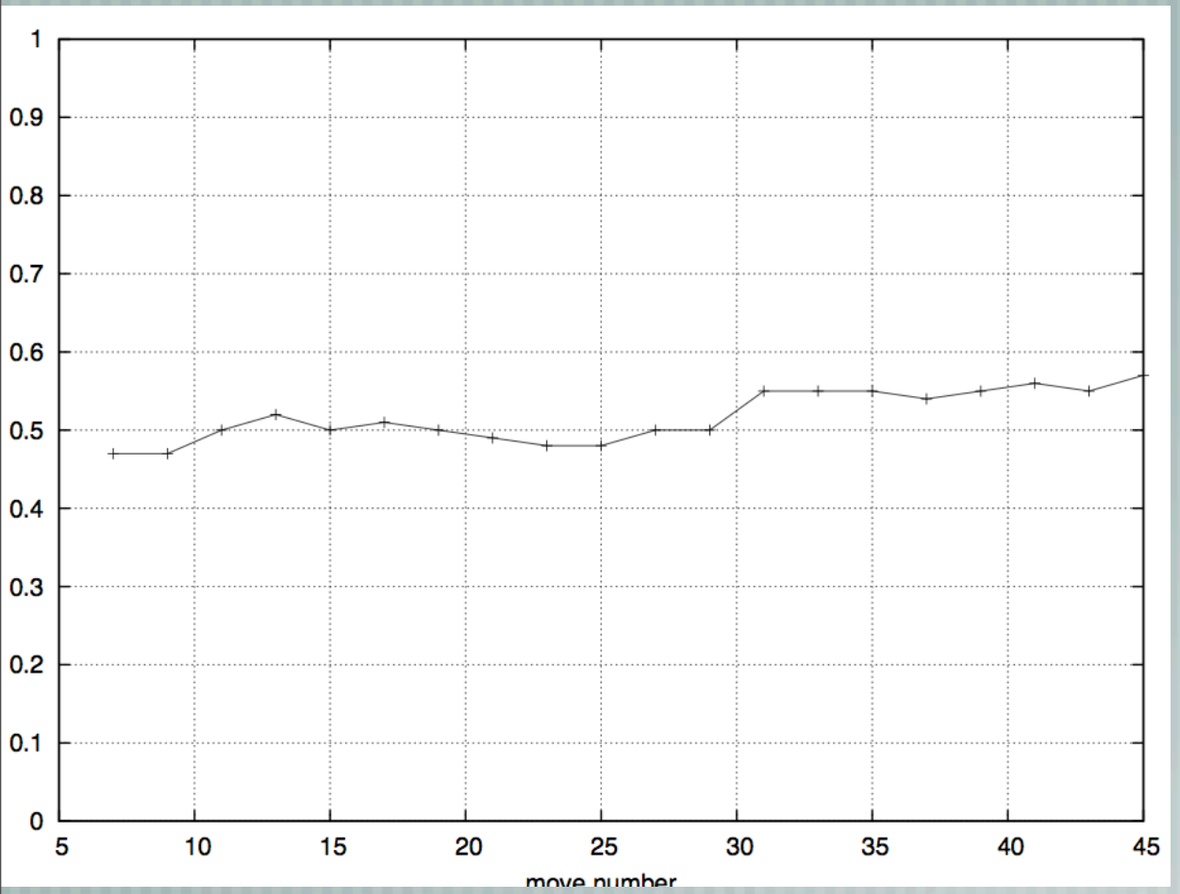
Barcelona 2010: 9x9 with Black vs Professionals

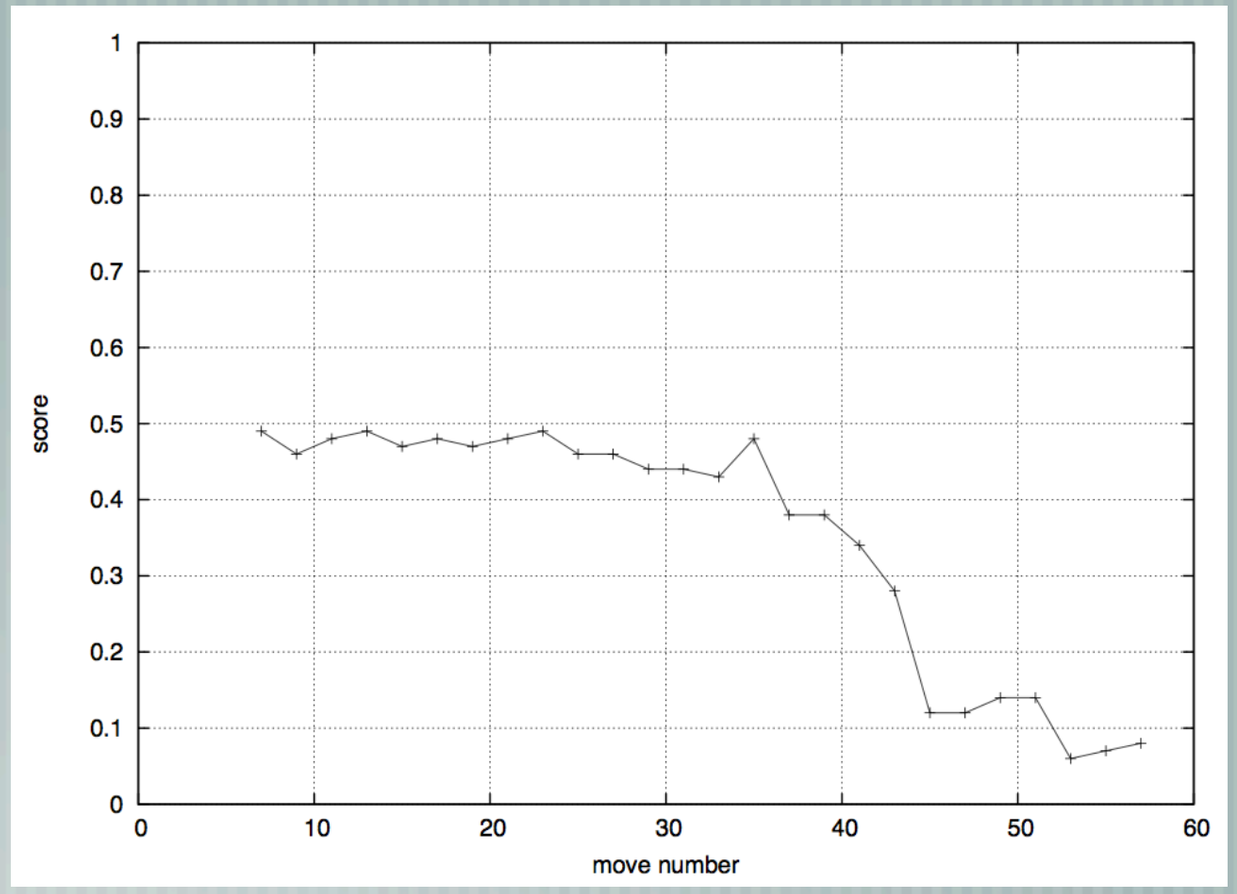




- Two quick losses, follow same pattern
- White quickly creates two safe groups (around move 10), Program does not “see” they are safe for long time

Fuego-GB Evaluation Scores



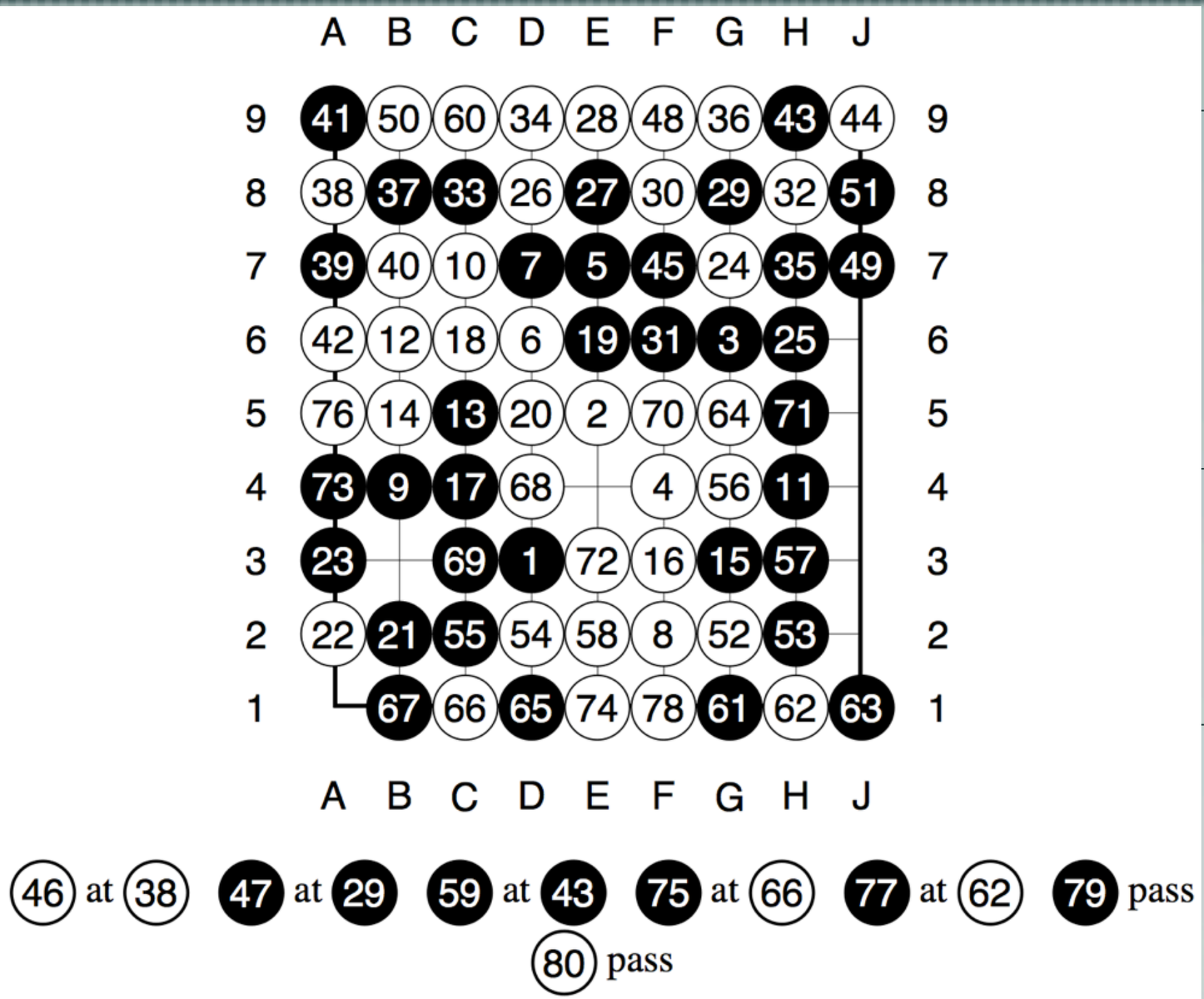


- Left - vs 4 Dan: seki misevaluation, program has no clue
- Right - vs 9 Dan: overoptimistic, game lost after 10 moves

What Goes Wrong?

- Simulations
  - systematic bias for attacker (Black here)
    - Often, one White group dies
    - I think some other programs such as *Zen*, *Valkyria* have more knowledgeable simulations
- Global Tree Search

9x9 Win with White



46 at 38 47 at 29 59 at 43 75 at 66 77 at 62 79 pass

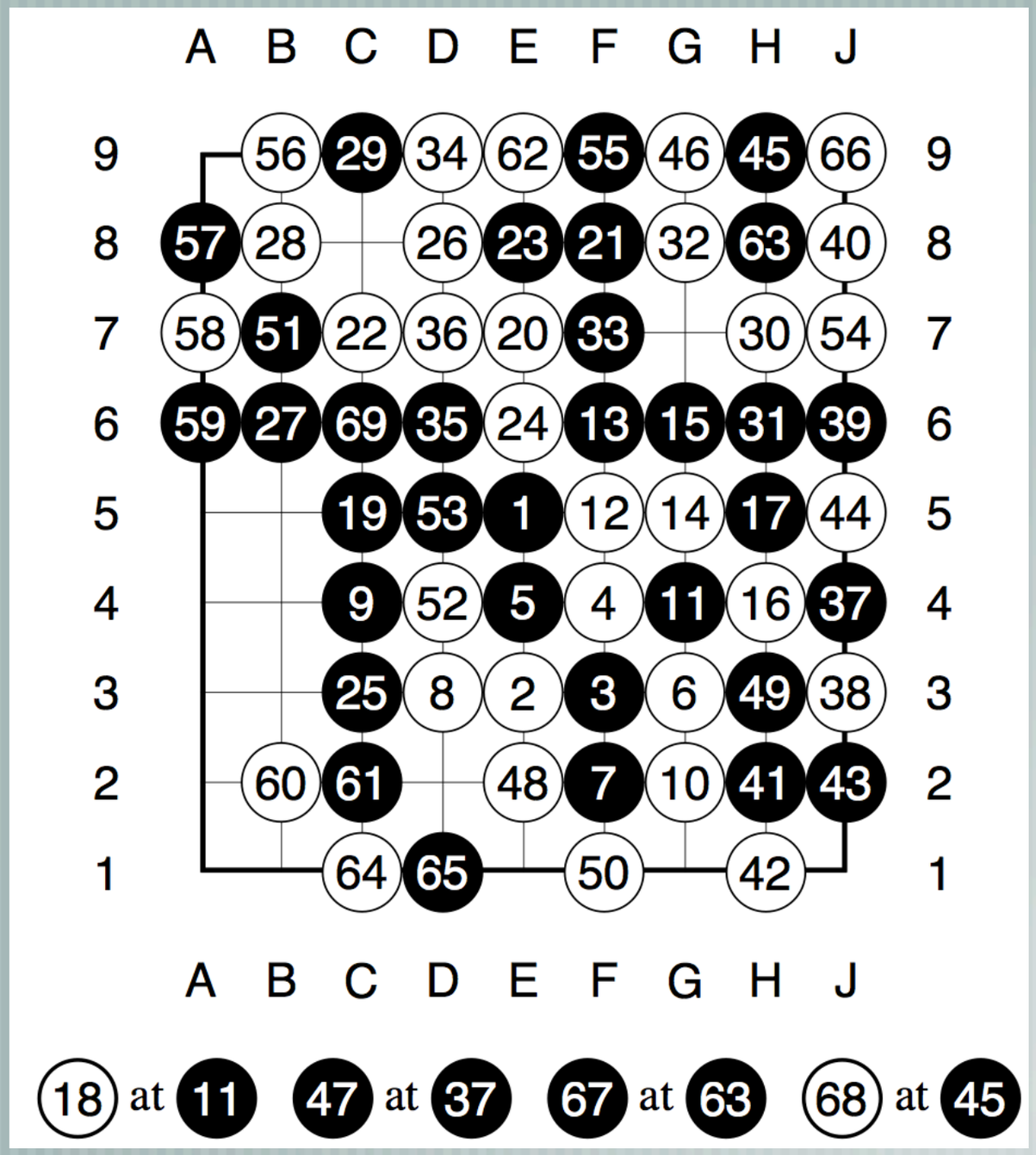
80 pass

- Difficult opening - lots of territory for human
- Good reduction in top right
- 0.5 point win for program

What Went Well?

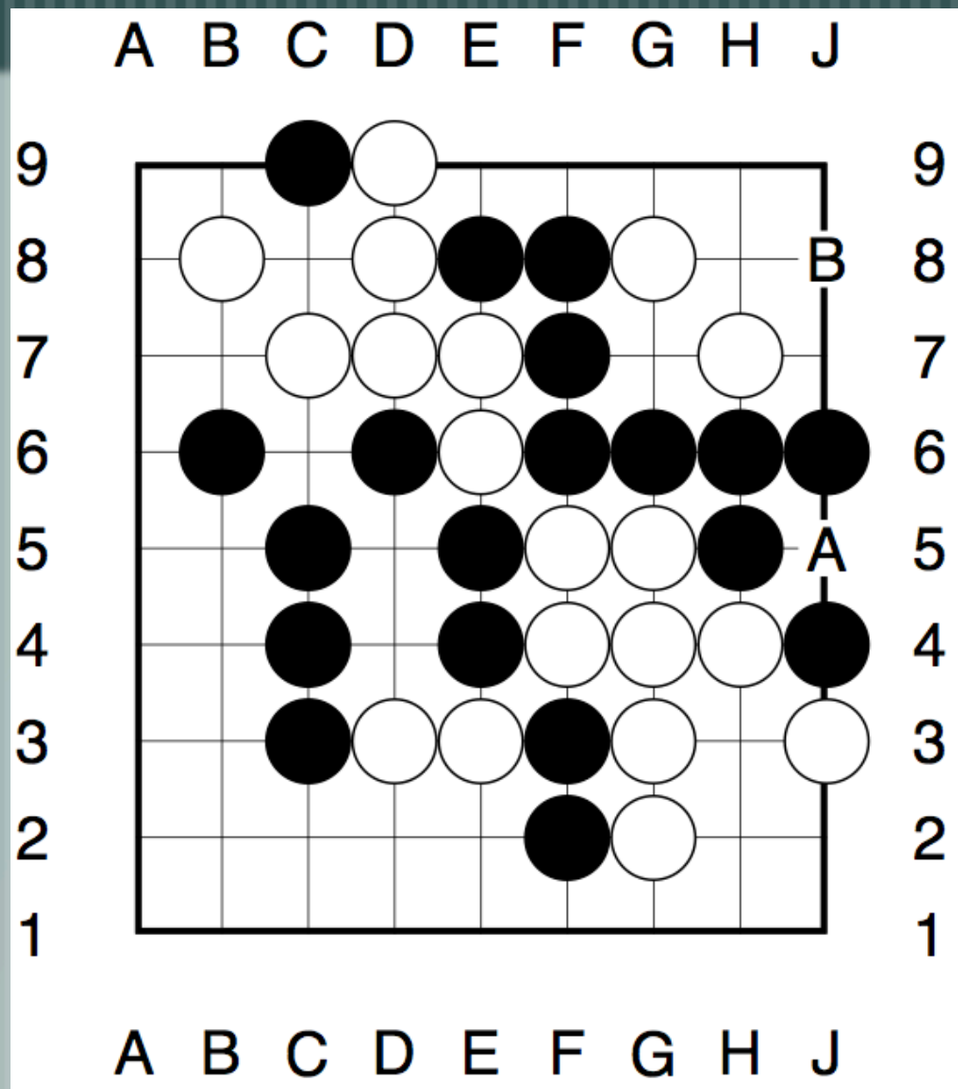
- Program knows exactly how much it needs to reduce the top right
- Single focus on the board at each time - global search does well

9x9 Loss with White vs 9 Dan

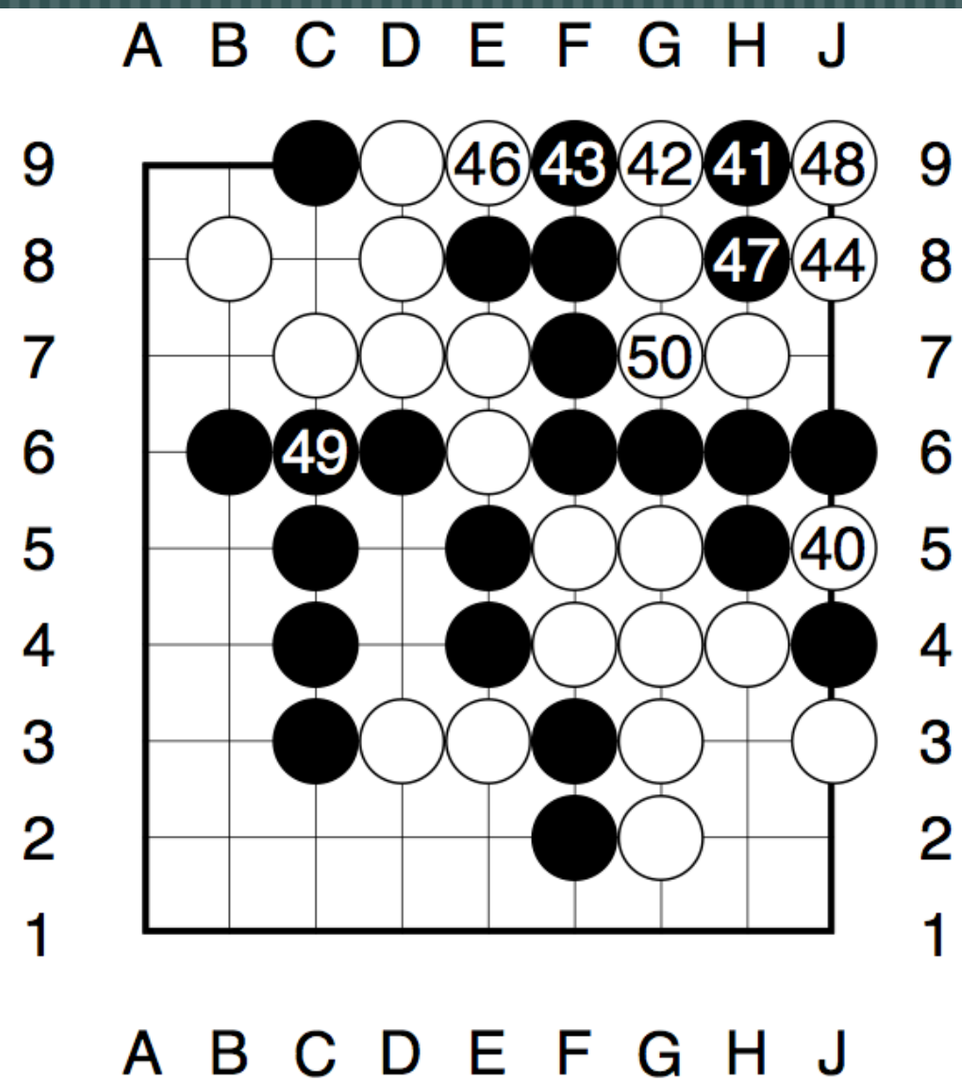


- Program played well in middle game
- Winning up to move 39
- Big fight covering 3/4 of board
- 40 is losing move - loses capturing race

Move 40: The Mistake



A would win. B loses



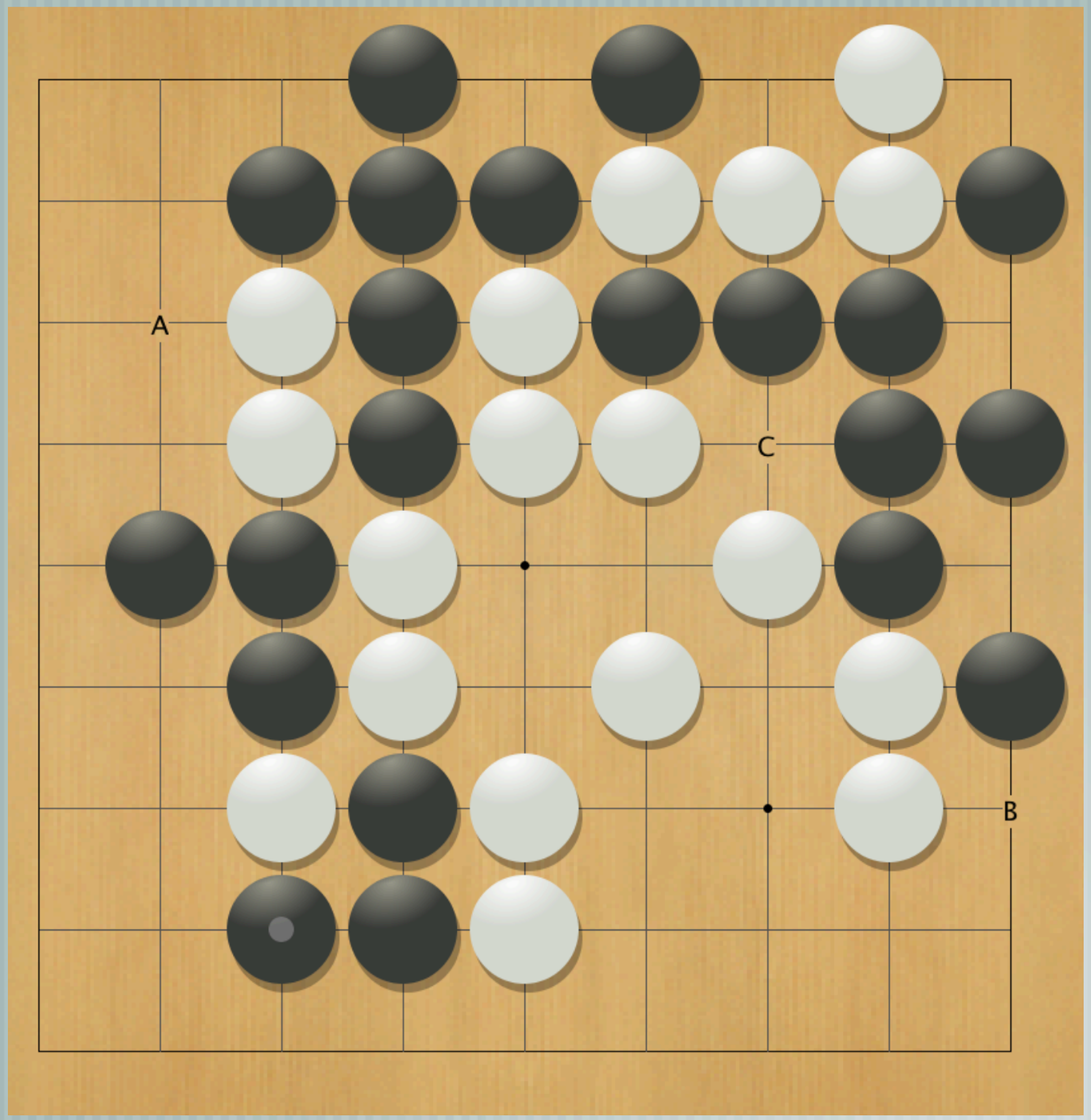
45 at J4

One possible sequence.
White wins the ko for everything

What Went Wrong?

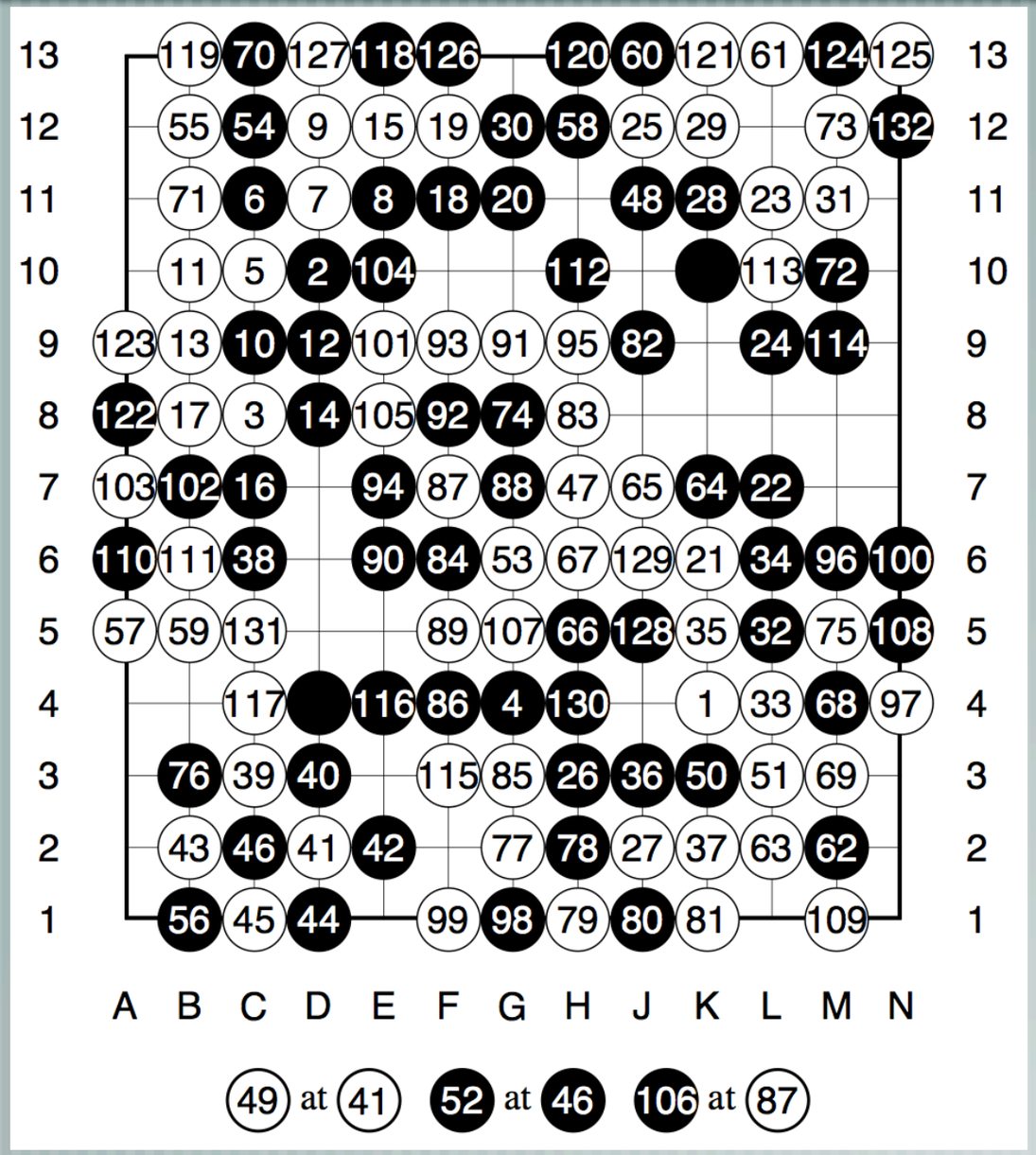
- Complex single fight involving many blocks of stones
  - Need to shift focus between top right, bottom right, top left
- MCTS too selective, misses crucial moves deep in the fight
- Human: even more selective, but based on sound Go knowledge

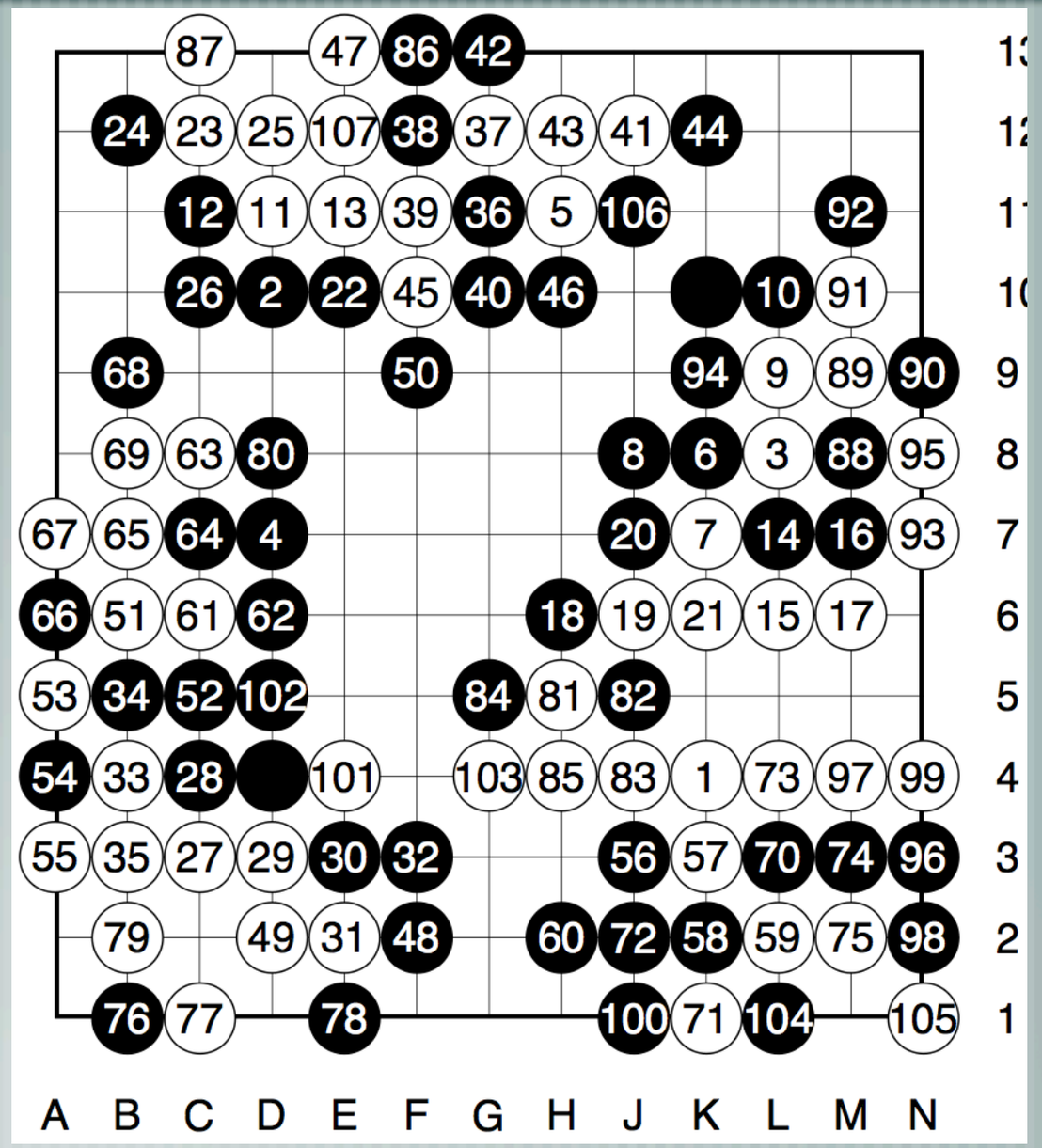
Sidebar: MoGo's Mistake



- MoGo won a good game vs 9 Dan
- Lost a good game vs 4 Dan - shown here
- White A loses semeai, B or C would win
- Similar kind of mistake?

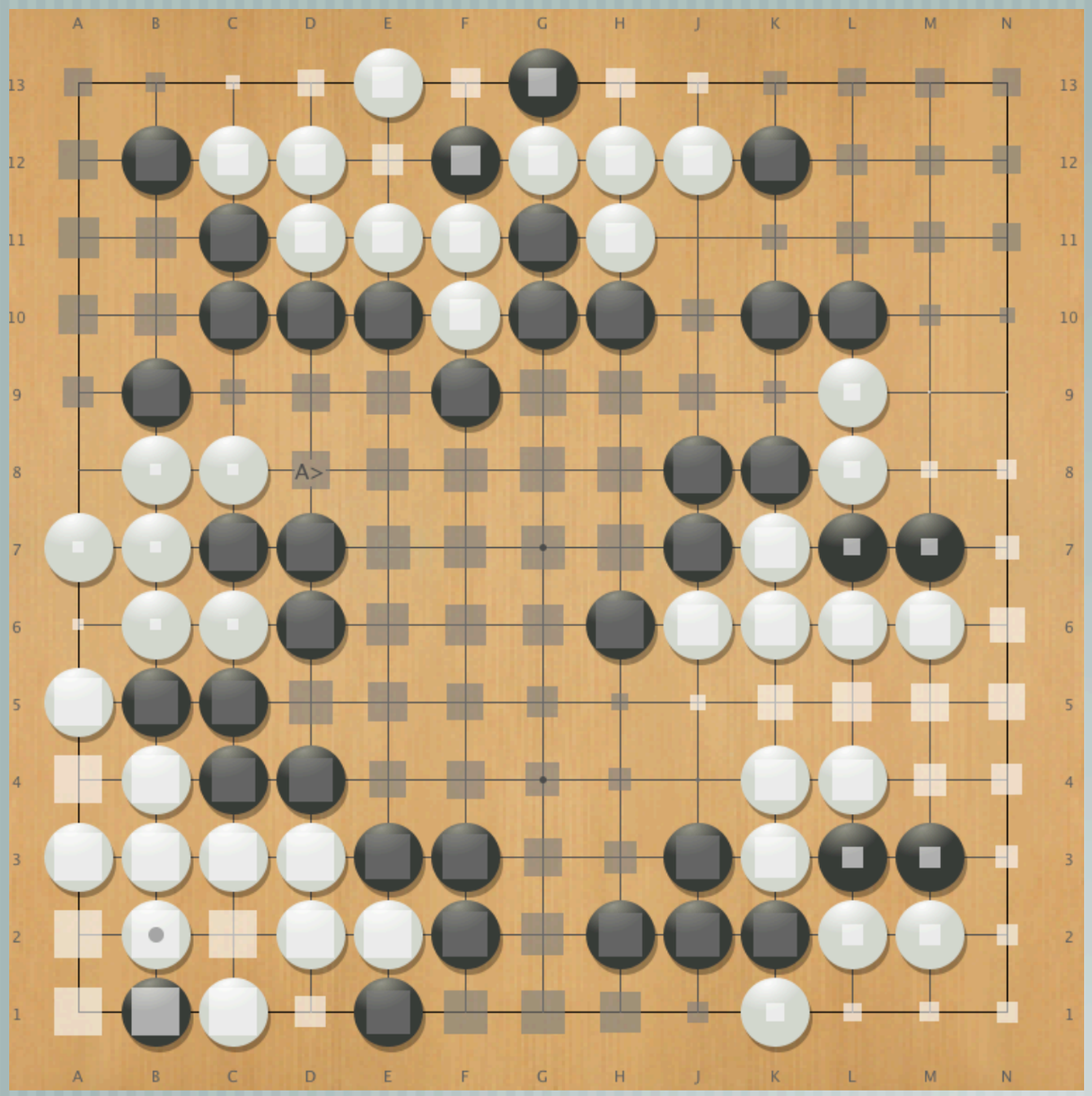
Two 13x13 Games





Left: vs Tsai 6 Dan amateur; Right: vs Yen 6 Dan amateur

Evaluation Problems



Main problem: high uncertainty about tactics in playouts

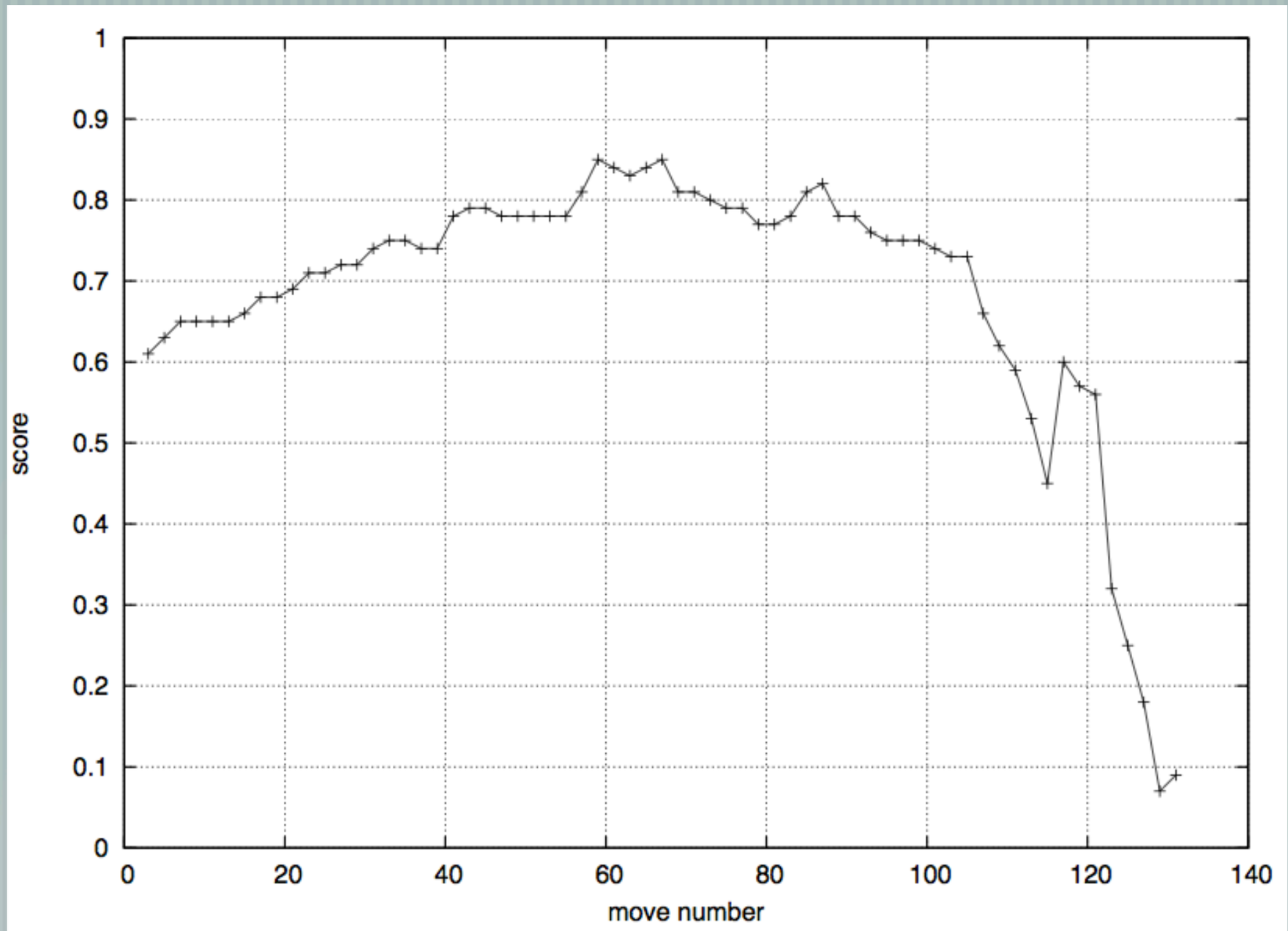
What Went Wrong?

- Randomized playouts in Fuego-GB are tactically weak
  - Outcome of capturing races is mostly random
  - On bigger boards, global search cannot cover all local fights
  - Selective search in MCTS often misses tactics

Evaluation Bias

- Each misevaluated fight introduces systematic bias of a number of points
- In both 13x13 games, all biases in same direction:
  - Program does not clearly see that opponent stones are safe
- Result: program is about 20 points off in its evaluation
  - Even 1 point would be enough to lose games

Evaluation in Game vs Tsai



Some Recent Approaches

- How to improve simulations?
- How to improve search?

Local Accuracy in Playouts

- Can we make playouts locally accurate?
  - Zen, Valkyria use much Go-specific knowledge
    - Knowledge arms race? Back to the bad old days?
  - Is this a problem specific to Go? Or a deeper, more general problem with simulations?
  - Is there a generic way to solve it?

Towards Dynamic Simulation Policies

- Tesauro, Silver: simulation balancing (offline)
- Rimmel: prefer RAVE moves in simulations
- Drake: last winning reply
- need more research

Using Domain Knowledge

- We can easily solve many tactical questions with traditional alphabeta or proof number search
- How to integrate such knowledge with MCTS?
  - Today: in-tree only
    - Hex: virtual connection solver, endgame solver
    - Go Examples: Many Faces of Go, Steenvreter, FuegoEx

Preserve Tactical Invariants

- Playouts should preserve "crucial properties" of position
- Examples:
  - Safety of territories
  - Tactics, semeai
  - Life and Death
- How to do that?

Improving on Global Search

- Global search becomes bottleneck for problems with lots of "local structure"
- Ideal: flexible combination of local and global searches
- How to do it?

Challenges and Ideas

- Find good local sequences
- Restrict search locally to those sequences
- Recent work: case study using endgame puzzles
  - Optimal player using combinatorial game theory available for evaluation
  - How to integrate with MCTS on rest of board?

Summary

- MCTS has come a long way in a very short time
- Now we seem to have hit some major road blocks
- I believe that to achieve the next level of performance, we must improve both:
  - content of simulations
  - global search