Rock'n'Roll
A Engine for the Board Game "EinStein Würfelt Nicht"^{TM}

Andreas Schäfer

Institut für Informatik
Friedrich-Schiller-Universität Jena

June 13, 2006

Outline

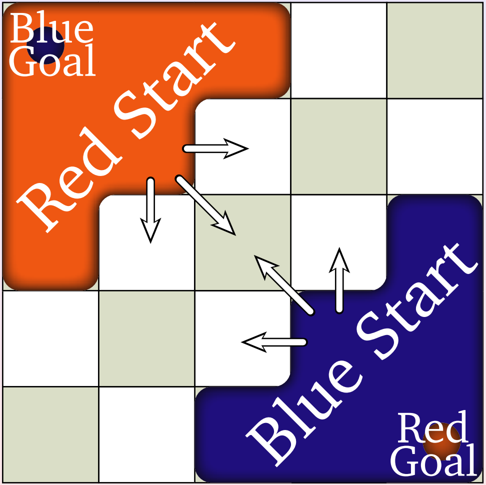
Outline

Outline

1. Introduction to EinStein
   - The Rules
2. Engine Algorithms
   - Heuristics
   - Minimax
   - Monte Carlo Method
3. Benchmarks
   - Best Parameter Sets
   - Optimal Starting Position
4. Summary

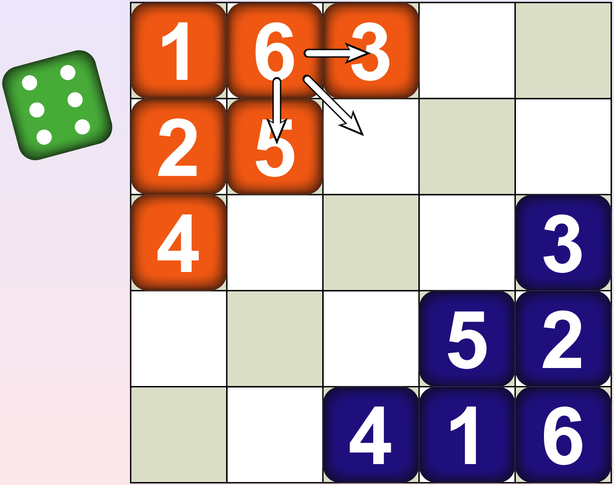
Outline

Outlook

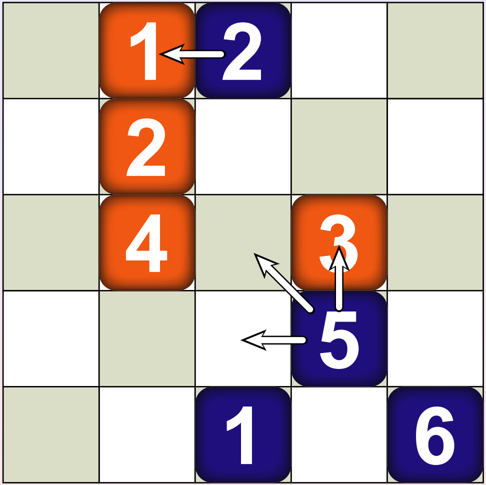


Objective:

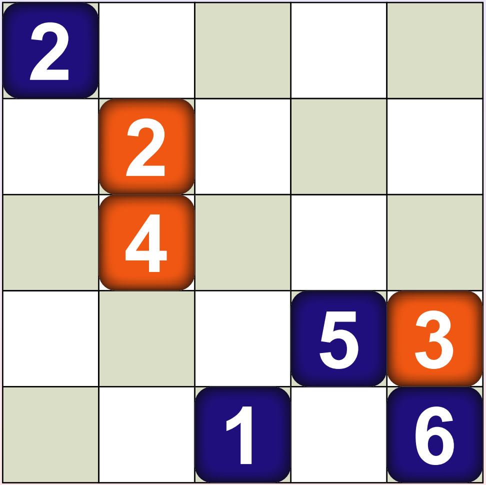
- reach goal or
- capture every of the opponent's stones



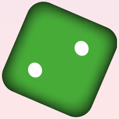
1. players take turns
2. dice determines which stone to move
3. self capturing is legal



If the rolled number is missing then an adjacent stone has to be moved



Blue won by reaching his goal



Outlook

Heuristics

- heuristic = rule of thumb,
- computer needs to choose *good* moves,
- heuristic maps boards to the reals,
- good boards *hopefully* get higher ratings

Schwarz Tables

- estimate for each player, how many moves excepted to reach goal
- difference of the two estimates is the board rating
- capturing is ignored

Example:
(one player only)

		4		

remaining moves = 4

Schwarz Tables (cont.)

Example 2:
(again just one player)

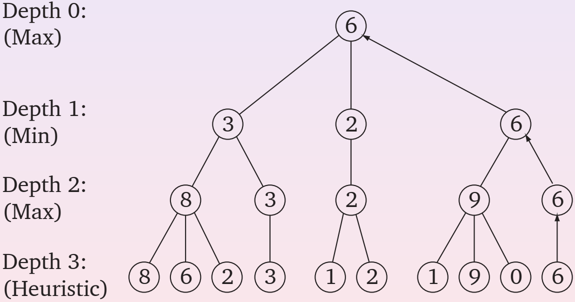
				6
			1	

$$\text{remaining moves} = \frac{5}{6} \cdot 1 + \frac{1}{6} \cdot 2 = \frac{7}{6}$$

Outlook

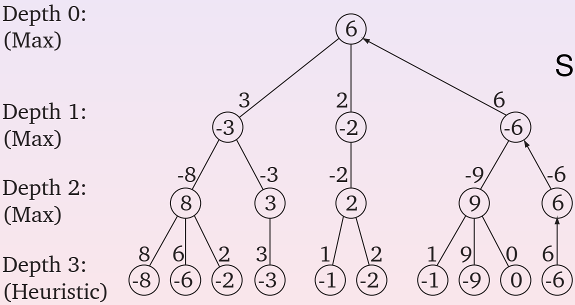
1. Introduction to EinStein
   - The Rules
2. Engine Algorithms
   - Heuristics
   - Minimax
   - Monte Carlo Method
3. Benchmarks
   - Best Parameter Sets
   - Optimal Starting Position
4. Summary

Basic Minimax



- look some moves ahead
- cut at a given depth and assign ratings
- two players:
  - Max tries to *maximize*,
  - Min tries to *minimize*
- propagate values upwards

Basic Minimax (cont.)

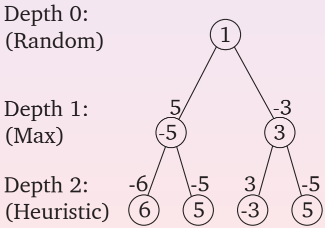


Simplification:

- invert values before propagation
- only maximizing nodes necessary

Expect Minimax

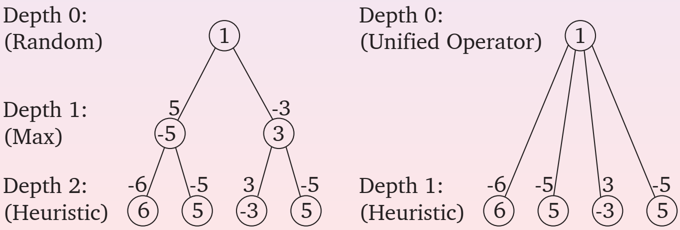
- problem: EinStein is randomized
- standard solution: introduction of dicing layers
- problem: much redundant evaluations



Dicing Layer Reduction

Unified Operator avoids unnecessary evaluations:

- calculate for each stone i the best move r_i
- determine for each stone i probability p_i to be moved
- rating $R = \sum_{i=1}^{6} r_i p_i$



Outlook

1. Introduction to EinStein
   - The Rules
2. Engine Algorithms
   - Heuristics
   - Minimax
   - Monte Carlo Method
3. Benchmarks
   - Best Parameter Sets
   - Optimal Starting Position
4. Summary

The Monte Carlo Method

board evaluation:

- 2 simple players finish game (e.g. simple heuristic)
- many repeats (e.g. 200 times)
- hopefully: win/loose ratio is correlated to move quality
- improved long-term foresight
- worse short-term foresight

Outlook

Monte Carlo vs. Minimax

	Player	1	2	3	4	5	6	7
1	Random(1)	–	11	9	10	7	7	7
2	Monte(1, 50, Schwarz(1))	88	–	45	55	44	41	40
3	Monte(1, 200, Schwarz(1))	91	55	–	62	49	46	45
4	Schwarz(1)	90	44	38	–	41	40	39
5	Schwarz(5)	93	55	51	59	–	47	46
6	Monte(3, 20, Farmer(1))	93	59	54	60	53	–	50
7	Monte(3, 40, Farmer(1))	93	60	55	61	54	50	–

- Schwarz(n) = Minimax with depth n and Schwarz Tables
- Monte(n, r, p) = Minimax with depth n and subsequent Monte Carlo (r repeats and p as simulation players)

Outlook

Optimal Starting Positions

6	2	4		
1	5			
3				

56.056 %

6	5	2		
1	4			
3				

56.021 %

1	5	4		
6	2			
3				

55.913 %

Worst Starting Positions

3	5	1		
4	6			
2				

42.175%

3	4	2		
5	6			
1				

42.446%

4	2	6		
3	1			
5				

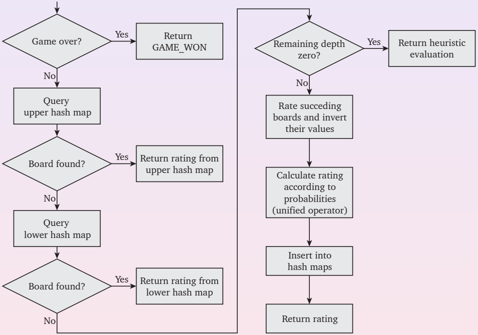
42.776%

Summary

- Unified Operator mitigates additional complexity introduced by dice
- a combination of Minimax and Monte Carlo yields strongest players
- strength limited by random component
- board setup important (1 and 6 behind other stones)

Thanks!


MISNI·IOHAN
COEPIT·DOCERE
FRIDERIC·DEI
GRA·DVX·SAXON
ELECT: LANTGR
THVR: MARCHIO


Game over?
Yes
Return GAME_WON
No
Query upper hash map
Board found?
Yes
Return rating from upper hash map
No
Query lower hash map
Board found?
Yes
Return rating from lower hash map
No
Remaining depth zero?
Yes
Return heuristic evaluation
No
Rate succeding boards and invert their values
Calculate rating according to probabilities (unified operator)
Insert into hash maps
Return rating