FairKalah: Towards Fair Mancala Play

Todd W. Neller, Taylor C. Neller

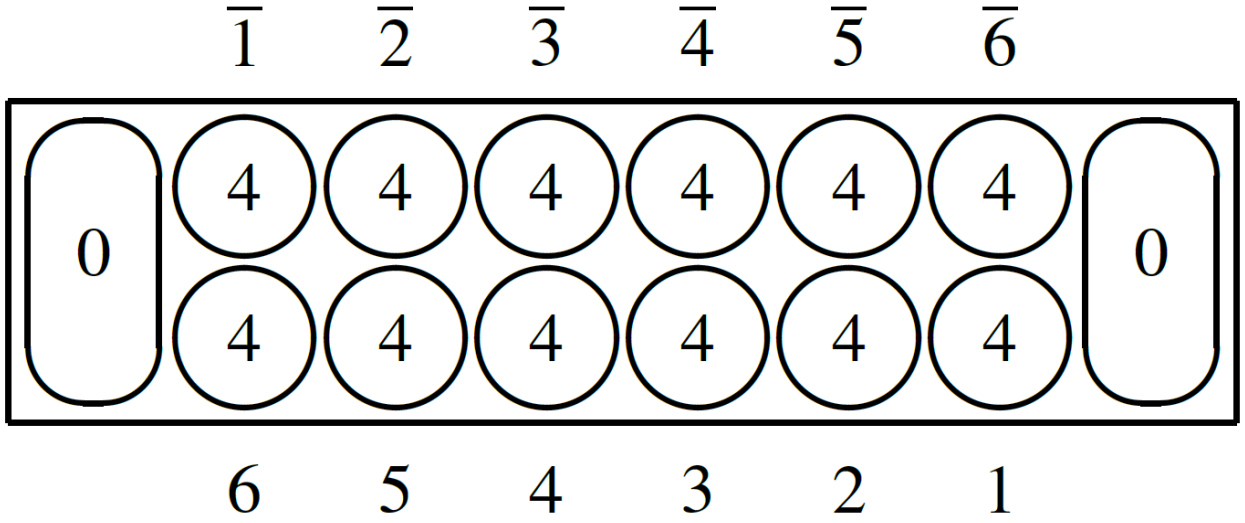
Gettysburg COLLEGE

Outline

- Mancala rules
- Mancala fair initial state (FairKalah) computation
- Fair Optimal Game Tree insights
- Diverse state optimal play modeling insights

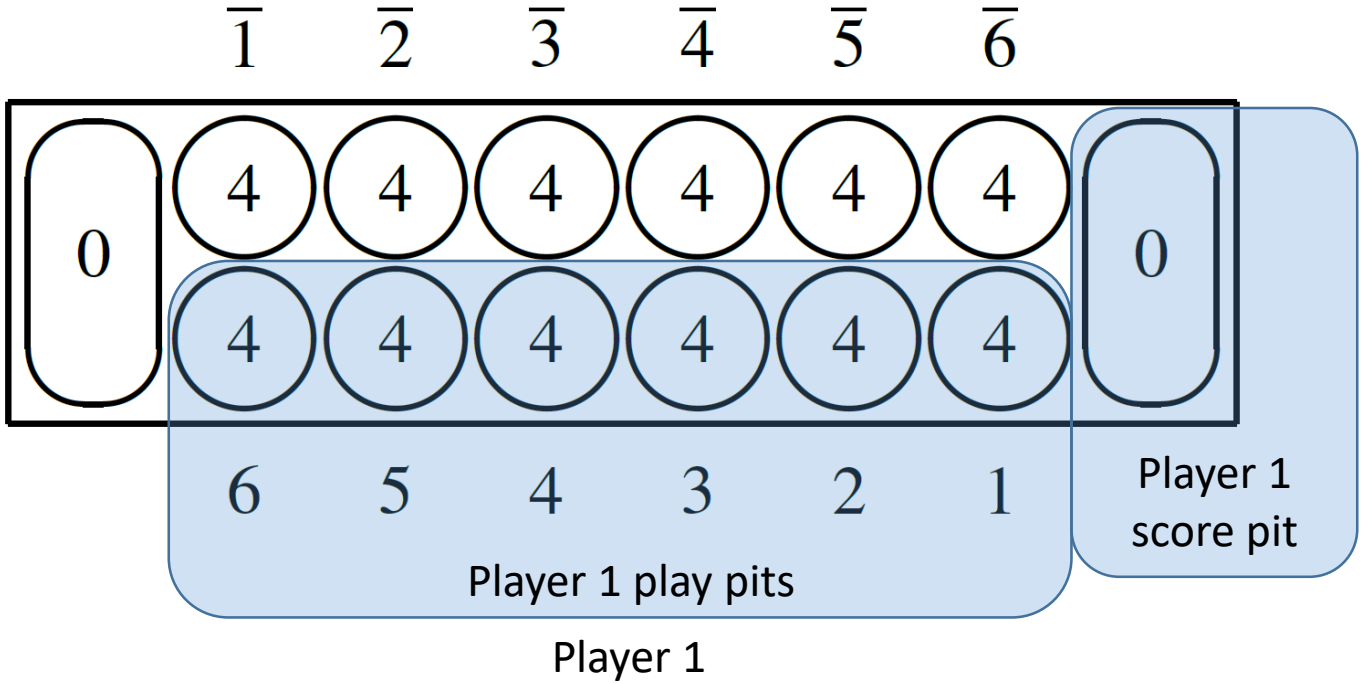
Mancala (a.k.a. Kalah) Materials

- Board with
  - 6 play pits per side for each player
  - 2 score pits, one to the right end of the board for each player
- 48 pieces initially distributed 4 per play pit in standard game



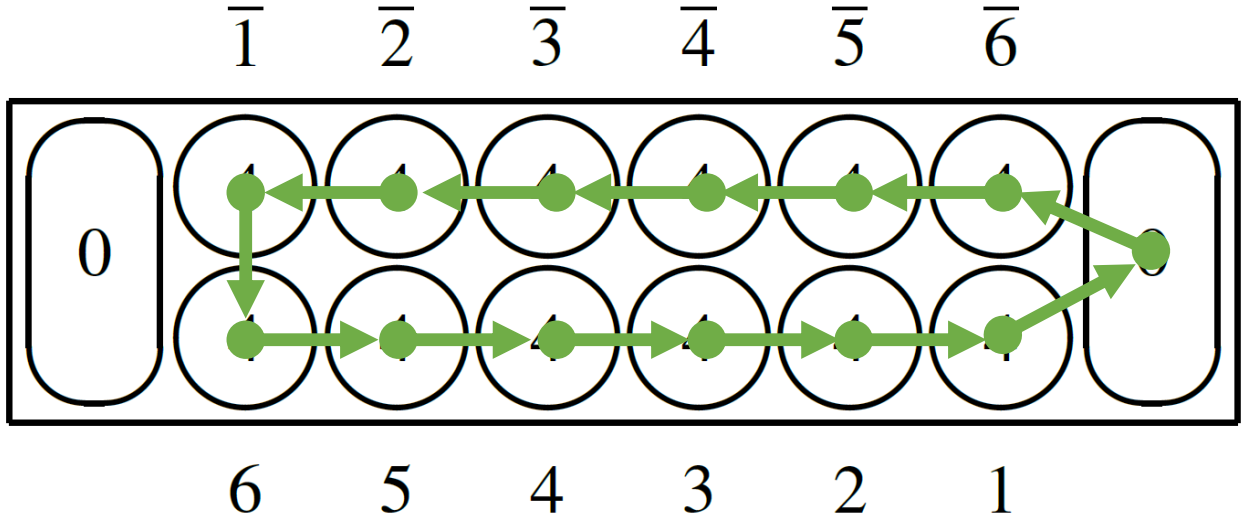
Mancala (a.k.a. Kalah) Materials

- Board with
  - 6 play pits per side for each player
  - 2 score pits, one to the right end of the board for each player
- 48 pieces initially distributed 4 per play pit in standard game



Mancala Move

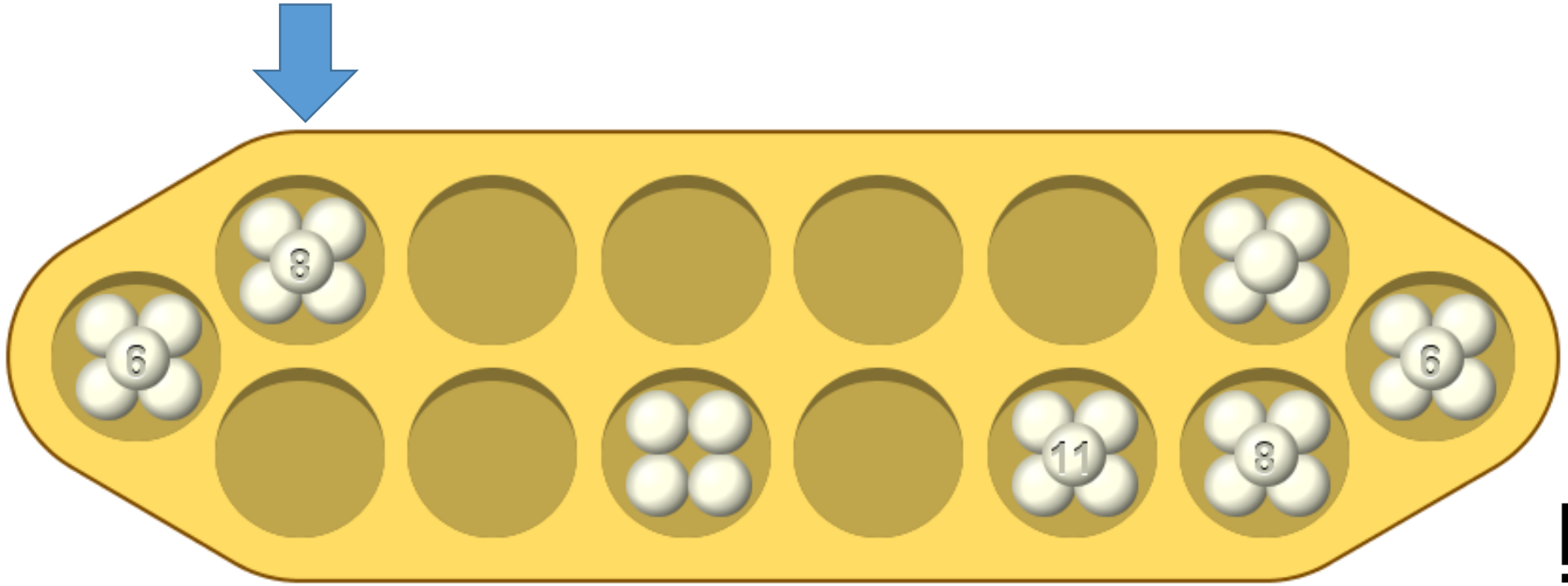
- A player's move in Mancala consists of
  - Selecting their non-empty play pit,
  - Picking up all pieces from that pit,
  - And "sowing" them counter-clockwise, one per pit, skipping the opponent's score pit.



Player 1 sowing pattern

Mancala Move Example

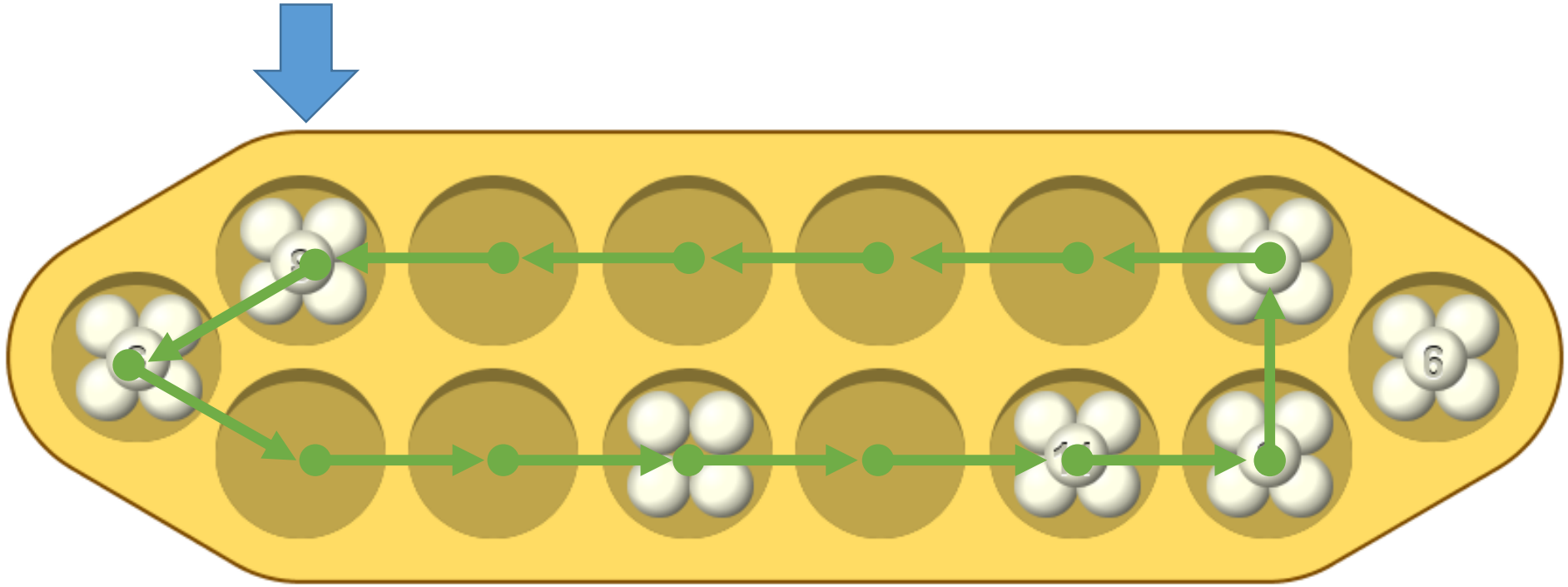
- Player 2 (top) plays 8 pieces from upper-leftmost play pit:



Screenshots from Ludii general game system:

Mancala Move Example

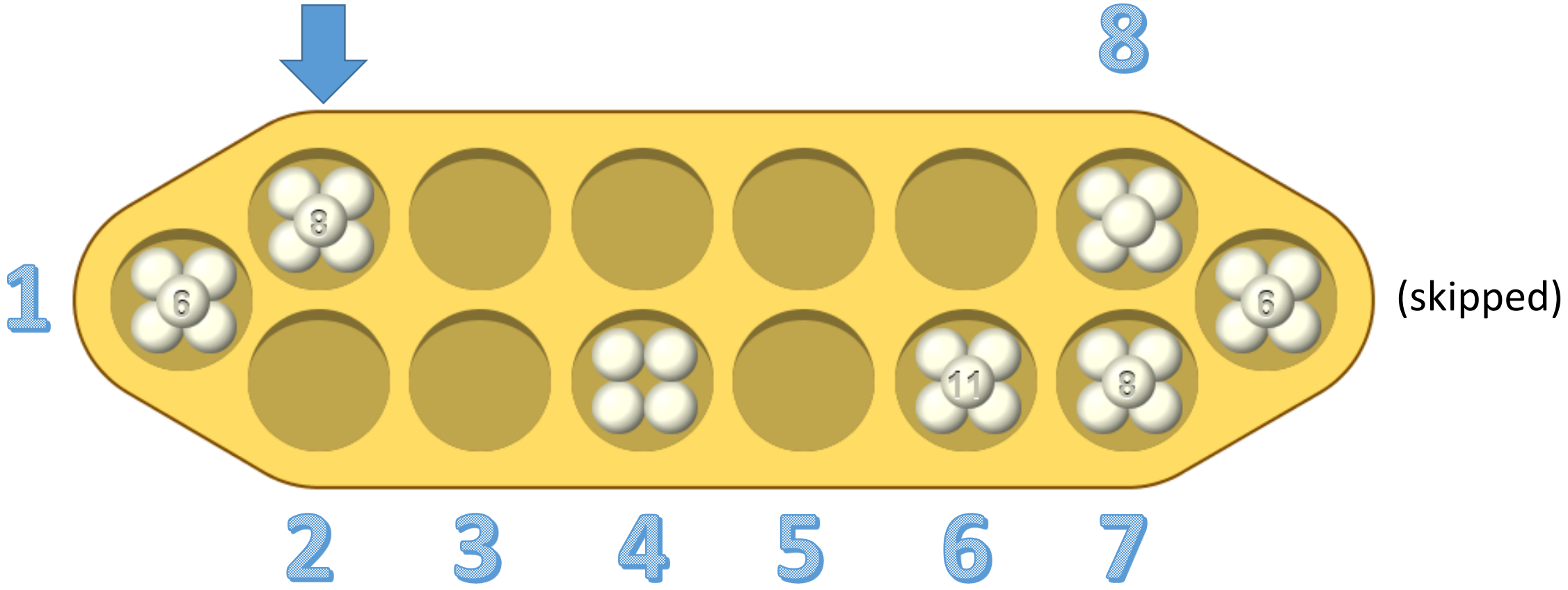
- Player 2 (top) plays 8 pieces from upper-leftmost play pit:



Player 2 sowing pattern

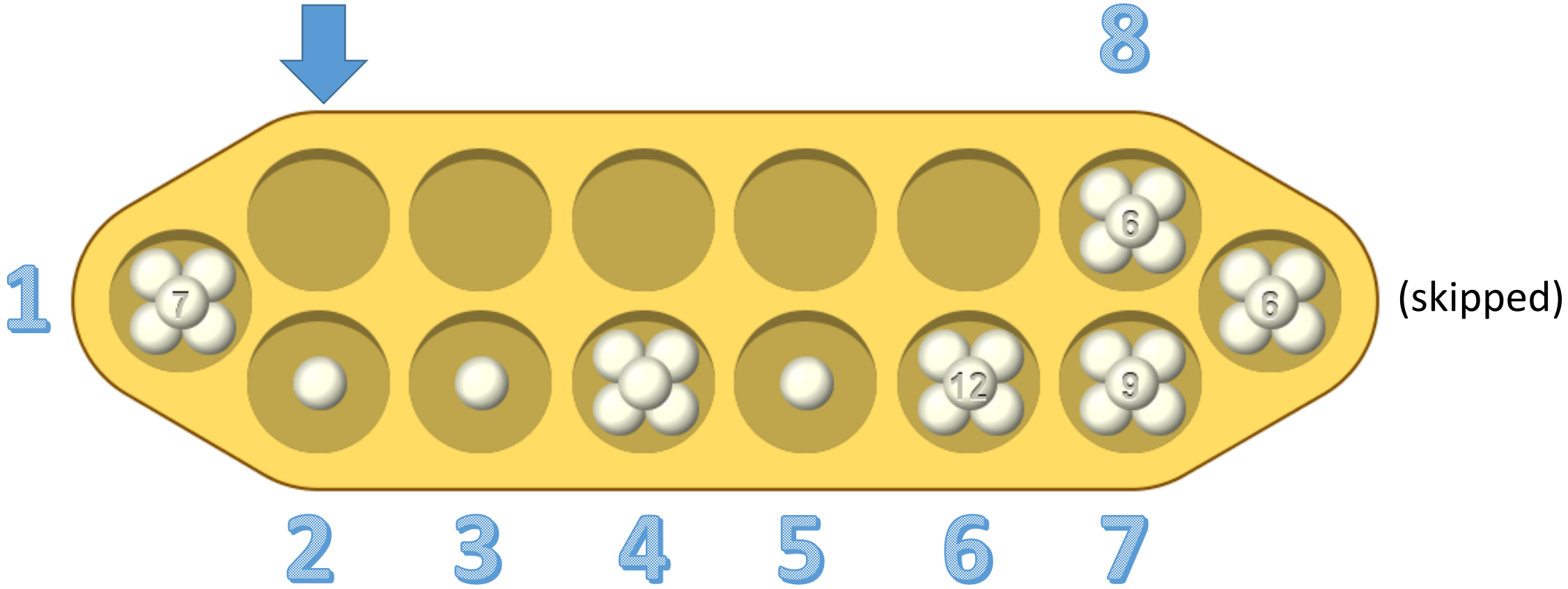
Mancala Move Example

- Player 2 (top) plays 8 pieces from upper-leftmost play pit:



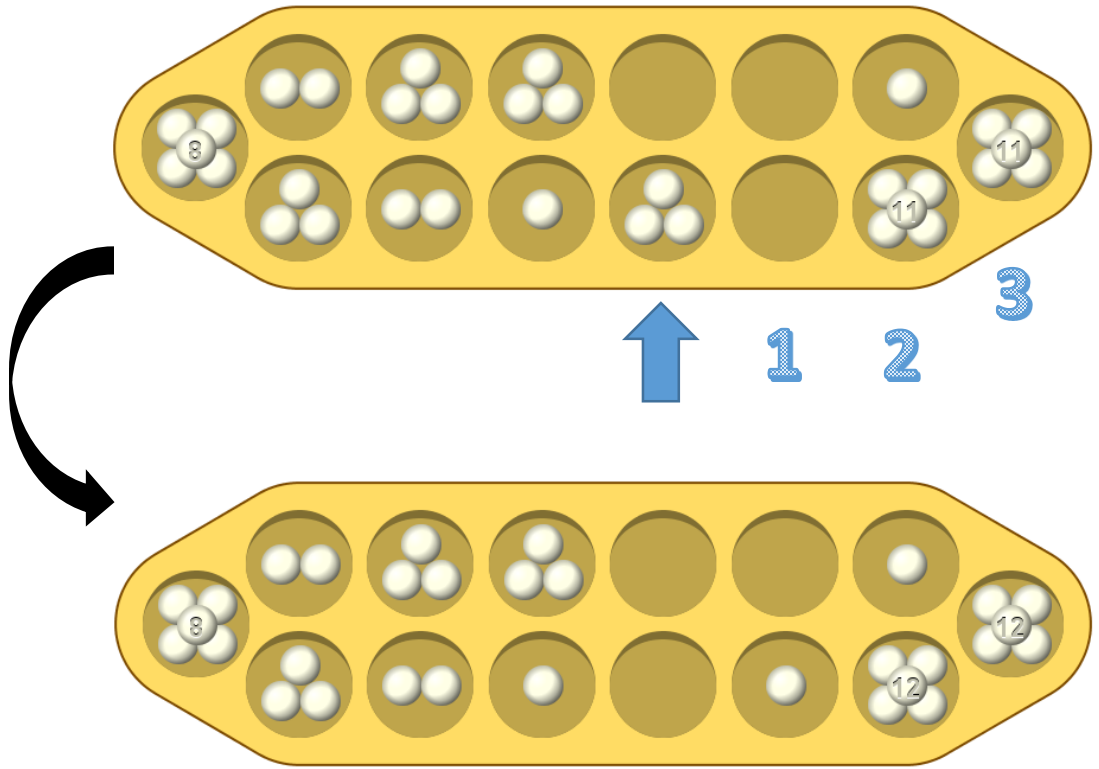
Mancala Move Example

- Player 2 (top) plays 8 pieces from upper-leftmost play pit:



Mancala Free Move

- If your last piece sown is to your score pit, take another turn.

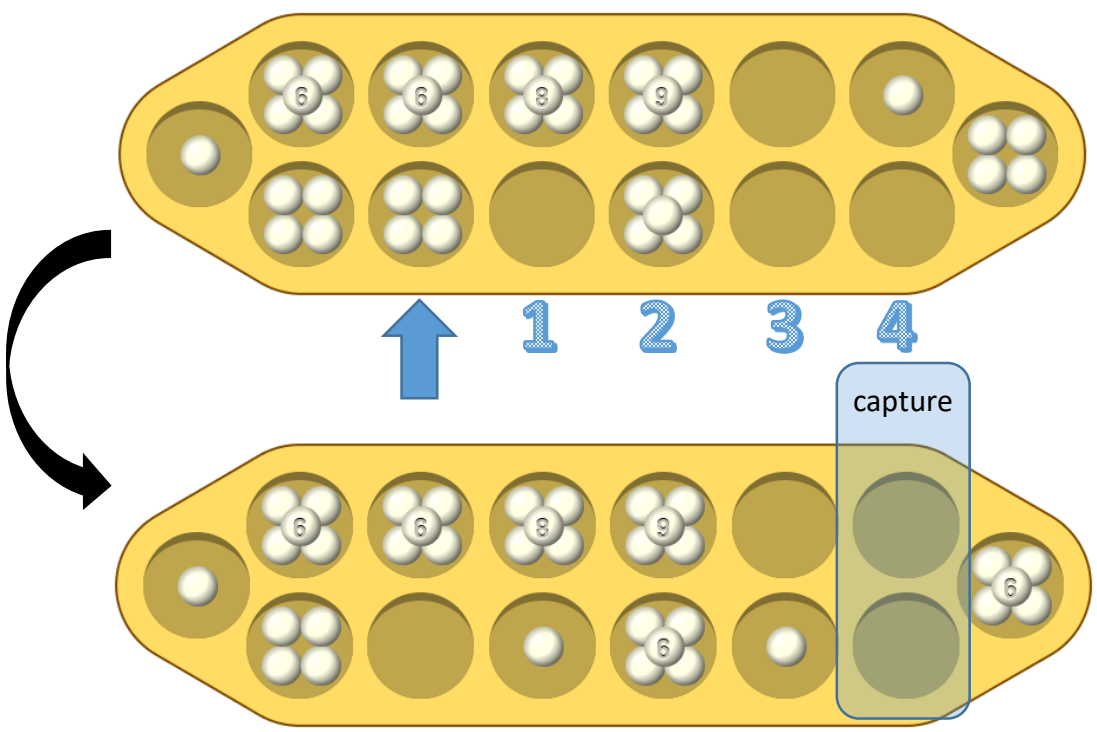


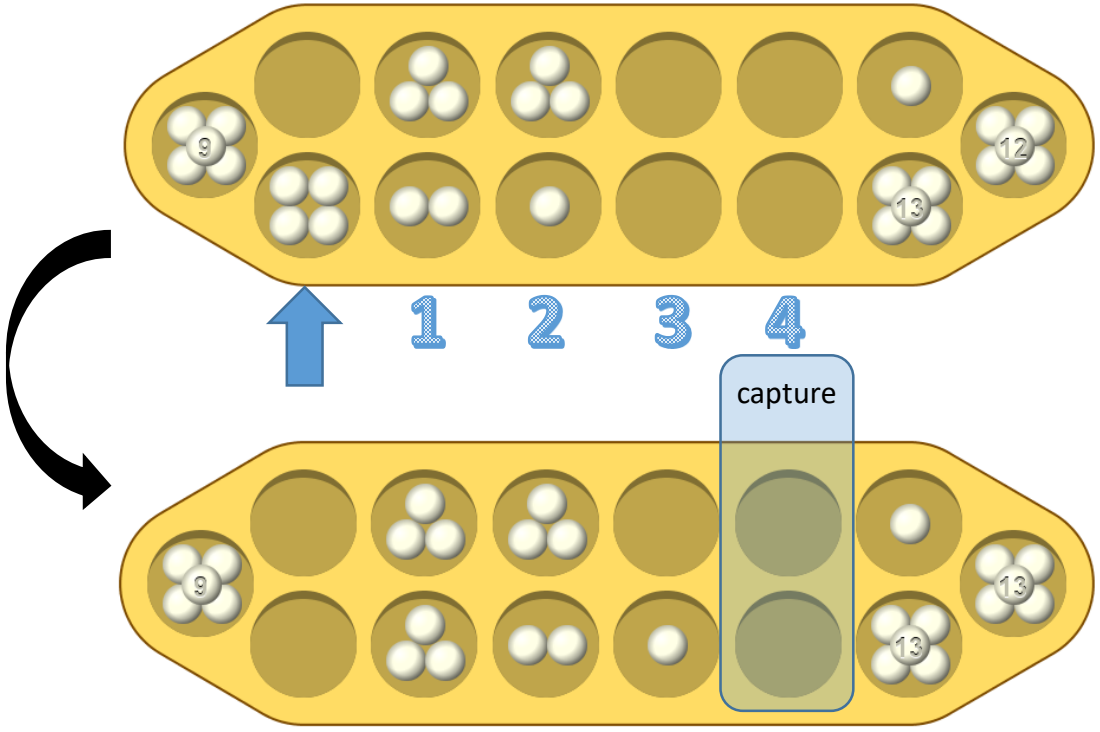
Player 1 to play

Player 1 to play again

Mancala Capture

- If your last piece sown is to an empty play pit on your side, capture that piece and any in the opponent's opposite pit (which may be empty). Captured piece(s) are placed in the player's score pit.





Mancala Game End

- “Starvation” - At the end of a turn, when no pieces remain in one player’s play pits, their opponent scores remaining play pits.
- The player that scores more pieces wins. If both players score the same number of pieces, the game is a draw (i.e. tie).

Problem: Mancala is Unfair

- The first of two perfect players will win by 10 points. (Irving, Donkers, Uiterwijk, 2000)

$m(n)$	Game value	Perfect game
4(1)	2 (W)	32-32-1
4(2)	6 (W)	3-3-230-231-1323
4(3)	8 (W)	13-1-032-2-3-0-1320-3-0-2
4(4)	2 (W)	02-2-3-0-2-21-303231-0-32-1-0
4(5)	2 (W)	0-2-1-3-0-2-1-0-32-3-021-2-0-02-2-1-3-3-313032-13031-1-2-2-3
4(6)	0 (D)	1-0-2-3-1330-01-2-2-21-201-2-23-3-23-230-32-32-12-1-0-32
5(1)	0 (D)	43-43-2-2
5(2)	0 (D)	31-32-24-0-0-31-3
5(3)	8 (W)	24-3-3-03-41-1-04342-3-3-2-2-4-414342-3
5(4)	12 (W)	12-04-03-2-20-41-1-42-32-3-2-40-0-1
5(5)	2 (W)	03-2-2-1-1-23-24-2-0-34414340-4240-2-14342
5(6)	2 (W)	2-0-0-3-3-0-2-1-440-4-42-2-121-12-3-1-4-20-0-34-24-324-1-3-3-1-41-43-2-2-3
6(1)	2 (W)	54-54-3-3-2
6(2)	10 (W)	42-42-30-0-1-1-4-5
6(3)	2 (W)	4-5-35-250-2-154-451535452-53-3-54-2
6(4)	10 (W)	25-10-3-3-5153-1-4-5-045-4-535452-53-4-1-3-2-0-54-1-3
6(5)	12 (W)	12-02-05-2-4-51-53-3-45-20-3-2-2-345-5-4-351-0-54-1-52-354-4-254-3-3

Table 9: Game values and perfect games for Kalah(m, n).

SOLVING KALAH

Geoffrey Irving[1] *Jeroen Donkers and Jos Uiterwijk*[2]

Pasadena, California Maastricht, The Netherlands

ABSTRACT

Using full-game databases and optimized tree-search algorithms, the game of Kalah is solved for several starting configurations up to 6 holes and 5 counters per hole. The main search algorithm used was iterative-deepening MTD(f). Major search enhancements were move ordering, transposition tables, futility pruning, enhanced transposition cut-off, and endgame databases.

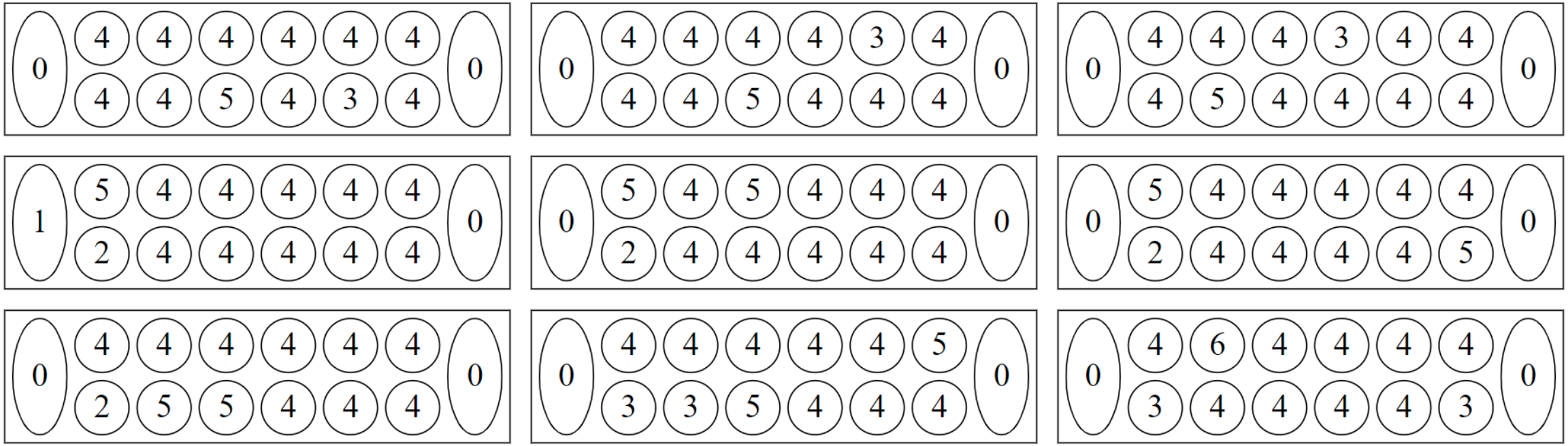
1. INTRODUCTION

Kalah is a modern, commercial variant of Mancala, introduced in the 1950s by a firm called "The Kalah Game Company" (owned by W.J. Champion). It has gained a large popularity especially in the United States and is still played in pubs and at home. In 1960, a first computerized version of the game was produced and many others followed. Remarkably, Kalah has a relatively long history in Artificial Intelligence: Bell (1968) already used Kalah to demonstrate game playing by a computer, and Slagle and Dixon (1970) used Kalah to illustrate their M & N search algorithm. Nowadays Kalah is often used as an example game in computer-science courses.

The term 'mancala' is used to indicate a large group of related games that are played almost all over the world (Murray, 1952; Russ, 2000). Mancala games (also known as 'pebble-and-pit games' or 'count-and-capture games') are played on a board that contains 2, 3 or 4 rows of holes. Sometimes these holes are simply dug in the soil or drawn on paper. Often there are two or four additional holes (called *stores*) with a special meaning. The games are usually played by two players, although one-player and three-player versions are known. Mancala games are played with a large set of equal counters. These counters can be pebbles, shells, seeds or any small round objects. The game starts with a certain distribution of the counters over the pits (usually an equal number per hole). A move is made by selecting one of the holes, lifting all counters out of it and putting back the counters one-by-one in adjacent holes in a certain direction. This is called 'sowing'. The hole in which the last counter is put determines what happens next. Sometimes a capture takes place and the turn is over, sometimes the sowing continues, and other times the player is allowed to do another move. The goal of the game is always to capture as many counters as possible.

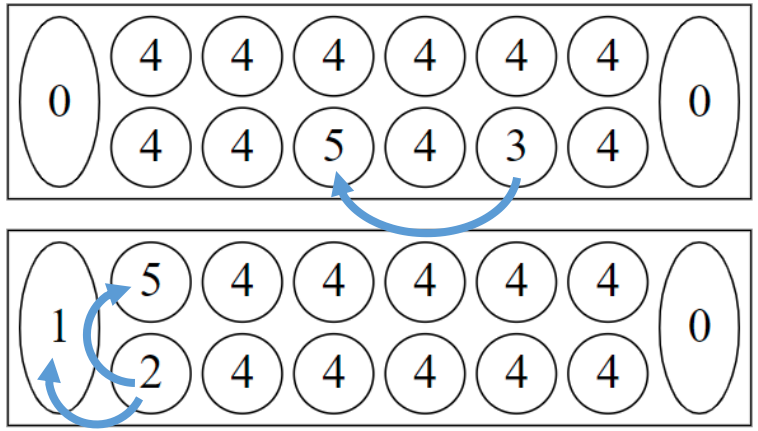
Solution: FairKalah – fair initial board states

- We have computed 254 initial states with 48 pieces arranged to be *fair*, i.e. two perfect players are proven to draw.
- This makes improvements to heuristic functions more apparent, as Mancala's unfairness obscured relative player strength.



Computation of FairKalah Boards

- For each possible board resulting from 1 or 2 pieces moved from the standard initial position, find all boards with 0 game value (1st player score – 2nd player score).
- Port of Irving's C code to Java with
  - 24-piece endgame database (1.16 GB)
  - MTD(*f*) algorithm
  - Heuristic node ordering: transposition table move, closest free moves, closest captures, closest remaining
- 3 and 251 fair boards with 1 and 2 pieces moved respectively

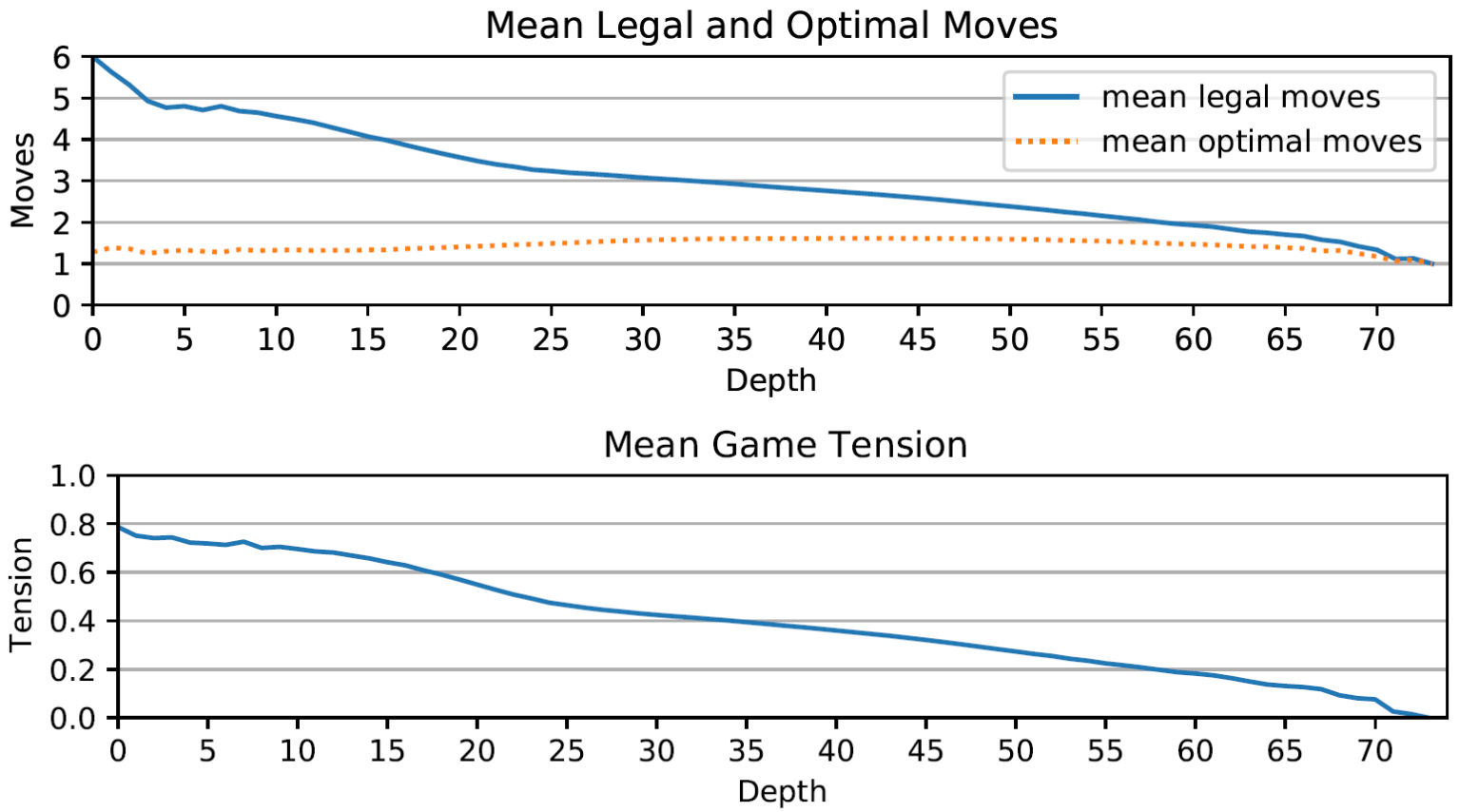


Fair Optimal Game Tree Analysis

- Fair Optimal Game Trees – subtrees of fair game tree with optimal play, i.e. all game value 0 (draw) states.
  - Pro: information about perfect play state distribution
  - Con: predictors lose performance on suboptimal game states
- MTD(*f*) analyses yielded 9,991,466 unique nonterminal optimal play states
  - 28,366,064 total plays and 15,828,178 optimal plays, yielding average play and optimal play branching factors of approximately 2.839 and 1.584, resp.
  - number of states with 1-6 optimal moves are, resp.: 5,619,419 (56.2%), 3,188,759 (31.9%), 933,219, 219,859, 29,112, and 1,098.
  - Thus, 88.2% of states have 1-2 optimal plays.

Fair Optimal Game Tree Analysis (cont.)

- With increasing depth:
  - Mean legal moves decreases from 6 to 1
  - Mean optimal moves is relatively steady
  - Thus, mean game tension decreases from .8 to 0. (.833 = unique optimal opening move)
- Conjecture: Time-management should allocate more time to opening play.



Fair Optimal Game Tree Insights

- 9,991,466 (100%) unique FOGT states:
  - 5,558,640 (55.63%) with free move(s): In 5,326,495 (**95.82%**) of these, the **closest free move is optimal**.
  - 4,432,826 (44.37%) with no free move: In 3,777,047 (85.21%) of these, capture move(s) are possible. Of these, 3,212,373 (85.05%) have an optimal capture move. Thus, **if there is no free move, in 72.47% of states, a capture move is optimal**.
  - The closer the pit, the larger the percentage of states have that pit as an optimal play.
- This supports the node ordering heuristic of Irving et al.: transposition table move, closest free moves, closest captures, closest remaining

Pit	States Optimal	Percent
1	3568534	35.72%
2	2945674	29.48%
3	2640664	26.43%
4	2527830	25.30%
5	2333925	23.36%
6	1811551	18.13%

Analysis of Diverse Play Data

- FOGT insights generalized poorly to non-fair states
- New diverse dataset:
  - First, randomly score 5 pieces
  - Breadth-First-Traversal: limited to 100 nodes per depth
  - 9,449,283 unique states, 6,151,746 after removing player and depth
- 50-50 train/test split
- Modeling packages used:
  - Linear and Logistic Regression (scikit-learn)
  - DecisionTreeRegressor/Classifier,
    RandomForestRegressor/Classifier (scikit-learn)
  - CatBoostRegressor/Classifier (CatBoost)

Game Value Regression Modeling

- Regression aims to predict “game value”
  - “game_val”: The current player score minus the opponent score if the game was played to completion with perfect play

Model	MSE	R^2
Linear regression	23.69	0.81
Decision tree	43.84	0.63
Random forest	5.70	0.95
CatBoost	4.37	0.96

- Hyperparameters:
  - DecisionTreeRegressor: min_samples_split=500000
  - RandomForestRegressor: n_estimators=100
  - CatBoostRegressor: iterations=30, learning_rate=0.9, depth=16.

Optimality Classification Modeling

- Regression aims to predict individual pit optimality, e.g. whether that move is optimal
- Pits with 0 pieces pruned from analysis
- Same hyperparameters
- Base predictor:
  - Accuracy/log loss if predictor guesses that for a pit *x*, chance of optimality is 50%
- Log loss: stronger than accuracy

Model	6	5	4	3	2	1
Base predictor	.68	.68	.67	.65	.57	.58
Logistic regression	.80	.80	.78	.77	.77	.86
Random forest	.87	.86	.86	.85	.86	.89
CatBoost	.89	.88	.88	.87	.87	.92

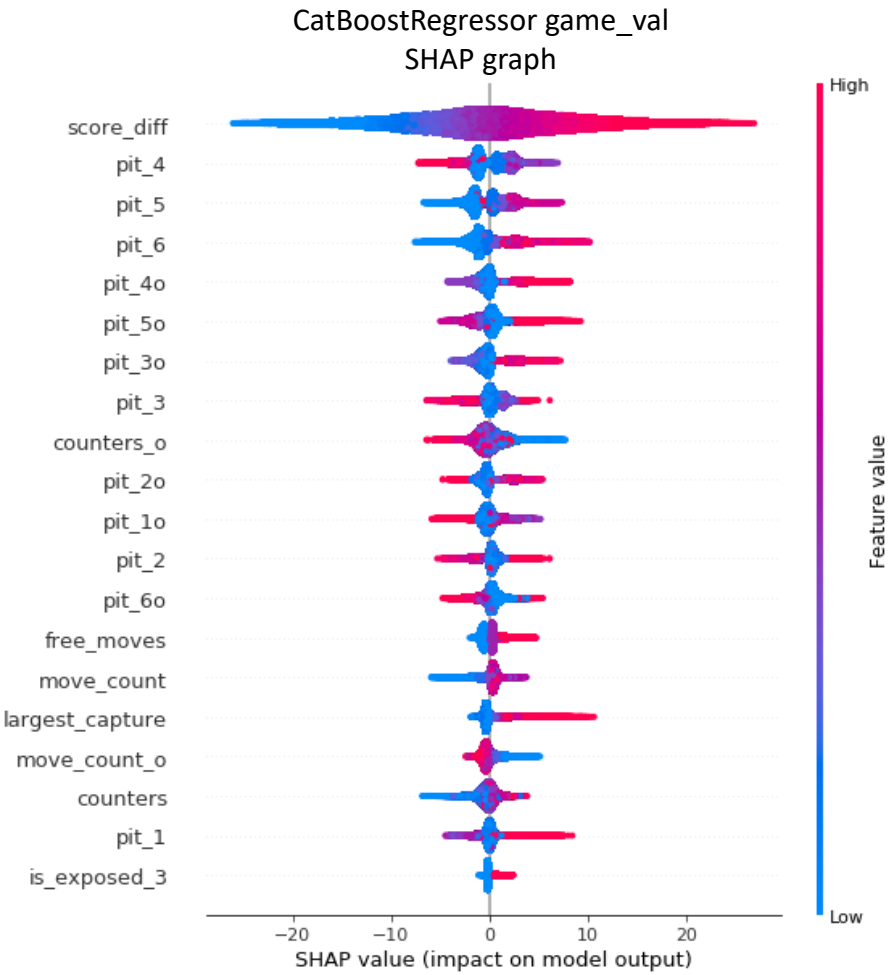
Classification Accuracies

Model	6	5	4	3	2	1
Base predictor	.63	.63	.63	.65	.68	.68
Logistic regression	.43	.44	.47	.48	.46	.31
Random forest	.31	.32	.33	.33	.32	.23
CatBoost	.27	.28	.28	.29	.29	.21

Classification Log Losses

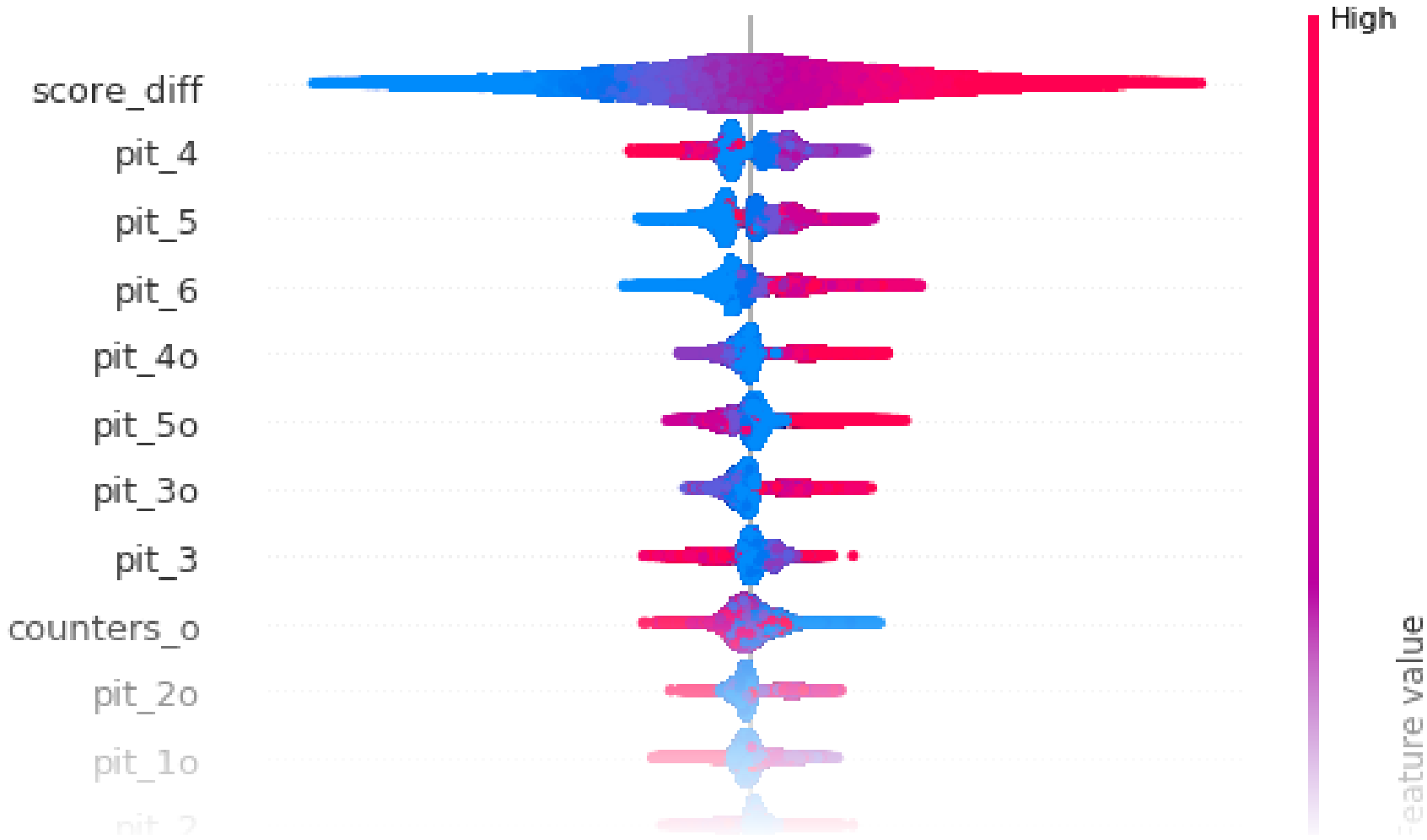
Game Value SHAP Interpretation

- SHapley Additive exPlanation (SHAP) values:
  - “assign each feature an importance value for a particular prediction.”
- Feature ordering is by significance, point density indicates number of occurrences
- X axis: effect on model prediction
- Interpretations:
  - Score difference highly correlated
  - pits 4, 5, 6, $\overline{4}$, $\overline{5}$, $\overline{3}$, and 3 are all strong predictors of game value



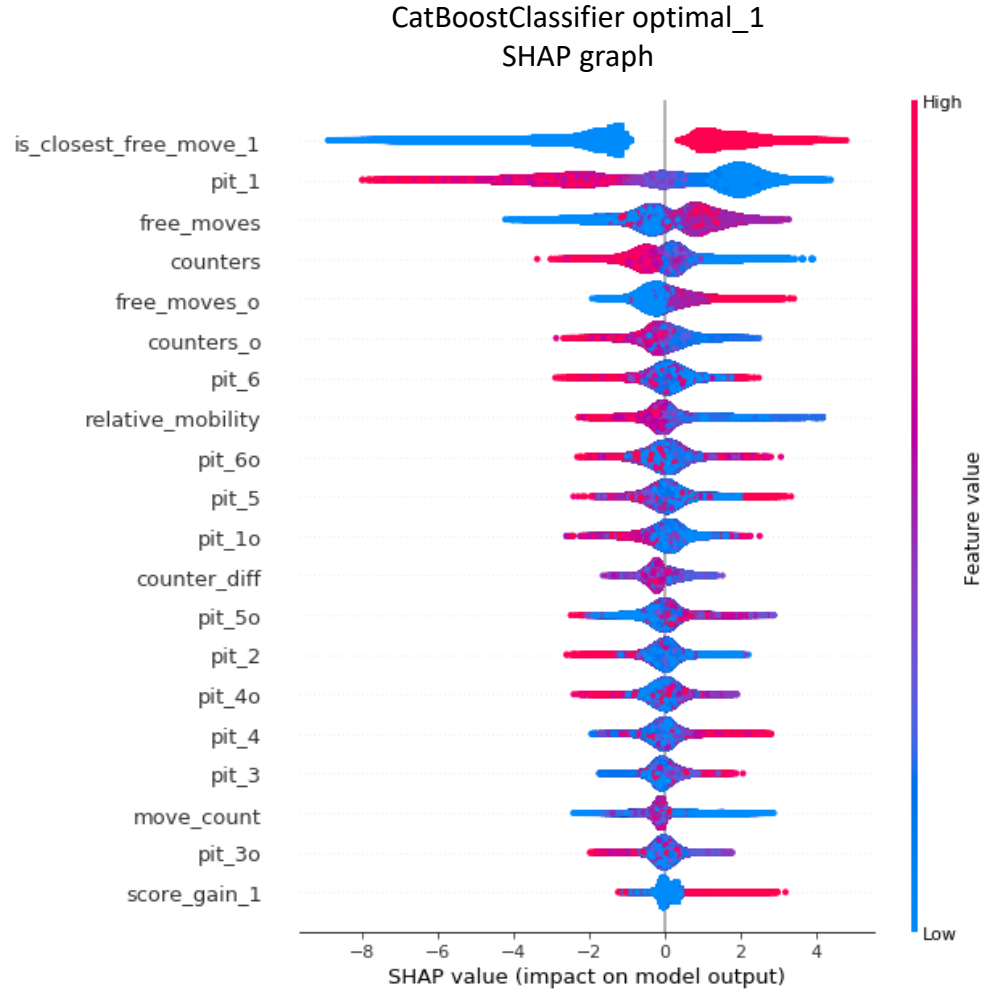
Game Value SHAP Interpretation

CatBoostRegressor game_val SHAP graph



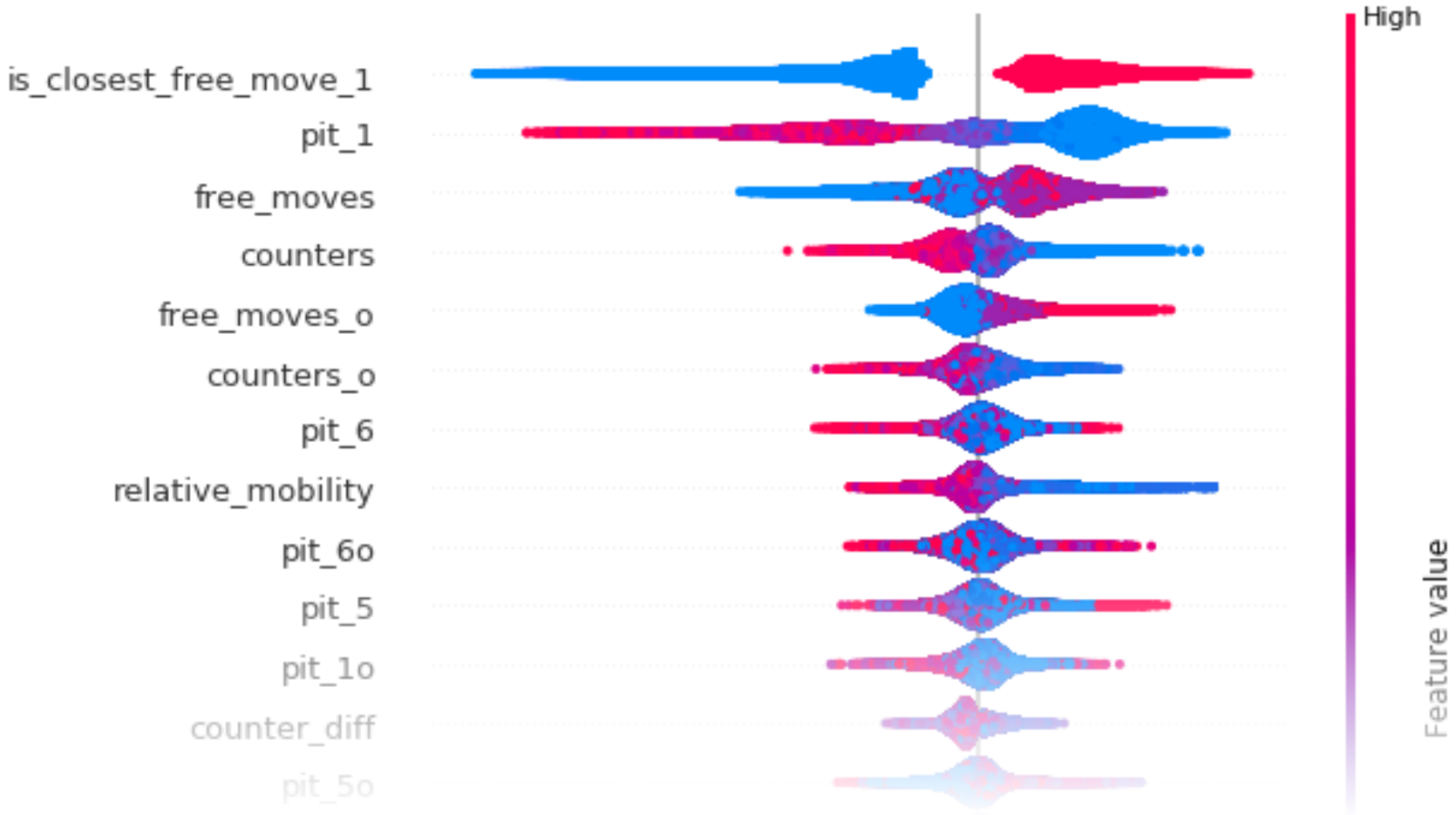
Pit 1 Optimality SHAP Interpretation

- Interpretations:
  - Free move nearly perfectly correlated
  - With some exceptions, having more pieces in pit 1 makes it less likely to be an optimal move
  - Free moves positively correlate, as taking a farther free move would add to pit 1, ruining the free move option.
  - More counters on either side negatively correlate with optimality
  - Opponent free moves positively correlates



Pit 1 Optimality SHAP Interpretation

CatBoostClassifier optimal_1 SHAP graph



Conclusions

- 254 fair starting states for Kalah (a.k.a. Mancala) moving 1-2 pieces from the standard 4-pieces-per-play-pit initial state.
- Analysis of unique fair optimal game tree states affirmed the node-ordering heuristic of Irving et al., underscoring a preference for closest free moves, closest captures, and closest plays to the scoring pit.
- Optimal play data from a diverse set of optimal and suboptimal game states yielded models with strength for predicting game values and optimal moves. Insights:
  - Decision trees ensembles perform well in modeling game value and optimality per pit
  - SHAP graphs indicate the importance of free moves, and also suggest there are overarching patterns where it is better to have more pieces in certain pits

Questions?