Auto Conjecturing and Konane

Ian Shehadeh IRSHEHADEH@SMCM.EDU
Division of Mathematics
St. Mary's College of Maryland
St. Mary's City, Maryland, USA

Abstract

Konane is an ancient Hawaiian game that plays similar to checkers. We present *Konjecture* a suite of tools for computional analysis of Konane. Using these tools combined with the program *Conjecturing*, we investigate the outcome class of stair-shaped patterns on Konane boards.

Keywords:

1 Kōnane

Kōnane is an ancient Hawaiian game played on a rectangular grid. The size of this grid varies. There are two players, one playing with black pieces, and the other white pieces.

1.1 Kōnane Starting Phase

A game begins with black removing one of their corner pieces or one of their center pieces (D. Thompson [1]). For example, in Figure 1 the game begins with black choosing one of the pieces outlined in red to remove. Once black removes a piece, white then chooses an adjacent piece to remove. For example, if black begins by removing $(3, 3)$, moving the position in Figure 2, white can now remove any of the highlighted pieces.

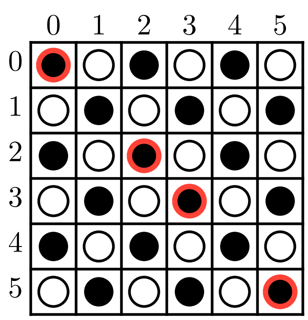


Figure 1: Starting Position

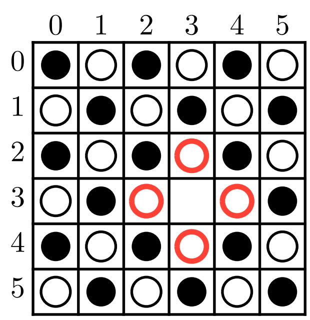


Figure 2: Turn 1

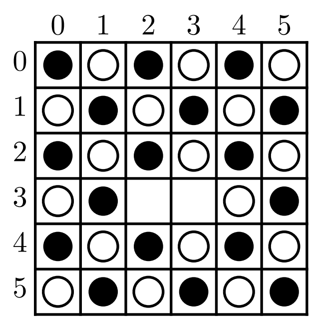


Figure 3: Turn 2

,

Once white removes a piece, normal play begins.

1.2 Kōnane Normal Play

On each player's turn, they must capture at least one of their opponent's pieces. The first player who is unable to capture any pieces loses.

A player may capture a piece of the opposing color is captured by *jumping* over it using one of their pieces into an empty space. Figure 4 shows white capturing a black piece by moving from $(0, 2)$ to $(0, 0)$.

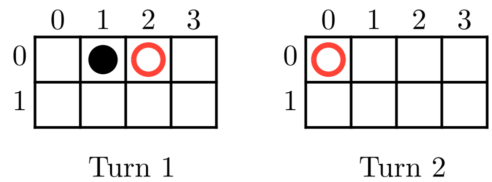


Figure 4: Capture

Multiple captures can be made in a single direction. For example, beginning from *Start* in Figure 5, if black plays first they can move to *Option 1*, *Option 2*, or *Option 3*.

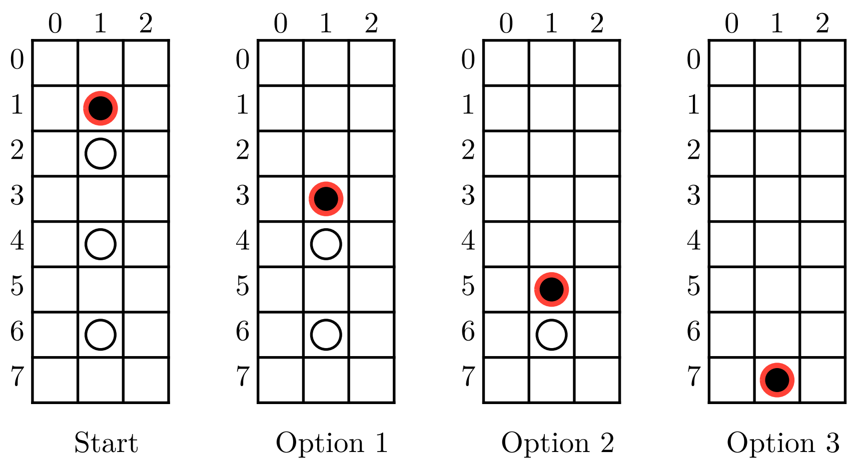


Figure 5: Capture

2 Conjecturing

Conjecturing (C. Larson and N. Van Cleemput [2]) is a projectthat generates a series of true iequalities based on a table of objects with associated *invariants*. In our case, an object is a single Kōnane position, and invariants include things like *number of moves*, *number of black pieces*, *number of white pieces*, etc.

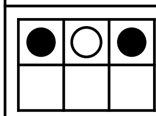
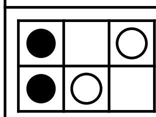
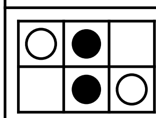
Games	**#Black** (I_1)	**#White** (I_2)	**#Moves** (I_3)
	2	1	0
	2	2	1
	2	2	2

Table 1: Conjecturing Inputs

We choose a single invariant to place on the left hand side of the inequality, and which operators to use. For example, given the data in Table 1, we could instruct *Conjecturing* to use I_2 on the left hand side of $\leq$, and create expressions using the operators $A+B, A*B, A+1, \sqrt{A}, \max(A,B)$. A few of the possible results are shown in Figure 6.

$$\begin{aligned} I_2 &\leq \max(I_1, I_2) \\ I_2 &\leq I_1 + I_2 \\ I_2 &\leq I_1 + 1 \\ I_2 &\leq I_1 * I_2 + 1 + 1 \end{aligned} \tag{1}$$

Figure 6: Conjecturing Output

3 Computation Analysis Kōnane

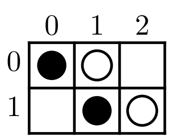
We were unable to find any open-source programs with a well-optimized implementation of Kōnane. But, computer-friendly grid game representations are well studied for games like Go, Chess and Checkers (J. Schaeffer *et al.* [3]).

3.1 Implementation

Definition 3.1.1: *Bit Field*: A bit field is a string of binary digits with a constant length. We write a bit fields right-to-left. Each bit has an associated index, beginning at 0.

Example: Given the bit field $B = 01101$, $B_0 = 1$, $B_1 = 0$, $B_2 = 1$, $B_3 = 1$, $B_4 = 0$,

We use two bit fields to represent a game of Kōnane. Given a board of width W and height H, we use bit fields with $W \times H$ elements. The first bit field (hereafter *BLACK*) has a 1 at index $y * W + x$ if and only if the game has a black piece in cell (x, y) (the top left cell is $(0,0)$). Similarly, the bit field representing white (hereafter *WHITE*) has a 1 at $y * W + x$ if and only if the game has a white piece in the cell (x, y).



is represented as

$$\begin{aligned} \text{BLACK} &= 100010 \\ \text{WHITE} &= 010001 \end{aligned} \tag{2}$$

Figure 7: Bit Field Representation

This format is compact, and it allows us to use only a few CPU instructions to calculate each player's moves.

3.1.1 Testing

To ensure accurate results, this implementation must be carefully tested.

The underlying bit-field representation is mostly tested using a method called property-based testing. These types of tests are concerned with a specific *property*, some check that theoretically always holds given a certain class of inputs, then we generate random input values and check the property to verify that it holds in practice.

The game's implementation has several specific tests, which come from simple cases (do we get the correct set of moves for a 2×2 game, etc.), and property tests based on established facts about specific Kōnane patterns. We've also begun to write tests based on well-known Kōnane positions, primarily from *Playing Kōnane Mathematically* M. Ernst [4].

The first of these tests builds a series of alternating black and white pieces in a line, then puts it in normal form and compares the result with the expected value, which is outlined in *Playing Kōnane Mathematically* M. Ernst [4]. Note although we only test these cases, the implementation is general enough to handle arbitrary games of Kōnane.

The test suite has expanded to the point we are reasonably confident about the program's accuracy. But, we'll continue to add unique test cases as they arise.

3.2 Invariants

Most invariants are picked arbitrarily, they're numbers that "seems interesting". But facts about particular positions from prior research inform the ideas. For example, in *Kōnane has Infinite Nim Dimensions* C. Santos and J.-N. Silva [5], construction of new positions relies on a "focal point", a piece that can be captured in a few ways, and opens both players up to new moves. It suggests that the number of pieces that can be captured, and the number of moves for a given player may be interesting attributes.

In the second week, we began implementing invariants. The underlined items have been implemented.

The current list of invariants is:

- Average distance from each piece to the nearest border.
- Total possible moves.
- Number of unique pieces that can be captured.
- Counts of each tile state.
- Distance between the highest and lowest piece.
- Distance between the furthest left and furthest right piece.
- The nim-value of a game
- The number-value of a game

All of these can (except for the game values), can be calculated for a single player, or both.

4 Known Facts

This section is a list of known facts about the game that informed our analysis.

4.1 Space Complexity

Kōnane is PSPACE-Complete R. A. Hearn [6], meaning it takes a polynomial amount of space relative to its input size, and any other problem in PSPACE can be solved in polynomial time if Kōnane can be solved in polynomial time.

4.2 Existence of Arbitrary Nim Dimension

If there is a $*n$ position on an $l_n \times c_n$ board, then there is a $*m$ position on an $l_m \times c_m$ board such that:

$$\begin{aligned} l_m &= l_n + c_n + 2 \\ c_m &= l_n + 3 \end{aligned} \tag{3}$$

The base case is $l_2 = 7$, $c_2 = 6$. C. Santos and J.-N. Silva [5]

4.3 Solid Linear Pattern

A solid linear pattern is series of N alternating white and black pieces, with jumps allowed on either side.

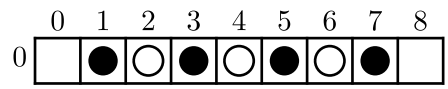


Figure 8: $N = 7$

The normal form for any solid linear pattern is:

$$\begin{array}{rl} -j & \text{if } N = 2j+1 \\ 0 & \text{if } N = 4j \\ * & \text{if } N = 4j+2 \end{array} \tag{4}$$

Source: Ernst, *Playing Kōnane Mathematically* M. Ernst [4]

4.4 Solid Linear Pattern With Tail

We say a solid linear pattern has a tail of length M, if there are M alternating pieces below the left-most piece in the solid linear pattern.

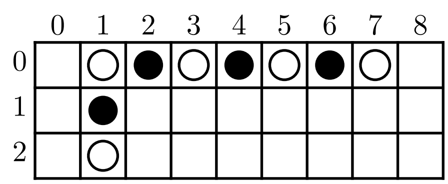


Figure 9: Linear with Tail $N = 7, M = 2$

4.4.1 Tail Length 1

Given a solid linear pattern of length N with a tail of length 1, the normal form is

$$\begin{array}{rll} * & \text{if } N \in \{2,3\} & \\ \{* \mid \downarrow\} & \text{if } N = 4 & \\ \{* \mid -2j+2\} & \text{if } N = 4j, & \text{where } j > 1 \\ \{2j-1 \mid *\} & \text{if } N = 4j+1, & \text{where } j > 0 \\ \{* \mid -2j+1\} & \text{if } N = 4j+2, & \text{where } j > 0 \\ \{2j \mid 0\} & \text{if } N = 4j+3, & \text{where } j > 0 \end{array} \tag{5}$$

Source: Ernst, *Playing Kōnane Mathematically* M. Ernst [4]

4.5 Solid Linear Pattern with Offset Tail

A solid linear pattern of length N with *offset* tail of length M is identical to a solid linear pattern with tail, but with the top left piece removed.

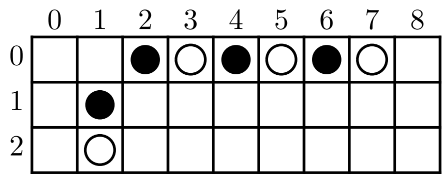


Figure 10: Linear with Offset Tail $N = 7, M = 2$

$$\begin{array}{rl} \downarrow & \text{if } N = 3 \\ j-1 & \text{if } N = 2j, \qquad \text{where } j > 0 \\ * & \text{if } N = 4j+1, \ \text{where } j > 0 \\ 0 & \text{if } N = 4j+3, \ \text{where } j > 0 \end{array} \tag{6}$$

Source: Ernst, *Playing Kōnane Mathematically* M. Ernst [4]

4.6 Somewhat Solid Linear Patterns

Let $S(a, b, c, ...)$ be a series of solid linear patterns beginning with a white tile, each separated by a space.

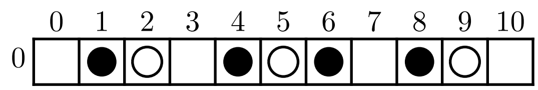


Figure 11: $S(2, 3, 2)$

Quick Facts [7]:

- $S(2a + 1, 2b + 1, 2c + 1) = a + b + c$
- $$S(2a+1, 2b+1, 2c) = S(2a+1, 2(b+1)) + S(2k-2) = \begin{cases} \star\,(i - (b+1) + (c-1)) & \text{if } a > b+1 \\ \star\,\left(2^{a+(b+1)-1} + (k-1)\right) & \text{if } a \leq b+1 \end{cases} \tag{7}$$

4.7 Penetration

The penetration of a particular Kōnane configuration is how far its members are able to move. Rectangular configurations have a maximum penetration of 4 M. Ernst [4].

4.8 Patterns in Real Play

This is a living list of patterns observed in real play, and general facts about the game:

- When the game was first documented by European settlers the most common board size was 14×17, although anything from 8×8, to 13×20 was also used. Modern games are often played on an 18×18 board [7].
- “In real Kōnane games, play frequently proceeds to the corners after center of the board has been largely cleared” M. Ernst [4]

5 Results

We’ve investigated several different Kōnane patterns, the most interesting so far are those with a stack of 2×1 alternating pieces, with each layer shifted by one.

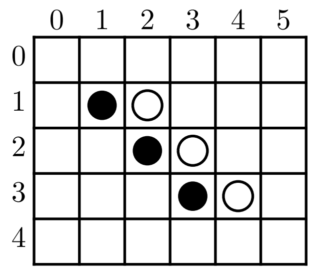


Figure 12: Stairs n=3

So far, computational analysis has show the canonical form for n between 1 and 10. Past 10, the analysis slows down enough that it quickly becomes difficult to compute.

Height	Canonical Form
1	$*$
2	$*2$
3	$*3$
4	$*$
5	0
6	$*$
7	0
8	$*$
9	0
10	$*3$

Table 2: Stair Canonical Form

5.1 Maximum Penetration of Stair Pattern

It is likely that no matter the height of this pattern, the furthest a piece can ever move vertically or horizontally outside of the bounding box of the staircase (top left to bottom left piece) is 1. This is true in all cases tested, futhermore, conisder the game in Figure 13, this is the result of white jumping left on the bottom stair. It's trivial to show that each piece can only move outside the bounding box by 1. The same is true for Figure 14, even if the pattern continues up to the left. Furthermore, in this configuration the bottom of the staircase is now inaccessible, given the top part has a penetration of 1. I suspect there's a (relatively easy) inductive proof of this.

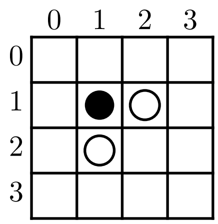


Figure 13: Broken Stair

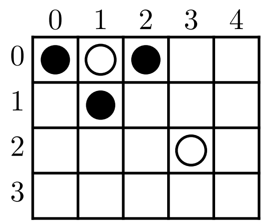


Figure 14: Broken Stair

6 Remaining Facts

Below are incomplete sections I would like to add for the final draft.

6.1 Analysis for Stair Case Outcome Class

Even if no general formula is found based on the height of the stair case, I'd like to dig deeper into the pattern.

6.2 Summary of Conjecturing Results

Summary of how conjecturing was used, and the kind of results it gave, even if they weren't useful

6.3 (Maybe) $*2$ 4x3 Games

I'm like 65% sure that $2*$ configurations within a 4×3 rectangle all more-or-less just the game stairs with $n = 2$. It seems like even in games that don't look like it all other pieces are superfluous, even if there are other moves available. Basically, its always that stair configuration that makes the value $*2$, and even if there are other moves those are strictly worse for both players.

This proof seems hard though, and I'm not actually sure its true.

References

[1] D. Thompson, "Teaching a Neural Network to Play Konane," 2005.

[2] C. Larson and N. Van Cleemput, "Automated Conjecturing I: Fajtlowicz's Dalmatian Heuristic Revisited," *Artificial Intelligence*, vol. 231, pp. 17–38, Feb. 2016, doi: 10.1016/j.artint.2015.10.002.

[3] J. Schaeffer *et al.*, "Checkers Is Solved," *Science*, vol. 317, pp. 1518–1522, Oct. 2007, doi: 10.1126/science.1144079.

[4] M. Ernst, "Playing Konane Mathematically: A Combinatorial Game-Theoretic Analysis," *UMAP Journal*, vol. 16, pp. 95–121, 1995.

[5] C. Santos and J.-N. Silva, "Konane Has Infinite Nim-Dimension," *Integers [electronic only]*, vol. 1, Jan. 2008.

[6] R. A. Hearn, "Amazons, Konane, and Cross Purposes Are PSPACE-complete," *Games of No Chance 3*. in Mathematical Sciences Research Institute Publications. Cambridge University Press, Cambridge, pp. 287–306, 2009. doi: 10.1017/CBO9780511807251.015.

[7] "1xn Konane: A Summary of Results," *Games of No Chance [2] More Games of No Chance*, no. 2. in Games of No Chance. Cambridge Univ. Press, Cambridge [u.a.], pp. 331–339, 2010.