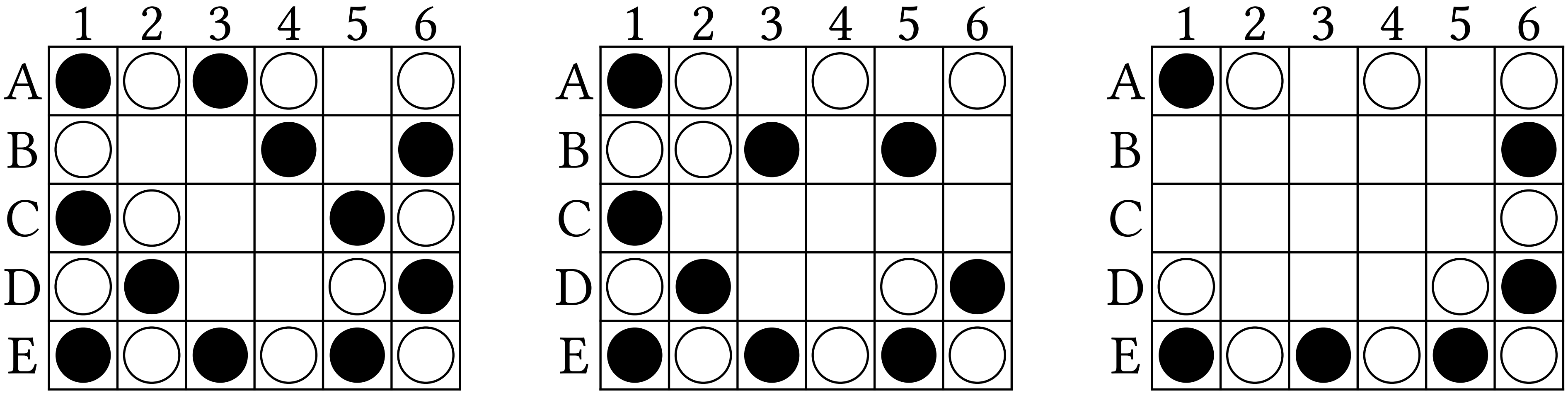
Computation and Kōnane

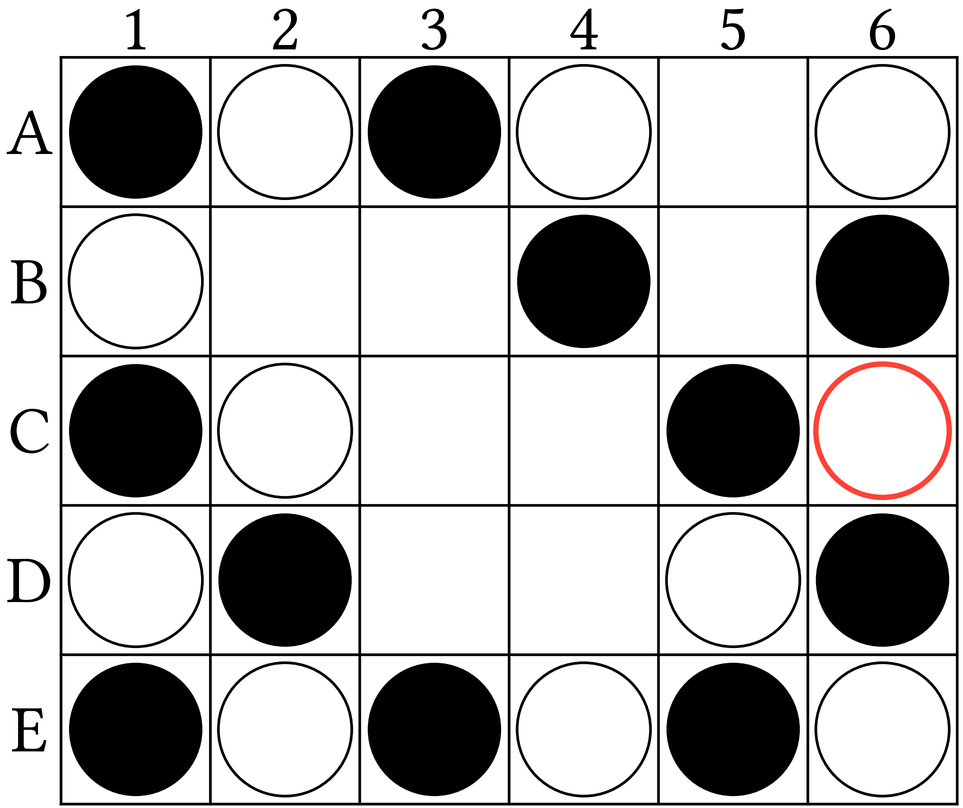
The Game

- 2 Players (Black and White)
- Rectangular grid (size varies)
- Move: Capture by jumping ↑, ↓, ←, →

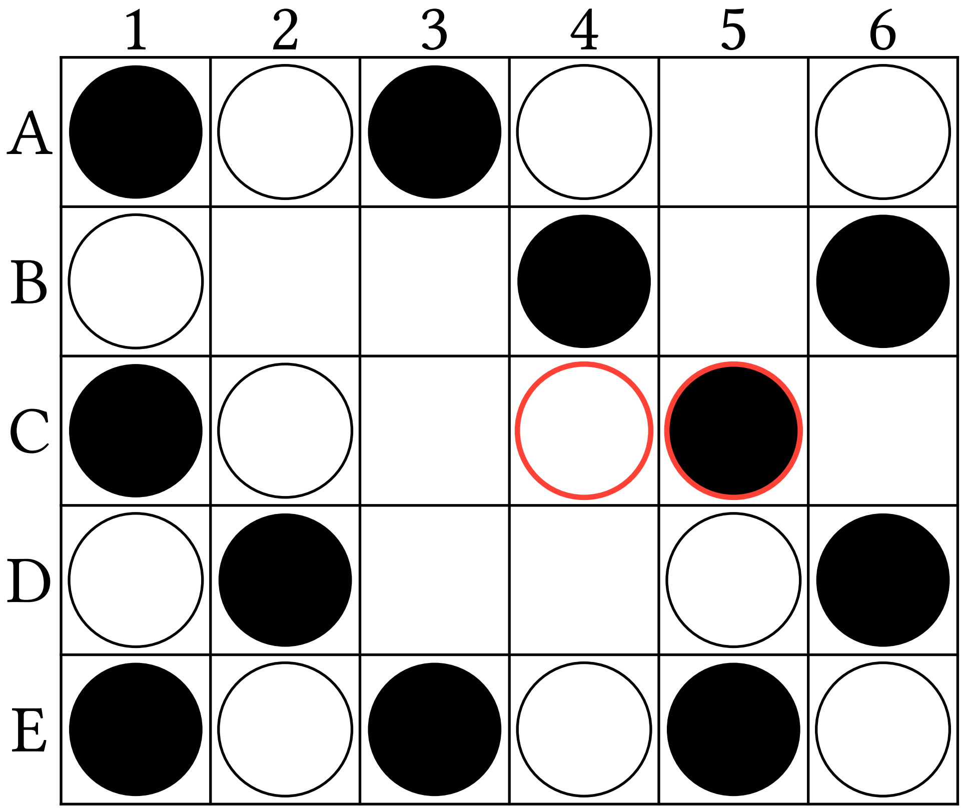


Example Games

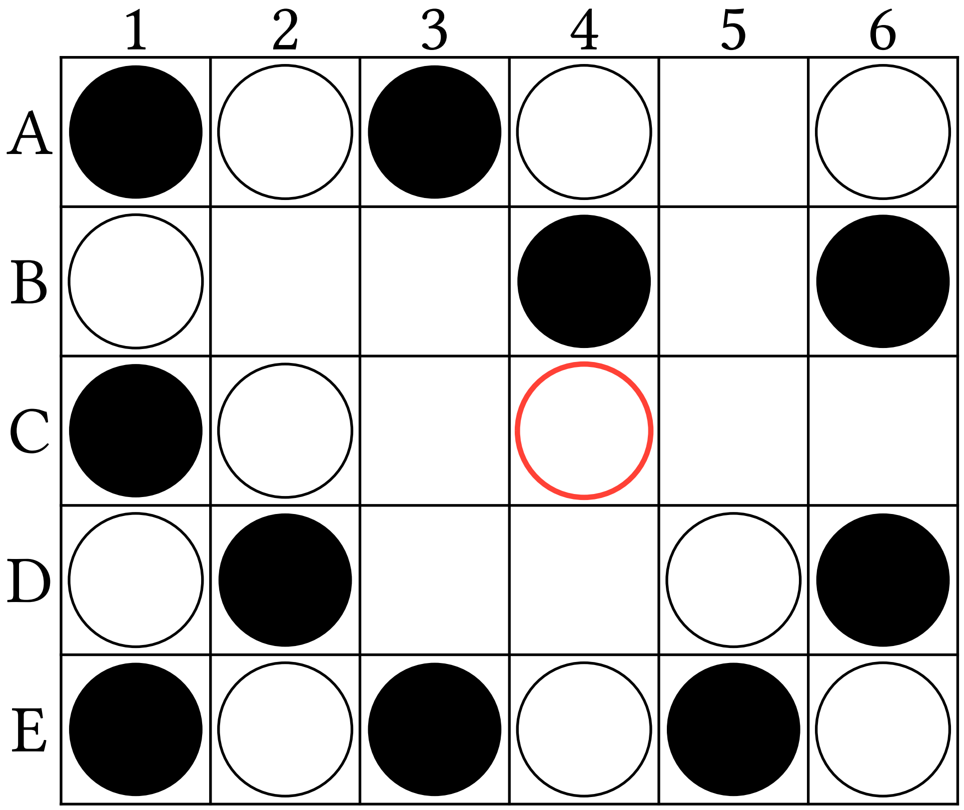
Example: Move



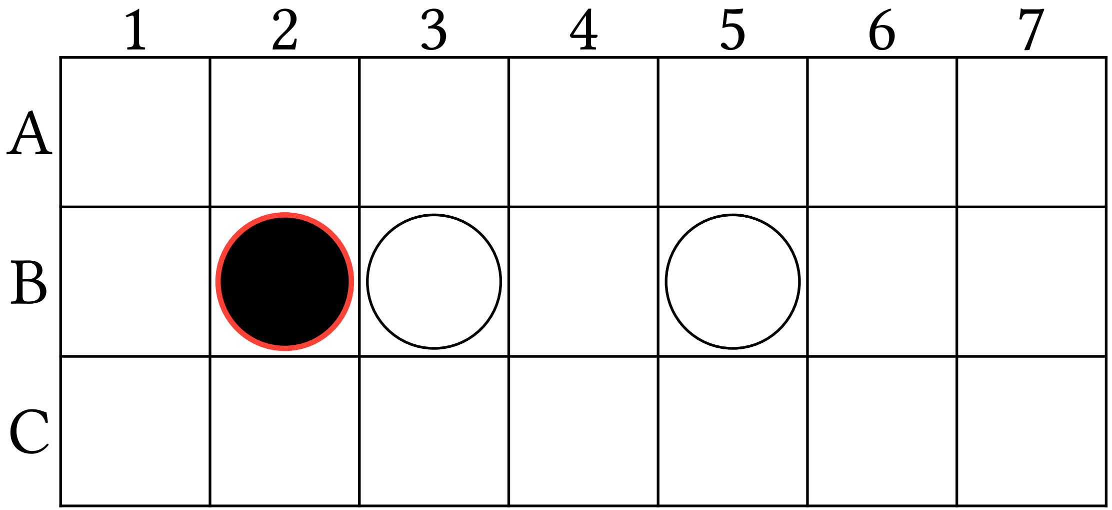
Example: Move



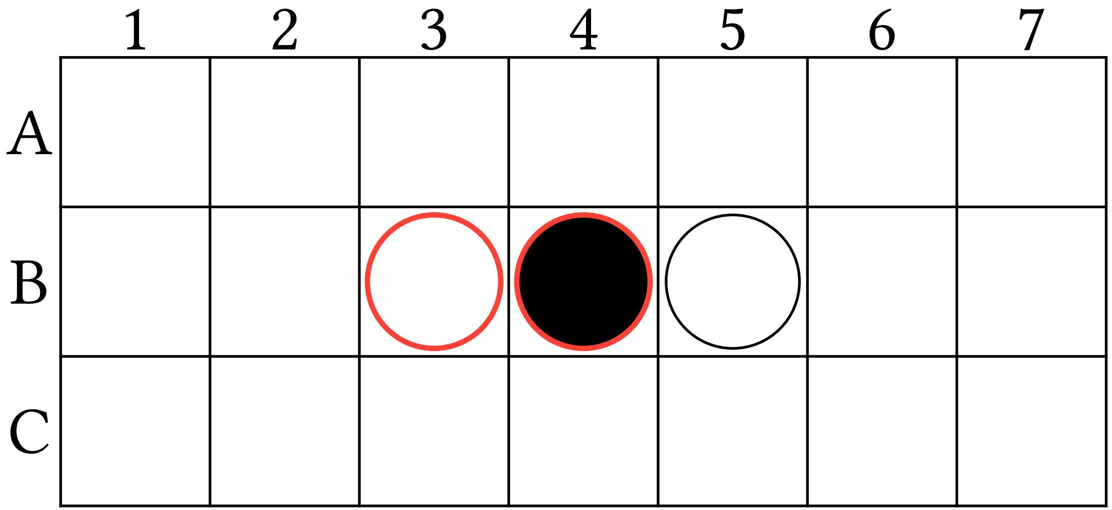
Example: Move



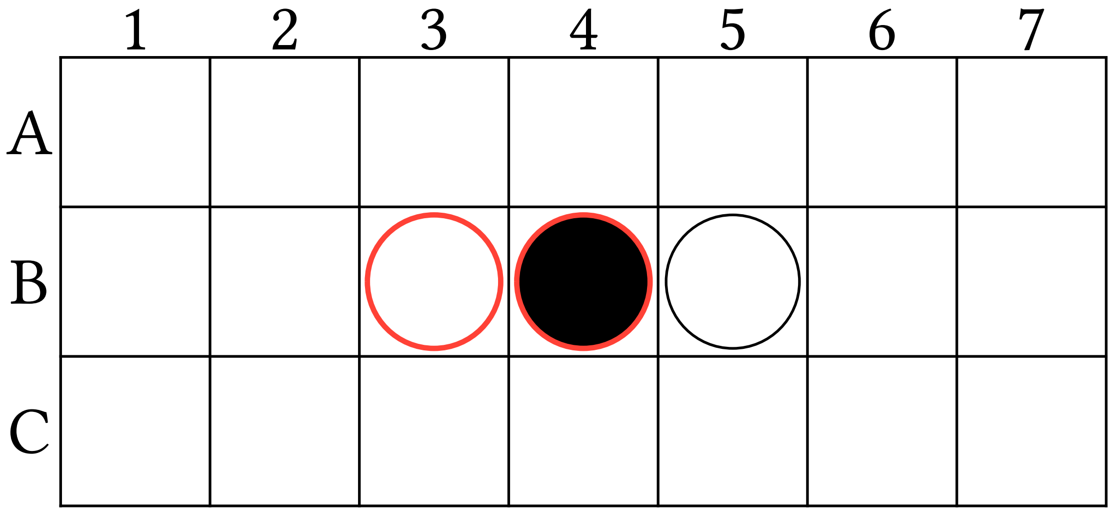
Example: Multiple Jumps



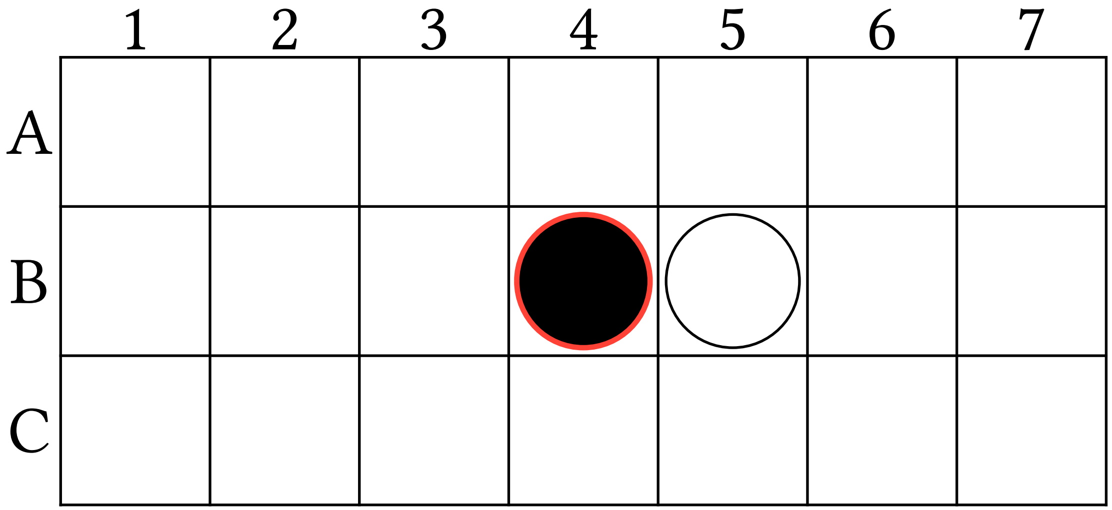
Example: Multiple Jumps



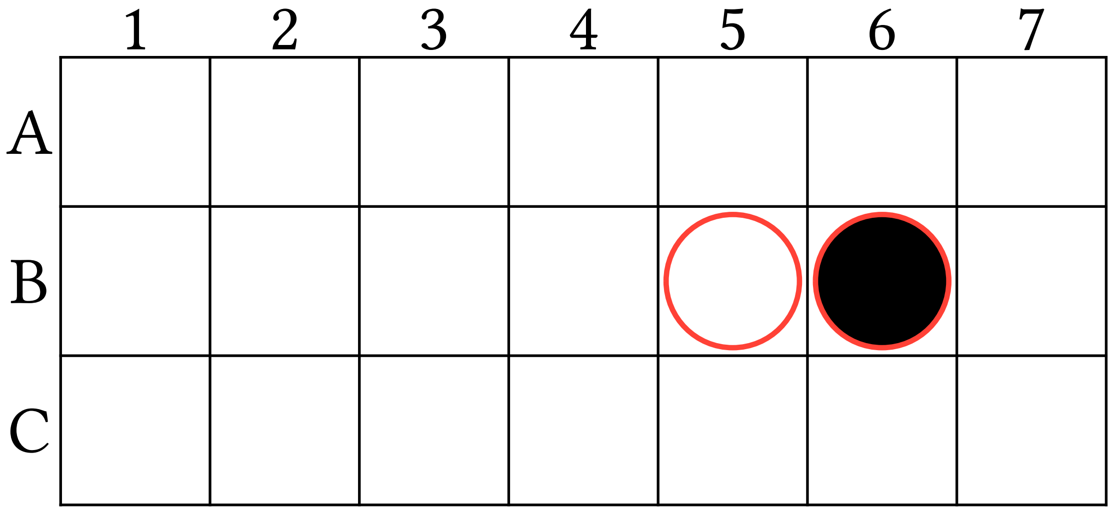
Example: Multiple Jumps



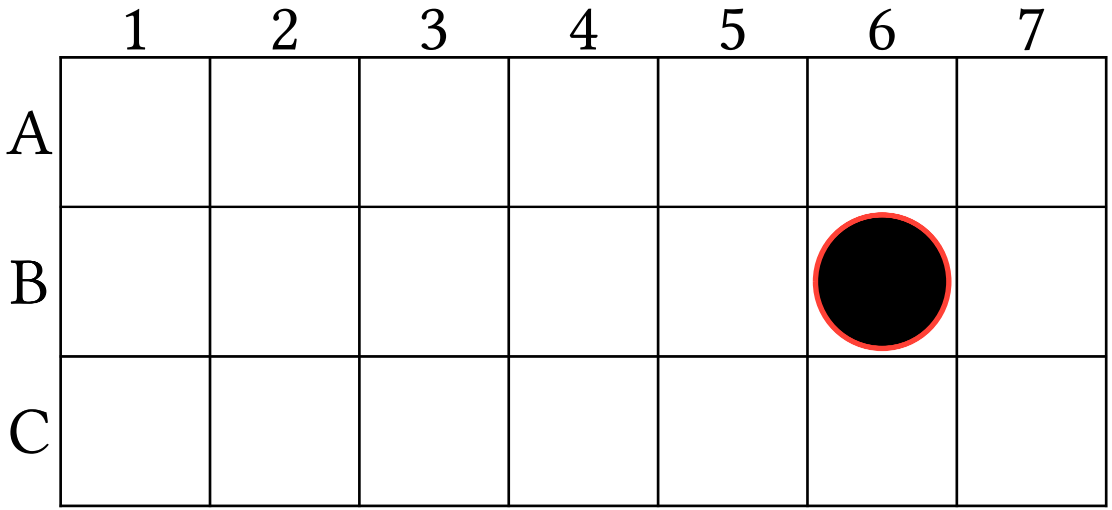
Example: Multiple Jumps



Example: Multiple Jumps

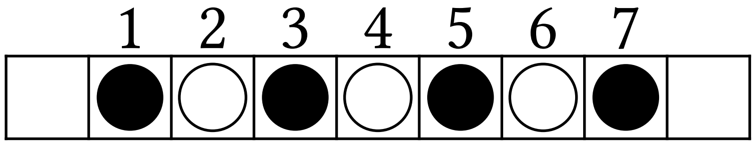


Example: Multiple Jumps

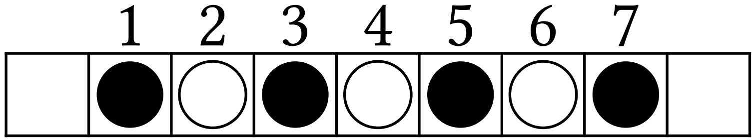


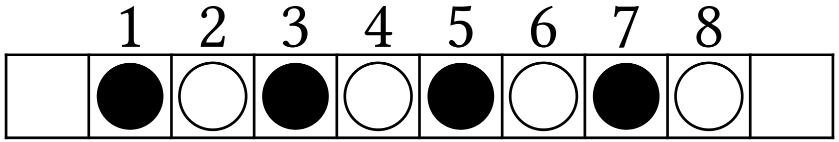
Solid Linear Patterns

Who Wins?

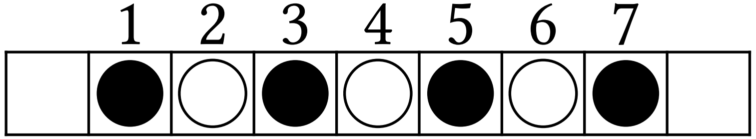


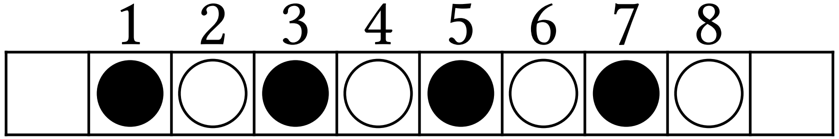
Who Wins?





Who Wins?

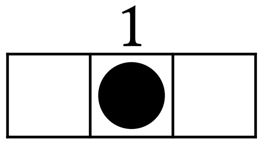


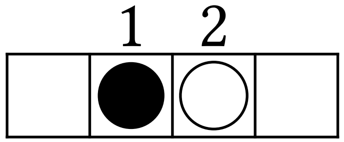


- Only *white* can move if there's an *odd* number of stones
- *Both* can move if there's an *even* number of stones

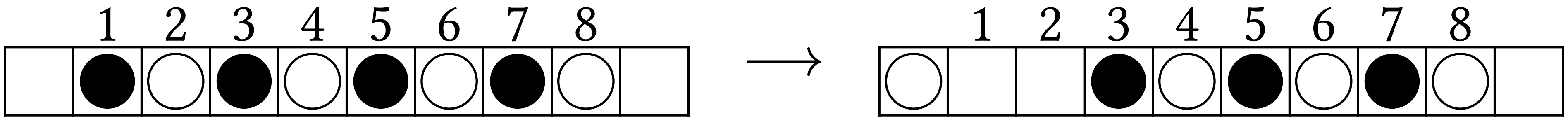
Base Case

Now we need to know the outcome of $\mathrm{SLP}(2)$ and $\mathrm{SLP}(1)$



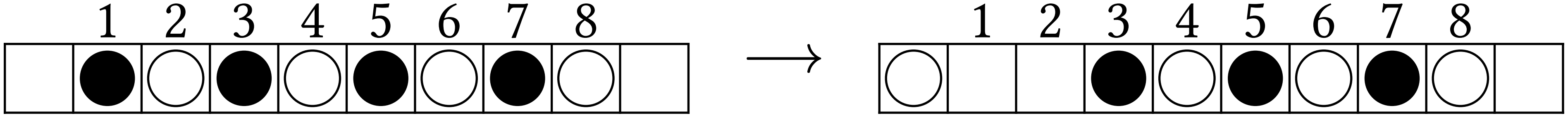


Inductive Case

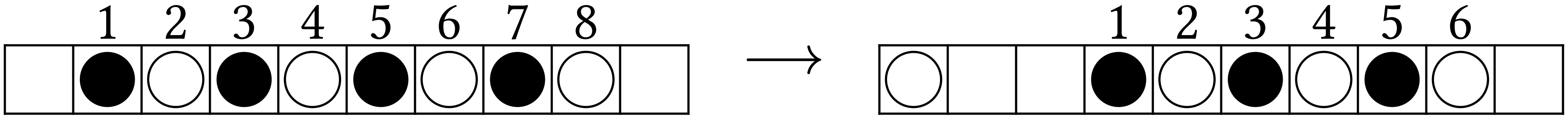


- After a jump to the left or right the stone moves out of reach

Inductive Case



- After a jump to the left or right the stone moves out of reach



- So, we can think of this new position as $\mathrm{SLP}(6)$
- In fact, any move by either player moves to $\mathrm{SLP}(N-2)$

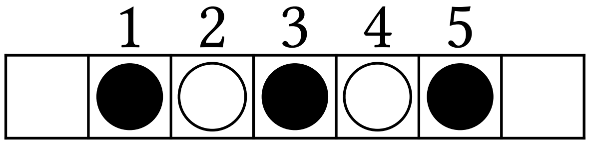
Putting it Together

- If n is *even* then $\mathrm{SLP}(n) = \{\mathrm{SLP}(n-2) \mid \mathrm{SLP}(n-2)\}$
- If n is *odd* then $\mathrm{SLP}(n) = \{\mid \mathrm{SLP}(n-2)\}$
- $\mathrm{SLP}(2) = *$
- $\mathrm{SLP}(0) = \mathrm{SLP}(1) = 0$

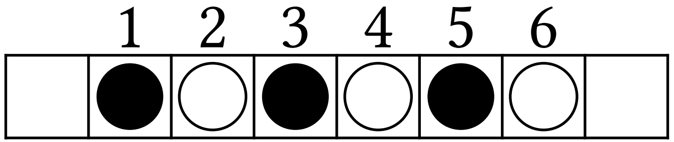
Putting it Together

- If n is *even* then $\mathrm{SLP}(n) = \{\mathrm{SLP}(n-2) \mid \mathrm{SLP}(n-2)\}$
- If n is *odd* then $\mathrm{SLP}(n) = \{\mid \mathrm{SLP}(n-2)\}$
- $\mathrm{SLP}(2) = *$
- $\mathrm{SLP}(0) = \mathrm{SLP}(1) = 0$

If n is **odd** then $n-2$ is still odd



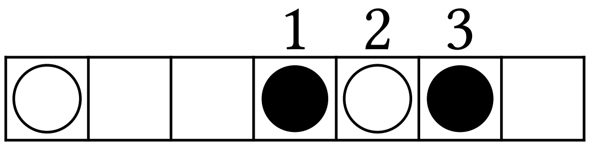
If n is **even** then $n-2$ is still even



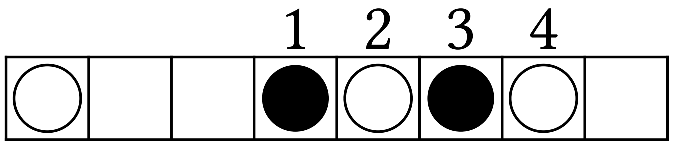
Putting it Together

- If n is *even* then $\mathrm{SLP}(n) = \{\mathrm{SLP}(n-2) \mid \mathrm{SLP}(n-2)\}$
- If n is *odd* then $\mathrm{SLP}(n) = \{\mid \mathrm{SLP}(n-2)\}$
- $\mathrm{SLP}(2) = *$
- $\mathrm{SLP}(0) = \mathrm{SLP}(1) = 0$

If n is **odd** then $n-2$ is still odd



If n is **even** then $n-2$ is still even

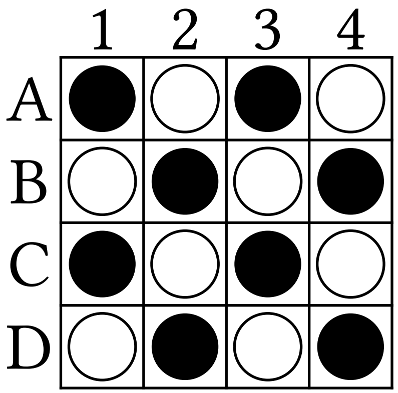


Solid Linear Pattern

- If n is odd, white always wins.
- If n is even, then the game has $n/2$ moves
  - Each player has the same option on every turn

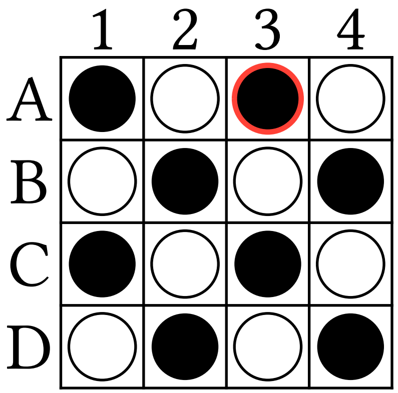
Computation

Representing Kōnane on Computers



	A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	C4	D1	D2	D3	D4
White	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
Black	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1

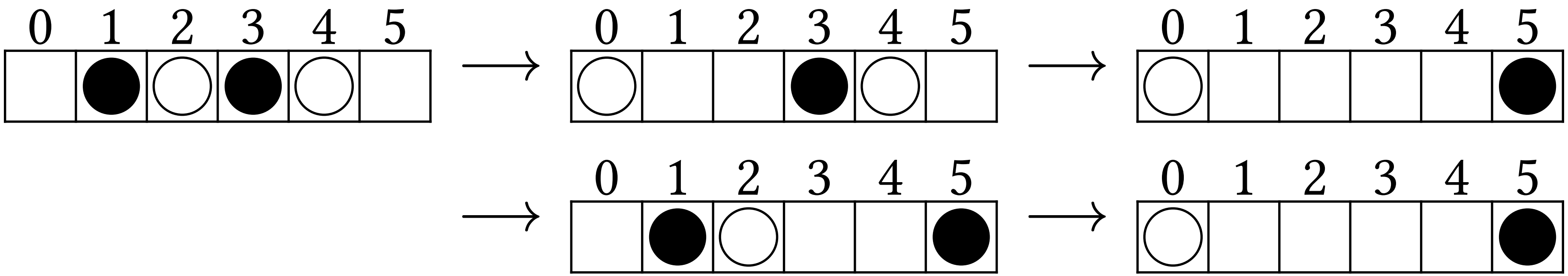
Representing Kōnane on Computers



	A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	C4	D1	D2	D3	D4
White	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
Black	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1

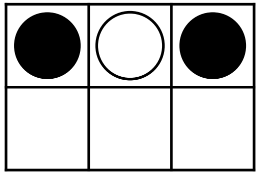
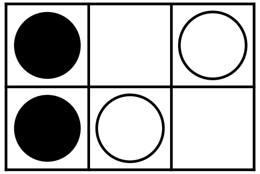
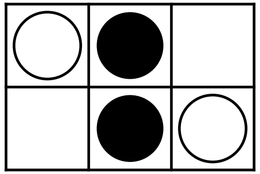
Solving Games

1. Recursively generate moves

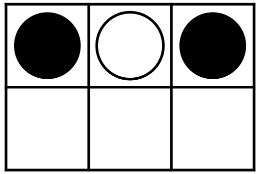
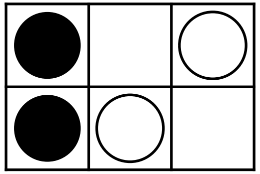
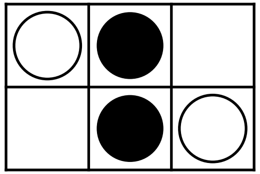


2. Determine winners of the leaf nodes
3. Determine winners of ancestor nodes

Conjecturing

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

Conjecturing

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

$$I_2 \leq \max(I_1, I_2)$$

$$I_2 \leq I_1 + I_2$$

$$I_2 \leq I_1 + 1$$

$$I_2 \leq I_1 * I_2 + 1 + 1$$

Looking at Results

[1]:

	wCnt	bCnt	wBrdr	bBrdr	wW	bW	wCaptures	bCaptures	wMoves	bMoves	nimber	number	Image
1	1	1	2	2	1	1	1	1	1	1	1	NaN	
2	2	2	2.5	2.5	2	2	2	2	3	3	2	NaN	
3	3	3	3	3	3	3	3	3	5	5	3	NaN	
4	4	4	3.5	3.5	4	4	4	4	7	7	1	NaN	

[11]:

```
from conjecturing import dfConjecture
indexed_df = df.with_row_index()
invariants =["nimber", "wH", "bH", "wCnt", "bCnt", "wW", "bW", "wCaptures", "bCaptures", "wMoves", "bMoves"]
dfConjecture(
    indexed_df,
    invariants,
    0,
    time = 10,
    theory=["bMoves", "wCnt"],
    operators=["+", "-", "*", "-()", "/", "*2", "+1", "-1"]
)
```

[11]:

```
[nimber(x) <= (2*wMoves(x) - 1)/wH(x),
 nimber(x) <= 2*wH(x) - wMoves(x) + 2,
 nimber(x) <= wMoves(x)/((wH(x) - 1)/bH(x) + 1),
 nimber(x) <= (2*(wH(x) + 1)/wMoves(x) - 1)*bMoves(x)]
```

[13]:

```
df.select("wH", "wMoves", "bMoves", "Image")
```

[13]:

wH	wMoves	bMoves	Image
1	1	1	

Questions?