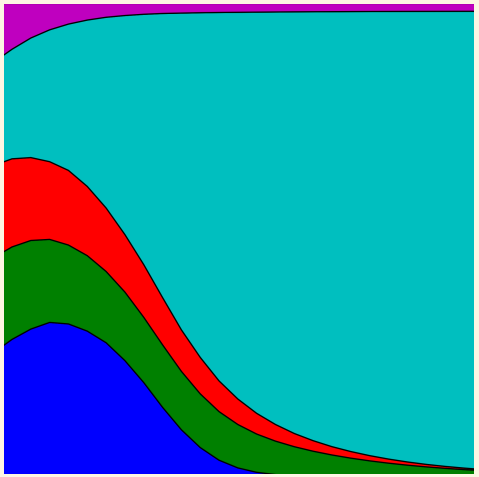
Vince: @drvinceknight

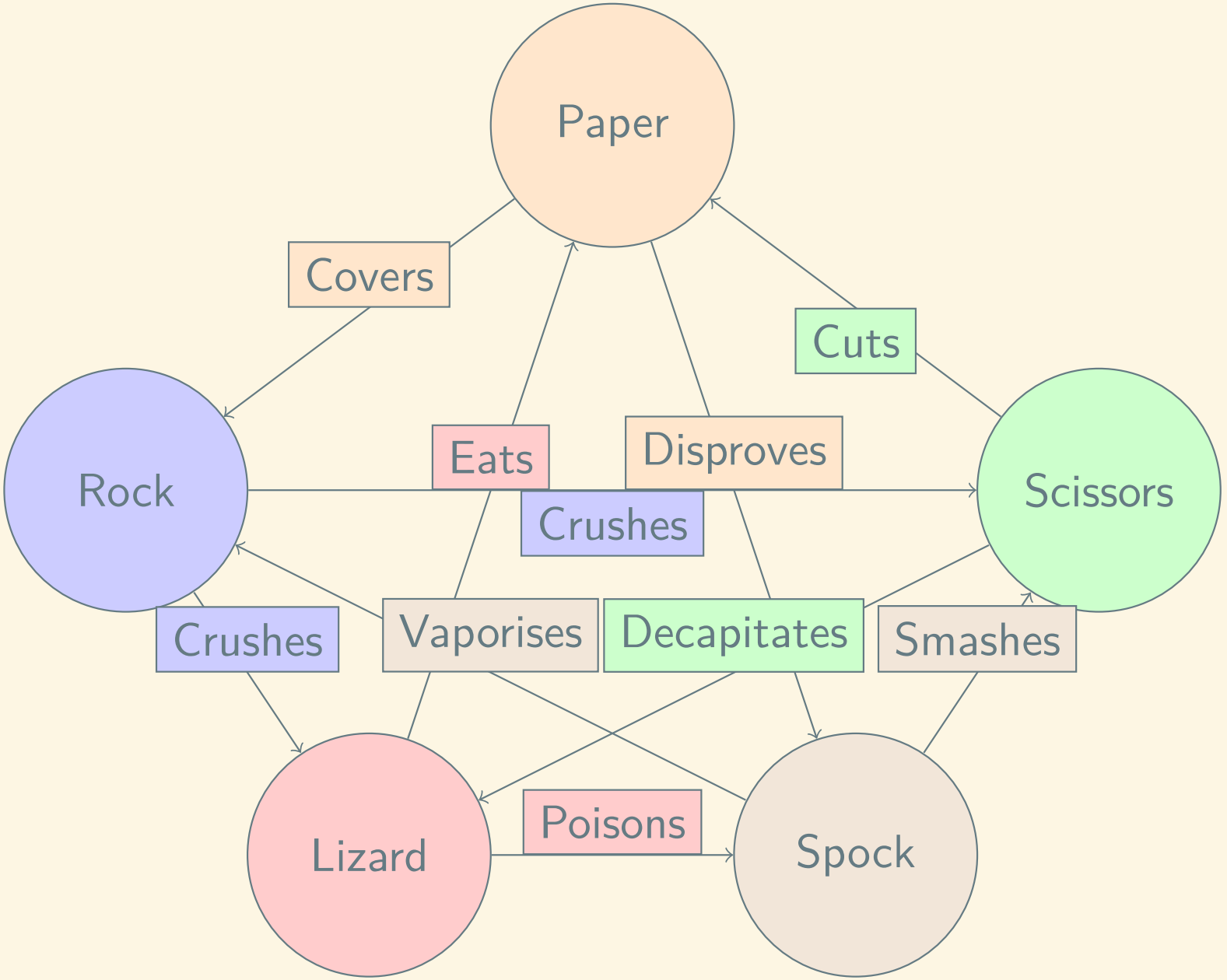
arxiv.org/abs/1707.06920

CARDIFF UNIVERSITY

PRIFYSGOL CAERDYDD

Software Sustainability Institute




Paper
Covers
Cuts
Rock
Eats
Disproves
Scissors
Crushes
Crushes
Vaporises
Decapitates
Smashes
Lizard
Poisons
Spock

```
pip install nashpy
```

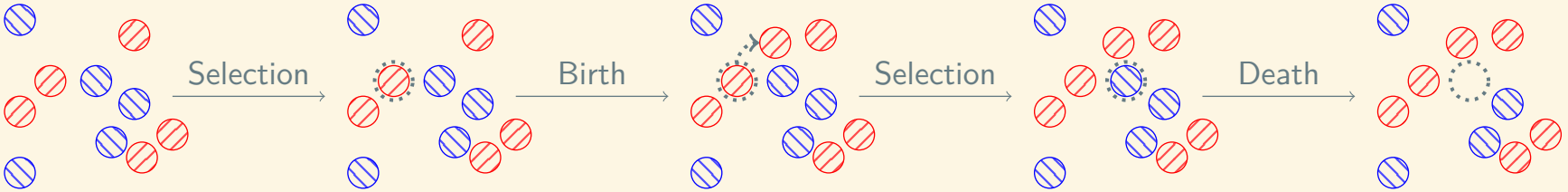
https://www.youtube.com/watch?v=p3Uos2fzIJ0

@kirstyjean (2 Jun 2017):

Me: sets up flawless heat competition trial, lizards will fight over hot podium, there can only be one winner!
Lizards:
#ALlizards2017

@kirstyjean (2 Jun 2017):

Me: sets up flawless heat competition trial, lizards will fight over hot podium, there can only be one winner!
Lizards:
#ALlizards2017


Selection
Birth
Selection
Death

```
def moran(N, game, i=1, seed=0):
    """
    Return the population counts for the Moran process on a game
    """
    population = [0 for _ in range(i)] + [1 for _ in range(N - i)]
    counts = [(i, N - i)]

    np.random.seed(seed)

    while len(set(population)) == 2:

        scores = []

        for i, player in enumerate(population):
            total = 0
            for j, opponent in enumerate(population):
                if i != j:
                    total += game[player, opponent]
            scores.append(total)

        total_score = sum(scores)
        probabilities = [score / total_score for score in scores]
        reproduce_index = np.random.choice(range(N), p=probabilities)

        eliminate_index = np.random.randint(N)
        population[eliminate_index] = population[reproduce_index]

        counts.append((population.count(0), population.count(1)))
    return counts
```

```
14        for i, player in enumerate(population):
15            total = 0
16            for j, opponent in enumerate(population):
17                if i != j:
18                    total += game[player, opponent]
19            scores.append(total)
20
21        total_score = sum(scores)
22        probabilities = [score / total_score for score in scores]
23        reproduce_index = np.random.choice(range(N), p=probabilities)
24
25        eliminate_index = np.random.randint(N)
26        population[eliminate_index] = population[reproduce_index]
```

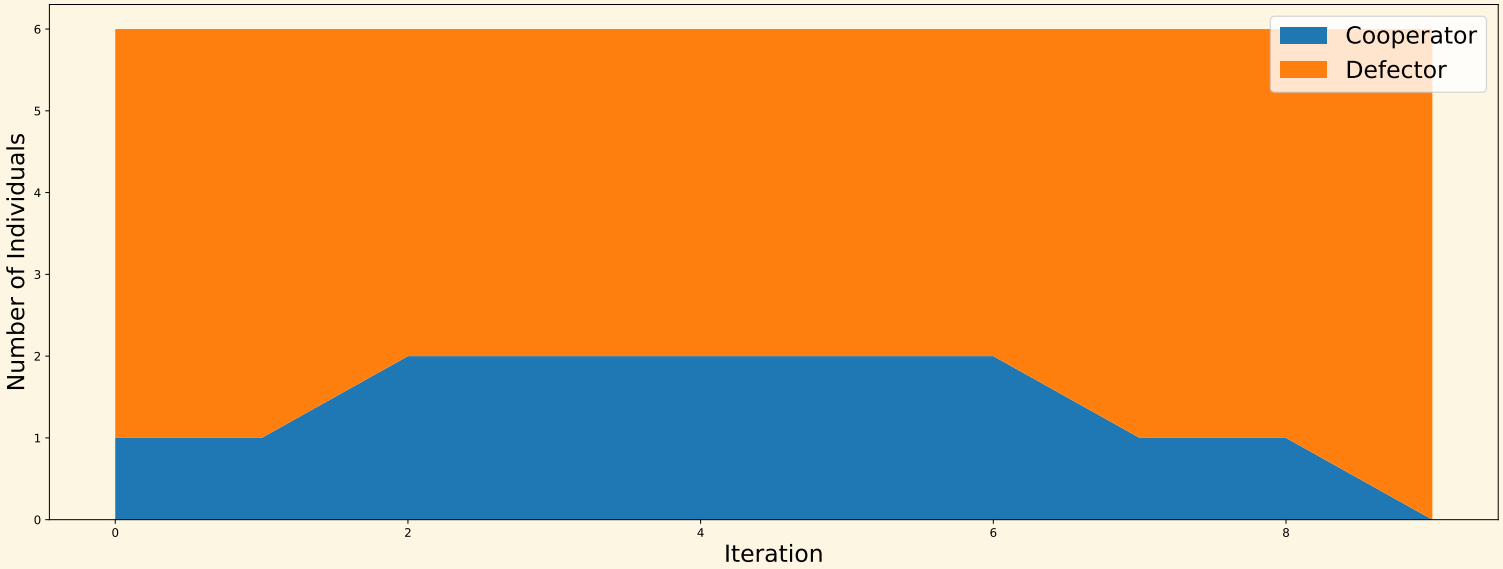
$$\begin{pmatrix} 3 & 0 \\ 5 & 1 \end{pmatrix} \qquad \begin{pmatrix} 3 & 5 \\ 0 & 1 \end{pmatrix}$$

Robert Axelrod

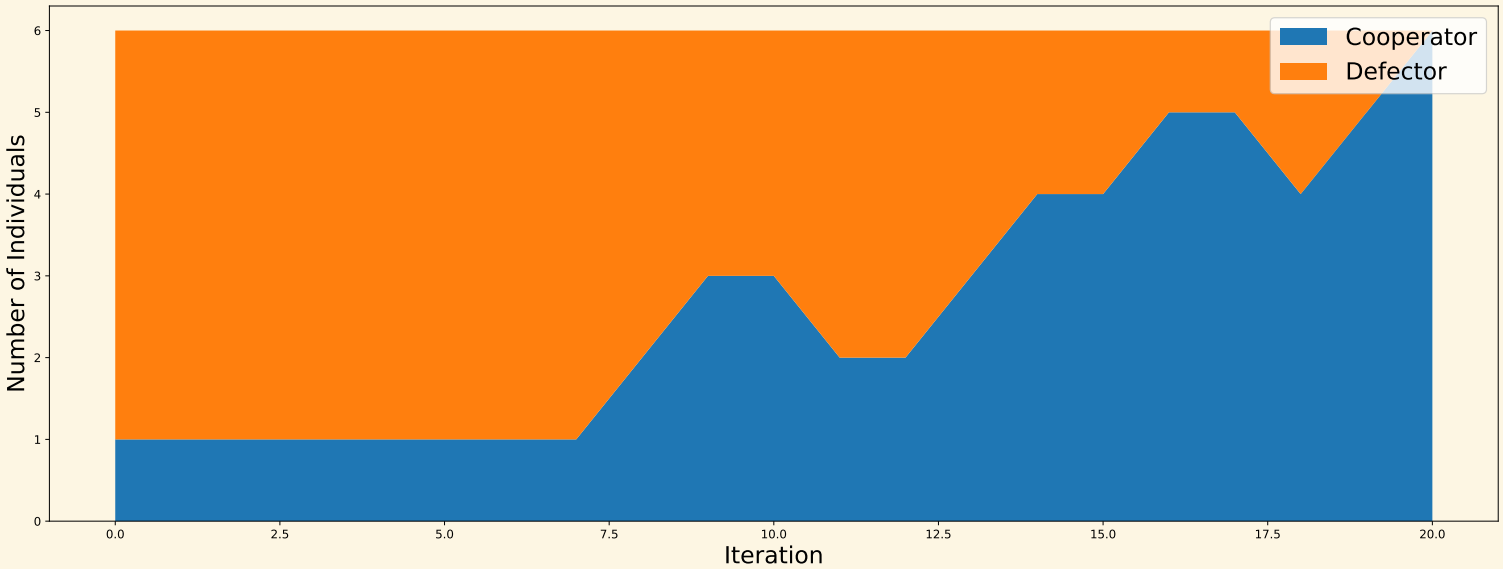
Robert Axelrod

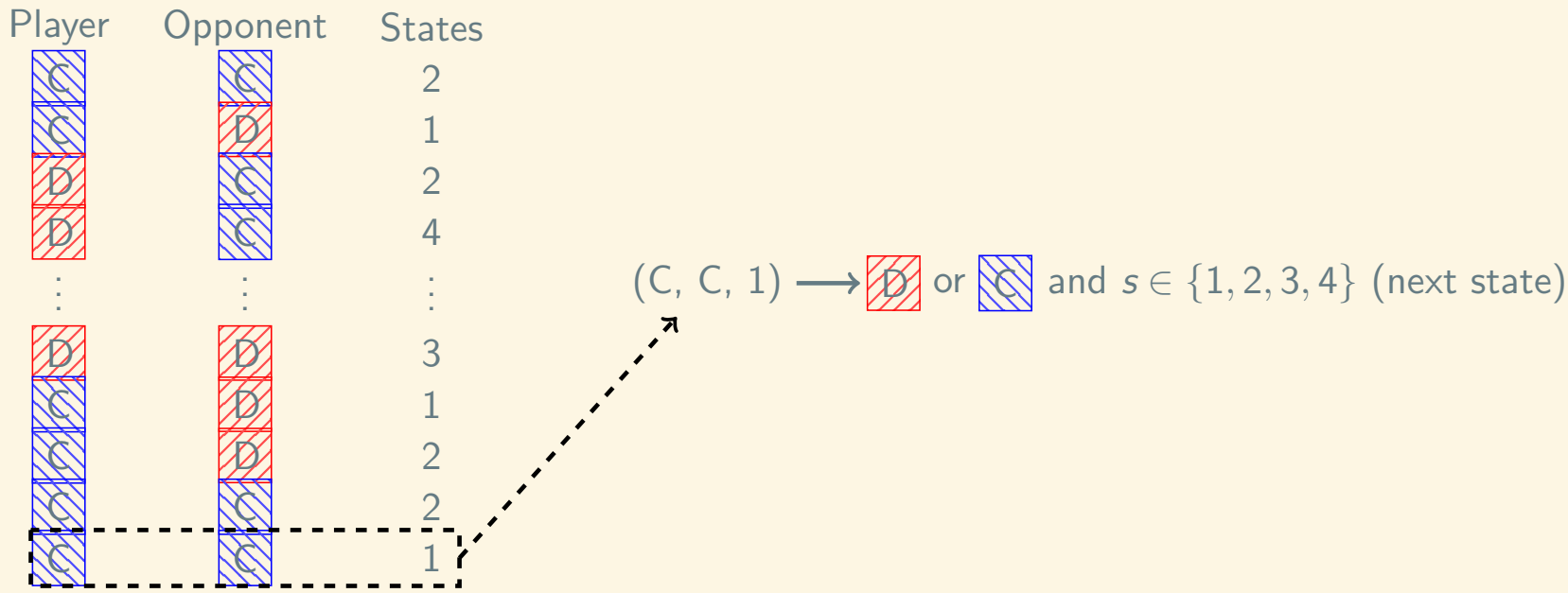
```
>>> import axelrod as axl

>>> players = (axl.TitForTat(),
...            axl.Cooperator())
>>> axl.Match(players, turns=5).play()
[(C, C), (C, C), (C, C), (C, C), (C, C)]

>>> players = (axl.TitForTat(),
...            axl.Defector())
>>> axl.Match(players, turns=5).play()
[(C, D), (D, D), (D, D), (D, D), (D, D)]

>>> players = (axl.TitForTat(),
...            axl.Alternator())
>>> axl.Match(players, turns=5).play()
[(C, C), (C, D), (D, C), (C, D), (D, C)]
```

# Resistance

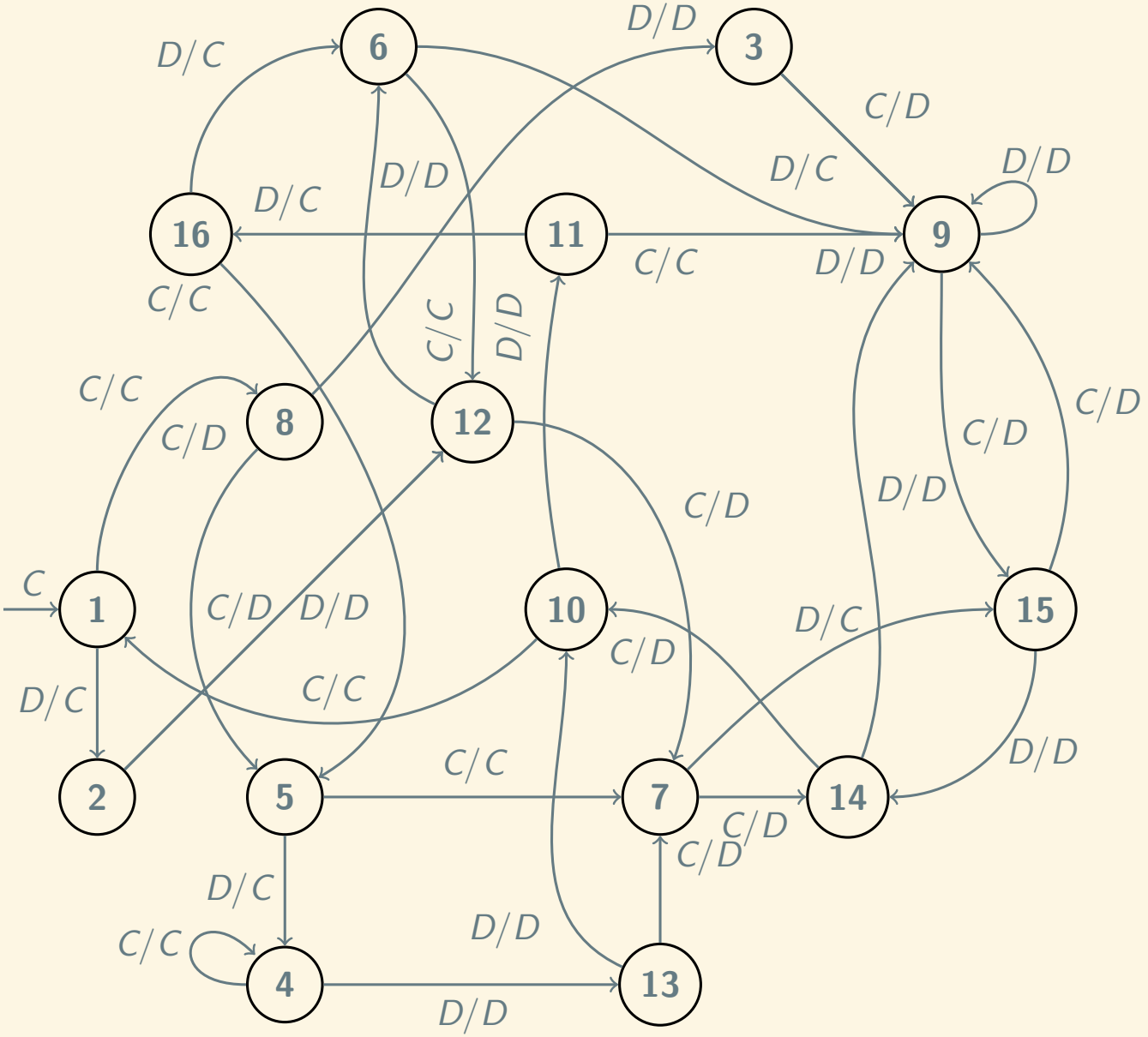



# Invasion




Player
Opponent
States
C
C
D
D
D
C
C
C
C
C
2
1
2
4
3
1
2
2
1
(C, C, 1) ⟶ D or C and $s \in \{1, 2, 3, 4\}$ (next state)

```
import axelrod_dojo
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
C
D/C
D/D
C/D
C/C

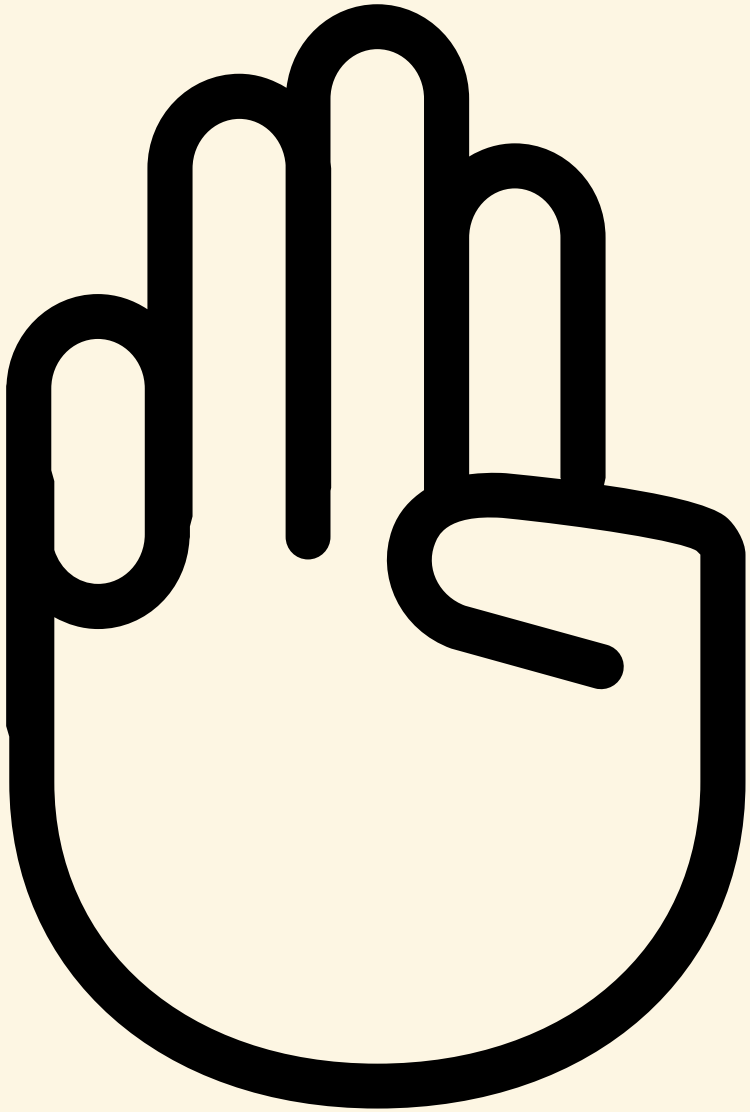


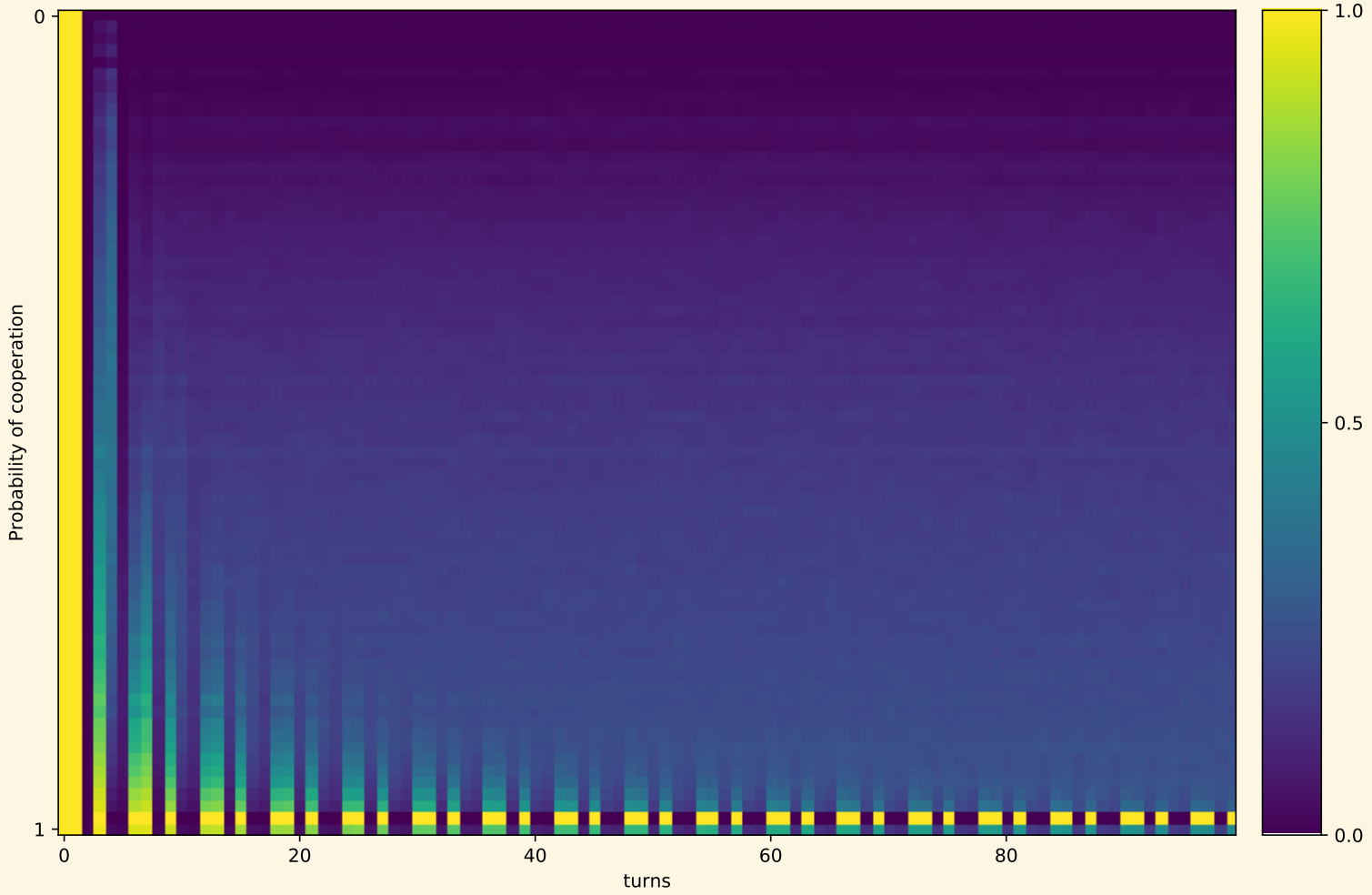
Image made by http://www.freepik.com from https://www.flaticon.com is licensed by CC BY 3.0

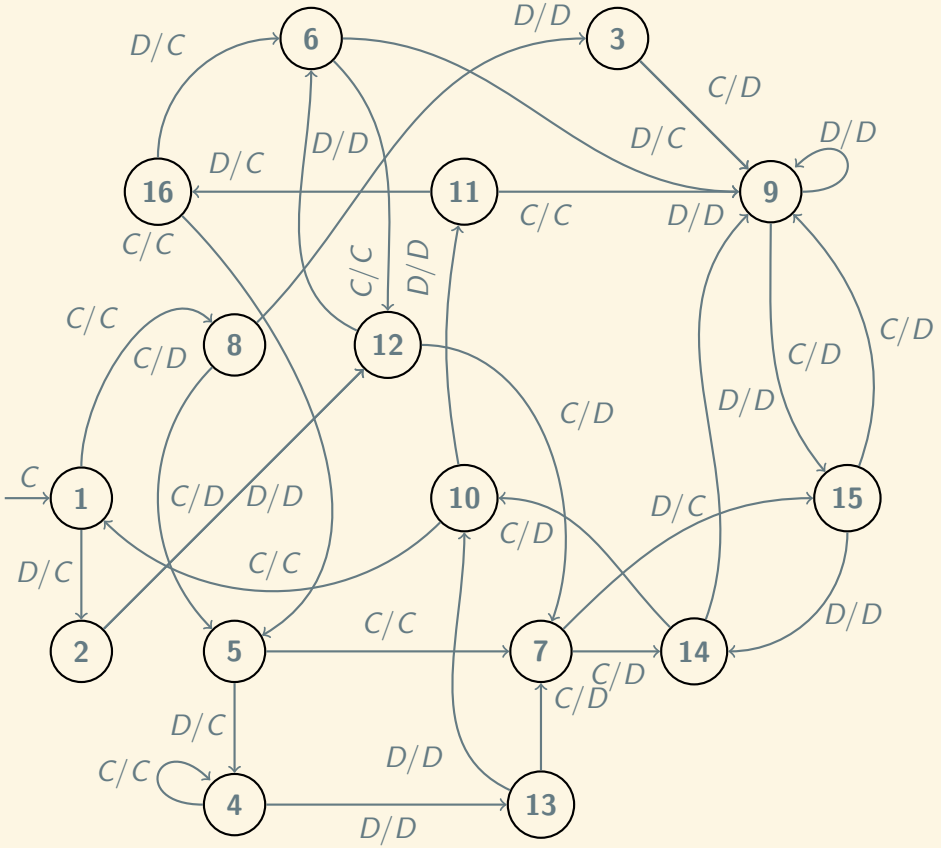
**Invasion ($N = 14$)**

| | Player | Mean $p_1$ |
|---|---|---|
| 1 | Evolved FSM 16 | 0.2096 |
| 2 | PSO Gambler 2_2_2 | 0.2042 |
| 3 | EvolvedLookerUp2_2_2 | 0.2014 |
| 4 | Evolved ANN | 0.2014 |
| 5 | Evolved ANN 5 | 0.2004 |
| 6 | Evolved HMM 5 | 0.1972 |
| 7 | PSO Gambler 1_1_1 | 0.1955 |
| 8 | Fool Me Once | 0.1955 |
| 9 | Evolved FSM 16 Noise 05 | 0.1943 |
| 10 | PSO Gambler Mem1 | 0.1920 |
| 11 | Evolved FSM 4 | 0.1918 |
| 12 | Meta Hunter | 0.1869 |
| 13 | Evolved ANN 5 Noise 05 | 0.1858 |
| 14 | Omega TFT | 0.1849 |
| 15 | Fortress4 | 0.1848 |
| 16 | **TF3** | 0.1846 |

**Resistance ($N = 14$)**

| | Player | Mean $p_{N-1}$ |
|---|---|---|
| 1 | CS | 0.9984 |
| 2 | **TF1** | 0.9973 |
| 3 | **TF2** | 0.9949 |
| 4 | Predator | 0.9941 |
| 5 | Prober 4 | 0.9863 |
| 6 | Handshake | 0.9812 |
| 7 | Winner21 | 0.9778 |
| 8 | Hard Prober | 0.9731 |
| 9 | Fortress4 | 0.9726 |
| 10 | Ripoff | 0.9669 |
| 11 | Tester | 0.9662 |
| 12 | Grudger | 0.9592 |
| 13 | **TF3** | 0.9589 |
| 14 | Davis | 0.9588 |
| 15 | Retaliate 3 | 0.9580 |
| 16 | Retaliate | 0.9576 |


0
1
Probability of cooperation
0
20
40
60
80
turns
1.0
0.5
0.0




| TF1 #1 | TF1 #2 |
|---|---|
| **1: C** | **1: C** |
| **8: C** | **8: C** |
| **5: D** | **5: D** |
| 4: C | 4: C |
| 4: C | 4: C |
| 4: C | 4: C |
| 4: C | 4: C |
| 4: C | 4: C |

164

~~164~~ 211+

**Julie Rymer - @Chadys - (10 May 2017):**

> And I really wanted to thank you all, I discovered your project because of a course where we needed to participate in an open source project, and I had the occasion to compare the welcome me and my coworkers received here compared to other people from my class who worked on different project. And I've got to said you are awesome on that part and on the help your provide to newbies  I like your project so I'll try to continue to contribute now and then !

- @NikoletaGlyn
- @opcampbell
- marcharper.codes

- github.com/Axelrod-Python/Axelrod
- gitter.im/Axelrod-Python/Axelrod
- arxiv.org/abs/1707.06920

@drvinceknight

- vknight.org/gt/

- github.com/drvinceknight/Nashpy