Solution Concepts

and the Algorithms that Respect Them

Sevan G. Ficici

Purpose is to Formalize:

- Relationship between *solution concepts* and *algorithms* that implement them
- Solution concept
- What it means to violate solution concept
- Ability of concept to order space
- (Helps deal with open-endedness?)

Purpose and Outline

- Give two examples of algorithms that don’t implement solution concept
- Present current formalism
- Give example application

Example I

Evolutionary game theory [Maynard-Smith 1982]

- infinite population
- complete mixing
- expected-value payoffs
- selection-only

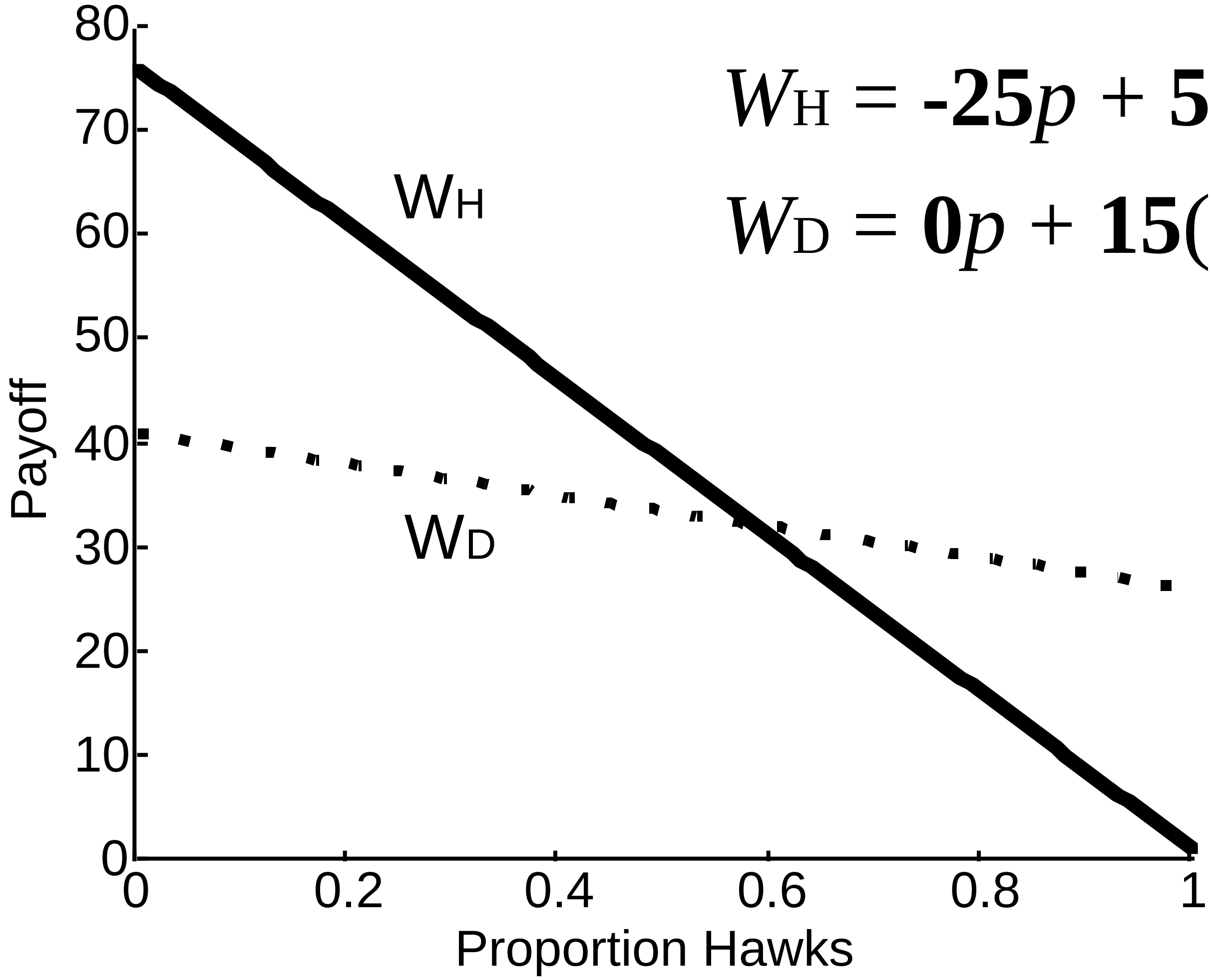
Hawk-Dove Game

Player 2

Player 1		H	D
	H	-25	50
	D	0	15

(symmetric, variable-sum game)

Payoffs



$$W_H = \mathbf{-25}p + \mathbf{50}(1\text{-}p) + w_0$$

$$W_D = \mathbf{0}p + \mathbf{15}(1\text{-}p) + w_0$$

$w_0 = 26$

Standard Replicator

Replicator

$$p' = \frac{pW_H}{pW_H + (1-p)W_D}$$

Evaluator

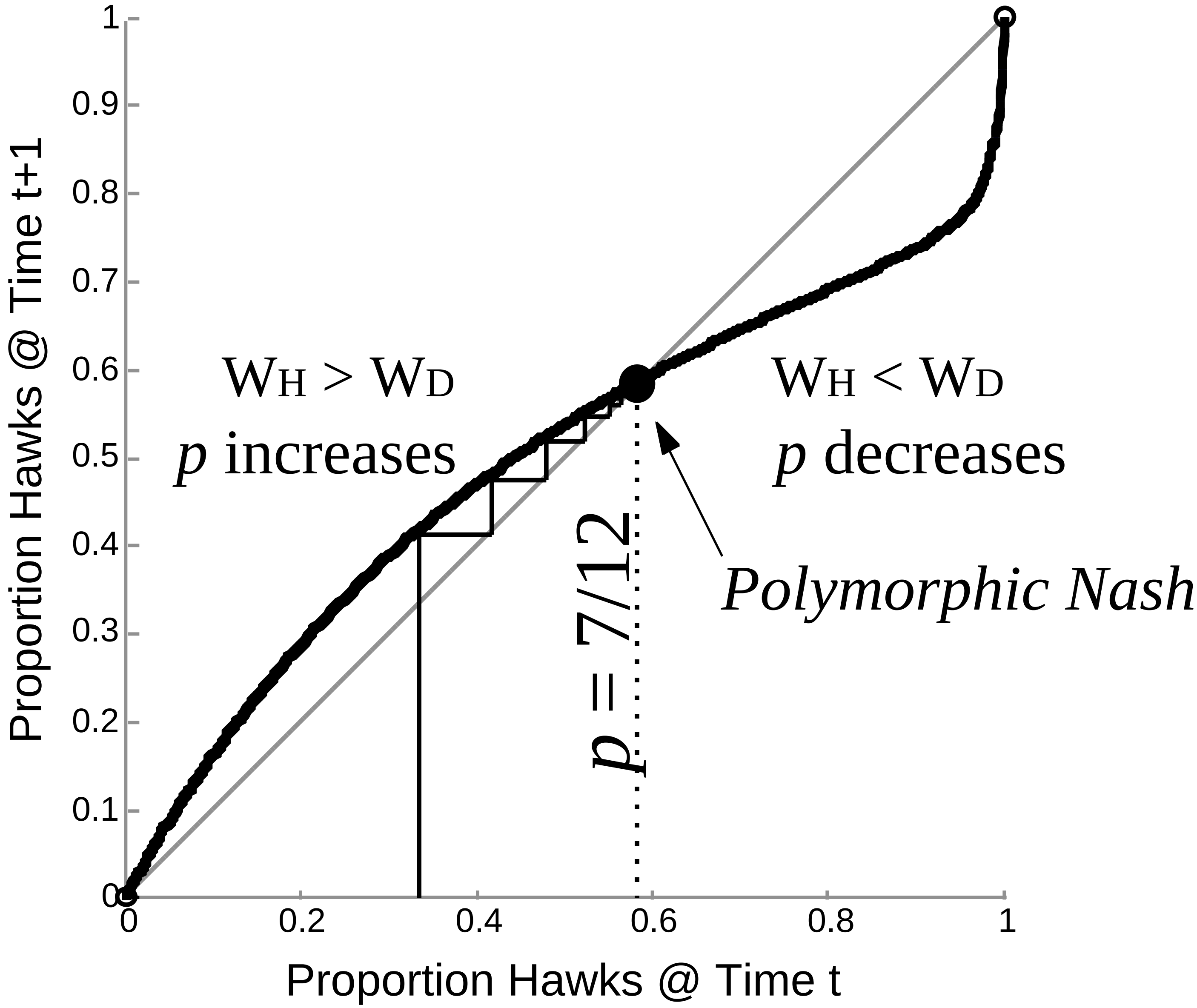
$$F(p) = \frac{W_H}{W_H + W_D}$$

Selection

$$S(f, p) = \frac{pf}{pf + (1-p)(1-f)}$$

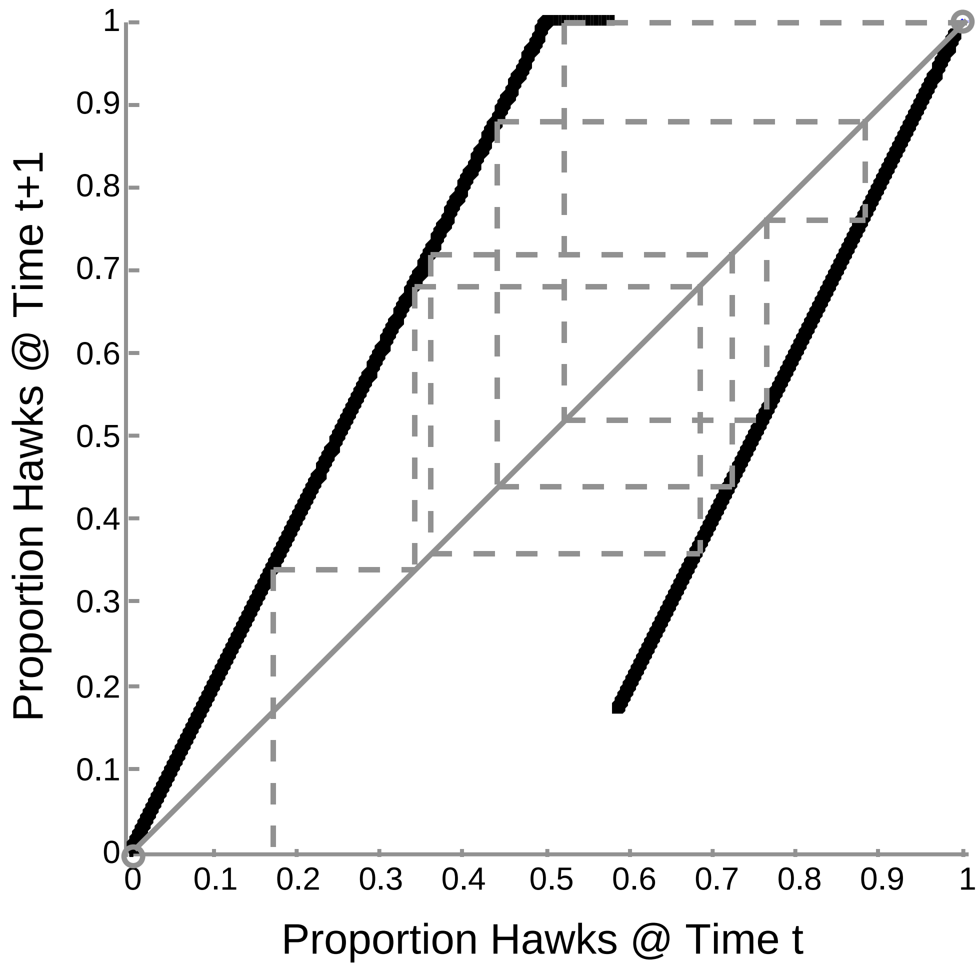
$$p' = S(\, F(\, p),\, p\,)$$

Fitness Proportionate



Truncation

Regime 1: 42% £ k £ 50%

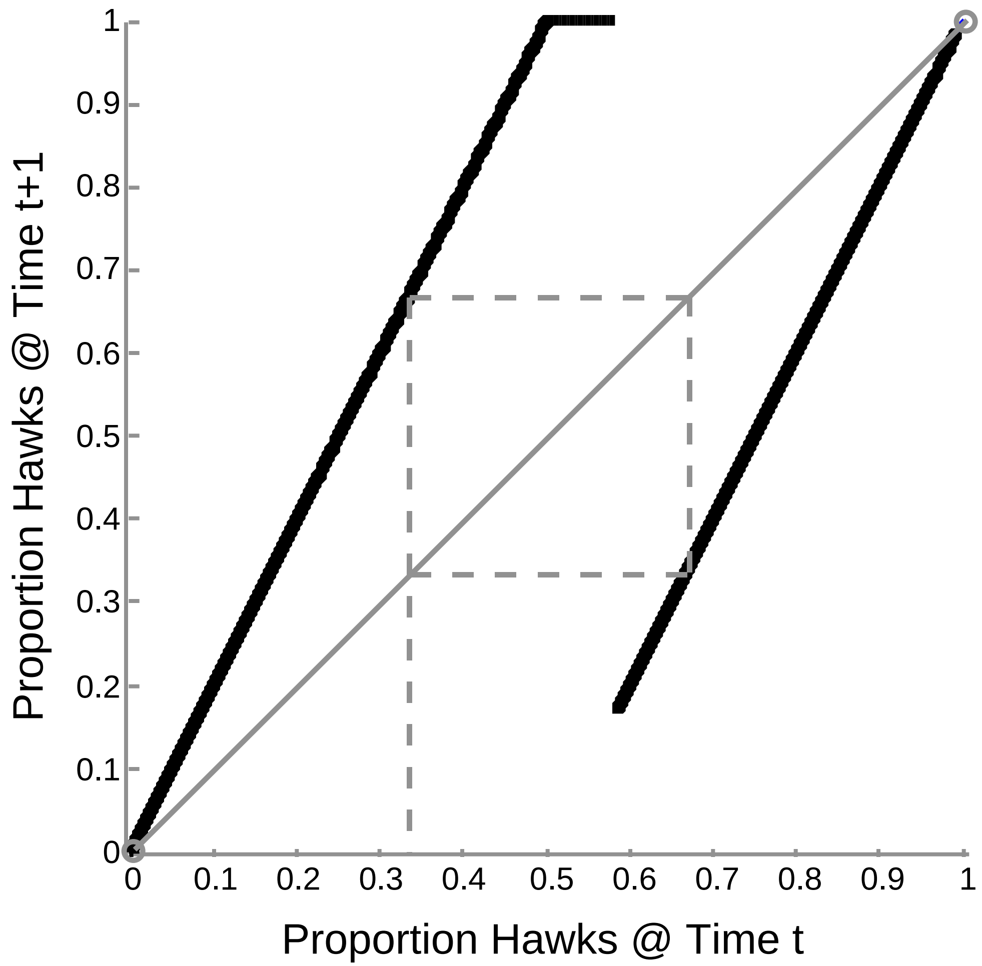


k = 50%

All Hawks

Truncation

Regime 1: 42% £ *k* £ 50%



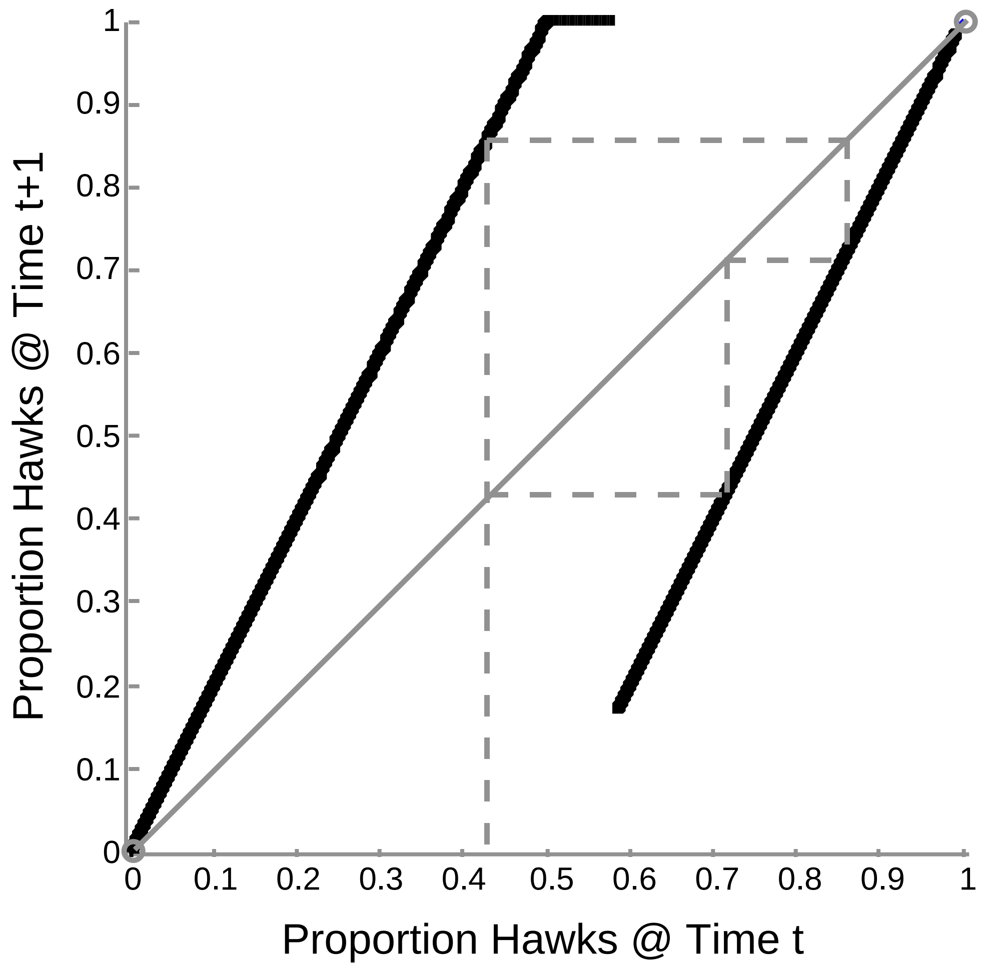
$k = 50\%$

2 Cycle

$P_H = 1/3$

Truncation

Regime 1: 42% £ *k* £ 50%



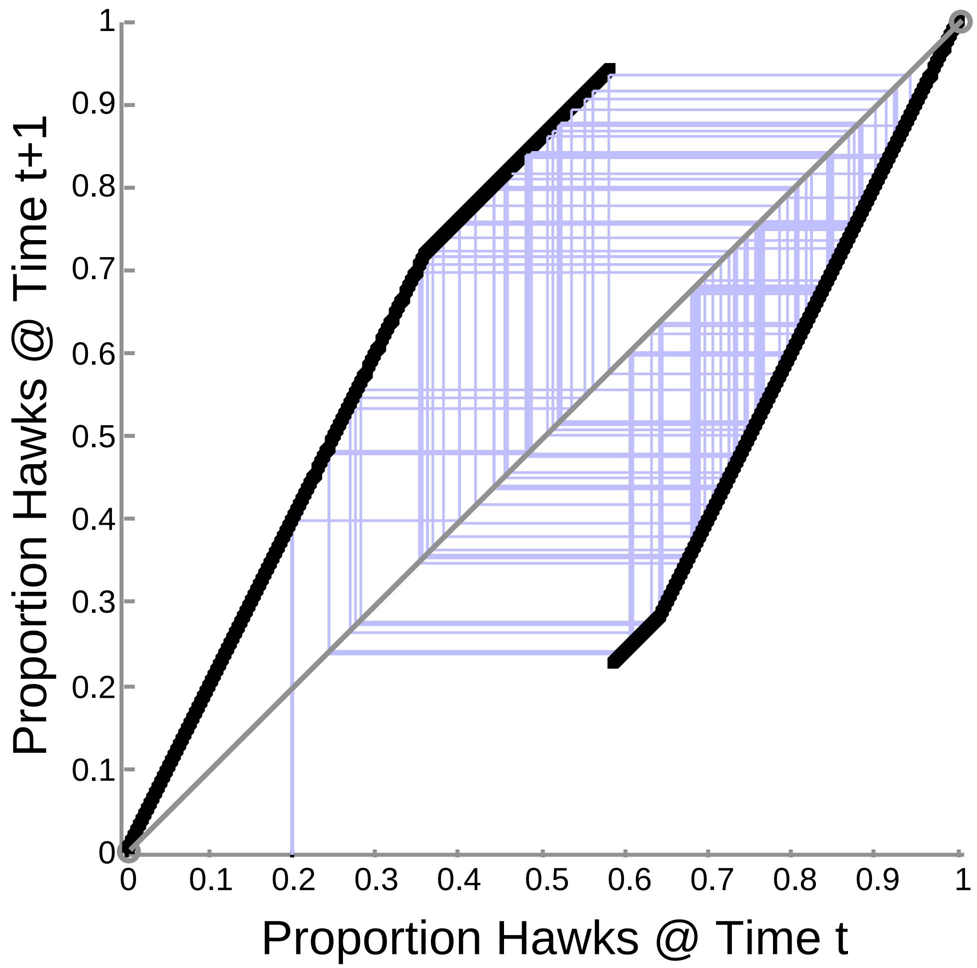
k = 50%

3 Cycle

P_H = 3/7

Truncation

Regime 2: 31% £ *k* £ 41%

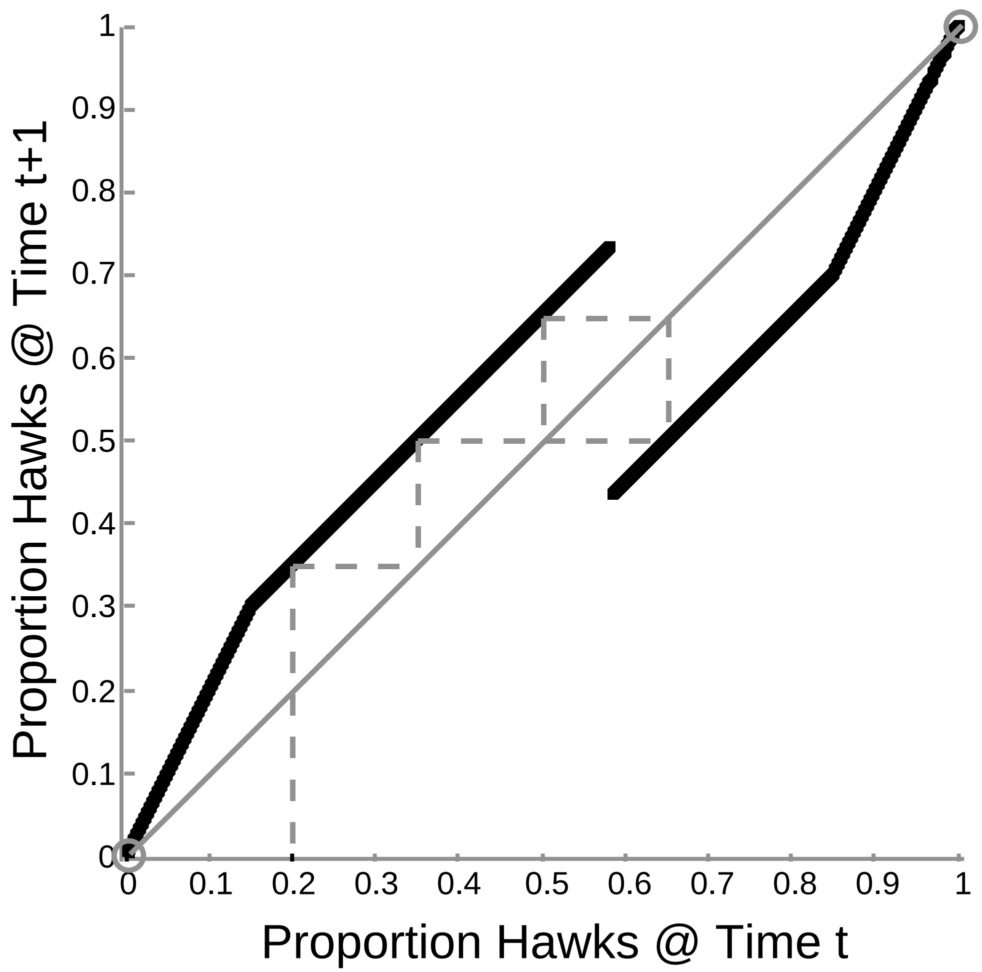


k = 36%

Λ = 0.69

Truncation

Regime 3: 0% < *k* £ 30%

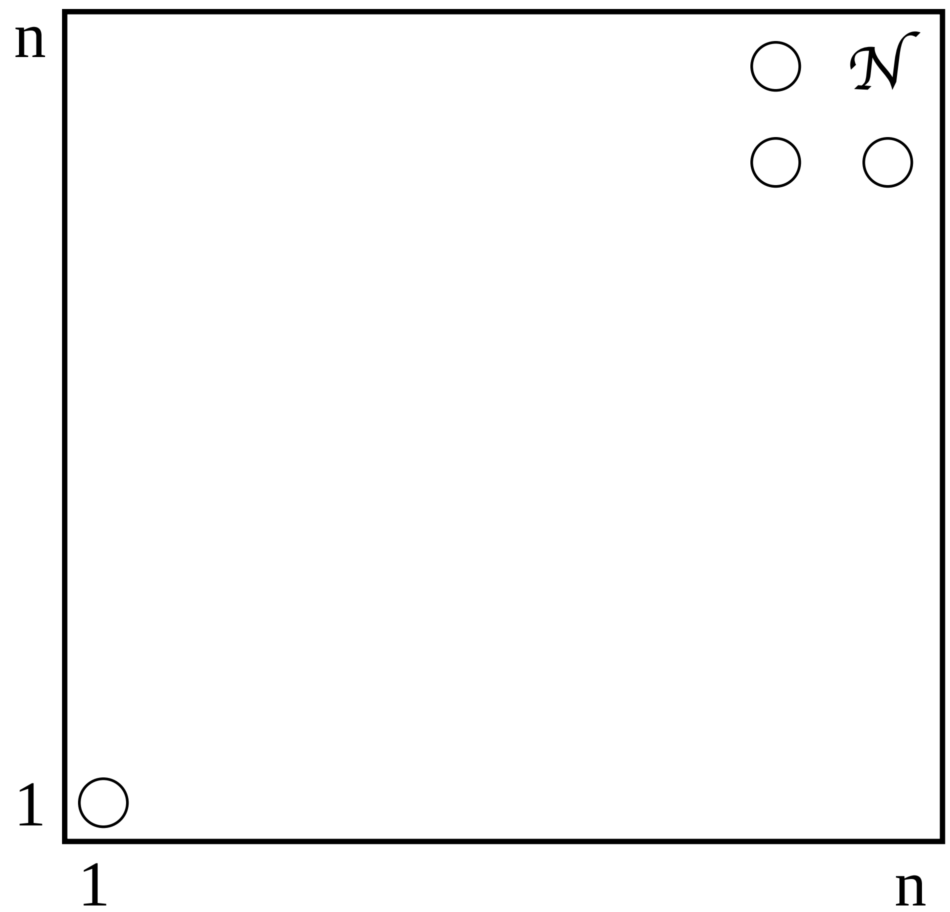


$k = 15\%$

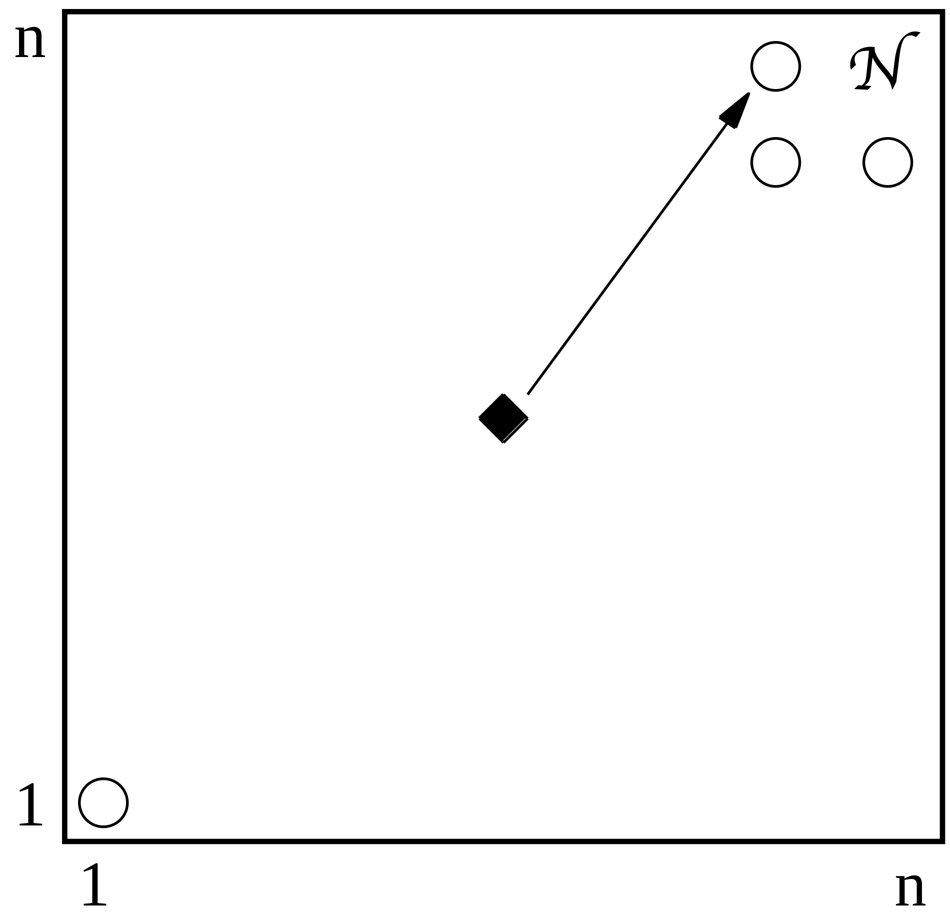
Example II

- Domination Tournament [Stanley and Miikkulainen 2002]
- Memory mechanism for symmetric zero-sum games
- Memory begins with a single strategy
- Each subsequent strategy must beat entire contents of memory to enter it
- Thus, each new additon *dominates* all previous strategies (no intransitivity!)

Intransitive Numbers Game



Violation in Numbers Game



Formalism I

- n-player *game* G, each player with a set of pure strategies
- *Sub-game*: each player has subset of strats
- *Configuration* K is an n-tuple of *strategy complexes*: < X1, X2 >
- *Strategy complex* X is a set of pure strategies; may have other attributes
- *Solution* K* is a configuration that meets certain requirements of solution concept

Formalism II

- *Solution set* is set of all possible solutions, give a game and solution concept: S*(G, O)
- *Solution concept* O defines solution set and a preference relation
- We *prefer* K* to K
- We *prefer* Ka to Kb iff:

FOR ALL Gb: THERE EXISTS Ga s.t.

Ga Gb (this gives us transitivity in preference)

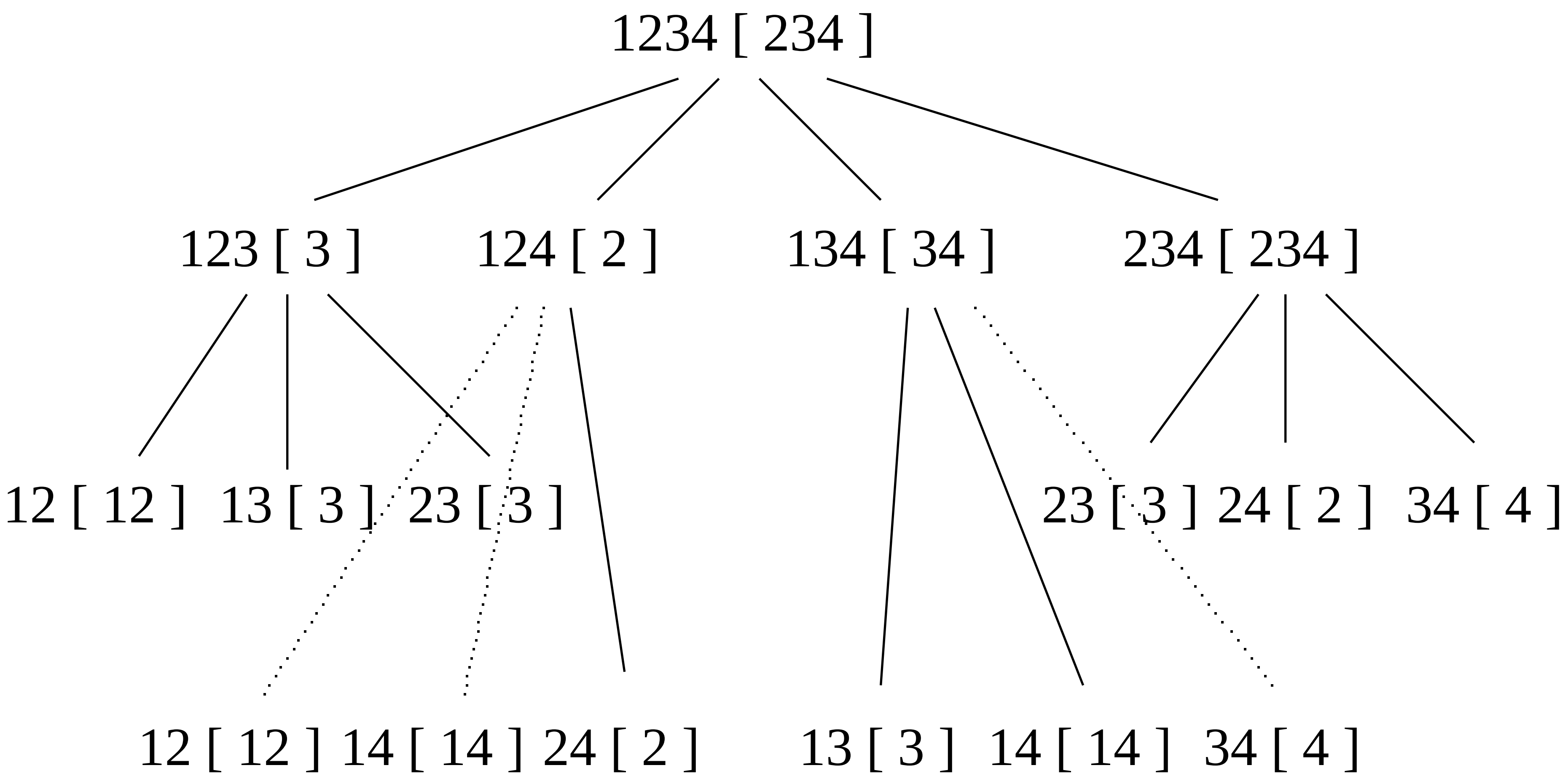
Example

$$G = \begin{matrix} 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 \\ 1 & 1 & 0 & -1 \\ 0 & -1 & 1 & 0 \end{matrix}$$

Pareto Solution Concept:

Layer 0: 2 3 4
Layer 1: 1

Subgames and Solutions



Prefence Order

Pareto Solution Concept:

Layer 0: 2 3 4
Layer 1: 1

Preference:

<u>2 3 4</u>
3 4
4
3
2
1 4
1 2
1

Features of Formalism

- Plug in any solution concept you want
- Gives pref. order an algorithm must respect
- Allows us to compare different solution concepts on the same game:
  – How structured is preference order?
  – "Most fit" solution concept may give little structure, hence belief that no objective measure can exist in coevolution

Features II

- All solutions are equally prefered
- If we really like one solution over another, then we want to *refine* the solution concept
  - e.g., Pareto dominant Nash, risk dominant Nash