Multi-agent learning

Comparing algorithms empirically

Gerard Vreeswijk, Intelligent Software Systems, Computer Science Department, Faculty of Sciences, Utrecht University, The Netherlands.

Sunday 21st June, 2020

Motivation

Motivation

- Until 2005, say, MAL research was typically done this way:

Motivation

- Until 2005, say, MAL research was typically done this way:

  1. Invent a nifty MAL algorithm

Motivation

- Until 2005, say, MAL research was typically done this way:

  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.

Motivation

- Until 2005, say, MAL research was typically done this way:
  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.

Motivation

- Until 2005, say, MAL research was typically done this way:

    1. Invent a nifty MAL algorithm
    2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
    3. Show that the new algorithm performs much better.

- Later, MAL algorithms were compared more systematically.

Motivation

- Until 2005, say, MAL research was typically done this way:

  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.

- Later, MAL algorithms were compared more systematically.

- By pitting games against each other.

Motivation

- Until 2005, say, MAL research was typically done this way:

  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.

- Later, MAL algorithms were compared more systematically.

  - By pitting games against each other.
  - Through elimination ("knock-out").

Motivation

- Until 2005, say, MAL research was typically done this way:

  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.

- Later, MAL algorithms were compared more systematically.

  - By pitting games against each other.
  - Through elimination ("knock-out").
  - Through evolutionary approaches.

Motivation

- Until 2005, say, MAL research was typically done this way:
  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.
- Later, MAL algorithms were compared more systematically.
  - By pitting games against each other.
  - Through elimination ("knock-out").
  - Through evolutionary approaches.
- We will look into the **methodology** (= procedures) and **methods** (= tools) of the various approaches.

Motivation

- Until 2005, say, MAL research was typically done this way:
  1. Invent a nifty MAL algorithm
  2. Compare the nifty algorithm with a handful of trendy MAL algorithms on a handful of well-known games.
  3. Show that the new algorithm performs much better.
- Later, MAL algorithms were compared more systematically.
  - By pitting games against each other.
  - Through elimination ("knock-out").
  - Through evolutionary approaches.
- We will look into the **methodology** (= procedures) and **methods** (= tools) of the various approaches.

  The **algorithms** themselves and the **outcomes** of the comparison are of secondary interest in our review today!

Work discussed

Work discussed

- Axelrod: organising tournaments to let algorithms play the IPD.

  Axelrod, Robert. *The evolution of cooperation.* (1984) New York: Basic Books.

Work discussed

- Axelrod: organising tournaments to let algorithms play the IPD.

  Axelrod, Robert. *The evolution of cooperation.* (1984) New York: Basic Books.

- Zawadzki *et al.*: straight but thorough.

  Zawadzki, Erik, Asher Lipson, and Kevin Leyton-Brown. "Empirically evaluating multiagent learning algorithms." arXiv preprint arXiv:1401.8074 (2014).

Work discussed

- Axelrod: organising tournaments to let algorithms play the IPD.

  Axelrod, Robert. *The evolution of cooperation.* (1984) New York: Basic Books.

- Zawadzki *et al.*: straight but thorough.

  Zawadzki, Erik, Asher Lipson, and Kevin Leyton-Brown. "Empirically evaluating multiagent learning algorithms." arXiv preprint arXiv:1401.8074 (2014).

- Bouzy *et al.*: elimination ("knock-out").

  Bouzy, Bruno, and Marc Métivier. "Multi-agent learning experiments on repeated matrix games" in: Proc. of the 27th Int. Conf. on Machine Learning (2010).

Work discussed

- Axelrod: organising tournaments to let algorithms play the IPD.

  Axelrod, Robert. *The evolution of cooperation.* (1984) New York: Basic Books.

- Zawadzki *et al.*: straight but thorough.

  Zawadzki, Erik, Asher Lipson, and Kevin Leyton-Brown. "Empirically evaluating multiagent learning algorithms." arXiv preprint arXiv:1401.8074 (2014).

- Bouzy *et al.*: elimination ("knock-out").

  Bouzy, Bruno, and Marc Métivier. "Multi-agent learning experiments on repeated matrix games" in: Proc. of the 27th Int. Conf. on Machine Learning (2010).

- Airiau *et al.*: evolutionary dynamics.

  Airiau, Stéphane, Sabyasachi Saha, and Sandip Sen. "Evolutionary tournament-based comparison of learning and non-learning algorithms for iterated games" in: *Journal of Artificial Societies and Social Simulation* **10**.3 (2007).

Round robin tournament

Round robin tournament

Given a pool of games $\mathcal{G}$ to test on, all approaches have in common that they have a grand table of head-to-head scores:

	A_1	A_2	$\dots$	A_{12}	avg
algorithm A_1	3.2	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

Round robin tournament

Given a pool of games $\mathcal{G}$ to test on, all approaches have in common that they have a grand table of head-to-head scores:

	A_1	A_2	$\dots$	A_{12}	avg
algorithm A_1	3.2	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

- Entries are performance measures for the protagonist (row), which almost always is average payoff

Round robin tournament

Given a pool of games $\mathcal{G}$ to test on, all approaches have in common that they have a grand table of head-to-head scores:

	A_1	A_2	$\dots$	A_{12}	avg
algorithm A_1	3.2	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

- Entries are performance measures for the protagonist (row), which almost always is average payoff (alternatives: no-regret, ...).

Round robin tournament

Given a pool of games $\mathcal{G}$ to test on, all approaches have in common that they have a grand table of head-to-head scores:

	A_1	A_2	$\dots$	A_{12}	avg
algorithm A_1	3.2	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

- Entries are performance measures for the protagonist (row), which almost always is average payoff (alternatives: no-regret, . . .).
- Often each entry is computed multiple times to even out randomness in algorithms (which are implementations of response rules).

Round robin tournament

Given a pool of games $\mathcal{G}$ to test on, all approaches have in common that they have a grand table of head-to-head scores:

	A_1	A_2	$\dots$	A_{12}	avg
algorithm A_1	3.2	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

- Entries are performance measures for the protagonist (row), which almost always is average payoff (alternatives: no-regret, ...).
- Often each entry is computed multiple times to even out randomness in algorithms (which are implementations of response rules).
- Sometimes there is a settling-in phase (a.k.a. burn-in phase) in which payoffs are not yet recorded.

Work of Axelrod (1980, 1984)

Axelrod receiving the National Medal of Science (2014)

Axelrod: tournament for the repeated prisoner's dilemma

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.
- Winner: Tit-for-tat.

Axelrod, Robert. "Effective choice in the prisoner's dilemma." Journal of conflict resolution 24.1 (1980): 3-25.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.
- Winner: Tit-for-tat.

Axelrod, Robert. "Effective choice in the prisoner's dilemma." Journal of conflict resolution 24.1 (1980): 3-25.

- Second tournament: 64 contestants.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.
- Winner: Tit-for-tat.

Axelrod, Robert. "Effective choice in the prisoner's dilemma." Journal of conflict resolution 24.1 (1980): 3-25.

- Second tournament: 64 contestants. All contestants were informed about the results of the first tournament.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.
- Winner: Tit-for-tat.

Axelrod, Robert. "Effective choice in the prisoner's dilemma." Journal of conflict resolution 24.1 (1980): 3-25.

- Second tournament: 64 contestants. All contestants were informed about the results of the first tournament. Winner: Tit-for-tat.

Axelrod: tournament for the repeated prisoner's dilemma

- One game to test: the prisoner's dilemma.
- Contestants: 14 constructed algorithms + 1 random = 15: Tit-for-tat, Shubik, Nydegger, Joss, . . . , Random.

  Response rules (algorithms) were mostly reactive. One could hardly speak of learning.
- Grand table: all pairs play 200 rounds. This was repeated 5 times to even out randomness.
- Winner: Tit-for-tat.

Axelrod, Robert. "Effective choice in the prisoner's dilemma." Journal of conflict resolution 24.1 (1980): 3-25.

- Second tournament: 64 contestants. All contestants were informed about the results of the first tournament. Winner: Tit-for-tat.
- In 2012, Alexander Stewart and Joshua Plotkin ran a variant of Axelrod's tournament with 19 strategies to test the effectiveness of the then newly discovered Zero-Determinant strategies.

Work of Zawadzki *et al.* (2014)

Zawadzki *et al.*

Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random.

Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.

Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.

- Games: a suite of 13 interesting families, $\mathcal{D} = D_1, \dots, D_{13}$: $D_1 =$ games with normal covariant random payoffs; $D_2 =$ Bertrand oligopoly; $D_3 =$ Cournot duopoly; $D_4 =$ dispersion games; $D_5 =$ grab the dollar type games; $D_6 =$ guess two thirds of the average games; ...

Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.

- Games: a suite of 13 interesting families, $\mathcal{D} = D_1, \dots, D_{13}$: $D_1 =$ games with normal covariant random payoffs; $D_2 =$ Bertrand oligopoly; $D_3 =$ Cournot duopoly; $D_4 =$ dispersion games; $D_5 =$ grab the dollar type games; $D_6 =$ guess two thirds of the average games; ...

- Game pool: 600 games: 100 games for each size 2^2, 4^2, 6^2, 8^2, 10^2, randomly selected from $\mathcal{D}$, and 100 games of dimension 2×2 from Rapoport's catalogue.

Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.
- Games: a suite of 13 interesting families, $\mathcal{D} = D_1, \dots, D_{13}$: $D_1 =$ games with normal covariant random payoffs; $D_2 =$ Bertrand oligopoly; $D_3 =$ Cournot duopoly; $D_4 =$ dispersion games; $D_5 =$ grab the dollar type games; $D_6 =$ guess two thirds of the average games; . . .
- Game pool: 600 games: 100 games for each size 2^2, 4^2, 6^2, 8^2, 10^2, randomly selected from $\mathcal{D}$, and 100 games of dimension 2×2 from Rapoport's catalogue.
- Grand table: each algorithm pair plays all 600 games for 10^4 rounds.

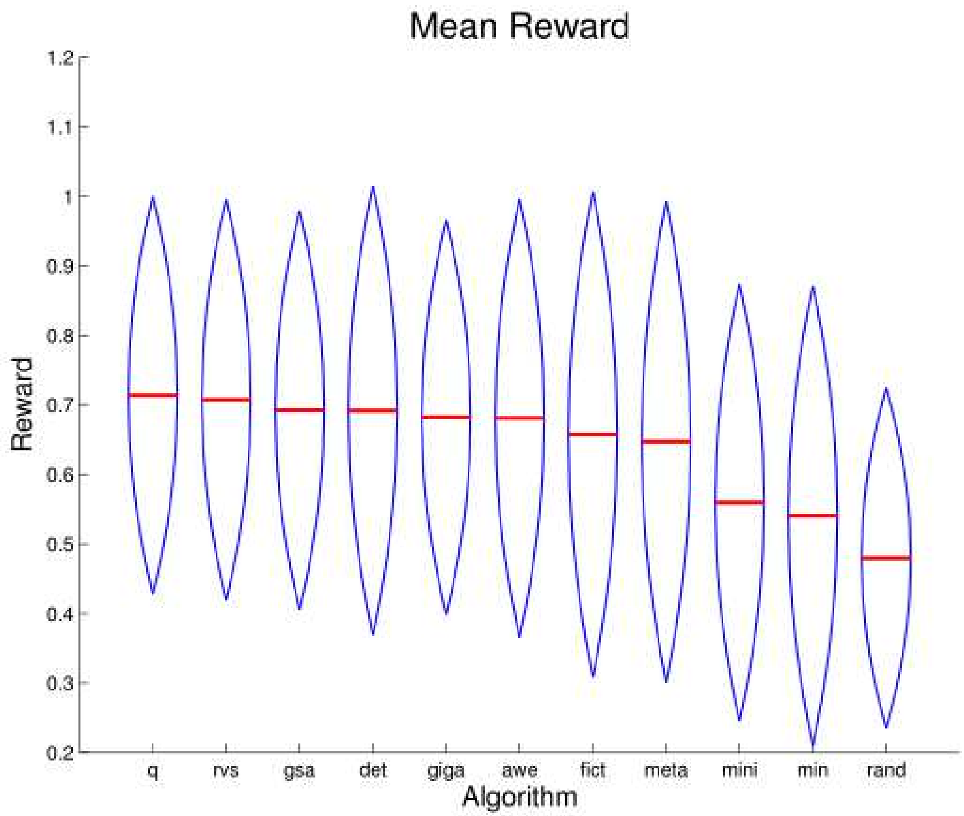
Zawadzki *et al.*

- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.
- Games: a suite of 13 interesting families, $\mathcal{D} = D_1, \dots, D_{13}$: $D_1 =$ games with normal covariant random payoffs; $D_2 =$ Bertrand oligopoly; $D_3 =$ Cournot duopoly; $D_4 =$ dispersion games; $D_5 =$ grab the dollar type games; $D_6 =$ guess two thirds of the average games; ...
- Game pool: 600 games: 100 games for each size 2^2, 4^2, 6^2, 8^2, 10^2, randomly selected from $\mathcal{D}$, and 100 games of dimension 2×2 from Rapoport's catalogue.
- Grand table: each algorithm pair plays all 600 games for 10^4 rounds.
- Evaluation: through non-parametric tests and squared heat plots.

Zawadzki *et al.*

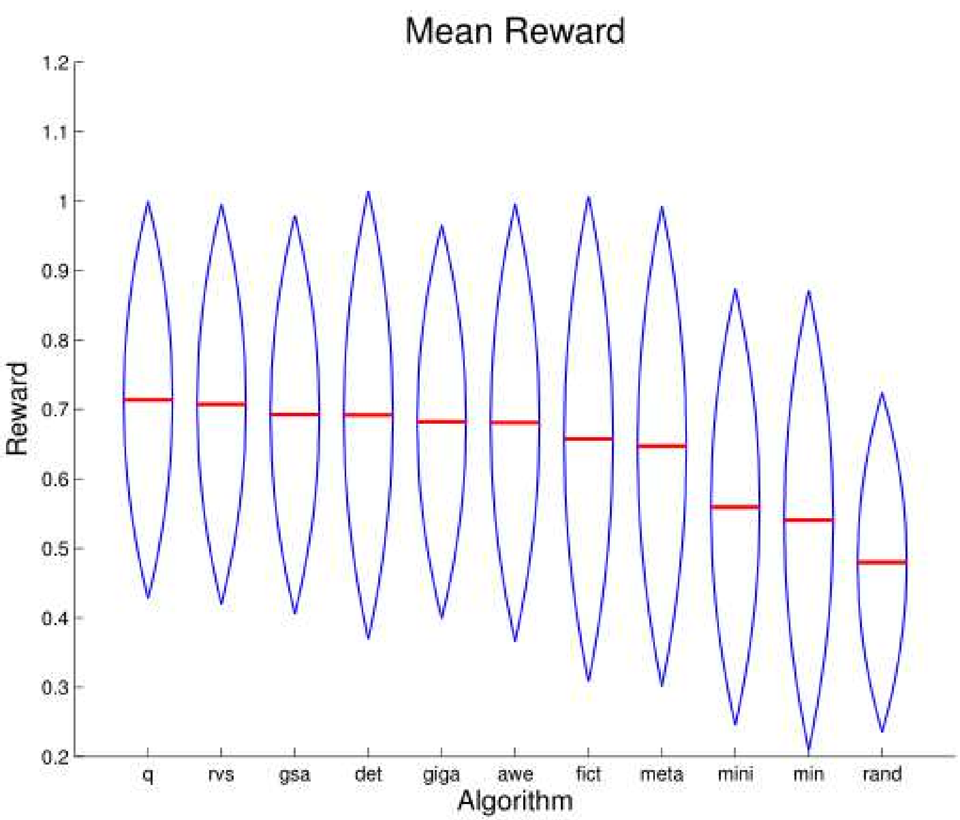
- Contestants: FP, Determinate, Awesome, Meta, WoLF-IGA, GSA, RVS, QL, Minmax-Q, Minmax-Q-IDR, Random. A motivation for this set of 11 algorithms, other than "state-of-the-art" wasn't given.
- Games: a suite of 13 interesting families, $\mathcal{D} = D_1, \dots, D_{13}$: $D_1 =$ games with normal covariant random payoffs; $D_2 =$ Bertrand oligopoly; $D_3 =$ Cournot duopoly; $D_4 =$ dispersion games; $D_5 =$ grab the dollar type games; $D_6 =$ guess two thirds of the average games; ...
- Game pool: 600 games: 100 games for each size 2^2, 4^2, 6^2, 8^2, 10^2, randomly selected from $\mathcal{D}$, and 100 games of dimension 2×2 from Rapoport's catalogue.
- Grand table: each algorithm pair plays all 600 games for 10^4 rounds.
- Evaluation: through non-parametric tests and squared heat plots.
- Conclusion: Q-learning is the overall winner.

Zawadzki *et al.*

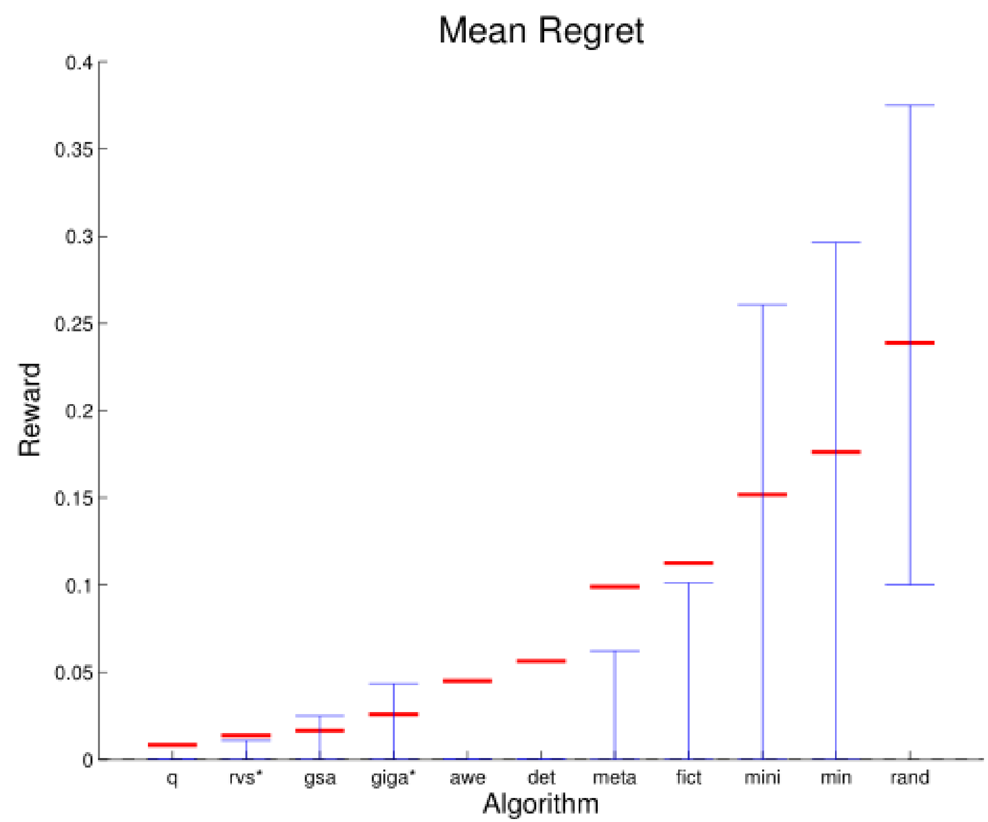


Mean reward over all opponents and games.

Zawadzki *et al.*

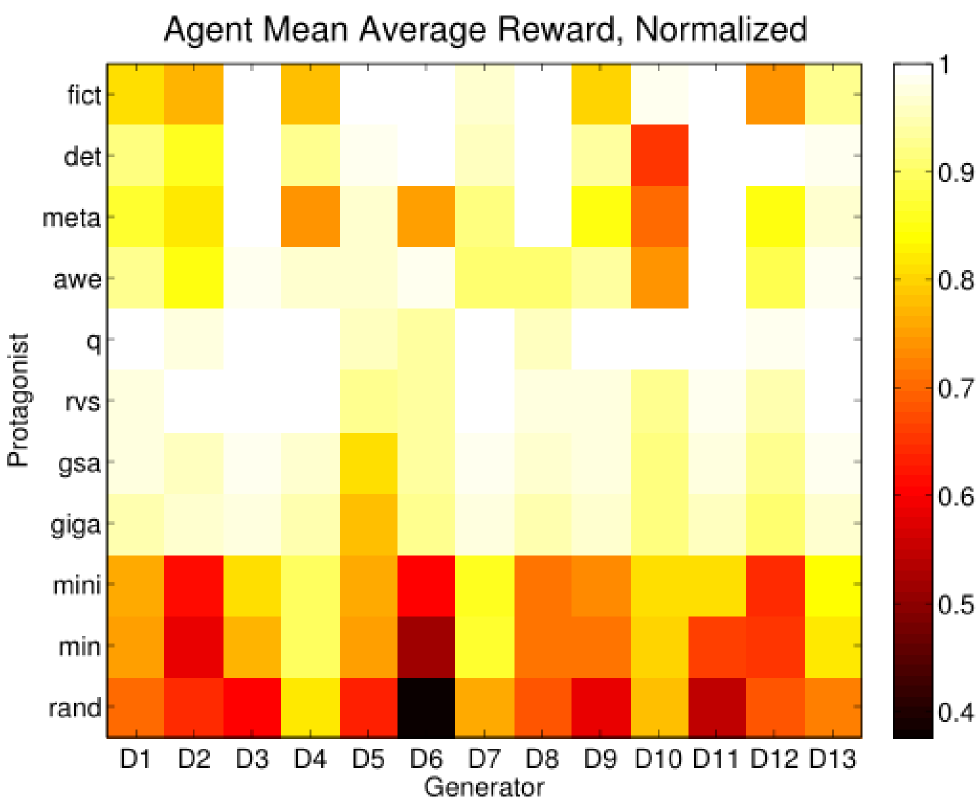


Mean reward over all opponents and games.



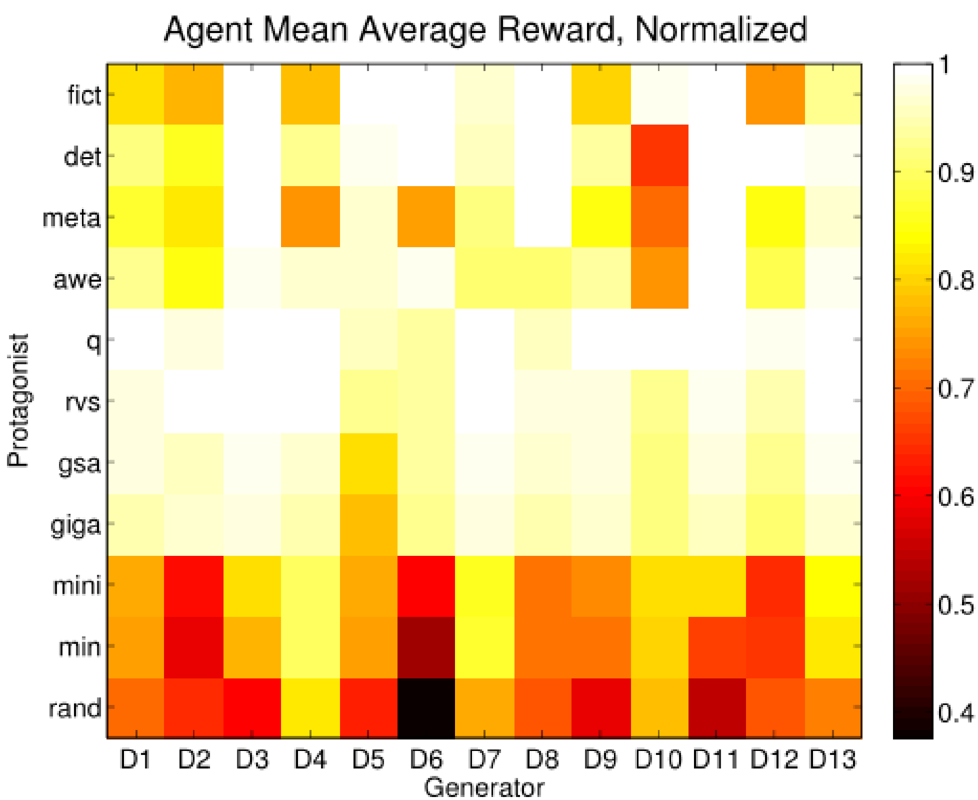
Mean regret over all opponents and games.

Zawadzki *et al.*

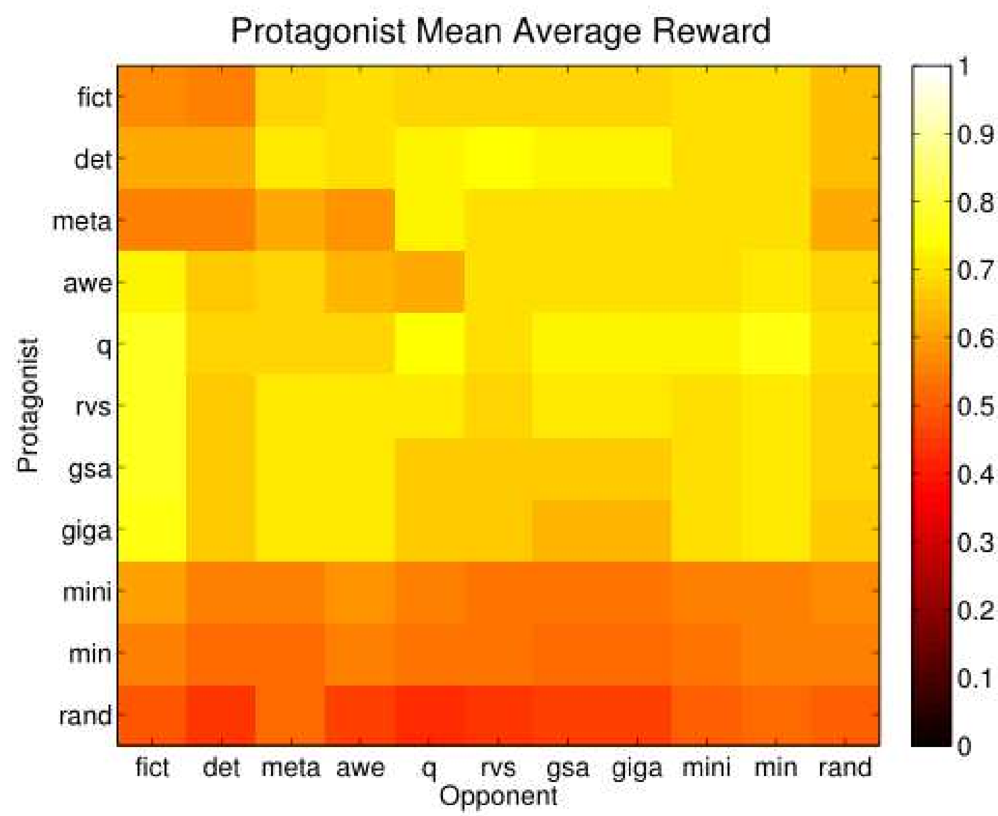


Mean reward against different game suites.

Zawadzki *et al.*



Mean reward against different game suites.



Mean reward against different opponents.

Parametric test: paired t-test

	A_1			A_2			A_3			*dots*		
A_1	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.1	3.1	4.7	5.1	1.1	1.2	3.5	4.2	3.8	...	...	...
A_2	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.7	3.5	4.1	4.9	0.9	1.9	3.7	4.7	4.5	...	...	...

Parametric test: paired t-test

	A_1			A_2			A_3			$dots$		
A_1	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.1	3.1	4.7	5.1	1.1	1.2	3.5	4.2	3.8	...	...	...
A_2	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.7	3.5	4.1	4.9	0.9	1.9	3.7	4.7	4.5	...	...	...

Paired t-test:

Parametric test: paired t-test

	A_1			A_2			A_3			*dots*		
A_1	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.1	3.1	4.7	5.1	1.1	1.2	3.5	4.2	3.8	...	...	...
A_2	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.7	3.5	4.1	4.9	0.9	1.9	3.7	4.7	4.5	...	...	...

Paired t-test:

- Compute the average difference $\bar{X}_D$, and the average standard deviation of differences $\bar{s}_D$, of all n pairs (we see nine here).

Parametric test: paired t-test

	A_1			A_2			A_3			*dots*		
A_1	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.1	3.1	4.7	5.1	1.1	1.2	3.5	4.2	3.8	...	...	...
A_2	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.7	3.5	4.1	4.9	0.9	1.9	3.7	4.7	4.5	...	...	...

Paired t-test:

- Compute the average difference $\bar{X}_D$, and the average standard deviation of differences $\bar{s}_D$, of all n pairs (we see nine here).
- If the two series are generated by the same random process, the test statistic $t = \bar{X}_D / (\bar{s}_D \sqrt{n})$ should follow the Student's t-distribution with mean 0 and $n - 1$ degrees of freedom.

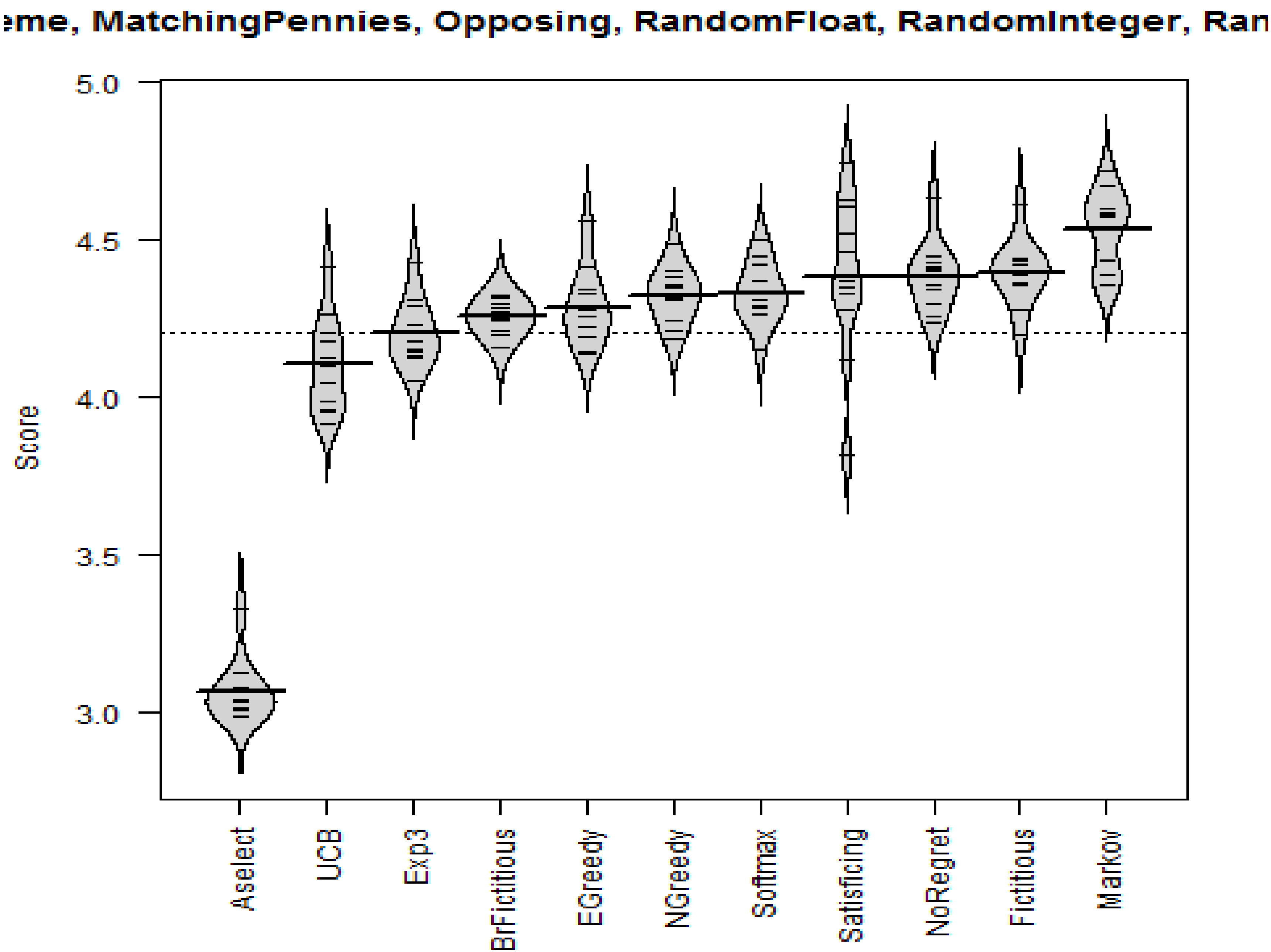
Parametric test: paired t-test

	A_1			A_2			A_3			*dots*		
A_1	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.1	3.1	4.7	5.1	1.1	1.2	3.5	4.2	3.8	...	...	...
A_2	G_1	G_2	G_3	G_1	G_2	G_3	G_1	G_2	G_3	...	...	...
	2.7	3.5	4.1	4.9	0.9	1.9	3.7	4.7	4.5	...	...	...

Paired t-test:

- Compute the average difference $\bar{X}_D$, and the average standard deviation of differences $\bar{s}_D$, of all n pairs (we see nine here).
- If the two series are generated by the same random process, the test statistic $t = \bar{X}_D / (\bar{s}_D \sqrt{n})$ should follow the Student's t-distribution with mean 0 and $n - 1$ degrees of freedom.
- If t is too eccentric, then we'll have to reject that possibility, since eccentric values of t are unlikely ("have a low p-value").

Parametric test: paired t-test



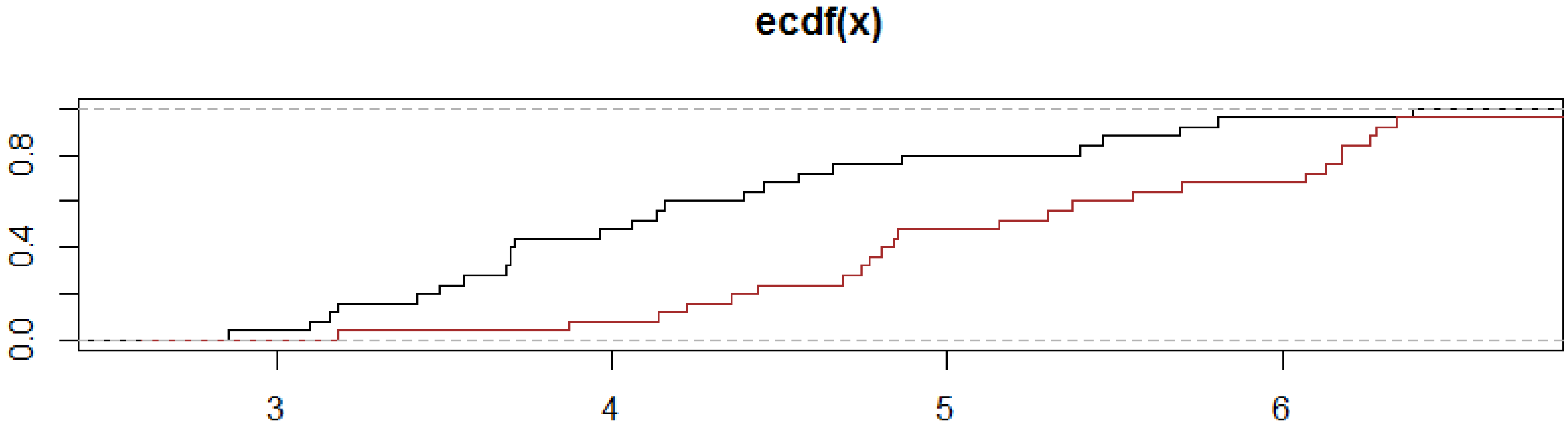
Non-parametric test: the Kolmogorov-Smirnov test

Non-parametric test: the Kolmogorov-Smirnov test

- Test whether two distributions are generated by the same random process. H_0: yes. H_1: no.

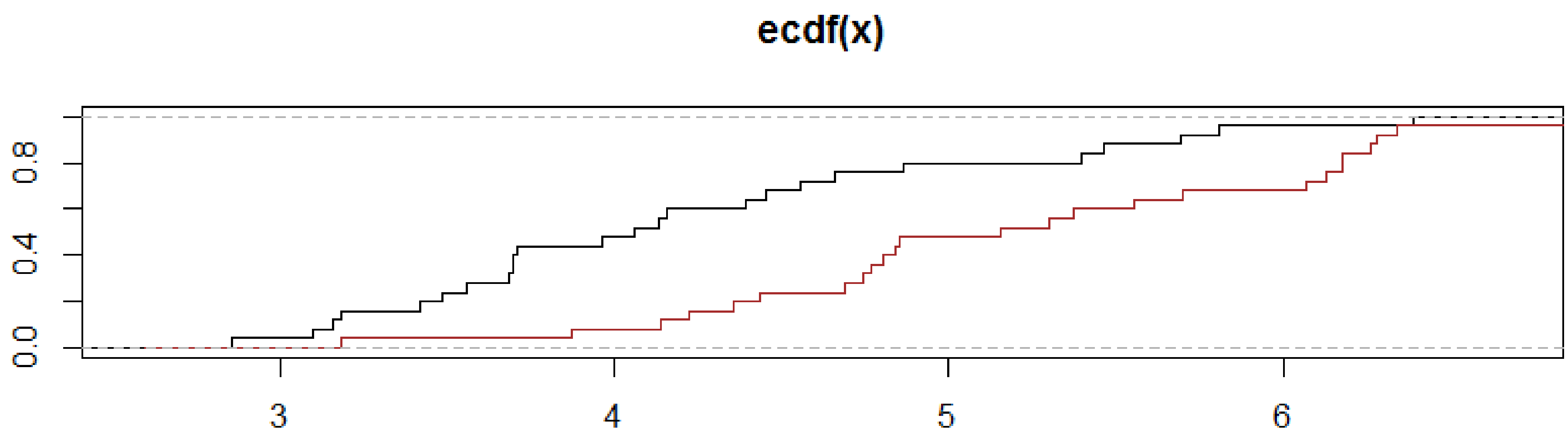
Non-parametric test: the Kolmogorov-Smirnov test

- Test whether two distributions are generated by the same random process. H_0: yes. H_1: no.
- The test statistic is the maximum distance between the empirical cumulative distribution functions of the two samples.



Non-parametric test: the Kolmogorov-Smirnov test

- Test whether two distributions are generated by the same random process. H_0: yes. H_1: no.
- The test statistic is the maximum distance between the empirical cumulative distribution functions of the two samples.



- The p-value is the probability of seeing a test statistic (i.e., max distance) as high as the one observed, under the assumption that both samples were drawn from the same distribution.

Zawadzki *et al.*

Variations / extensions. Investigate:

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards.

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.
- The correlation between distance to nearest Nash and average reward.

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.
- The correlation between distance to nearest Nash and average reward.

- Which algorithms probabilistically dominate which other algorithms. (Cf. article for a definition of this concept.)

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.
- The correlation between distance to nearest Nash and average reward.

- Which algorithms probabilistically dominate which other algorithms. (Cf. article for a definition of this concept.)

  Outcome: Q-Learning is the only algorithm that is not probabilistically dominated by other algorithms.

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.
- The correlation between distance to nearest Nash and average reward.

- Which algorithms probabilistically dominate which other algorithms. (Cf. article for a definition of this concept.)

  Outcome: Q-Learning is the only algorithm that is not probabilistically dominated by other algorithms.
- the difference between average reward and maxmin value (enforceable payoff).

Zawadzki *et al.*

Variations / extensions. Investigate:

- The relation between game sizes and rewards. Outcome: no relation.
- The correlation between regret and average reward.
- The correlation between distance to nearest Nash and average reward.

- Which algorithms probabilistically dominate which other algorithms. (Cf. article for a definition of this concept.)

  Outcome: Q-Learning is the only algorithm that is not probabilistically dominated by other algorithms.

- the difference between average reward and maxmin value (enforceable payoff).

  Outcome: Q-Learning attained an enforceable payoff more frequently than any other algorithm.

Work of Bouzy *et al.* (2010)

Bouzy *et al.* (2010)

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.
- Final ranking: UCB, M3, Sat, JR, . . .

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.
- Final ranking: UCB, M3, Sat, JR, . . .
- Evaluation: Plot with x-axis = log rounds and y-axis the rank of that algorithm w.r.t. performance.

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.
- Final ranking: UCB, M3, Sat, JR, ...
- Evaluation: Plot with x-axis = log rounds and y-axis the rank of that algorithm w.r.t. performance.
- Bouzy *et al.* are familiar with the work of Airiau *et al.* and Zawadzki *et al.*.

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.
- Final ranking: UCB, M3, Sat, JR, . . .
- Evaluation: Plot with x-axis = log rounds and y-axis the rank of that algorithm w.r.t. performance.
- Bouzy *et al.* are familiar with the work of Airiau *et al.* and Zawadzki *et al.*.

  Contrary to Airiau *et al.* and Zawadzki *et al.*., the ranking still fluctuates after 3×10^6 rounds

Bouzy *et al.* (2010)

- Contestants: Minimax, FP, QL, JR, Sat, M3, UCB, Exp3, HMC, Bully, Optimistic, Random (12 games).
- Games: random 2-player, 3×3-actions, with payoffs in $\mathbb{Z} \cap [-9, 9]$ (if I understand correctly, else it's $[-9, 9]$).
- Grand table: each pair plays 3×10^6 rounds (!) on a random game. Restart 100 times to even out randomness.
- Final ranking: UCB, M3, Sat, JR, ...
- Evaluation: Plot with x-axis = log rounds and y-axis the rank of that algorithm w.r.t. performance.
- Bouzy *et al.* are familiar with the work of Airiau *et al.* and Zawadzki *et al.*.

  Contrary to Airiau *et al.* and Zawadzki *et al.*, the ranking still fluctuates after 3×10^6 rounds ... ?!

Bouzy *et al.* (2010)

Variation: eliminate by rank.

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, . . .

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, . . .

Variation: eliminate by lag.

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, ...

Variation: eliminate by lag.

- Algorithm: Repeat:

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, ...

Variation: eliminate by lag.

- Algorithm: Repeat:
  - Organise a tournament. If the difference between the cumulative returns of the two worst performers is larger than $600/\sqrt{n^T}$ (n^T the number of tournaments performed), then the laggard is removed.

Bouzy *et al.* (2010)

Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, ...

Variation: eliminate by lag.

- Algorithm: Repeat:
  - Organise a tournament. If the difference between the cumulative returns of the two worst performers is larger than $600/\sqrt{n^T}$ (n^T the number of tournaments performed), then the laggard is removed. The laggard is also removed if the global ranking has not changed during $100n_p^2(n_p - 1)^2$ tournaments since the last elimination (n_p is the current number of players).

Bouzy *et al.* (2010)

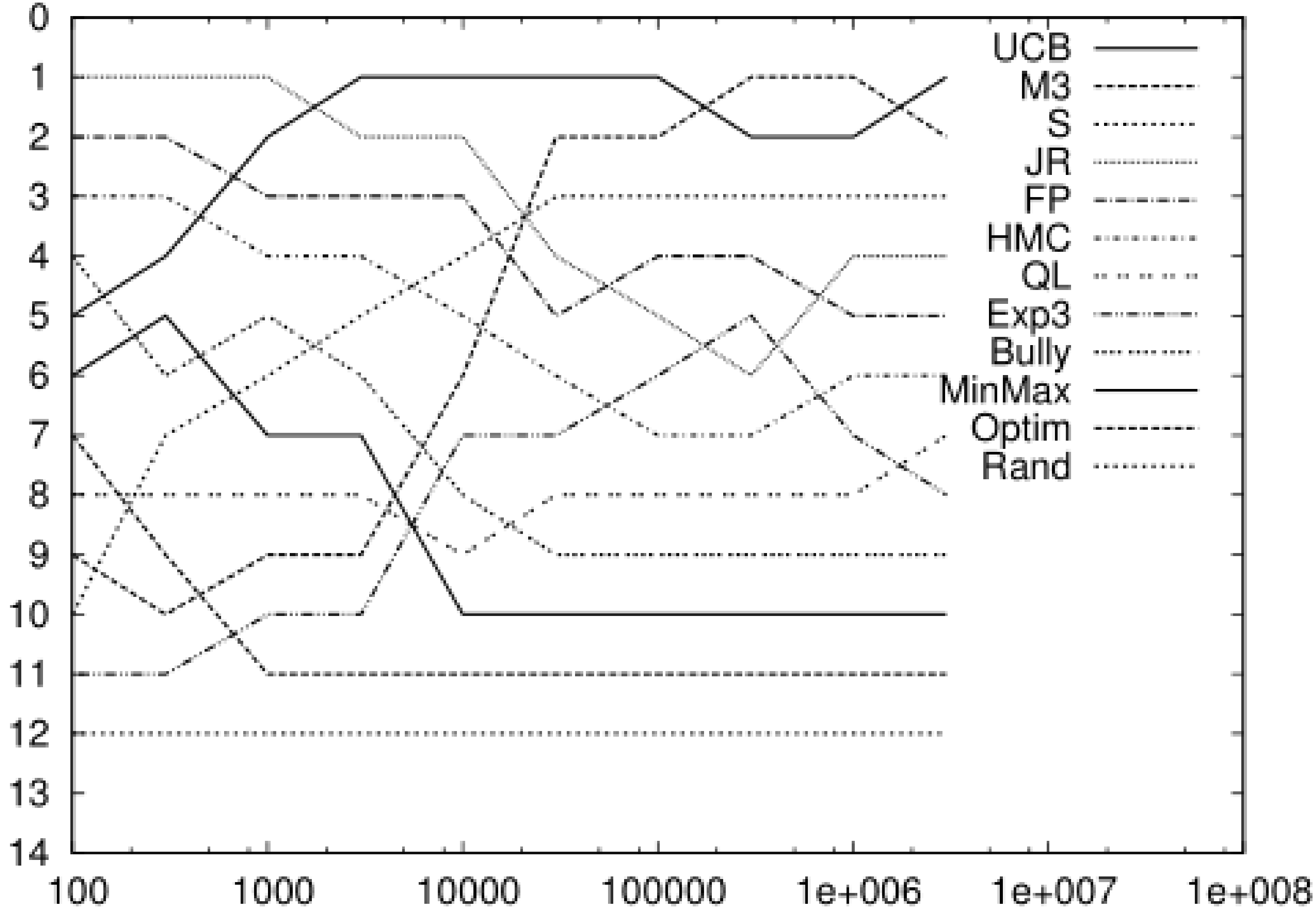
Variation: eliminate by rank.

- Algorithm: Repeat:
  - Rank, eliminate the worst, and subtract all payoffs earned against that player from the revenues of all survivors.
- Final ranking: M3, Sat, UCB, JR, ...

Variation: eliminate by lag.

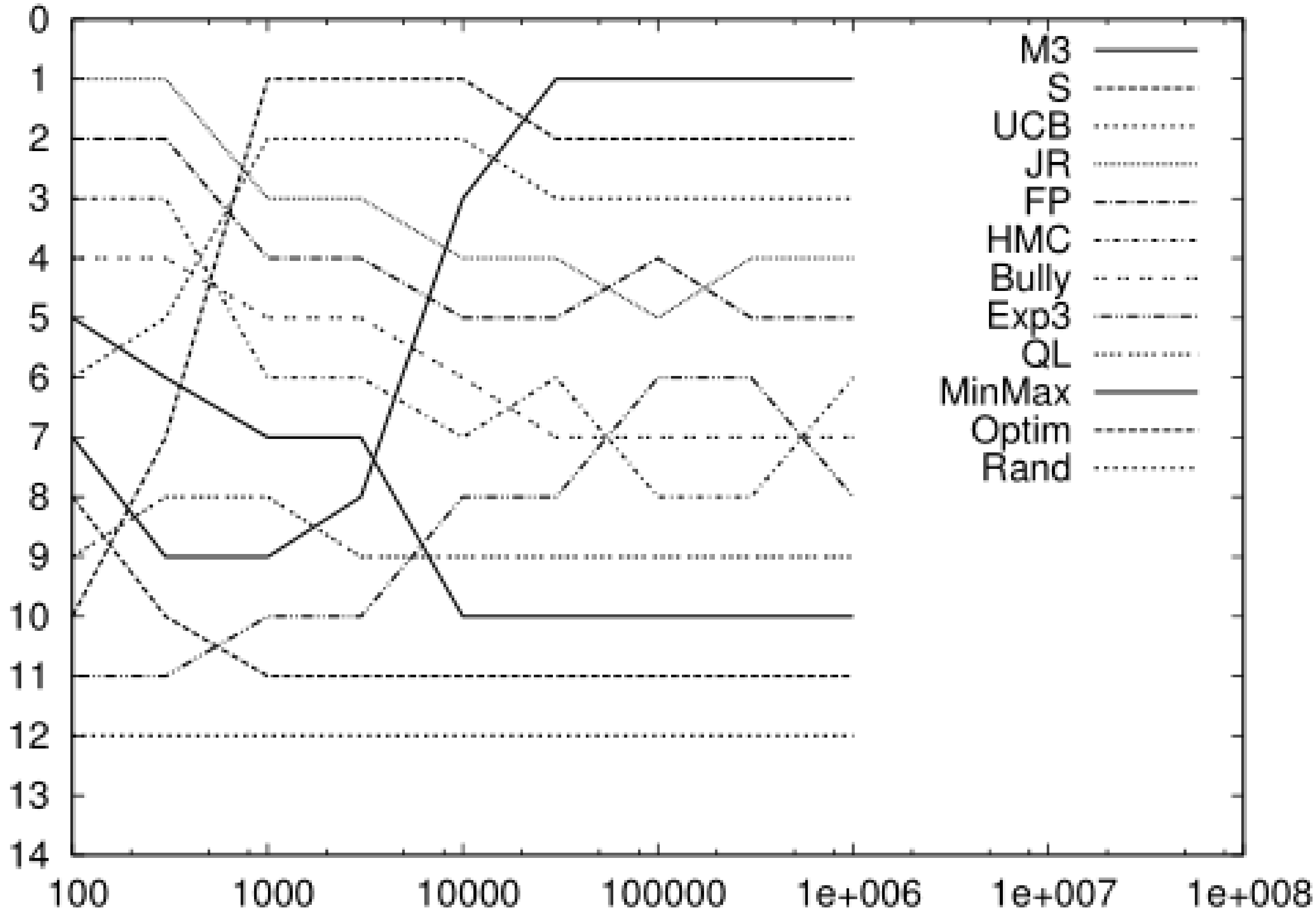
- Algorithm: Repeat:
  - Organise a tournament. If the difference between the cumulative returns of the two worst performers is larger than $600/\sqrt{n^T}$ (n^T the number of tournaments performed), then the laggard is removed. The laggard is also removed if the global ranking has not changed during $100n_p^2(n_p-1)^2$ tournaments since the last elimination (n_p is the current number of players).
- Final ranking: M3, Sat, UCB, FP, ...

Bouzy *et al.*



Ranking evolution according to the number of steps played in games (logscale). The key is ordered according to the final ranking.

Bouzy *et al.*



Ranking based on eliminations (logscale). The key is ordered according to the final ranking.

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...
- Only competitive games (zero-sum payoffs): Exp3, M3, Minimax, JR, ...

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...
- Only competitive games (zero-sum payoffs): Exp3, M3, Minimax, JR, ...
- Specific matrix games: penalty game, climbing game, coordination game, ...

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...
- Only competitive games (zero-sum payoffs): Exp3, M3, Minimax, JR, ...
- Specific matrix games: penalty game, climbing game, coordination game, ...
- Different number of actions ($n \times n$ games).

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...
- Only competitive games (zero-sum payoffs): Exp3, M3, Minimax, JR, ...
- Specific matrix games: penalty game, climbing game, coordination game, ...
- Different number of actions ($n \times n$ games).

Conclusion: M3, Sat, and UCB perform best. Do not maintain averages but geometric (decaying) averages of payoffs.

Bouzy *et al.* (2010)

Variation: select sub-classes of games.

- Only cooperative games (shared payoffs): Exp3, M3, Bully, JR, ...
- Only competitive games (zero-sum payoffs): Exp3, M3, Minimax, JR, ...
- Specific matrix games: penalty game, climbing game, coordination game, ...
- Different number of actions ($n \times n$ games).

Conclusion: M3, Sat, and UCB perform best. Do not maintain averages but geometric (decaying) averages of payoffs.

Another interesting direction is exploring why Exp3 is the best MAL player on both cooperative games and competitive games, but not on general-sum games, and to exploit this fact to design a new MAL algorithm.

Work of Airiau *et al.* (2007)

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (. . .) algorithms."

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: “We chose well-known (…) algorithms.”

- Games: 57 distinct 2×2 normal form games (Brams, 1994) “provides a sufficiently rich environment for our purpose”.

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (...) algorithms."

- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".

- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (...) algorithms."

- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".

- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.

- Methodology = evolutionary dynamics:

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (. . .) algorithms."

- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".
- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.
- Methodology = evolutionary dynamics:

  1. Start with a population of algorithms.

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (. . .) algorithms."

- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".

- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.

- Methodology = evolutionary dynamics:

  1. Start with a population of algorithms. Initial distribution is reciprocal to performance in previous experiments.

Airiau *et al.*

- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

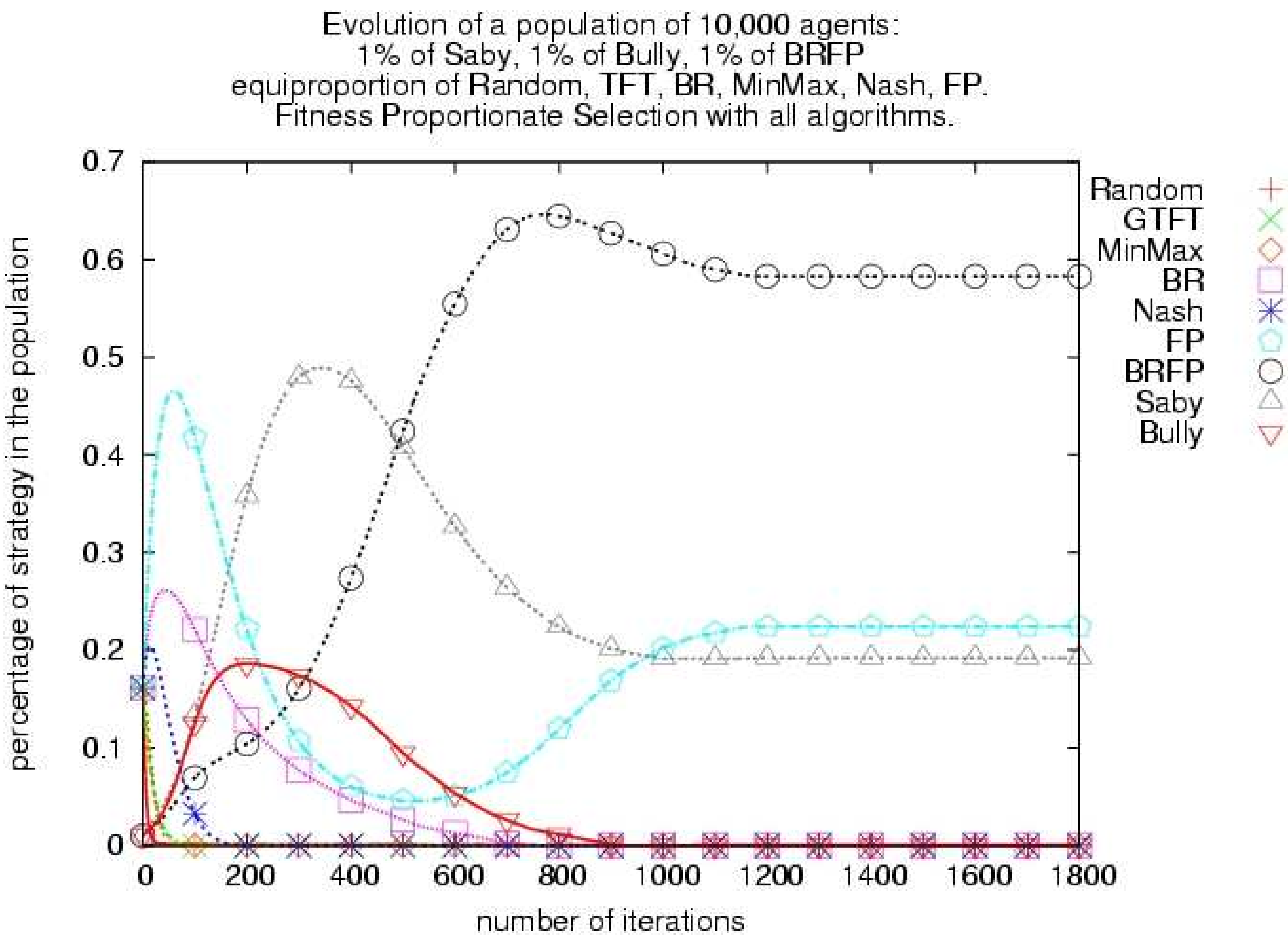
  Motivation: "We chose well-known (...) algorithms."
- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".
- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.
- Methodology = evolutionary dynamics:
  1. Start with a population of algorithms. Initial distribution is reciprocal to performance in previous experiments.
  2. Apply, e.g., fitness-proportionate selection through $p_{\text{select}} = (1 + \delta/\delta_{\max})/2$, where δ is normalised performance measure.

Airiau *et al.*

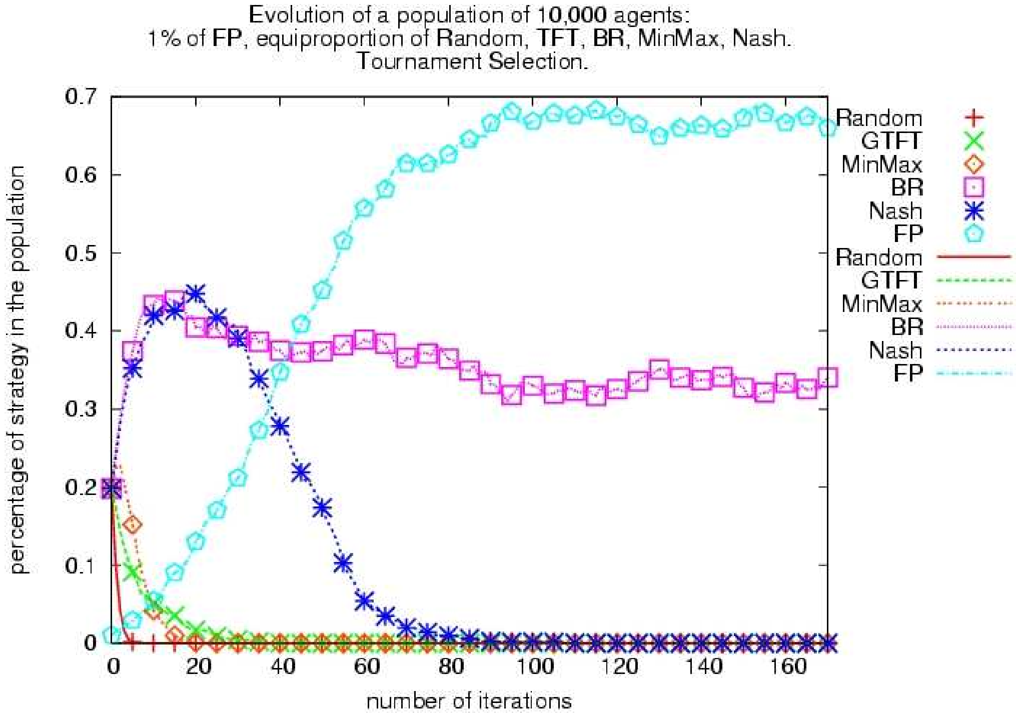
- Contestants: Maxmin, Nash, Generalised TFT, BR, FP, BRFP, Bully, Saby (unpublished!?), Random (9).

  Motivation: "We chose well-known (. . .) algorithms."
- Games: 57 distinct 2×2 normal form games (Brams, 1994) "provides a sufficiently rich environment for our purpose".
- Grand table: Each algorithm pair plays a random game for 1000 rounds. Restart 20 times.
- Methodology = evolutionary dynamics:
  1. Start with a population of algorithms. Initial distribution is reciprocal to performance in previous experiments.
  2. Apply, e.g., fitness-proportionate selection through $p_{\text{select}} = (1 + \delta/\delta_{\max})/2$, where δ is normalised performance measure.
- Final ranking: BRFP, FP, Saby, . . .

Airiau *et al.* results



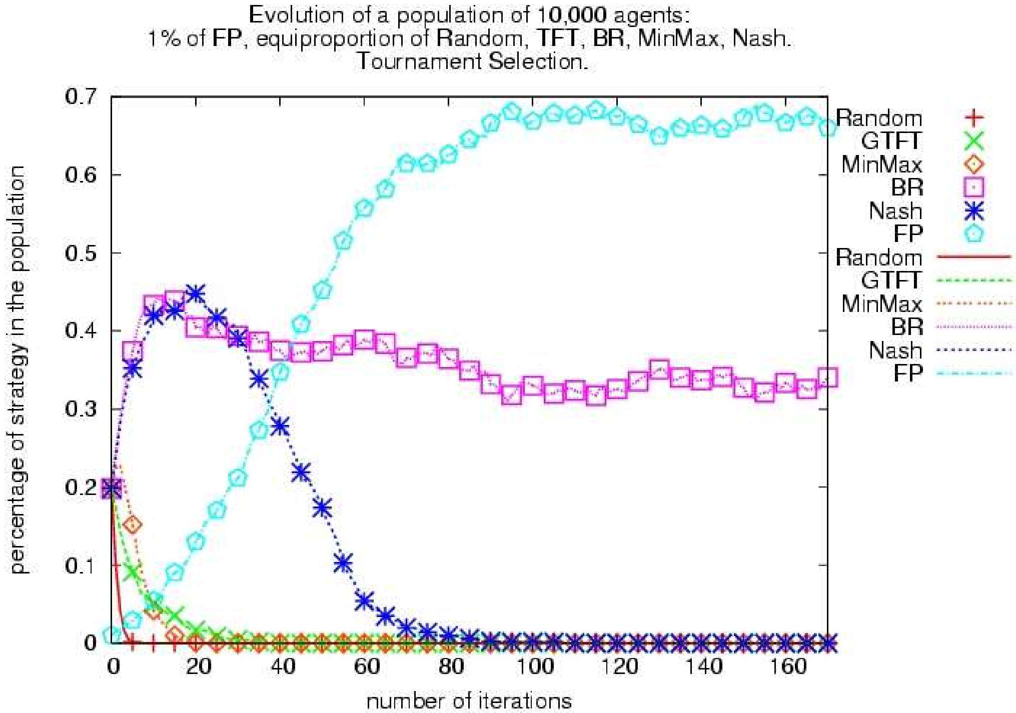
Evolutionary tournament with six algorithms: 1% FP and equiproportion of R, GTFT, BR, MaxMin, Nash each.



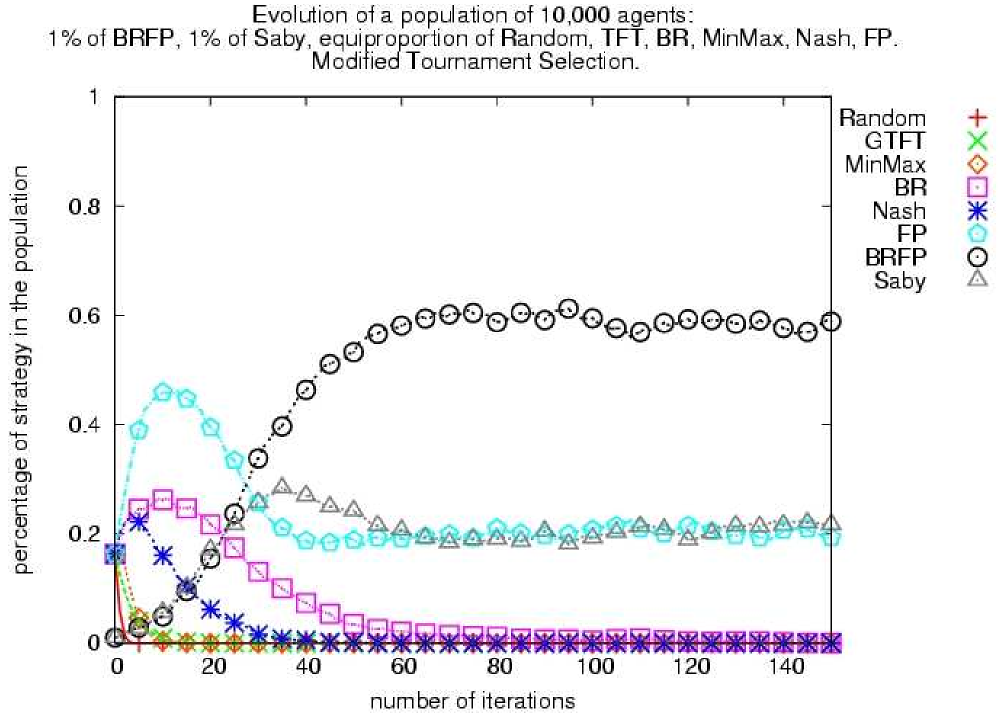
With tournament selection.

Airiau *et al.*

Evolutionary tournament with six algorithms: 1% FP and equiproportion of R, GTFT, BR, MaxMin, Nash each.



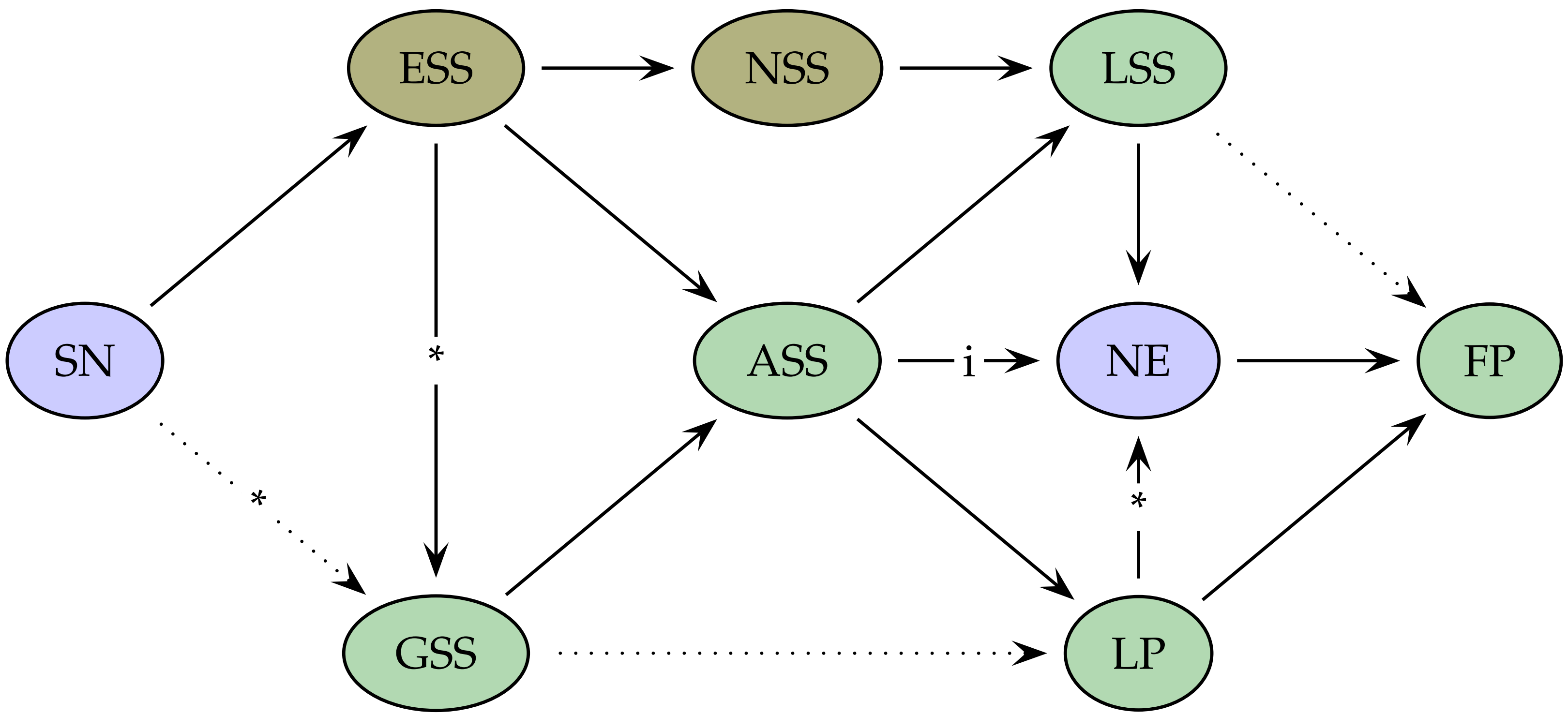
With tournament selection.



With modified tournament selection.[a]

[a] Modified tournament selection is a hybrid of fitness-proportionat selection and 2-sample tournament selection. Cf. Sec. 2.7. of Airiau *et al.*' 2007 paper.

Implications (green concerns the replicator dynamic)



SN = strict Nash, ESS - evol'y stable strategy, GSS = glob'y stable state, ASS = asymp'y stable state, NSS = neutrally stable strategy, LP = limit point, LSS = Lyapunov stable state, NE = Nash eq., FP = fixed point, * = only if fully mixed, i = isolated Nash eq. Dotted lines are indirect implications.

Evaluation by computing NE of grand table

Evaluation by computing NE of grand table

Reconsider the grand table:

	A_1	A_2	$\ldots$	A_{12}	avg
A_1	3.1	5.1	$\ldots$	4.7	4.1
A_2	2.4	1.2	$\ldots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\ldots$	3.8	4.2

Evaluation by computing NE of grand table

Reconsider the grand table:

	A_1	A_2	$\dots$	A_{12}	avg
A_1	3.1	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

See this as a game in normal form:

$$
\begin{array}{c c}
 & \begin{array}{cccc} A_1 & A_2 & \dots & A_{12} \end{array} \\
\begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_{12} \end{array} &
\begin{pmatrix}
3.1 & 5.1 & \dots & 4.7 \\
2.4 & 1.2 & \dots & 2.2 \\
\vdots & \vdots & \ddots & \vdots \\
3.1 & 6.1 & \dots & 3.8
\end{pmatrix}
\end{array}
$$

Evaluation by computing NE of grand table

Reconsider the grand table:

	A_1	A_2	$\dots$	A_{12}	avg
A_1	3.1	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

See this as a game in normal form:

$$\begin{array}{c c} & \begin{array}{cccc} A_1 & A_2 & \dots & A_{12} \end{array} \\ \begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_{12} \end{array} & \begin{pmatrix} 3.1 & 5.1 & \dots & 4.7 \\ 2.4 & 1.2 & \dots & 2.2 \\ \vdots & \vdots & \ddots & \vdots \\ 3.1 & 6.1 & \dots & 3.8 \end{pmatrix} \end{array}$$

Compute all Nash Equilibria.

Facts. (1) Every fully mixed limit point of the replicator dynamic is a Nash equilibrium. (2) Every Nash-equilibrium is a fixed point of the replicator dynamic.

Evaluation by computing NE of grand table

Reconsider the grand table:

	A_1	A_2	$\dots$	A_{12}	avg
A_1	3.1	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

See this as a game in normal form:

$$\begin{array}{c c} & \begin{array}{cccc} A_1 & A_2 & \dots & A_{12} \end{array} \\ \begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_{12} \end{array} & \begin{pmatrix} 3.1 & 5.1 & \dots & 4.7 \\ 2.4 & 1.2 & \dots & 2.2 \\ \vdots & \vdots & \ddots & \vdots \\ 3.1 & 6.1 & \dots & 3.8 \end{pmatrix} \end{array}$$

Compute all Nash Equilibria.

Facts. (1) Every fully mixed limit point of the replicator dynamic is a Nash equilibrium. (2) Every Nash-equilibrium is a fixed point of the replicator dynamic.

The converse of (1) and (2) are not true (absent species violate the contra-implication), but (1) and (2) surely help to isolate interesting "response profiles".

Evaluation by computing NE of grand table

Reconsider the grand table:

	A_1	A_2	$\dots$	A_{12}	avg
A_1	3.1	5.1	$\dots$	4.7	4.1
A_2	2.4	1.2	$\dots$	2.2	1.3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_{12}	3.1	6.1	$\dots$	3.8	4.2

See this as a game in normal form:

$$\begin{array}{c c} & \begin{array}{cccc} A_1 & A_2 & \dots & A_{12} \end{array} \\ \begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_{12} \end{array} & \begin{pmatrix} 3.1 & 5.1 & \dots & 4.7 \\ 2.4 & 1.2 & \dots & 2.2 \\ \vdots & \vdots & \ddots & \vdots \\ 3.1 & 6.1 & \dots & 3.8 \end{pmatrix} \end{array}$$

Compute all Nash Equilibria.

Facts. (1) Every fully mixed limit point of the replicator dynamic is a Nash equilibrium. (2) Every Nash-equilibrium is a fixed point of the replicator dynamic.

The converse of (1) and (2) are not true (absent species violate the contra-implication), but (1) and (2) surely help to isolate interesting "response profiles".

It is interesting to interpret Nash equilibria among reply rules on the grand table.

General evaluation

General evaluation

General evaluation

1. **Selection of learning algorithms to test on**.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.
2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

   We should have different leagues: coupled (disposal of Nash equilibria), uncoupled, completely uncoupled.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

   We should have different leagues: coupled (disposal of Nash equilibria), uncoupled, completely uncoupled.

3. **Parametrisation of algorithms**. The parameter search space is often in the order of $[0, 1]^k$, $k \geq 1$ per algorithm (k the number of parameters), which gives uncountably many sub-algorithms.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.
2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

   We should have different leagues: coupled (disposal of Nash equilibria), uncoupled, completely uncoupled.
3. **Parametrisation of algorithms**. The parameter search space is often in the order of $[0,1]^k$, $k \geq 1$ per algorithm (k the number of parameters), which gives uncountably many sub-algorithms.
4. **Selection of games**.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.
2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

   We should have different leagues: coupled (disposal of Nash equilibria), uncoupled, completely uncoupled.
3. **Parametrisation of algorithms**. The parameter search space is often in the order of $[0,1]^k$, $k \geq 1$ per algorithm (k the number of parameters), which gives uncountably many sub-algorithms.
4. **Selection of games**. Approaches demonstrate poor method.

General evaluation

1. **Selection of learning algorithms to test on**. All approaches demonstrate arbitrariness and lack of method.

2. **Conditions on algorithms**: coupled, uncoupled, completely uncoupled. All methods give contestants full information $\Rightarrow$ unequal playing field. Further, accessibility to computationally expensive game properties (mixed Nash equilibria) is taken for granted.

   We should have different leagues: coupled (disposal of Nash equilibria), uncoupled, completely uncoupled.

3. **Parametrisation of algorithms**. The parameter search space is often in the order of $[0,1]^k$, $k \geq 1$ per algorithm (k the number of parameters), which gives uncountably many sub-algorithms.

4. **Selection of games**. Approaches demonstrate poor method. Justification of choices is weak and subjective.

General evaluation (continued)

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form.**

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**.
   Why not > 2 players?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.
6. **Selection of "sub-contestants" yields other rankings**.

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.
6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C.

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C.

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

7. **Evolutionary selection**:

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

7. **Evolutionary selection**: How to select algorithms? Fitness-proportional selection, tournament selection, reward-based selection, stochastic universal sampling, replicator dynamic?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

7. **Evolutionary selection**: How to select algorithms? Fitness-proportional selection, tournament selection, reward-based selection, stochastic universal sampling, replicator dynamic?

8. **Knock-out tournament**:

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of “sub-contestants” yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

7. **Evolutionary selection**: How to select algorithms? Fitness-proportional selection, tournament selection, reward-based selection, stochastic universal sampling, replicator dynamic?

8. **Knock-out tournament**: How to decide when and where to drop the worst performer?

General evaluation (continued)

5. **All approaches evaluate learning 2-player games in normal form**. Why not > 2 players? Why not games in extensive form? Too much work? Then say so.

6. **Selection of "sub-contestants" yields other rankings**. How to pit? Suppose algorithm A performs best in the presence of algorithms B and C, thanks to C. But A perform worse than B in the absence of C. How to deal systematically with that?

7. **Evolutionary selection**: How to select algorithms? Fitness-proportional selection, tournament selection, reward-based selection, stochastic universal sampling, replicator dynamic?

8. **Knock-out tournament**: How to decide when and where to drop the worst performer? The order of elimination puts a (plausible but arbitrary) bias on the ranking.

Now you know enough . . .

Now you know enough . . .
to design your own MAL algorithm . . .

Now you know enough . . .
to design your own MAL algorithm . . .
and MAL comparison methods . . .

The end

Good luck and *au revoir* ...