Algorithm Diversity - A Mechanism for Distributive Justice in a Socio-Technical MAS

Vivek Nallur, Eamonn O'Toole, Nicolás Cardozo, Siobhán Clarke
Future Cities, Trinity College Dublin
Dublin, Ireland
{vivek.nallur|otooleea|cardozon|siobhan.clarke}@scss.tcd.ie

ABSTRACT

Socio-technical MAS are an intrinsic part of our daily lives. Domains like energy, transport, etc. are increasingly using technology to allow individual users to adapt to, and even influence the aggregate performance of the system. This raises expectations of fairness and equitability, while being engaged in such MAS. Given the autonomous and decentralized nature of socio-technical MAS, it can be difficult to ensure that each agent gets a fair reward for participating in the system. We introduce the notion of algorithmic diversity as a mechanism for nudging the system, in a decentralized manner, to a more equitable state. We use the minority game as an exemplar of a transportation network in a city, and show how diversity of algorithms results in a fairer reward distribution than any individual algorithm alone.

General Terms

Algorithms; Design

Keywords

Algorithm Diversity; Methodologies for agent-based systems; Engineering Multi-Agent Systems

1. INTRODUCTION

The modern world is filled with self-adaptive socio-technical multi-agent systems (socio-technical MAS). From homes in a Smart Grid environment to commuters using the latest traffic and route updates on their smartphones, the number of multi-agent systems where human beings explicitly utilize device/machine intelligence to make decisions or participate in an aggregate common-pooling of resources, is increasing. In this context, there is a legitimate expectation that such socio-technical MAS ensure an equitable allocation of resources, or reward the participating agents using a fair mechanism. However, by design, socio-technical MAS are decentralized and autonomous in nature. That is, there is no centralized authority that can control the actions of the individual agents, except through indirect mechanisms such as incentives and penalties. This implies that any aggregate property that the MAS exhibits, such as efficiency / fairness (or the lack of it) must result from the actions of the individual agents themselves. The field of computational social choice is particularly interested in the design and analysis of methods that result in choices being made through collective action [9]. Computational social choice is concerned with multiple types of social choices, such as preference aggregation, voting, fair division of resources, coalition formation, judgement aggregation, ranking systems, etc. In this paper, we are concerned with only one aspect of social choice, *viz*, fairness in resource allocation. Typical multi-agent resource allocation strategies concern themselves with *Pareto Efficiency*, which refers to an allocation such that no other allocation would increase the utility of some agents without being worse for the others. Fairness conditions are typically codified as *Envy-Freeness*, where no agent would rather obtain the resource bundle held by another. Human beings, even though aiding or aided by computational devices/agents, perceive equity to be more than simple utility of a resource bundle [37]. Hence, *envy-freeness does not suffice* for a socio-technical MAS. A more nuanced notion of fairness is introduced by the idea of distributive justice, based on the principles of legitimate claims, as elucidated by Rescher [35]. As an example, a route-choice problem through a city (*i.e.*, pick road A or road B to reach a destination) must yield a satisfactory outcome for the agent making a particular choice, *at least some of the time*. This is in contrast to problem formulations, where the only concern is to optimize the traffic flow through that route. This is a familiar situation in many cities, where multiple humans, utilizing commonly known (albeit coarse-grained) information about traffic flows, along with their own private heuristics (*e.g.*, road A is generally clogged from 7:30 - 8:30 am), make route choices to reach their destination. These agents may use additional device/machine intelligence in the form of real-time traffic updates through GPS devices/smartphones, while making their choices. The reward for this choice is determined dynamically, based on the aggregate number of agents that make a particular choice without prior coordination, *i.e.*, the driver that chooses the 'road less travelled' would experience lower congestion. Distributive justice, *in this case*, would be the evenness of the rewards that accumulate to agents, over multiple days. We **hypothesize** that *introducing algorithm diversity can lead to distributive justice, without centralized oversight of the game or the agents.* That is, the presence of algorithm diversity will contribute to a higher degree of evenness of accumulated reward.

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

Through the rest of this paper, we first describe what we mean by algorithm diversity (section 2), explain the MAS that we use for experimentation (section 3) and the experimental setup (section 4). Finally, we look at the efficacy of algorithm diversity in ensuring evenness (section 5) and offer some concluding thoughts.

2. ALGORITHM DIVERSITY

By algorithm diversity, we refer to the simultaneous use of multiple algorithms (having the same functional goal), or in the *same running* instance of the system. Algorithm diversity exploits the natural diversity of algorithms that are present in the same domain. For instance, there are a multiplicity of sorting algorithms, compression algorithms, load-balancing algorithms and typically, the informed programmer makes an intelligent choice amongst the various available algorithms based on the particular characteristics of the operating domain. However, in situations where there is a high degree of uncertainty, it has been shown that diversity can lead to a better performance on many parameters [16, 22, 30, 34, 40]. In Ecology, this is referred to as the *Insurance Hypothesis* [42], where a species copes with a changing environment, by exhibiting a diversity of features that may / may not be useful all the time.

However, in all these works of diversification of software systems, there has been no work that actively *measures* the amount of diversity present in a system, and relates this to the performance of the system. This paper introduces a quantification of diversity present in the system, and relates it to system performance, specifically evenness of rewards.

From our definition of algorithmic diversity, it is easy to envisage a system that diversifies its output, based on environmental cues, by switching between multiple algorithms. However, a more interesting case is when for the same input environment, different parts of the system use different algorithms. A MAS is an example of such a system, where multiple agents can potentially use different algorithms to achieve the same goals, for *exactly* the same input environment. This allows us to clearly identify the gains made due to diversification, as opposed to some other modification or environmental change. In this paper, we use the *Minority Game* as an exemplar MAS for algorithm diversification. In doing so, we deviate from the traditional research on the Minority Game, which is focussed on analysis of system dynamics due to a newly introduced strategy.

Measuring Algorithm Diversity.

A diversity index is a quantitative measure of the number of different types and the abundance of those types, in a given dataset. Intuitively, the higher the diversity index, the higher the diversity of the dataset. There are no mechanisms for differentiating amongst algorithms, that can be commonly applied across multiple domains, apart from the *Big-O* complexity calculation. The *Big-O* notation does not, however, yield useful information about the *differentness* of algorithms. There may be two algorithms that have the same *Big-O class*, but are completely different in terms of data-structures. For instance, Quicksort and Heapsort are very different algorithms, but their average case time complexity is the same, $O(nlog(n))$! It also does not help us differentiate between two algorithms that use the same data-structures but return *different outputs*. Our interest in algorithm diversity is due to the potential differences in the functional pathways, and consequently the outputs, for the same given input-sequence. This is so because it is this diversity of outputs that we wish to exploit, as a strategy to deal with dynamic environments. In such cases, the Big-O calculation does not yield any useful insights. Hence, we look to Ecology for inspiration, where the notion of diversity has been discussed extensively [41]. There are many measures of species diversity that are used in ecological literature. One of the most famous ones is the *Shannon Index* (eq. 1). The Shannon Index measures the entropy (or uncertainty) in picking an individual of a particular type, from a dataset. Thus, for instance, if all the agents in a MAS implement exactly the same algorithm, the entropy (or uncertainty) will be zero. The higher the diversity in the population, the higher is the Shannon Index. In eq. 1, R is the number of types in the population, and p is the proportion of individuals in the population with type i.

$$H' = -\sum_{i=1}^{R} p_i \ln p_i \tag{1}$$

The Shannon Index is a measure of both, the *richness of species* (number of different algorithms) and *evenness of species* (abundance of agents implementing an algorithm). We use the Shannon Index[1] to measure how much algorithms differ, in their key parameters. Thus, if a particular initialization parameter (that has an effect on the functional pathway) can take multiple values, the Shannon Index will measure how many such differentiated types exist and how many individuals implement that type, as a proportion of the total population.

3. MINORITY GAME

We use the Minority Game [8] as our exemplar MAS, primarily because it has been well-studied and well-understood. The Minority Game (MG), introduced by Challet and Zhang, consists of an odd number (N) of agents, playing a game in an iterated fashion. At each timestep, each agent chooses from one of two actions (+1 or -1), and the group that is in the minority, wins. Since N is an odd integer, there is guaranteed to be a minority. Winners receive a reward, while losers receive nothing. After each round, all agents are told which action was in the minority. This feedback loop induces the agents to re-evaluate their action for the next iteration. While simple in its conception, this game has been used in many fields like econophysics [5, 7, 43], multi-agent resource allocation [25, 27], emergence of cooperation [13], and heterogeneity [21, 26]. The Minority Game (MG) is intuitively applicable to many domains, such as traffic (the driver that chooses the 'less-travelled' road experiences less congestion), packet traffic in networks, ecologies of foraging animals, etc. The Minority Game has been used, primarily, to investigate the *efficiency* of strategies in the system when individuals have bounded rationality (for an extensive review, see [29]).

3.1 Best Play Strategy

Challet and Zhang also introduced a very simple strategy, called *Best Play* strategy [8], which in spite of being simple, exhibited interesting dynamics at the aggregate level. In this strategy, all players had access to a bounded collective

[1] We also have measurements using the Gini-Simpson Index, but we omit them here due to lack of space

memory (m) of the game, along with a pool of strategies(s). Collective memory is implemented in the form of a history of which action (+1 or -1) was in the minority, at each timestep. Each player drew s strategies randomly from a strategy space, that indicated which action to play next, based on the history of the game. Each player allocated virtual points amongst all its strategies, based on the continually changing history of the game. Strategies that correctly predicted the correct output, regardless of whether they were used or not, received virtual points. Each player, for the next round, used the top-ranking strategy amongst his pool of strategies. Table 1 shows a sample strategy pool used by a player. The size of each strategy depends on m, the memory of the game, while the total number of strategies in the pool depends on the computational power the agent possesses. Table 1 is therefore taken from a game with a memory of size 3, while the agent has a pool size (k) of 4. Typically, each agent in the entire game has the same pool size (k). While it is intuitively obvious, that the greater the pool size, the greater the probability of an agent drawing good strategies, it is impossible for an agent to have a pool size that exhaustively covers every possible strategy, since the strategy space grows hyper-exponentially(2^{2^m}) with increase in m.

Strategy	Output
[+1 -1 +1]	-1
[-1 -1 -1]	-1
[-1 -1 +1]	+1
[+1 +1 +1]	-1

Table 1: Sample Strategy Pool for a Player

3.2 Reinforcement Learning Strategy

We implement a Reinforcement-Learning algorithm, called Roth-Erev [36]. Reinforcement-Learning(RL) is a class of machine-learning algorithms where an agent tries to learn the optimal action to take, in a certain environment, based on a succession of *action-reward* pairs. Given a set of actions (a) that an agent can take, the Roth-Erev learning algorithm performs the following steps:

1. Initialize initial propensities (q) amongst all actions (N)
2. Initialize strength (s) of initial propensities
3. Initialize recency (φ) as the 'forgetting' parameter
4. Initialize epsilon (ε) as the exploration parameter
5. Choose action (a_k) at time (t), based on q
6. Based on reward(r_k) for a_k, update q using the following formula:

$$q_j(t+1) = [1-\varphi]q_j(t) + E_j(\varepsilon, N, k, t)$$

$$E_j(\varepsilon, N, k, t) = \begin{cases} r_k(t)[1-\varepsilon] & \text{if } j = k \\ r_k(t)\frac{\varepsilon}{N-1} & \text{otherwise} \end{cases}$$

7. Probability of choosing action j at time (t) is given by:

$$P_j(t) = \frac{q_j(t)}{\sum_{n=1}^{N}[q_n(t)]}$$

8. Repeat from step 5

3.3 Evolutionary Strategy

Evolutionary algorithms have been used quite successfully in multiple domains [1,11]. As pointed out in [14], evolutionary algorithms are now being compared to human beings, in their ability to solve hard problems. We have implemented an evolutionary algorithm that uses *BestPlay*'s strategy representation as a genome and performs crossover and mutation on it. The strategy is as follows: At every reproduction cycle, the ten most poorly performing agents get rid of their strategies and adopt a combination of strategies from two parents randomly selected from the ten best performing agents. Effectively, two of the ten best performing agents have reproduced to create an offspring that replaces one of the ten worst agents. The offspring mutates the strategies obtained from its parents, by some small mutation probability. This allows the offspring to get better, as well as explore more of the strategy space. The reproduction cycle refers to the rounds after which evolution occurs. A frequency of 20 cycles means that after every 20 rounds of the game, the worst 10 agents discard their strategies and inherit good strategies from 2 of the top 10 agents. These are then mutated according to the agents' own mutation probability.

4. EXPERIMENTAL SETUP

Table 2 shows the constant factors in each experiment. Each minority game was played with a population size of 501 agents, through a simulation time period of 2000 steps. For each variation in the experimental setup, the data is reported as an average of 100 simulations.

Variable	Value
Population Size	501
Simulation Period (rounds)	2000
Repetition of Variation	100

Table 2: Experimental constants

The parameters that were varied were as follows:

1. **Number of Algorithms**: This reflects the effective number of families (or *genera*) of species that are present in the population.
2. **Proportion of population implementing an algorithm**: Depending on the relative number of each specie, the aggregate payoff in the game changes.
3. **Parameters within algorithms that were diversified**: Variation in critical parameters for an algorithm results in effectively creating a new specie, since the functional pathway (and resultant output) begins to change depending on the environment. For each of the three families, the following parameters were diversified within the algorithm:
   (a) Best Play Strategy: k – which is the pool of strategies, a player has(strategies-per-agent).
   (b) Evolutionary Strategy: The mutation probability that allows an agent to explore the strategy space, by mutating known good strategies.
   (c) Reinforcement Learning Strategy: The parameters of *recency* (φ), and *exploration* (ε)

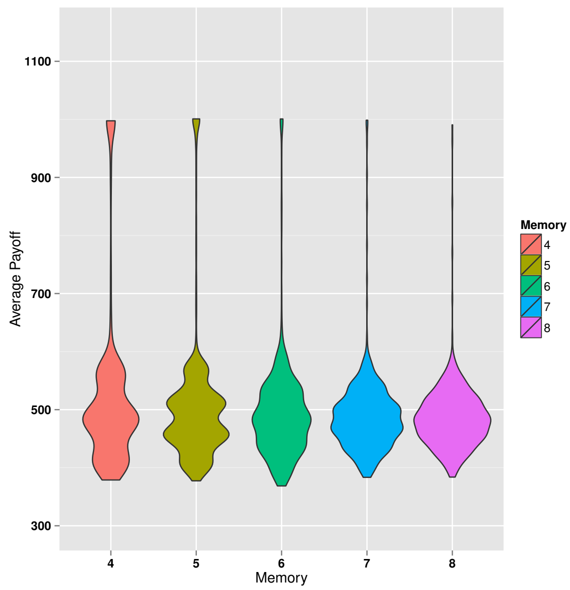

Figure 1: Best-Play strategy with different values of memory

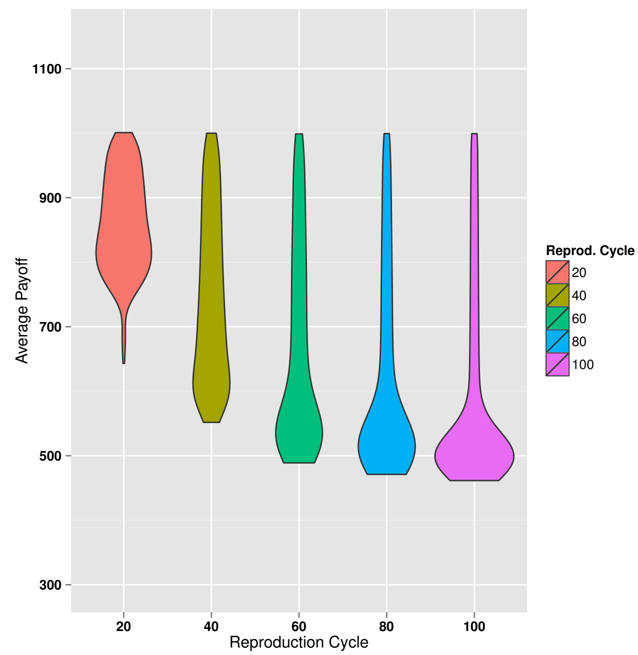

Figure 2: Evolutionary strategy

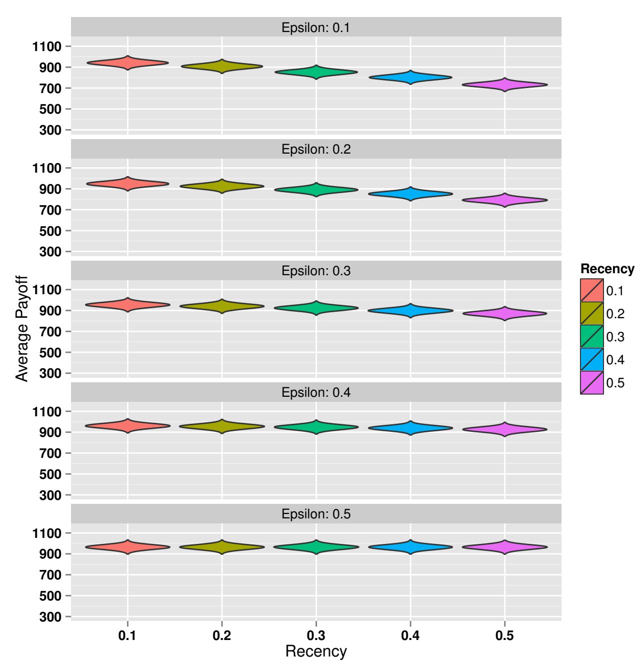

Figure 3: Roth-Erev strategy

5. RESULTS

We now show the results of diversification of algorithms, on two levels: *(a)* Parameter Diversification, and *(b)* Strategy Diversification[2]. *Parameter Diversification* refers to each agent in the population implementing the same algorithm, however they change their initialization parameters. For instance, agents playing the Evolutionary strategy would draw their mutation probabilites from a gaussian distribution, drawn from the best mean of *Normal Play*. Normal Play refers to a situation, where each agent in the game plays with `exactly` the same algorithm and initialization parameters. *Strategy Diversification* refers to creating a population that has a mix of strategies. The diversity level of the population would differ according to the proportion of agents implementing a particular strategy. First, we show results of all three strategies with Normal Play, since they form the baseline with which the results of diversification are compared. In all cases, the graphs show the distribution of the population achieving a particular level of average payoff. The 'fatter' the graph, the more evenly rewards are distributed, and vice-versa.

5.1 Normal Play — Baseline

In Figure 1, we see the average payoff (y-axis) from the Best-Play strategy, with different levels of memory(x-axis). The average payoff is the average number of times, an agent is able to make the correct choice and be in the minority. The graph shows the distribution of population that was able to secure rewards, during the entire game. The *fatter* the distribution, the fairer the result, since it implies that, on average, most players secured the same amount of payoff. In Figure 1, we see that with a memory of 8, the distribution of payoff is more equitable, since most of the population is concentrated in one area.

[2]All code and results are available at: `https://www.bitbucket.org/viveknallur/aamas2016.git`

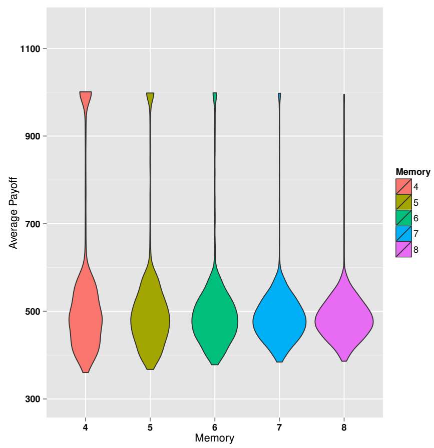

Figure 4: Parameter Diversification of Best Play

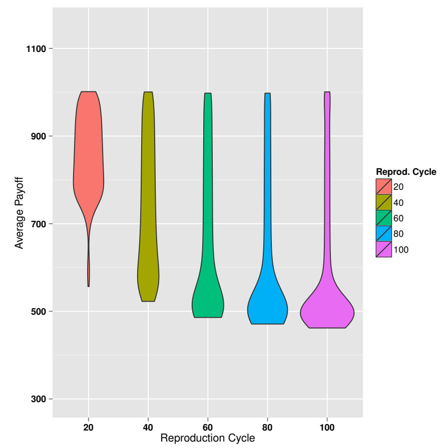

Figure 5: Parameter Diversification of Evolutionary strategy

In Figure 2, we see the average payoff from the Evolutionary strategy. The x-axis shows the frequency of the reproduction cycle, *i.e.*, reproduction and mutation takes place every $20, 40, 60, 80$ and 100 rounds. Figure 2 clearly shows that when reproduction and mutation is frequent (every 20 rounds), more of the population earns a higher payoff. That is, most of the population earns a payoff between 700 and 1000, whereas with reproduction cycles of 60 and greater, most of the population earns a lower reward, with a few individuals earning higher payoff.

Figure 3 shows the average payoff when the Roth-Erev strategy is used. Again, the fatter the distribution, the fairer the outcome. Hence, we see that low values of *Recency* $(0.1, 0.2)$ contribute to a fairer distribution of payoff.

In all of these, the diversity of the population is *zero* (0), since there is only one algorithm, and all agents implement exactly the same code.

5.2 Parameter Diversity

Parameter diversity refers to diversity in the initialization parameters of the algorithm being implemented by the agents. That is, for any game, all agents still implement the same code, but some parameter in the algorithm is diversified. The specific parameters being diversified are given in section 4. For the Evolutionary and Roth-Erev strategies, even minute differences within the initialization parameters can cause aggregate average payoff to vary.

Parameter Diversity with Best Play.

In general, there is less parameter diversity with the Best Play strategy, since the parameter being diversified is the number of strategies that player holds. These are drawn from a gaussian distribution with the mean being the size of the history available to agents, and a standard deviation of 1. This leads to there being a low diversity, since many agents end up with the same amount of strategies-per-agent. Figure 4 therefore shows very little difference in distribution of payoffs obtained, as compared to Figure 1.

Parameter Diversity with Evolutionary Strategy.

The evolutionary strategy changes the performance of the agents, depending on the strategies that the ‘evolved’ child agents get. The evolutionary process allows poorly performing agents to quickly change their strategies to match the best agents. However, this also leads to a homogeneity in the strategies being used by the agents. The parameter that diversifies each agent is the mutation probability that it possesses. This causes agents to mutate their inherited strategies. Figure 5 shows the difference in average payoff, when the reproduction cycle is varied.

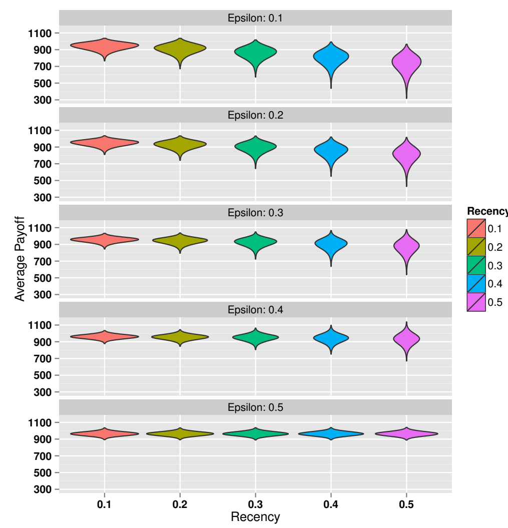

Figure 6: Parameter Diversification of Roth-Erev strategy

Parameter Diversity with Roth-Erev.

Roth-Erev allows for the highest amount of diversification, since there are two parameters (recency and epsilon) that can be changed for each agent. Also, the algorithm is extremely sensitive to the values of these parameters, and each agent’s decision-making is affected significantly. Hence, the population as a whole becomes extremely diverse. Figure 6 shows the effect of diversification for mean values of recency and epsilon.

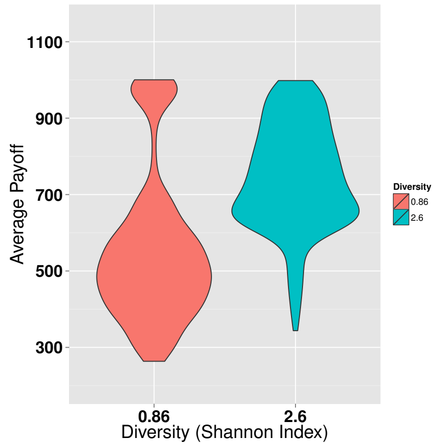

Figure 7: Mix of Best-Play and Evolutionary Strategies

Both Roth-Erev and the Evolutionary strategy exhibit higher levels of diversity, due to the probabilistic nature of the variable being diversified. In the Evolutionary strategy, the mutation-probability is different for every agent, thereby making each agent into its own separate type. In Roth-Erev, *both* recency and epsilon are diversified, which results in a higher level of diversity. This increase in diversity is reflected in both, the fairness of the average payoff, and the levels of average payoff. That is, for Roth-Erev populations, most of the payoff is at the 800 level, while for the Evolutionary strategy, most of the payoff is around 500.

5.3 Strategy Diversity

We go further and mix agents implementing different strategies into one population. The diversification here takes place on two levels, parameter-diversification (as before), as well as proportion of population implementing a particular strategy. The total population of agents in all scenarios is kept constant, but the proportion of agents playing a particular strategy is varied. This leads to different levels of diversification, for each combination of strategies. Due to lack of space, we show only the population mixes with the *lowest* and *highest* amount of diversity, for each combination of strategies. The x-axis shows the diversity levels (as measured by the Shannon Index) and the y-axis, the distribution of average payoff.

Figure 7 shows the distribution of payoffs when a set of agents playing Best Play, is mixed with a set of agents playing the Evolutionary strategy. Figure 8 shows the distribution of payoffs when a set of agents playing Best Play, is mixed with a set of agents playing the Roth-Erev strategy. Figure 9 shows the distribution of payoffs when a set of agents playing Evolutionary strategy, is mixed with a set of agents playing the Roth-Erev strategy. Figure 10 shows the distribution of payoffs, when all three strategies are combined into one population.

All compared together.

Finally, we overlay the results from Figures 7–10 to illustrate the comparative payoffs at different levels of diversity. Figure 11 clearly shows that higher the diversity level, the fairer the average payoff. Note that the highest level (5.13) was **not** a consequence of mixing all three strategies, but rather by mixing Evolutionary and Roth-Erev strategies (see

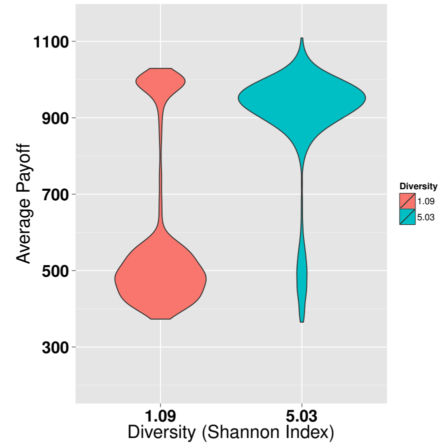

Figure 8: Mix of Best-Play and Roth-Erev Strategies

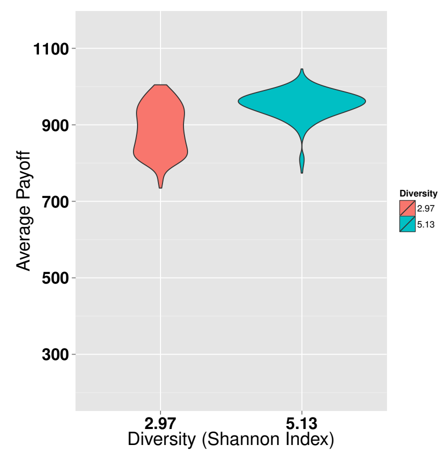

Figure 9: Mix of Evolutionary and Roth-Erev Strategies

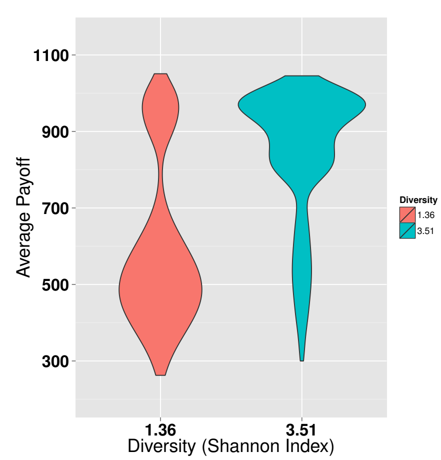

Figure 10: Mix of three strategies

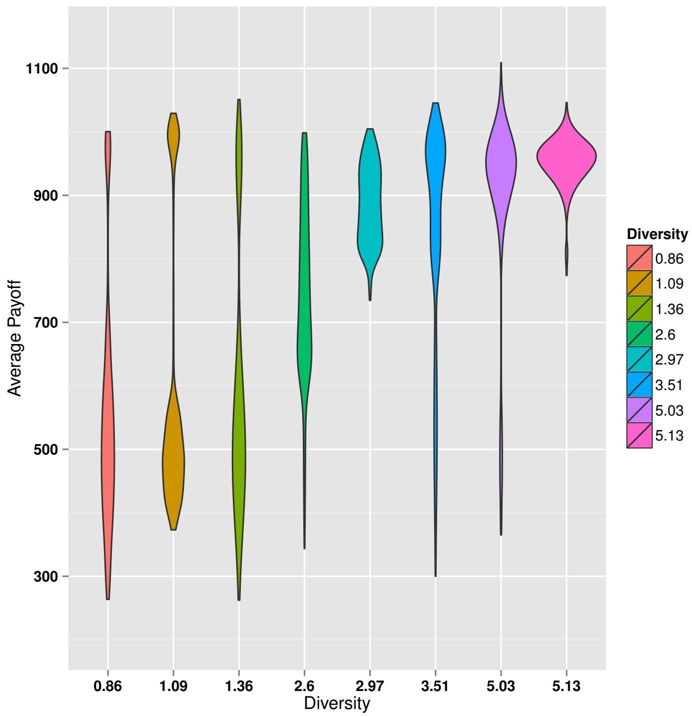

Figure 11: Payoff across all mixes

Fig 9). This confirms our intuition that merely mixing more strategies does not lead to greater diversity and measurement using indices is essential.

Table 3 shows the payoff at the lowest, median and highest agent reward levels, due to the different algorithms used. Notice that the median levels of payoff are the greatest when diversity is greatest ($d = 5.13$). Recall from our hypothesis, that we seek to check whether algorithm diversity leads to distributive justice, without centralized oversight. From Table 3 and the graphs (Figure 11), it is apparent that the hypothesis is true. The *effect of diversity is to push the population, as a whole, towards a more even distribution of rewards, without needing a centralized guiding hand*.

6. RELATED WORK

There has been much work on distributive justice in a sociological and organizational context [10, 12, 24], but most work in multi-agent systems is characterized by *Envy Freeness* [18]. Envy-freeness is simply understood to be an allocation of a resource bundle in such a manner, that for any agent, the utility of its own allocation is greater than any other allocation. However, utility-based models fail to capture the nuances of distributive justice, particularly so in a socio-technical MAS. There has been some work on making multi-agent systems adapt in an organized fashion, obeying certain norms [3, 17, 20, 32, 33]. There has also been a plethora of work on the introduction of diversity in software engineering. *Forrest et al.* [19] were among the first to advocate diversity as a mechanism to prevent attackers from learning internal details of a computer system. Memory obfuscation [4] and instruction-set randomization [23] have been used as an instantiation of the diversification principle, to increase security of computer systems. Efforts have been made at introducing diversity into the source code of systems, in order to reduce the chances of bugs through N-version programming [2], and genetic programming [16]. Other work in security such as creation of moving targets [22], increasing dependability [30] have all advocated diversification as a fundamental idea. Even in other domains, such as machine-learning [28], sensor-networks [34, 38], and fault-tolerant systems [39] diversity has been identified as an idea that improves the robustness of systems, in the presence of uncertainty. The domain of distributed and multi-agent systems [6, 15] have also considered diversity as a critical component in systems with varying degrees of intelligence. However, this is the first work that we are aware of, that **quantifies** the amount of diversity present in a system, and then systematically compares level of diversity with aggregate performance. A closely related agent-based domain, electronic trading markets, have been studied in [31] with an emphasis on auction mechanism design. The focus of that study, however, has been on economic outcomes and game-theoretic methods, without any investigation into diversity as a distinguishing attribute.

7. CONCLUSION

A caveat about measuring algorithm diversity in this manner, is that it requires knowledge of the algorithms and their functional pathways. While this is not ideal from a black-box engineering point of view, nevertheless, we believe that it holds great potential as a technique for achieving certain aggregate properties in a socio-technical MAS. Another potential drawback is the requirement for problem decomposition into individuals and species. This may not be possible in all domains. Algorithm diversity is not a silver bullet for ensuring distributive justice of rewards in a socio-technical MAS. In further work, we would like to work on quantifying diversity in algorithms, even if the problem is not as easily decomposed into agents, such as the minority game. We believe that there are many domains that would benefit from a rigorous mechanism for quantifying diversity in algorithms, and this would allow them to make reasoned tradeoff decisions between diversity, and other system metrics such as performance.

Acknowledgments

This work was partially supported by the EU FET-Project Diversify FP7-ICT-2011-9

REFERENCES

[1] D. Ashlock. *Evolutionary Computation for Modeling and Optimization*. Springer-Verlag New York, 2006.

[2] A. Avizienis. The methodology of n-version programming. *Software fault tolerance*, 3:23–46, 1995.

[3] T. Balke, M. De Vos, and J. Padget. I-abm: combining institutional frameworks and agent-based modelling for the design of enforcement policies. *Artificial Intelligence and Law*, 21(4):371–398, 2013.

[4] S. Bhatkar, D. C. DuVarney, and R. Sekar. Address obfuscation: An efficient approach to combat a board range of memory error exploits. In *12th Conference on USENIX Security Symposium*, 12:105–120, Berkeley, CA, USA, 2003.

Strategy	Lowest Payoff	Median Payoff	Highest Payoff
Pure			
BestPlay	368	483	1001
Evo	460	680	1003
Roth-Erev	666	929	1035
Combination			
Evo-BestPlay (d = 0.86)	263	509	1001
Evo-BestPlay (d = 2.60)	340	733	1002
RothErev-BestPlay (d = 1.09)	371	499	1030
RothErev-BestPlay (d = 5.03)	362	939	1112
Evo-RothErev (d = 2.97)	734	883	1004
Evo-RothErev (d = 5.13)	**774**	**957**	**1046**
BestPlay-Evo-RothErev (d = 1.36)	260	526	1053
BestPlay-Evo-RothErev (d = 3.51)	297	905	1048

Table 3: Various levels of diversity compared to 'pure play' of single strategies

[5] G. Bianconi, T. Galla, M. Marsili, and P. Pin. Effects of Tobin taxes in minority game markets. *Journal of Economic Behavior & Organization*, 70(1-2):231–240, May 2009.

[6] A. Campbell, C. Riggs, and A. Wu. On the impact of variation on self-organizing systems. In *5th Intl. Conf on Self-Adaptive and Self-Organizing Systems*, pages 119–128, Oct 2011.

[7] D. Challet. Inter-pattern speculation: Beyond minority, majority and $-games. *Journal of Economic Dynamics and Control*, 32(1):85–100, Jan. 2008.

[8] D. Challet and Y. Zhang. Emergence of cooperation and organization in an evolutionary game. *Physica A: Statistical Mechanics and its Applications*, 246(34):407 – 418, 1997.

[9] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. volume 4362 of *Lecture Notes in Computer Science*, pages 51–69. Springer Berlin Heidelberg, 2007.

[10] Y. Cohen-Charash and P. E. Spector. The role of justice in organizations: A meta-analysis. *Organizational Behavior and Human Decision Processes*, 86(2):278 – 321, 2001.

[11] K. De Jong. *Evolutionary Computation: A Unified Approach*. MIT Press, 2006.

[12] M. Deutsch. Equity, equality, and need: What determines which value will be used as the basis of distributive justice? *Journal of Social Issues*, 31(3):137–149, 1975.

[13] D. Dhar, V. Sasidevan, and B. K. Chakrabarti. Emergent cooperation amongst competing agents in minority games. *Physica A: Statistical Mechanics and its Applications*, 390(20):3477–3485, Oct. 2011.

[14] A. E. Eiben and J. Smith. From evolutionary computation to the evolution of things. *Nature*, 521:476–482, 2015.

[15] L. Esterle, P. R. Lewis, X. Yao, and B. Rinner. Socio-economic vision graph generation and handover in distributed smart camera networks. *ACM Transactions on Sensor Networks*, 10(2), 2014.

[16] R. Feldt. Generating diverse software versions with genetic programming: an experimental study. *IEE Proceedings-Software*, 145(6):228–236, 1998.

[17] D. Fitoussi and M. Tennenholtz. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119(1–2):61–101, 2000.

[18] D. Foley. Resource allocation an the public sector. *Yale economic essays*, 7(1):45–98, 1967.

[19] S. Forrest, A. Somayaji, and D. Ackley. Building diverse computer systems. In *6th Workshop on Hot Topics in Operating Systems*. IEEE Comp Soc, 1997.

[20] T. F. Gordon. *The Pleadings Game: An Artificial Intelligence Model of Procedural Justice*. Springer Netherlands, 1995.

[21] G. W. Greenwood. Deceptive strategies for the evolutionary minority game. In *5th Intl Conf on Computational Intelligence and Games*, pages 25–31, 2009.

[22] Y. Huang and A. K. Ghosh. Introducing diversity and uncertainty to create moving attack surfaces for web services. volume 54, pages 131–151. Springer, 2011.

[23] G. S. Kc, A. D. Keromytis, and V. Prevelakis. Countering code-injection attacks with instruction-set randomization. In *10th ACM conference on Computer and communications security*, pages 272–280. ACM, 2003.

[24] G. Leventhal, J. Karuza, and W. R. Fry. *Beyond fairness: A theory of allocation preferences*. Justice and Social Interaction. Springer, 1980.

[25] Y. Li and R. Savit. Toward a theory of local resource competition: the minority game with private information. *Physica A: Statistical Mechanics and its Applications*, 335(1-2):217–239, Apr. 2004.

[26] B. A. Mello and D. O. Cajueiro. Minority games, diversity, cooperativity and the concept of intelligence. *Physica A: Statistical Mechanics and its Applications*, 387(2-3):557–566, Jan. 2008.

[27] R. Metzler and C. Horn. Evolutionary minority games: the benefits of imitation. *Physica A: Statistical Mechanics and its Applications*, 329(3-4):484–498, Nov. 2003.

[28] L. Minku and X. Yao. Ddd: A new ensemble approach for dealing with concept drift. *Knowledge and Data Engineering, IEEE Transactions on*, 24(4):619–633, April 2012.

[29] E. Moro. The minority game: an introductory guide. In *Advances in Condensed Matter and Statistical Physics*, pages 263–286. Nova Science Publishers, 2003.

[30] A. Nguyen-Tuong, D. Evans, J. Knight, B. Cox, and J. Davidson. Security through redundant data diversity. In *Dependable Systems and Networks With FTCS and DCC*, pages 187–196, 2008.

[31] J. Niu, K. Cai, S. Parsons, E. Gerding, P. McBurney, T. Moyaux, S. Phelps, and D. Shield. Jcat: A platform for the tac market design competition. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1649–1650, 2008.

[32] J. Pitt, D. Busquets, and S. Macbeth. Distributive justice for self-organised common-pool resource management. *ACM Trans. Auton. Adapt. Syst.*, 9(3):14:1–14:39, Oct. 2014.

[33] J. Pitt, D. Ramirez-Cano, M. Draief, and A. Artikis. Interleaving multi-agent systems and social networks for organized adaptation. *Computational and Mathematical Organization Theory*, 17(4):344–378, 2011.

[34] A. Prasath, A. Venuturumilli, A. Ranganathan, and A. Minai. *Sensor Networks: Where Theory Meets Practice*, pages 39–59. Springer Berlin, 2009.

[35] N. Rescher. *Distributive Justice: A Constructive Critique of the Utilitarian Theory of Distribution*. The Bobbs-Merrill Co. Inc., New York, 1966.

[36] A. E. Roth and I. Erev. Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term. *Games and Economic Behavior*, 8(1):164–212, 1995.

[37] A. E. Roth, V. Prasnikar, M. Okuno-Fujiwara, and S. Zamir. Bargaining and market behavior in jerusalem, ljubljana, pittsburg, and tokyo: An experimental study. *American Economic Review*, 81(5):1068–1095, December 1991.

[38] N. Salazar, J. Rodriguez-Aguilar, and J. Arcos. Self-configuring sensors for uncharted environments. In *4th IEEE Conf. on Self-Adaptive and Self-Organizing Systems*, pages 134–143, Sept 2010.

[39] T. Schnier and X. Yao. Using negative correlation to evolve fault-tolerant circuits. In *5th International Conference on Evolvable Systems: From Biology to Hardware*, volume 2606, pages 35–46. Springer, 2003.

[40] H. Song, A. Elgammal, V. Nallur, F. Chauvel, F. Fleurey, and S. Clarke. On architectural diversity of dynamic adaptive systems. In *37th Intl Conf on Software Engineering: NIER Track*, 2015.

[41] H. Tuomisto. A consistent terminology for quantifying species diversity? yes, it does exist. *Oecologia*, 164(4):853–860, 2010.

[42] S. Yachi and M. Loreau. Biodiversity and ecosystem productivity in a fluctuating environment: the insurance hypothesis. *National Academy of Sciences of the United States of America*, 96(4):1463–1468, 1999.

[43] C. A. Zapart. On entropy, financial markets and minority games. *Physica A: Statistical Mechanics and its Applications*, 388(7):1157–1172, Apr. 2009.