Self-organization in a distributed coordination game through heuristic rules

Shubham Agarwal\textsuperscript{1}, Diptesh Ghosh\textsuperscript{2}, and Anindya S. Chakrabarti\textsuperscript{3,}\textsuperscript{a}

\textsuperscript{1} Indian Institute of Technology, 600036 Chennai, India
\textsuperscript{2} Production & Quantitative Methods Area, Indian Institute of Management, Ahmedabad 380015, India
\textsuperscript{3} Economics Area, Indian Institute of Management, Ahmedabad 380015, India

Regular Article

Received 4 August 2016 / Received in final form 20 September 2016
Published online 7 December 2016 – c⃝ EDP Sciences, Società Italiana di Fisica, Springer-Verlag 2016

Abstract. In this paper, we consider a distributed coordination game played by a large number of agents with finite information sets, which characterizes emergence of a single dominant attribute out of a large number of competitors. Formally, \( N \) agents play a coordination game repeatedly, which has exactly \( N \) pure strategy Nash equilibria, and all of the equilibria are equally preferred by the agents. The problem is to select one equilibrium out of \( N \) possible equilibria in the least number of attempts. We propose a number of heuristic rules based on reinforcement learning to solve the coordination problem. We see that the agents self-organize into clusters with varying intensities depending on the heuristic rule applied, although all clusters but one are transitory in most cases. Finally, we characterize a trade-off in terms of the time requirement to achieve a degree of stability in strategies versus the efficiency of such a solution.

1 Introduction

Understanding collective behavior of large-scale multi-agent systems is an important question in the econophysics and the sociophysics literature \cite{1,2}. Often, in social and economic worlds, we find emergence and evolution of global characteristics that cannot be explained in terms of microscopic properties of the systems \cite{3}. We find examples of particular social norms or technologies that become more popular than their competitors, which are not necessarily worse in terms of attributes. Similarly, norms and opinions emerge as an equilibrium through reinforcement among the social and economic agents \cite{4}. Leaders emerge in the political context through a complicated process of competition and interaction among millions of individuals \cite{5}. In this paper, we present a simple multi-agent game to study the emergence of one dominant attribute out of many potential competitors through complex and adaptive interactive processes \cite{1}; see also Ref. \cite{6}).

We focus on two properties of large scale interaction. One, agents can coordinate to specific choices from a number of potentially identical choices which may also be interpreted as the emergence of cooperation \cite{7}; and two, such coordination may take time to arrive, but once it does, it can be quite stable. Therefore, we address the dynamic (and potentially non-equilibrium) process through which coordination takes place, as well as the stability of the eventual equilibrium \cite{5} under the learning schemes considered. We consider a prototype model to study this kind of situation. In particular, we consider a simple coordination game with \( N \) agents and \( N \) choices. Individual agents aim to converge upon a single universally chosen outcome; i.e. the game can be thought of as a majority game.

In the language of game theory, this relates to the idea of equilibrium selection. In our game, there are \( N \) possible pure strategy Nash equilibria, each of which is equally attractive to the agents. The question is how, in the absence of communication, do the agents converge to only one equilibrium? Naturally, we do not allow a central planner to dictate the solution because that would make the problem trivial as well as unrealistic.

In our model, agents play the game repeatedly, and they always want to be in the majority. We first present several strategies based on naive learning that allow the agents to solve this coordination problem in a distributed manner \cite{8}. We next assume that the agents want to minimize their cost of experimentation, i.e. to come up with some fixed strategy as soon as possible even if it results in not being in the absolute majority. This leads to a trade off between the degree of stability (time to attain an approximate fixed rule of thumb) and the efficiency of the solution, i.e. degree of coordination. We propose multiple heuristic strategies for coordination that solves the problem to different degrees. We propose a Polya scheme following the famous Pólya’s urn model, which allows us to interpolate between multiple types of reinforcement learning processes \cite{9–11}.

This paper is intimately related to the literature on the minority game \cite{12–14}, and the generalization of the minority game known as the Kolkata Paise Restaurant.
In this paper, we show that agents converge to specific choices due to reinforcement learning. In particular, depending on the degree of reinforcement, agents may become stuck to different choices, creating clusters of different sizes. Clustering behavior has been studied in the context of minority games [20]. Here, such behavior also implies that, due to reinforcement, non-equilibrium configurations may also survive, and hence, it is not necessarily a “winner-take-all” scenario. Finally, we show that, if the agents value not only coordination but also the time requirement to achieve absolute coordination, then there would be a trade-off in terms of efficiency versus stability of the final solution.

## 2 N-agent coordination game

We consider $N$ agents and $M$ options. Time is discrete and, at every point in time, each agent makes a choice about which among the $M$ options to use. To fix the idea, one can imagine each option to represent one restaurant which an agent will visit in a time slice. Therefore, each of the $N$ agents’ strategy is to choose a restaurant to visit in each time slice. Any given restaurant can accommodate a maximum of $N$ agents in any particular time slice. The agents’ objective is to stay in the majority, i.e. the agents would like to move to a restaurant which has a higher number of agents. In principle, $N$ may not be equal to $M$. To impose symmetry on the problem, we assume $N = M$, i.e. the number of agents is equal to the number of restaurants.

We also emphasize here that the game is necessarily non-cooperative and no communication is allowed among the agents. The information set for all agents is constrained to only their history and partial knowledge about past evolution of the restaurant occupations. Naturally, allowing a full set of history across all restaurants to be available to the agents would immediately solve the problem because the agents can employ a strategy that in time slice 1 they choose randomly and, in the next time slice, they move to the restaurant that attracted the greatest number of agents in the first time slice. To have a non-trivial solution, we allow only a partial set of history to be available to the agents. We elaborate on the specifics of the information sets for each type of strategies below.

Figure 1 shows the payoff matrix for a general convergence game for two players. Both players have strategies A and B, i.e. they may choose to visit either restaurant A or restaurant B. If both of them decide to visit the same restaurant (either A or B), then the outcome for both would be better than if they chose different restaurants. A couple of points may be noted. This game is a simplified version of the famous Battle of Sexes game (see for example [21] for a textbook treatment). The Battle of Sexes game is played by two players, who try to converge upon a single restaurant although they differ in their preferences over the restaurants. In this paper, we assume a multi-agent multi-choice scenario with $2 \leq N < \infty$ agents, but assume that all agents have identical preference over the restaurants.

The agents decide on their strategies based on the attractiveness of a restaurant. We define the attractiveness (A) of a restaurant as the number of agents that have chosen that restaurant. Thus attractiveness depends on the information set that the agent possess. Naturally, at any given time slice, it is not possible to know how many other agents are choosing a given option.

For the sake of completeness, we define Nash equilibria for the coordination game. A Nash equilibrium is defined as a strategy collection such that given every other agent’s strategy each agent is weakly better off by not switching to a different strategy. For our purpose, this description suffices. For a textbook description, see reference [21]. From Figure 1 it can be verified that there are two pure-strategy Nash equilibria, viz. both go to either restaurant A or both go to B. In a general $N$-agent game, there would be $N$ pure strategy Nash equilibria.

It may be noted that a Nash equilibrium is an equilibrium description and a static concept. It does not explain how one equilibrium would be chosen from many candidate equilibria in reality. So the essential question is how do agents coordinate to converge on one equilibrium out of $N$ possible choices, in the absence of any information about what the other agents are thinking?

We specify a set of strategies below that solves this problem using finite sets of information and, in certain cases, with no information about the other agents.

## 3 Heuristic updating strategies

In this section, we present a set of updating strategies that the agents may employ in the coordination game.
These can be thought of as rule-of-thumb strategies. In particular, they do not exhaust all possible strategies, but provides a comprehensive set that is useful for solving the game.

In the following, we define a strategy of an agent as a vector of probabilities that she assigns to the restaurants, i.e. each of the elements of the vector would represent the probability with which she chooses one restaurant. Formally, we denote the \( i \)th agent’s strategy at time slice \( t \) as \( \{p_{ijt}\} \) for \( j \in N \) where the index \( j \) denotes the \( j \)th restaurant. Learning is introduced as updating the probability vector based on success or failure in the past.

### 3.1 No learning

We begin with a No Learning strategy. This entails zero probability updating and represents a baseline case.

#### 3.1.1 Zero updating

This strategy has two parts. Consider any generic time slice \( t \). First, the \( i \)th agent \( (i \in N) \) assigns the following probability to the restaurants,

$$ p_{ijt} = \frac{1}{N} .$$

Naturally, this would lead to a randomly distributed allocation of agents across restaurants. In particular, reference [15] shows that the occupancy fraction, i.e. the number of restaurants occupied as a fraction of the total number \( N \), would be 63.5%. So the first part is far from sufficient to ensure coordination.

The second part of the strategy allows the agent, at time slice \( t \), to make a comparison between the choice made at time slice \( t \) and the restaurant she is in at time slice \( t \). Because attractiveness depends on the number of agents in a restaurant, we denote the \( j \)th restaurant’s attractiveness at time slice \( t \) by \( A_{jt} \). Therefore, the strategy of an agent who is at restaurant \( k \) is to go to restaurant \( j \) if

$$ A_{jt} \geq A_{kt},$$

and otherwise, the agent stays at \( k \).

#### Information required

The information set of the \( i \)th agent who is at restaurant \( k \) at the \( t \)th time slice comprises \( A_{kt} \) and \( A_{jt} \) where \( j \) is the outcome of the random selection scheme (Eq. (1)) for the \( i \)th agent. Note that this entails gathering information about the \( j \)th restaurant that the \( i \)th agent has not visited at time slice \( t \), implying that we are allowing for local information. In principle, one can imagine that the agents may have to pay a cost to gather that information. This is a point we will later take on in fuller details.

![Fig. 2. Simulation results for the ‘No learning’ strategy. Number of time slices required for convergence, averaged over 10 parallel simulations. \( T(N) \) denotes the time of convergence with \( N \) number of agents. The vertical bars show standard deviations of the simulation results. In the inset, we plot \( T(N)/N \) as a function of the system size \( N \). Time required for convergence scales linearly with \( N \).](image)

![Fig. 3. Simulation results for the ‘No learning’ strategy. This strategy leads to convergence linearly with time. On the y-axis we plot the number of people in the restaurant with the largest (red), 2nd largest (black) and 3rd largest (blue) number of agents. On the x-axis we plot time.](image)

### Results

We present simulation results in Figures 2 and 3. Figure 2 shows the time required for absolute convergence \( T(N) \), i.e. the minimum number of time slices required for all agents to converge at one restaurant, as a function of the number of agents \( N \). It shows a linear trend with a coefficient about 8 on an average. In the inset, we show the ratio \( T(N)/N \) as a function of \( N \), which fluctuates around 8 after an initial steep rise. In the main diagram, we also provide an estimation of the standard deviation across 10 simulations.

Figure 3 shows the dominance of one restaurant over others (we show the second and the third most populated ones) over time in one simulation with \( N = 1000 \).
The second and the third most crowded restaurant initially starts attracting more agents before decaying completely in terms of the number of agents as the dominant one becomes absolutely dominant and attracts all agents.

These results show that symmetry-breaking occurs due to stochastic choices. All restaurants start off by being equally popular. But at the end, only one of them emerges as the most popular choice and all other restaurants have no agents.

In the following sections, we introduce updating rules based on the success or failure of the past choices. Agents can learn and reinforce their probabilistic strategies based on the available information set.

### 3.2 Learning strategies with ex-ante knowledge

This is a direct extension of the previous strategy. At each time slice, the ith agent \(i \in N\) makes a choice of restaurants using a probability vector \(\{p_{ijt}\} \forall j \in N\). Then she compares the attractiveness of the chosen restaurant and the restaurant she is currently in, and moves to the one with higher attractiveness in the next time slice.

Finally, the ith agent updates her probability vector based on attractiveness. This last step of probability updating differentiates this strategy from the No Learning strategy.

We call this strategy \textit{ex-ante} because the agents can decide whether or not to move to a chosen restaurant by gathering information about the attractiveness of the current restaurant and the newly chosen one. Later in Section 3.4 we study a case with ex-post updating that relaxes this assumption.

We extend the strategy under consideration in multiple dimensions. In the first case, agents reward for higher attractiveness and punish for lower attractiveness. Formally, higher attractiveness implies that the agent will assign higher weight in the probability vector, and will reduce weight for restaurants with lower attractiveness. This strategy we label as \textit{symmetric} in updating.

In the second case, the agents only reward higher attractiveness. We label this strategy as \textit{asymmetric} updating. Further, we consider the cases where the agents are allowed to choose more than one restaurant to pick the best option. Formally, the information set increases to \(K\) choices per agent, where \(K = 1, 2, 3, \ldots\), etc. Naturally, setting \(K = N\) makes the problem trivial. So we concentrate on cases with sufficiently small values of \(K\).

Below we describe the strategies in details.

#### 3.2.1 Symmetric updating

Consider agent \(i\) where \(i \in N\), at any generic time slice \(t\). Suppose she is at restaurant \(r\) and, given her probability vector \(\{p_{ijt}\}\), she probabilistically picks restaurant \(l\). If \(A_{lt} < A_{rt}\), she stays at restaurant \(r\). Otherwise, she moves to restaurant \(l\).

Simultaneously, the agent updates the probability of restaurants \(l\) and \(r\) such that the one with higher attractiveness will gain in probability by fraction \(f_{1}\) while the other will decrease by fraction \(f_{2}\). Naturally, the resulting sum is normalized to 1. Formally, if \(A_{lt} < A_{rt}\),

\[
p_{ij(t+\frac{1}{2})} = \begin{cases} p_{ijt} + f_{1}(1 - p_{ijt}) & \text{for } j = r, \\ p_{ijt} - f_{2}(p_{ijt}) & \text{for } j = l. \\ \end{cases}
\]

If \(A_{lt} = A_{rt}\),

\[
p_{ij(t+\frac{1}{2})} = p_{ijt} \quad \text{for } j \in N,
\]

and if \(A_{lt} > A_{rt}\),

\[
p_{ij(t+\frac{1}{2})} = \begin{cases} p_{ijt} + f_{1}(1 - p_{ijt}) & \text{for } j = l, \\ p_{ijt} - f_{2}(p_{ijt}) & \text{for } j = r. \\ \end{cases}
\]

Finally, probabilities are normalized:

\[
p_{ij(t+1)} = \frac{p_{ij(t+\frac{1}{2})}}{\sum_{i} p_{ij(t+\frac{1}{2})}}.
\]

**Information required**

The information set is identical to the No Learning strategy for \(K = 1\). For higher values of \(K\), we allow the agents to have more information about the occupancy of the restaurants in the previous time slice to make a comparison.

**Results**

Figure 4 shows the simulation results for this strategy with \(N = 1000\). On the \(x\)-axis, we plot the restaurants, and on the \(y\)-axis, we plot the number of agents \(n_{i}\) that
go to the $i$th restaurant for all restaurants $i \in N$. We show two snapshots. One at time slice $t = 5000$ and the other at $t = 10000$. The three rows show the distribution of agents under three different information sets, $K = 1, 2, 3$.

The first thing to notice is that the dynamics become considerably slower. Even after 10,000 time slices (for $N = 1000$), attaining coordination is very difficult, as the panels on the right in Figure 4 show very clearly. However, we note that convergence is guaranteed. As an explanation, consider a case with a distribution of agents across all restaurants $(501, 499, 0, \ldots, 0)$. Given the current strategy, only the first restaurant can attract agents, and the second one can only lose agents, however slow the process might be.

The next important feature is that increasing the information set even by a limited amount (going from $K = 1$ to 2 and 3) drastically improves the degree of coordination, although the dynamics become slow after a certain point. For example, in the bottom row, we see that the distribution changes very slowly going from $t = 5000$ to $t = 10000$.

Therefore, we see that for a long time there are clusters of agents in different restaurants before they all collapse into one giant cluster, i.e., absolute convergence takes place. Such clustering behavior is transitory.

### 3.2.2 Asymmetric updating

Consider agent $i$ at time $t$ in restaurant $r$, probabilistically picking another restaurant $l$. If $A_{lt} < A_{rt}$, she stays at restaurant $r$. Otherwise, she moves to restaurant $l$. The asymmetric updating scheme differs from the symmetric scheme in the way she updates the probability vector $\{p_{ijt}\}$.

If there is a difference between attractiveness of the current restaurant and the probabilistically picked one, the agent assigns a higher weight to the more attractive option and reduced weight to every other restaurant. Formally, if $A_{lt} < A_{rt}$

$$ p_{ij(t+\frac{1}{2})} = \begin{cases} p_{ijt} + f(1-p_{ijt}) & \text{if } j = r \\ (1-f)p_{ijt} & \text{otherwise.} \end{cases} $$

If $A_{lt} = A_{rt}$,

$$ p_{ij(t+\frac{1}{2})} = p_{ijt} \quad \text{for } j \in N, $$

and if $A_{lt} > A_{rt}$

$$ p_{ij(t+\frac{1}{2})} = \begin{cases} p_{ijt} + f(1-p_{ijt}) & \text{if } j = l \\ (1-f)p_{ijt} & \text{otherwise.} \end{cases} $$

Finally, probabilities are normalized:

$$ p_{ij(t+1)} = \frac{p_{ij(t+\frac{1}{2})}}{\sum_l p_{ij(t+\frac{1}{2})}}. $$

### Information required

This strategy requires exactly the same set of information as the symmetric updating strategy.

### Results

Figure 5 presents the simulation results with the asymmetric updating strategy for $f = 0.25$. The results are comparable to the symmetric updating scheme. We see that the dynamics become slower. As we expand the information set from $K = 1$ to 2 and 3, convergence takes place much faster in the initial phase. After sufficient time, it becomes slow for all information sets. However, with a sufficient number of iterations, absolute convergence takes place.

We see that for a long period of time there are clusters. But as with the symmetric updating, this behavior is transitory. By varying the parameter $f$, we studied the dynamics before convergence. Figure 6 presents simulation results for two different values of $f$ with multiple information sets ($K = 1, 2, 3$). In order to quantify the degree of stability before convergence, we compute the average of the maximum probabilities that the agents assign to any restaurant. With smaller values of $f$ ($f = 0.1$), the average probability goes up very fast compared to larger values ($f = 0.9$). Also, with bigger information sets, the average of the maximum probabilities rise more slowly than with smaller information sets. This is consistent with the finding that coordination occurs much faster with bigger information sets because that requires multiple switching to ensure convergence. Naturally, switching happening...
at a higher frequency leads to lesser reinforcements to specific restaurants.

3.3 Reinforcement learning through Pólya’s urn model

We introduce a new strategy using the Pólya’s urn model [9–11] that effectively captures reinforcement learning. Let us define

$$\phi = \frac{m_{\text{min}}}{N - m}$$

(3)

where \(m\) is a tunable parameter taking discrete values within 0 and \(N\). We denote the number of times the \(i\)th agent has visited restaurant \(j\) before time slice \(t\), by \(n_{ijt}\). Then the probability of choosing restaurant \(j\) is given by:

$$p_{ijt} = \frac{1 + \phi n_{ijt}}{N + \phi \sum n_{ijt}}$$

(4)

Intuitively, this is an extension of the basic No Learning strategy (which would require \(p_{ijt} = 1/N\)) by embedding reinforcement learning through Pólya’s urn model. The parameter \(m\) effectively controls the degree of reinforcement. When \(m = 0\), we have \(\phi = 0\) which implies zero reinforcement. On the other hand, when \(m \to N\), \(\phi \to \infty\) implying infinite reinforcement. Thus we can interpolate between the two extremes.

Information required

The required information set for the \(i\)th agent is derived only from the full sequence of success of the agent at different restaurants. It is reasonable to assume that the agents keep track of their own visits. Also note that, at any time slice, the agent does not require any information from a restaurant that she is not visiting as was required with the earlier strategies. This is possible because there is no comparison involved. The probabilistic strategies are devised based on historical success.

Results

Figure 7 presents numerical results for different values of \(m\) (see Eq. (3)) with \(N = 500\) and \(t = 5000\). In the left panel, we show the number of restaurants occupied (\(n_{\text{occ}}\)) with at least one agent for different values of the factor \(m\) and at different time slices \(t\). Clearly when \(m = 0\), the Pólya’s scheme will converge to the No Learning case and absolute convergence occurs. This implies only one restaurant would be occupied. This can be seen from the figure by looking at the bars for different time slices by fixing \(m = 0\). In the other extreme with \(m = N - 5\) (for simulations, we can not set \(m = N\)), we see that around 318 restaurants out of 500 have been occupied. This is consistent with the notion that setting the factor \(m\) very close to \(N\) leads to infinite reinforcement, implying that, if an agent goes to one restaurant, she would stay there for the remaining time slices. So, effectively, the choices in the first time slice themselves determine the distribution of agents across restaurants because that distribution will never change due to infinite reinforcement. It is easy to show analytically that because the agents are starting with uniformly distributed probabilities (\(p_{ij0} = 1/N\)), in the first time slice 63.2% of the restaurants would be occupied in the limit. In the other limit \(m \to \infty\), we see that around 318 restaurants out of 500 have been occupied. This is consistent with the notion that setting the factor \(m\) very close to \(N\) leads to infinite reinforcement, implying that, if an agent goes to one restaurant, she would stay there for the remaining time slices. So, effectively, the choices in the first time slice themselves determine the distribution of agents across restaurants because that distribution will never change due to infinite reinforcement. It is easy to show analytically that because the agents are starting with uniformly distributed probabilities (\(p_{ij0} = 1/N\)), in the first time slice 63.2% of the restaurants would be occupied. We are skipping the derivation of this fraction. Interested readers can refer to [15]. One can easily verify that 318/500 is close to 63.5% and hence this validates our numerical results. The right panel in the same figure shows
the fraction of restaurants occupied, i.e. \( f_{\text{occ}} = n_{\text{occ}} / N \). The results are perfectly consistent with the left panel.

We also note that having \( m = N \) in the Polya’s scheme (i.e. infinite reinforcement) is identical to assuming \( f = 1 \) in the asymmetric updating strategy. Therefore, at the limits, these two strategies are identical. This strategy allows us to interpolate between a wide spectrum of reinforcement by changing the factor \( m \). In particular, it allows us to cover the same range as are separately done by the symmetric and asymmetric updating strategies.

3.4 Learning strategies with ex-post knowledge

In the case of ex-ante knowledge in Section 3.2, we studied strategies with which agents can obtain information about the newly chosen restaurant’s attractiveness, and make a comparison between the chosen restaurant’s and the current restaurant’s attractiveness. However, it might be a costly activity to know the attractiveness of another restaurant before actually visiting it. In the present section, we study the same set of strategies where the agents can obtain information about attractiveness only after she moves to the chosen restaurant. An important distinction from the earlier cases is that the present strategy allows for regret. After the agent moves to a new restaurant, she comes to know about its attractiveness and hence cannot do comparison prior to switching. Updating the probability vector happens the same way depending on relative attractiveness as was done in Section 3.2.

3.4.1 Symmetric updating

Consider agent \( i \) where \( i \in N \), at any generic time slice \( t \). Suppose she is at restaurant \( r \) and, given her probability vector \( \{ p_{ijt} \} \), she probabilistically picks restaurant \( l \). After knowing both \( A_l \) and \( A_{lt} \), probability vector \( p_{ijt} \) is updated exactly the same way as in Section 3.2.1. To avoid repetition, we are skipping the probability updating schemes.

**Information required**

The required information comes from the restaurants that the agent has visited. Hence, there is no external information acquired.

**Results**

Figure 8 shows the simulation results in the top panels for \( f_1 = 1 \) and \( f_2 = 0.1 \) (panels \( a_1 \) and \( a_2 \) respectively). We show results for two time slices, at \( t = 5000 \) and \( t = 10000 \). As in the earlier case, this strategy is also quite slow but eventually converges to a single restaurant in the limit. Naturally, this is slower than the ex-ante knowledge case.

![Fig. 8. Simulation results for the ‘ex-post updating’ strategy with symmetric (top panels, \( f_1 = 1 \) and \( f_2 = 0.1 \)) and asymmetric (bottom panels, \( f = 0.5 \)) reinforcements. On the x-axis, we plot the identity of the restaurants. Symmetric updating leads to higher crowding for the restaurants.](image)

3.4.2 Asymmetric updating

Similarly to above, consider agent \( i \in N \), at any generic time slice \( t \). Suppose she is at restaurant \( r \) and, given her probability vector \( \{ p_{ijt} \} \), she probabilistically picks restaurant \( l \). After knowing both \( A_l \) and \( A_{lt} \), probability vector \( p_{ijt} \) is updated exactly the same way as in Section 3.2.2.

**Information required**

Required information comes solely form the restaurants she visited and hence no external information is acquired.

**Results**

Simulation results have been reported in the lower panels of Figure 8 (panels \( b_1 \) and \( b_2 \)) and Figure 9. We see that the results for distribution of agents across the restaurants are qualitatively similar to those in the case of symmetric updating, except that coordination is poorer because there are many restaurants with small numbers of agents.

We study the degree of stability of the transient clusters thus formed in Figure 9. For higher values of \( f \), the average over maximum probabilities rises quite quickly compared to lower values of \( f \), though eventually their behavior is similar. Finally, we study the evolution of coordination among agents with ‘ex-ante’ and ‘ex-post’ knowledge in Figure 10 (panels \( a_1 \) and \( a_2 \)) for ex-ante knowledge; \( b_1 \) and \( b_2 \) for ex-post knowledge). It is clearly seen that ex-post knowledge leads to faster convergence.

4 Self-organization and coordination

In the present section, we discuss the extent to which self-organization occurs in the multi-agent system that solves the coordination problem.
4.1 Emergence of coordination

We have seen that some of the strategies, especially those which require ex-ante information or knowledge, can in principle be thought of as requiring some costs to be paid in order to acquire the information. Also, realistically, the agents might have a trade-off in terms of how quickly they can converge upon a solution versus the efficiency of the solution. That is, they may find it useful to be in the majority, but not necessarily an absolute majority, given a lower time requirement to reach the configuration.

A parallel theme is that initially all restaurants are identical. But with absolute convergence, only one of them emerges as the winner. This can be interpreted as how a specific social norm may emerge from multiple possibilities that are a priori equally likely.

Thus emergence of absolute coordination has two contributing factors that can be potentially costly. The first one is obviously the cost of lack of coordination. The second one is the cost of waiting to reach coordination. This can be most clearly seen in the clustering behavior where multiple choices survive as the agents achieve partial coordination reasonably quickly.

4.2 Cluster formation

We have already seen that clustering behavior can be transient, but in almost all cases they are very slowly evolving. This implies that we observe clusters of agents in different restaurants for a very long time. Figure 11 shows four instances of the probability density function of clusters. We tracked choices of $N = 1000$ agents over $t = 10000$ time slices. We assumed all four cases (ex-ante, ex-post and symmetric-asymmetric) with the previously mentioned parameter values. The resulting probability density function has been averaged over 10 simulations. Both ex-ante and ex-post with symmetric updating rules (panels (a) and (c)) show strong clustering behavior, whereas the other two cases show very moderately distributed clusters. Panel (b) shows fit with exponential distribution $\exp(-S/\tau)$ with an estimated parameter value of 4.85 (method of moments) and panel (d) shows fit with gamma distribution $\alpha \exp(-S/\beta)$ with estimated parameter values 2.71 and 1.38 (method of moments). For checking robustness, we carry out a maximum likelihood estimation as well. For the exponential fit, the estimated...
efficient and time of convergence with Pólya updating scheme. Simulation results show a clear monotonic decay for $N = 500$.

The parameter value is 4.78 with 95% confidence interval between [4.16, 5.56]. For the gamma density function, the estimated values are 2.26 and 1.80 with 95% confidence interval between [1.94, 2.62] and [1.52, 2.14], respectively.

4.3 Efficiency and cost of waiting

As discussed above, the agents may incur a cost to execute the strategies that are computationally intensive, and hence if there are strategies that take a very long time to reach a state of absolute convergence, the agents may prefer less efficient solutions, if that is achievable sooner. We study this trade-off in Figure 12 which plots the number of time slices required by the average over maximum probabilities to reach at least 0.8 versus the percentage of restaurants occupied. The variable in the $y$-axis represents the cost in terms of waiting time. The variable in the $x$-axis represents the cost in terms of inefficiency of the solution (smaller percentage occupancy would be more efficient). We plot the trade-off by simulating a system of $N = 500$ agents with the Pólya updating scheme ($m = 50, 75, 100, \ldots, 475, 495$). The values on the $x$-axis shows the occupancy at the time slice when $\langle P_{\text{max}} \rangle$ reaches 0.8.

The trade-off is clearly seen in terms of cost minimization. A lower waiting cost leads to higher occupancy and hence to inefficiency and vice versa. This is a useful feature of the model for understanding the trade-off between the waiting cost to arrive at an allocation and the efficiency of the allocation.

Potentially, there could be other definitions of efficiency including usage of other threshold values. In general, we find that the time to convergence depends crucially on the exact parametric specifications of the strategies. The functional dependence of the time of convergence, although monotonically varying, is difficult to describe in a single algebraic form.

5 Summary

In this paper, we have studied a model of distributed coordination in the context of a multi-agent, multi-choice system. We consider a game with multiple Nash equilibria all of which are equally likely. The basic problem is to find which equilibrium will materialize if the agents engage in repeated interactions, and how quickly they converge upon the equilibrium. Essentially, we solve the problem of equilibrium selection through distributed coordination algorithms.

We propose a number of strategies based on different types of naïve learning. In particular, reinforcement learning via the Pólya’s urn model provides a very useful benchmark. We show that the system self-organizes with very slow dynamics and transient clusters. Finally, we characterize a trade-off between waiting cost to attain an allocation and the efficiency of the allocation. If the agents try to lower the waiting costs, then stability is attained sooner, but efficiency of the solution is also low. The inverse is also true that if the agents try to reduce the cost of inefficiency of the solution, then the waiting cost increases.

This problem sheds light on the complexity of equilibrium selection, and may provide a useful model for multi-agent coordination games and collective dynamics in general.

Author contribution statement

A.S.C. and D.G. formulated the problem. S.A. ran the simulations. S.A., D.G. and A.S.C. contributed equally while writing the paper.

This work was partially supported by the institute grant, IIM Ahmedabad.

References

1. A. Namatame, S.-H. Chen, Agent-Based Modeling and Network Dynamics (Oxford University Press, 2016)
2. P. Sen, B.K. Chakrabarti, Sociophysics: An Introduction (Oxford University Press, 2014)
3. M.J. Salganik, P.S. Dodds, D.J. Watts, Science 311, 854 (2006)
4. D.J. Watts, P.S. Dodds, J. Cons. Res. 34, 441 (2007)
5. D. Stauffer, Sociophysics Simulations II: Opinion Dynamics, arXiv:physics/0503115 (2005)
6. E. Pugliese, C. Castellano, M. Marsili, L. Pietronero, Eur. Phys. J. B 67, 319 (2009)
7. D. Challet, Y.-C. Zhang, Physica A 246, 407 (1997)
8. D. Challet, J. Econ. Dyn. Control 32, 85 (2008)
9. G. Pólya, Sur quelques points de la théorie des probabilités, Annales de l'I.H.P. 1, 117 (1930)
10. N.L. Johnson, S. Kotz, Urn Models and Their Application: An Approach to Modern Discrete Probability Theory (Wiley, 1977)
11. H. Mahmoud, *Pélya Urn Models, Texts in Statistical Science series* (Taylor and Francis Ltd, Hoboken, NJ, 2008)
12. D. Challet, M. Marsili, Y.-C. Zhang, *Minority games: interacting agents in financial markets* (Oxford University Press, 2004)
13. D.B. Fogel, K. Chellapilla, P.J. Angeline, IEEE Trans. Evol. Comput. *3*, 142 (1999)
14. J.-Q. Zhang, Z.-X. Huang, Z. Wu, R. Su, Y.-C. Lai, Sci. Rep. *6*, 20925 (2016)
15. A.S. Chakrabarti, B.K. Chakrabarti, A. Chatterjee, M. Mitra, Physica A *388*, 2420 (2009)
16. A. Ghosh, A. Chatterjee, M. Mitra, B.K. Chakrabarti, New J. Phys. *12*, 075033 (2010)
17. W.B. Arthur, Am. Econ. Rev. *84*, 406 (1994)
18. A. Chakraborti, D. Challet, A. Chatterjee, M. Marsili, Y.-C. Zhang, B.K. Chakrabarti, Phys. Rep. *552*, 1 (2015)
19. D. Challet, Competition between adaptive agents: learning and collective efficiency, in *Collective and the design of complex systems*, edited by K. Turner, D. Wolpert (Springer, 2004)
20. S. Hod, E. Nakar, Phys. Rev. Lett. *88*, 238702 (2002)
21. M.J. Osborn, *An introduction to game theory* (Oxford University Press, 2004)