DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch146

Conformity and Nonconformity in Collective Robotics: A Case Study

Gregory Vorobyev*, Andrew Vardy, and Wolfgang Banzhaf

Memorial University of Newfoundland, St. John's, Canada, *gvorobyev@mun.ca

Abstract

In this work, we develop a social behavioral model designed for multi-agent systems for solving the collective sorting task. Experiments show that under this model agents are capable of improving their performance significantly and can achieve better results than conventional swarms of agents lacking communication and social abilities.

Introduction

In his fascinating book "The Social Animal", Elliot Aronson defines conformity "as a change in a person's behavior or opinions as a result of real or imagined pressure from a person or group of people" (Aronson, 2007). Conformity is one of the essential and most important aspects of human society. Failure to conform to the rules issued by society may turn out to be not only merely inconvenient, but even dangerous. Driving down the wrong side of the street can be an example of a nonconformist behavior that will most likely lead to tragic consequences (Aronson, 2007). Nonconformity, however, often works to the long-term benefit of the society as a whole. One example of "useful" nonconformity could be a scientist attempting to look at a well-known problem from an entirely new angle, which can be controversial to the viewpoint generally accepted. Sometimes this could result in a revolution in science (i.e., consider Einstein's theory of relativity vs. classical Newtonian mechanics).

In this paper, we develop a simple model of conformity and nonconformity in an artificial society. This model is not intented to be a valid counterpart of the relevant phenomena in the human society. Rather, our aim is to attempt to incorporate some degree of *social intelligence* (specifically, the ability to choose between conformity and nonconformity in behavior) in artificial agents and to study whether this additional "social" part of agent reasoning could be beneficial in terms of the performance on the task being executed.

As a case study, we use the sorting task in the context of multi-agent systems, which is formulated as follows: given a set of objects of different types $\{x_1, x_2, ..., x_n\}$, the group of N agents is to collect them into homogeneous clusters. *Swarm robotics* offers distributed algorithms for solving this problem (Bonabeau et al., 1999; Deneubourg et al., 1991; Bayindir and Sahin, 2007; Beckers and Holland, 1994; Melhuish and Hoddell, 1998; Wang and Zhang, 2004; Verret et al., 2004; Vorobyev et al., 2012). The distinguishing property of the swarm-based approach is that agents operate and perceive only locally; thus, no global supervision and/or knowledge is required. While swarm-based algorithms typically show slower convergence than centrally-controlled approaches, their advantages are simplicity, flexibility and robustness (Sahin, 2005). A swarm agent has very limited sensing capabilities. As it is highlighted in the next section, the only input to one popular swarm-based sorting algorithm by (Deneubourg et al., 1991) is $f(x)$, which roughly estimates the density of objects of type x in the immediate neighborhood. To obtain f, an agent only needs a sensor which allows to recognize the type of an object, if there is any, right in front of the agent. In many contributions in this field, agents are even not aware of each other; i.e., *kin recognition* is not present (Bayindir and Sahin, 2007). Such "social ignorance" is viewed as beneficial, because it guarantees scalability and robustness of the approach.

This work primarily concentrates on extending Deneubourg et. al.'s sorting algorithm by introducing additional input information. We refer to this information as *social*, because it represents knowledge about the goals of other agents. Obtaining this new type of information will require explicit communication, as opposed to implicit communication, commonly employed by robotic swarms (e.g., through *stigmergy* (Beckers and Holland, 1994)). Since it is generally accepted that communication capabilities in swarm-based systems should be kept as minimal as possible, we also refer to the group of our "socially intelligent" agents as a *society* as opposed to a *swarm*. This work offers some evidence that socially aware agents could perform more effectively and "intelligently" than their swarm counterparts. The term *social intelligence*, as well as *artificial social intelligence* are probably too broad to cover in one paper; rather, we concentrate on just one social phenomenon – conformity and nonconformity. Much like traditional swarm robotics is inspired by social insects (see,

e.g., (Bonabeau et al., 1999)), we are inspired by arguably the most successful social beings we know – humans.

Although sharing information between agents as such is not new in the field of collective and swarm robotics, most of the contributions that employ this concept tend to focus on extending individual's sensory capabilities by accessing perception, or memory, of others (Verret et al., 2004; Grech et al., 2012). For example, in the collective clustering task, agents can share information about the clusters they have seen, and perform different actions based on that information; in some sense, this would be equivalent to having non-communicating agents with enhanced sensing capabilities (e.g., increased sight range). In contrast, this paper studies how agents with limited sensing capabilities make decisions based purely on the number of other agents that made a similar (or different) decision. *Conformist* agents always tend to make decisions that are similar to the course of the majority, whereas *nonconformist* agents are more independent. In some sense, our agents resemble *zero-intelligent* particles described in (Bentley and Ormerod, 2011).

One important assumption made in this paper, in addition to the ability of agents to distinguish objects of different types and to explicitly communicate with each other, is that they can remember *home locations* in the environment to which they can return if they wish so. This assumption is, in fact, biologically plausible. For example, honeybees can travel long distances and return to their hives (Seeley, 2010). We are not concerned with the details of the implementation of a homing mechanism; rather, we just assume that our agents can store their home locations in their *local coordinate system*. This information will further be subject to social exchanges.

The next section describes the sorting algorithm followed by socially intelligent agents. We then present experimental results, a comparison with Deneubourg et. al.'s "socially ignorant" agents, and the analysis of different parameters of the social model.

The model

The proposed "social" algorithm is derived from Deneubourg et. al.'s model, first introduced in 1991 (Deneubourg et al., 1991) (which from now on is referred to as "Ant-Like Robots", or "ALR", model). The behavior of ALR agents can be summarized as follows. Each agent moves randomly. If an agent who is not carrying an object encounters an object of type x, it decides whether or not to pick it up. The probability $p_p(x)$ of doing so is defined as follows:

$$p_p(x) = \left(\frac{k_p}{k_p + f(x)} \right)^2 \tag{1}$$

where $p_p(x)$ is the probability to pick up the object of type x, $0 \leq f(x) \leq 1$ is a function estimating the relative density of objects of type x in the current neighborhood, and k_p is an arbitrary constant. Each agent has a short-term memory m of size N_m for storing the information of what kind of objects (if any) it has encountered in the recent past. $f(x)$ is calculated based on that memory:

$$f(x) = \frac{1}{N_m} \sum_{i=1}^{N_m} \begin{cases} 1, & \text{if } m_i \equiv x, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In a similar manner, the probability of depositing the object being carried upon encountering an empty cell is defined as follows:

$$p_d(x) = \left(\frac{f(x)}{k_d + f(x)} \right)^2 \tag{3}$$

where k_d is a constant. Thus, $p_p(x)$ decreases with $f(x)$ from 1 (when $f(x) = 0$) to 0.25 (when $f(x) = k_p$), and $p_d(x)$ increases with $f(x)$ from 0 (when $f(x) = 0$) to 0.25 (when $f(x) = k_d$).

Home locations and division of labor

In the model described, agents pick up and put down objects as they walk randomly in the environment. The performance of the sorting task, however, can be significantly improved if each agent has a *home location*, that is, the location where the cluster of objects is to be formed. The algorithm can then be modified as follows. An agent starts looking for an object to pick up by roaming randomly. Upon encountering an object, the agent picks it up with probability p_p. The agent then deterministically returns to its home location. Once the home location is reached, the agent starts roaming randomly and tries to put down the object into any empty cell it finds with the probability p_d. When the object is deposited successfully, the agent starts looking for another object.

It is obvious that bringing objects of different types to the same home location will not solve the sorting problem. Thus, there must be only one type of object associated with each particular home location and hence with each agent. To be more general, from now on, we will refer to the object type as *task*; e.g., "agent A is executing task x" is equivalent to "agent A looks for objects of type x and brings them to its home location". The question is then how to assign tasks to agents (that is, how to configure *division of labor*). One simple solution (perhaps not optimal, but acceptable in our case) would be to assign a task to an agent according to the type of the very first object which that agent encounters at the beginning of the simulation. Another question is how to assign initial home locations to agents; in this paper, this assignment is uniformly random.

As it will be shown further, a group of agents employing the homing algorithm demonstrates better performance on average than randomly roaming ALR agents. It is obvious, however, that the homing approach itself has a significant drawback: it is not flexible. The assignment of tasks and

home locations to agents is fixed. Therefore, if none of the agents has been assigned to a task x, then objects of that type will be unaffected by the sorting process. On the other hand, if more than one agent have been assigned to task x, then convergence to a single cluster of type x will never be achieved (assuming that the home locations of x-agents are sufficiently far from each other). The situation is even worse if the number of agents is large, or the distribution of objects of different types is not uniform.

Conformity and nonconformity

We propose to solve the problem of a fixed assignment of home locations and tasks by using explicit communication between agents. If two agents A and B are currently located within the communication range r of each other, they can share information about their home locations (denoted as h_A and h_B, respectively) and tasks (denoted as x_A and x_B).

If both of them are working on the same task x, then they should agree on a single home location to guarantee convergence to a single cluster. In our model, the probability $p(h_B \leftarrow h_A)$ of that h_A will convert to h_B is defined as follows:

$$p(h_B \leftarrow h_A) = \min\left[1, \left(\frac{|h_B|}{|h_A| + l_h}\right)^2\right] \qquad (4)$$

where $0 \leq |h_X| \leq 1$ is the estimated proportion of other agents (excluding X) that have their home locations at h_X, and l_h is a constant, which we interpret as *home loyalty*. Thus, $p(h_B \leftarrow h_A)$ increases with $|h_B|$ and decreases with $|h_A|$.

$|h_X|$ can be estimated as follows. Each agent A has a memory M containing information about other agents met (agents have unique identifiers associated with them). If a piece of that memory M_B contains information about some other agent B, then M_B^h will denote h_B at the time of the last conversation between A and B, and M_B^x will denote x_B at the same moment in time. Note that M_B^h at any given moment is not necessarily equal to the current h_B, because B might have changed its home position since the last time A and B communicated. Then A can estimate $|h_X|$ using the following formula:

$$|h_X| = \frac{1}{|M|}\sum_{i=1}^{|M|}\begin{cases}1, & \text{if } M_i^h \equiv h_X,\\ 0, & \text{otherwise.}\end{cases} \qquad (5)$$

Note that M_i^h and h_X should be defined with respect to the same reference frame, for example, with the local coordinate system associated with A. Thus, whenever B informs A about its home location h_B defined with respect to B's reference frame, A should transform this vector to A's coordinate system. We assume that A is able to perform this operation by using the information about the location of B with respect to A at the time of communication.

If during a conversation A and B discover that they work on different tasks, there is a chance that one of them will convert to the other's task. The probability of doing so $p(x_B \leftarrow x_A)$ is defined similar to that of converting to the other's home location, namely:

$$p(x_B \leftarrow x_A) = \min\left[1, \left(\frac{|x_B|}{|x_A| + l_x}\right)^2\right] \qquad (6)$$

where $0 \leq |x_X| \leq 1$ is the estimation of the number of other agents (excluding X) working on the same task as X, and l_x is a constant interpreted as *task loyalty*. Thus, $p(x_B \leftarrow x_A)$ increases with $|x_B|$ and decreases with $|x_A|$. For obvious reasons, upon switching to B's task, agent A will also have to deterministically switch to B's home location.

From Eq. 4 and Eq. 6 one can try to predict the dynamics of the reassignment of home locations and tasks. First of all, if there are originally N agents, each assigned to a unique task and a unique home position, then no conversions will occur, since $|h_B|$ and $|x_B|$ in those equations will always be zero. If, however, more than one agent are assigned to one task, then they will eventually "recruit" all other agents and converge to a single homogeneous cluster. If there are several groups of size more than 1 assigned to different tasks, all agents will still end up with the same task and the same home position as time continues indefinitely. We refer to the behavior of such agents as *conformity*, because it resembles the similar phenomenon in human society. The conformist behavior in our model can briefly be summarized as follows: "always follow the majority, both in terms of task and home location".

The idea behind conformity is *cooperation*. Agents should not pursue their own individual goals, which may interfere with each other; rather, they should work as a team. In this case, the team is an example of self-organization; the decision of where and what kind of clusters should be formed is collective and emergent. As experimental results show, conformity helps avoid conflicting goals, e.g., different home locations of the same object type. Conformity also helps improve the clustering performance of the objects of a given type x, because the number of agents involved in the process of clustering tends to increase up to N.

It is clear, however, that once agents have all converged to a single task, objects of other types will never become subject to sorting again. Thus, there must be a probability p_n for a conformist agent to give up its current task and home location and to switch its attention to objects of neglected types. This probability may be fixed. However, we suggest that it may be more reasonable to calculate it by estimating the *performance* u of the agent's work, e.g., how many useful actions the agent has accomplished within the last T_U steps. The only actions considered useful are picking up an object or putting it down. Random roaming in search for objects, direct routing to the home location, or random roaming in

search for empty cells to deposit an object are not considered useful actions. We could estimate the performance as follows:

$$u = \min\left(1, \frac{n_u}{N_U}\right) \tag{7}$$

where n_u is the number of useful actions accomplished within the last T_U time steps, and N_U is the required maximum number of useful actions, at which u is saturated at 1. For example, we may assume that accomplishing $N_U = 10$ useful actions within the last $T_U = 500$ time steps should be considered ideal performance. Note that large values of n_u with respect to T_U will hardly be ever achieved, because generally agents spend much more time roaming randomly than picking up or putting down objects.

Having estimated u, we can calculate p_n using the following equation:

$$p_n = \left(\frac{1-u}{1-u+c}\right)^2 \tag{8}$$

where c is a constant which we refer to as *conformity threshold*. Thus, p_n decreases with u from $\left(\frac{1}{1+c}\right)^2$ (when $u = 0$) to 0 (when $u = 1$).

If agent A has decided to give up its current task x_A at its current home location h_A, it starts random walk for a period of time (in our experiments, this period is equal to 300 iterations). Once 300 iterations have passed, the current location of A becomes its new home location. The behavior of A is then completely identical to the behavior of the agent that has just started the simulation: that is, A starts random walk, and the first object met defines A's new task x'_A. A then starts looking for x'-objects and brings them to h'_A.

Suppose that $x'_A \equiv x_A$ and $|h_A|$ is relatively large. Then it is likely that A will sooner or later encounter one of the agents still working at h_A. Since $|h_A| > |h'_A|$, A will be likely to convert back to h_A. This mechanism prevents agents from starting a new cluster of the same type to which the group has previously converged to.

Suppose, however, that $x'_A \not\equiv x_A$, and all agents but A are working on x_A. Then A upon encountering any of its former colleagues will be likely to convert back to x_A (because $|x_A| > |x'_A|$). To prevent this, we consider A a *nonconformist*. In our model, a nonconformist, i.e., an agent that has recently given up the task carried out by the majority and started executing a new task, gains a special "ability" that allows it not only to keep executing its new task, but also to recruit other agents.

To distinguish formally A from other agents that are working within groups, we define a measure of *nonconformity* ψ associated with each agent. $\psi_A = 1$, as a reflection of the fact that A does not conform to the majority. For any agent X that works within a group, $\psi_X = 0$. We now update Eq. 6 taking into consideration nonconformity ψ:

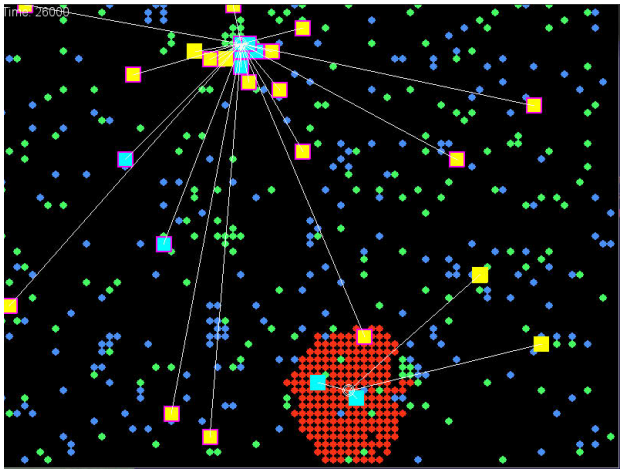


Figure 1: A screenshot from the simulation taken shortly after a nonconformist appeared. Agents are depicted as squares, and objects to be sorted are drawn as smaller circles. There are three types of objects – red, blue, and green pucks. Yellow squares correspond to unladen agents, and blue squares denote agents that are carrying pucks. A magenta border around an agent X indicates that $\psi_X > 0$. For each agent, a white line is drawn to show the distance from its home location. In this experiment, there are 600 pucks (200 pucks of each color), $N = 30$, and the grid size is 80×60.

$$p(x_B \leftarrow x_A) = \begin{cases} \min\left[1, \frac{\psi_B}{\psi_A + l_c}\right], \text{if } \psi_A > 0 \text{ or } \psi_B > 0 \\ \min\left[1, \left(\frac{|x_B|}{|x_A| + l_x}\right)^2\right], \text{if } \psi_A = \psi_B = 0 \end{cases} \tag{9}$$

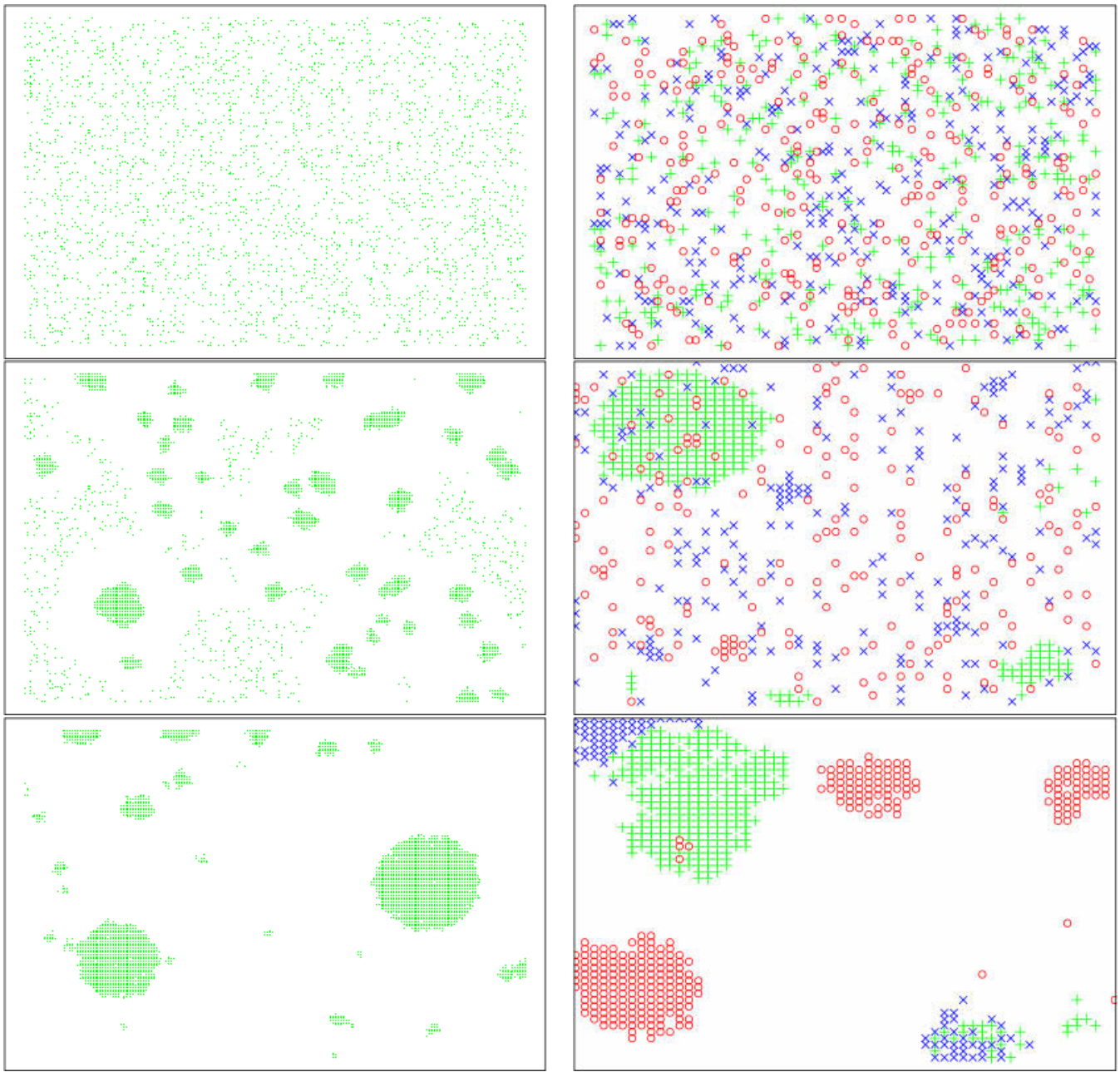
where l_c is a constant which can be referred to as *loyalty to crowd*.

Thus, any agent A with positive nonconformity will have a zero probability of joining a conformist B (since $\psi_B = 0$). Conformist B, however, will have a positive probability of joining nonconformist A: $p(x_A \leftarrow x_B) = \frac{\psi_A}{l_c} > 0$. If both A and B are conformists, Eq. 9 is reduced to Eq. 6.

The last detail is updating ψ. For conformists, ψ always remains zero. Once a conformist B has been recruited by a nonconformist A, it accepts its value of nonconformity: $\psi_B = \psi_A$. Furthermore, each nonconformist A decreases its nonconformity as the number of its colleagues increases:

$$\psi'_A = \begin{cases} 0, & \text{if } \psi_A = 0 \\ 1 - \min\left(1, \frac{|h_A|}{N_G}\right), & \text{otherwise} \end{cases} \tag{10}$$

where N_G is the maximum size of a group that can be recruited by a nonconformist. Upon reaching this limit, A



(a) (b)

Figure 2: **(a)**: Clustering of 3,000 objects of the same type after 1, 10,000, and 250,000 steps; grid 200×150. The size of the environment is 3-4 times larger than $r = 50$; two stable teams emerge, working at a significant distance from each other. Nonconformists tend to rejoin one of those teams shortly. **(b)**: Sorting of 900 objects of 3 types (300 objects of each type) after 1, 10,000 and 50,000 steps; grid 80×60. The relative magnitudes of the grid size and r allow the entire agent population work as a single team most of the time. First, a cluster of one type of objects is consistently formed. Then nonconformists start to appear, switching the entire group to new tasks. (Note that pucks being carried by agents are not displayed here.)

ceases its nonconformist status and starts acting like any other conformist agent. Note that A will not gain nonconformity again from its colleagues which may still be nonconformists (because of using slightly outdated estimations of $|h_A|$). Therefore, the entire group will eventually lose its nonconformist status and will become subject to honest conformist competition with other groups. If $N_G > \frac{N}{2}$, then, whenever a nonconformist is spawned, it will tend to create a new majority, switching the entire agent population to a new task.

As an illustration, Fig. 1 shows how the simulation of the sorting process looks like roughly one hundred iterations after a nonconformist appeared. A few important things can be noted. First, just before a nonconformist emerged, the entire population had been collecting red pucks to the cluster situated at the bottom. Since there are very few red pucks left to be collected (in this example, no isolated pucks are left, but often this is not the case), the agents have very low estimations of u, resulting in p_n large enough to create a nonconformist. Next, note that the nonconformist has already recruited many other agents which now have $\psi > 0$. According to Eq. 9, it is very likely that 4 agents that are still working on red pucks will shortly join the nonconformists. Interestingly enough, a couple of agents working in the "nonconformist" group have already stopped being nonconformists (ψ has become zero). This is because their $|h_X|$ values have surpassed the N_G threshold (see Eq. 10). Agents that are farther away from the nonconformist home location have less chance to communicate with their colleagues; thus, their $|h_X|$ increase more slowly, and they still consider themselves nonconformists. In 100-200 iterations, the entire population switches to green pucks, and all nonconformists become conformists again. Once almost all green pucks are collected, we can expect another nonconformist to appear.

The model described contains a number of parameters that can be tuned to achieve a desired balance between conformity and nonconformity. The next section offers experimental results showing how different values of some of those parameters affect the performance of the developed social model.

Experiments

Our experiments are conducted in a Monte-Carlo simulation which is functionally as close as possible to the simulation used by Deneubourg et. al. (Deneubourg et al., 1991). We use a grid-based environment and do not allow cells to be occupied by more than one object and one agent at the same time. Each iteration of the simulation, the agents are updated in random order. During the update cycle, each agent communicates with one random member within its communication range r and then performs an action depending on its current state and perception (e.g., moves one cell toward its home location, or picks up an object). After the communication session is completed, the agent is not allowed to communicate for the next $T_S = 5$ time steps (it continues, however, its sorting work). This is done to reduce computational costs of the simulation. In addition, whenever a conformist becomes a nonconformist, it is not allowed to communicate until it settles down at its newly generated position. This is because the new nonconformist does not yet know its own task – it will be determined based on the type of the first object it encounters after its new home location is found. Finally, if an agent tries to become a nonconformist by generating a random number and comparing it with the probability p_n and fails, its next chance to do so is scheduled in $T_N = \frac{T_U}{2}$ time steps. This is done to let n_u accumulate updates before firing p_n again. The general process of clustering and sorting can be seen in Fig. 2.

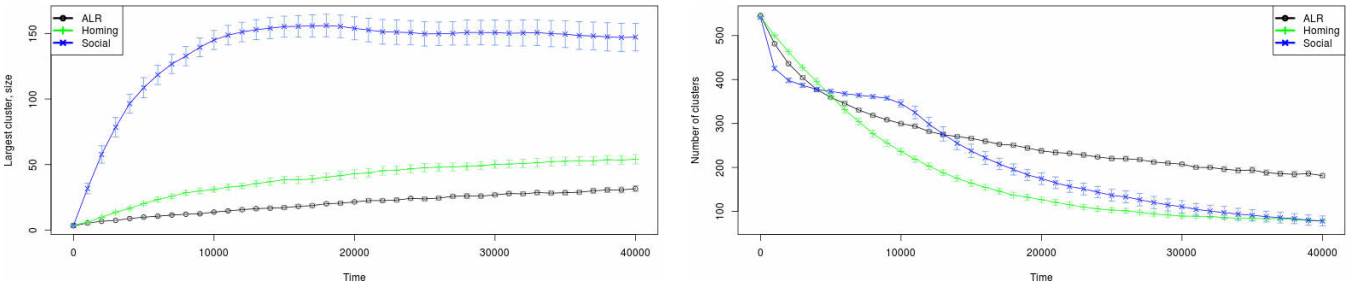
To assess the performance of the developed model, we collect two types of statistics: the size of the largest cluster and the number of clusters. We define clusters as follows: 1) An isolated object is a cluster of size 1; 2) An object q belongs to a cluster Q if q is of the same type as objects in Q and is located in an adjacent cell to any of the objects in Q.

Each experiment uses the following defaults: $N_U = 10$, $T_U = 500$, $N_G = \frac{N}{4}$, $N = 50$, $r = 40$, $l_h = l_x = 1$, $l_c = 0.1$, $c = 5$, grid: 80×60, 3 types of objects, 300 objects of each type. If for any of these parameters another value is used, it is explicitly stated so in the caption of the relevant figure. For each experiment, we ran 30 trials and averaged the results. All plots show mean values accompanied by error bars of ± 1.96 standard errors of the mean; thus, the error bars correspond to 95% confidence intervals of the mean. For each set of experiments, Shapiro-Wilk tests have been conducted which consistently produced p-values that are greater than 0.05; thus, there is no reason to reject the hypothesis that our experimental data are not normally distributed.

As it is clear from Fig. 3, the group of socially intelligent agents demonstrates better performance than ALR agents. We included homing agents without a social model into benchmarks in Fig. 3 as well, in order to understand what part of the performance boost acquired by social agents is actually due to their social abilities as opposed to the performance boost gained due to the homing mechanism.

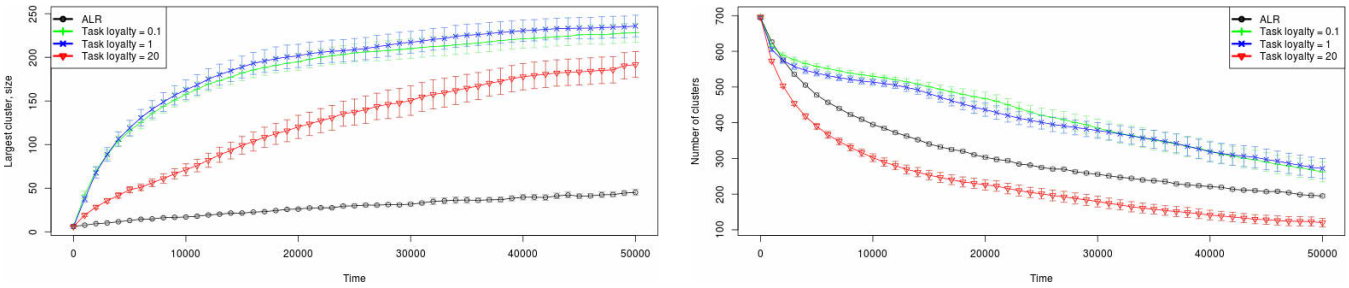
To estimate the influence of different parameters on the overall performance of the model, we conducted a series of experiments where we varied l_x, c, l_h, l_c, and r.

As one might expect, smaller values of l_x result in a relatively quick convergence of the population to one task; thus, the number of clusters decreases relatively slow (because objects of the other two types are consistently ignored), but the size of the largest cluster grows faster (Fig. 4). Note that at $t \approx 12,000$ the first nonconformists appear; conformists are recruited by nonconformists (see Eq. 9) and start form-



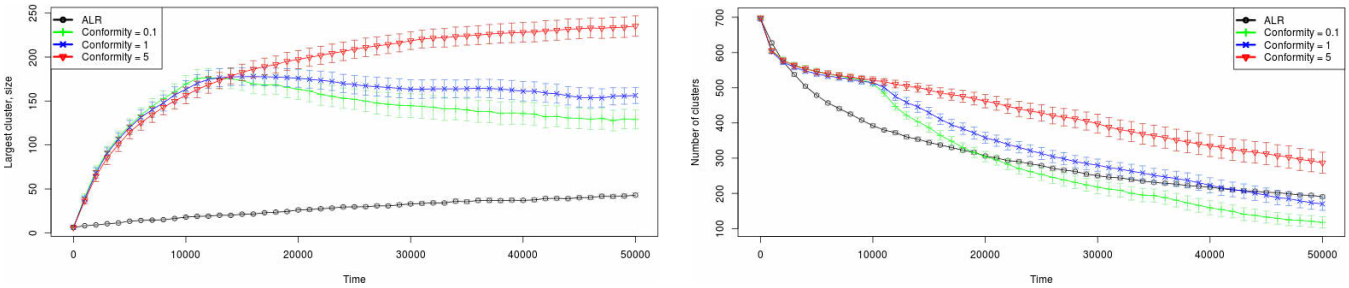
(a) The size of the largest cluster. (b) The number of clusters.

Figure 3: Experimental results for three modes: ALR agents, homing agents (no social model), and social agents; grid: 120×100.



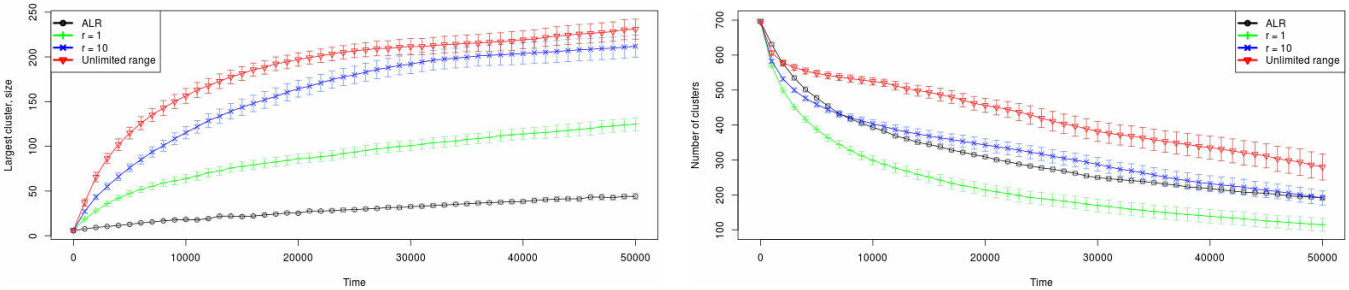
(a) The size of the largest cluster. (b) The number of clusters.

Figure 4: The influence of task loyalty l_x on the performance. Four modes: ALR agents, social agents with $l_x = 0.1$, $l_x = 1$, and $l_x = 20$.



(a) The size of the largest cluster. (b) The number of clusters.

Figure 5: The influence of conformity c on the performance. Four modes: ALR agents, social agents with $c = 0.1$, $c = 1.0$, and $c = 5.0$.



(a) The size of the largest cluster. (b) The number of clusters.

Figure 6: The influence of communication range r on the performance. Four modes: ALR agents, social agents with $r = 1$, $r = 10$, and $r = \infty$. The grid size, similarly to previous experiments, is 80×60.

ing larger groups, and, as a result, the number of clusters starts decreasing faster.

Conformity c prevents agents from defecting from the course of majority. Fig. 5 shows that with large values of conformity the entire population works steadily on one task as few or no nonconformists emerge. Thus, the size of the largest cluster increases, but the overall number of clusters declines relatively slowly. On the contrary, smaller values result in a situation when many nonconformists with different home locations appear, effectively dismantling the agent population. In this case, the largest cluster is gradually destroyed by competing nonconformists.

The influence of the l_c and l_h parameters is less significant; relevant figures are omitted due to lack of space.

In our final series of experiments, we estimated the influence of the communication range r (Fig. 6). Agents with limited r have less chance to encounter each other; therefore, establishing cooperation is rather unlikely. It is quite natural that such agents show performance similar to non-social homing agents.

Similarly to the trade-off between exploitation and exploration in genetic algorithms, there is a trade-off between conformity and nonconformity in the proposed social model. Conformity is vital for cooperation, avoiding conflicting goals, and convergence; nonconformity, on the other hand, is useful for exploring the task space in search for new goals. Small values of task loyalty l_x combined with large values of conformity c may be used to generate agents that tend to work collaboratively and consistently on creating one large cluster, ignoring other tasks. If it is more important to quickly decrease the number of clusters, large values of l_x and home loyalty l_h may be used. Note that the described social model can be reduced to the homing algorithm described on page 2 by assuming $l_x = \infty$, $l_h = \infty$, and $c = \infty$.

Conlusions

In this paper, we have proposed a model of conformity and nonconformity, a social phenomenon observed in human society. We tackled a well-known problem, collective distributed sorting. Our approach originated in the domain of swarm intelligence, but evolved into *socially intelligent* approach as the social awareness of agents increased. We provided evidence that using the ideas of conformity and nonconformity can be beneficial in artificial multi-agent systems and can increase performance of a task.

The algorithm described in this paper is extreme in the sense that our agents act like *zero-intelligent* particles (Bentley and Ormerod, 2011); that is, the decision to change the home location and/or the task being carried out is based purely on the number of other agents working at that home location and/or on that task. Our experimental results indicate that even this extreme model, based only on social information, is useful and can be applied in collective robotics. Further research could be conducted to reveal whether adding the social component of intelligence to the agents that are already capable of making informed decisions can increase the effectiveness of the robot group as a whole.

Acknowledgements

W.B. acknowledges funding from NSERC under the Discovery Grant Program RGPIN 283304-07 and RGPIN 283304-12.

References

Aronson, E. (2007). *The Social Animal*. W. H. Freeman and company, New York.

Bayindir, L. and Sahin, E. (2007). A review of studies in swarm robotics. *Turkish Journal of Electrical Engineering*, 15:115–147.

Beckers, R. and Holland, O. (1994). From local actions to global tasks: Stigmergy and collective robotics. *Artificial life*, IV:181–189.

Bentley, A. and Ormerod, P. (2011). Agents, intelligence, and social atoms. In Slingerland, E. and Collard, M., editors, *Creating Consilience: Integrating the Sciences and the Humanities*. Oxford University Press.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.

Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., and Chretien, L. (1991). The dynamics of collective sorting robot-like ants and ant-like robots. *Simulation of Adaptive Behaviour*, pages 356–363.

Grech, R., Florez-Revuelta, F., Monekosso, D. N., and Remagnino, P. (2012). Robot teams: Sharing visual memories. In *Distributed Autonomous Robotic Systems*.

Melhuish, C., H. O. and Hoddell, S. (1998). Collective sorting and segregation in robots with minimal sensing. In *Proceedings of the fifth international conference on simulation of adaptive behavior on From animals to animats*.

Sahin, E. (2005). Swarm robotics: From sources of inspiration to domains of application. *Swarm robotics*, 3342:10–20.

Seeley, T. D. (2010). *Honeybee democracy*. Princeton University Press.

Verret, S., Zhang, H., and Meng, M. (2004). Collective sorting with local communication. *Intelligent Robots and Systems*, 3:2687–2692.

Vorobyev, G., Vardy, A., and Banzhaf, W. (2012). Supervised learning in robotic swarms: From training samples to emergent behavior. In *Distributed Autonomous Robotic Systems*.

Wang, T. and Zhang, H. (2004). Collective sorting with multiple robots. *Robotics and Biomimetics*, pages 716–720.