Incremental Social Learning Applied to a Decentralized Decision-Making Mechanism: Collective Learning Made Faster

Marco A. Montes de Oca, Thomas Stützle, Mauro Birattari, and Marco Dorigo
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
Email: {mmontes, stuetzle, mbiro, mdorigo}@ulb.ac.be

***Abstract*—Positive feedback and a consensus-building procedure are the key elements of a self-organized decision-making mechanism that allows a population of agents to collectively determine which of two actions is the fastest to execute. Such a mechanism can be seen as a collective learning algorithm because even though individual agents do not directly compare the available alternatives, the population is able to select the action that takes less time to perform, thus potentially improving the efficiency of the system. However, when a large population is involved, the time required to reach consensus on one of the available choices may render impractical such a decision-making mechanism.**

In this paper, we tackle this problem by applying the incremental social learning approach, which consists of a growing population size coupled with a social learning mechanism. The obtained experimental results show that by using the incremental social learning approach, the collective learning process can be accelerated substantially. The conditions under which this is true are described.

***Keywords*-Incremental Social Learning; Self-Organization; Opinion Dynamics; Swarm Intelligence; Collective Learning.**

I. Introduction

Self-organization plays an important role in the life of social insects and other animals [1], [2]. Self-organization also plays an important role in the control of artificial swarm intelligence systems [3], [4]. One of the main reasons is that self-organization places no requirements on the intelligence level of the agents that form a "swarm". Nevertheless, even without intelligent agents, a swarm can still tackle classes of problems that are thought to require some level of intelligence to solve [5].

A class of problems that swarms are able to solve is the selection of the best choice from a set of available alternatives. For example, ant colonies are able to select the shortest path from their nest to a food source [6], or choose the best nest site from a set of available candidates [7]. Artificial swarms face similar choice problems. For instance, in ant colony optimization algorithms [8], good solutions to hard optimization problems are selected from a vast candidate solutions set, or in swarm robotics [9], a robot swarm needs to select collective actions that optimize its performance. In this paper, we focus our attention on this last class of choice problems. In particular, we study the simplest case, which is when a swarm of robots faces a binary choice problem. The choices are actions to execute while performing a task. For example, in an object transportation task, robots may need to choose between pulling or pushing objects. Each alternative action has a different average execution time associated with it. In our example, pulling may be faster than pushing because pushing may make a robot crash into obstacles due to a loss of visibility. In tasks with a time execution limit, it is desirable that the robot swarm chooses the action that is the fastest to execute.

In [10], we proposed a collective decision-making mechanism based on opinion dynamics models that allows a swarm of robots to choose the fastest of two alternative actions. This mechanism involves some characteristic elements of self-organizing systems [1], such as positive feedback, repeated interactions among robots, and amplification of fluctuations of the initial preferences of the robots. Additionally, this collective decision-making mechanism can be seen as a collective learning algorithm because robots do not directly compare the available alternatives, and thus do not learn individually which action is the fastest. Nevertheless, the swarm does select the fastest action. However, as the size of the swarm increases, so does the time needed for reaching consensus. For some applications, this phenomenon could effectively render the proposed approach impractical.

In this paper, we tackle the problem of slow convergence of the collective decision-making mechanism proposed in [10] when large swarms are involved. To this end, we apply the incremental social learning (ISL) framework [11], which consists of a growing population size and a social learning mechanism. The two components of the framework have the following functions: (i) starting from a small population, adding agents allows the system to converge faster than when a large population is used from the beginning of the learning process, and (ii) the social learning mechanism is used to transfer knowledge from experienced agents to naive ones in order to save exploration time. Our simulation results show that, by using the ISL framework, the collective learning process can be accelerated substantially, especially when a relatively small fraction of the swarm concurrently tries all the available choices. These performance improvements may allow one day the deployment of the system on real robots.

The rest of the paper is structured as follows. In Section II, we describe the decentralized decision-making mechanism that is the focus of this paper. The incremental social learning framework is described in Section III. In Section IV, we describe the methodology used to evaluate our proposal. Results are presented and discussed in Section V. A brief summary of related work is presented in Section VI. We present our conclusions and propose future work in Section VII.

II. A Decentralized Decision-Making Mechanism based on an Opinion Dynamics Model

We have recently proposed to use an opinion dynamics model as the core of a decentralized decision-making mechanism for swarms of robots [10]. Opinion dynamics is a branch of statistical physics that studies the processes of agreement in large populations of agents [12]. Opinion dynamics models are used to study large-scale social, economic, and natural phenomena that involve many interacting agents [12].

In this section, we describe the opinion dynamics model that we used, and how it becomes a decentralized decision-making mechanism when the opinions of the agents represent actions that take time to perform. The description that follows is a summary of what is presented and discussed in [10].

A. *Opinion Dynamics Model and Aggregation Rules*

Krapivsky and Redner [13] proposed a model that operates on a population of N agents, each of which can be in one of two possible states, called opinions. The system evolves as follows: A group (that we call *team*) is formed by sampling, randomly and without replacement, three agents from the population. The individual opinions of the team members are then aggregated by the majority rule (see Figure 1 (a)). The opinion of the majority becomes the opinion of all the team members. The team members are then put back in the population and a new team is formed. The process is repeated until all the agents in the population have the same opinion, that is, when a consensus has been reached.

In our work, we used Krapivsky and Redner's model using the majority and the expert rules as opinion aggregation rules (also called *decision rules*). With the expert rule, agents assume the opinion of a single agent, called expert (see Figure 1 (b)). The criterion that we used to choose the agent that plays the role of the expert is the overall experience, that is, the number of times an agent has performed an action (regardless of which action). The rationale is that an agent with a greater experience is more likely to have executed more often the fastest-to-execute action, thus it is reasonable to adopt his opinion. When there is no expert in a team, the majority rule is applied.

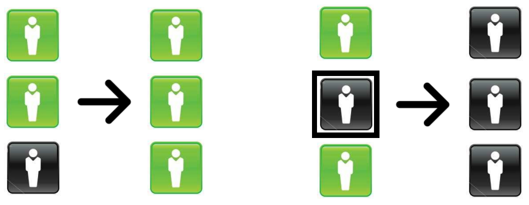


Figure 1: Example application of the majority and the expert rules on a group of three agents with different opinions (represented by different color shades). Figure (a) shows the outcome of the majority rule: team members adopt the opinion of the majority. Figure (b) shows the outcome of the expert rule: team members adopt the opinion of the local expert, which is indicated by a frame around it.

B. *Opinions as Actions*

When the opinions of the agents represent actions that they can perform, the opinion dynamics model described above can be used as the basis of a decentralized decision-making mechanism.

In swarm robotics, a robot's opinion could be interpreted as one or more actions. Examples of an opinion representing a single action are whether to turn left or right at some point in an environment, or whether to connect or not with another robot. Examples of an opinion representing more than one action could be whether to follow the rules for moving with other robots in a formation, or whether to follow the rules for assembling one shape or another for a specific task. Two inherent properties of swarm robotics systems that enable self-organized decision-making are: (i) actions take time to perform, and (ii) several teams can execute actions in parallel. To simulate these properties, we model the time required to execute an action as a random variable, and parallelism is simulated by creating k teams instead of just one as in the original model. Actions are executed immediately after the members of a team adopt the opinion that results from the application of an opinion aggregation rule. During execution, robots cannot participate in the formation of a new team, and thus cannot change opinion. When a team finishes executing an action, its members become available again to form another team. The process continues until the population reaches consensus, the time allocated for the task is over, or the demand for the task ceases to exist.

Figure 2 shows an example of the process just described. In the example, the majority rule is used on a population of 8 agents with 2 groups of 3 agents each. Note how the population changes from a heterogeneous opinion state to a homogeneous one that corresponds to the fastest-to-execute action.

In [10], we showed that the dynamics of the system

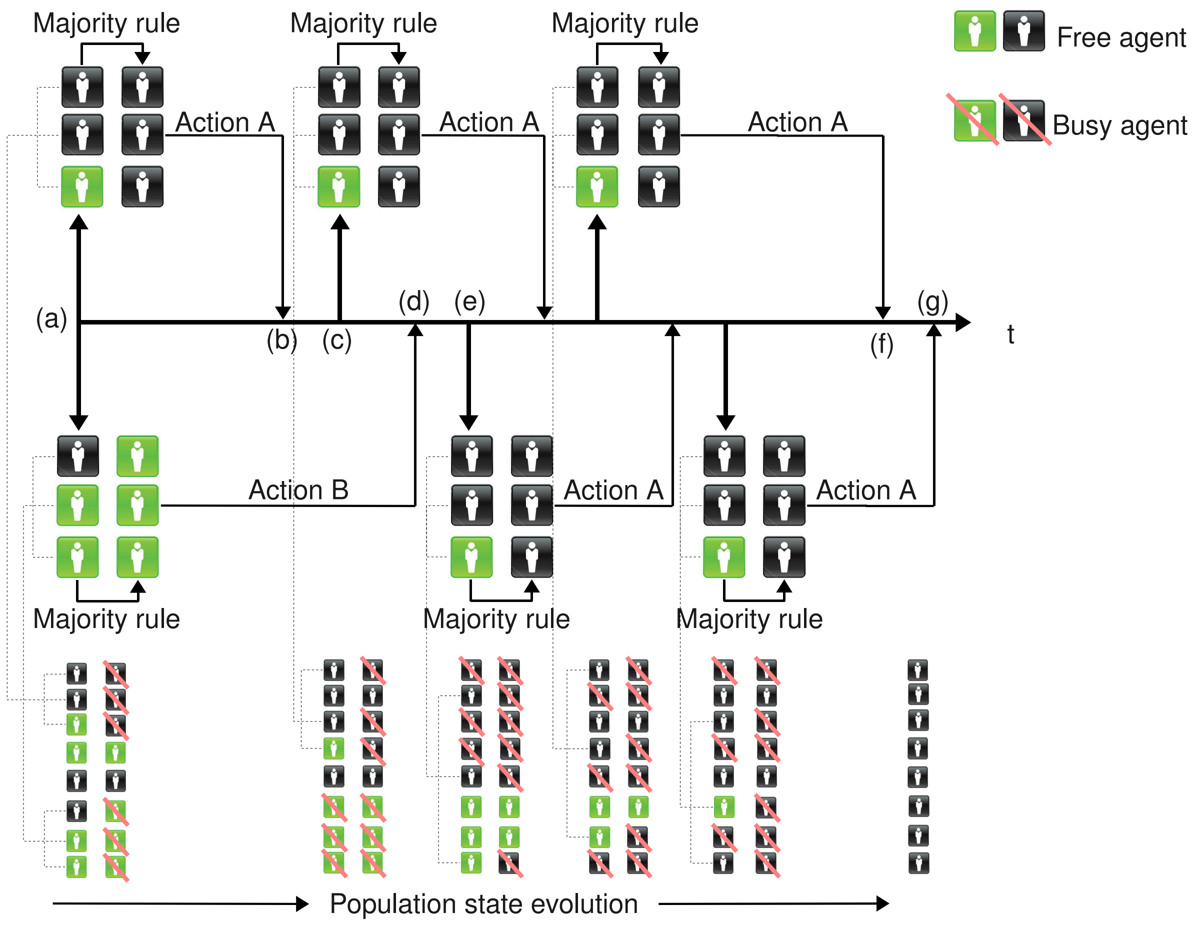


Figure 2: Example of the dynamics induced by the majority rule on a population of 8 agents with 2 groups of 3 agents each. Teams choose and execute actions. The opinion represented in black is associated with action A, while the opinion represented in light color is associated with action B. The two actions have a different execution time (action A is faster to execute than action B, on average). In the bottom part of the figure, one can see the evolution of the population state over time. First, in point (a), two teams of 3 agents each are formed at random and the majority rule is applied on each one of them. Each team then executes the selected action. In point (b), a team finishes. In point (c) a new team is formed from the set of free agents (busy agents cannot be considered when the selection occurs). The time it takes to form a team is represented by the distance between points (b) and (c). After the application of the majority rule, the team performs the agreed action (action A, in this case). In point (d), the other team finishes executing an action and a new team is formed (point (e)). The decision rule is applied once more to decide which action to perform (action A, again). The opinion dynamics process continues until the population reaches consensus. This figure is a slight variation of the one published in [10].

makes the population reach consensus on the fastest-to-execute action. Positive feedback is responsible for this result: Agents that choose the fast action finish before agents that choose the slow action. Consequently, the rate at which agents that choose the fast action spread their opinion is higher than the rate at which agents that choose the slow action do.

Reaching consensus on the fastest-to-execute action is a necessary but not sufficient condition to maximize the swarm's performance. The robot swarm should also reach consensus as fast as possible. Unfortunately, we observed that with the aforementioned mechanism, the time necessary for the swarm to reach consensus increases with the size of the population if the number of teams concurrently executing actions remains constant. Such a situation would not be rare in environments that can hold only a certain number of teams executing a task in parallel (e.g., when the robots must travel through a corridor).

III. Incremental Social Learning

When multiple simultaneously adapting agents coexist, learning is difficult. One of the reasons is interference, which occurs when agents adapt their behavior in response to the observed behavior of other agents, who are in turn changing their behavior as well. Thus, interference slows down learning and hinders scalability.

The incremental social learning (ISL) [11] framework tackles the problem of interference in systems composed of multiple learning agents. ISL consists of two elements: (i) an initially small population of agents that grows over time, and (ii) a social learning mechanism upon agent addition. A small population of agents learns faster than a large one because of the reduced interference. Agents are added to the population according to some predefined criterion. An agent that is added to the population learns socially from those that have been in the population for some time. This element of ISL is attractive because new agents acquire knowledge from more experienced ones without incurring the costs of acquiring that knowledge individually. Thus, ISL allows the new agents to save time that they can use to perform other tasks. After the inclusion of a new agent, the population needs to re-adapt to the new conditions, but the agents that are part of it do not need to learn everything from scratch.

The algorithmic structure of the incremental social learning framework is outlined in Algorithm 1, where the environment and the population of agents are initialized before the main loop begins. While no agents are to be added to the current population, the agents that belong to it learn individually, or through some other mechanism (labeled "default") which may include elements of social or centralized learning. When the agent addition schedule dictates that a new agent should join the current population, the new agent first learns socially from a subset of the already experienced agents. The agent addition schedule controls the rate at which agents are added to the population. It also creates time delays that allow the agents in the population to learn from the interaction with the environment and with other agents. In Algorithm 1, the environment is updated explicitly in order to stress the fact that the environment might change over time (although it does not need to be so). In a real implementation, the environment can change at any time and not necessarily at the end of a training round.

The actual implementation of the individual (or default) and social learning mechanisms is independent of the incremental social learning framework outlined above. Both generic or application-specific mechanisms may be used. In this paper, the default learning mechanism is the one described in the previous section, whereby robots collectively learn which action improves the performance of the system. The social learning mechanism that we used is detailed in Section IV-D.

Algorithm 1 Incremental social learning

```
/* Initialization */
$t \leftarrow 0$
Initialize environment $\mathbf{E}^t$
Initialize population of agents $\mathbf{X}^t$

/* Main loop */
while Stopping criteria not met do
  /* Agents are added according to a schedule */
  if Agent addition criterion is not met then
    $\mathbf{X}^{t+1} \leftarrow$ ilearn$(\mathbf{X}^t, \mathbf{E}^t)$ /* Individual or default learning mechanism */
  else
    Create new agent $a_{new}$
    slearn$(a_{new}, \mathbf{X}^t)$ /* Social learning mechanism */
    $\mathbf{X}^{t+1} \leftarrow \mathbf{X}^t \cup \{a_{new}\}$
  end if
  $\mathbf{E}^{t+1} \leftarrow$ update$(\mathbf{E}^t)$ /* Update environment */
  $t \leftarrow t + 1$
end while
```

IV. Evaluation

In this section, we describe the methodology used to evaluate the effectiveness of ISL when used with the decision-making mechanism described in Section II. We also describe the simulation environment used as well as the experimental setup. In Section IV-D we explain how ISL was implemented.

A. Methodology

The goals of our evaluation are: (i) to determine whether ISL improves the performance of the decentralized decision-making mechanism described in Section II, and if improvement is indeed achieved, (ii) to measure the magnitude of the improvement and to determine the conditions under which such an improvement occurs.

As mentioned in Section II, the performance of the system can be measured as the number of times an action is performed in a given amount of time. We adopt this measure of performance in this paper. With it, we put emphasis on the amount of useful work performed by the system. Given two system settings, the one that lets agents execute more actions in the same amount of time is preferred. Additionally, we also look at the average number of times each agent in the population executes each of the two available actions. This measure allows us to observe whether ISL reduces the time agents spend trying the available alternative actions as discussed in Section III.

B. Simulation Environment

We used Monte Carlo simulation to carry out our experiments. The execution times of the fast and slow actions were modeled as two normally distributed random variables with means μ_{fast} and μ_{slow}, and standard deviations σ_{fast} and σ_{slow}, respectively. We study the system's behavior as a function of the action execution time ratio $r = \mu_{slow}/\mu_{fast}$. Different action execution time ratios were obtained by varying μ_{slow}. The standard deviations σ_{fast} and σ_{slow} were kept constant.

Simulation proceeds as follows. Teams are formed at random, the opinion aggregation rules are applied to each team, and the resulting opinions are adopted by the involved agents. The execution times for each team are drawn from a normal distribution with the appropriate parameters and the resulting number is rounded to the nearest integer. The time steps counter runs until a team's execution time expires. At that point, a new team is formed and the process is repeated until the maximum number of time steps is reached.

C. *Experimental Setup*

We ran simulations with two different maximum population sizes ($N \in \{100, 1000\}$). Since a team is composed of three agents, different numbers of teams for each population size were used ($k_{100} \in \{2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30\}$, and $k_{1000} \in \{2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, \ldots, 90, 100, 200, 300\}$). The initial opinion of each agent was set at random. Three different execution time ratios were tried ($r \in \{2, 4, 8\}$). The means and standard deviations of the action execution times were set as follows, $\mu_{fast} = 10$, $\mu_{slow} \in \{20, 40, 80\}$, and $\sigma_{fast} = \sigma_{slow} = 2$. This value was chosen in order to allow a clear separation of execution times between the alternative actions. Each simulation was run for 10,000 time steps. 500 simulations were run for each combination of parameters.

D. *ISL implementation*

In our implementation, we start with a population of size $N = 6$, which means that we start with $k = 2$ teams. The reason for this choice is that the system needs at least two teams to execute actions concurrently, so that an execution time difference, if it exists, can be detected. One team would make the population converge, as demonstrated by Krapivsky and Redner [13], but the consensus opinion would be essentially random with our setup. The agent addition schedule used is the fastest possible, that is, we add an agent to the population every time step until the maximum population size is reached. With this schedule, by the time the first team finishes executing an action, new agents will be ready to form a different team. If the number of teams to build is greater than two, a new team is created as soon as there are enough free agents. Once the maximum number of teams is reached, no new teams are created even if the population is still growing.

The second element of ISL is social learning when new agents are created. In our experiments, when a new agent is added to the population, its initial opinion is copied from one random agent chosen from the set of free agents, that is, the agents that are not engaged in an action execution. If such agent does not exist, for example, when all agent are active, the new agent is initialized at random.

V. Results

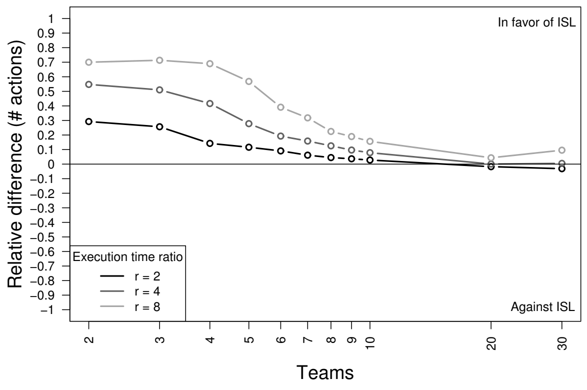
The results of our simulations are reported in this section. First, we look at the relative difference of executed actions after a given number of time steps between the ISL-based and the constant population size systems. We then look at the exploration time savings due to the use of ISL.

A. *Number of executed actions*

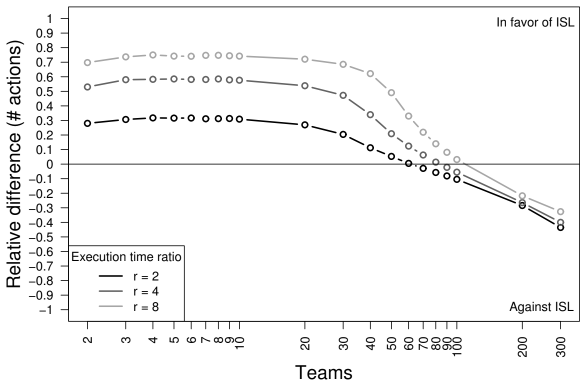
The relative difference of the median number of executed actions between ISL-based and constant population size systems per number of teams using the majority and expert rules are shown in Figures 3 and 4, respectively. The normalizing factor used is the expected number of executed actions if the fastest-to-execute action was chosen from the beginning of the simulation. This number is estimated as $k \cdot T / \mu_{fast}$, where k is the number of teams simultaneously exploring the space of alternatives, T is the maximum number of time steps (in our case $T = 10000$), and μ_{fast} is the mean execution time of the fastest-to-execute action. A positive difference indicates that the difference is in favor of ISL-based systems. If a constant population size system produces more action executions than its ISL-based counterpart, the difference is negative.

We analyze the results along the following influencing factors:

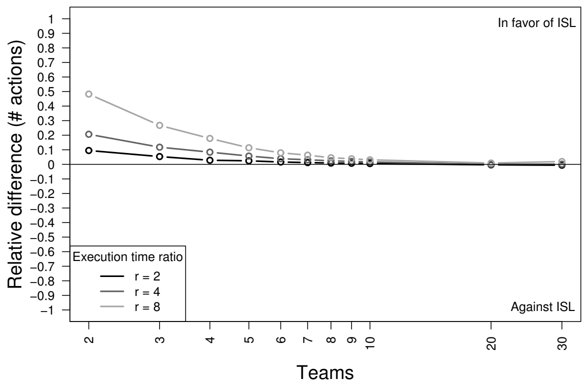
- **Action execution time ratio.** A general trend is that the greater the action execution time ratio, the stronger is the effect of ISL on the performance of the system. This phenomenon may be due to the small population size with which the system begins. Contrary to what would happen with a constant population size system where many teams would choose to execute the slowest action, with ISL only one team (on average) would execute the slow action at the beginning. If the action execution time ratio is large, that team will not have many chances to influence other agents once it finishes. The result is an accelerated convergence towards a consensus on the fastest-to-execute action.
- **No. of active teams.** The effects of ISL diminish as the number of active teams increases. In fact, the differences due to changes in the value of the action execution time ratio disappear when many teams are active in the environment. Clearly, with large populations and many active teams, a small population cannot execute as many actions as a large number of teams working in parallel even if many of them execute the slowest action.
- **Maximum population size.** The effects of ISL increase as the size of the population increases. The reason is that small populations converge rapidly as a result of the rapid amplification of fluctuations in the opinions of the population due to team formations. For example, if $N = 10$, a single team can alter the opinion of $1/10$



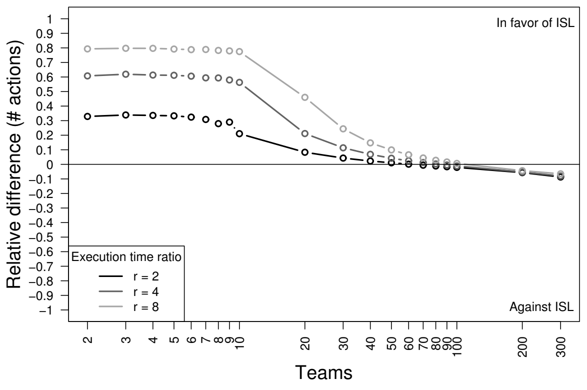
(a) Majority rule: N =100 after 1,000 time steps



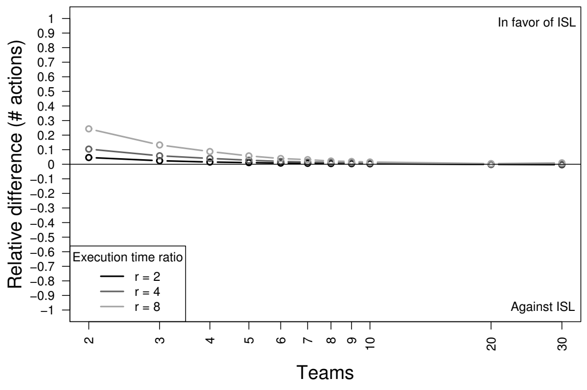
(b) Majority rule: N =1,000 after 1,000 time steps



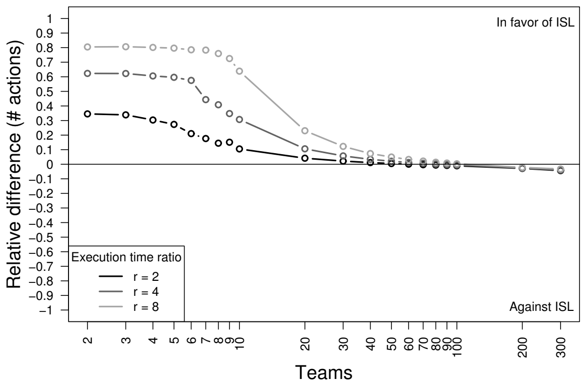
(c) Majority rule: N =100 after 5,000 time steps



(d) Majority rule: N =1,000 after 5,000 time steps



(e) Majority rule: N =100 after 10,000 time steps

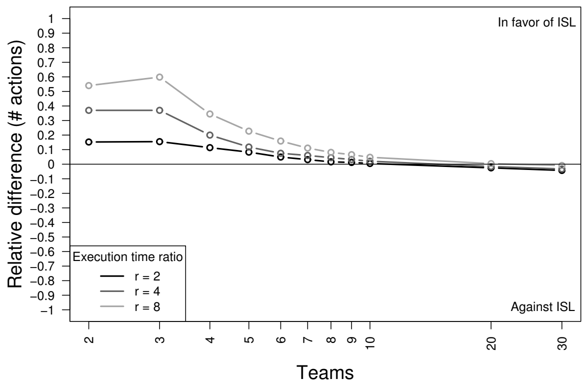


(f) Majority rule: N =1,000 after 10,000 time steps

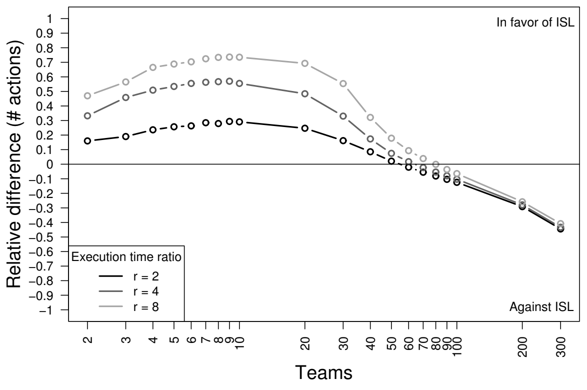
Figure 3: The relative difference of the median number of executed actions between ISL-based and constant population size systems per number of teams. The normalizing factor used is the expected number of executed actions if the fastest-to-execute action was chosen from the beginning. These results were obtained using the majority rule as the opinion aggregation rule.

of the population, while if $N = 1000$ a team can only alter the opinion of $1/1000$ of the population.

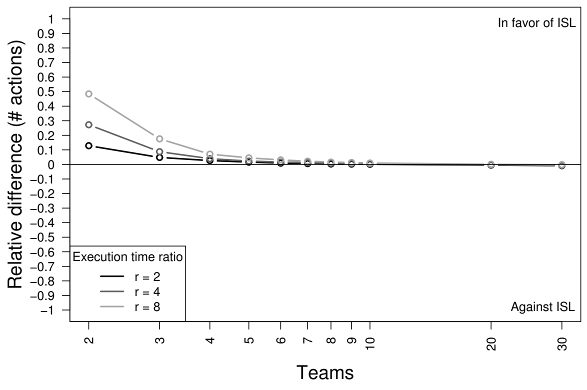
- **Available time.** The accelerated convergence that results from the application of ISL proves more useful if time constraints exist. This is true even with not-so-large populations and a relatively large number of active teams.
- **Aggregation rule.** The main trends described above do



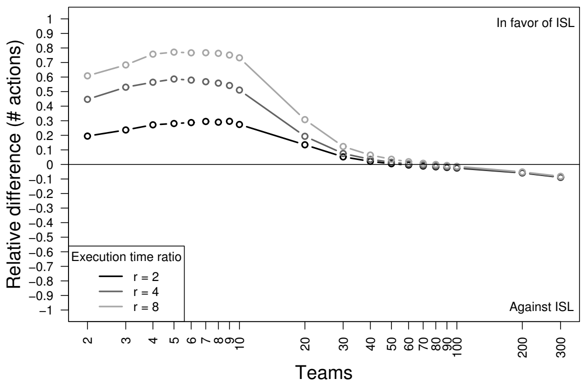
(a) Expert rule: N =100 after 1,000 time steps



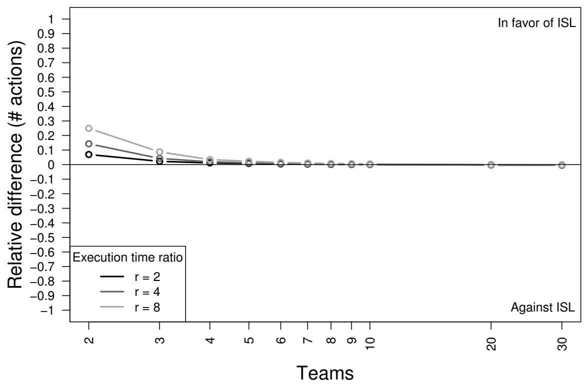
(b) Expert rule: N =1,000 after 1,000 time steps



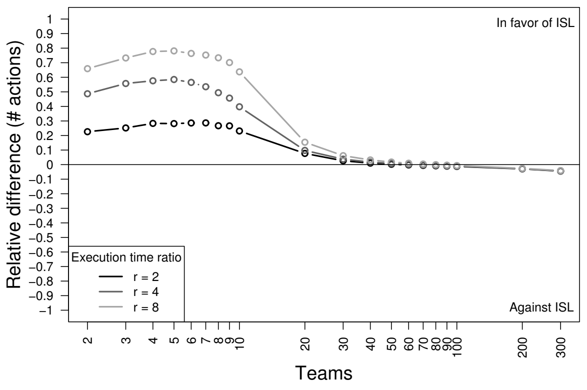
(c) Expert rule: N =100 after 5,000 time steps



(d) Expert rule: N =1,000 after 5,000 time steps



(e) Expert rule: N =100 after 10,000 time steps



(f) Expert rule: N =1000 after 10,000 time steps

Figure 4: The relative difference of the median number of executed actions between ISL-based and constant population size systems per number of teams. The normalizing factor used is the expected number of executed actions if the fastest-to-execute action was chosen from the beginning. These results were obtained using the expert rule as the opinion aggregation rule.

not change when the opinion aggregation rule changes. This may be due to a reduced likelihood of finding experts in teams.

B. Exploration time

It is usually assumed that social learning allows agents that practice it to save time that otherwise they would spend learning things by themselves [14]. In this way, agents can spend more time performing the actions that seem more rewarding.

To see whether ISL allows agents to save the time otherwise needed to try the different available actions (that is, to learn individually), we proceeded as follows. During each simulation run we counted the number of times each agent executed each available action. The sum of these "experiences" at the end of the simulation was then divided by the maximum population size. The resulting quantities were the average individual experience on each action. The difference between these quantities served as a measure of the balance between exploration and exploitation.

To have a direct comparison between ISL-based systems and constant population size systems, we computed the ratio of the median average individual experiences for each action and for each execution time ratio. The results are shown in Figure 5.

ISL reduces the time spent by agents exploring the space of alternatives. The actual reduction depends on a number of factors including the maximum population size, the number of active teams in the environment, the action execution time ratio, and the opinion aggregation rule used. In some cases, the reduction is substantial. For example, with a constant population size of 1000 agents using the expert rule and 10 active teams, agents execute 100 times more the slowest action than agents in an ISL-based system. However, as the number of active teams increases the advantage of using ISL is reduced.

VI. Related Work

We have shown that ISL can enhance the performance of a system in which agents search collectively for the best action in a set of alternatives. This is a task that the system presented in this paper shares with optimization algorithms. The difference, of course, is in the nature of the search space. Thus, it is reasonable to apply ISL for enhancing the performance of population-based optimization algorithms. In fact, ISL has been applied to a family of swarm-based optimization algorithms, namely, the particle swarm optimization algorithm [15], [16], [17], and encouraging results have been obtained [18], [19].

Regarding the decentralized decision-making mechanism described in Section II, related work is the one based on the simulation of the pheromone-laying and pheromone-following behavior of some ant species [6]. For example, in ant colony optimization algorithms [8], pheromones are numbers associated with components of solutions to a combinatorial optimization problem. In robotics, pheromones have been simulated with chemical substances [20], with images projected on the ground [21], [22], with RFID tags [23], with message-relay devices or beacons [24], [25], [26], and with other techniques [27].

Recently, Parker and Zhang tackled the problem of selecting the best of a set of alternatives with robots [28]. In their work, robots need to know whether there is a sufficient number of robots in favor of one alternative before committing to it. This is done through a quorum test that depends on a parameter that the designer needs to set before deployment. This is a critical issue because the first alternative that is identified as dominant through the quorum test will be the alternative chosen by the swarm. In our work, the collective decision is the result of self-organization.

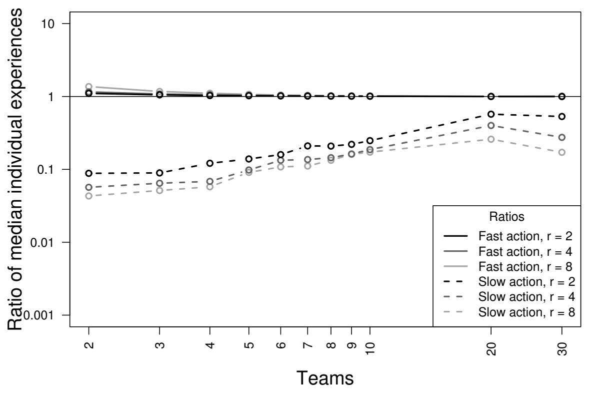
The work of Wessnitzer and Melhuish [29] is also related to ours. In both works the majority rule is used for breaking the symmetry of the decision problem (although they use it in a completely different setting). However, we go a step further by considering the effects of implicit time costs in the robots' actions.

VII. Conclusions and Future Work

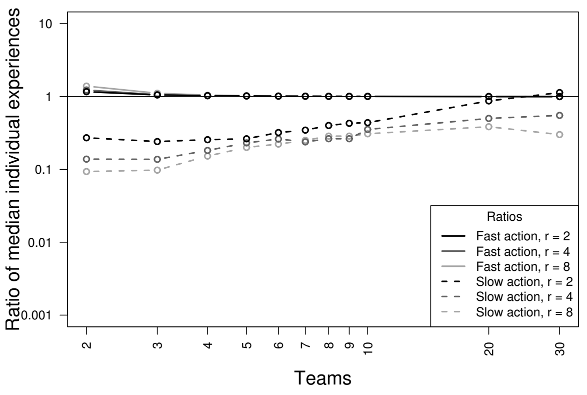
To design intelligent systems composed of many unintelligent agents is one of the main goals of people working in the swarm intelligence field. The main approach to meet this goal has been to use self-organization principles, so far producing many successful systems. However, in addition to being functional, swarm intelligence systems must be efficient if they are to be used for solving practical problems.

In this paper, a decentralized decision-making mechanism based on an opinion dynamics model was used as subject of study. Such a mechanism allows a large population of agents to reach consensus on one of two possible choices. If these choices are actions that take time to execute, the population reaches consensus on the fastest to execute. However, when large populations are involved, the time necessary for the system to converge may make impractical to use it in some applications. We tackle this problem by applying the incremental social learning (ISL) framework. This framework consists of a growing population size and a social learning rule. By starting with a small population and increasing its size over time, the underlying learning algorithm converges faster. The social learning rule allows new agents to learn from more experienced ones, thus saving exploration time. Our simulation results show that through the application of ISL, the performance of the decision-making mechanism can be substantially improved in situations where a small fraction of the population concurrently tries the different available alternatives and when time constraints exist. This result is very positive because in many situations reducing the number of active agents without sacrificing the amount of work performed may allow the spared agents to perform other tasks.

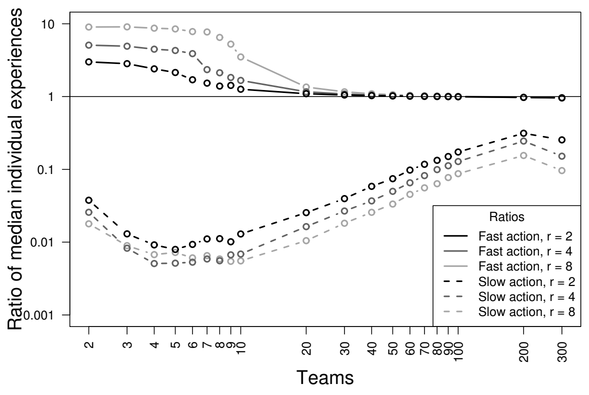
Many possibilities exist for future research. Related to the decision-making mechanism, strategies for allowing the system to work on more than two alternatives should be explored. Also of interest is to know whether the system could be extended to handle sequences of actions with



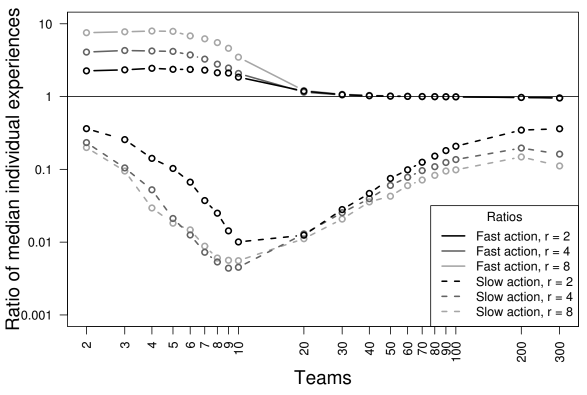
(a) Majority rule: N =100 after 10,000 time steps



(b) Expert rule: N =100 after 10,000 time steps



(c) Majority rule: N =1,000 after 10,000 time steps



(d) Expert rule: N =1,000 after 10,000 time steps

Figure 5: Exploration *vs.* exploitation behavior. Each plot shows the ratio between the average individual experience obtained with ISL and the average individual experience obtained with a constant population size. Figures (a) and (c) show the results obtained using the majority rule. Figures (b) and (d) show the results obtained using the expert rule. The size of the gap between the ratios for the fast and slow actions is a measure of the time agents saved exploring the space of alternatives thanks to ISL. The solid line at ratio 1 represents the same measure but for constant population size systems.

interdependence between them. In the long term, the goal is to design a self-organized mechanism that works with multiple, sequential and conditional choices. Regarding ISL, it is still to be defined what the features of a learning problem/algorithm should be in order for ISL to speed up convergence. Lastly, the decentralized decision-making mechanism, and its combination with ISL should be tested on a real swarm robotics system.

Acknowledgment

This work was supported by the SWARMANOID project, funded by the Future and Emerging Technologies programme (IST-FET) of the European Commission, under grant IST-022888, and by the VIRTUAL SWARMANOID project funded by the Fund for Scientific Research F.R.S.-FNRS of Belgium's French Community. The information provided is the sole responsibility of the authors and does not reflect the European Commission's opinion. The European Commission is not responsible for any use that might be made of data appear- ing in this publication. Thomas Stützle, Mauro Birattari, and Marco Dorigo acknowledge support from the F.R.S.-FNRS of Belgium's French Community.

References

[1] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*. Princeton, NJ: Princeton University Press, 2003.

[2] I. D. Couzin and J. Krause, "Self-organization and collective behavior in vertebrates," *Advances in the Study of Behavior*, vol. 32, pp. 1–75, 2003.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. New York: Oxford University Press, 1999.

[4] M. Dorigo and M. Birattari, "Swarm intelligence," *Scholarpedia*, vol. 2, no. 9, p. 1462, 2007, http://www.scholarpedia.org/article/Swarm_intelligence.

[5] G. Beni, "From swarm intelligence to swarm robotics," in *Swarm Robotics. SAB 2004 International Workshop*, E. Şahin and W. M. Spears, Eds. Berlin, Germany: Springer, 2005, pp. 1–9.

[6] S. Goss, S. Aron, J.-L. Deneubourg, and J. M. Pasteels, "Self-organized shortcuts in the argentine ant," *Naturwissenschaften*, vol. 76, no. 12, pp. 579–581, 1989.

[7] N. R. Franks, E. B. Mallon, H. E. Bray, M. J. Hamilton, and T. C. Mischler, "Strategies for choosing between alternatives with different attributes: exemplified by house-hunting ants," *Animal Behavior*, vol. 65, no. 1, pp. 215–223, 2003.

[8] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA: MIT Press, 2004.

[9] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *Swarm Robotics. SAB 2004 International Workshop*, E. Şahin and W. M. Spears, Eds. Berlin, Germany: Springer, 2005, pp. 10–20.

[10] M. A. Montes de Oca, E. Ferrante, N. Mathews, M. Birattari, and M. Dorigo, "Opinion dynamics for decentralized decision-making in a robot swarm," in *LNCS 6234. Proceedings of the International Conference on Swarm Intelligence (ANTS 2010)*, M. Dorigo *et al.*, Eds. Berlin, Germany: Springer, 2010, pp. 252–263.

[11] M. A. Montes de Oca and T. Stützle, "Towards incremental social learning in optimization and multiagent systems," in *Workshop on Evolutionary Computation and Multiagent Systems Simulation of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, W. Rand *et al.*, Eds. New York: ACM Press, 2008, pp. 1939–1944.

[12] C. Castellano, S. Fortunato, and V. Loreto, "Statistical physics of social dynamics," *Reviews of Modern Physics*, vol. 81, no. 2, pp. 591–646, 2009.

[13] P. L. Krapivsky and S. Redner, "Dynamics of majority rule in two-state interacting spin systems," *Physical Review Letters*, vol. 90, no. 23, pp. 238 701.1–238 701.4, 2003.

[14] K. N. Laland, "Social learning strategies," *Learning & Behavior*, vol. 32, no. 1, pp. 4–14, 2004.

[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*. Piscataway, NJ: IEEE Press, 1995, pp. 1942–1948.

[16] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization. An overview," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[17] M. Dorigo, M. A. Montes de Oca, and A. P. Engelbrecht, "Particle swarm optimization," *Scholarpedia*, vol. 3, no. 11, p. 1486, 2008, http://www.scholarpedia.org/article/Particle_swarm_optimization.

[18] M. A. Montes de Oca, K. Van den Enden, and T. Stützle, "Incremental particle swarm-guided local search for continuous optimization," in *LNCS 5296. Proceedings of the International Workshop on Hybrid Metaheuristics (HM 2008)*, M. J. Blesa *et al.*, Eds. Berlin: Springer, 2008, pp. 72–86.

[19] M. A. Montes de Oca, T. Stützle, K. Van den Enden, and M. Dorigo, "Incremental social learning in particle swarms," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2010, forthcoming. Preliminary version available at http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2009-002r003.pdf.

[20] R. A. Russell, "Ant trails – An example for robots to follow?" in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1999)*. Piscataway, NJ: IEEE Press, 1999, pp. 2968–2703.

[21] K. Sugawara, T. Kazama, and T. Watanabe, "Foraging behavior of interacting robots with virtual pheromone," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*. Piscataway, NJ: IEEE Press, 2004, pp. 3074–3079.

[22] S. Garnier, F. Tache, M. Combe, A. Grimal, and G. Theraulaz, "Alice in pheromone land: An experimental setup for the study of ant-like robots," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS 2007)*. Piscataway, NJ: IEEE Press, 2007, pp. 37–44.

[23] M. Mamei and F. Zambonelli, "Physical deployment of digital pheromones through RFID technology," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*. New York: ACM Press, 2005, pp. 1353–1354.

[24] B. Werger and M. Matarić, "Robotic "food" chains: Externalization of state and program for minimal-agent foraging," in *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats (SAB 1996)*. Cambridge, MA: MIT Press, 1996, pp. 625–634.

[25] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone robotics," *Autonomous Robots*, vol. 11, no. 3, pp. 319–324, 2001.

[26] S. Nouyan, R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Teamwork in self-organized robot colonies," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 695–711, 2009.

[27] F. Ducatelle, G. Di Caro, and L. M. Gambardella, "Cooperative self-organization in a heterogeneous swarm robotic system," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010)*, M. Pelikan and J. Branke, Eds. New York: ACM Press, 2010, pp. 87–94.

[28] C. A. C. Parker and H. Zhang, "Cooperative decision-making in decentralized multiple-robot systems: The best-of-N problem," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 240–251, 2009.

[29] J. Wessnitzer and C. Melhuish, "Collective decision-making and behaviour transitions in distributed ad hoc wireless networks of mobile robots: Target-hunting," in *LNCS 2801. Proceedings of the European Conference on Artificial Life (ECAL 2003)*. Berlin, Germany: Springer, 2003, pp. 893–902.