Spatial Clustering by Schelling's Ants

László Gulyás[0000−0002−6367−6695]

ELTE Eötvös Loránd University, Budapest, Hungary
Faculty of Informatics, Institute for Industry-Academy Innovation
Department of Artificial Intelligence
lgulyas@inf.elte.hu

Abstract. In this paper we revisit the distributed, collective spatial clustering algorithm motivated by ants and point out its fundamental similarity to one of the most cited and earliest agent-based models. Based on this observation, we propose a variant of the algorithm and analyze its behavior and performance.

1 Introduction

Distributed collective intelligence is concerned with the joint problem solving behavior of large groups of autonomous agents, typically with relatively simple capabilities. The last decade of the previous millennium has seen a surge of interest in the observation of the surprisingly rich and efficient problem solving behavior of various species of social insects and in the application of bio-inspired algorithms resulting from these observations to computational problems. These early studies were the precursor of the research domain now known as *swarm intelligence*. Swarm Intelligence (SI) is an active field of research today with a growing and wide variety of proposed methods and algorithms. [1] [2]

Interestingly, however, the algorithms discussed in SI today seem all to have emerged from the foraging behavior of ant colonies, despite the fact that in the 1990s several other bio-inspired methods were observed and discussed, many of them applicable to tasks and problem domains vastly different from the problems addressed by SI today. For example, the flexible and demand driven task differentiation of wasp colonies is a robust method for dynamic task allocation [3], whereas bees have an efficient way to reach consensus about the location of the next hive, optimising the quality of the location, without a complex language or protocol for communication. [4]

The *ant foraging* behavior was first described in [5] and generalized by *Marco Dorigo* under the name of *Ant Colony Optimization.*[6] The reason why foraging-inspired optimization methods dominate the field of SI today is likely due to the lack of proper generalizations of the other bio-inspired methods of the early era of swarm intelligence research.

In this paper, we revisit another early 'ant algorithm' as a first step in our attempt to generalise it. After pointing out the behavioral similarity of this algorithm to Schelling's Segregation model [7], one of the most cited agent-based models of computational social science [8], we propose a novel variant of the algorithm and demonstrate its properties. The outlook of this research and a longer-term research plan will be discussed in the last Section of this paper.

2 The Ant Clustering Algorithm

The Problem Let's consider a two-dimensional regular lattice (grid). Each cell of the lattice may contain a single object. Objects can be of different type (or color). The task is to move the objects around in such a way that objects of the same type are placed next to each other (i.e., they are grouped or spatially clustered). In the simplest version of this problem, all objects are of the same type, but there are empty locations on the grid. The task is to collect the objects together in a single, contiguous patch.

We have a group of simple, identical agents (ants) that are capable of moving around on the grid and picking up, carrying and putting down objects as they go. Ants are always located at one of the cells. They can also step on cells holding objects. Ants can pick up the object at their current location, provided they are not carrying. Similarly, they can put down the object that they have with them at their current location, if the cell is empty. Ants are assumed to be simple: they have limited processing power and have no ability to communicate with one another. Their vision is also constrained: they can only see their immediate neighborhood. Given these serious limitations of capabilities, the agents are assumed to be cheap and thus available in larger quantities.

The Classic Algorithm The *ant clustering algorithm* inspired by the brood sorting [9] or cemetery organisation [10] behavior of ants, can be summarized as follows. [11] The ants wander around randomly on the grid, always moving to one of the neighboring cells of their current location. As they go, they observe the object at their current location. They also have a limited short term memory and thus remember the objects at the last couple of cells they have visited. If an ant is not carrying anything when it encounters an object, it will decide stochastically, whether to pick it up or not. The probability of picking it up will decrease proportionally to the number of objects of the same type recently observed by the ant. Similarly, if an ant carries an object when it encounters an empty cell, it will put down its carry with a probability proportional to the number of similar objects recently encountered.

This simple algorithm, while placing minimal processing burden on the agents, is capable of improving the spatial clustering of objects with surprising efficiency.

The Schelling Variant *Thomas C. Schelling*, a Nobel laureate in economics, proposed a simple model to study residential segregation in the US. He assumed a 2D rectangular lattice, the cells representing residential blocks. [7] Each cell may be occupied by a family of any of the two colors considered (red or blue), or may remain empty. Every once in a while the families will assess their satisfaction with their neighborhood. They will calculate the ratio of their same-color neighbors.

If this ratio falls below a *tolerance threshold* they pack up and move – in the model, to any of the empty locations on the lattice.

Schelling asked the question, what would follow from these simple rules, assuming that families are tolerant. What if they only move if the ratio of same-color neighbors falls, say, below 0.7? To answer this questions, Schelling performed simulations (originally on an actual chess-board with pennies and dimes, later, his followers, using computers), starting from a random initial configuration. Quite remarkably, these simulations show that Schelling's rules lead to families of the same color grouping together, with corridors of empty cells between the clusters.

It is not our task to discuss the meaning and consequences of Schelling's segregation model, especially, as it was already done extensively. [12]. However, the striking similarity between the model's emergent outcome and the desired solution of the spatial clustering problem suggest a simple collective algorithm, a variant of the ant sorting model. In this versions, the ants will not even have memory to remember recently visited locations.

Let's have the ants (agents) wander around randomly on the 2D lattice. If an ant does not carry anything and there is an object at its current location, it observes the neighboring cells. It calculates the ratio of neighbors with an object that has the same color as the object at its location. If this ratio is below a predefined threshold (a parameter of the algorithm), the ant will pick up the object at its cell. Similarly, if an agent arrives at an empty cell with an object in carry, it will calculate the ratio of neighboring cells with objects sharing the color of its carry. It this ratio is above the predefined threshold, it will put down the object.

3 Methods

We implemented the above distributed spatial sorting algorithm as an agent-based computational simulation. For this, We used NetLogo [13], a popular platform for multi-agent simulations. This implementation (available from GitHub[1]) allows for the extensive computational testing and analysis of the algorithm, but it is not suitable for any particular application domain.

The simulation takes the size of the world ($s > 0$) as a parameter, the number of ants ($n > 0$), as well as the number of object types (or colors, $c > 1$) as parameters. The world is initialized with a certain percentage of the locations ($d \in [0, 1]$, for *density*) holding objects – the object holding locations are determined randomly. The types (color) of the objects are also uniform random – each type having equal probability. Initially, ants are randomly located and their 'hands' are empty. Finally, the ants' threshold for picking up or dropping an object ($t \in [0, 1]$), that is the same for all ants, is a tuneable parameter of the implementation.

Our implementation works on a regular $s \times s$ grid with periodic boundary conditions. That is, the world is 'wrapped around' so that cells at the edges

[1] https://github.com/lgulyas1972/Schelling-s-Ants

are neighbors with the cells at the opposing end of the world. This torus topology ensures that all locations in the world are structurally similar, having the same number of neighbors and same average distances to other locations, etc. In our implementation, 8 neighbors are considered for each cell, the ones to the immediate *north*, *south*, *west* and *east*, as well as the ones to the *northwest*, *northeast*, *southwest* and *southeast*. This implies that the possible values for ratio of same-color neighbors are 0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875 and 1. Correspondingly, these are the meaningful choices for the tuneable threshold parameter t as well.

4 Results

Our computer simulations confirm that the Schelling-inspired collective ant algorithm results in efficient spatial clustering of objects. The top row of Figure 1 illustrates the workings of the algorithm in the case of a single object type (i.e., color). The left panel shows the initial, random distribution of objects, while the right one displays the world after 1000 time steps (iterations). The white cells are empty while the black ones contain objects.

The improvement of spatial clustering is clearly visible. Following the 719^{th} time step, the state of the world does not change anymore. However, by the nature of the algorithm, the ants (not shown) keep wandering around randomly. This can be useful in a dynamic environment where objects may disappear, or new ones may appear, or external forces may disturb the spatial allocation of objects. In such cases, the algorithm is capable of dynamically reacting to the changes.

As a comparison, the bottom row of Figure 1 shows a problem with 3 object types. As above, the left panel shows the initial, while the right one the improved configuration. Here the overall density of objects in the world is $d = 0.45$ and convergence took 483 iterations.

4.1 Measures of Performance

Figure 1 demonstrated the results of the collective behavior of ants qualitatively. In order to assess the performance of the algorithm, however, we need more formal measures.

The task of spatial ordering concerned in this paper is about 'increasing order' in the distribution of objects in space, or decreasing the amount of randomness in the spatial configuration. The first idea one has when considering the randomness of a system is *entropy*. [14] However, *spatial entropy*, i.e., the interpretation of entropy to a spatial configuration is not straightforward. [15] [16] Most importantly, it often involves the partitioning of space in a number of sub-areas (e.g., a coarser lattice laid over the space). This arbitrary partitioning lattice, sometimes with varying resolution, seems impractical for our purposes. Therefore, we propose another, simpler measure.

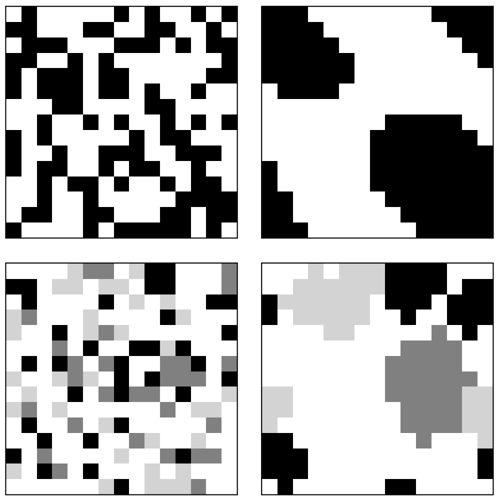

Fig. 1. The workings of the algorithm: the initial configuration (left column) and the state after 1000 iterations (right column) for a case of $s = 15, n = 25, d = 0.45$. In the top row $c = 1$ and $t = 0.4$, while in the bottom $c = 3$ and $t = 0.3$. White cells are empty, darker ones contain objects.

In order to assess how well objects of the same type (color) are grouped together spatially, we will consider the immediate neighborhood of each cell (i.e., the 8 cells surrounding it) In particular, we calculate the ratio of neighboring cells containing same color objects. For the purpose of this calculation, we handle empty cells as if containing a special type (color) of objects. We will use the average of this ratio across all cells as the first version of our measure of order.

$$q = \frac{\sum_{i=1}^{s} \sum_{j=1}^{s} \rho_{ij}}{s^2} \tag{1}$$

where ρ_{ij} is the ratio of same color neighbors of the cell at location (i, j) of the 2D grid of the world.

By considering possible differences in type (color) between neighboring cells, the measure, in effect, approximates the total length of boundaries between cells containing objects of different types. The higher the average ratio of same color neighbors, the fewer the number of 'edges' between cells that are part of the boundary. The shorter the length of the boundary is, the larger the patches containing same color object and thus the higher the quality of spatial clustering. Figure 2 shows the evolution of q during the course of the simulation whose results were depicted on the top row of Figure 1.

All the cases discussed so far were successful in the sense that the ants were able to reconfigure the spatial distribution of objects (and increase their clustering) in a finite number of time steps. After this period, the ants kept moving around randomly without ever touching the objects again. However, this is not always the case. Depending on the value of some parameters (in particular that of t, in relation to c and d), the ants may never find the final location of all ob-

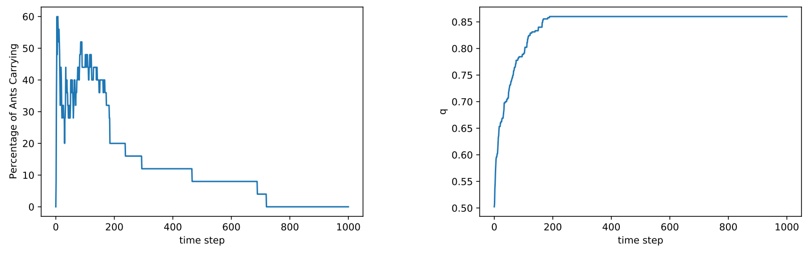

Fig. 2. The in-run time series of the percentage of agents carrying objects (left) and that of q (right) of the run on the top row of Figure 1.

jects. They may keep carrying objects with them. Figure 3 depicts such a case. The top row has the initial state on the left panel, while the emerging configuration on the right. The bottom row shows the percentage of ants carrying an object versus time on the left panel and the time evolution of q on the right.

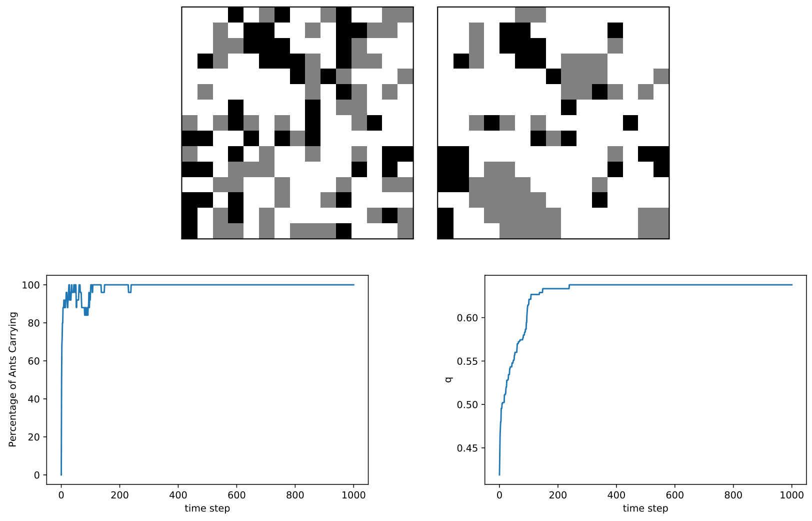

Fig. 3. Top row: spatial configurations of a non-converging case. The initial configuration is on the left, while the state after 1000 iterations is on the right. Parameter settings: $s = 15, n = 25, d = 0.45, c = 2$ and $t = 0.4$. White cells are empty, darker ones contain objects. Bottom row: the in-run time series of the percentage of agents carrying (left) and the q measure (right).

The problem with non-converging cases is that, while the spatial order of the objects 'on the ground' increases, as exemplified by our example, the ants keep carrying a significant number of objects. This means, that the emerging spatial configuration is not a legal solution to the original problem, as it contains fewer objects. Unfortunately, this is not reflected by our q measure of quality.

To fix this potential problem, we modify our measure by adding a term penalizing objects in carry. In addition to the $s \times s$ cells of the 2D grid of the world, we consider each ant as a 'virtual cell' without neighbors. The ρ_i ratio of same color neighbors of ant i will be 1 if the ant is not carrying as this is the best possible ratio. Conversely, for carrying ants the ratio will be 0, the worst possible ratio. This yields the following corrected measure of quality of spatial ordering:

$$Q = \frac{\sum_{i=1}^{s} \sum_{j=1}^{s} \rho_{ij} + \sum_{i=1}^{n} \rho_i}{s^2 + n} \tag{2}$$

Figure 4 shows the in-run evolution of Q for the converging case of Figure 2 (left panel) and for the non-converging case of Figure 3 (right panel). In the following, we will use the difference of Q between the emerging final state and that of the initial configuration as a concise way to discuss the improvement of quality achieved by the algorithm. We will denote this value by Δ_Q.

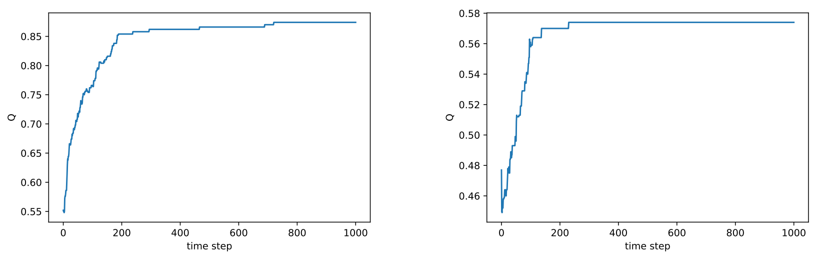

Fig. 4. The in-run time series of measure Q for the cases depicted on Figure 1 (left) and Figure 3 (right).

Another aspect of performance is the time it takes for the collective of ants to achieve the above measured improvement on spatial clustering. This is the number of time steps from the start until the last time step in which there are ants carrying objects. We denote this by t_{conv}, or time to converge.

Naturally, t_{conv} depends on the number of ants. Presumably, more ants can order objects quicker. Therefore, we also count the number of moves ants make carrying an object. That is, we sum, for all the ants, the number of time steps, when they had an object in carry. In other words, this is the number of elementary moves objects need to go through in order to reach the spatial configuration with improved order. This measure will be called the 'number of carrying hops' and will be denoted by H_c. On the other hand, the number of moves needed may also depend on the number of objects in the world ($m = d \cdot s^2$). Figuring m in, we can also calculate the number of carrying hops per object, $H_{cpo} = \frac{H_c}{m}$.

The left panel of Figure 2 shows the time series of the percentage of agents carrying during the simulation whose results were depicted on the top row of Figure 1. The t_{conv} value in this case is 719. On the other hand, the number of carrying hops was $H_c = 3306$ and H_{cpo} =32.652.

As seen earlier, in the problem depicted on the top row of Figure 1 and on Figure 2, the quality improvement was Δ_Q =0.321999, or 32%. However, it is important to consider the costs as well. Since it took H_c =3306 object movements to reach this increase in quality, we can say that the ant algorithm worked with effectiveness $\varepsilon = \Delta_Q / H_c$ =0.000097, using the actual parameter settings. This value gives the average increment in quality per carrying move. Notice that as H_c goes to infinity for non-converging runs, this measure of effectiveness (ε) goes to 0 in such cases.

As a comparison, in the run depicted on the bottom row of Figure 1, the quality improvement was Δ_Q =0.33, reaching a top of Q =0.754. The time to convergence was t_{conv} =483, with H_c =5192 carrying hops and H_{cpo} =51.279, yielding ε =0.000064.

4.2 Dependence on Parameters

In order to assess the characteristics of the performance of the proposed algorithm, we have extensive computational experiments varying some of the parameters governing the collective behavior of ants.

During these experiments, we have worked with a 15×15 world ($s = 15$). The number of ants were 10, 30 and 50. The density of objects (d) took 9 uniformly placed values between 0 and 1, while the number of colors were varied from 1 to 5. The only parameter governing the ants' behavior, the t threshold iterated over the meaningful values between 0 and 1, by an increment of 0.125. (The boundary values were included as a baseline.) The parameter values for the experiments are summarized by Table 1. Given the stochastic nature of both the task to solve and of the algorithm, the simulation was run 10 times for each parameter combination. The 13500 runs, in total, were stopped after 2000 time steps.

Parameter	**Meaning**	**Value(s)**
s	Size of grid (world)	15
n	Number of ants	10, 30, 50
d	Density of objects	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 1
c	Number of object types (colors)	1, 2, 3, 4, 5
t	Ants' threshold for pick-up/put-down	0, 0.125, 0.25, 0.375, 0.5 0.625, 0.75, 0.875, 1.0

Table 1. The parameter configurations of the computational experiments

Figure 5 summarizes the convergence properties of the experiments. Clearly, the algorithm has two regimes of behavior: a converging and non-converging one. There is a phase transition between them as a function of t, with large variance around the critical value. This critical value for t depends on c and d. In the converging region, t_{conv} grows with t. Note, however, that quick convergence may not mean optimal results. It may also signal the algorithm's inability to perform its task. Such premature convergence is demonstrated by the baseline cases of high density (e.g., $d = 1.0$).

The quality increase (Δ_Q) achieved during the runs is summarized on Figure 6. The general picture is similar to our observations regarding t_{conv}. The two regimes are clearly visible with a high-variance critical threshold separating them along the axis of t. In the convergence region Δ_Q increases with t (except for regions of likely premature convergence, e.g. at high densities and few colors). This supports our earlier point: too quick convergence does not always mean good results. We also observe a decreasing Δ_Q as t increases in the non-convergence region. This is due to the correction factor in the definition of Q, i.e., the number of ants continuously carrying. This also explains the negative values as no ant is carrying at the beginning.

Finally, the effectiveness of the various parameter settings are plotted on Figure 7. Here we see a different picture: the two regimes of behavior have disappeared. This is due to the construction of ε, as non-convergence means ever increasing H_c's and thus is responsible for ε going to o in the non-convergence region. (Notice also the missing values for high densities and few colors, caused by H_c=0, when the agents could not even start.) On the other hand, in the convergence region we see a monotonic decrease with increasing t. This suggest that with lower t the ants are able to find the highest-gain moves, but fine-tuning the spatial configuration requires a threshold closer to the critical value.

Overall, combining Figures 5, 6 and 7, we can conclude that for each combination of c and d, there is a sweet spot of t (between 0.25 and 0.675) that yields the best results.

5 Conclusion and Future Works

In this paper, we revisited one of the early swarm intelligence methods, spatial clustering algorithm inspired by the brood sorting behavior of ants. We pointed out its qualitative similarity to Schelling's Segregation model and proposed a novel variant based on this observation. We studied the behavior of the new method in detail, both discussing individual runs and analysing the algorithm's performance as a function of problem and algorithm parameters.

In addition to the case discussed in this paper, there are several variants and generalizations of both the problem and the algorithm. For example, more than a single object (i.e., piles) may be allowed at a cell and the discrete categorization of objects into types (or colors) may also be loosened, using a more general measure of similarity. The generalisation of the space is also possible, e.g., using a continuous 2D space instead of a discrete, regular lattice. Another, more important, direction of the generalisation of the problem domain concerns the topology of space. Instead of working with immediate neighbors on a grid, the algorithm may be defined on networks (graphs), the objects being placed on nodes (vertices). This direction will be pursued in subsequent papers.

The generalisation to networks may significantly widen the applicability of the ant sorting method. A method for the collective clustering of objects in 2D space may find applications in *swarm robotics*, a domain of emerging popularity. However, clustering (virtual) objects on networks by independent, non-coordinated entities could see applications distributed computing environments as well. For example, with the advent of 5G communication networks, novel methods for *mobile software agents* are needed to ensure efficient information processing in *edge computing* environments. [17] [18]

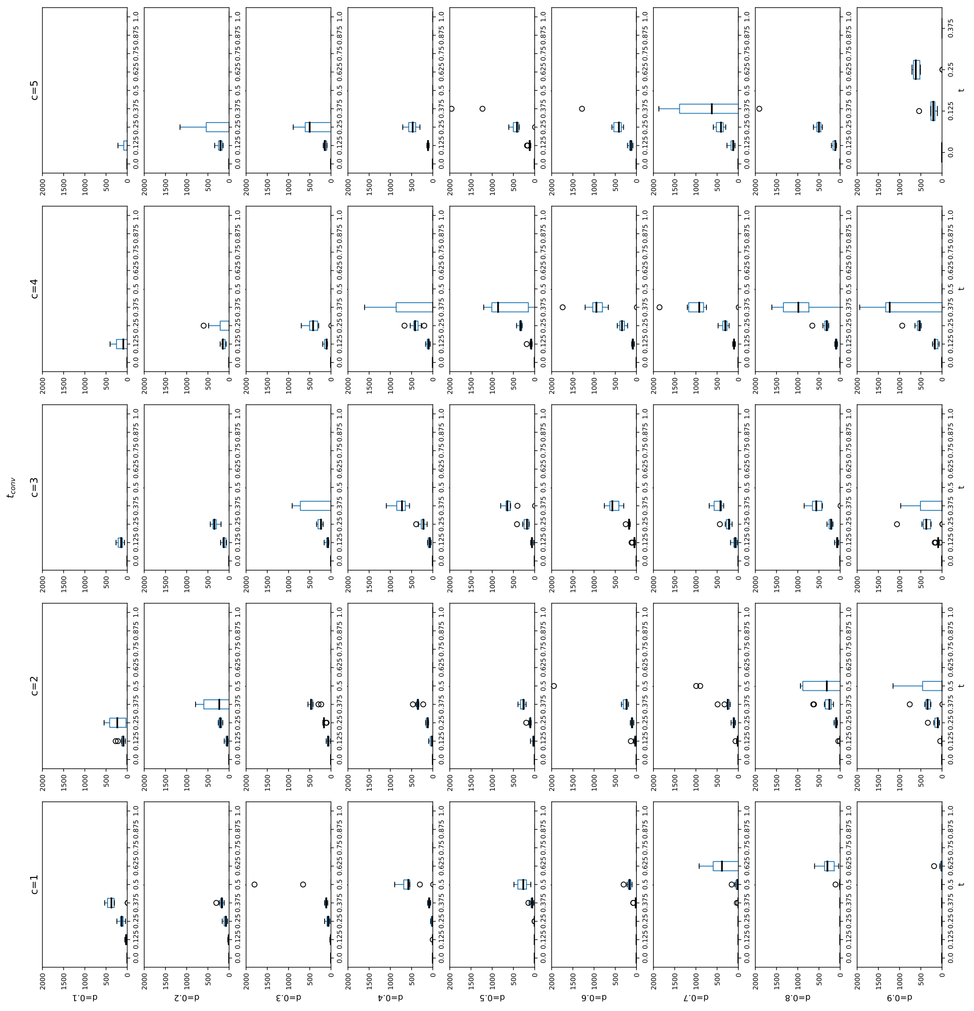

Fig. 5. Boxplot of t_{conv} summarizing the results of 10 runs for $n = 10$ as a function of d, c and t. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots. Non-converging runs have no data points.

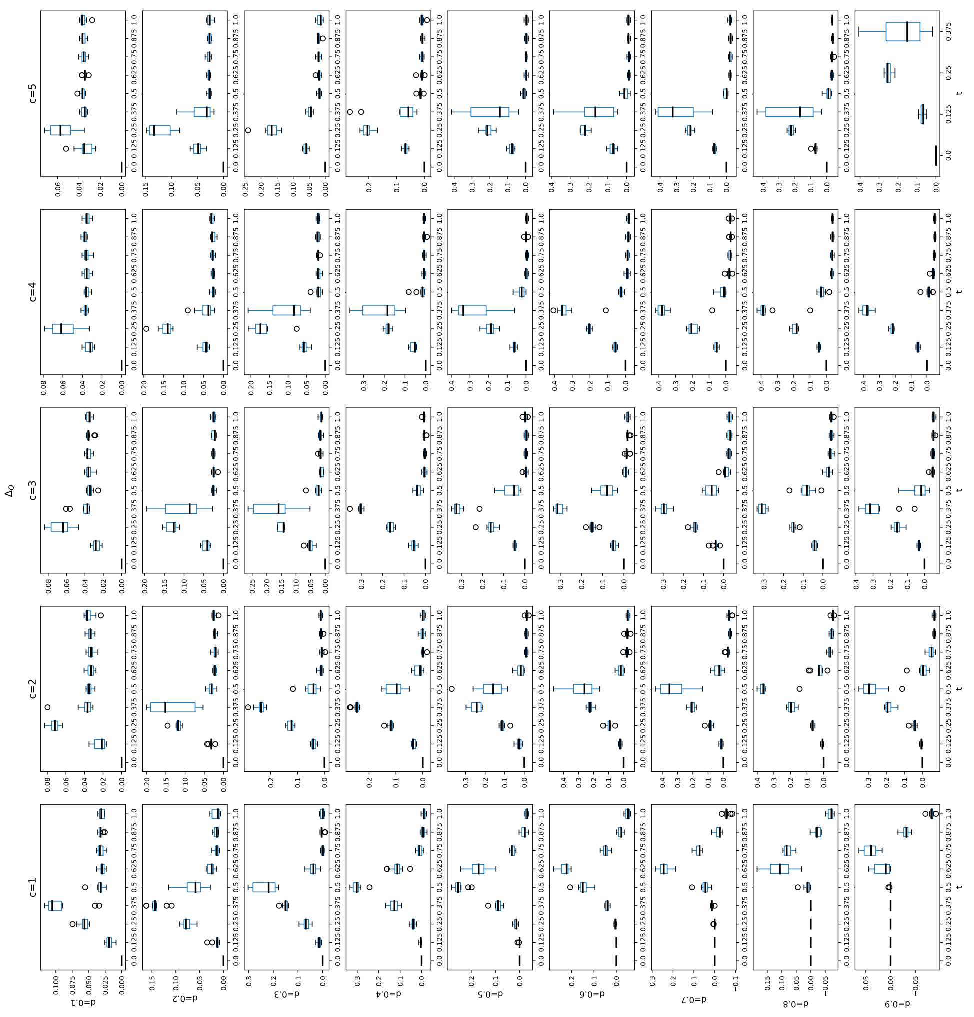

Fig. 6. Boxplot of Δ_Q summarizing the results of 10 runs for $n = 10$ as a function of d, c and t. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots. Note the varying range of the vertical axis on the individual plots.

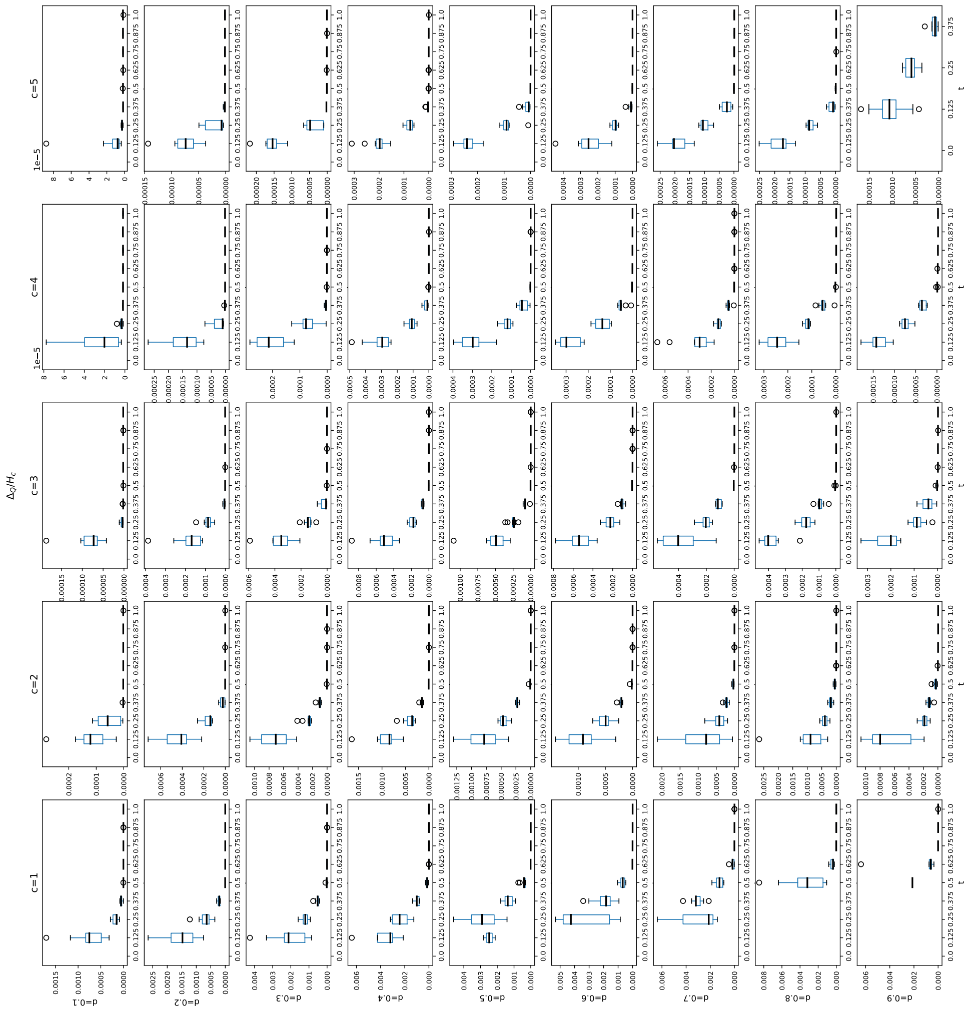

Fig. 7. Boxplot of ε (Δ_Q/H_c) summarizing the results of 10 runs for $n = 10$ as a function of d, c and t. The boxes extend from the Q1 to Q3 quartile values of the data, with a line at the median (Q2). The whiskers extend from the edges of the boxes to show the range of the data. Outliers are plotted as separate dots.

References

1. Chakraborty, Amrita & Kar, Arpan. (2017). Swarm Intelligence: A Review of Algorithms. 10.1007/978-3-319-50920-4_19.
2. Lones, M.A. Mitigating Metaphors: A Comprehensible Guide to Recent Nature-Inspired Algorithms. SN COMPUT. SCI. 1, 49 (2020). https://doi.org/10.1007/s42979-019-0050-8
3. Jean-Arcady Meyer; Stewart W. Wilson, "Task differentiation in Polistes wasp colonies: a model for self-organizing groups of robots," in From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior , MIT Press, 1991, pp.346-355.
4. Seeley, T. D. (2010). Honeybee democracy. Princeton, NJ: Princeton University Press.
5. Beckers, Ralph, Goss, S., Deneubourg, Jean-Louis & Pasteels, Jacques. (1989). Colony Size, Communication and Ant Foraging Strategy. Psyche. 96. 10.1155/1989/94279.
6. M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," in IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28-39, Nov. 2006, doi: 10.1109/MCI.2006.329691.
7. Schelling, T. C. (1978). Micromotives and macrobehavior. New York: Norton.
8. The powers and perils of using digital data to understand human behaviour (Editorial), Nature 595, 149-150 (2021), doi: https://doi.org/10.1038/d41586-021-01736-y
9. Ana B. Sendova-Franks, Samuel R. Scholes, Nigel R. Franks, Chris Melhuish, Brood sorting by ants: two phases and differential diffusion, Animal Behaviour, Volume 68, Issue 5, 2004, pp. 1095-1106,, https://doi.org/10.1016/j.anbehav.2004.02.013.
10. Bonabeau, Eric, Theraulaz, Guy, Fourcassié, Vincent & Deneubourg, Jean-Louis. (1998). The Phase-Ordering Kinetics of Cemetery Organization in Ants. Santa Fe Institute, Working Papers. 57. 10.1103/PhysRevE.57.4568.
11. Parunak, H.V. (1997). "Go to the ant": Engineering principles from natural multiagent systems. Ann. Oper. Res., 75, 69-101.
12. Hatna, Erez and Benenson, Itzhak (2012) 'The Schelling Model of Ethnic Residential Dynamics: Beyond the Integrated - Segregated Dichotomy of Patterns' Journal of Artificial Societies and Social Simulation 15 (1) 6 ¡http://jasss.soc.surrey.ac.uk/15/1/6.html¿. doi: 10.18564/jasss.1873
13. Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.
14. Robert M. Gray. 1990. Entropy and information theory. Springer-Verlag, Berlin, Heidelberg.
15. Batty, M. (1974), Spatial Entropy. Geographical Analysis, 6: 1-31. https://doi.org/10.1111/j.1538-4632.1974.tb01014.x
16. Wang C, Zhao H. Spatial Heterogeneity Analysis: Introducing a New Form of Spatial Entropy. Entropy. 2018; 20(6):398. https://doi.org/10.3390/e20060398
17. Guo, Y, Jiang, C, Wu, T-Y Wang, A. Mobile agent-based service migration in mobile edge computing. Int J Commun Syst. 2021; 34:e4699. https://doi.org/10.1002/dac.4699
18. Inés Sittón-Candanedo, Ricardo S. Alonso, Juan M. Corchado, Sara Rodríguez-González, Roberto Casado-Vara, A review of edge computing reference architectures and a new global edge proposal, Future Generation Computer Systems, Volume 99, 2019, pp. 278-294,, https://doi.org/10.1016/j.future.2019.04.016.