The Chinese Journal of Artificial Intelligence, **2022**, *1*, **e090322201940**

RESEARCH ARTICLE

Parameter Sensitivity Analysis of the Democratic Behavior of Swarm Robots

Andreas Hügli[1], Marco A. G. Pereira[1], Rolf Dornberger[2] and Thomas Hanne[2,*]

[1]*Medical Informatics, University of Applied Sciences and Arts Northwestern Switzerland, Muttenz, Switzerland;* [2]*Institute for Information Systems, University of Applied Sciences and Arts Northwestern Switzerland, Basel, Switzerland*

Abstract: ***Aims*:** The purpose of this study is to determine under what conditions, such as noise and malfunction, successful consensus achievement in swarm robotics is possible.

***Background*:** Swarm robots can be used to solve exploration problems, such as the best-of-n problem. Consensus achievement plays a crucial role as the swarm must collectively agree on a solution. This task can be even more challenging considering noise and malfunctioning or rogue agents.

***Objective*:** This study aims to determine how robust the consensus achievement algorithm is against noise and rogue agents, considering the effect of adding memory to the agents and further parameter tuning.

***Methods*:** We implement a baseline based on the democratic honeybees algorithm and investigate the performance and robustness of the consensus achievement during a number of computational experiments. In particular, the number of agents in the swarm, the number of iterations, the number of positions an agent can visit per iteration, the number of neighbors an agent shares its best option with, and the majority threshold defining the majority based on a fraction of agents in the swarm, and the minimum number of iterations to achieve consensus are investigated regarding their impact.

***Results*:** For better performance, memory has been implemented so that each agent remembers and retains their previous highest quality score if no one better has been found in the current exploration phase. We show that the algorithm is viable and offers robustness in the considered scenarios when memory is added. In particular, we establish a baseline for the democratic honeybees algorithm and ascertain adequate parameter values to ensure the algorithm's best performance. The algorithm is sufficiently robust against noise, and to an extent, against rogue agents. Furthermore, parameter tuning also proved to help the swarm explore very large search spaces.

***Conclusion*:** The consensus algorithm appears sufficiently effective under adverse conditions such as noise and rogue agents, especially when countermeasures are considered.

***Other*:** Further scenarios such as specific communication topologies could be investigated in future research.

ARTICLE HISTORY

Received: June 28, 2021
Revised: November 08, 2021
Accepted: December 02, 2021

DOI:
10.2174/2666782701666220309104945

Keywords: Index Terms, best-of-n, swarm robotics, democratic honeybees, consensus achievement, collective decision making.

1. INTRODUCTION

The decision-making of a collective of agents, called a swarm, is often not traceable to any single agent and is denoted as collective decision-making. Each agent of this swarm has only partial knowledge concerning the global solution to which the swarm has to consent. Therefore, the agents need to share their partial knowledge with the other agents and collectively agree on a particular solution. This principle is called self-organization.

This has been observed in multiple swarms of social insects in nature [1, 2]. Collective decision-making can be divided into two categories: consensus achievement and task allocation. We investigate the consensus achievement. Swarm robotics aims to develop scalable and robust systems to noise along with malfunctioning or rogue agents. Accordingly, swarms are designed to have no central authority; we speak of a decentralized swarm. A specific task for swarm robots is to solve the best-of-n problem, especially if n > 2 [3]. Here, the agents collect *n* different options and need to consent as a swarm to a particular option.

In one study [4], the authors used a nature-inspired model that uses the democratic behavior of honeybees,

*Address correspondence to this author at the Institute for Information Systems, University of Applied Sciences and Arts Northwestern Switzerland, Basel, Switzerland; E-mail: thomas.hanne@fhnw.ch

2666-7835/22 $65.00+.00

inspired by Seeley's study on Honeybee Democracy [5]. In this approach, honeybees individually explore new positions for a hive, but decide as a collective for the best position to solve the best-of-n problem. As it is a unique approach in the literature, we set a baseline for future research of the algorithm, by implementing the approach and tuning its parameters.

Because of the nature of collective decision-making, the swarm might be highly susceptible to malicious or rogue agents. Such rogue agents can influence the performance of the swarm in various ways. They either slow down the consensus achievement or make it impossible in a limited time frame. Moreover, such rogue agents may cause the swarm to make a wrong decision and possibly fail to complete its task, which should not occur when using this algorithm in a real deployment [6]. Recent research addressing these issues uses the emerging technology of blockchain to track rogue agents and eventually exclude them from the swarm [6, 7]. Further discussions of the problem are provided in [8-13]. However, none of these publications investigate the influences of parameters that affect the success of consensus achievement under the conditions of such rogue agents. This also applies to the effects of noise, which can be disastrous as it gives the agent a false sense of an option when it has been altered by environmental changes, sensor malfunctions or inaccuracies [4].

2. DEMOCRATIC HONEYBEE ALGORITHM

In [4] the best-of-n problem is approached using a finite state machine with four states. The agents explore distinct parts of the entire search space, visit three random positions and save the best scores in an exploration table (exploration state). The agents then communicate their best opinion to their neighbors, with implicit hints as to where to find it (dissemination state). In the broadcast state, the agents vote for the bestoption. Consensus is only achieved if the majority votes for the same option three consecutive times, and exploration can be ended (final state). In Algorithm 1, we provide the pseudo-code for our implementation of the honeybees' democratic behavior. The full code for our implementation in Python can be found in our public GitLab repository[1].

Note that the general convergence of related algorithms has occasionally been analyzed in the literature, also under noise conditions (*e.g.* [14]). However, the analysis focuses on general properties such as almost sure convergence, while it is usually not possible to derive specific results such as achieving consensus with limited time. We therefore focus on an experimental approach to analyze these aspects.

From the description of the algorithm, it is clear that there are many parameters that can be adjusted for efficient results or even a robust algorithm (Table **1**).

[1]https://gitlab.com/anhue/hyperparameter-analysis-of-the-democratic-behaviour-of-swarm-robots

Algorithm 1. Democratic honeybee behavior

Result: Position and value of the solution.

Initialize population with *n_visits* random locations per agent;

for *i iterations* **do**

  Evaluate quality scores for the positions of each agent;

  Evaluate the best position for each agent, taking last best position into consideration;

  Advertise the best position to other agents to visit in next iteration;

  Vote for best position;

  if *consensus was achieved n_consensus times consecutively on the best position* **then**

    Solution found;

end

Table 1. Parameters of democratic honeybee swarm.

Parameter	Description
n_agents	The total number of agents in the swarm.
$n\_iterations$	The total number of iterations with which a swarm can run through its optimization loop. If this number is reached, the swarm stops its optimization without achieving consensus.
n_visits	The number of positions an agent can visit per iteration.
$n\_neighbors$	The number of neighbors with whom an agent shares its best option. Since an agent will visit this position in the next iteration, the parameter n_visits should be greater or at least equal to $n\_neighbors$.
$majority\_threshold$	Setting a threshold for a majority based on a fraction of agents in the swarm. This parameter should be within $(0,1)$.
$n\_consensus$	Setting a threshold for the number of iterations the swarm has to consent to consecutively for the same option to achieve consensus. To consent to an option in a given iteration, the majority has to vote for this option.

3. EXPERIMENTS

To measure the performance of the swarm, we use two quality indicators. Firstly, the best solution of the swarm is compared with the true best solution of the search space, resulting in a relative deviation [in %] from the true best solution. If the relative deviation is 0%, the swarm has obviously found the true best solution. On the other hand,

we measure the number of iterations needed to achieve consensus, because speed is also an important factor. To reduce bias, we always run a swarm 10 times with the same configuration.

As a search space we set up the well-known Rastrigin function in discrete form. Rastrigin is a highly multi-modal function with a symmetric structure and bounds defined as $(-5.12, 5.12)$ and has roughly 100 local minima in the dimension $\mathbb{R}^2$ with a global minima at the position (0, 0) with a value of 0 [15]. In our case, we transform this continuous function into a discrete function in the form of a matrix, which we call search space. The search space is populated in different sizes with the steps m and n (corresponding to x and y in the continuous function), which leads to different sampling rates and thus to the resolution of the original continuous function. As we have to be sure that the global minimum is also sampled in our discrete matrix with a value of 0, we have to use an uneven sampling rate for m and n. Therefore, we will always find the global minimum in the centre of the matrix.

To create a starting point, the first experiments were performed under perfect conditions, *i.e*. no noise was fed in, and no rogue agents were used, see algorithm 2. All experiments were done and plotted against the relative deviation from the global minima and against the number of iterations needed to achieve consensus. To start initial experiments, the influence of the number of agents $n\_agents \in \{5, 10, 15, 30, 100\}$ and the search space size $\in \{25, 121, 255, 441, 961, 10201, 90601\}$ was investigated (Figs. **1** and **2**).

Towards the end, we tried to stick to the parameters used in [4], but adaptations were necessary. Parameters for our algorithm with the original values from [4] in parenthesis, are: number of agents (15), number of rogue agents (0), number of visits per individual (3), number of rounds necessary to achieve consensus (3), noise (0), majority threshold (0.5) and search space (255).

With the setup found, further experiments were conducted to evaluate the impact of rogue agents or noisy values in the algorithm to determine its robustness. When implementing rogue agents, they were given a position corresponding to a solution that they were to advertise, disregarding other positions. For the purpose of bias reduction, four different experiments were performed, using four different positions with small to large deviations from the true best solution.

Because agents can explore more of the search space with more visits, we test how we can take advantage of this parameter to achieve increased robustness in the swarm. The number of rogue agents used was $\in \{1, 2, 3, 4, 5, 6, 7\}$ and the number of visits $n\_visits \in \{2, 3, 4, 5, 6, 7, 8\}$

As the number of neighbors and visits determines how the search space is explored, we suspect that tuning them can improve the algorithm's performance under perfect but also under manipulated conditions. Therefore, we test the influence of the number of neighbors $n\_neighbors \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and the number of additional visits $n\_visits \in \{1, 2, 3, 4, 5, 6, 7, 8\}$.

Subsequently, we experimented with the value of the number of consensus $n_{consensus} \in \{1, 2, 3, 4, 5, 6, 7, 8\}$, as this should have an impact on the consensus achievement of the swarm.

To evaluate the impact of the majority threshold on consensus achievement, the values $majority_threshold \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$were tested.

To finalize the comparison of multiple parameters, we compare the two parameters that dictate how a swarm achieves consensus. In this case, the number of rounds to achieve consensus $n\_consensus \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ and the majority threshold *majority_threshold* $\in$ {0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} were observed.

When considering noise, a value was sampled from a normal distribution, multiplied by a factor $sigma$ and inserted into the quality score found by an agent. The absolute values of noise are $\sigma \in \{0.0, 0.5, 1.0, 5.0, 10.0, 20.0, 30.0\}$.

4. RESULTS

We started by evaluating the differences in the relative deviation from the best solution in the search space and the number of iterations needed to achieve consensus when the search space is expanded (Fig. **1**). We observe that as the search space increases, the accuracy of the swarm tends to get worse, although the number of iterations needed to achieve consensus seems to be constant, with medians between five and seven. For the subsequent experiments we have chosen the search space with size 961 as the baseline as it has a similar performance to the search spaces with size (255, 441), but is the largest of them.

When plotting the effect of the number of agents (Fig. **2**) in the swarm against the predefined search space, we see that the more agents a swarm has, the better the solution gets. The original paper [4] defined a swarm with 15 agents showing promising results. Although they need the most iterations to achieve consensus, we use this as our baseline.

To test the robustness of the algorithm, the number of rogue agents were increased from zero to ten, with a quality score of 57.849, corresponding to the relative deviation of 74.3% to the best solution (Fig. **3**). We observe that the algorithm behaves robustly against a maximum of seven rogue agents and completely collapses with eight rogue agents. Therefore, this is expected behavior, as rogue agents > = 8 directly lead to a majority (swarm with 15 agents). However, the number of iterations needed to achieve consensus increases slightly as the number of rogue agents increases, unless the number of rogue agents is > = 8, in which case the algorithm converges to the wrong solution very quickly. It is also worth mentioning that the algorithm most often does not consent to one solution.

When we improve the quality score for the rogue agents to 33.114, which corresponds to a relative deviation

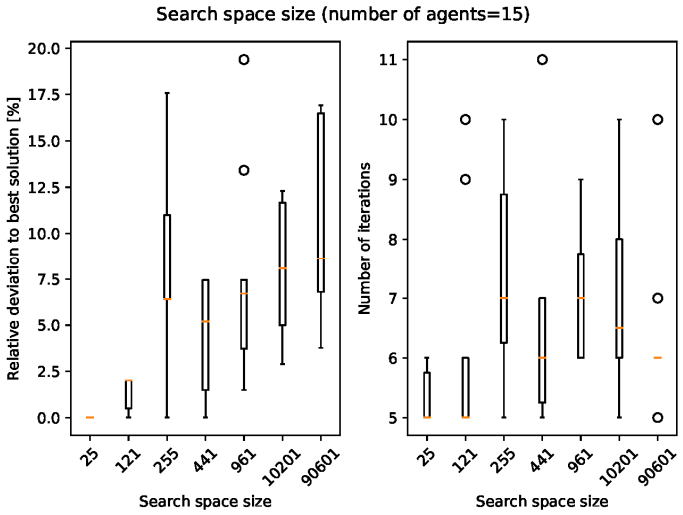

Fig. (1). Search space size plotted against relative deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

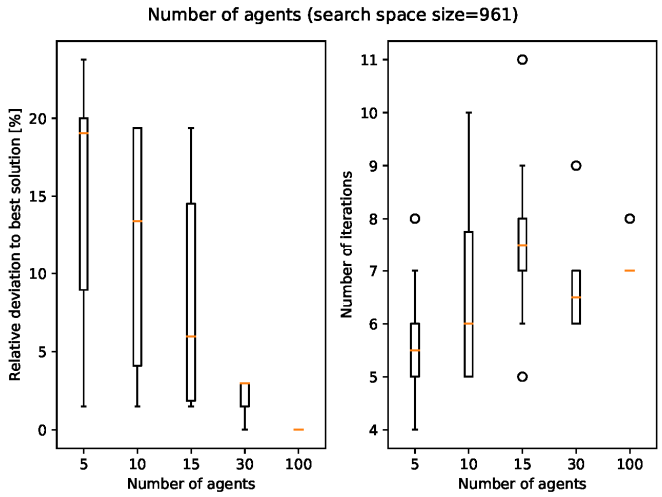

Fig. (2). Number of agents plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

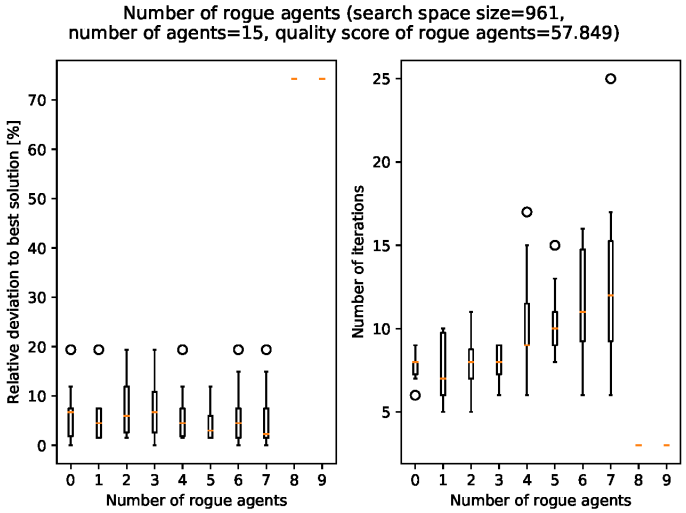

Fig. (3). Number of rogue agents advertising the quality score 57.849 against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

of 42.5% from the true best solution, we observe similar behavior (Fig. **4**). Again, the swarm collapses completely collapses with rogue agents > = 8. Also very well interpretable is the graph to the right as we see that the time to achieve consensus increases from zero to seven rogue agents and is static for eight and nine agents as rogue agents completely take over the swarm and achieve consensus after three iterations.

By increasing the quality score for the rogue agents to 18.493, with a corresponding relative deviation of 23.8%, we observe an increased influence of the rogue agents (Fig. **5**). Here the swarm starts to struggle with seven rogue agents, as in two runs the swarm collapses and is led astray by the rogue agents. With rogue agents > = 8 the swarm completely collapses. We also see that the overall performance of the swarm has decreased compared to the previous two.

Finally, we experiment with rogue agents advertising a quality score of 9.291, which corresponds to a relative deviation of 11.9% and obtain results very close to the correct quality score of 0 (Fig. **6**). Here, even a single rogue agent can lead the swarm astray. Nevertheless, in most runs the swarm is able to protect itself against a single rogue agent. With rogue agents [2, 3, 4] the swarm is led astray most of the time, but manages in several runs to achieve consensus with a better solution than that advertised by the rogue agents. With rogue agents > = 5 the swarm collapses

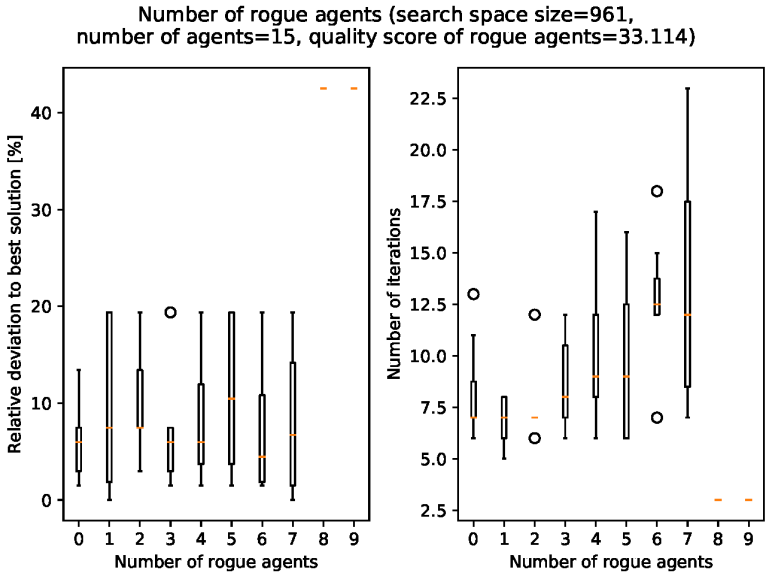

Fig. (4). Number of rogue agents advertising the quality score 33.114 against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

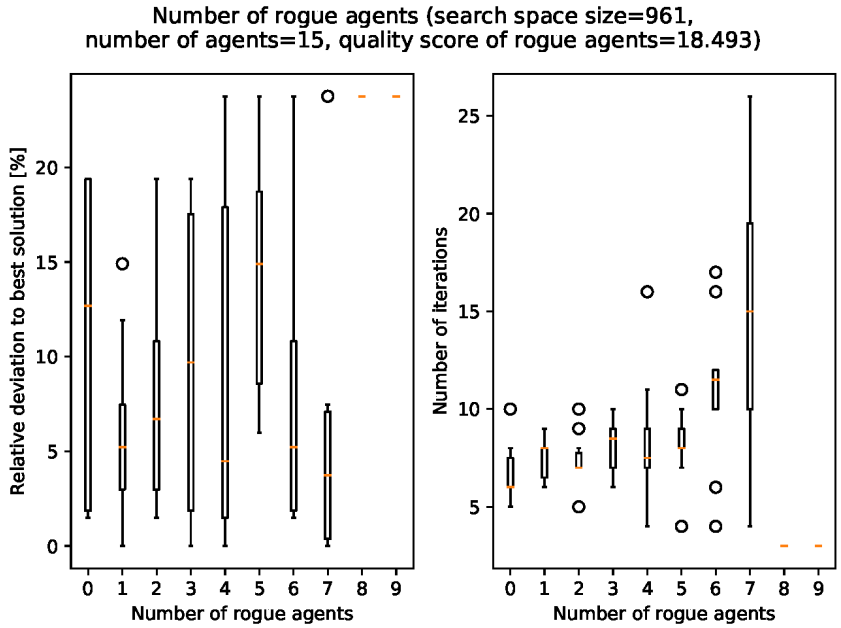

Fig. (5). Number of rogue agents advertising the quality score 18.493 against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

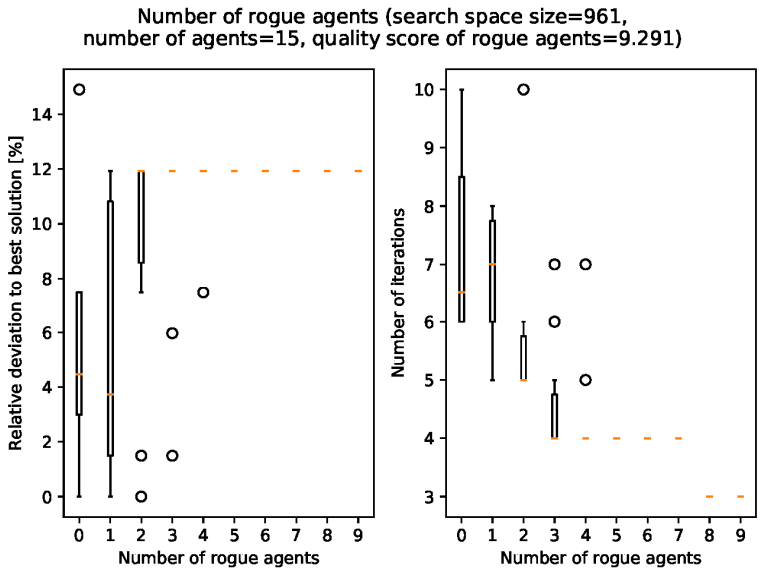

Fig. (6). Number of rogue agents advertising the score 9.291 against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

completely. This is understandable because when analyzing the search space, we see that only 25 fields are equal to or better than 9.291, which is 2% of the entire search space.

Next, we evaluate parameters of the algorithm other than the number of visits per agent and iteration. We observe a large error if the number of visits is only two, as each agent only visits positions that have been shared with him. Therefore, new positions are only observed in the first iteration and only shared afterwards. With an increased number of visits, the accuracy increases but remains constant from then on, which slightly reduces the variance. As for the number of iterations (excluding the number of visits 2, which consents quite quickly without new positions), we notice a downward trend suggesting that an increase in the number of visits could accelerate the swarm's consensus achievement (Fig. **7**).

With prior knowledge of how the rogue agents affect the swarm, the parameters for achieving consensus, have been adjusted. In Fig. (**8**), we can clearly see that while the number of visits performed per agent and iteration increases, the deviation from the best solution decreases, which illustrates a power law relationship between the two variables. We also see that more rogue agents lead to more iterations needed to achieve consensus, while increasing the number of visits does not have a significant impact, however a slight upward trend can be inferred.

By reducing the quality score advertised by rogue agents, we observe a similar trend as before, albeit the impact becomes clearer in the iterations. Again, increasing the number of visits per agent and iteration seems to reduce the influence of the rogue agents in the consented solution, as illustrated in Fig. (**9**). With a better quality score, we can now clearly see that increasing the number of visits increases the iterations needed for consent.

The same tendency applies to the situation with a quality score that is very close to the true best solution, although we lose the power law quality, which becomes rather linear. Again, it is noticeable that with seven rogue agents, the swarm does not converge to a better solution unless the number of visits is increased to a number > 4, as illustrated in Fig. (**10**). There is an even clearer upward trend in iterations to convergence, which indicates an increase in iterations as the number of visits increases.

Next, the influence of the number of neighbors is explored (Fig. **11**), with a search space of 961. We see that our algorithm no longer improves its accuracy for a neighborhood larger than three, while a narrower neighborhood reduces its accuracy. However, as the the number of neighbors increases, a reduction in the iterations needed to reach consensus is highlighted.

In Fig. (**12**), we increase the agents' difficulty in finding the true best solution by increasing the size of the search space by a factor of approximately 94. Increasing the number of neighbors initially seems to increase the relative deviation from the true best solution, but then stagnates at the number of neighbors > 3. Although for the number of neighbors < = 3 the accuracy has a large variance and abruptly becomes very small when the number of neighbors becomes > 3. In terms of the speed at which concensus is reached, we again observe an increase in the speed of convergence with a power law relationship quality.

As the rate of convergence increased when different neighborhoods were plotted, it was hypothesized that different combinations of neighborhoods and the number of visits performed could be beneficial to the swarm's performance. In this particular case, we have to consider that the number of visits depends on the neighborhood size, as the advertised positions should be visited by each agent they were advertised to. Therefore, for the number of visits, we increase the size of the neighborhood by a specific amount so that the agents have advertised positions but also need to visit some new random positions, *e.g.* in Fig. (**13**), the blue curve illustrates the example of a neighborhood of size two and 2 + 1 visits.

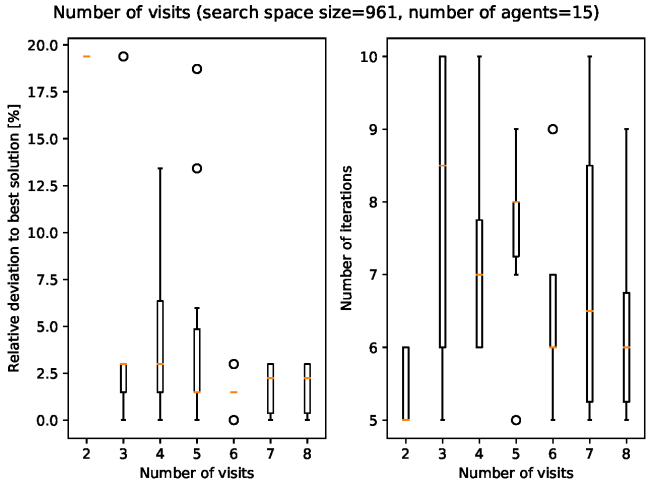

Fig. (7). Number of visits plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

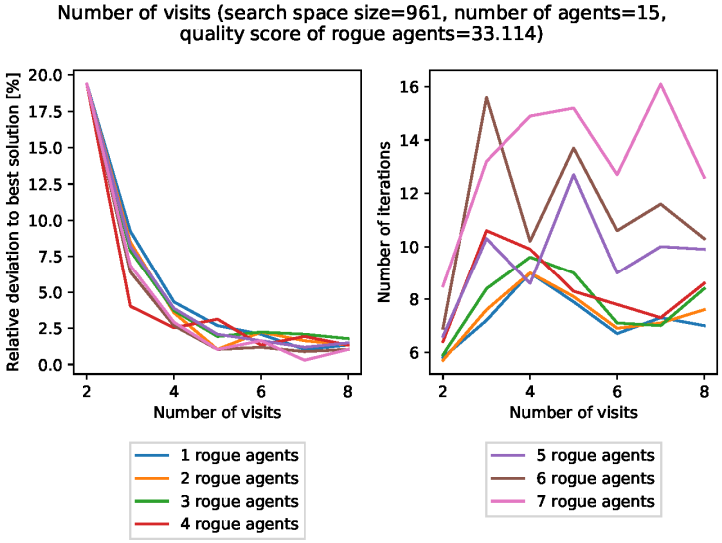

Fig. (8). Number of visits with different number of rogue agents, advertising the quality score 33.114, plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

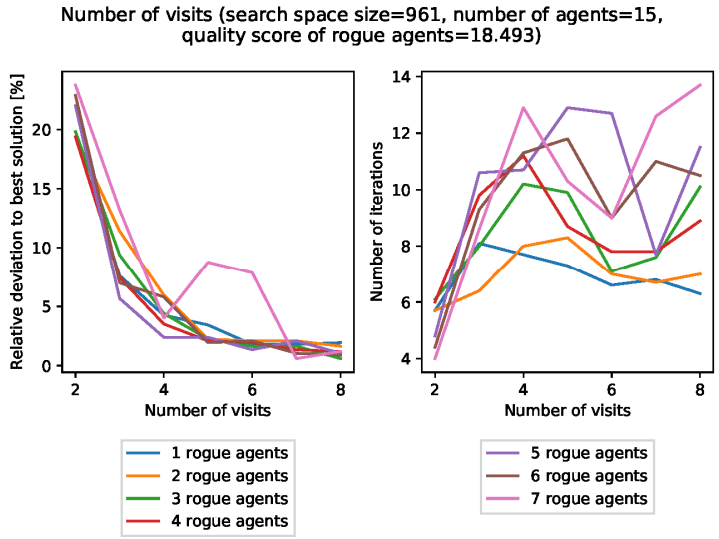

Fig. (9). Number of visits with different number of rogue agents, advertising the quality score 18.493, plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

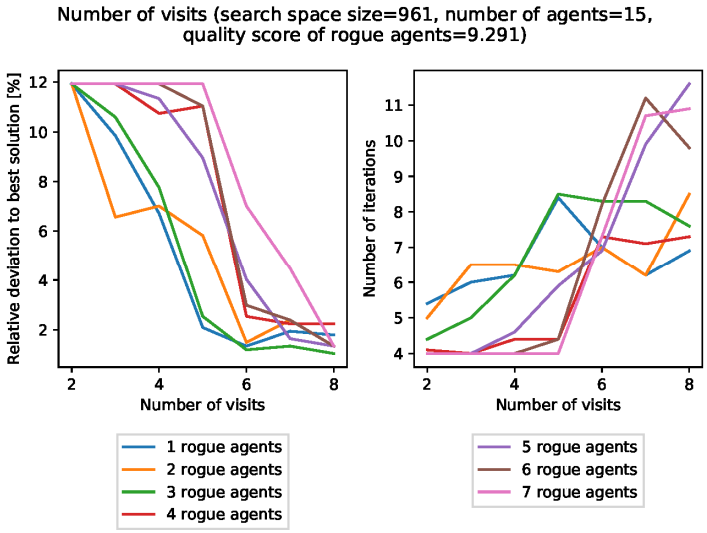

Fig. (10). Number of visits with different number of rogue agents, advertising the quality score 9.291, plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

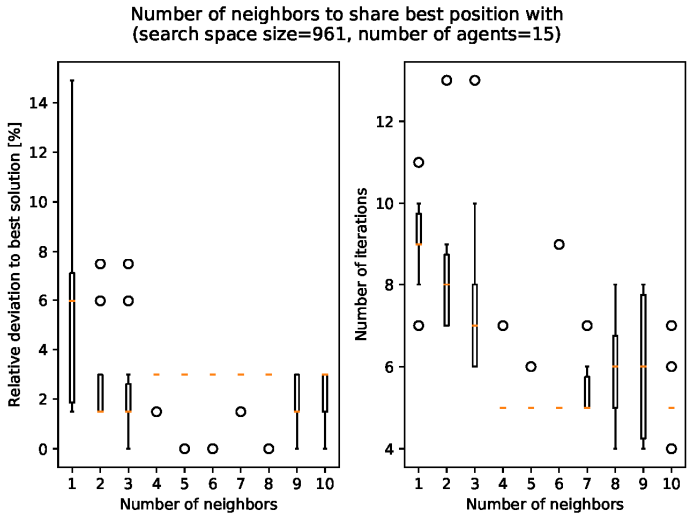

Fig. (11). Number of neighbors with search space size of 961 plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

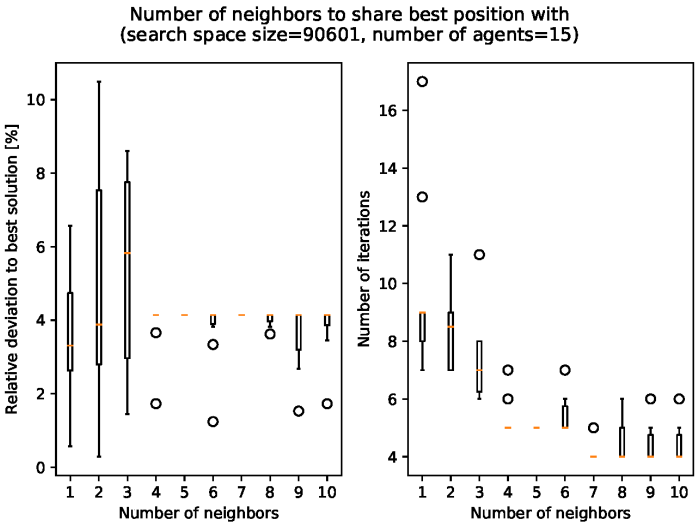

Fig. (12). Number of neighbors with search space size of 90601 plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

In Fig. (**13**) we observe a peak when using two and four as neighborhood size, with $+2$ and $+1$ visits respectively, but it disappears completely with a size of > 4. For scenarios with a higher number of visits such peaks do not occur at all. It is also notable that the accuracy remains constant until a given number of neighbors is reached, where it improves in all cases up to the best solution with the number of visits $> neighbor + 3visits$. Increasing the number of visits shows an eventual convergence of the swarm towards the true best solution. In terms of convergence speed, we notice an improvement with an increasing number of visits, which is plausible as with more visits, each agent is able to explore a larger part of the search space. Note that faster convergence towards a solution does not mean a faster computational method as the visit of a position also takes time. In a real-world scenario, the effective time resulting from the considered parameter setting should be taken into account.

For further investigations, we increase the search space, first by a factor of approximately 11 (Fig. **14**) and then 94 (Fig. **15**). We notice similar plots as in the previous one, with a slightly improved accuracy along with a higher number of neighbors and visits. There is a sudden decrease

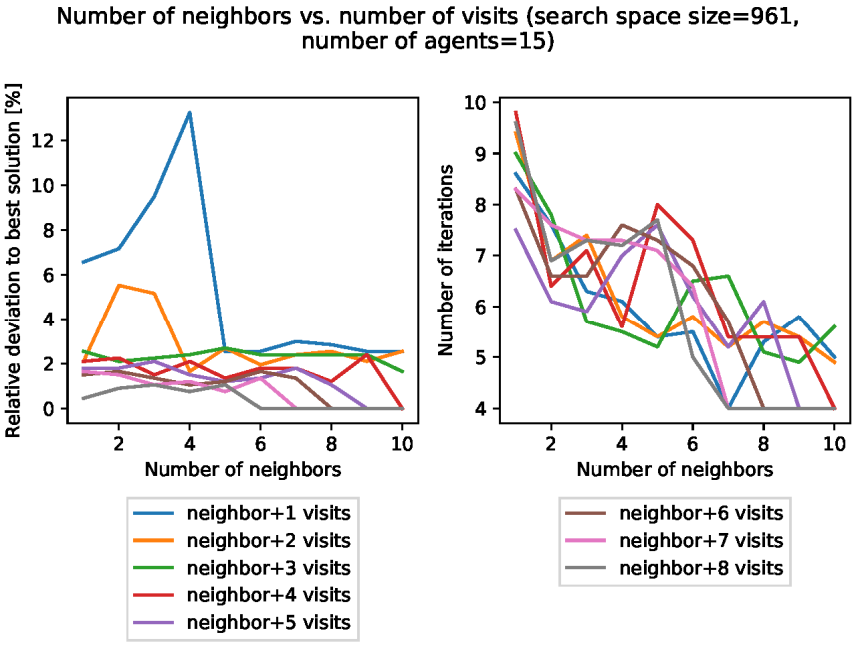

Fig. (13). Number of neighbors with different number of visits with search space 961 plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

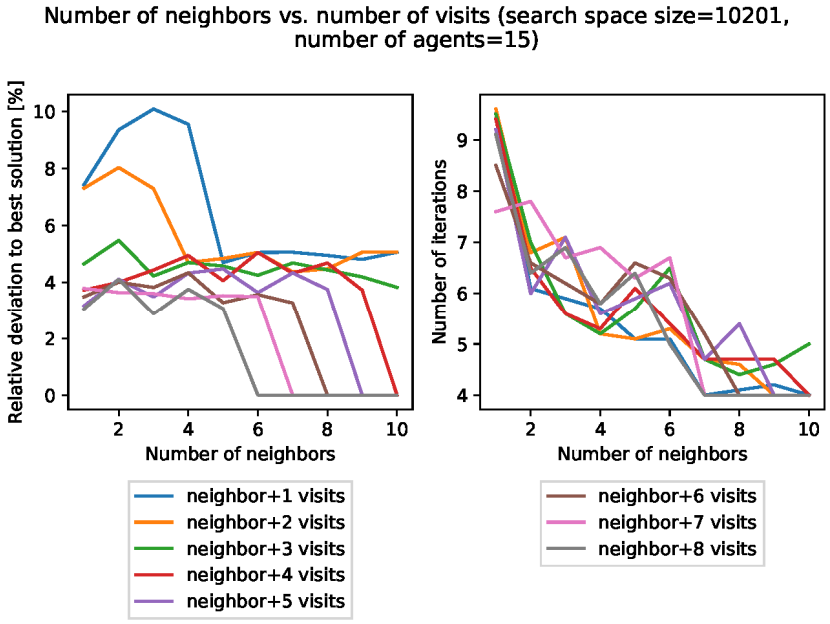

Fig. (14). Number of neighbors with different number of visits with search space 10201 plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

in the deviation from the best solution when the number is increased and when the number of visits is $> neighbor + 3\ visits$. In terms of iterations, in a search space of 961, we see the neighboring combinations [+1,+2,+3] stop decreasing when they are lost in a local minima for a search space of 961, however, for larger search spaces, they follow the trend and decrease to a total number of four iterations.

When increasing the threshold for the majority of the swarm to achieve consensus, we also observe an indifference of the algorithm to this parameter (Fig. **16**). Although the accuracy seems to be better on a small < 0.3 threshold and on greater > = 0.6 thresholds, the variance on all thresholds is quite high, reaching from the true best solution to a relative deviation of almost 20%. As opposed to the deviation from the true best solution, we see that the number of iterations increases slightly as the threshold increases, as a larger part of the swarm needs to vote for the same solution.

Analyzing how the number of consensus affects the performance of the swarm, we can clearly see a trend in both subplots (Fig. **17**). The performance increases significantly as the number of consensus rounds increases. The same is true for the number of iterations needed, where it is visually clear that while the performance improves, there is also an increase in the number of iterations needed to reach consensus, which is the definition of the plotted parameter.

Plotting the majority threshold against the number of rounds to consensus, we notice that when the majority

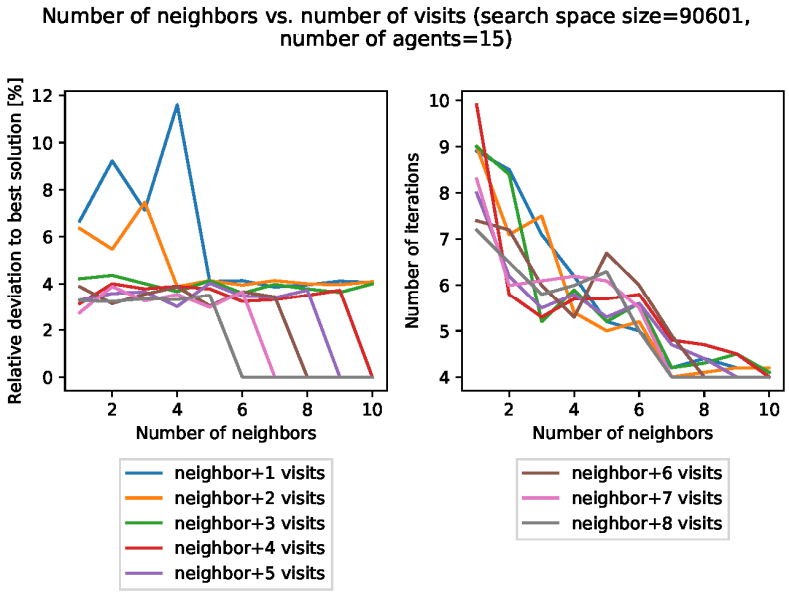

Fig. (15). Number of neighbors with different number of visits with search space 90601 plotted against deviation of true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

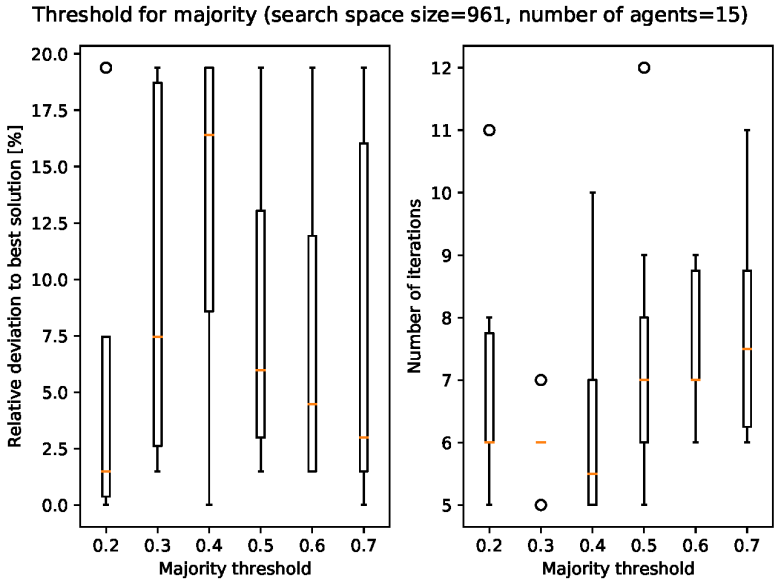

Fig. (16). Threshold for majority plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

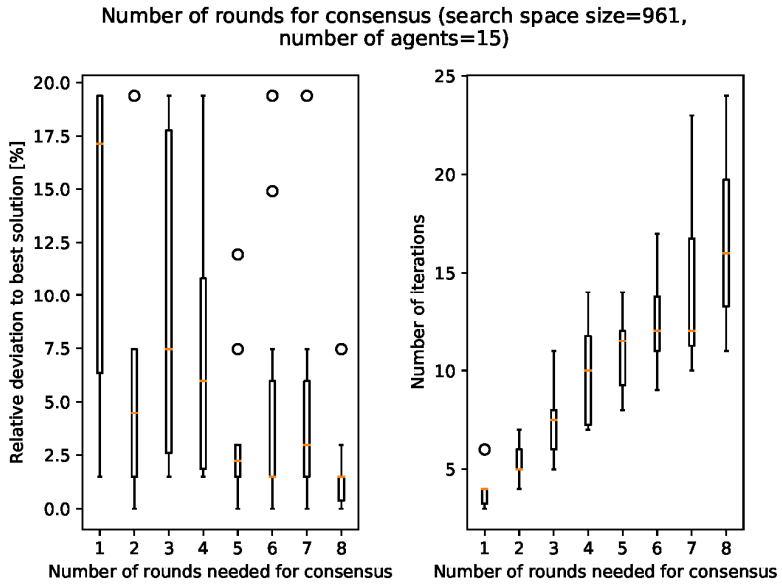

Fig. (17). Number of consensus plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

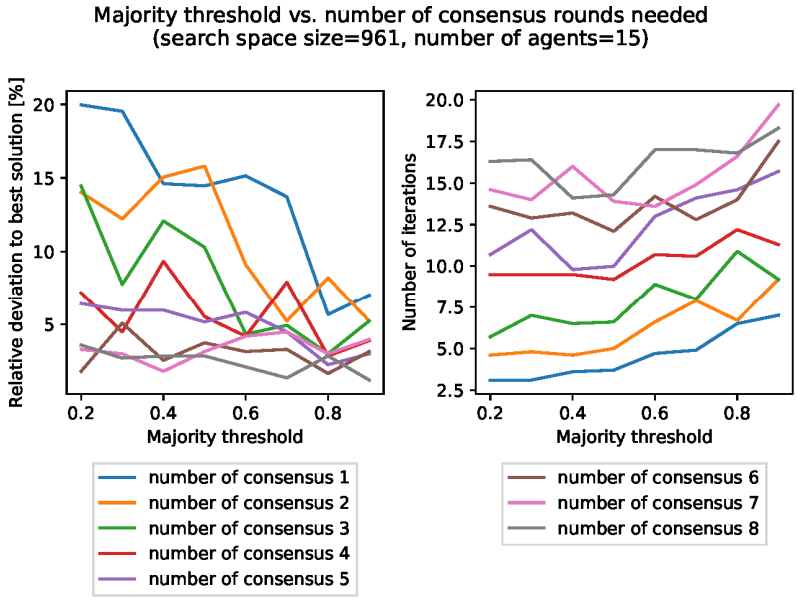

Fig. (18). Majority threshold with number of consensus plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

threshold is increased, there is a downwards trend in each possible combinations (Fig. **18**). We also notice that the larger the number of consensus rounds, the better the performance of the swarm. The number of consensus rounds [6, 7, 8] delivers similar performance, indicating that a higher number might not be useful. As for the number of iterations in the plot, we observe that each setup behaves similarly and increases the number of iterations regardless of the number of consensus.

When exploring the parameters of the algorithm the most compelling finding was its remarkable robustness against noise. In Fig. (**19**), it appears as if the algorithm does not succumb to noisy solutions, with a consistent median, with a slight trend upwards for noise > 10, albeit the speed seems to decrease at a faster pace. The numbers shown here are absolute values of the added noise to the quality score found by the agents.

CONCLUSION

Implementing the model described in [4], we tuned the parameters to test its accuracy, speed and robustness. We establish a baseline for the democratic honeybee algorithm and determine the most adequate parameters to ensure the algorithm's best performance. The algorithm is resistant to noise, and to a certain extent, to rogue agents. To improve performance, a memory has been implemented so that each agent remembers its previous best quality score and retains

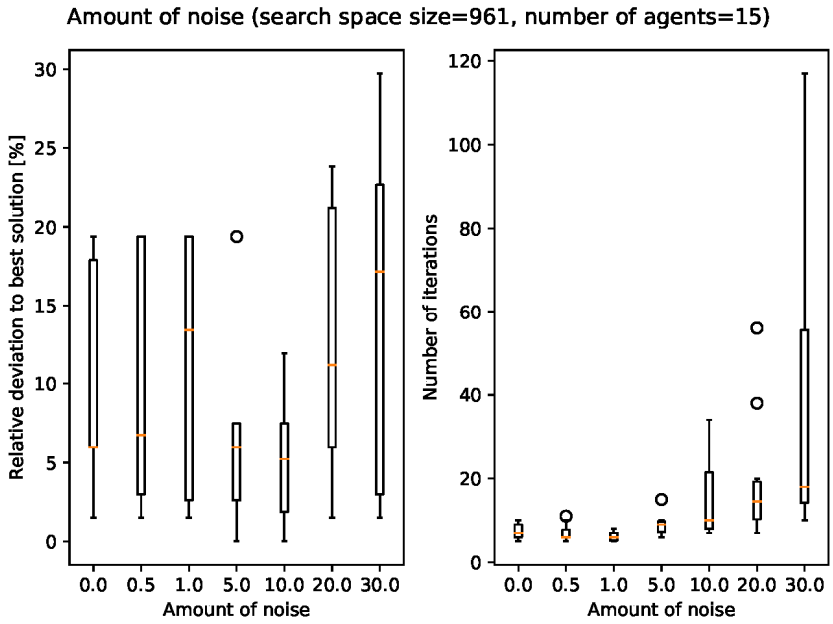

Fig. (19). Noise plotted against deviation from true best solution and number of iterations needed. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

it if no better one has been found in the current exploration phase. Using three visits for each agent with a neighborhood of two, we notice a certain tolerance towards rogue agents, which is most noticeable when the quality score advertised by the rogue agents is further away from the true best solution. Increasing, however, the number of allowed visits for each agent, the influence of rogue agents in the swarm is significantly reduced, but this leads to slower convergence. In the case of noise, we notice a slower convergence of the swarm to the global minima and a slightly reduced accuracy.

Nevertheless, the performance of the swarm is influenced by many parameters. As expected, the number of agents and the size of the search space have a positive and negative effect on the search, respectively. The number of visits proved to have a very low effect on the swarm, while the number of neighbors improved its efficiency, simultaneously reducing the required number of iterations. Moreover, in terms of the number of runs needed to reach consensus, positive results in terms of swarm accuracy were obtained as expected when more iterations were performed. We observe a similar trend, where the swarm appears to consent more frequently to the true best solution as the majority threshold increases. Lastly, regarding the number of visits and the size of the neighborhood, the swarm reacts very well, *i.e*. an increase of both parameters, leads to increased accuracy in finding the true best solution, even in large search spaces, already in a neighborhood of ten and 14 visits with an overall trend downwards.

For further investigations, functions other than the Rastrigin function could be used to test the parameters of the algorithm. It also would be interesting to build a dynamic small world topology over the swarm, which would dynamically reduce the number of connecting edges [16]. This idea combined with a network with trust edges [17], where a swarm would reduce its trust in a rogue agent when it spams bad positions, could significantly improve the robustness of the swarm. This hypothesis is based on the observation that our algorithm will never be robust against a number of rogue agents which is larger than half of the total number of agents. With such an implementation, the robustness of the swarm would become invariant to the number of rogue agents.

CONSENT FOR PUBLICATION

Not applicable.

AVAILABILITY OF DATA AND MATERIALS

Not applicable.

FUNDING

None.

CONFLICT OF INTEREST

Dr. Thomas Hanne is Editorial Advisory Board Member for the journal The Chinese Journal of Artificial Intelligence.

ACKNOWLEDGEMENTS

Declared none.

REFERENCES

[1] Fister, I.; Jr. Yang, X-S.; Fister, I.; Brest, J.; Fister, D. A brief review of nature-inspired algorithms for optimization. *Elektrotehniski Vestnik,* **2013**, *80*(3)*, 116-122*.

[2] Nicolis, S.C.; Dussutour, A. Self-organization, collective decision making and resource exploitation strategies in social insects. *Eur. Phys. J. B,* **2008**, *65*(3), 379-385. http://dx.doi.org/10.1140/epjb/e2008-00334-3

[3] Valentini, G.; Ferrante, E.; Dorigo, M. The best-of-n problem in robot swarms: Formalization, state of the art, and novel perspectives. *Front. Robot. AI,* **2017**, *4*, 9.

http://dx.doi.org/10.3389/frobt.2017.00009

[4] Pochon, Y.; Dornberger, R.; Zhong, V.J.; Korkut, S. *Investigating the Democracy Behavior of Swarm Robots in the Case of a Best-of-n Selection,* In: *IEEE Symposium Series on Computational Intelligence (SSCI)*, Bangalore, India, November **2018**, pp. 743-748. http://dx.doi.org/10.1109/SSCI.2018.8628646

[5] Seeley, T.D. Honeybee democracy, *In Honeybee Democracy*. Princeton University Press: Princeton, **2010**.

[6] Strobel, V.; Castelló Ferrer, E.; Dorigo, M. Managing byzantine robots *via* blockchain technologyin a swarm robotics collective decision making scenario. In: *AAMAS '18: Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems,* Stockholm, Sweden; July 10-15, **2018**, 541-549.

[7] Strobel, V.; Dorigo, M. Blockchain technology for robot swarms: A shared knowledge and reputation management system for collective estimation. IRIDIA - Technical Report Series, Technical Report No. TR/IRIDIA/2018-009, May **2018**, pp. 1-12.

[8] Ebert, J.T.; Gauci, M.; Nagpal, R. Multi-feature collective decision making in robot swarms. In: *AAMAS '18: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems,* Stockholm, Sweden; July 10-15, **2018**, 1711-1719.

[9] Petrenko, V.I.; Tebueva, F.B.; Ryabtsev, S.S.; Struchkov, I.V. Consensus achievement method for a robotic swarm about the most frequently feature of an environment. *IOP Conf. Ser.: Mater. Sci. Eng.,* **2020**, *919*, 042025. http://dx.doi.org/10.1088/1757-899X/919/4/042025

[10] Valentini, G. How robots in a large group make decisions as a whole? From biological inspiration to the design of distributed algorithms. *arXiv:1910.11262v2,* **2019**.

[11] Maître, G.; Tuci, E.; Ferrante, E. Opinion dissemination in a swarm of simulated robots with stubborn agents: A comparative study. In: *2020 IEEE Congress on Evolutionary Computation (CEC)*, **2020**, 1-6. http://dx.doi.org/10.1109/CEC48606.2020.9185581

[12] Ramsey, M.T.; Bencsik, M.; Newton, M.I.; Reyes, M.; Pioz, M.; Crauser, D.; Delso, N.S.; Le Conte, Y. The prediction of swarming in honeybee colonies using vibrational spectra. *Sci. Rep.,* **2020**, *10*(1), 9798. http://dx.doi.org/10.1038/s41598-020-66115-5 PMID: 32546693

[13] Ebert, J.T.; Gauci, M.; Mallmann-Trenn, F.; Nagpa, R. Bayes Bots: Collective Bayesian decision-making in decentralized robot swarms. In: *2020 IEEE International Conference on Robotics and Automation (ICRA),* Paris, France **2020,** 7186-7192. http://dx.doi.org/10.1109/ICRA40945.2020.9196584

[14] Aysal, T.C.; Barner, K.E. Convergence of consensus models with stochastic disturbances. *IEEE Trans. Inf. Theor.,* **2010**, *56*(8), 4101-4113. http://dx.doi.org/10.1109/TIT.2010.2050940

[15] Hansen, N.; Finck, S.; Ros, R.; Auger, A. *Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions*. *Ph.D Thesis, INRIA, France,* **2009**.

[16] Liu, Q.; van Wyk, B.J.; Du, S.; Sun, Y. Dynamic small world network topology for particle swarm optimization. *Int. J. Pattern Recognit. Artif. Intell.,* **2016**, *30*(09), 1660009. http://dx.doi.org/10.1142/S0218001416600090

[17] Rodriguez, M.A.; Steinbock, D.J.; Watkins, J.H.; Gershenson, C.; Bollen, J.; Grey, V.; deGraf, B. *Smartocracy: Social networks for collective decision making.* In*:* 40th Annual *Hawaii International Conference on System Sciences (HICSS'07)*, Waikoloa, HI, Hawaii, **2007**, 90.