Scalability and Robustness of Ant Search in Edge-Fog-Cloud Distributed Knowledge Graphs

Oleksandr Chepizhko[1,*], Péter Forgács[1] and Melanie Schranz[1]

[1]Lakeside Labs GmbH, 9020, Klagenfurt, Austria

Abstract

This paper investigates the scalability of a variant of the well-known Ant Colony Optimization (ACO) algorithm for search operations in a distributed knowledge graph, with a focus on the algorithm's performance under increasing query loads. The data is represented as RDF triples, and the underlying network topology models a typical edge–fog–cloud architecture, reflecting scenarios relevant to the edge-cloud domain. Assuming nodes with infinite processing capacity, we show that the hit rate, the primary performance metric, initially increases with query frequency and eventually saturates. In contrast, when node processing capacity is finite, the hit rate exhibits a critical threshold after which performance declines. We analyze the behavior of the system under different query regimes, focusing on network utilization and pheromone-level dynamics. Finally, we discuss the feasibility of implementing such a system in practice and its potential implications for real-world applications.

Keywords

Edge-Fog-Cloud Computing, Ant Colony Optimization, Distributed Knowledge Graphs, Scalability in Swarms

1. Introduction

The Edge-Fog-Cloud continuum represents a transformative paradigm in distributed computing, driven by the increase of complexity in IoT (Internet of Things) devices, its number in a running system, and data-intesive applications to be processed. Traditional cloud-centric models face key limitations, such as latency, privacy concerns, and energy inefficiency. Distributed computation paradigms are closer to the data sources in the edge and fog layers, thus enhancing overall system performance and responsiveness [1]. The edge layer, in particular, offers critical advantages that include increased security, reduced latency and energy consumption, support for autonomous operations, and adaptability to dynamic data volumes and heterogeneous users [2].

Robustness in such distributed environments is essential due to the dynamic and often unpredictable nature of edge-fog-cloud networks. Systems must withstand challenges posed by malicious agents that can disrupt data integrity or routing, nodes that fail unexpectedly, intermittent link breaks, and the continuous addition or removal of nodes reflecting network mobility and scaling. These factors create a volatile environment where search and data management operations must adapt to maintain efficiency and accuracy. Testing for node movement and network changes is thus critical to evaluate the resilience of algorithms and protocols designed for this continuum. The search in dynamic environments of distributed knowledge graphs requires scalable, robust, and efficient solutions capable of handling such uncertainties [1, 2].

1.1. Contribution

Ant Colony Optimization (ACO), inspired by the foraging behavior of ants, offers a promising approach to address these challenges. By leveraging decentralized, self-organizing mechanisms based on pheromone trails, ACO algorithms can dynamically adapt to network changes, optimize routing paths, and maintain high search performance even under varying loads and network conditions [3, 4]. This paper investigates

CPS Workshop 2025, September 22, 2025, Italy

*Corresponding author.

chepizhko@lakeside-labs.com (O. Chepizhko); forgacs@lakeside-labs.com (P. Forgács); schranz@lakeside-labs.com (M. Schranz)

0000-0003-3001-0827 (O. Chepizhko); 0000-0001-6031-6291 (P. Forgács); 0000-0002-0714-6569 (M. Schranz)

the scalability and robustness of an ACO variant applied to search operations over Resource Description Framework (RDF) triple-based knowledge graphs distributed across an edge-fog-cloud architecture, analyzing performance under different query loads and network dynamics.

1.2. Outline

The paper is organized as follows: Section 2 gives a general description of the problem and a detailed system model of the DKG in the edge-fog-cloud continuum. Section 3 summarizes the related work for RDF (Resource Description Framework), ACO and scalability analyses in swarms. Section 4 presents the main principles for the ACO-related pheromone search in the network. The evaluation of the hit rate and network utilization is described in Section 5 forming the base for the actual scalability analysis in Section 6. The paper is concluded in Section 7.

2. Background

2.1. Problem Statement

We study search and data movement on a distributed knowledge graph (DKG) using the ACO algorithm. The architecture of our knowledge graph is reflecting an example of the edge-fog-cloud continuum. We assess the complex system properties, such as scalability, fault tolerance and robustness, and adaptability. Our approach is compared with other known search algorithms with respect to their efficiency.

2.2. System Model

The edge-fog-cloud continuum is modeled as a hierarchical network that dynamically maps different slices of the DKG across the edge, fog, and cloud layers. An example of a hierarchical network is shown in Figure 1, where the proportion of edge-fog-cloud nodes is $5 : 4 : 1$. The DKG is a novel approach to knowledge representation and data management [5], and provides a real-time perspective of relevant data distributed throughout the network. Physically, each part of the DKG is stored on a device such as an IoT device, or a (micro) data center, forming the edge-fog-cloud continuum. The DKG is represented using the RDF, which stores data as a directed graph. RDF expresses data in the form of triples: subject-predicate-object [6]. In this work, the DKG triples are distributed across various devices, without a central overview. Therefore, we model the DKG as an undirected graph $G(V, E)$, where the vertices V represent the physical nodes $\mathcal{N} = \{n_1, n_2, \ldots, n_N\}$, and the edges E represent the (physical) connections between the nodes $\mathcal{N}$, forming the neighborhoods $\aleph \subseteq \mathcal{N} : \Longleftrightarrow \ \forall n \in \aleph : n \in \mathcal{N}$. The graph is finite, but dynamic, as nodes can be added or removed at runtime. For this study, we consider a constant number of nodes. The data partitions $\mathcal{P} = \{p_1, p_2, \ldots, p_j, \ldots, p_N\}$ are distributed over the nodes $\mathcal{N}$, where each partition consists of a set of triples $p_j = \bigcup_{i=1}^{|p_j|} \langle t_{ij}^s, t_{ij}^p, t_{ij}^o \rangle$, with $|p_j|$ denoting the number of triples on the node n_j. A query Q represents a pattern in which one or more positions of the triple subject-predicate-object are replaced by a wildcard (an asterisk). For example, a query for an unknown subject with a known predicate p and an object o can be formalized as $Q_s = \langle *, p, o \rangle$. Other types of query follow the same logic. If a node n_j contains at least one triple that matches the query pattern, we call it a matching node m_j. For each query, there exists a set of all matching nodes, $\mathcal{M} = \{m_1, m_2, \ldots, m_M\}$, where $\mathcal{M} \subseteq \mathcal{N}$ and $M = |\mathcal{M}|$ is the number of matching nodes in the network. If no matching triples are present on any node, then the set is empty $\mathcal{M} = \emptyset$. A node can hold more than one matching triple.

Each query Q maps to a keyword k derived from its predicate p and object o, typically via string concatenation or hashing. This keyword uniquely identifies a pheromone in the ACO system.

Each node n_i maintains: a list of direct neighbors $\aleph_i$; a record of forward ants $\mathcal{F}_i$, tracking which ants have visited; a pheromone table $\mathcal{T}_i = \bigcup_{j=1}^{|\aleph_i|} \lambda_{ij}^{(k)}$, storing pheromone values for each keyword k toward neighbor n_j; a link cost table $\mathcal{L}_i = \bigcup_{j=1}^{|\aleph_i|} \eta_{ij}$, listing the costs to each neighbor.

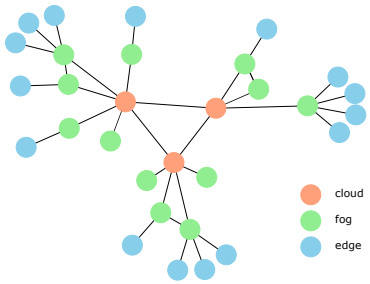

Figure 1: A random network containing 30 nodes in 3 layers. The first layer (cloud) has 3 (red), the second layer (fog) has 12 nodes (green). The nodes in the second layer with the same parent have a 0.3 probability of being interconnected. The third layer (edge) has 15 nodes (blue).

3. Related Work

Processing large-scale RDF datasets [7] presents significant challenges that demand modern algorithmic techniques and optimization strategies, given the ever increasing data volumes. Efficient slicing and querying of such vast knowledge bases can be achieved through algorithmic improvements [8]. Several enhancements to these engines have been proposed [9]; for instance, leveraging the FLINK framework has shown promise for processing distributed RDF stores [10]. In addition, advanced methods such as evolutionary algorithms have been explored to support efficient search in DKG [11], and the use of ACO techniques has also been suggested.

Ant optimization is a robust model introduced in seminal works [12, 13]. By employing virtual ants that navigate a network in search of “food" and deposit pheromone trails, optimal routes between key resources can be swiftly identified. ACO algorithms represent problems as graphs, where each solution corresponds to a path. The cost of a solution is determined by the cumulative costs of the path’s edges, with the objective being to discover the path with the minimum cost. This algorithm has demonstrated success across diverse applications and variations, including vehicle routing [14], vehicle routing with pickup and delivery [15], and job-shop scheduling [16]. Also, ACO has been proven to be successful when used for dynamic routing for mobile networks [17]. Ant optimization has also been applied effectively within the realm of knowledge graphs. For example, the use of ant optimization algorithms has been explored to support chained queries for large RDF datasets [18]. We would like to utilize it for the search for data in DKGs, as already proposed by the authors in [3, 4].

The properties of complex systems, such as scalability, robustness, or fault tolerance, require precise formulations and measurements [19, 20]. The question of scalability both in computing and in robotics has attracted much research attention [21]. In these studies, it has been shown that the performance of a swarming system may increase (to a limit) with increasing the number of involved agents and that this is an important characteristic of a self-organized swarm system. We are interested to study whether the ACO algorithm for DKG shows this important property.

4. Pheromone-based Search

The ACO-based search algorithm operates through two primary mechanisms: **trail following** and **trail laying**.

Trail following When a query reaches a node n_i, it generates a *forward ant* that follows one of two strategies: *exploration* or *exploitation*. The ant randomly selects between these strategies with probability w. In the exploration strategy, the ant probabilistically selects its next node $n_j \in \aleph_i$ based

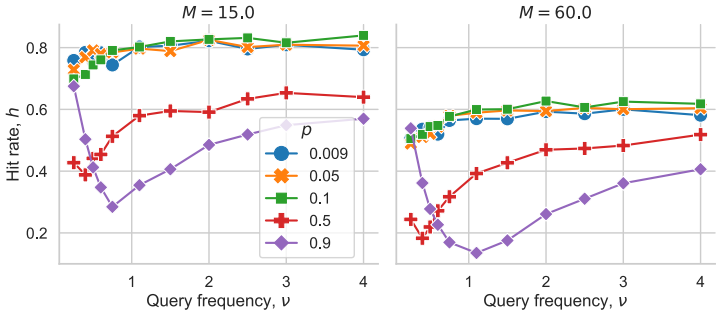

Figure 2: Steady state hit rate as function of query frequency ν for different values of the evaporation coefficient p, given in the legend, for two different values of matching nodes number, M. Total number of the nodes in the network, $N = 150$. Data averaged over 100 simulation runs.

on local pheromone levels $\lambda_{ij}^{(k)}$ and link costs η_{ij}, as defined by:

$$P_{ij}^{(k)} = \frac{{\lambda_{ij}^{(k)}}^{\beta} \cdot \eta_{ij}}{\sum_{l=1}^{|\aleph_i|} {\lambda_{il}^{(k)}}^{\beta} \cdot \eta_{il}}, \tag{1}$$

where β controls how strongly the pheromone level influences the decision. In this study, we set $\beta = 1$ and the link costs to 1. Future extensions could dynamically adjust these costs based on experience, e.g., increasing the cost for links with high latency or low computational performance. The probability in Eq. (1) is calculated across all neighbors of n_i, meaning that the forward ant can split and send clones to multiple neighbors simultaneously. In the exploitation strategy, the ant selects all neighbors n_j satisfying:

$$\lambda_{ij}^{(k)} \geq \frac{1}{|\aleph_i|} \sum_{l=1}^{|\aleph_i|} \lambda_{il}^{(k)}, \tag{2}$$

thus focusing only on the best-performing options. In this study, the probability of choosing either strategy is set to 0.5. When a forward ant arrives at node n_j, it checks the forward ant record of the node ($\mathcal{F}_j$) and terminates if a clone of the same ant has already visited that node, preventing redundant exploration.

Trail laying As the query traverses the network, a *backward ant* is launched to retrace the route and update pheromone levels, reinforcing successful paths. The pheromone value at node n_x is updated as: $\lambda_{xj}^{(k)} \leftarrow \lambda_{xj}^{(k)} + Z$, where $Z = w_d \frac{M}{M_0} + (1 - w_d) \cdot \frac{t_0}{2 \cdot t_{qr}}$. Here, w_d is the weighting factor (with $w_d = 0.5$ in this study), M is the number of matching triples, $M_0 = 10$ and $t_0 = 3$ are constants, and t_{qr} is the query response time. This ensures that the amount of pheromones is proportional to both the quality of the resources found and the efficiency of the path. Pheromone trails evaporate over time with an evaporation factor $p \in [0, 1)$, gradually removing less optimal paths: $\lambda(t + 1) = \lambda(t)(1 - p)$.

5. Hit Rate and Network Utilization

Hit Rate The global hit rate is a key performance metric as it reflects the proportion of matching nodes found on average. Long-time behavior shows that time-dependent hit rate reaches a steady state, $h := \lim_{t \to \infty} \langle h(t) \rangle$, where $\langle \dots \rangle$ stands for the average over different simulation runs. In Figure 2 we report h for two different matching node numbers: $M = 15$ and $M = 60$, with a total number of nodes $N = 150$. For each of them we consider the curves $h(\nu)$ for selected values of evaporation coefficient p: we consider three small values, $p = 0.009, 0.05, 0.1$, one intermediate, $p = 0.5$, and one large, $p = 0.9$. For small values of p, we see high hit rates, which increase only very slightly with ν and decrease steadily with M. The behavior at intermediate values of p is notable: it begins with low hit rates at

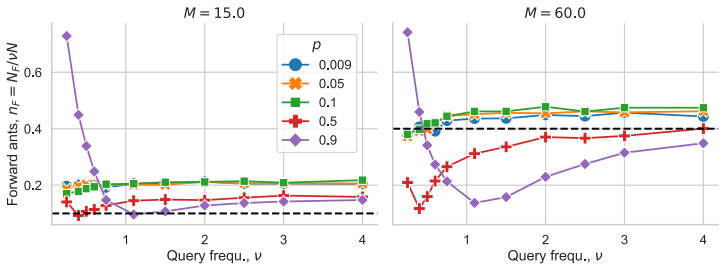

Figure 3: Reduced forward ants number n_F as function of ν for different p and M. Black dashed line shows a baseline value of M/N.

small ν, but increases rapidly as the query frequency increases. At the largest p, it exhibits quite large hit rates, comparable to low p values. Then, as ν increases, the hit rate quickly drops. Further increase of ν leads to a steady growth of the hit rate, again associated with the crossing of a critical threshold. This drastic difference in $h(\nu)$ behavior can be explained if we refer to the characteristic time, $\tau = 1/p$, after which a significant quantity of the pheromone evaporates. For $p \leq 0.1$, it is $\tau \geq 10$ ticks, which is enough to traverse the largest distance in the network, $n = 5$, twice. This ensures a stable pheromone network and good performance reflected in hit rate even for low query frequencies. However, for $p = 0.5$, the characteristic time is $\tau = 2$, which is small, but might be enough for the pheromones to stay on the short paths with the situation improving when the query frequency increases. For $p = 0.9$ the characteristic time $\tau \approx 1.11$. This leads to a very quick pheromone evaporation. Higher frequencies are beneficial for the pheromone routes as they allow pheromone tables to be updated many times per time step to resist evaporation.

Forward Ants We measure the average number of forward ants $N_F(t)$ present in the system at each time step. This quantity serves as a proxy for network utilization, as each ant can be interpreted as a message sent between nodes. To make things comparable, we introduce relative number of forward ants, $n_F(t) := \frac{N_F(t)}{\nu N}$, we normalize the number of forward ants $N_F(t)$ by the number of nodes N and by query frequency ν. We measure the long-term average values of $n_F(t)$ and show them for different parameter values in Figure 3. The curves for small $p \leq 0.1$ values all follow each other at the value of $n_f \approx 0.2$ for $M = 15$ and the value of $n_f \approx 0.45$ for $M = 60$. This can be compared with a baseline value of M/N. For a small number of matching nodes, $M = 15$, we see that the actual values of the forward ants are twice higher. For large $M = 60$, the value of n_f is much closer to the baseline. The intermediate value, $p = 0.5$, shows a consistently small number of forward ants, which grows with ν. For the case of $M = 15$, the $n_f(\nu)$ values are still higher than the baseline. The behavior of the curve $n_f(\nu)$ for the highest value of the evaporation coefficient, $p = 0.9$, is the most peculiar. At first, we see a large number of forward ants, especially when M is relatively small. Thus, one can speculate that the relatively higher hit rate for $p = 0.9$ at low ν is connected to the fact that many forward ants are created, while it is relatively low for $p = 0.5$, so there are very few forward ants. At low query frequency and large evaporation coefficient, we might see the behavior of the algorithm where it produces too many forward ants.

6. Scalability

We follow [19] for scalability and adopt the general formula $S := \frac{\%\Delta P}{\%\Delta N}$, where $\%\Delta P$ is the change in performance and $\%\Delta N$ is the change in agent number. Our performance is measured as the hit rate h. Then, we can use ν as a proxy for the number of agents N. We then re-write the formula as $S = \frac{(h_{\nu+\delta\nu} - h_\nu)/h_\nu}{\delta\nu/\nu}$.

Scalability, S exhibits a different behavior for different sets of parameters (see Figure 4). We know that for low p values, there is almost no dependence of h on ν, so scalability stays around zero in

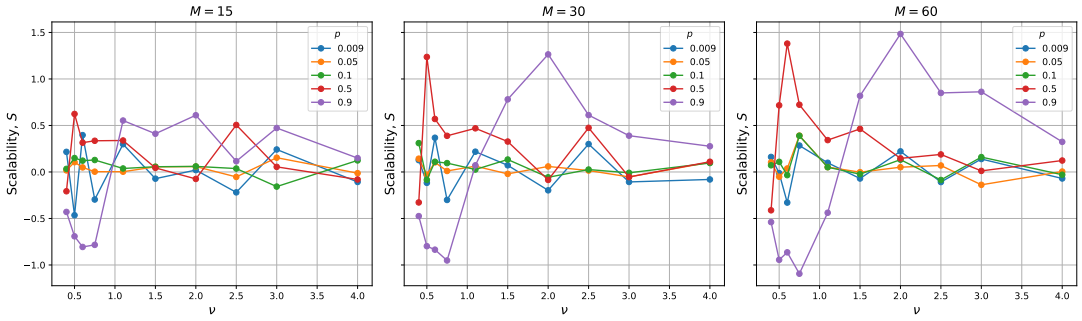

Figure 4: Scalability of the system, S, for increasing frequency of queries, ν. Case of infinite capacity. Three numbers of matching nodes are considered, $M = 15, 30, 60$. Total number of nodes $N = 150$. Different values of evaporation coefficient are given in the legend.

this case. Since the system already performs well at low values of ν, increasing the query frequency does not produce significant performance gains. Furthermore, because nodes are assumed to have infinite processing capacity, higher values of ν do not lead to any performance degradation. For the intermediate value of $p = 0.5$ we see that scalability goes above zero and in some cases even above one as ν increases highlighting the emergence of the pheromone routes which facilitate the search. For the highest value of the evaporation coefficient $p = 0.9$, the picture is even more interesting. As we know, for very little ν values a large number of forward ants is present, resulting in relatively high values of h and overall performance of the search algorithm. Thus, increasing ν marks a transition from many to few forward ants, while the routes are not stable enough to guide the ants. Then, with increasing ν, stable pheromone routes start to appear, and the scalability not only becomes positive but also $S > 1$, indicating super-critical growth. For even higher values of ν, the scalability steadily approaches zero, which means that the system is reaching its peak performance.

7. Conclusion

In this work, we applied the Ant Colony Optimization (ACO) algorithm to the problem of searching in distributed knowledge graphs (DKGs) structured according to an edge–fog–cloud continuum. We demonstrated that in systems with infinite processing capacity, increasing the query frequency leads to more efficient pheromone trails and improved hit rates. In particular, in settings with a high evaporation coefficient, we observed an emergent scalability effect, where higher query frequency reinforced pheromone-based routing.

Future work will examine this hypothesis, along with the model's robustness and fault tolerance, such as resilience to node or link failures or to errors in pheromone signal interpretation. Another key direction is the practical implementation of this search mechanism in real-world edge–cloud infrastructures. Our ongoing work focuses on implementing ACO using forward and backward ants as messages exchanged between nodes, dynamically updating local pheromone tables.

Acknowledgments

Funded by the European Union, project GLACIATION, by grant No. 101070141. Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Declaration on Generative AI

The authors used ChatGPT for grammar and spelling checks. All content was subsequently reviewed and edited by the authors, who take full responsibility for the final publication.

References

[1] F. Firouzi, B. Farahani, A. Marinšek, The convergence and interplay of edge, fog, and cloud in the AI-driven internet of things (IoT), Journal of Systems Architecture 122 (2022) 102193.

[2] A. O'Mahony, F. Buining, P. Forgács, M. Schranz, Swarm intelligence for green data orchestration - a vision for energy-efficient computing in the edge continuum, in: Proceedings of the Edge AI meets Swarm Intelligence Technical Workshop, 2024, pp. 1–11.

[3] O. Chepizhko, P. Forgacs, M. Schranz, Ant-search algorithm for distributed knowledge graphs, in: Proceedings of the 14th International Conference on Swarm Intelligence, Springer, 2024.

[4] O. Chepizhko, P. Forgacs, M. Schranz, Ant colony optimization for data search and movement in the edge-fog-cloud continuum, in: Proceedings of the 9th IEEE International Conference on Fog and Edge Computing 2025, 2025, p. 5.

[5] D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, A. Wahler, Knowledge Graphs: Methodology, Tools and Selected Use Cases, Springer International Publishing, 2020.

[6] N. Gibbins, N. Shadbolt, Resource description framework (RDF), Wide Web Consortium 2004 (2009) 4539–4547.

[7] W. Ali, M. Saleem, B. Yao, A. Hogan, A.-C. N. Ngomo, A survey of rdf stores: Sparql engines for querying knowledge graphs, The International Journal on Very Large Data Bases 31 (2021) 1–26.

[8] E. Marx, S. Shekarpour, S. Auer, A.-C. Ngonga Ngomo, Large-scale RDF dataset slicing, in: Proceedings of the 7th IEEE International Conference on Semantic Computing, 2013, pp. 228–235.

[9] A. Potter, B. Motik, Y. Nenov, I. Horrocks, Dynamic data exchange in distributed RDF stores, IEEE Transactions on Knowledge and Data Engineering 30 (2018) 2312–2325.

[10] A. Azzam, S. Kirrane, A. Polleres, Towards making distributed RDF processing FLINKer, Innovate Data (2018).

[11] C. Gueret, S. Schlobach, K. Dentler, M. Schut, G. Eiben, Evolutionary and swarm computing for the semantic web, IEEE Computational Intelligence Magazine 7 (2012) 16–31.

[12] M. Dorigo, L. M. Gambardella, Ant colonies for the travelling salesman problem, biosystems 43 (1997) 73–81.

[13] M. Dorigo, T. Stützle, The ant colony optimization metaheuristic: Algorithms, applications, and advances, Handbook of metaheuristics (2003) 250–285.

[14] Y. Kao, M.-H. Chen, Y.-T. Huang, et al., A hybrid algorithm based on ACO and PSO for capacitated vehicle routing problems, Mathematical Problems in Engineering 2012 (2012).

[15] H. Wu, Y. Gao, An ant colony optimization based on local search for the vehicle routing problem with simultaneous pickup–delivery and time window, Applied Soft Computing 139 (2023) 110203.

[16] S. Zhang, T. N. Wong, Flexible job-shop scheduling/rescheduling in dynamic environment: a hybrid MAS/ACO approach, International Journal of Production Research 55 (2017) 3173–3196.

[17] S. Chatterjee, S. Das, Ant colony optimization based enhanced dynamic source routing algorithm for mobile ad-hoc network, Information Sciences 295 (2015) 67–90.

[18] A. Hogenboom, F. Frasincar, U. Kaymak, Ant colony optimization for RDF chain queries for decision support, Expert Systems with Applications 40 (2013) 1555–1563. doi:http://dx.doi.org/10.1016/j.eswa.2012.08.074.

[19] E. Milner, M. Sooriyabandara, S. Hauert, Swarm performance indicators: Metrics for robustness, fault tolerance, scalability and adaptability, 2023. ArXiv:2311.01944.

[20] G. M. Madroñero Pachajoa, W. Achicanoy, D. G. Ramos, Automating the evaluation of the scalability, flexibility, and robustness of collective behaviors for robot swarms, in: Proceedings of the 2024 Brazilian Symposium on Robotics, and 2024 Workshop on Robotics in Education, IEEE, 2024, p. 144–149.

[21] H. Hamann, A. Reina, Scalability in computing and robotics, IEEE Transactions on Computers 71 (2022) 1453–1465.