International Journal of Informatics and Applied Mathematics
e-ISSN:2667-6990 Vol. 2, No. 1, 37-47

A Multiple-Place Algorithm for Sustainable Foraging Scenarios

Ouarda Zedadra[1] and Abderahmane Benkirat[2]

[1] LabSTIC, Department of Computer Science, 8 Mai 1945 University. P.O.Box 401, 24000 Guelma, Algeria
zedadra.ouarda@univ-guelma.dz

[2] Department of Computer Science, 8 Mai 1945 University. P.O.Box 401, 24000 Guelma, Algeria
haydaradz24@gmail.com

Abstract. This paper presents a Multi-Place Foraging algorithm called Lévy Walk and Firefly Recruiting Algorithm (LWFR). Unlike most of the studies on foraging available in the literature, the proposed foraging robots aim to maintain the survivability of their nests and collaborate to maintain the survivability of other depots when needed. The algorithm uses: (1) Lévy Walk to search for objects;(2) Firefly algorithm to attract robots in neighborhood. The attraction model, inspired by the behavior of Fireflies, provides an indirect and costless communication. Numerical simulations show that the proposed algorithm can maintain the survivability of different nests.

Keywords: Swarm Intelligence · Swarm Robotics · Central-Place Foraging · Multiple-Place Foraging · Survivability of nests

1 Introduction

Swarm Robotics is a new approach to the coordination of large numbers of robots whose main inspiration stems from the observation of social insects [1]. It has

emerged as the application of swarm intelligence (SI) to Multi-Robot Systems (MRS). Swarm robotics emphases on the physical embodiment of individuals and realistic interactions among the individuals, and between the individuals and the environment. It is characterized by the emergence of a synchronized behavior at system level, which emerges despite the individuals being relatively incapable, the lack of centralized coordination and the simplicity of interactions [2].

Foraging is considered as the main benchmark in the field of swarm robotics. It is known as the act of searching for objects and when found, transport them to a common location. Robot foraging is considered an effective metaphor for addressing complex real-world applications such as target detection, object transportation, motion planning and different agricultural tasks. Foraging related works use either one central nest (Central Place Foraging) or multiple nests (Multiple Place Foraging). Central-Place Foraging (CPF) uses a centrally-placed nest from which robots depart and return as they collect resources [3]. The large study of CPF in literature provides efficient strategies for different resource distributions. Unfortunately, inter-robot collisions and long paths travelled to the central-place decrease the foraging efficiency and affect the scalability when increasing robots number and environment size. Multiple Place Foraging (MPF) uses multiple nests that robots depart from and return to. It is inspired by behaviors observed in biology. For example, the polydomous colonies of Argentine ants are comprised of multiple nests spanning hundreds of square meters. In addition, spider monkeys has been characterized as multiple central place foragers [3].

Previous research studies have focused on the design of robotic systems with a single nest and multiple nests. Most of classical foraging works do not consider some pillar characteristics such as the sustainability of the system. In such scenarios, the foraging system is required to satisfy external demands. Thus, robots need to maintain high levels of food in their nests. This paper presents an MPF algorithm which considers the sustainability of the system. Robots in the proposed algorithm use an attraction model inspired by the behavior of fireflies to recruit other robots when their nest is in a critical situation.

The reminder of the paper is organized as follows: Section 2 describes several related works. Then, Section 3 presents the flowchart and the pseudo code of the proposed algorithm in, while experimental results are discussed in Section 4. Finally, Section 5 draws conclusions and provides several future perspectives.

2 Related Works

Castello et al. [4] proposed the Adaptive Response Threshold Model (ARTM), an extension of the Fixed Response Threshold Model (FRTM) [5], in which the response threshold is calculated dynamically rather than remaining fixed. ARTM uses the level of food shortage as stimulus. The behavior of a robot is composed of three states: (1) *wait*: At the beginning of the foraging mission all robots are in *wait* state. The robot will initialize a timer that is used to control the

intensity of stimuli. When the timer expires, robots will sense the current value of stimuli and will calculate *probability of foraging* which allows it to go look for food tokens and meet the demands of an external system that extracts food from the swarm nest reserves at a given rate or to remain in *wait* state, (2) *search*: Robot starts a random search for food tokens. If the robot finds a food token, it switches to *collect* state, otherwise it continues searching, (3) *collect*: Once a food is obtained, the robot returns home. it deposits food token into the nest and switches to the *wait* state. A certain amount of food tokens were randomly scattered within the simulation area. Every food token was replaced once it was collected, maintaining the food distribution always constant. The system aims to maintain a positive level of food at the nest during the whole foraging mission. The efficiency and adaptability of ARTM were tested against the fixed response threshold model RTM.

Castello et al. [6] presented an in depth explanation of the ARTM [4]. The authors studied the adaptability of the model through computer simulations. In the CPFA proposed in [7], foraging performance decreases as swarm size and search areas scale up. In fact, increasing robot's number produces more inter-robot collisions and larger search areas produce longer travel distances. To overcome the aforementioned problems, Lu et al. [3] proposed the Multiple-Place Foraging Algorithm static ($MPFA_{static}$). The authors proposed to use multiple depots rather than a single one. Robots were initially assigned in equal numbers to static nests. Each nest is placed at the center of one quadrant of a foraging arena with $1/4$ of the robots assigned to each nest. The robots return and deliver targets to the closest nest, then they return to the position of the found target. Different from CPFA, pheromone waypoints are only reported to the closest depot. The main difference to the CPFA algorithm is the use of multiple nest rather than a single one.

Lu et al. [8] proposed an extension of the $MPFA_{static}$ [3] the Multiple-Place Foraging Algorithm with dynamic depots ($MPFA_{dynamic}$). The main difference with $MPFA_{static}$ [3] is that depots are dynamic and change their position according to targets distribution. Depots are special robots initially distributed in the search area and can carry multiple targets. The spatially distributed design reduces robot transport time and reduces collisions among robots. Robots depart from a depot to forage for targets and then return to the closest depot to deliver these targets (the closest depot may be different from the one the robot departed from). Depots move to new locations based on the mean positions of the remaining targets sensed by robots. The positions of the sensed targets are stored at each depot when each robot returns to that depot.

For a successful deployment of foraging robots in real world, a new characteristic of real world applications is considered in foraging works. This characteristic is the sustainability of the system, where the system needs to serve external demands on the basis of objects deposed in the nests. Thus, robots must maintain high levels of objects in their nest or other nests.

3 Proposed algorithm

This paper presents an MPF algorithm called Lévy Walk Recruiting Algorithm (LWFR). The proposed algorithm considers demands provided by external systems. It is a hybridizing of two bi-inspired algorithms: Lévy Walk (LW) [9] and Firefly (FF) [10]. The former is used to explore the search space, while the latter is used as recruiting model when collaboration is needed by a specific nest. The flowchart illustrated in Figure 1 represents the behavior of LWFR robots, where, continuous and dotted lines represent respectively positive and negative responses. A textual description of the states of the proposed algorithm, its parameters and the pseudo code of corresponding states are given in below.

- *Start*: in this state all the robots are around the nest with intensity *I=0*. Depots are equidistant from each other and have the same quantity of food $F(t_0)$. To simulate the extraction of an extern system, the $F(t_0)$ decreases in time with predefined value. Objects are distributed randomly or in clusters in the search space.
- *Wait*: in this state the robot is waiting in the nest for an activation to *search state*. Each time, the robot calculates the decision function F_d using Equation 2 according to the stimulus in the nest. Equation 1 was used to calculate the stimulus. The pseudo code of this state is given by algorithm 1

$$S(t) = F(t) - F(t_0) \tag{1}$$

$$F_d = \begin{cases} Wait \quad if & S(t) < 0 \\ Search \ if & 0,2 \times F(t_0) < S(t) < 0,4 \times F(t_0) \\ Recruit \ if & S(t) > 0,4 \times F(t_0) \end{cases} \tag{2}$$

```
repeat
    Stay at Nest;
    t=t+1;
until (Fd)(t) ≠ waiting);
if (Fd(t) = search) then
    Goto Search (Algorithm 2);
```

Algorithm 1: Wait

- *Search*: In this state, the robot starts searching its space using Lévy Walk search strategy, where the random steps are generated according to Equation 3. Each robot can carry one object at a time. It can communicate with a robot in its neighborhood if this latter has *I>0* and belongs to another nest. Then, the robot moves towards the other robot and changes to *Recruit* state, else it changes to *Collect* state. The pseudo code of this state is given by algorithm 2

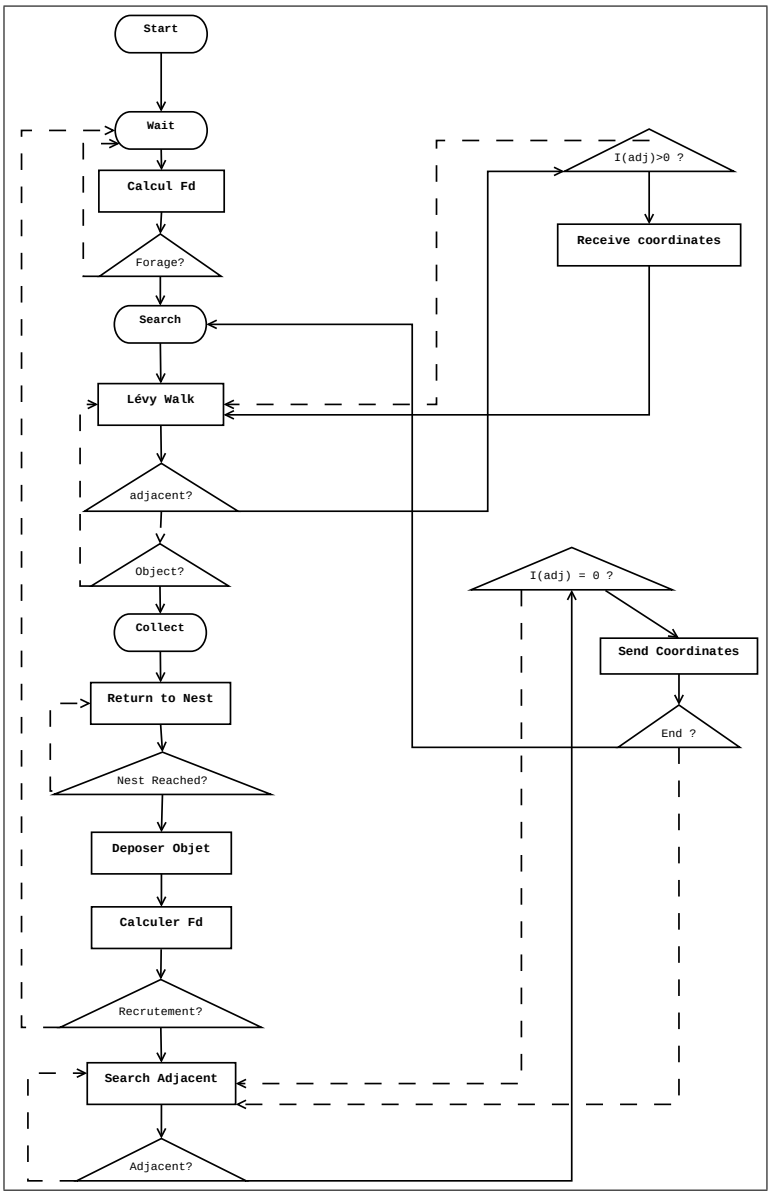


Fig. 1: Flowchart of the proposed algorithm

$$l = l_0 \times (\frac{1}{beta^{\frac{1}{\alpha}}} - 1) \tag{3}$$

```
repeat
    Walk using LévyWalk (Equation 3);
until (∃ Robot_adj ∨ ∃ Object);
if (∃ Object) then
    Goto Collect (Algorithm 3);
else
    if (∃ Robot_adj ∧ IntensityRobot_adj > 0 ∧ Robot_adj ∈ Other Nest) then
        Move towards Robot_adj ;
        Receive Coordinates;
```

Algorithm 2: Search

– *Collect*: In this state, the robot collects the object and returns to nest. Then it changes to *Wait* state. If the robot is recruited, it deletes the coordinates of the current nest and return to its original one. The pseudo code of this state is given by algorithm 3.

```
repeat
    Walk towards Nest;
until (∃ Nest);
Depose Object;
Update Nest Information;
if (Robot is recruited) then
    Update Nest Coordinates;
    Goto Search (Algorithm 2);
else
    Calculate F_d(t);
    if (F_d(t) = waiting ) then
        Goto Wait (Algorithm 1);
    else
        if (F_d(t) = search) then
            Goto Search (Algorithm 2);
        else
            Goto Recruit (Algorithm 4);
```

Algorithm 3: Collect

– *Recruit*: In this state, the intensity I of the robot is substituted by the quantity of objects missing in a nest (O_m), that should be collected to guarantee the nest survivability. The robot starts at searching objects and robots in the search space, it should recruit O_m robot. Robots in neighborhood are attracted to the recruiting robot; the latter exchanges with them the coordinates of its nest and updates its intensity I according to Equation 4. The pseudo code of this state is given by algorithm 4.

$$I_{t+1} = I_t - 1 \tag{4}$$

```
1 repeat
2 |   repeat
3 |   |   Search for Robot_adj;
4 |   until (∄ Robot_adj);
5 |   if (IntensityRobot_adj = 0 ∧ Robot_adj ∈ Other Nest) then
6 |   |   Send Coordinates;
7 |   |   Decrease Intensity using Equation 4;
8 |   Goto Search (Algorithm 2);
9 until (Nb_recruitR ≥ O_m);
```

Algorithm 4: Recruit

4 Experimental Results

The algorithm was implemented on the simulation platform Netlogo [11]. This Section presents the modeling of the proposed system components in Section 4.1. Then, the experimental scenarios and the performance metrics in Section 4.2 are presented. The obtained results are compared and discussed in Section 4.3.

4.1 Modeling of System Components

Robots, search space, targets and nests are modeled as follows:

- **Robots**: modeled as agents of 4 patches size, with velocity of 1 step per tick, a transport capacity of 1 object each time, a range vision of 15 patches and a communication range of 4 patches.
- **Search space**: modeled as a 2D limited space with variable size.
- **Targets**: represented with one patch and are distributed randomly or in clusters. Targets are regenerable.
- **Nests**: represented with 5 patches. The quantity of objects in each nest decreases in time with a predefined value.

4.2 Experimental Scenarios

The performance metric used is presented in Equation 5:

- *The survival percentage of nest* ϕ S_{rate_ϕ} : represents the survival percentage of a nest. A higher value of S_{rate_ϕ} indicates that robots were able to maintain

positive object levels at that nest. Equation 5 was used to calculate the S_{rate_ϕ}.

$$S_{rate_\phi} = \sum_{i=1}^{N} S(\phi, i)/N \tag{5}$$

$$F_d = \begin{cases} 1 \; if \; F(\phi, i) >= 0 \\ 0 \; if \;\; F(\phi, i) < 0 \end{cases} \tag{6}$$

Two scenarios were used to test the performances of the LWFR (Table 1). Where, *Scenario 1* was used to test the influence of increasing the environment size. Several configurations were applied with different environment sizes:30×30, 50×50, 80×80 and 100×100; and *Scenario 2* is used to test the influence of increasing the number of robots. Configurations with 5, 10 and 20 robots were used.

Table 1: Parameters of scenarios 1 and 2

Parameter	**Value**
Scenario 1: *Varying Size of Environment*	
Environment size	30×30, 50×50, 80×80, 100×100
Robots Number	5 per depot
Depots Number	4
Targets Number	50 regenerable
Scenario 2: *Varying Number of Robots*	
Robots Number	$5, 10, 20, 30$
Depots Number	4
Targets Number	50 regenerable
Environment size	50×50

4.3 Results and Discussions

Table 1 presents the results obtained in the two scenarios. In addition, this sections provides a comparison of LWFR when using long steps (LLWFR), and the SLWFR which is the LWFR when using short steps.//

Scenario 1: Table 2 shows the S_{rate_ϕ} of the 4 depots independently (Figure 2(a)) when increasing environment size (30×30, 50×50, 80×80, 100×100). Figure 2(a) illustrates that S_{rate_ϕ} is higher in small environment size (30×30), then while increasing the environment size the S_{rate_ϕ} decreases progressively until environments with 50×50, but it decreases dramatically below 0.5 with large environment size 80×80 and 100×100. Thus, the search strategy is still effective

in small environment size, but becomes ineffective in larger environments. LLWFR gives always better results than the SLWFR one. It can be observed that in some scenarios, SLWFR robots could not preserve positive values of S_{rate_ϕ}.

Table 2: Results of scenario 1, when increasing environment size

Env size	30×30				50×50				80×80				100×100			
Algorithm	D1	D2	D3	D4	D1	D2	D3	D4	D1	D2	D3	D4	D1	D2	D3	D4
LLWFR S_{rate_ϕ}	0.9	0.8	0.9	0.9	0.8	0.5	0.4	0.6	0.8	0.4	0.5	0.7	0.4	0.2	0.5	0.3
SLWFR S_{rate_ϕ}	0.9	0.8	0.8	0.8	0.2	0.2	0.1	0.2	0.2	0.1	0.1	0.1	0.1	0	0.1	0

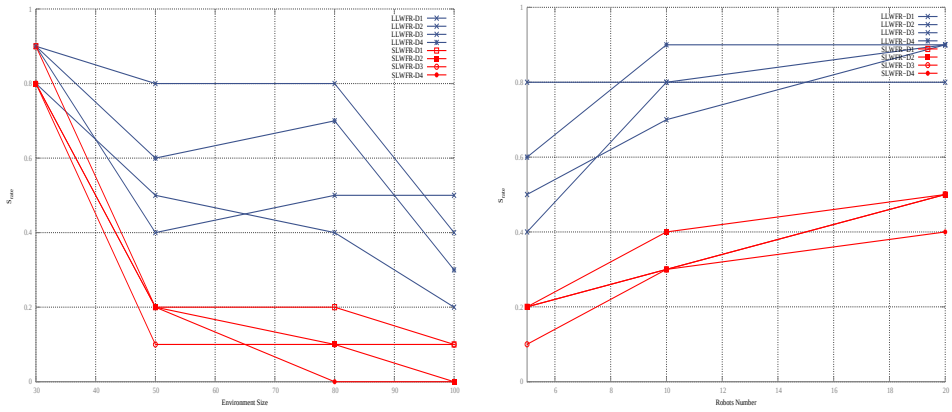
Scenario 2: Table 3 shows the S_{rate_ϕ} of the 4 depots independently (Figure 2(b)) when increasing number of robots from 5 to 20 robot. With 5 robots, S_{rate_ϕ} of depots is positive but do not reach higher values. But, with more robots (10 and 20 robots), the S_{rate_ϕ} reaches higher values meaning that while increasing robots number, robots can equilibrate the S_{rate_ϕ} in most of depots helping the system to function for long time. Unfortunately, the SLWFR gives worst results with 5 and 10 robots, the S_{rate_ϕ} does not reach the 0.4 and even with 20 robots the S_{rate_ϕ} does not attend the 0.5, this will not help the system to work for long time.

Table 3: Results of scenario 2, when increasing number of robots

Robots Number	5 robots				10 robots				20 robots			
Algorithm	D1	D2	D3	D4	D1	D2	D3	D4	D1	D2	D3	D4
LLWFR S_{rate_ϕ}	0.8	0.5	0.4	0.6	0.8	0.7	0.8	0.9	0.9	0.9	0.8	0.9
SLWFR S_{rate_ϕ}	0.2	0.2	0.1	0.2	0.3	0.4	0.3	0.3	0.5	0.5	0.5	0.4

5 Conclusion

This paper itroduced a Multi-Place Foraging algorithm called Lévy Walk and Firefly Recruiting Algorithm (LWFR). Robots in LWFR algorithm use: (1) Lévy walk as search strategy, which constitutes one of the best random walks to explore the whole search space efficiently, (2) Firefly Algorithm as recruiting model, where robots can use light to attract indirectly robots in neighborhood and recruit them if possible to collaborate in transporting the needed quantity



(a) (b)

Fig. 2: Results obtained in scenario 1 and in scenario 2.

of objects to their nest. Numerical results prove the effectiveness of the proposed algorithm in maintaining the survivability of their nests, allowing the system to work for long time.

Based on the results of this study, future work include: (1) the implementation of the proposed algorithm in the multi-robot simulator ARGoS [12], (2) the use of other scenarios with varying the quantity of objects extracted from depots and the time of extraction, (3) the test of other search strategies and their effect on maintaining the survivability of depots.

References

1. E. Sahin, Swarm Robotics, Vol. 3342 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. arXiv:arXiv:1206.1208v2, doi:10.1007/b105069.
2. E. Åđahin, S. Girgin, L. Bayindir, A. E. Turgut, Swarm Robotics, in: Swarm Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 87–100. doi:10.1007/978-3-540-74089-6-3.
3. Q. Lu, J. P. Hecker, M. E. Moses, The MPFA: A multiple-place foraging algorithm for biologically-inspired robot swarms, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 2016-Novem, IEEE, 2016, pp. 3815–3821. doi:10.1109/IROS.2016.7759561.
4. E. Castello, T. Yamamoto, Y. Nakamura, H. Ishiguro, Task Allocation for a robotic swarm based on an Adaptive Response Threshold Model, in: 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013), no. Iccas, IEEE, 2013, pp. 259–266. doi:10.1109/ICCAS.2013.6703905.
5. E. Bonabeau, G. Theraulaz, J. L. Deneubourg, Fixed response thresholds and the regulation of division of labor in insect societies, Bulletin of Mathematical Biology 60 (4) (1998) 753–807. doi:10.1006/bulm.1998.0041.

6. E. Castello, T. Yamamoto, Y. Nakamura, H. Ishiguro, Foraging optimization in swarm robotic systems based on an adaptive response threshold model, Advanced Robotics 28 (20) (2014) 1343–1356. doi:10.1080/01691864.2014.939104.
7. J. P. Hecker, M. E. Moses, Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms, Swarm Intelligence 9 (1) (2015) 43–70. doi:10.1007/s11721-015-0104-z.
8. Q. Lu, J. P. Hecker, M. E. Moses, Multiple-place swarm foraging with dynamic depots, Autonomous Robots 42 (4) (2018) 909–926. doi:10.1007/s10514-017-9693-2.
9. X. Yao, Y. Liu, G. Lin, Evolutionary programming using mutations based on levi probability distribution, IEEE Trans. on Evolutionary Computation 8 (1) (2004) 1–24.
10. X.-S. Yang, Firefly Algorithms for Multimodal Optimization, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 5792 LNCS, 2009, pp. 169–178.
11. S. Tisue, U. Wilensky, Netlogo: Design and implementation of a multi-agent modeling environment, in: Proceedings of agent, Vol. 2004, 2004, pp. 7–9.
12. C. Pinciroli, V. Trianni, R. OâĂŹGrady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, et al., Argos: a modular, parallel, multi-engine simulator for multi-robot systems, Swarm intelligence 6 (4) (2012) 271–295.