Nature-inspired Swarm Robotics Algorithms for Prioritized Foraging

Jade Abbott and Andries P. Engelbrecht

Computational Intelligence Research Group at the Department of Computer Science, University of Pretoria, South Africa jabbott,engel@cs.up.ac.za

Abstract. This paper introduces the problem of prioritized foraging. The performance of existing foraging algorithms, a naïve foraging algorithm and a desert ant foraging algorithm on the prioritized foraging problem is evaluated. The evaluation is used to motivate the necessity of creating an algorithm whose performance is independent of the initial robot configuration. A novel honey bee based foraging algorithm is then presented and its performance and adaptability to different environments is evaluated against the other foraging algorithms. This paper concludes that a relationship exists between the performance of the naïve and desert ant foraging algorithms and the initial ratio of robots configured to forage prioritized items. The honey bee algorithm was shown to perform reasonably well across all of the initial configurations, suggesting that the algorithm may perform well in a dynamic environment where robots and items could be destroyed or created over time.

1 Introduction

Swarm robotics is the co-coordination of large numbers of relatively simple robots to perform a single collaborative function, inspired from the observation of social insects such as ants, termites, and bees [1]. An important activity of all natural swarms is foraging for resources. Foraging is defined as the search and collection of resources from sources in an environment and returning the resources to a collection point [2]. These resources could be food, water, or building materials.

In times of stress, the collection of one resource may be prioritized over others - such as water during a drought or food before winter. Individuals in a natural swarm often adapt behaviour appropriately to enable greater collection of the prioritized item.

Item prioritization during foraging exists in real-world robot foraging problems such as search and rescue and gold mining. In the case of a building collapsing, robots need to get to the survivors as quickly as possible; however, it is important that some robots move waste material to reach the trapped survivors. Prioritized foraging can also be applied to the gold mining problem where gold needs to be foraged as a priority and the waste needs to be moved out of the way. In the mentioned applications, the distribution of objects and ratios of prioritized to non-prioritized types is often unknown.

This paper introduces the problem of prioritized foraging. The performance of a naïve foraging implementation and an existing nature-based foraging algorithm [3] based on the desert ant [4] are evaluated on the prioritized foraging problem. The evaluation is used to motivate the necessity of creating an algorithm whose performance is independent of the initial robot configuration. Finally, a novel honey bee based foraging algorithm, which has the ability to divide its labourers between items of different priorities, is presented. The performance and adaptability to environment item ratio of the honey bee algorithm is evaluated against the other foraging algorithms.

The remainder of the article is organized as follows. Section 2 provides background on foraging, while the problem of prioritized foraging is discussed in section 1. The robots that are used are described in section 4, and section 5 presents the algorithms. Section 6 outlines the experimental setup and the results are reported and discussed in 7. Section 8 concludes the paper.

2 Background

The purpose of this section is to present the swarm robotic foraging problem as well as provide details on foraging algorithms that exist in nature.

2.1 Foraging

In swarm robotics, robot foraging has become a benchmark problem due to its complex nature involving coordination of numerous sub-tasks. The problem was initially addressed by biologists, particularly in the foraging behaviour of ants [5, 6], bees [7] and bacteria [8]. Foraging has a variety of real-life applications such as search and rescue [9, 10], and waste clean-up [11]. Numerous swarm robotics algorithms have been developed to improve robustness, time and energy efficiency of the foraging process for multiple variations of the foraging problem as outlined in [2].

2.2 Ant Foraging

Although algorithms based on ant foraging behaviour are used to solve optimization problems, the algorithms are often difficult to replicate in a real-life robotic environment as many of the techniques conceptually use pheromones. A robotics algorithm that uses pheromone either requires robots to be equipped with a substance-distributor, beacon-deployer or a complex simulation of pheromone, requiring communication [12].

The desert ant (*Cataglyphis bicolor*) does not make use of pheromone-based communication to forage, as pheromone deposited on desert sand would be blown away by the wind. Instead, desert ants use a technique called path-integration for navigation to relocate food sources that have already been found by random exploration [13].

The desert ant is thus simpler to model for real-robot interaction. Desert ant foraging has been modelled and experiments have been performed on real robots in [4]. Experiments were performed using a small set of robots in a small arena on different environment types. Two algorithms were evaluated: one based on the desert ant foraging and one including pheromone-like communication. It was shown that communication improved performance; however, the desert-ant algorithm still performed reasonably.

2.3 Bee Foraging

Honey bee foraging is made up of three different roles [7]: Employed foraging, unemployed foraging and scouting. Scouts explore the environment to locate new food sources. Once a source has been found, scouts return to the hive to communicate information about the located food source. To communicate foraging information, scout bees perform a waggle-dance on the hive, containing information about the distance and bearing of the resource. Communication of good foraging sites is known as recruitment.

Unemployed foragers wait on the dance floor, evaluating the dances of the scout bees. An unemployed bee chooses a location described by the scout bees, and becomes an employed forager. Employed forager bees use the information about the resource, attempt to locate the source, load themselves with food and return to the hive where unemployed foragers are ready to offload the food. Jansen [14] suggests that unemployed bees become exploring scouts when they do not detect any dancing scout bees.

Bee swarms have been used in swarm robotics for problems such as path planning [15], aggregation [16], collective perception [17].

3 Prioritized Foraging

The multi-foraging problem is a more complex version of standard foraging, where more than one type of item must be foraged to a separate sink [18]. That in mind, the prioritized foraging problem, shown in Fig.1, is defined as a modified version of the multi-foraging problem where an environment has two types of items: prioritized items and non-prioritized items. The goal is to forage all the items of the prioritized type. The possibility exists that prioritized items become trapped among non-prioritized items and thus the non-prioritized items need to be removed from the environment to clear an access route to the prioritized items F provides a visual representation of the problem.

The prioritized foraging problem has increased difficulty due the fact that foraging the non-prioritized item more than required will result in a waste of time and energy. The goal of research in prioritized foraging is to develop an algorithm to efficiently adapt the number of robots searching for prioritized items to those moving non-prioritized items out the way.

Prioritized foraging could be applied to the gold mining problem where the gold needs to be foraged as a priority and the waste needs to be moved out the way.

4 Artificial Robots

The artificial robots modelled in this study are based on e-puck robots [19], adapted with grippers. Each robot is equipped with a 360 degree camera to identify objects around them as well as eight local distance sensors spaced equally around the circular perimeter of the robot. Both camera and distance sensors have a depth of view of five times the robot's size. Robots use local communication, which can occur in a radius of five times the robot's size. The sensor and communication range is sufficiently localized. A robot can forage a single item at a time. The robots do not have a global positioning system (GPS) capability to locate items and to position themselves in the environment. As a

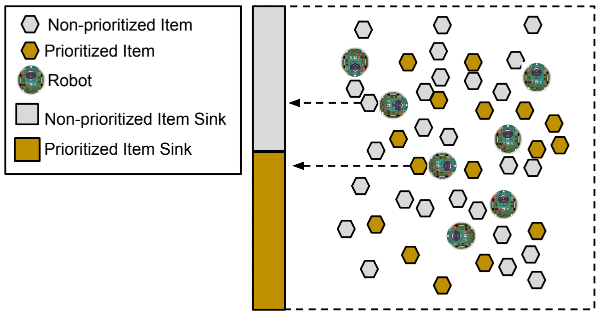

Fig. 1: Prioritized Foraging Problem

result robots have to explore the environment to find the prioritized items. Due to the lack of GPS capabilities, robots can not see an item hidden by another item.

Due to the environmental complexity, advanced navigation and obstacle avoidance technique is required. The technique is based on the flocking behaviour of birds used in [20], where robots are pulled to a global attractor, while a local attractor directs robots away from local obstacles but while maintaining a course to the destination. All algorithms use in this paper use the same obstacle avoidance and navigation technique.

5 Algorithm Description

The algorithms that have been evaluated are naïve foraging discussed in section 5.1, desert ant foraging discussed in section 5.2, and honey bee foraging discussed in 5.3.

5.1 Naïve Foraging

Naïve foraging includes only the most minimal set of foraging actions. Naïve foraging is included as a baseline for comparison to evaluate how a memory-based technique such as desert-ant foraging compares to a standard model [3, 12].

Naïve foraging consists of the following tasks: searching for an item, grabbing an item, returning home with the item and storing the item at the sink as shown in Figure 2. The robots repeat the following steps:

1. Robots perform a random walk until they find an item.
2. On locating an item, the robot grips the item. If the item has been moved before the robot is able to pick it up, the robot will continue to explore; otherwise the robot returns the item to the correct sink using a beacon-based homing algorithm.

The following random walk is used: A robot chooses a random direction, σ, and a random distance $m \in (0, M)$ where M is a chosen maximum path length. The robot walks in direction σ for distance m. The robot then chooses new values for σ and m.

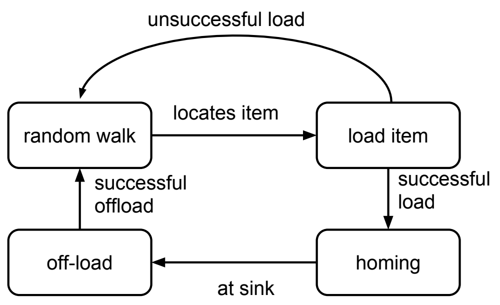

Fig. 2: Naïve Foraging State Diagram

5.2 Desert Ant Foraging

Due to the lack of pheromone, desert ant foraging behaviour is a very suitable model for robot foraging. Desert ants use path integration to memorize the location of an existing food source and later to return to the memorized source to find more food. The notion of returning to a previously explored site is known as site fidelity [21]. The desert ant algorithm does not require communication between robots or the dispersal of beacons, and is thus simpler than other many swarm robotics foraging algorithms.

Path integration (PI) is the integration of an ant's odometry such that the ant can maintain a position and heading estimate of where the ant is going, by continuously updating a heading direction vector that points to the starting position [22]. The heading-direction vector is known as the PI vector. This type of behaviour theoretically appears suitable for environments where multiple items occur in similar areas. The disadvantage of PI is the accumulation of localisation errors. The robot control algorithm consists of five states (illustrated in Fig 3):

1. **Exploration** – the robot performs a random walk while performing PI.
2. **Loading** – on finding an item, the robot loads the item and memorizes the PI vector.
3. **Homing** – the robot uses the path integration vector to move to the sink.
4. **Off-loading** – when the robot is at the sink, the object is offloaded.
5. **Locating** – the robot follows the memorized PI vector to the location of the previous item. If another item is found, the item is loaded; otherwise the robot returns to the exploration state.

All robots begin at random positions adjacent to the sink in the exploration state.

5.3 Honey Bee Foraging

The presented honey bee algorithm is based on the mathematical model of honey bee foraging in [7]. A portion of the robots are initialized as scouts and the rest as unemployed foragers in a waiting state. All robots are initialized adjacent to the item sinks.

Fig 4 is a simplified state diagram for the honey bee algorithm. State transitions are described as follows:

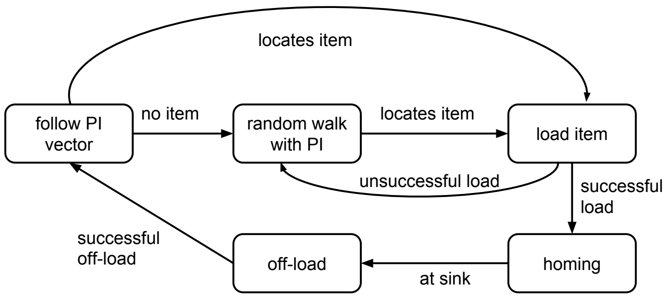

Fig. 3: Desert Ant Foraging State Diagram

1. A scout robot performs a random walk and, upon finding an item and evaluating the site, forages the item by returning the item to the sink using PI.
2. A scout decides to dance, based on the quality of the site. If the estimated quality of the site, μ, is less than the dance threshold, ϕ, the scout robot does not dance and instead continues to forage the site as an employed forager.
3. Otherwise, if the estimated quality of the site, μ, is greater or equal to the dance threshold, ϕ, the scout robot dances for the unemployed foragers.
4. After a dance is complete, the scout robot decides with probability ρ to start exploring again or recruit itself and begins foraging the found site.
5. On detecting a dance, waiting bees become foraging bees with a probability α and switch to foraging the dancer's item type.
6. Employed foragers become unemployed foragers (waiting foragers) if the foraging site has been depleted.
7. Unemployed foragers become scouts if no dances are detected t_{max} time steps.

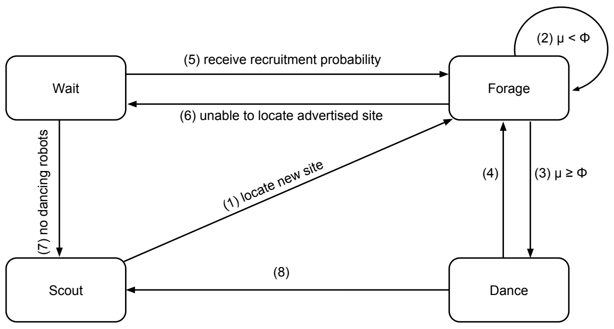

Fig. 4: Honey Bee Foraging State Diagram

Site quality, μ_t, for a robot scouting items of type t, is calculated as the estimated density of items of type t in the local vicinity of the found item. The robot has distance sensor values $k_i \in [0, 1]$ for $i = 1...n$, where 0 means that nothing is detected in sensor range and 1 indicates that the robot is touching an item and n is the number of distance sensors. The item density of type t, μ_t, is calculated using

$$\mu_t = \frac{1}{n} \sum_{i=1}^{n} k_{i_t} \tag{1}$$

where the sensor value for item type t, k_{i_t}, is calculated using

$$k_{i_t} = \begin{cases} k_i & \text{if item } i \text{ is type } t \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

In nature, in times of drought, bees prioritize water over nectar or pollen. Bees are sent out to forage water; however, if they happen to encounter pollen, they will forage it but will not communicate the discovery to the unemployed foragers [7]. The rules for item-type division of labour are based on the behaviour of bees under environmental pressure as follows:

1. A robot foraging the prioritized type, forages a non-prioritized type only if a prioritized item can not be located for max time f_{max}.
2. A robot foraging a non-prioritized item forages the non-prioritized item type until the robot fails to relocate the site or the robot locates a prioritized item. The robot switches to foraging the prioritized item.
3. A robot foraging a non-prioritized item will not communicate the location of the non-prioritized item site by dancing.

For the purpose of this study, honey bee robots are initialized to start foraging a particular item type to show the effectiveness of the division of labour. The presented algorithms differ in three main properties as given in Table 1.

Table 1: Properties of the foraging algorithms used in this study

Property	Naïve	Desert Ant	Honey Bee
Memory	✗	✓	✓
Communication	✗	✗	✓
Division of Labour	✗	✗	✓

6 Experimental Setup

For simplicity, a 2D grid world was used to evaluate the performance of the foraging algorithms. Each robot fits into one grid block and each item takes up one grid block. Items can be picked up or can form obstacles that robots must navigate around.

The prioritized and non-prioritized sinks were placed next to each other, on a single side of the environment to more accurately represent the type of environment that the problem is most likely to be applied to e.g. in mine tunnels. The sink placement differs from the more common placement at the centre of the environment. The sinks were marked by light beacons that all robots can detect and navigate towards. Four different classes of environments were used as follows:

1. Environments where items of each type are uniformly distributed (Fig 5a).
2. Clustered environments with clusters of item types generated by randomly relabelling items in clusters generated by Lumer-Faieta ant cemetery clustering [23] as either prioritized or non-prioritized items (Fig 5b).
3. Vein environments resembling the natural occurrence of gold [24] (Fig 5c).
4. Gaussian environments where prioritized items are focused at the environment center. (Fig 5d).

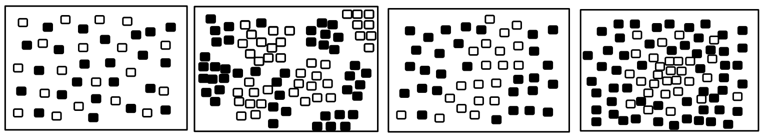
(a) Uniform (b) Clustered (c) Vein (d) Gaussian

Fig. 5: Environment Classes

For each class of environment, the following configurations were tested: The environment grid size, $S = 50, 100, 200$ where S is the width and length of the grid and the percentage p of the grid covered by objects, with $p = 5\%$, 20%, 50%, 70%, 90%. The ratio of prioritized to non-prioritized items r, is varied, where $r = 0$, 0.2,0.25, 0.333, 0.5, 0.667, 0.75, 0.8, 1. Honey bee specific parameters were selected based on [7] as $t_{max} = 200$ timesteps, $f_{max} = 100$ timesteps, $\phi = 0.8$ and $\rho = 0.1$. The following performance measures were used: the percentage of each item foraged over time, average time a robot spent in waiting state, and the entropy of robot movement. Due to space constraints, only the percentage, σ, of prioritized items foraged is presented.

For all algorithms, robots were initially configured to forage either the prioritized item or the non-prioritized item with a ratio of $\tau = 0$, 0.2, 0.25, 0.333, 0.5, 0.667, 0.75, 0.8,1. Different numbers of robots, c, were used with $c = 10, 30, 50, 70, 100$; c is defined as the percentage of cells of the grid size S that are occupied by robots.

7 Results

Section 7.1 focuses on providing a general overview of algorithm performances, and the motivation for the development of an algorithm that can adapt to item type ratio is presented in section 7.2. Section 7.3 highlights how efficiently the honey bee algorithm adapts to item type ratio.

7.1 General Overview

When comparing two foraging algorithms, a pairwise Wilcoxon test was performed to determine if a statistical difference occurs, at a significance level of 95%. The test was performed over all environments. The null hypothesis is that the results of the two algorithms come from the same distribution. Table 2 gives a final algorithm ranking where 2 is the best performing algorithm and 0 is the worst performing algorithm.

Table 2: The overall Pairwise Mann Whitney U ranking, averages and standard deviations of for σ for each algorithm

Algorithms	Wins	Average	Std Dev
Naive	0	0.528	0.394
Desert Ant	1	0.643	0.387
Honey Bee	2	0.807	0.294

Statistical tests indicate a significant difference between the results of all algorithms. Desert ant foraging performed better than naïve foraging showing the positive effect of site fidelity. The honey bee algorithm out-performed the naïve foraging algorithm and desert ant algorithm indicating the positive effect of communication and adaptivity of the honey bee foraging algorithm. The standard deviation is high for all algorithms due to the extremely large variations in the environments provided.

7.2 Analysis of Relationship between Item Ratio in Environment and Item Ratio of Robots

The following hypotheses are addressed:

1. An algorithm that forages a portion of non-prioritized items will have greater performance than an algorithm that does not forage any non-prioritized items.
2. Algorithm performance depends on the r as well as τ and that as r increases, the value of τ that yields the greatest value of σ, τ_{best}, will increase approximately linearly for the naïve and desert ant algorithms.

An algorithm configured with $\tau = 1$ is where only prioritized items are foraged. Analysing Table 3, for the naïve and desert ant algorithms, for all values of r where $r \neq 1$, τ_{best} is never equal to 1, proves the hypothesis that the algorithms achieved the best performance when some robots are configured to forage non-prioritized items. The result may be because non-prioritized items are moved out of the way to allow for easier, faster access to prioritized items or allow access to inaccessible prioritized items.

Fig 6 shows the region in parameter space where the desert ant algorithm performs the best. The naïve and desert ant algorithms performed best when τ was slightly greater than r. The existence of the relationship motivates the development of an algorithm that adapts τ to correspond the environment item ratio r.

7.3 Adaptability of the Honey Bee Foraging Algorithm to Item Type Ratio and Comparison to the Naïve the Desert Ant Algorithms for Naïve and Desert Ant Algorithms

Analysis of Table 3 indicates that the honey bee foraging algorithm has similar performance throughout all configurations for r and τ, which highlights that the performance

Table 3: The performance, σ, for each foraging algorithm, for each combinations of r and τ. If τ_{best} exists, τ_{best} is provided. The best value of σ is shown in bold.

Algorithm	r	τ									τ_{best}
		0	0.2	0.25	0.333	0.5	0.667	0.75	0.8	1	
	0	1	1	1	1	1	1	1	1	1	
	0.2	0	0.492	0.526	0.567	**0.597**	0.595	0.587	0.577	0.471	0.5
	0.25	0	0.484	0.526	0.557	0.588	**0.595**	0.585	0.575	0.477	0.667
	0.333	0	0.467	0.507	0.544	0.586	**0.596**	0.592	0.584	0.495	0.667
Naïve	0.5	0	0.428	0.46	0.508	0.568	0.588	**0.591**	0.589	0.528	0.75
	0.667	0	0.4	0.433	0.487	0.544	0.583	**0.591**	0.593	0.554	0.75
	0.75	0	0.377	0.425	0.47	0.531	0.576	0.585	**0.591**	0.567	0.8
	0.8	0	0.372	0.409	0.455	0.53	0.571	0.584	**0.592**	0.575	0.8
	1	0	0.336	0.375	0.433	0.5	0.552	0.57	0.581	**0.618**	1
	0	1	1	1	1	1	1	1	1	1	
	0.2	0	0.698	0.724	**0.737**	**0.737**	0.712	0.694	0.67	0.519	0.333
	0.25	0	0.678	0.711	0.73	**0.735**	0.715	0.697	0.673	0.530	0.5
	0.333	0	0.65	0.693	0.722	**0.739**	0.725	0.71	0.686	0.562	0.5
Desert Ant	0.5	0	0.596	0.645	0.684	0.729	**0.734**	0.725	0.701	0.621	0.667
	0.667	0	0.554	0.607	0.648	0.706	0.737	**0.738**	0.716	0.675	0.75
	0.75	0	0.533	0.587	0.63	0.691	0.731	**0.739**	0.72	0.703	0.75
	0.8	0	0.523	0.577	0.62	0.682	0.725	0.736	**0.74**	0.718	0.8
	1	0	0.488	0.543	0.588	0.654	0.702	0.718	0.726	**0.758**	1
	0	1	1	1	1	1	1	1	1	1	
	0.2	**0.687**	**0.687**	0.686	0.686	0.686	0.685	0.686	0.685	**0.687**	
	0.25	0.678	**0.679**	0.678	0.678	**0.679**	**0.679**	0.678	0.677	**0.679**	
	0.333	**0.674**	**0.674**	**0.674**	**0.674**	**0.674**	**0.674**	0.673	**0.674**	**0.674**	
Honey Bee	0.5	0.668	**0.669**	0.668	0.668	0.668	0.668	0.668	0.668	**0.669**	
	0.667	0.671	0.671	0.671	0.671	0.671	**0.672**	0.671	0.671	0.671	
	0.75	0.672	**0.673**	0.671	0.671	0.672	**0.673**	0.672	**0.673**	**0.673**	
	0.8	0.674	0.674	0.674	0.674	0.674	**0.675**	**0.675**	**0.675**	**0.675**	
	1	**0.691**	0.69	**0.691**	0.69	**0.691**	**0.691**	0.69	0.69	0.69	

of the honey bee algorithm is independent of the configuration of τ, resulting in an algorithm that is more flexible and robust. This could mean that the honey bee algorithm could perform well in dynamic environments where robots and items can be destroyed.

However, according to Table 3, the desert ant algorithm performs better than the honey bee algorithm, for particular configurations of r and τ. This indicates that, if the value of r is known for a particular environment, then it is beneficial to use desert ant foraging and choose τ appropriately. A possible reason why the desert ant algorithm performs better when optimally configured for a particular environment than the honey bee algorithm is that the honey bee algorithm takes time to adapt to the environment, while the desert ant algorithm with optimal configurations has no division of labour overhead and may outperform the honey bee algorithm under those circumstances.

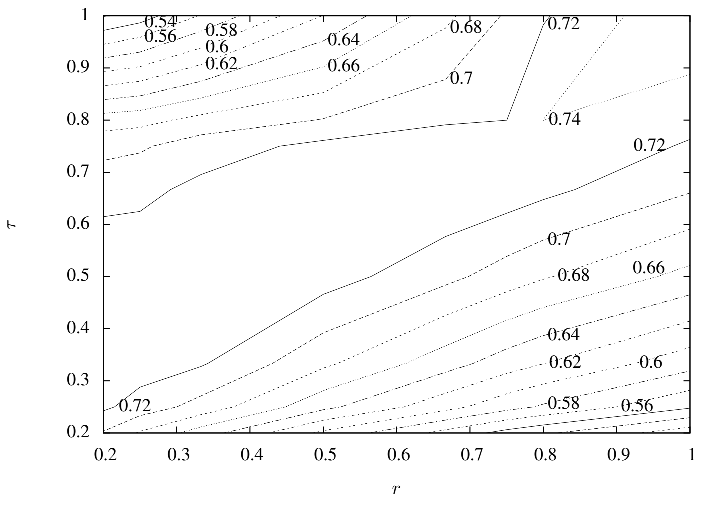

Fig. 6: Contour plot of values for σ over values for r and τ for desert ant foraging

8 Conclusions and Future Research

This paper concludes that an algorithm that forages a portion of non-prioritized items will have greater performance than an algorithm that does not forage any non-prioritized items and that the performance of the naïve and desert ant foraging algorithms on the foraging problems is dependant on the ratio of robots initially configured to forage the prioritized item and the ratio of prioritized items in the environment.

The honey bee algorithm was shown to perform well across all of the initial configurations; however, the desert ant algorithm still outperformed the honey bee algorithm for certain environment object ratios and robot forage type ratios.

Results suggest that the honey bee algorithm may perform well in a dynamic environment where robots and items could be destroyed or created over time. Future work will also evaluate the performance of the honey bee algorithm in dynamic environments. Future work will also include an evaluation of other performance measures as well as an in depth discussion of the scalability of the algorithms.

References

1. Dorigo, M., Sahin, E.: Swarm robotics. Auton. Robots **17**(2-3) (2004) 111–113
2. Winfield, A.F.: Foraging robots. (2009)
3. Ø stergaard, E.H., Sukhatme, G.S., Matari, M.J.: Emergent bucket brigading: a simple mechanisms for improving performance in multi-robot constrained-space foraging tasks. In: Proceedings of the fifth international conference on Autonomous agents, ACM (2001) 29–30
4. Hecker, J.P., Letendre, K., Stolleis, K., Washington, D., Moses, M.E.: Formica ex machina: ant swarm foraging from physical to virtual and back again. In: Swarm Intelligence. Springer (2012) 252–259

5. Hölldobler, B.: The ants. Harvard University Press (1990)
6. Bernstein, R.A.: Seasonal food abundance and foraging activity in some desert ants. American Naturalist (1974) 490–498
7. Seeley, T.D.: The wisdom of the hive: the social physiology of honey bee colonies. Harvard University Press (2009)
8. Resnick, M.: Turtles, termites, and traffic jams: Explorations in massively parallel microworlds. Mit Press (1994)
9. Jennings, J.S., Whelan, G., Evans, W.F.: Cooperative search and rescue with a team of mobile robots. In: Advanced Robotics, 1997. ICAR'97. Proceedings of the 8th International Conference, IEEE (1997) 193–200
10. Murphy, R.R.: Biomimetic search for urban search and rescue. In: Proceedings of Intelligent Robots and Systems, 2000.(IROS 2000). 2000 IEEE/RSJ International Conference on. Volume 3., IEEE (2000) 2073–2078
11. Balch, T., Boone, G., Collins, T., Forbes, H., MacKenzie, D., Santamar, J.C.: Io, ganymede, and callisto a multiagent robot trash-collecting team. AI magazine **16**(2) (1995) 39
12. Hoff, N.R., Sagoff, A., Wood, R.J., Nagpal, R.: Two foraging algorithms for robot swarms using only local communication. In: Proceedings of Robotics and Biomimetics (ROBIO), 2010 IEEE International Conference on, IEEE (2010) 123–130
13. Collett, M., Collett, T.S., Bisch, S., Wehner, R.: Local and global vectors in desert ant navigation. Nature **394**(6690) (1998) 269–272
14. Janson, S., Middendorf, M., Beekman, M.: Searching for a new homescouting behavior of honeybee swarms. Behavioral Ecology **18**(2) (2007) 384–392
15. Lin, J.H., Huang, L.R., et al.: Chaotic bee swarm optimization algorithm for path planning of mobile robots. In: Proceedings of the 10th WSEAS international conference on evolutionary computing, World Scientific and Engineering Academy and Society (WSEAS) (2009) 84–89
16. Kernbach, S., Thenius, R., Kernbach, O., Schmickl, T.: Re-embodiment of honeybee aggregation behavior in an artificial micro-robotic system. Adaptive Behavior **17**(3) (2009) 237–259
17. Schmickl, T., Möslinger, C., Crailsheim, K.: Collective perception in a robot swarm. In: Swarm robotics. Springer (2007) 144–157
18. Balch, T.: The impact of diversity on performance in multi-robot foraging. In: Proceedings of the third annual conference on Autonomous Agents, ACM (1999) 92–99
19. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: Proceedings of the 9th conference on autonomous robot systems and competitions. Volume 1. (2009) 59–65
20. Antoniou, P., Pitsillides, A., Blackwell, T., Engelbrecht, A., Michael, L.: Congestion control in wireless sensor networks based on bird flocking behavior. Computer Networks (2012)
21. Switzer, P.V.: Site fidelity in predictable and unpredictable habitats. Evolutionary Ecology **7**(6) (1993) 533–555
22. Ronacher, B.: Path integration as the basic navigation mechanism of the desert ant cataglyphis fortis (forel, 1902)(hymenoptera: Formicidae). Myrmecological News **11** (2008) 53–62
23. Lumer, E.D., Faieta, B.: Diversity and adaptation in populations of clustering ants. In: Proceedings of the third international conference on Simulation of adaptive behavior: from animals to animats, MIT Press (1994) 501–508
24. Frimmel, H., Minter, W.: Recent developments concerning the geological history and genesis of the witwatersrand gold deposits, south africa. Special Publication-Society of Economic Geologists **9** (2002) 17–46