Decentralised Autonomic Self-Adaptation in a Foraging Robot Swarm

Liam McGuigan, Roy Sterritt, George Wilkie, Glenn Hawe
School of Computing, Faculty of Computing, Engineering and the Built Environment
Ulster University
Jordanstown, N. Ireland
email: mcguigan-l8@ulster.ac.uk, r.sterritt@ulster.ac.uk, fg.wilkie@ulster.ac.uk, gi.hawe@ulster.ac.uk

***Abstract*— The deployment of a swarm of robots in domains, such as mine clearance or search and rescue operations, requires that they are self-adaptive in order that they may adjust to unforeseen events and up to date information. Much of the research on swarm self-adaptation focuses on the adaptation of individual swarm behaviour, however a top-down approach may allow a swarm to adjust its behaviour on the basis of the combined knowledge of the swarm. This research looks at producing a decentralised autonomic manager to handle such adaptation in a task of foraging robots, by adjusting the range over which robots broadcast help requests based on the perceived density of the swarm. First, the robots are tasked with recognising the initial situation, before responding to two possible events which alter the scenario, namely the destruction of a proportion of the swarm, and a change in the effective communication range. A centralised system is first developed as an idealised system with full swarm knowledge, and then a decentralised version is created to perform the same role on a per-robot basis, using only the information available to it. The performance of the swarm using each autonomic manager is compared against the performance when using a fixed broadcast range identified to be most suitable for the initial circumstances. It is found that both approaches are capable of recognising the initial situation, and of responding to events, however the effectiveness of the response may depend upon additional parameters not taken into account here. The decentralised autonomic manager presented is also found to require the ability to dynamically alter its own parameters in order to be of use.**

Keywords- Swarm robotics; Self-adaptation; Autonomic Computing; Simulation.

I. INTRODUCTION

This paper is an extended version of the work published in [1], extending those results and presenting further research.

A swarm of robots, in which the aggregate behaviour of many relatively simple individuals combines to create a more complex set of behaviours [2], can have applications in areas, such as mine clearance [3], search and rescue [4] and space exploration [5][6]. A robot swarm can reduce the demands on any single robot, may accomplish the task more quickly, and can be deployed where sending humans is too dangerous, difficult, or costly.

The ability to self-adapt, that is to adjust behaviour in response to newly acquired information without the need for external guidance, is a requirement of a robotic swarm [7]. Unforeseen events may occur that require adjustment, and factors, such as distance and time, may restrict the ability of a human operator to act successfully. Self-adaptation can be applied to the swarm in a variety of ways [8], including the development of emergent behaviours [9], evolutionary systems [10] and swarm-level decision making [11].

Autonomic Computing concepts [12][13] can be used for swarm-self adaptation. At the swarm level, an Autonomic Manager (AM) employing a control loop, such as the Monitor, Analyse, Plan, Execute system described by [12] can be used to allow the swarm to assess the current situation and take any action necessary, as seen in Figure 1. This may be implemented in either a centralised manner, with individual robots communicating with a central command unit, or in a decentralised manner with each robot using its own control loop in order to modify its own behaviour in response to shared information and experience.

The objective of this work is to explore the potential for using swarm-level self-adaptation in a swarm of robots to improve performance in a foraging task, specifically the time it takes the swarm to complete the task which may often be an application priority, such as in search and rescue.

Robot swarms are typically decentralised in nature [3], and a similarly decentralised approach for the swarm's self-adaptation is desired. Initially, a centralised approach is used as an exploratory stage to determine if an AM provides any benefits. This is followed by an implementation of a decentralised approach to performing the same self-adaptation. As the centralised AM exists only to explore self-adaptation options, it will not take a more active role in coordinating the swarm in its task.

Each approach to the AM aims to achieve performance improvement through the modification of the range at which individual robots communicate with their neighbours for

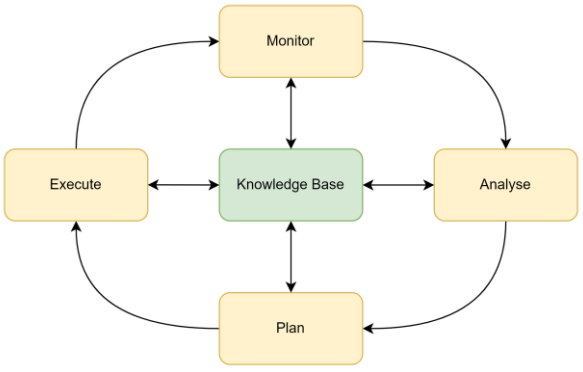


Figure 1. MAPE-K loop, as used by an autonomic component. The loop proceeds through each of the four stages in turn.

assistance. The swarm is tasked with deciding the appropriate communication range, and then two unforeseen events are introduced. The first, robot destruction, tests the swarm's ability to react to the sudden change in swarm size, such as a loss of robots in a search and rescue task due to the hazardous environment. The second, a change to communications quality, represents a situation where the ability of the robots to communicate with each other may be hampered by a change in environmental conditions.

The rest of this paper is structured as follows. Section II discusses related work in swarm-level adaptation. Section III describes the simulation and the foraging task used. Section IV explains how the two approaches to autonomic management have been applied to the swarm, while Section V describes the test scenarios. Section VI reports the results and explores the implications, and Section VII concludes the paper with a summary, and any future research directions.

II. Related Work

The location where adaptation is applied to a swarm is important when considering the intended goal. Much of the research in adaptation focuses on the level of the individual agent, where the resulting swarm performance is affected by the aggregate of these individual behaviours [8]. This level of adaptation can have a dramatic impact on performance, but it is difficult for any single robot to take advantage of information that is only available when viewing the larger picture, or to make decisions affecting the behaviour of other members of the swarm, such as cooperation or communication.

Adaptation at the swarm level can counter some of these problems. [14] describes an approach to moderating the size of the swarm in order to reduce degraded performance due to congestion. Robots keep track of the conflicts that occur when two robots attempt to occupy the same cell. If the number of conflicts crosses a threshold, virtual pheromones can be deposited at the entrance in order to instruct robots to leave or join the area. Hence, the swarm can adjust its size based on the combination of each robot's collision tracking data.

In [15], a group of unmanned aerial vehicles (UAVs) are patrolling an area defined by a set of cells, with the aim of ensuring that cells are visited often enough during the mission. Individual UAVs decide their next target on the basis of values assigned to the cells by a central system based on UAV visitation. Different strategies for assigning those values are explored, and so the central system becomes an effective behaviour adaptation method for the group.

As discussed in Section I, autonomic concepts may be used for swarm self-adaptation. [16] describes an adaptation pattern in which one robot in the swarm takes on the role of an AM, running a control loop with visibility of the whole system. In the case study presented, the swarm was tasked with exploring an unknown area. Robots communicate their positional and explorational information with the AM, which can direct them to underexplored areas. Recognising that a centralised system may be a bottleneck, a decentralised variant is also used in which the robots share the information with their neighbours. Both approaches perform much better than a basic pheromone-based approach.

A partially distributed approach described in [17] uses a group of UAVs, together with communication base stations taking on the role of AMs, engaged in a search task. If one of the UAVs leaves the active area and loses the communication link, the base stations are able to recognise the failure and reposition themselves in order to retrieve the UAV, while also minimising disruption to the rest of the swarm.

In a previous paper [8], cooperation strategies for swarms were investigated to determine the potential for using an AM to select between them based on the situation. Here, we build on that research by using the Help Recruitment strategy, exploring the potential for using an AM to modify the broadcast range parameter in order to improve performance, and then creating a decentralised implementation to achieve that.

III. Simulation of Foraging Robots

This research employs a time-stepped simulation of a heterogeneous swarm of agents, engaged in a variant of a foraging task, as in a previous work [8]. The following subsections describe the simulation and task, the behaviour of the robots, and how communications and energy are handled by the simulation.

A. Simulation and Foraging Task

The simulation presents a world consisting of a rectangular grid, in which several items and robots are placed at random. Each item or robot may be one of two possible types, represented by their colours, as shown in Figure 2. A single cell may contain only one item, however it can contain any number of robots. Each cell can therefore be considered to represent an area much larger than the footprint of a single robot, and thus the simulation may ignore potential collisions between robots.

The simulation is updated in a time-stepped manner.

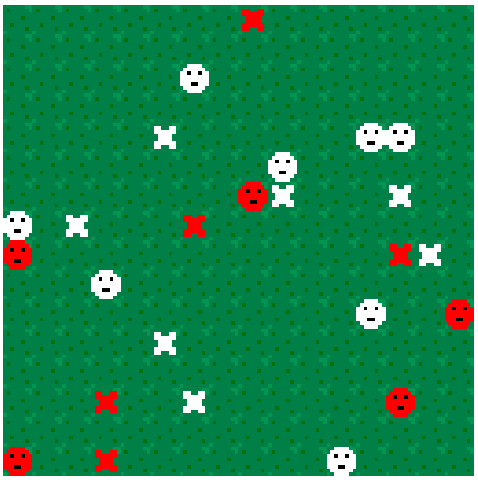


Figure 2. A portion of the world state during a simulation. The colour of a robot (face) or item (cross) indicates its type.

Each tick of the update loop, all robots are updated in turn, for a fixed time delta. Hence, when measuring the time taken for the swarm to complete a task, it is the number of ticks of the simulation that must be measured, rather than the real time taken for the simulation to complete, which may be impacted by factors such as the simulation playback speed, host platform, and so forth.

The robots are given the task of foraging the items, with each robot capable only of foraging an item that matches its own type and occupies the same cell. The task ends once all items have been successfully foraged. Each item is foraged in the position found, with no requirement to return the item to a home base – this process is therefore analogous to applications, such as the analysis of mineral deposits, or environmental cleanup.

B. Robot Behaviour

The behaviour of the robots follows the Help Recruitment strategy developed in [8]. Figure 3 is a state machine diagram showing the states and transitions employed by the robots in this strategy.

A robot begins in the Explore state. Each tick of the simulation, the robot first checks for an item in the current cell, and if none are found, it selects a random adjacent cell to move into. If an item is found, the robot will transition to the Forage state.

In the Forage state, the robot first checks the item to see if it matches the robot's type. If it does, the item is successfully foraged and the robot resumes exploration. However, if the type does not match, the robot broadcasts an initial help message to neighbouring robots, stating that it has found an item of a given type, and requires assistance from suitable robots. After sending the message, it moves to the Wait For Offers state.

A robot remains in the Wait For Offers state for two ticks, listening for responses from other robots. After this period, the robot will select the offer from the nearest robot and send an assignment message to them, before resuming exploration. If no offers have been received, exploration is resumed with no assignment.

A robot receiving a help request will only respond to it if it matches the type, and is in the Explore state. In this case, the robot sends a offer message to the robot requesting help, and moves to the Wait For Assignment state, where it can remains for up to three simulation ticks before resuming exploration. If it receives an assignment message in this time, it then moves to the Respond state.

In the Respond state, the robot moves directly towards the location of the item to be foraged, but continues to check the cells it passes through, sending help requests if necessary. The robot continues responding until it arrives at the target destination, where it can forage the item if it is still present, or resume exploration if another robot has foraged the item ahead of it.

In this way, robots are able to cooperate. Where a robot is unable to forage an item it finds, it can recruit a nearby robot to carry out the task instead. As reported in [8], engaging in this cooperation increases the performance of the swarm compared to having no cooperation in place.

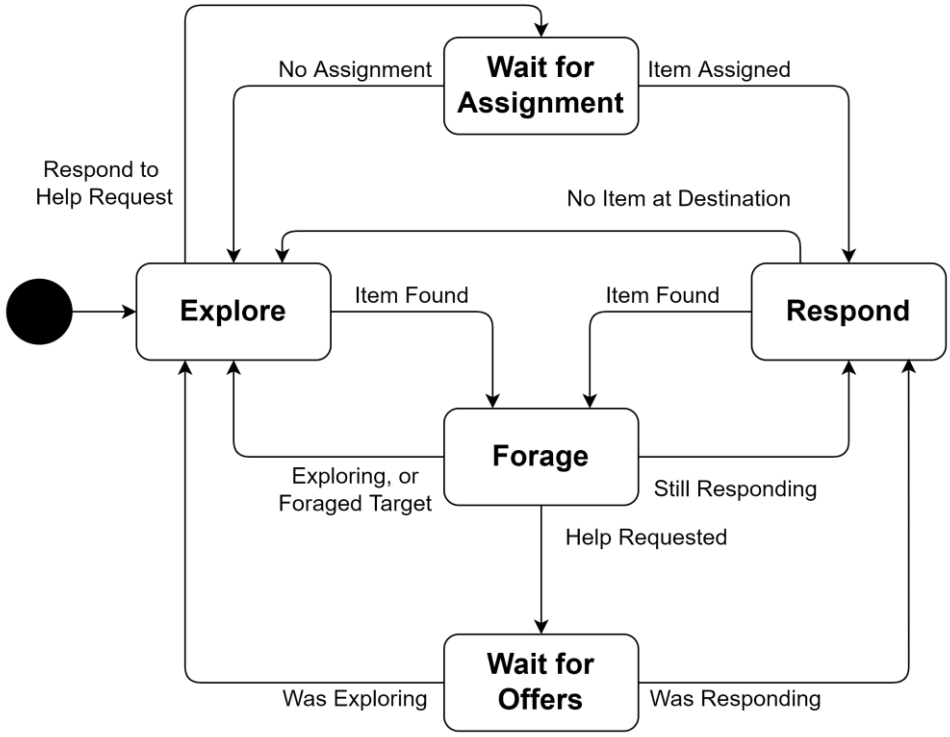


Figure 3. State Machine for the Help Recruitment cooperation strategy.

C. Communication and Energy

Communication within the simulation works by queuing each message, and processing all messages at the end of each simulation tick. Each message has a point of origin and a range, and this is used to ensure that it is sent to all robots within range. Each robot stores the received messages in a list, and will process them at the beginning of their next simulation tick, but before doing so it will shuffle the list in order to avoid the update order of robots affecting the underlying behaviour.

To explain why this is necessary, consider a robot requesting help, and receiving a response from two robots the same distance away. If the list of received messages is not shuffled, the first of these robots to be updated by the simulation would always be the robot selected to assist. When the message list is shuffled, the second robot has an equal chance of being selected.

Communications within the simulation are further affected by a global quality setting, represented as a percentage. The effective range of each message is then determined by multiplying the message's intended broadcast range by this percentage expressed as a value between 0 and 1, hence a value of 50% results in a message only reaching half the distance it intended. This is a very simple approach, however it allows for the conditions faced by the robots to change in a way that requires recognition and action.

The energy expended by each robot is measured by assigning each action a cost, in arbitrary units. Each robot has an upkeep cost of 1 unit per tick, which is incurred in addition to the costs of actions taken. Foraging an item costs 1 unit, while movement costs 1 unit per cell moved, or 1.41 units when moving diagonally. The energy cost of communications depends on the maximum range of the broadcast, prior to the effects of the global communications quality setting. This is applied using the power law stated in (1), where r is the range of the broadcast in cells.

$$cost = 0.01 \times r^2 \tag{1}$$

Measuring the energy expended by the robots is included as a means of exploring the potential impacts of using increased broadcast ranges within the swarm.

IV. Autonomic Robots

When robots require assistance, they send a help message with a broadcast range set prior to the mission. However, as robots that receive a message will pause awaiting assignment, messages that reach a higher number of potential helpers may have a negative impact on the performance of the swarm. In addition, as the energy costs of broadcasting for help increase exponentially with distance, shorter broadcasts are preferred.

A preliminary study was carried out to determine best performing broadcast range for each robot density. The foraging task was conducted using a variety of swarm sizes and broadcast ranges, and selecting the best performing range for each swarm size. The density of each robot type was calculated as in (2),

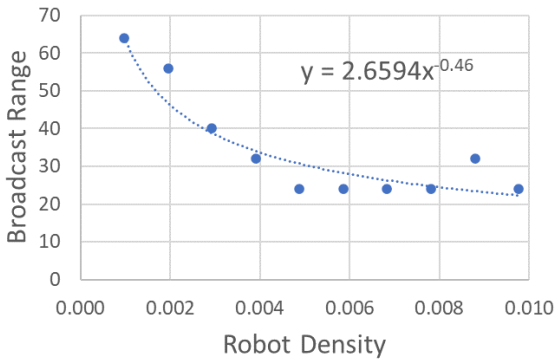


Figure 4. Derivation of the ideal broadcast range function. The points indicate the best performing range for the given density, based on mean ticks to completion.

$$\delta = r / A, \tag{2}$$

where δ is the density of robots of a given type, r is the number of robots of that type, and A is the area of the map in square cells. Using this information allowed a plot of ideal broadcast range against density, as shown in Figure 4. Fitting an approximate trend line to the plot leads to an equation for determining the broadcast range to use based on the density of the smallest group of robots, as in (3).

$$range = 2.6594 \times \delta_{min}^{-0.46} \tag{3}$$

The broadcast range used by the robots in the swarm may be set prior to the mission based on the known densities and operating area, however if an event occurs which changes these parameters mid-mission, or the parameters are not known ahead of time, then it is necessary to give the swarm the ability to manage the broadcast range itself.

This has been accomplished in two ways, employing either a centralised or a decentralised system. The following subsections describe how these approaches work.

A. Centralised Autonomic Manager

In the centralised approach, each robot sends a pulse message to a Central Autonomic Manager (CAM) every 16 simulation ticks, containing its type, exploration information, and the maximum distance from which it received messages from neighbours since the last pulse.

With the same period, the CAM can use the data to determine the composition of the swarm, estimate the area of the map, and detect any changes in communications quality.

Composition is determined simply by counting the number of received messages from each type of robot since the last period.

The area of the map is calculated by having each robot keep track of the rectangular bounds within which it has thus far explored. As each is received by the CAM, it updates its own rectangular bound, increasing it in size on each axis as needed to contain the latest information. The

area of this rectangle will then approximate the area of the map within which the robots operate.

To detect changes in communications quality, the robots are required to send out local pulse messages with a range of 8 cells. Each pulse contains the position of the robot that sent it. Between each update to the CAM, a robot keeps track of the maximum distance of a received pulse, and that information is sent in its message to the CAM. The CAM similarly tracks the maximum pulse range received. As a randomly scattered swarm may expect to receive pulses up to the expected range of 8 cells, a lower value can be used to indicate a drop in communications quality, with the perceived quality being calculated as in (4), where $range_{max}$ is the maximum received pulse range, and the denominator of 8 is the broadcast range of any given pulse.

$$quality = range_{max} / 8 \quad (4)$$

The CAM uses the information obtained each period to calculate the density of each robot type within the world, and thus the ideal broadcast range using (3). This is then divided by the perceived communications quality in order to compensate, or if the quality is zero, the CAM can instruct the robots to avoid sending help requests to avoid wasted energy and time.

B. Decentralised Autonomic Manager

In the decentralised approach, each robot has a Decentralised Autonomic Manager (DAM), which only sends a local pulse every 32 ticks with a fixed broadcast range of 8 cells, containing the sending robot type. Between each pulse, a robot updates a count for each robot type from which it receives a pulse message.

For each period, the robot calculates a density value as in (2), with the area used being that of a circle with the same radius as the pulse messages. If this density is non-zero, it is then used to calculate an ideal broadcast range using (3), which the robot uses for any subsequent help requests. However, if no messages were received, the calculated density is zero, so the robot sets the broadcast range at twice the pulse range, reasoning that no robots are within the pulse range, but may potentially lie just outside it.

The DAM does not attempt to determine the communications quality. Instead, it is assumed that if the quality drops, the number of received messages from other robots may also drop, resulting in a lower density of robots in the local area and a corresponding increase in attempted broadcast range.

It can be noted that this approach does not take into account swarm-level knowledge, and is instead localised adaptation based on the knowledge available to individual robots. However, there is some degree of knowledge about the swarm as each robot is aware of the composition of their local neighbourhood. The focus here is on implementing a method by which the robots can correctly adjust their broadcast range without relying on any central system.

V. Test Scenarios

In determining the viability of the two autonomic approaches, three sets of tests were conducted for each of autonomic approach. First, the performance of the swarm was measured in a set of fixed scenarios. Second, the ability of the AMs to react to a sudden drop in the number of robots in the swarm was tested. The third test was to determine the AMs' abilities to react to a change in communications quality.

Each of the tests was conducted in a 128x128 map, seeded with 256 items which were equally distributed between red and white types. Each of the setups within a test scenario was run 100 times to obtain a sample of results, and the performance of the swarm was measured based on the simulation ticks taken to successfully forage all items. In addition to performance data, the energy costs for the entire swarm have been measured to determine the relative efficiency by which the robots complete the task.

The following subsections describe each of the test scenarios.

A. Autonomic Manager Performance

To test the hypothesis that the autonomic managers used are capable of determining a suitable broadcast range for the swarm and perform no worse than the appropriate fixed range for a given swarm density, tests were performed comparing the swarm using a set fixed broadcast ranges, and then using each autonomic manager.

The fixed broadcast ranges used were 4, 8, 16, 24, 32, 40, 48, 56 and 64 cells. The tests were repeated with 64, 128 and 256 robots, always equally distributed between the two types.

The mean time taken by the swarm over the 100 runs was then compared across each test setup, with the best performing fixed broadcast range identified, and then compared to both the centralised and decentralised autonomic approaches.

B. Robot Destruction

To test the ability of the autonomic managers to recognise a sudden change in the swarm composition, an event was set up to occur after 300 simulation ticks, in which an equal number of robots of each type are removed from the simulation. This has the effect of decreasing the density of the swarm, which the AMs should be able to detect and react accordingly.

The test with 256 robots was run in four scenarios, with the percentage of robots destroyed in each scenario being 25%, 50%, 75% and finally 90%. These tests were carried out using the best performing fixed broadcast range identified in the autonomic manager performance tests described previously, and also with each AM.

The mean time taken by the swarm over the 100 runs was then compared across each test setup, and also compared to the corresponding fixed broadcast range test with no robot destruction, so it may be seen how the robot destruction affects performance, and the impact of using an autonomic manager.

C. Communications Quality Change

To test the ability of the autonomic managers to react to a change in the communications quality, an event was set up

to occur after 300 simulation ticks, in which the communications quality was changed from its initial setting to another value. This has the effect of either reducing or increasing the effective broadcast range of the robots, and the AMs should be able to adjust.

The test with 256 robots was run in four scenarios, with the communications quality changes being 100%-25%, 25%-100%, 100%-0% and 0%-100%.

As in the robot destruction scenario, these tests were carried out using the best performing fixed broadcast range identified in the AM Performance scenario, and then with each AM.

The mean time taken by the swarm over the 100 runs was then compared across each test setup, and also compared to the corresponding fixed broadcast range test without any change in communications quality, so it may be seen how the change affects performance, and the impact of using an autonomic manager.

VI. RESULTS

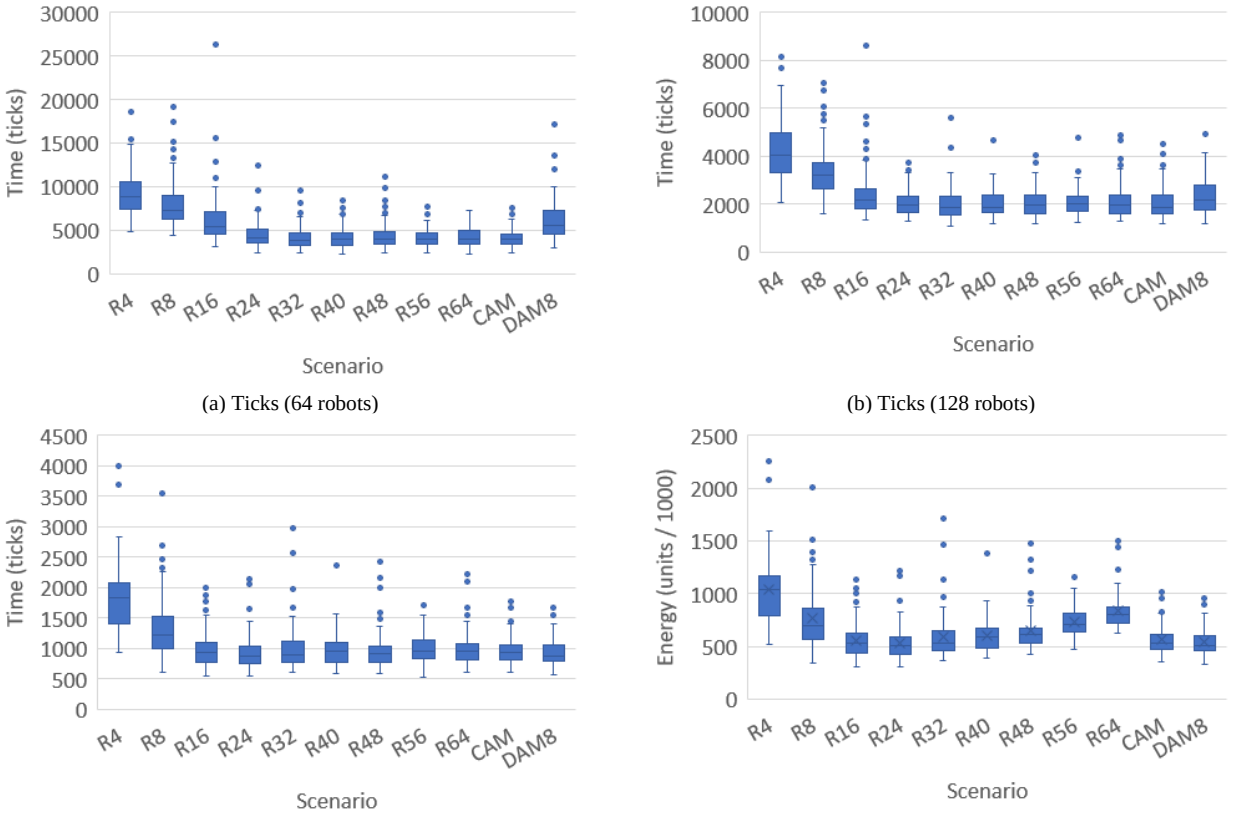
The following subsections discuss the results of the three main test scenarios, followed by an additional focus on the DAM pulse period, and an overall summary.

A. *Autonomic Manager Performance*

Figure 5 shows the simulation ticks taken for task completion by swarms of 64, 128 and 256 robots respectively, and the energy costs for the 256-robot swarm. Independent t-tests were performed between the identified best broadcast range for each swarm size, against the performances of both the CAM and DAM, and the results of this are summarised in Table I. The target is for the AMs to match or better the performance of the swarm, defined as taking fewer simulation ticks to complete, using a fixed broadcast range, and this is shown by either $p \geq 0.05$ (no statistical difference, hence performance is matched) or by $p < 0.05$ and $t > 0$ (AM performs better).

It can be seen that the CAM is capable in each case of performing as well as the best performing fixed broadcast range, completing the task in a similar time. It can also be seen that while the performance of the swarm does not appear to degrade with higher broadcast ranges, the energy cost does increase, and so using a CAM may be more efficient than simply selecting a high broadcast range. However, it should be noted that the cost of communications between a CAM and individual robots has not been factored in during this work.

The results show that the DAM with a pulse range of 8



(a) Ticks (64 robots)

(b) Ticks (128 robots)

(c) Ticks (256 robots)

(d) Energy (256 robots)

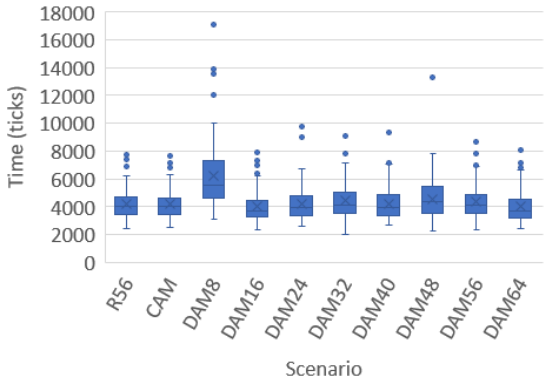
Figure 5. Ticks taken for a each swarm of robots to complete the task (a-c), and energy usage for the 256-robot swarm (d), for each broadcast range tested, and the two AMs. Circles represent outliers in the data.

TABLE I. AM PERFORMANCE T-TEST RESULTS

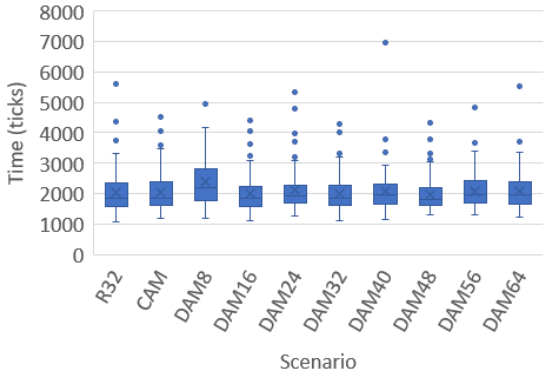
Swarm Size	Ideal Range	Fixed Range		CAM		Deg. Of Freedom	t-statistic	p-value
		Mean	SD	Mean	SD			
64	56	4151.59	1031.553	4153.54	1044.386	198	-0.013	0.989
128	40	2022.26	564.917	2024.05	596.051	198	-0.022	0.983
256	24	936.90	267.334	983.77	248.969	198	-1.283	0.201
				DAM8				
64	56	4151.59	1031.553	6230.53	2347.897	135.847	-8.107	0.000
128	40	2022.26	564.917	2382.46	809.580	176.932	-3.649	0.000
256	24	936.90	267.334	948.82	250.759	198	-0.325	0.745
				DAM16				
64	56	4151.59	1031.553	4024.84	1152.987	198	0.819	0.414
128	40	2022.26	564.917	1987.78	611.588	198	0.414	0.679
256	24	936.90	267.334	945.80	258.645	198	-0.239	0.811

cells does not perform as well when the swarm size is low. The pulse messages sent between robots to facilitate the DAM approach will be subject to the same performance degradation as the help broadcast messages, in lower robot densities.

Figure 6 shows the performance of swarms of 64 and 128 robots, with a variety of DAM configurations with pulse ranges set between 8 and 64 cells. These results show



(a) 64 robots



(b) 128 robots

Figure 6. Ticks taken for each swarm to complete the task, comparing performance of the DAM with various pulse ranges. Circles represent outliers in the data.

that an increase of the pulse range to 16 cells will correct this problem for these swarm sizes. Table I includes the results of t-tests comparing the performance of a fixed broadcast range to the DAM with pulse ranges of 8 and 16 cells.

This suggests that some means of adjusting the pulse range dynamically based on the local robot densities may be useful to allow the swarm to not only adapt its help broadcast range, but also its own autonomic processes.

B. Robot Destruction

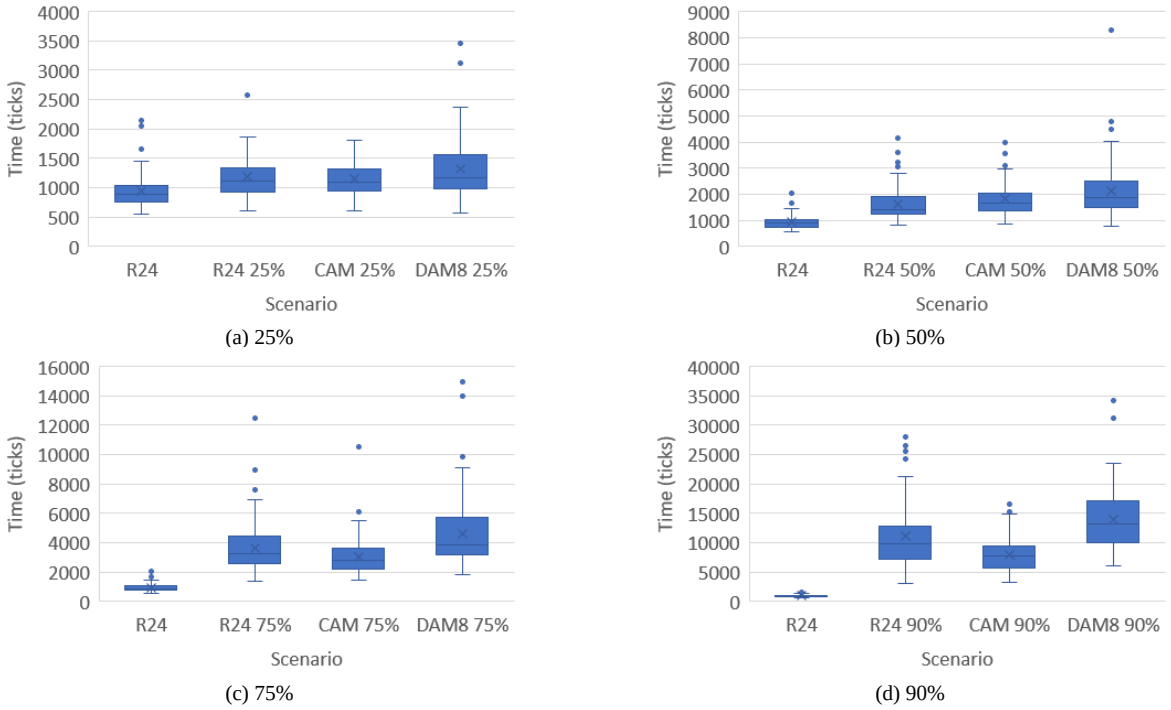
Figure 7 shows the performance of the swarm for the robot destruction scenario, for 25%, 50%, 75% and 90% of robots destroyed. Independent t-tests were performed between the identified best broadcast range from the previous subsection, and the performances of both the CAM and DAM, and the results of this are summarised in Table II. The target is for the AMs to exceed the performance of the fixed broadcast range, which is the case where $p < 0.05$ and $t > 0$.

Comparing the performance of the swarm in the two fixed-range scenarios shows that the destruction of robots has a detrimental impact on the swarm's performance, as can be expected as reducing the size of the swarm means fewer robots are left to complete the same task.

The results show that for the case where 25% of robots are destroyed, the CAM performs as well as the fixed broadcast range but confers no advantages to the swarm, while at 50%, it appears to perform worse. However, as the number of robots destroyed increases, the CAM starts to show its worth, completing the task faster in both the 75% and 90% destruction scenarios. Throughout, the energy cost correlates with the performance.

As more robots are destroyed, it appears that the CAM's ability to increase the help broadcast range, and therefore increase the chances of remaining robots successfully signalling for other robots to assist in foraging an item, results in an improvement in the swarm's performance over retaining the original fixed broadcast range.

As with the AM Performance tests, the results for the DAM show that a pulse range of just 8 cells does not allow



(a) 25% (b) 50%

(c) 75% (d) 90%

Figure 7. Ticks taken by the swarm to complete the task during the Robot Destruction scenario. Circles represent outliers in the data.

it to improve performance, and in fact it performs worse than even a fixed broadcast range. Figure 8 shows the performance of the swarm using a variety of DAMs with pulse ranges set from 8 to 64 cells, and compares against the fixed broadcast range and the CAM. It can be seen that increasing the pulse range to 24 cells would produce similar performance to the CAM, again highlighting the need for the ability to dynamically change this value. T-tests for the DAM for 90% robot destruction, with pulse ranges of 8 and 24 cell respectively are shown in Table II, showing that while DAM8 performs worse than the fixed range ($p < 0.05$, $t < 0$), DAM24 performs better ($p < 0.05$, $t > 0$).

C. *Communications Quality Change*

Figure 9 shows the performance of the swarm in the communications quality change scenarios. Independent t-tests were run comparing the fixed range performance with that where the AM is active, and the results are summarised in Table III. The equivalent tests comparing energy usage are shown in Table IV.

It is only when the communications quality drops from 100% to 25% that the CAM confers any advantage to the swarm, performing better than the fixed broadcast range ($p < 0.05$, $t > 0$), and performing as well as the swarm without any change in communications quality, suggesting it is successful in counteracting the reduction in effective broadcast range. It can also be seen that despite an effective four-fold increase in the broadcast range it uses, which amounts to a 16-fold increase in the cost of each broadcast, the energy requirements are still lower than using a fixed

TABLE II. ROBOT DESTRUCTION T-TEST RESULTS

Destroyed Robots / %	**R24**		**CAM**		**Deg. of Freedom**	**t-statistic**	**p-value**
	Mean	Std. Dev.	Mean	Std. Dev.			
25	1187.51	342.602	1150.36	303.995	198	0.811	0.418
50	1630.59	617.813	1811.13	606.876	198	-2.085	0.038
75	3664.05	1709.930	3041.41	1226.075	179.516	2.959	0.004
90	11196.18	5458.347	7974.65	2753.011	146.305	5.270	0.000
			DAM8				
90	11196.18	5458.347	13930.47	5191.608	198	-3.630	0.000
			DAM24				
90	11196.18	5458.347	7831.72	2820.210	148.341	5.476	0.000

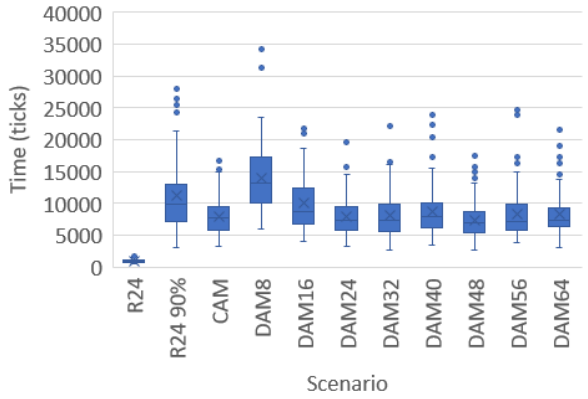


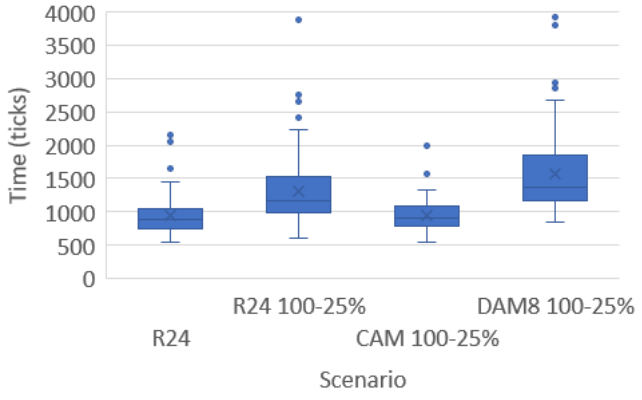
Figure 8. Ticks taken for the swarm during the 90% Robot Destruction scenario, for each DAM configuration. Circles represent outliers in the data.

broadcast range ($p > 0.05$, $t > 0$).

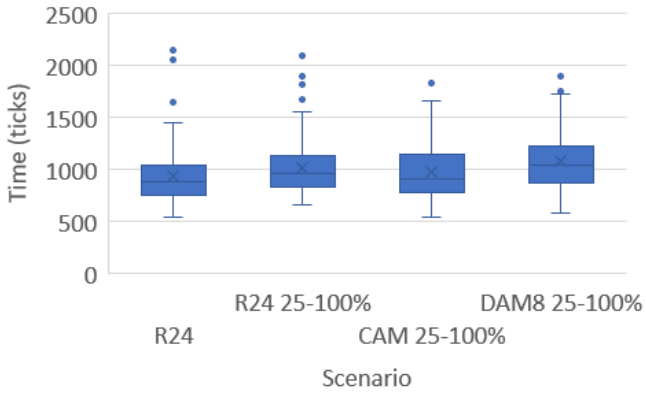
In the other scenarios, we do not see any benefit to using a CAM. In the 25%-100% change, the CAM uses more energy than using a fixed broadcast range ($p < 0.05$, $t < 0$). To explain this, it is likely that during the first 300 ticks of the simulation, where the communications quality is reduced to 25%, there are much more items in the world for robots to find, meaning a much higher chance of an item being found by a robot of the wrong type, and therefore many more help requests being sent. In the 100%-25% scenario, there are fewer items remaining in the world by the time the communications quality drops, and so fewer help requests are sent.

In the cases where the communications quality begins or ends at 0%, no statistical differences can be seen between the CAM and the fixed broadcast range. This is likely because at 0% communications quality, no cooperation is possible, and the performance of the swarm is dominated by the random search for items.

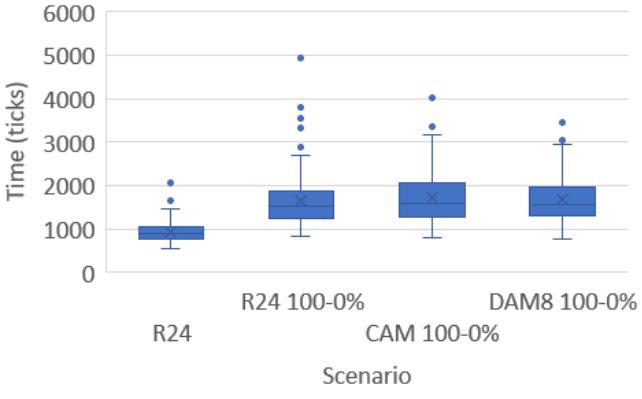
Focusing on the 100%-25% scenario for the DAM, we again see that the 8 cell pulse range is detrimental to the performance of the swarm ($p < 0.05$, $t < 0$). Figure 10 shows the performance of the swarm with a variety of DAMs with pulse ranges set between 8 and 64 cells. It can be seen that a pulse range of 40 cells performs best in this scenario, however unlike the CAM, there is no statistical difference between the performance of the DAM and the fixed broadcast range, so no improvement is gained through its use.



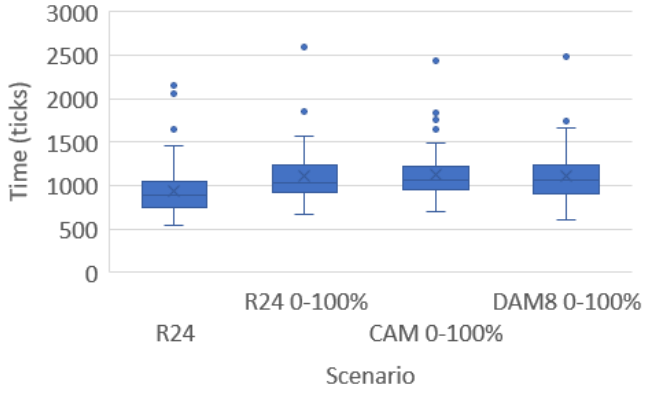
(a) 100%-25%



(b) 25%-100%



(c) 100%-0%



(d) 0%-100%

Figure 9. Ticks taken by the swarm to complete the task during the Communications Quality Change scenario. Circles represent outliers in the data.

TABLE III. COMMUNICATIONS QUALITY T-TEST RESULTS (TICKS)

Quality Change	R24		CAM		Deg. of Freedom	t-statistic	p-value
	Mean	Std. Dev.	Mean	Std. Dev.			
100 – 25%	1314.86	518.360	944.96	239.680	139.481	6.477	0.000
25 – 100%	1015.27	268.303	980.89	286.952	198	0.875	0.383
100 – 0%	1663.61	706.816	1724.27	625.697	198	-0.643	0.521
0 – 100%	1108.01	276.859	1132.90	302.442	198	-0.607	0.545
			DAM8				
100 – 25%	1314.86	518.360	1572.25	612.837	198	-3.207	0.002
			DAM32				
100 – 25%	1314.86	518.360	1031.65	358.278	198	4.494	0.000
			DAM40				
100 – 25%	1314.86	518.360	947.32	254.206	143.958	6.367	0.000

TABLE IV. COMMUNICATIONS QUALITY T-TEST RESULTS (ENERGY)

Quality Change	R24 / 1000		CAM / 1000		Deg. of Freedom	t-statistic	p-value
	Mean	Std. Dev.	Mean	Std. Dev.			
100 – 25%	746.49	292.797	563.98	136.668	140.184	5.648	0.000
25 – 100%	574.95	151.535	730.59	163.040	198	-6.993	0.000
100 – 0%	943.32	399.203	986.42	356.575	198	-0.805	0.422
0 – 100%	627.33	156.363	646.28	172.379	198	-0.814	0.417
			DAM8				
100 – 25%	746.49	292.797	899.07	349.310	198	-3.348	0.001
			DAM32				
100 – 25%	746.49	292.797	687.40	231.925	198	1.582	0.115
			DAM40				
100 – 25%	746.49	292.797	684.42	176.740	162.689	1.815	0.071

D. Decentralised AM Pulse Period

The above results show that adjusting the broadcast range of the DAM can have an impact on the performance of the swarm. In light of this, it is necessary to investigate the effects of the only other parameter in the DAM, that of the period between pulses.

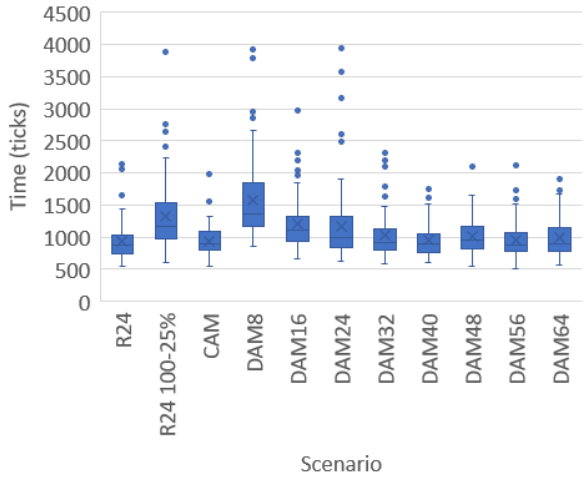


Figure 10. Ticks taken for the swarm during the 100%-25% communications quality scenario, fore each DAM configuration. Circles represent outliers in the data.

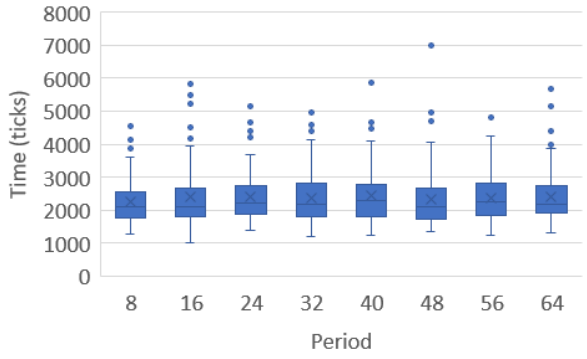
To test this, a swarm of 128 robots, each equipped with a DAM set to a broadcast range of 8 cells, was tested in the standard performance scenario, using periods of 8, 16, 24, 32, 40, 48, 56 and 64 ticks. A one-way ANOVA test was used to determine if any statistical difference exists between the resulting sets of data using the period as the independent value.

Figure 11 shows the performance and energy cost of the swarm in completing this task, and the one-way ANOVA test showed there is no statistically significant difference in the data ($F(7, 792) = [0.578]$, $p = 0.774$).

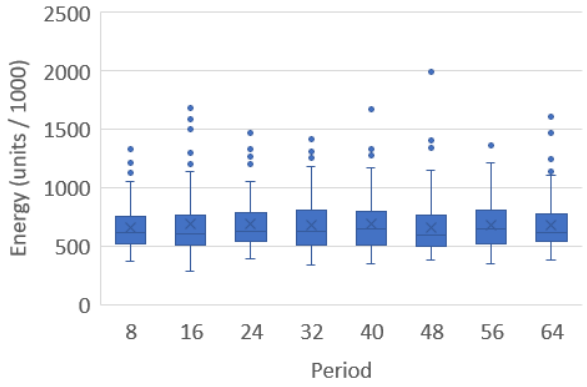
As such, the period with which the pulse broadcasts are sent does not appear to have any impact on the performance of the DAM. However, further tests may be useful to explore the impact of the period when using other broadcast ranges or swarm sizes.

E. Summary

The results above show that the presence of an Autonomic Manager can have benefits for the performance of the swarm, however it appears possible for the AM to reduce performance in some circumstances, such as the 50% robot destruction scenario for the CAM, and these will require further investigation to determine what other parameters may be affecting the swarm's performance. If



(a) Ticks



(b) Energy

Figure 11. Ticks taken (a) and energy cost (b) of a swarm of 128 robots, using a DAM8 with a variety of pulse periods. Circles represent outliers in the data.

the AM can be developed to account for these further variables, it may be able to counter their effects.

For example, if estimates of the density of items in the world can be made by the AM, this can be used to reduce the communication range when the item density is high, avoiding interruptions that occur during the help broadcasts which may lead to poorer performance.

It can also be seen that the idealised Centralised Autonomic Manager produces a performance that can be replicated by a decentralised approach, without the problems associated with having a central bottleneck for the swarm. However, the pulse range used by the system for allowing robots to detect neighbours needs to be dynamic for the DAM to account for unknowns either in the initial situation, or that arise through events that occur during the mission.

VII. Conclusion and Future Work

This research used a simulation of a robotic swarm equipped with two different autonomic management capabilities, able to manage the performance of the swarm through adjustment of the intra-swarm communication range.

A centralised approach was used as an idealised situation in which a central system has full knowledge of the swarm with which to act, as a means of investigating the potential for improving performance through the use of an AM. Further to this, a decentralised approach was developed that allowed individual robots to monitor their own environment and select an appropriate broadcast range without requiring any central component.

The findings show that a centralised AM is capable of finding an appropriate broadcast range when given a task where the map size and number of robots in the swarm is not initially known to the AM, and must be deduced from information gathered by the individual robots.

When a robot destruction event occurs, the centralised AM proves beneficial to the swarm when the robot loss is high, capable of completing the task faster than using a fixed broadcast range. No benefit is seen when the robot loss is low. In the case of 50% robot loss, the CAM appears to be detrimental to swarm performance. This should be further explored to determine the cause in future work.

In the event of a change in communications quality, the centralised AM is capable of improving performance when the quality drops from high to low without dropping out entirely, but not when the quality starts low and increases. This is likely due to the increased item density during the early stages of the task, and it is worth exploring this factor to see how the AM might measure and take item density into account.

A centralised system has problems that have not been replicated directly in this work, such as the potential for the central AM to be a bottleneck on performance, the need and energy cost required for individual robots to maintain a link to the central AM, and reduced autonomy of any one robot. To counter this, a fully decentralised approach was developed and tested.

The decentralised approach was found to be capable of matching the performance of the centralised system in some cases, though it confers no advantage over the fixed broadcast range when the communications quality changes. This performance is achieved despite robots not having complete knowledge of the swarm, instead tracking local pulse messages sent by neighbouring robots to estimate the swarm density. The findings however show that a means of dynamically adjusting the range of these pulse messages is required for the decentralised AM to reach the desired performance, while the pulse period does not appear to have an effect.

The simulation used in this work did not test for collisions between robots, and is explained in this case by individual cells being much larger than the robots contained within, allowing plenty of room for manoeuvre. However, it can be recognised that avoidance behaviours and the potential for high traffic bottlenecks may impact the results, and this should be considered in further investigations.

Future work will investigate methods to allow the dynamic adjustment of the pulse range, as well as exploring other scenarios in which a decentralised autonomic manager may be of use to a swarm, and the possibility of sharing decisions made by individual robots in order to help guide other robots, thus restoring the concept of swarm-level knowledge that is not present in the current decentralised approach. Additional work may also explore other situations that may affect performance, such as more complex maps

containing obstacles, differing distributions of robot types, more complexity in the foraging task, on-board batteries that drain and require recharging, and further events that may occur to unexpectedly change the world state.

REFERENCES

[1] L. McGuigan, R. Sterritt, and G. Wilkie, 'Centralised Autonomic Self-Adaptation in a Foraging Robot Swarm', in *The Seventeenth International Conference on Autonomic and Autonomous Systems (ICAS 2021) IARIA*, May 2021, pp. 11–17.

[2] H. Hamann, *Swarm Robotics: A Formal Approach*. Springer International Publishing, Cham, Switzerland, 2018.

[3] I. Navarro and F. Matía, 'An Introduction to Swarm Robotics', *ISRN Robot.*, vol. 2013, pp. 1–10, 2013.

[4] L. Bayindir, 'A review of swarm robotics tasks', *Neurocomputing*, vol. 172, pp. 292–321, Jan. 2016.

[5] R. Sterritt, G. Wilkie, C. Saunders, M. D. C. Gama, G. Hawe, and L. McGuigan, 'Inspiration for Space 2.0 from Autonomic-ANTS (Autonomous NanoTechnology Swarms) Concept missions', presented at the Reinventing Space Conference, 2019.

[6] A. Farahani, G. Cabri, and E. Nazemi, 'Self-* properties in collective adaptive systems', in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, Heidelberg, Germany, Sep. 2016, pp. 1309–1314.

[7] G. Beni, 'From Swarm Intelligence to Swarm Robotics', in *Swarm Robotics*, Berlin, Heidelberg, 2005, pp. 1–9.

[8] L. McGuigan, C. Saunders, R. Sterritt, and G. Wilkie, 'Cooperation Strategies for Swarms of Collaborating Robots: Analysis of Time-Stepped and Multi-Threaded Simulations', *Int. J. Adv. Syst. Meas.*, vol. 14, no. 3 & 4, pp. 44–58, 2021.

[9] J. Prasetyo, G. De Masi, and E. Ferrante, 'Collective decision making in dynamic environments', *Swarm Intell.*, vol. 13, no. 3, pp. 217–243, Dec. 2019.

[10] N. Bredeche, E. Haasdijk, and A. Prieto, 'Embodied evolution in collective robotics: A review', *Front. Robot. AI*, vol. 5, p. 12, 2018.

[11] C. Saunders, R. Sterritt, and G. Wilkie, 'Collective Communication Strategies for Space Exploration', *J. Br. Interplanet. Soc.*, vol. 72, no. 12, pp. 416–430, 2019.

[12] J. O. Kephart and D. M. Chess, 'The vision of autonomic computing', *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[13] E. Vassev, R. Sterritt, C. Rouff, and M. Hinchey, 'Swarm Technology at NASA: Building Resilient Systems', *IT Prof.*, vol. 14, no. 2, pp. 36–42, Mar. 2012.

[14] J. Zelenka, T. Kasanický, and I. Budinská, 'A Self-adapting Method for 3D Environment Exploration Inspired by Swarm Behaviour', in *Advances in Service and Industrial Robotics*, Cham, 2018, pp. 493–502.

[15] K. S. Kappel, T. M. Cabreira, J. L. Marins, L. B. de Brisolara, and P. R. Ferreira, 'Strategies for Patrolling Missions with Multiple UAVs', *J. Intell. Robot. Syst.*, vol. 99, pp. 499–515, Sep. 2019.

[16] M. Puviani, G. Cabri, and F. Zambonelli, 'Agent-based Simulations of Patterns for Self-adaptive Systems':, in *Proceedings of the 6th International Conference on Agents and Artificial Intelligence*, ESEO, Angers, Loire Valley, France, 2014, pp. 190–200.

[17] M. Allison, M. Robinson, and G. Rusin, 'An Autonomic Model-Driven Architecture to Support Runtime Adaptation in Swarm Behavior', in *Advances in Information and Communication*, Cham, 2020, pp. 422–437.