Aemass: robotic life in a petri dish

Stephen Makonin
School of Computing Science
Simon Fraser University, Burnaby, BC, Canada
smakonin@sfu.ca

Abstract

Aemass is a robotic organism that lives in a Petri Dish. It can eat food and excrete waste. If Aemass cannot eat food at a rate greater than or equal to the rate that it excretes waste Aemass will slowly shrink and then die; therefore, an unsustainable environment. Can we make this environment self-sustainable? If so, how? What if Aemass is really not an organism, but a swarm of robots. What emergent behaviours would it have?

Introduction

Aemass is a robotic organism that lives in a Petri Dish. It can eat food and excrete waste. If Aemass cannot eat food an a rate greater than or equal to the rate that it excretes waste Aemass will slowly shrink and the die; therefore, an unsustainable environment. Can we make this environment self-sustainable? We define self-sustainability as a system that can independently maintain itself without outside effort. What if Aemass is really not an organism, but a swarm of robots. What emergent behaviours would it have?

Swarm robotics and swarm intelligence are relatively new fields in the artificial intelligence and artificial life landscape. Projects such as I-Swarm (Seyfried et al., 2005), Microbotics (Abbott et al., 2007), and Symbrion (Kernbach et al., 2008) are some examples of where the field of swarm robotics is focused.

When looking at artificial life and swarms we focus on exploring the idea of flocking; and algorithm Reynolds introduced in 1986 labeled Boids (Reynolds, 1987). Simply, we explore the use of a simple robot controller to form a swarm that has emergent behaviours that no one robot could have.

These emergent behaviours can be seen in nature as ant ant bee (to name a few) colonies (Bonabeau et al., 1999). No one ant or bee could survive but it can as part of a colony. And, what interests us in the area of swarm robotics is creating a self-sustainable system in which a swarm of robots (which we call Aemass) lives in a circular world (which we call Petri Dish).

Further more, these robots do not communicate with each other, nor do they have any idea of a global position or location. They only know what they locally see. What each robot sees are obstacles and other robots that are distinguished by their state which is reflected by their fiducial identifier.

We develop our system using a simulator called Stage (Vaughan, 2008) to show how Petri Dish can be a self-sustaining environment and to show the emergent behaviours of Aemass. We investigate the issue of battery size as it pertains to the plausibility of self-sustainability in both an adversarial and cooperative environment. This adversarial and cooperative environment difference is distinguished in how willing or unwilling food allows itself to be consumed. To our knowledge no one has published literature about an environment like this before.

We will first discuss the Petri Dish at a distance and describe a high level view of our system (Section 2.1). We then take a closeup view of Petri Dish and describe this view and how is is different than the view from afar (Section 2.2). We continue with a discussion of the principles and design constraints (Section 2.3) of our Petri Dish world. Aemass and the robot controller design is discussed in Section 3 and self-sustainability in Section 4. We then design a series of experiments that explore the ideas of self-sustainability and the experimental results/findings (Section 5).

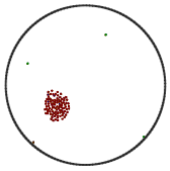

Figure 1: Petri Dish at a distance. Aemass is the larger red organism. Food is the smaller green organisms.

Life in Petri Dish

Clearly, as with most other swarm systems, there are two views to study and consider. A view that sees Aeamass as

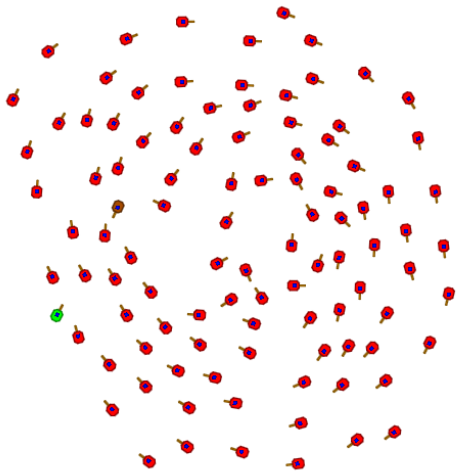

Figure 2: Aemass is a swarm of robots that travel in a clockwise circular path around a perceived center of mass that each *drone* has. This emergent behaviour is also known as flocking.

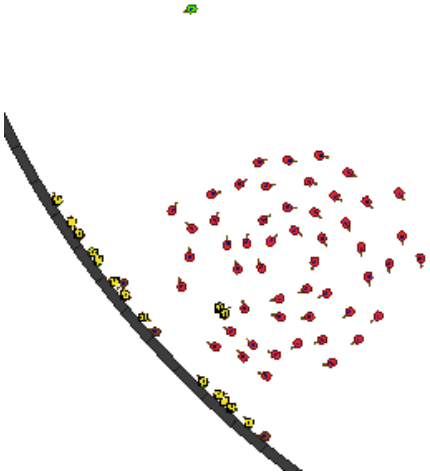

Figure 3: An example of wall crowding when Aemass does not have the emergent behaviour of distancing itself from walls.

a single organism, and a view that sees Aemass as a swarm of robots (as our next two sections discuss). We also want to add some constraints that have an effect on how robots in Petri Dish live.

At a Distance

Our robotic world is a circular arena we call Petri Dish which is shown in Figure 1. Aemass is the larger red organism. Food are smaller green organisms and waste are smaller brown organisms. Waste can be seen traveling towards the Petri Dish wall where it will regenerate and become food. If Aemass does not consume enough food (at a rate greaterthen or equal to the rate it excretes waste) it will continually shrink, and eventually die.

Petri Dish can run in two environment modes *adversarial* and *cooperative*. Food organisms are naturally curious and tend to travel close to other organisms (except waste) when not alone and foraging. In both environment modes food can recognize Aemass when it sees enough of it (explained in more detail in the next section). This is where the similarity between modes ends.

In adversarial environment mode, when food realizes that it is too close to Aemass it will try to escape so it is not consumed; a self preservation mechanism. However, if after a period of time this does not happen food will become a part of Aemass. At the same time Aemass tries to surround the food once it sees that food is near.

In cooperative environment mode, when food realizes that it is close to Aemass it will freely allow itself to be consumed; a euphoric sense of wanting to belong. In this case, it is easier for Aemass to surround the food and consume it.

In both modes Aemass possesses the desire to stay away from walls so it dies not crowd up against them. Such crowding would have a negative effect on the lifespan of Aemass.

Up Close

Up close (as shown in Figure 2) Aemass is actually a robot swarm. Drone robots are part of the swarm. Food robots join the swarm (willingly or unwillingly). Energy-depleted waste robots leave the swarm to find a recharging station to recharge. Once recharged waste robots regenerate and become become food. These stages (or states) continually repeat for every robot in the Petri Dish to make a complete and continuous life cycle.

The individual robots that make up Aemass display a collective or emergent behaviour that one robot possesses. We define emergent behaviour in the exact same way 1,730 (a search on google.com) websites and blogs have defined: “emergent behaviour arising from simple rules that are followed by individuals and does not involve any central coordination”. Therefore, we have designed in constants to meet this definition (as detailed in the following section). We have found that Aemass possesses 3 truly emergent behaviours: food surrounding, avoiding walls, and flocking.

Aemass develops its first emergent behavior (food surrounding, **E1**) from the actions of a number of individual drone robots. When a drone robot identifies food in front of it, it speeds up and tries to drive around it so the the food is between it and the swarm. When enough individual robots do this then the food is indeed surrounded and cannot escape.

Aemass develops its second emergent behavior (distancing itself from walls, **E2**), again, from the actions of a number of individual drone robots. When a drone robot identi-

fies a wall it will steer away from it. Individual robots doing this results in Aemass distancing itself from walls. Without this behaviour Aemass would crowd against a wall which would cause a number of waste robots to bang up against each other. This, in turn, causes the population of operative robots to shrink. As robots leave Aemass (as waste) the size of Aemass shrinks. When the size is too small Aemass will die. This *wall crowding* effect is shown as Figure 3.

Aemass develops its third emergent behavior (flocking, **E3**) from the actions of a number of individual drone robots. Each drone attempts to travel close to and around its perceived center of mass in the swarm. Each drone drives in a clockwise circular direction as shown in Figure 2.

Design Principles & Constraints

There are a number of design principles and constraints that we use. They are listed here.

1. Individual robots cannot communicate with each other. The actions they take are independent of each other.

2. Individual robots do not have a GPS or another global positioning sensor. They do not have any knowledge of global positions or locations.

3. Individual robots use a sick laser and ranger to identify obstacles (within a local view) and prevent collisions.

4. Individual robots have a fiducial sensor (5 meter range) to identify what state another robot is in (e.g. drone, waste, charge, and food).

5. Each and every robot must use the same robot controller. This controller can have distinguished behaviours based on a number of different states (see Figure 4).

6. State changes (like *escape*, explained in the Robot Controller section) are governed by a universal law: *the Law of Swarm Attraction* (LoSA). This is the number of times an event, observation, or condition needs to occur before a change can happen. For example, we have set this to 16.

With these constraints all robots are without any central coordinating systems and thus the emergent behavior that Aemass possesses is truly emergent.

In a previous section we mentioned that "food can recognize Aemass when it sees enough of it". By this we mean that we use LoSA. If food sees a number of drones $\geq$ LoSA then food recognizes that it has identified the swarm.

Robots of Aemass

Each and every robot uses the same robot controller. In this section we detail how the robot controller works as a finite state machine (FSM). For our robot controller we used modified parts of existing Stage robot controllers (Pioneer_Flocking and Fasr to be exact). A modified version of

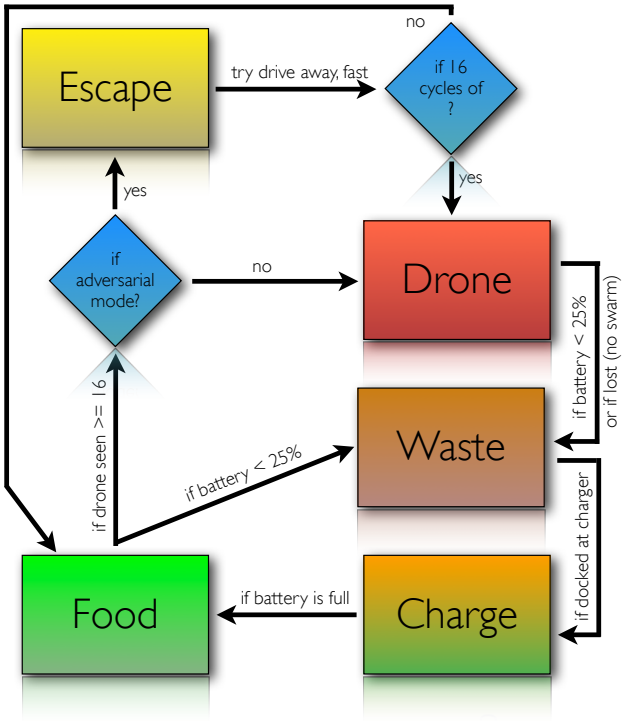

Figure 4: State diagram of the robot controller uses for all robots.

the flocking algorithm was used from Pioneer_Flocking and the obstacle avoidance and recharging algorithms used from Fasr.

At any given time a robot is in one of 4 main states: *drone*, *food*, *waste*, and *charge*. If Petri Dish is in an adversarial mode then *food* has an additional sub-state *escape*. This follows the continuous life cycle as mentioned previously. The state diagram of our robot controller is shown in Figure 4 and explained in detail below.

Drones

When a robot is in the *drone* state it travels around its perceived centre of mass. This creates the swarm and from faraway makes the swarm look like a large organism (**E3**). If a robot that is in the state of *food* travels within the fiducial sensor range of a *drone*, the *drone* attempts to stay near the swarm and drive around the *food*. When a number of *drones* do this the swarm can capture *food* by preventing it from escaping (**E1**). The swarm moves around Petri Dish slowly over time, but there is no code specifically purposed to do this; it just happens. *drones* distance themselves away from walls (using a repulsive bearing force). This in turn allows Aemass to avoid crashing into walls (**E2**) and what we call a wall crowding effect (as shown in Figure 3) where robots end up stalled along the walls.

Food

When a robot is *food* it wanders around the Petri Dish. *food* is programmed to be curious and is attracted to other robots. If *food* finds the swarm it may react in 2 different ways depending on the environment mode. If the environment mode is cooperative, *food* will join the swarm if it sees $\geq$ LoSA *drones*. If the environment mode is adversarial, once *food* sees $\geq$ LoSA *drones* it avoids joining the swarm by changing its state to *escape*. If the *food* cannot escape in $\geq$ LoSA time units of seeing $\geq$ LoSA *drones*, then it becomes a *drone*. However, if this does not happen a robot in the *escape* state will change back to the *food* state once there are no *drones* in sight.

Waste & Charge

If the battery level of any robot in any state becomes lower than 25% of the total capacity, then the robot's state changes to *waste*. That robot will search for a charging station. If a *drone* wanders too far from the swarm and cannot return after a period of time (based on LoSA) then it will also become *waste*.

Once *waste* finds a charging station it docks and its state changes to *charge*. Once docked, the robot is recharged. The walls of Petri Dish are the charging stations. Robot battery charge levels are randomly set at initialization so that all robots do not need to recharge at the same time causing Aemass to die (i.e. no more swarm). Robot battery sizes vary by experiment, but in each experiment all robots have the same battery size. More about battery sizes are discussed in the next 2 sections. We believe that the size of the battery has direct influence on whether Petri Dish is a self-sustaining environment.

Self-Sustainability

As we defined earlier for our purposes, self-sustainability is a system that can independently maintain itself without outside effort. If we were to plot a line graph of *swarm size over time* we would see different scenarios that would depict how self-sustainable a Petri Dish could be. These 3 scenarios would be: (**S1**) considered self-sustainable, (**S2**) considered unsustainable, and (**S3**) considered temporarily self-sustainable. See Figure 5. **S2** and (at some point) **S3** become unsustainable because the rate at which robots leave the swarm as waste is $>$ the rate that food robots become part of the swarm.

Hypotheses

We hypothesize that the robot's battery size directly affects whether the Petri Dish is self-sustainable (**S1**) or not (**H1**). We further hypothesize that an adversarial environment would require robots to have a larger battery size (and ultimately use more energy) than in a cooperative environment (**H2**).

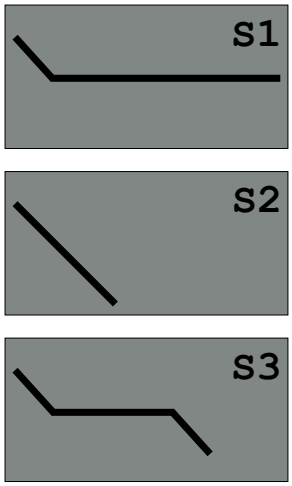

Figure 5: Three possible scenarios of testing sustainability: **S1** is considered self-sustainable, **S2** is considered unsustainable, and **S3** is only temporarily sustainable. Each scenario is a graph of swarm size over time.

We test these hypotheses by setting up a number of experiments that allows us to measure how self-sustainable a Petri Dish is, based on the battery size of the robots and environment mode.

Simulation Results & Findings

After designing a robot controller in Stage (Vaughan, 2008) we ran a number of experiments that allowed us to measure and test our hypotheses. Nineteen experiments (7 in cooperative mode, 12 in adversarial mode) were ran and we report our findings here.

Experimental Setup

For our simulated experiments we will consider 432,000 seconds (or 120 hours or 5 days) as the goal post for determining self-sustainability (**S1**). If Aemass survives that long then we will categorize the experimental test as **S1** (self-sustainability). To have a true test of self-sustainability we would need to run the experimental test forever.

The Petri Dish will have a radius of 49.5 meters and there are 104 robots (100 drones, 4 food) within it. As mentioned before, we are using Stage (Vaughan, 2008) to simulate these experimental tests. The Petri Dish walls are 2x0.5x0.8 meter in size and act as the charging stations. The walls are positioned to form a circle at runtime with a separate robot controller we call Wall.

Each robot's battery level is randomly set at runtime so that all robots do not need to recharge at the same time. Once docked the robot is instantaneously recharged fully to 100%.

An Adversarial Environment

In each experiment we changes the battery size to find the smallest battery size that would enable self-sustainability (**S1**) in an adversarial environment. Battery sizes that where tested where (in kJ): 250, 500, 625, 650, 675 , 680, 685, 690, 700, 750, 875, and 1000. We found that battery size of $\geq$ 680 kJ provided a self-sustainability (**S1**) in an adversarial environment. Figure 6 plots out 4 of these tests and Table 1 summaries all test results. This finding supports our hypothesis **H1**.

Battery Size	End Time	Swarm Size	Scenario
250 kJ	11,411 sec	–	**S2**
500 kJ	55,559 sec	–	**S2**
625 kJ	183,119 sec	–	**S3**
650 kJ	207,313 sec	–	**S3**
675 kJ	230,107 sec	–	**S3**
680 kJ	431,999 sec	55 robots	**S1**
685 kJ	431,999 sec	46 robots	**S1**
690 kJ	431,999 sec	52 robots	**S1**
700 kJ	431,999 sec	57 robots	**S1**
750 kJ	431,999 sec	66 robots	**S1**
875 kJ	431,999 sec	52 robots	**S1**
1000 kJ	431,999 sec	71 robots	**S1**

Table 1: Results of all experimental tests ran in adversarial mode.

A Cooperative Environment

In each experiment we changes the battery size to find the smallest battery size that would enable self-sustainability (**S1**) in a cooperative environment. Battery sizes that where tested where (in kJ): 150, 160, 175, 180, 185, 200, and 250. We found that battery size of $\geq$ 185 kJ provided a self-sustainability (**S1**) in an cooperative environment. Figure 7 plots out 4 of these tests and Table 2 summaries all test results. This finding supports our hypothesis **H1**.

Battery Size	End Time	Swarm Size	Scenario
150 kJ	122,529 sec	–	**S3**
160 kJ	82,015 sec	–	**S3**
175 kJ	274,346 sec	–	**S3**
180 kJ	350,449 sec	–	**S3**
185 kJ	431,999 sec	33 robots	**S1**
200 kJ	431,999 sec	28 robots	**S1**
250 kJ	431,999 sec	46 robots	**S1**

Table 2: Results of all experimental tests ran in cooperative mode.

When comparing the results of our adversarial environment tests to our cooperative environment tests, we find that there is a large difference in battery size to achieve **S1**. A battery size of 3.67-times greater (680 kJ *vs* 185 kJ) is needed to have self-sustainability (**S1**) in an adversarial environment.

Conclusions

We have described in detail a swarm robotic system called Aemass that has a number of emergent behaviours. These emergent behaviours are: food surrounding, avoiding walls, and flocking. We then show how the size of the robot's battery can create a self-sustaining environment where Aemass can continually live. Finally, we show that a cooperative environment is over 3.5 times more energy efficient than an adversarial environment.

Code Publication

All source code and scripts used to produce the results reported in this paper is release as open source and is available online at https://github.com/smakonin/Aemass

References

Abbott, J., Nagy, Z., Beyeler, F., and Nelson, B. (2007). Robotics in the small, part i: Microbotics. *Robotics & Automation Magazine, IEEE*, 14(2):92–103.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. Number 1. Oxford University Press, USA.

Kernbach, S., Meister, E., Schlachter, F., Jebens, K., Szymanski, M., Liedke, J., Laneri, D., Winkler, L., Schmickl, T., Thenius, R., et al. (2008). Symbiotic robot organisms: Replicator and symbrion projects. In *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, pages 62–69. ACM.

Reynolds, C. (1987). Flocks, herds and schools: A distributed behavioral model. In *ACM SIGGRAPH Computer Graphics*, volume 21, pages 25–34. ACM.

Seyfried, J., Szymanski, M., Bender, N., Estana, R., Thiel, M., and Wörn, H. (2005). The i-swarm project: Intelligent small world autonomous robots for micro-manipulation. *Swarm Robotics*, pages 70–83.

Vaughan, R. (2008). Massively multi-robot simulation in stage. *Swarm Intelligence*, 2(2):189–208.

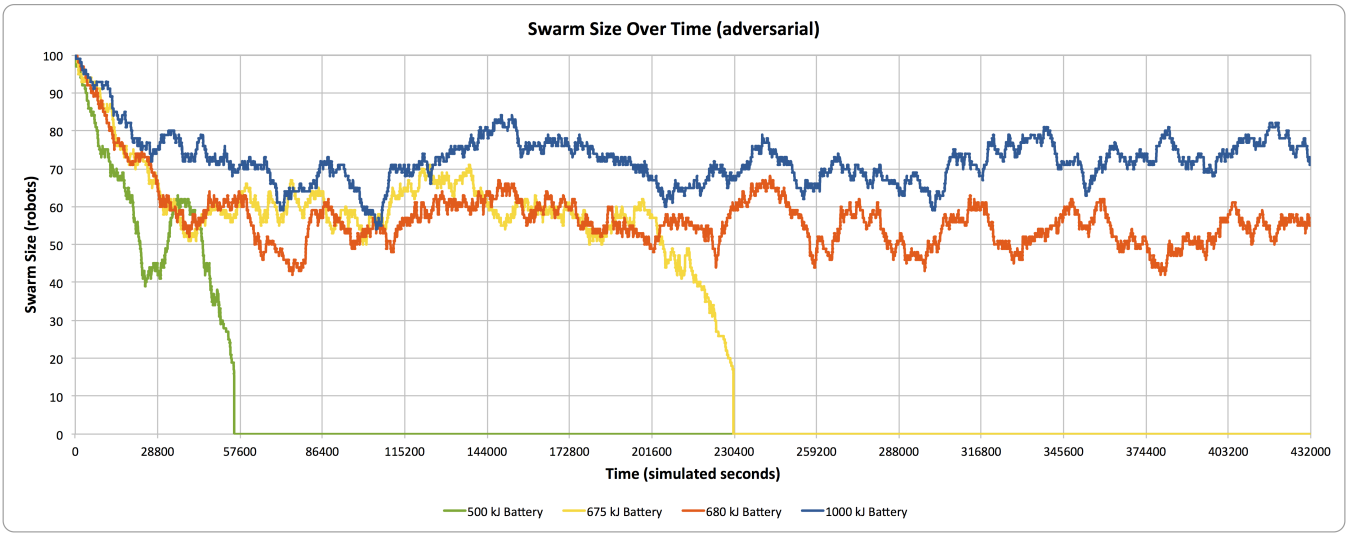

Figure 6: Battery size has an effect as to whether or not there is a self-sustainable Petri Dish. A self-sustainable environment (**S1**) resulting in having a battery size of $\geq$ 680 kJ in an adversarial environment. A battery of size 675 kJ results is a **S3** scenario and a battery size of 500 kJ results in a **S2** scenario.

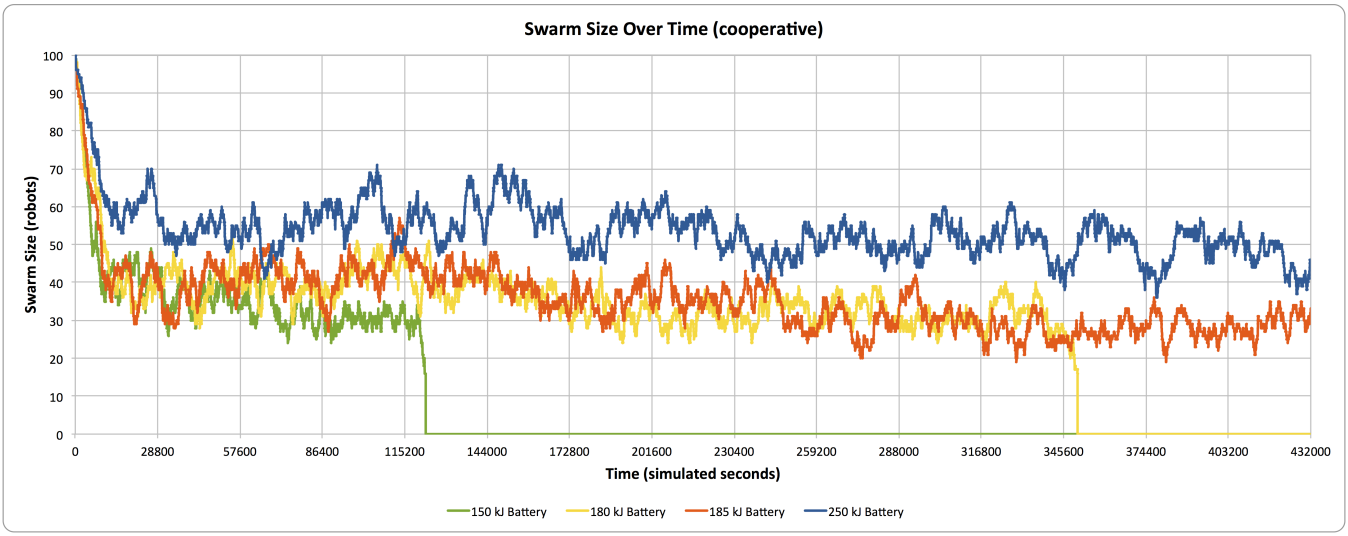

Figure 7: Battery size has an effect as to whether or not there is a self-sustainable Petri Dish. A self-sustainable environment (**S1**) resulting in having a battery size of $\geq$ 185 kJ in an cooperative environment. A battery of size 180 kJ and 150 kJ results is a **S3** scenario.