COMP2310

Systems Networks and Concurrency

Assignment - 1 Report

Hongming Zhang

U6693651

Undergraduate

26/09/2019

Australian National University

1 Problem Statement

The spacecraft energy source competition problem in this assignment is a very classic problem about using coordination to control behaviors and resources in a concurrent and distributed system. The goal of this assignment is to build a controlling model for space ships (tasks) to utilize resources (energy globes) in a concurrent system and avoid starvation and deadlocks or other problems that can kill space ships without access to energy.

1.1 Goals and the importance of achieving them

In this assignment, I mainly designed a solution based on the ideas of Artificial Bee Colony (ABC) algorithm model and electronic cloud model. The goal of my model is to achieve the minimum stable number of vehicles (such as two vehicles or one vehicle that is off the network), and on this basis to achieve the maximum number of vehicles that can survive, just like the formula in mathematical induction, which is to start from the minimum and increase step by step. At the same time, as little as possible use deterministic operations (such as setting certain coordinates) to implement a universal concurrent model system. This is because, concurrent systems are mostly unsupervised (such as complex human societies), and a system in which all actions can change in real time based on the actual situation makes more sense.

Achieving a minimum amount of vehicle survival is fascinating and of enough practical significance. For instance, if the four or five soldier's reconnaissance team (vehicles) hope to find the unknown enemy (energy source), commander to specify their every step of the action will be difficult to effectively in the wrong direction, while a soldier (a vehicle) can be based on the current situation for the actual situation of real-time judgments are effective. In the future, with the development of robots, drones and so on, this less supervisory system will be more meaningful.

1.2 Resource Globes States Problems

There are two different resource globes states in this assignment. In Stage-b, the state of globe is a moving single globe. The energy globe is an energy point independent from other interference, with random stationary and the characteristics of the different mobile speed. In Stage-c, a random energy situation, each individual energy ball not only has the situation in stage b, but also has additional interference that will disappear randomly, which will lead to the death of the cars around the disappearing ball in a large area.

The main problems can happen are analyzed as follows:

1. High-speed mobile: The high-speed movement of the energy source will make it difficult for the slower vehicle to keep up with the resource point and lead to disconnection from the information connection network.

2. Charging limit: The number of resource points that can be charged per unit of time is still

limited even if they can be charged instantaneously, causing starvation. In the energy ball's setting, the sum of constant and Propulsion charge per millisecond is 5%, so it takes 20ms to charge each car, which means that 1s can charge up to 50 cars, the figure that is totally smaller in real because vehicles are mutually exclusive and at different speeds to go to charge.

3. Randomly disappear: The globe then disappears, making it difficult for vehicles to find new sources of energy in time.

4. Energy globe overlap: Overlapping energy points make it difficult to identify them and cause deadlocks. One deadlock may happen such as a vehicle find two balls in at same position, so it cannot decide to go to which one, so that have dead lock on how to go to charge.

1.2 Swarm Intelligence Problems

To let my model unsupervised, some swarm intelligence my model should have. The collaboration between vehicles in this assignment concurrent system should behave more like a population with swarm intelligence. If the vehicle group can make full use of space and time to adapt to the unknown energy points, it will be an excellent concurrent system. Therefore, every vehicle should not only have greedy individual behaviors, but also design and implement a macro strategy that reflects the wisdom of the group to ensure the continuation of the group.
Some possible strategies are summarized as follows:

1. Cooperative: A model based on game theory suggests that the larger the proportion of individuals with cooperative intentions, the greater the overall benefit, and therefore the number of selfish vehicles should be reduced.

2. Sacrifice: For individuals who find it difficult to benefit from the traffic, such as low-energy vehicles that clog up, selective death can make the population more likely to survive.

3. Swarm intelligence: To realize an unsupervised or semi-supervised model, the vehicle needs to be able to adjust the current strategy in real time according to the actual situation of the energy ball, instead of thinking that the supervised model should build a corresponding efficient model for every possible situation.

1.3 Small Group Problem

A detailed analysis of the possible problems will be necessary to enable as few vehicles as possible to survive in this vehicle concurrency system.
Some possible considerations are as follows:

1. The system needs to keep the vehicle as close to the energy source as possible when the number is small to maintain stability.

2. As the number of vehicles decreases, communication between vehicles will become more difficult, so a path to ensure that a small number of vehicles can still communicate needs to be designed.

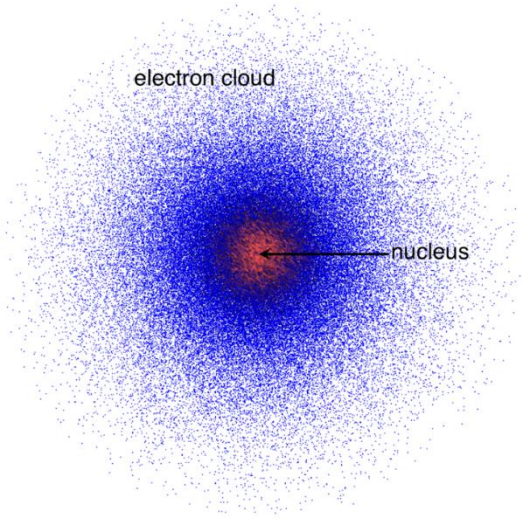
3. Because it will be more difficult to detect the energy ball when the number of cars is small (such as the random disappearance in stage c), a model that can detect as many places as possible in the case of concentration is needed

The above is the analysis of vehicle energy competition before model design, and then a universal unsupervised concurrent vehicle system will be tried to design based on the possible points obtained from the analysis. At the same time, because the above problem is also common in other concurrent systems, it should be paid attention to in real time at any time.

2 Model Design

2.1 Basic Idea Comes From

When I first read through this assignment about ship simulation, the first thing come to my mind is the electron clouds model. This is an energy point as the center, vehicles randomly appear in the sphere corresponding to their energy, the more the vehicles in the outer circle should have more energy, as the energy decreases, the vehicles will be in the inner layer. This is the model state in line with my idea. The sphere is the ideal shape that is most likely to be produced in the three-dimensional space under the condition of aggregation as much as possible. Meanwhile, the car can gradually approach the energy point with the energy. However, vehicles and electronics are different, the electron can be instantaneous movement and long span, but vehicles after getting a random location will soon get the next random location on ball layer, then the vehicle will appear in the case of situ trembling unable to move, because the vehicle cannot immediately go to a random location but new position is coming. Here is basic schematic of the electron model:



Reference from: https://www.sutori.com/item/electron-cloud-model-9092

Figure-1: Electron cloud model

However, the electron cloud model still has some advantages. In nature, small electron clouds can have hundreds or thousands of electrons, but also can keep only one electron, this is a very efficient method for spherical space. Therefore, if I can make the car in each energy layer move orderly instead of random motion, but at the same time, the position of the spherical motion can change according to the energy in real time, it will be a possible motion model for the car to make spherical motion according to its own radius.

Secondly, if I want my model to be as unsupervised as possible, I need to find a possible algorithm to implement it as a reference. Luckily, I found it.

One of the most famous and classic swarm intelligence models is Artificial Bee Colony (ABC) algorithm. [1] According to Karaboga and Basturk (2007), the smallest model of collective intelligence includes basic three components: a food source, employed foragers and unemployed foragers and two most basic behavioral models: Bees moving as a food source for recruitment or abandon a food source.

The model constructed by ABC algorithm is to solve the problem of finding food for a concurrent cluster in the real environment, which is very similar to the problem of finding energy sources for the carrier cluster in this assignment. Therefore, when optimizing the model, I chose to divide and implement the vehicle group based on ABC algorithm. The detailed design will be described later.

2.2 Basic Motion Model

To do my own expected model from my idea, I firstly design a basic model. My basic motion model is that the vehicle will make two circular motions around the detected energy point around the z-axis and the y-axis, first rotating around the z-axis and then rotating around the y-axis (Figure-2), The intersection of the two circles on the x-axis is the position of the vehicle in the two rotation paths.

The two intersecting points on the X-axis are very important when the number of vehicles is small, because this is a certain place where information can be exchanged between vehicles, thus solving the problem of information interaction when the number is small and maintaining the survival of the system.

For the rotation radius of the vehicle in the path, the rotation radius will change in real time with the energy, so that the energy source is closer in the smaller the energy, to maintain the aggregation of the vehicle.

This simple motion has allowed no more than ten vehicles to survive on a single ball. It can basically realize the stability of the system in a very small number of vehicles, which is a good start for me. So, after that, I can base on by this model to let it works on more complex situations and for more vehicles step by step.

Clear schematic of the model in Figure-2 is as follows:

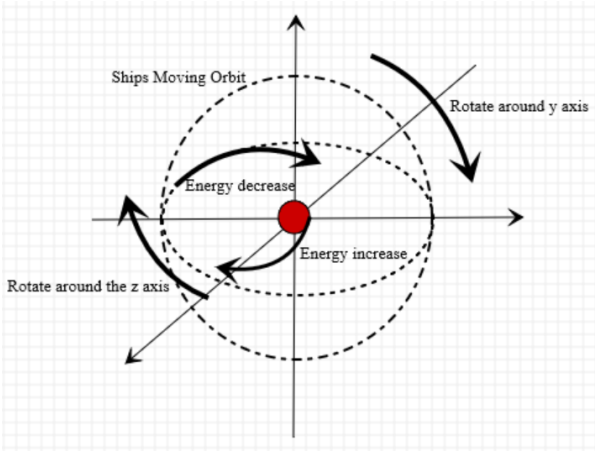


Figure-2: Basic Motion Model

To make full use of the spherical space, which will make the vehicle less congested for longer periods of time, and allow greater communication and detection area (from two circles to a whole sphere) when the vehicle is not more, the vehicle should be able to distribute on the spherical surface as evenly as possible when there are many vehicles and maintain the same motion attitude as the previous model to avoid conflicts (Red path in Figure-3). The red path in Figure-3 is let vehicles running same as the black circles, and the red paths are multiple and uniformly distributed on the sphere.

Clear schematic of the model in Figure-3 is as follows:

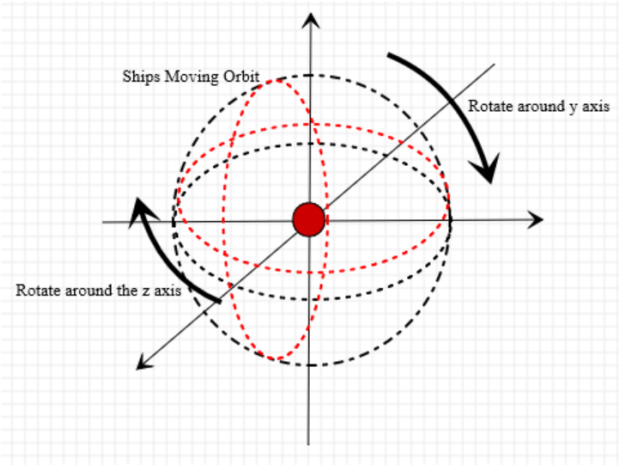


Figure-3: Spherical Trajectory Model

2.2 Basic Message Based Model

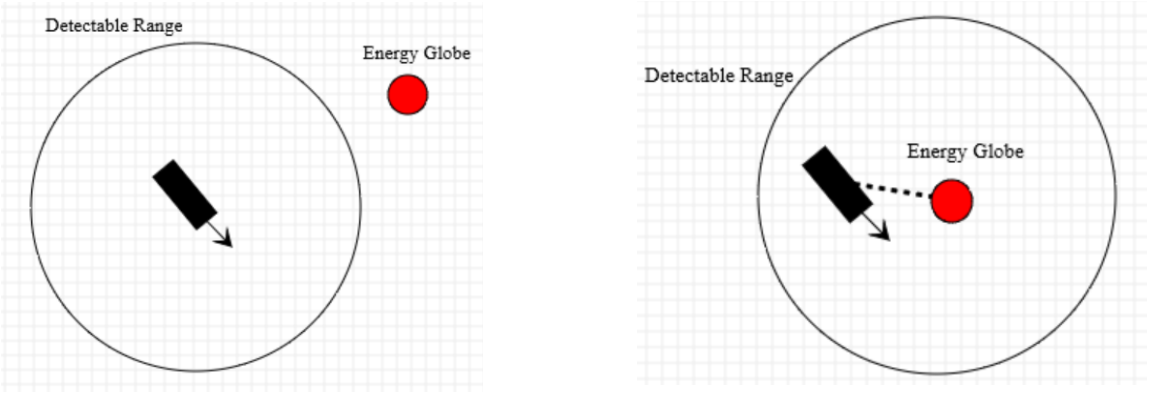
For stage b, they all message-based model, which means the vehicle needs to keep the system running by detecting the ball's position and passing information to other vehicles.

Firstly, searching for energy globes. Each vehicle has its own specific range of exploration. Only when the vehicle finds the power source and has other vehicles in the larger communication range can the information of the power source be transmitted. There should be two conditions for a vehicle:

1. In Figure-4-(a), it shows that without energy in the vehicle's range of exploration, the car will continue to wait for information.

2. In Figure-4-(b), in the detection area, the vehicle will transmit the detected information to other vehicles in time.

Here is the Figure-4 of how vehicle can find a energy Globe:



(a) The sphere of energy is out of range (b) The sphere of energy is detect able

Figure-4: Spherical Trajectory Model

2.4 Improvement (Stage-b and c)

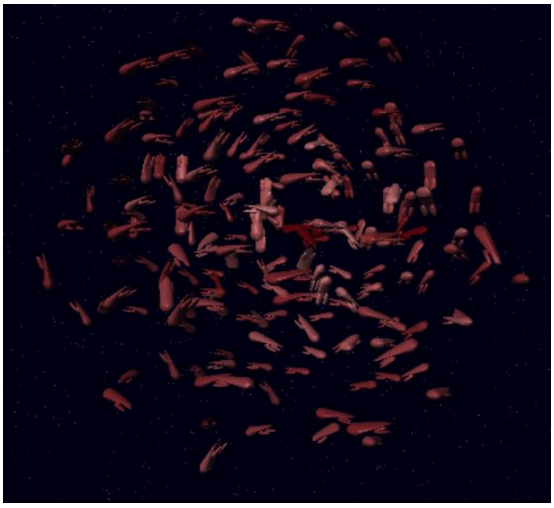


Figure-5: Final Model result

After finished the basis of the above design, I pursue minimum survival model can be completed in the case of a single energy ball good stage b and made at least two outstanding achievements in the vehicle to survive, and also can make the total after increasing vehicle maintain control in 110, proved the reliability of the foundation design and versatility, but for the complexities of stage c, using only basic design will make a system crash, need further improvement.

Firstly, I further improved my information acceptance and transmission mechanism. Each vehicle would first look for whether there was an energy source nearby. If there was more than one energy source in the detection range, I would assume that the car was greedy and would always choose the

current energy source coordinate nearest to itself and carry out transmission. If there is no power source around, the vehicle will only act as an information bridge, immediately transmitting the received information without any modification. Finally, the car will keep waiting for guidance to receive the information. After receiving the information, I assume that the car pursues the timeliness and effective line of the information. Only when the received message is new enough and the energy point coordinate is not too far away from the current energy point, will the information be updated.

As for the motion model, I started to design and optimize it based on the Artificial Bee Colony (ABC) algorithm, so that my model could achieve certain results in both small Numbers and large Numbers.

Karaboga and Basturk (2007) design three basic elements for a swarm system, and according to the actual situation in the assignment, the vehicle movement behavior was optimized as follows:

1. Energy globe selection and elimination: Vehicle in random globe system small Numbers of cases, the energy globe randomly disappear and the occurrence of overlap can be fatal, if a simple greedy system. For instance, all the vehicles only around a point of energy, this is likely to work in the vehicle number is large and stable (because of peripheral vehicle can detect the new), but the number is small will be difficult to achieve. So, I design a vehicle after receiving new energy points if the distance is within an acceptable range (for example, under the current energy radius), then will choose around new energy points, it makes the position of the system is able to independently according to the current energy block. At the same time, there are intersecting parts between each block to prevent accidents.

2. Leader: This type of vehicle will always be fixed around the power source, sending real-time updates to other vehicles around the power point, which will enable the system to detect at least one power point and thus remain stable.

3. Scouter: The number of this type of vehicle in my system uses the lower bound set in the algorithm paper, 5%. The scout type vehicles will be selected in my model to find out whether the known energy globe is still in place or slightly shifted regardless of whether the energy is dangerous. The return will bring the latest information to the system to keep the system stable.

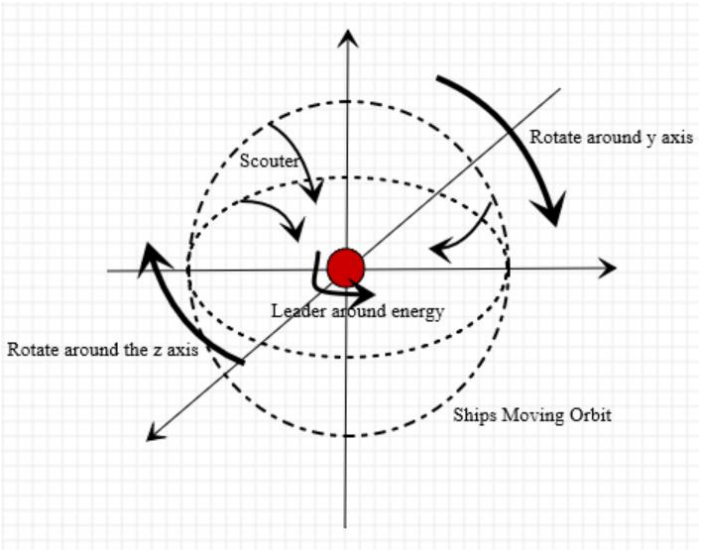


Figure-6: Leader and Scouter

Finally, regarding the optimization of swarm intelligence, I added sacrificial behavior, energy level grouping and adjusting the size of radius according to the current number.

1. Sacrificial behavior: A setting threshold for a vehicle that is about to die, passing information about its imminent death to other vehicles in time causes the radius of the group to decrease as the total number increases. Free to leave the current location at the same time to avoid blocking vehicles that may therefore survive

2. Energy level grouping: three levels (Low, Mid and High), Low speed vehicle can be a stable bridge of information transmission, which will let the current system not be interrupted the transmission of information. Mid-level energy cost vehicles are a repeater layer for too few vehicles at high or low energy levels, so that the communication of the system will not be interrupted. The high energy level vehicles will be charged at high frequency, which increases the possibility of acquiring effective new information and improves the system stability.

3. Radius controlling: To automatically adapt to the change from the minimum number of vehicles to many vehicles, with the increase of the total number of vehicles, my spherical model will expand a certain rotation radius to make the capacity of the space inside the ball larger, and vice versa. In this way, my model will be able to change from a minimal case to a more general one.

2.5 Stage-D Design

For stage-d, I will mainly face two major problems: how to accurately know that the vehicle needs to die, and how to control the accurate death of the vehicle.

For the first problem, I choose to use the “Vehicle_No” in the assignment. However, it is not very helpful because vehicles can only know their own ID, but if some other ID cars is not exit, just kill the ID larger than “Target_No_of_Elements” will not work. Therefore, each vehicle will want to know the survival status of all other vehicles as much as possible before starting to die so that they can make sure die or not. My method is to maintain a common dictionary structure and a private dictionary structure when communicating information. The number of each car will be the key of the dictionary, and the survival state is the key value. I assume that every car is greedy, so every time the vehicle receives a message, it updates itself to live status if the received dictionary does not have itself and tell others. After receiving the message, each vehicle will retrieve the survival status of all vehicles whose serial number is less than its own ($V_1 \ldots V_{n-1}$). If the number is found to exceed the limit, it means that there is no more possible to survive. The vehicle will update its status in the common table to death and go to death. No matter the ID bigger than its, because there is no more room.

Another question for stage-d I solved with three operation, first I set Throttle to 0.0, because if I hadn't done that, the cars going to die would have crashed into my model at high speed and destroyed the sphere structure, and a lot of cars would have died. Then use the “Flight_Termination.Stop” to let it not do more speed up. Finally exist the loop for this vehicle task, which means the vehicle task is end.

Here is a little demo for my algorithm working in Figure-7: Assuming have ID:1, 3, 5, 7, 9 (T-live, F-Die)

	Mess dics:	Dic-1:	Dic-3:	Dic-5:	Dic-9:
Beginning: Limit number=3		1-T	3-T	5-T	9-t
Some rounds after: Limit number=3	Mess dics: 1-T, 3-T, 5-T, 9-T	Dic-1: 1-T, 5-T	Dic-3: 1-T, 3-T	Dic-5: 5-T, 9-T	Dic-9: 1-T, 3-T, 5-T, 9-T
After 9 comes to die: Limit number=3	Mess dics: 1-T, 3-T 5-T, 9-F	Dic-1: 1-T, 3-T 5-T, 9-F	Dic-3: 1-T, 3-T 5-T, 9-F	Dic-5: 1-T, 3-T 5-T, 9-F	Dic-9:
After 5 comes to die: Limit number=3	Mess dics: 1-T, 3-T 5-F, 9-F	Dic-1: 1-T, 3-T 5-F, 9-F	Dic-3: 1-T, 3-T 5-F, 9-F	Dic-5:	Dic-9:

Figure-7: Sample Demo of Stage-D

Note: Message dics is an example of all dictionaries that are in communication, not a Shared memory part outside of message passing, because there is a delay in updating public information tables that are passed to and from each other.

3 Implementation

3.1 Message Passing and Message Processing

In Figure-8 as follows, the flow chart clearly shows the specific process of information transmission and reception. This part of the code also combines my implementation of stage-d.

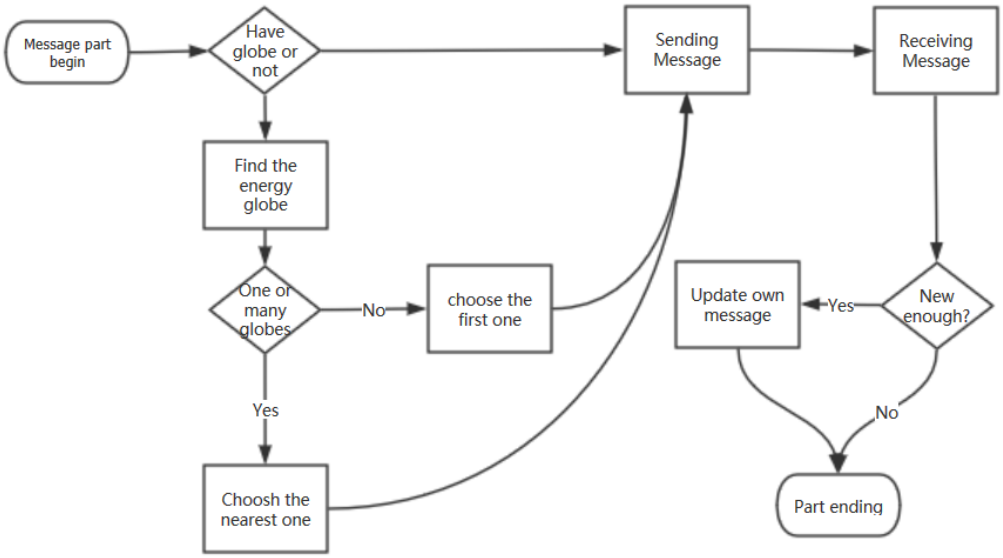


Figure-8: Flow chart of information transfer

As I mentioned in the design section, in message part, first for all vehicles is to find is there globes around and then receiving message and determine if an update is needed.

To implement my design in stage-d, I needed to find a way in Ada that was like the dictionary types in python. Thanks for my kind tutor Will discussed with us and told me that there is one thing called “Hash_Map”. At the first beginning, every vehicle will insert themselves in own map, every time sending message will insert themselves in the map in “Send_message”. When after update the own message, the vehicle will update its own map depend on the map “Live_Death_table” sending in the message. Finally, if it should go to die as I design, the function “Live_or_death” will tell it to die. Another little thing I did here is the sacrifice. I set the threshold of death energy is 0.025, because if less than this, the vehicle may not have enough time to tell others it will die soon. If more than that, too many vehicles will die that may live longer.

3.2 Charging and Moving

The charging part is a very simple rule, if the vehicle current charge is less than a half it will go to charge as fast as it can, if it is full charge, leave as quick as it can.

“My_energy” is a flag for the Scout_bee vehicles, Scout_bee will have My_energy as 0.1 so that it can go to find is the energy globes still there or not. If the vehicle is not a Scout bee, “My_energy” is equal to current charge and vehicle will moving as normal. Depended on it, if the total number of the vehicles is very small, the system can have more chance to know where the correct energy ball is. The way to design a vehicle is a Scout bee or not is by using Random number. Only generate numbers from 0 to 19 and if the random value is 1, it can be a Scout bee. So that when the vehicle number is large enough, there would be about 5% of vehicles be Scout bee, and when facing small groups, the probability will be low enough to avoid interference.

The “Orbital_radius” is the radius of the car during rotation or random free distribution. This radius is changing with the current charge in real time, so that the vehicle can have closer and closer path with the energy globe.

The X, Y, Z, are the spherical coordinate formed by the Angle generated by two independent random Numbers N1 and N2 of each vehicle. With the adoption of random Numbers, the random spherical coordinates obtained by the vehicle will tend to be uniform as the number increases, and because the final position is determined by the radius, the vehicle group will generate a multi-layer spherical shell state stratified by energy.

The speed matrix can divide the vehicle's energy consumption into three levels (Low-0.5, Mid-0.7 and High-0.9). Setting on these numbers for grouping is because if vehicles have less than 0.5 throttle cost to speed up, the vehicle will struggle to keep up with the fast-moving globe of energy. The energy output difference of each group is 0.2 can avoid the situation, which I think is the most difficult case that many vehicles need charging at the same time.

The Count is the is the counter of which point on the ball matrix the vehicle should go. Ball matrix has

16 different points that to let the vehicles can running as the two circles in Basic Motion Model. In the if condition of count, the throttle setting is depending on the current charge the vehicle have, lower energy means lower speed, which can let the vehicle have more probability to find a car and after 16 doing all around the 16 points, it will go to its randomly (X, Y, Z) on its own ball layer.

4 Analysis & Evaluation

When testing I found a phenomenon, that is my design based on the minimum number of surviving model when tested big data refresh the frame rate is very low, about 4 to 6 HZ, this means that the lower frames I got, the longer time I should test to get the assessment same as the general model of around 10 HZ, and not just use time as the only index calculation. The reason for this phenomenon is that almost all my parameters are calculated and changed in real time when the vehicle is running, which will take up a lot of system resources when the number is large. So, when I present the experimental data I'm going to show both the current frame rate and the test time.

4.1 Single Globe Tests

In total, I conducted three different tests, the first test to verify the primary objective of the model, the second test to verify the compatibility of the model with more vehicle states, and the third long time test to determine the stability of the model

Step 1. Very low vehicle number detection

Frame: 30HZ	5 min			
Test_No \ Begin number	5	4	3	2
1	4	4	3	2
2	5	3	3	2
3	5	4	3	2
4	5	4	3	2
5	4	4	3	2
Average	4.6	3.8	3	2

Table 4.1-1

Step 2. large vehicle number detection

Frame: >200 6HZ; <200 9HZ	5 min		
Test_No \ Begin number	300	250	150
1	240	137	123
2	237	128	127
3	288	133	119
4	229	131	118
5	239	120	103
Average	246.6	129.8	118

Table 4.1-2

Step 3. 1-hour detection

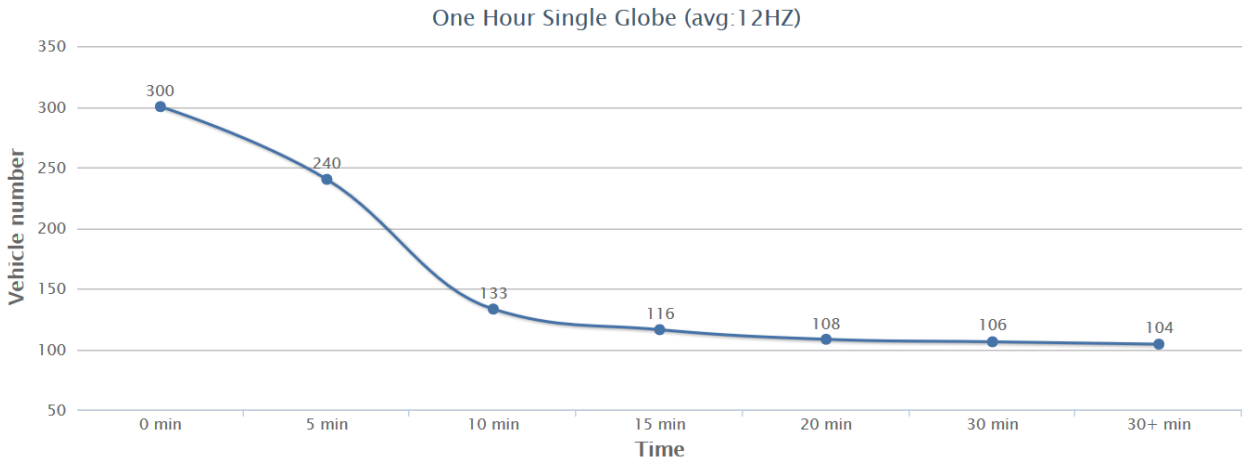


Table 4.1-3

Test Analise:

1. Table 4.1-1: This table shows one of the main challenges of the design, dealing with very few vehicles. Based on the experimental data obtained, it can be verified that the model can solve the problem well in the case of extremely low vehicle number, so that the vehicle can survive in a small number of cases.

2. Table 4.1-2: This chart details the problems my model faces with large numbers. It is obvious that the stability of the model tested from 300 is completely different from that tested from 250. After additional tests, the survival number above 230 can be maintained for 6-7min from 300. After the study, I found that my model would increase the radius of the ball more when the total number was greater than 250, so that the congestion around the central energy point would be smaller than when it started from 250, and the charge would be more fluid. This indicates that I need to make better adjustments to the radius for numbers up to 250. The test of base 150 basically tested the stable upper bound for a single globe for my model solving minimum survival at large numbers.

3. Table 4.1-3: After a long time of observation and test, I found that there was a cliff-like drop from 300 to 130 within 7 minutes, just like the previous analysis, and the survival number after that was basically stable. Therefore, it can be found that the minimum survival model I designed would have excellent performance for small numbers.

4.2 Random Globe Tests

Step 1. Very low vehicle number detection

Frame: 28HZ	10 min			
Test_No \ Begin number	5	4	3	2
1	5	4	3	2
2	5	4	3	2
3	4	4	3	2
4	5	4	3	2
5	5	4	3	2
Average	4.8	4	3	2

Table 4.2-1

Step 2. Large vehicle number detection

Frame: >200 5HZ; <200 8HZ	5 min	
Test_No \ Begin number	300	250
1	110	82
2	123	102
3	109	95
4	107	93
5	108	87
Average	111.4	91.8

Table 4.2-2

Step 3. 2-hour detection

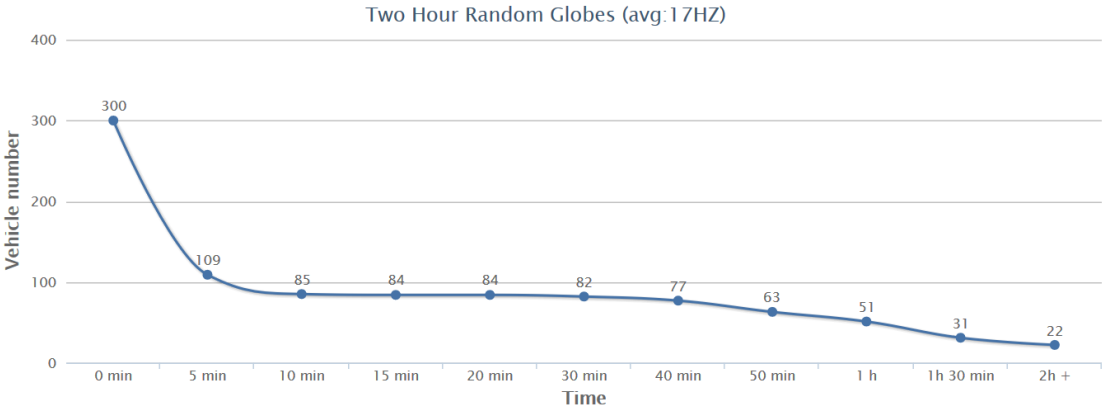


Table 4.2-3

Test Analise:

1. Table 4.2-1: I think I did a very good job on the minimize size cases of vehicles. For the case of a large base, the model I designed may cause a lot of deaths due to over-concentration and the existence of intersection points, but for the case of a small number, these are beneficial to the small size case.

2. Table 4.2-2: It can be seen from this table that the model I used to solve the minimum survival problem does not perform well in the random case of large numbers. This is already my implementation of the improved model I designed, and if I use the basic model I introduced at the beginning, this basically won't survive. After several tests and observations, it was found that without stage-d, before the total number was 140, my vehicle often lost contact, turned blue and died out of control. After testing, I found that there might be three reasons as follows:

1. The threshold of sacrifice I designed was still too high, and no vehicle knew whether a globe would appear in front of it in the next second under random conditions, so there were more active deaths.

2. The larger number and complexity of the situation resulted in a larger amount of calculation, resulting in the occurrence of a jam in the system and the active termination of some tasks.

3. Time-based update mechanism alone still cannot make the vehicle to charge without interference and thus wobble.

3. Table 4.2-3: I tested the random system for 2h to determine the upper limit of neat stability, and found that two of them maintained a relatively long survival period of 70-80 about 1h, and almost no death was caused by the random disappearance and overlap of globes within 30 after 2h.

4.3 Stage-D Test

All my tests on stage d were conducted under a single globe, because after previous tests, the random case was not stable in the case of large Numbers, which could not be well used for the detection of algorithm results

Frame: 20HZ	5 min		
Test_No \ Begin number	128->42	64 -> 42	5 -> 1
1	42	42	1
2	42	42	1
3	42	42	0
4	40	42	1
5	42	42	1
Average	41.6	42	0.8

Table 4.3

Test Analise:

Basically, the algorithm can solve stage d stably, no matter in the case of large number or small number. The not correct result happened in the Table 4.3, when doing 128 to 42, two vehicles surrounded by dead cars and crashed out of control range, when doing 5 to 1, unlucky same situation happened again and it is more dangerous for a little number case, because the message range is so small that no one can save a vehicle knocked out of communication range.

5 Problems and Conclusion

Throughout the actual implementation process of the assignment, my model still has some room for improvement:

1. Further improve the ABS algorithm model: The leader type vehicle I designed in this assignment has not been realized. After discussion with other students, I found that the implementation of such a flag car can greatly improve the implementation of the model for large Numbers.

2. More perfect communication mechanism: There is still hesitation caused by continuous updating of information in the model. If ABS algorithm can be implemented to evaluate the value of energy sources, I believe this situation will be significantly reduced.

3. Stage-D is not very perfect: Stage-d still has problems that are difficult to solve, such as the sudden death of vehicles with small ID, but the surrounding cars cannot find it in time, and the map transmitted by information is difficult to update. I have tried to add a table for each car to record whether the nearby cars exist or not, and a table to record the energy of each car around so that the dead cars always have the priority of dying with low energy and give the dead cars the chance to be reconnected to save, so that the trouble of accidental death can be solved well. However, due to the large amount of computation, this system has serious insufficient memory, which is obviously a design that is difficult to apply under my large computation model.

I learned a lot during the assignment. The first is that when faced with the smallest problem, a concurrent system will be faced with a state that is difficult to communicate and cooperate, just ordinary mutually exclusive operation is no longer enough, but a design that can constantly adjust the degree of control. For large numbers, I am most proud of the fact that I have successfully constructed stable layered spherical models using the mutual exclusion of small workshops, as Figure-5 shown before, It's also very much like an electron cloud. This was the first time I had ever felt that mutually exclusive tasks could display such a beautiful and exquisite state but not need to compute every possible point for it. Finally, what impressed me most was the terrible chaos in the concurrent system, which made me feel desperate in the previous weeks. For example, when trying to implement the leader vehicle, he was often scheduled to run by the system almost at the end, and no corresponding point could be found when he went to the energy point. Feeling the complexity and beautiful of the concurrent system was one of the most memorable things about this assignment.

Thanks to

1. Will gave very helpful Hash_Map tips for my Stage-D.
2. The unknown old student u5870997's github project give me the Idea about how to round a circle. Project is from: https://github.com/WrynnWang/COMP2310.
3. By talking with JinGuo Dong and Jiazong Gong, I learned many programming skills of Ada and the stage-D ideas.

Reference List

[1] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of global optimization*, *39*(3), 459-471.