Ecosphere

Team Leader: 杨浩斌 202030430240
Team Members: 赵旻昆 202030430394
陈冠宇 202030430042
张恭利 202030430271

Part1: Introduction and Objectives

The project we select is Ecosphere (Closed Ecosystem).

In our project, we are going to simulate a miniature world. In this miniature world, we are going to create various kinds of species something like those we see in a real ecosystem. And we are going to simulate the habits and characteristics of the creatures included in our miniature world.

Introduction

The ecosphere designed by us is a simple virtual ecosystem designed according to the structure and operation law of natural ecosystem. Among the living substances, we mainly introduce producers and consumers of different nutritional levels. Grass as the producer, horse, sheep and rabbit are the primary consumers, while tiger and wolf are the secondary consumers and also the highest nutritional level. At the same time, we are going to try our best to bring in inanimate substances from natural ecosystems, such as sunlight or rain, to change the living conditions of various creatures and make our ecosphere more realistic.

Objectives

Through the design of this virtual ecosphere, we aim to simulate a natural ecological system, to establish a long-term stable micro ecosystem, observe and record the living conditions of various biological creatures, learn to judge and determine the stability of an ecological system, figure out various factors which influence the stability of an ecosystem, and get the hang of the inseparable connection relationship between various species as well as that between species and the environment. In addition, through this virtual ecosphere, we are able to study what kind of living strategies a certain species should do in order to improve survival and research how to assign scientific human intervention into an ecosystem to make it more steady so as to achieve the purpose of scientific management. What’s more, by combining the ecological design and ecological planning, we can strengthen the management of ecosystem, maintain a healthy ecosystem and enhance the ecosystem service function. We hope that this designed virtual ecosphere can provide some inspiration and help for the research and management of natural ecosystem, and also promote the effective utilization and protection of natural ecosystem.

Part2: Requirements and Functionality

In the project, we are trying to use the functionality and features of data structure to simulate a real ecosystem. With the intention to correspond with our real life, we need to achieve some corresponding functions in our program, such as the design of aggregation, propagation, predation of the species, so that the consequence of our program is consistent with the characteristics of ecosystems in real world.

1. Tick

In our program, we need to set an appropriate time interval tick, and in each tick, the animals will determine which strategy they should take based on their current state to maximize the benefits for themselves and their species. In our assumption, the time interval is about 500ms. With 20 ticks accumulated, the age of the entity will increase by 1.

2. State

Each entity needs a variable as a symbol to represent its situation in the life cycle. In our assumption, this variable is state. State consists of three parts: age, energy and tiredness. In our design, there are three trophic levels in the whole ecosystem. Creatures at different trophic levels have their maximum age. Entities at second trophic level and third trophic level have another two properties energy and tiredness. The relationship between state and these three properties is:

$$State \begin{cases} e + 0.2a - 0.4t & age \le \frac{age_{max}}{2} \\ e - 0.2a - 0.4t & age > \frac{age_{max}}{2} \end{cases}$$

$$\boxed{\begin{array}{c} e: energy \\ a: age \\ t: tiredness \end{array}}$$

Figure 1

3. Observation:

In our assumption, every entity has its range vision. And only entities in this range will have an impact on its strategy. In each tick entity will send message to inform others to determine whether it is in the range vision of them.

4. Aggregation

In a real ecosystem, sometimes animals tend to take collective actions. Therefore, our program takes this into consideration. We are required to design a reasonable algorithm to analyze when and how entity will take actions in group.

5. Predation

Predation is the basic functions of animals we need to achieve. In our assumption, the main part is that we try to make it reasonable for an entity to prey on targets to ensure it can maintain its state while achieving sustainable development.

6. Propagation

In this part, we ignore the sex of animals and consider this problem from a holistic perspective. We assume that we can use some algorithms to divide entities of the same species into groups based on their location. In each group, the number of new entities is related to the original population.

7. Movement

In our setting, the speed of an entity remained unchanged during its movement in each tick. For each entity, the value of speed is related with the value of state, and the value of state will be consumed during movement. The detailed relationship is

$$e' = e - s \tag{1}$$

$$t' = t - v_0 + v^2 dT \tag{2}$$

$$v_{max} = v_0 + \alpha \times state \tag{3}$$

$e: energy$

$a: age$

$t: tiredness$

$s: distance$

$v: speed$

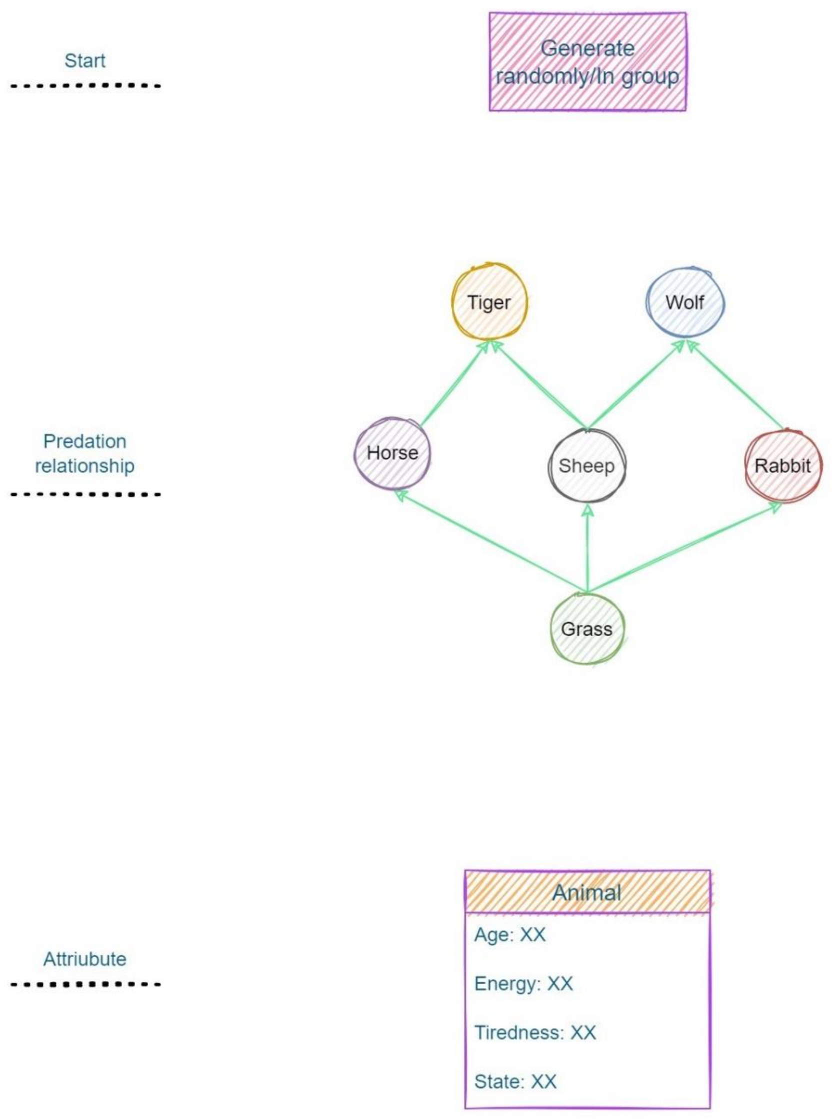
$v_0: critical\ value\ of\ velocity$

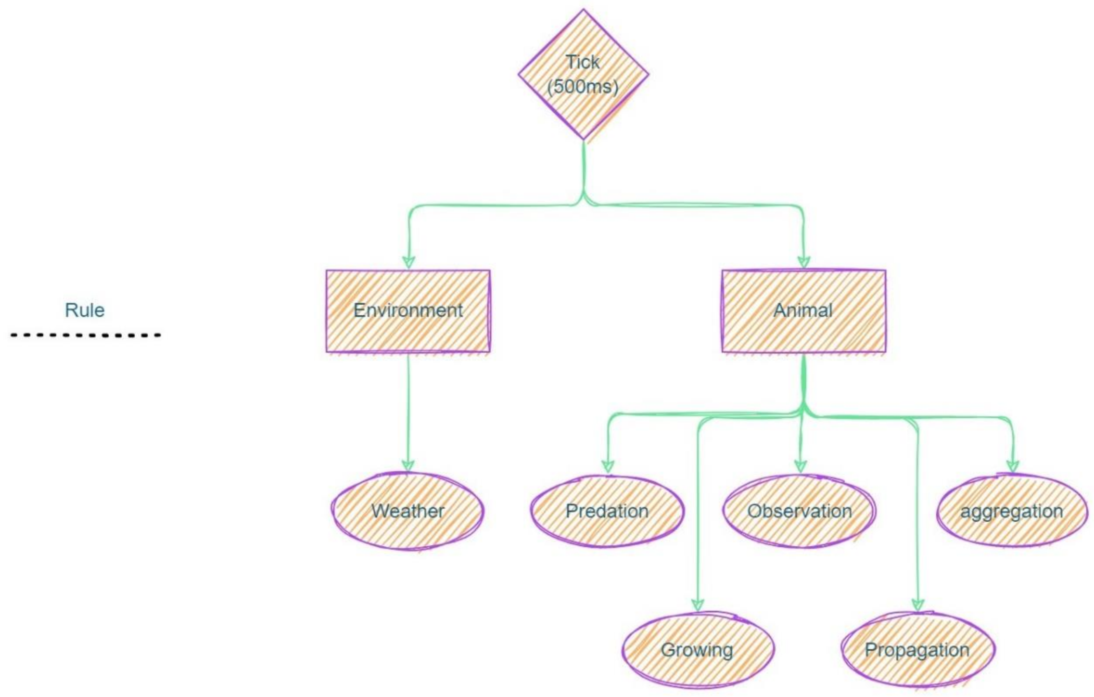
$v_{max}: the\ maximum\ value\ of\ veloctiy$

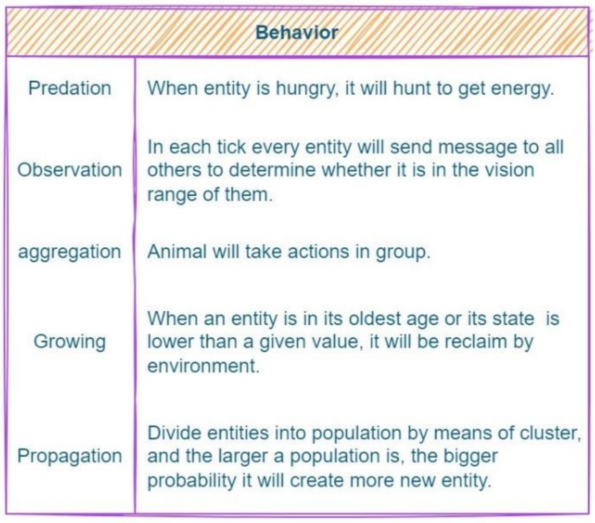
Figure 2

Part3: Logic Flow (Preliminary version)

The main part of our program includes the start part, predation relationship, attribute of animals and rule. The rule part will discuss more detailed later.


Start
Generate randomly/In group
Predation relationship
Tiger
Wolf
Horse
Sheep
Rabbit
Grass
Attriubute
Animal
Age: XX
Energy: XX
Tiredness: XX
State: XX





Behavior	
Predation	When entity is hungry, it will hunt to get energy.
Observation	In each tick every entity will send message to all others to determine whether it is in the vision range of them.
aggregation	Animal will take actions in group.
Growing	When an entity is in its oldest age or its state is lower than a given value, it will be reclaim by environment.
Propagation	Divide entities into population by means of cluster, and the larger a population is, the bigger probability it will create more new entity.

Figure 3

In the start part, program will generate the initial scene containing a large number of animals randomly or in group. The predation relationship will determine which kind of animal will prey on another. Attributes apply for all animal in the scene and some assumption has been mentioned in the second part of this report.

The rule part is the core of our program. Here we will use pseudocode to simply describe our design of the logic. and

In each tick, entity will take some action, including predation, aggregation and propagation. Some action will happen by default, including growing and observation.

- **Growing or aging**

As has been mentioned, animal will be growing and aging as time goes by. When an entity is in its oldest age, it will die. Besides, when the state of an entity is lower than a critical value, it will also die.

Pseudocode:

```
while this is in oldest age or the state of it is lower than a critical value
{
    reclaim this
}
```

- **Observation**

In each tick every entity will send message to all others to determine whether it is within the sight of them.

Pseudocode:

```
//emit part:
define TICK as 500ms
setTimeInterval(TICK, lambda(entity)
{
    emit message to all other animal
})
//receive part
define observeSet as null
function receiveMessage(entity)
{
    push entity into observeSet
}
```

- **Aggregation**

Animal will take actions in group. Each entity will follow the center of its population.

Pseudocode:

```
define populationSet as null
for each entity which is the same kind as this in observeSet
{
    push entity into populationSet
}
define centerPoint as the average coordinate point in populationSet
this follows centerPoint
```

- **Predation**

For animal in second tropical level, when it is hungry, it will eat the nearest grass.

Pseudocode:

```
define target as null
for each entity which is grass in observeSet
{
    if target is null
    {
        target = entity
    }
    else if entity is nearer than target
    {
        target = entity
    }
}
this run to target
```

For animal in second tropical level, when its energy is lower than one third of the threshold, it will run to the prey with the smallest distance in a straight line. When its energy is lower than the threshold and larger than on third of the threshold, it will run to the target with the largest value of a F function.

$$F(distance, state) = \frac{1}{m \times state + n \times distance^2} \tag{4}$$

where m and n are given values, which means that they will be adjust to some reasonable value.

Pseudocode:

```
define target as null
if energy is lower than threshold/3
{
    for each entity which is prey in observeSet
    {
        if entity is nearer than target
        {
            target = entity
        }
    }
}
else if energy is larger than threshold/3 and lower than threshold
{
    for each entity which is prey in observeSet
    {
        if entity's F value is larger than target's F value
        {
            target = entity
        }
    }
}
this run to target
```

- **Propagation**

Divide entities into population by means of cluster, and the larger a population is, the bigger probability it will create more new entity.

The method to divide entities of the same kind into population is:

Find all distance between any two entities, and classify them to the same class if the distance of them is smaller than a given value. And more detailed process will be discuss in the next part, that is, relation between data structure.

Then, how does entity choose action in each tick? Here shows the priority of the optional actions.

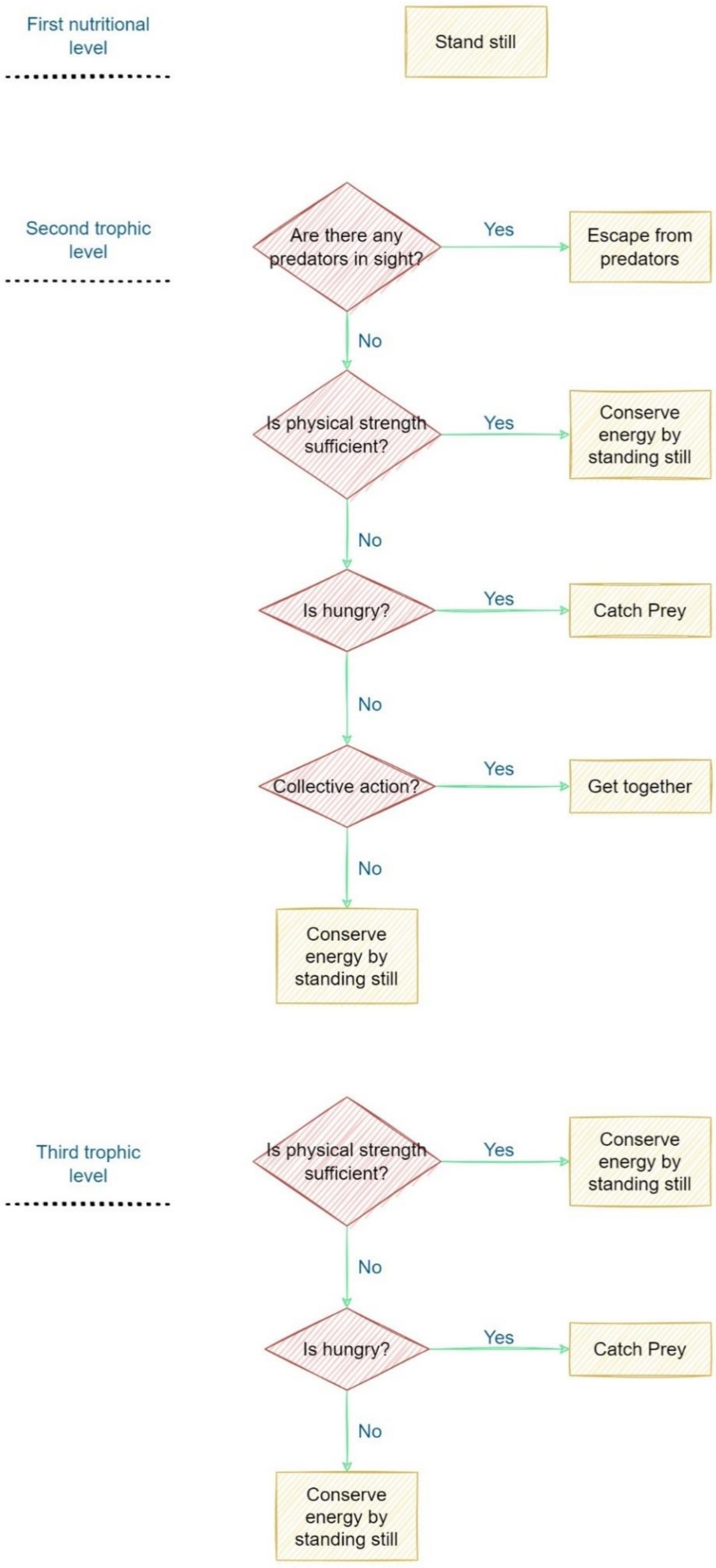

First nutritional level
Stand still
Second trophic level
Are there any predators in sight?
Yes
Escape from predators
No
Is physical strength sufficient?
Yes
Conserve energy by standing still
No
Is hungry?
Yes
Catch Prey
No
Collective action?
Yes
Get together
No
Conserve energy by standing still
Third trophic level
Is physical strength sufficient?
Yes
Conserve energy by standing still
No
Is hungry?
Yes
Catch Prey
No
Conserve energy by standing still

Figure 4

Part4: Relation between Data Structure

1. Breadth First Search

We use the technique of BFS in order to specify the shortest path between two individuals. And since our map takes landform into account and contains some obstacles, it is necessary to choose such an appropriate method to find the shortest path.

On the one hand, when an animal makes the decision of "Predation", it has to find the best path. In this case, it will not only keep away from its natural enemies and some obstacles on the line between its prey and itself, but also spend the least energy for predation. On the other hand, when an animal makes the decision of "Aggregation", it is obliged to find the best path as well.

Here are the reasons that we choose BFS rather than other searching algorithms. Firstly, the shortest path that needs to specify is only about one target destination. So, Floyd's Algorithm is unsuitable. Besides, the distance between two adjacent nodes in our grid map is a constant value. As a consequence, Dijkstra's Algorithm, which is used for irregular graph particularly, is also inappropriate. What's more, BFS performs better than DFS in our project as well. That is because, the number of obstacles in our map is so few that the depth of recursion levels will be extremely large and even exceed the max recursion depth if we use DFS.

2. Min-Heap

We use Min-Heap when dealing with "Aggregation", which is something like Greedy Algorithm.

As we have mentioned in the logic flow, some species especially herbivores are going to take actions in group every several ticks. In this case, we select some individuals as the center of a group randomly at certain ticks. And all the livings of the same species that are near to those center ones will gather towards the center.

In order to realize such mechanism, we put all the distances between the animals chosen as a group's center and the other living of the same species into a Min-Heap. Then, we set a threshold value of distance and fetch the values from the Min-

Heap one by one. And we stop when the root's value in the Min-Heap is larger than the threshold value. In this way, we correctly select the nearest several livings to form a group.

3. Equivalence Class

The Equivalence Class makes sense mainly in the propagation stages. As we have mentioned before, the species will propagate when they form several clusters and the probability of propagation depends on the number of species in the group and the distribution of their states. That is to say, propagation only happens inside one cluster and different clusters do not influence each other.

So, we define that each cluster belongs to an Equivalence Class. And the root of Equivalence Class is the individual selected as a group's center. In addition, after each time one individual is chosen as member of the cluster (each time the Min-Heap's root is removed), it points to the center living directly. In this case, we let each cluster belong to one faction and realize the classification problem, which is the premise of "Propagation".

Part5: Plan and Schedule

Plan

Period	**Planned Mission**
before 2022.2.28	Construct the frame of the whole project. (requisite classes; the inheritance relationship of the classes; requisite data members and functions……)
2022.3.1-2022.3.20	Finish the implementation of the classes and main functions.
2022.3.21-2022.3.31	Finish the design of the GUI and build the relation between the interface and the kernel.
2022.4.1-2022.4.15	Polish the functions in the GUI, improve the correctness and readability of our code and adjust the parameters by

	testing again and again.
2022.4.16-deadline	Debug, finish the final report and prepare for the presentation.

Table 1