G53DIA COURSWORK2 REPORT

Jianheng Qiu

4276906

psyjq@nottingham.ac.uk

Task Environment:

1. Task

The basic target is to gain as much as possible scores (total waste collection) within fixed time (10000 time steps) with multiple agents (at least two tankers). For each agent, there is four general goals that need to be considered.

- First is exploration, the tanker needs to a station with task, if there is no current work existing.
- Second is collection, if a task exists, the tanker needs to move to that position to collect waste.
- Third is dispose, if the tanker has carried on waste, it needs to find a well to dispose.
- Fourth is refuel, the tanker has to evaluate that whether the fuel is sufficient to reach the target position, if not, then goes to refuel.

Obviously, for each tanker, there exists achievable goals (exploration, collection, dispose) and maintenance goals (refuel). Except refuel, all other goals are weakly committed to tanker because the tanker should make sure it would not stop before the time is ran out. Each tanker also contains two parameter, waste level and fuel level and both have a max value of 100. The score is the number of waste that has been disposed to the well and the final score is the sum of each tank's individual score. Fuel is charged by movements and movements can be fallible. Besides, all the tankers will be designed as totipotent and they cooperate with each other to finish task together. These features imply that there will be different solutions in different time. The constraints for the target are fixed time steps and the two parameters.

2. percepts

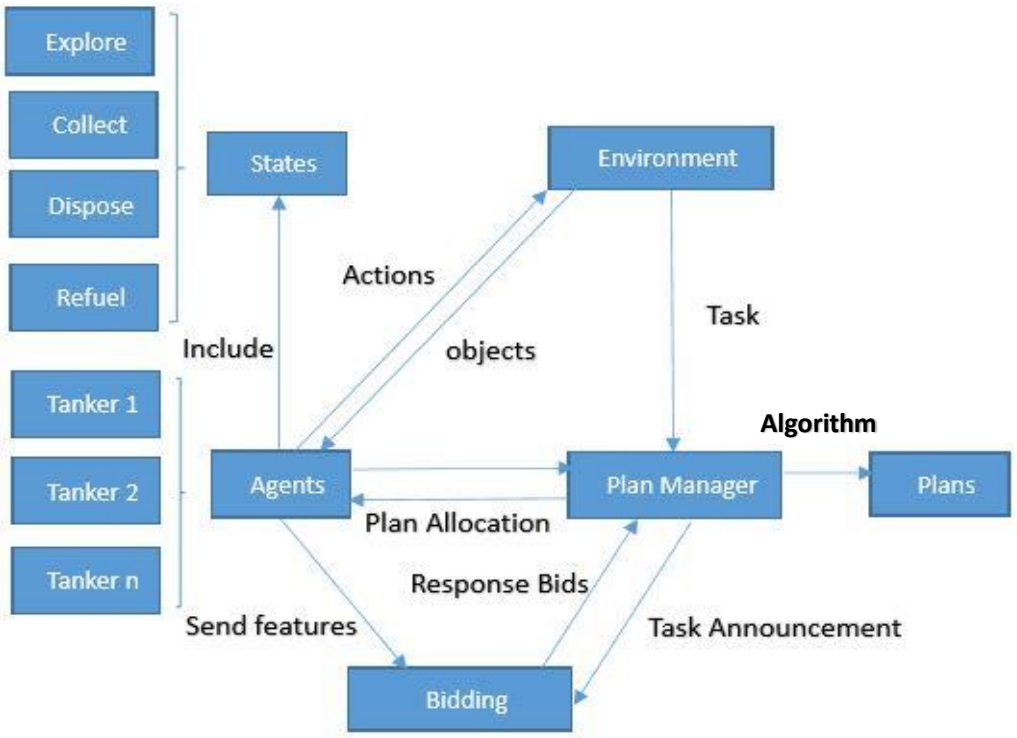
This is a **discrete** environment because of the limitation of object quantities and time steps, and the distinct distribution of objects. This is a **multiple-agents** system, there will be at least two tankers in the environment. All the agents are **totipotent**. Besides, the environment is **partially observable** for tankers, for each tanker, it has a view of 41 * 41. Only the objects shown in the view can be observed by tanker. In addition, the environment is **dynamic**, the objects are generated randomly and the tasks are activated randomly, too. Besides, the tankers should cooperate with each other. These features also imply that the environment is **nondeterministic.**

3. Actions

For each tanker, four **fallible** actions: Moving, Loading, Disposing and Refueling will be implemented. Moving actions help tankers to move between departures and targets. Loading actions are used to load waste at station with task while disposing actions are used to dispose waste at well. And the refueling actions are used to refuel at pump station. Each movements will cost one time step and except refuel, other movements will cost one fuel.

Design:

1. High Level:



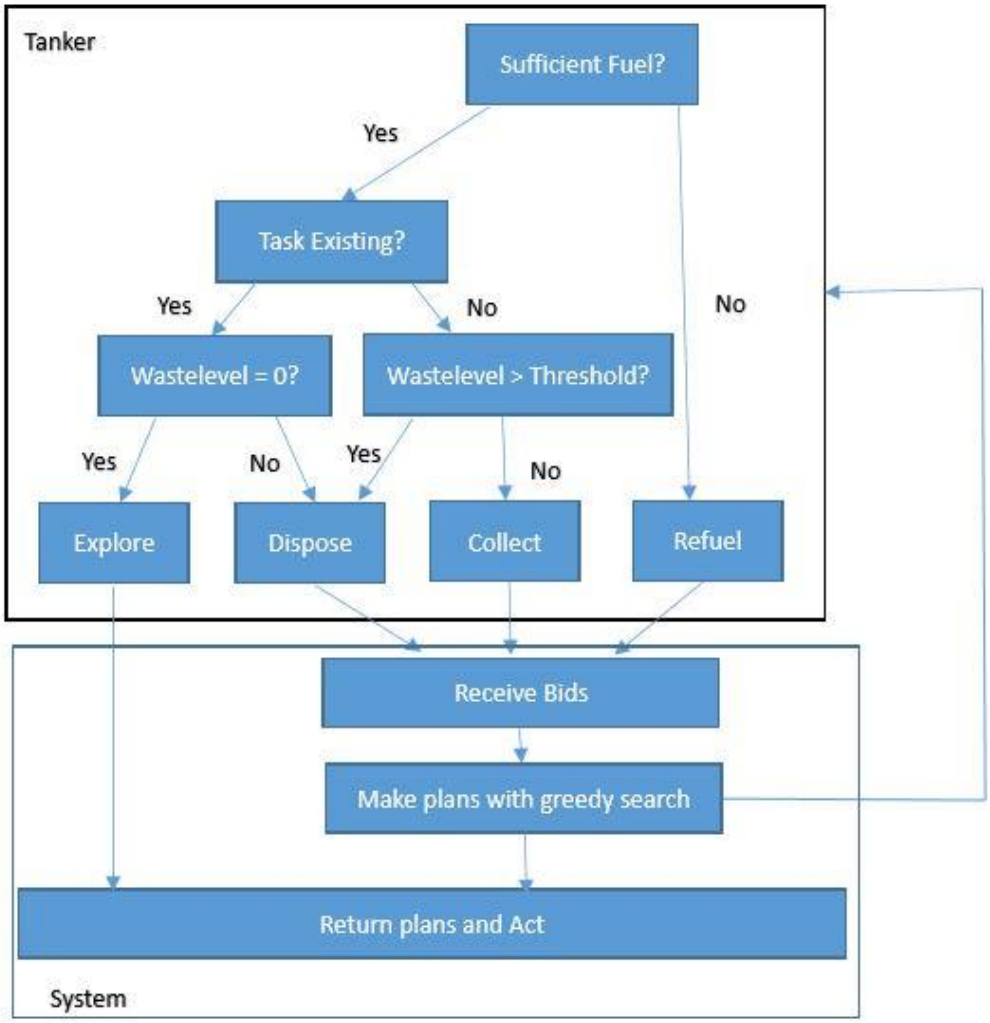
From the figure of structure, the whole system is designed as **deliberative**, **egalitarian**, **predefined** organizational structure. There are multiple agents in the environment and each agent is totipotent. For each agent, it will have four states and a 41 *41 range view. The tanker will draw a virtual map during its movement, in other words, it will store the objects that have been observed and remember the virtual positions. The plan manager will arranges routes for each agents. First, the manager sends **task announcements** to all the tankers, the announcements are all the tasks that tankers had observed and are stored in Memory Map. Second, the tankers **response their bids** to manager to concern. There can be several bids and each of them will include position, target, state and calculated cost. Then, the manager using algorithms to evaluate tankers' bids cording to the cost and find optimal one to **allocate to** its tanker. The cost in this system is calculated by the sum of spent fuel and distance. Finally, manager will **output plans** to guide the agents to move.

2. Low Level:

2.1. **For system:**

Every tanker in the environment will be assigned an ID to distinguish from each other. As mentioned, any tanker will save the observed objects and draw a virtual map for further use. The map will be kept updating during the moving of tankers. Then those maps described by each tanker will be combined together for plan manager. Then the manager rely on the whole map to make decisions for tankers. This is described as task announcements in high level part. The tankers will also pass their own features to the bidding function. Then the function keeps estimating possible bids for each tanker. As exploration step is an individual movement, the bids can only arrange other

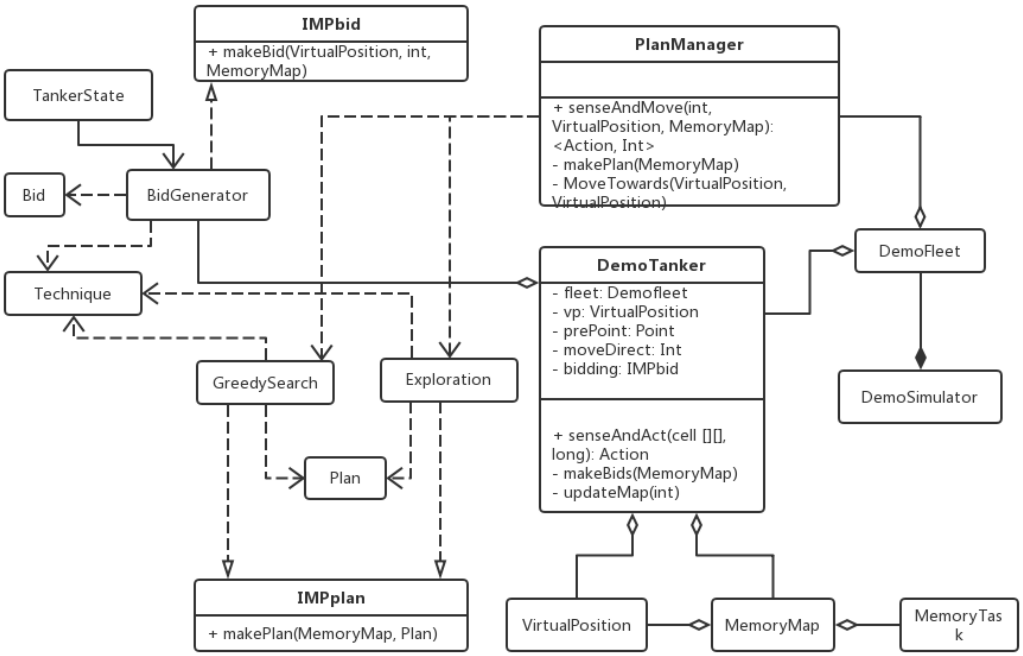
movements (collect, dispose and refuel). Every bids will contain task position and costs to achieve this bid. After that, all possible bids will be sent to manager and manager will choose the optimal one for the each tanker to execute according to greedy search. In other words, manager generates the plans for each tanker. The tanker with plans will try to move towards target and finish the goal with specific activations (load, dispose or refuel).
The general logic expression:



2.2. **For tanker:**
The basic idea is to collect waste as much as possible if the fuel level allows. There is a waste level threshold, if the waste level is not larger than this threshold, the tanker can stay in collection state. If the waste carried on is more than this threshold, the tanker will change to Dispose state. Besides, it is obvious that the refueling activity will decrease the system's efficiency, so fuel pump will not be recommanded to tanker until there is a requirement for refueling.
When the tanker is moving, tanker will generate bids according to current environmrnt. The list of bids include the tanker sending this bid, the target, the cost of finishing the target and the way to get to the target. Tanker can send several bids before it takes next step, then the plan manager will evaluate these bids and choose the optimal one (lower cost) as the moving plan and return such plan to tanker. Then the tanker will follow the lead of the plan to move.

In state exploration, tanker will face the situation that no task station in the view or memory, so the tanker will try to find one. In this project, tanker will follow a certain route to explore the map.
In State Collection, the tanker tries to move to a task station to load waste. If the station is not reachable, for example, after the collection of waste, the tanker does not have enough fuel to move to the nearest fuel pump. Then such task station is not avaliable for tanker, tanker will never choose such position as a target. In other words, tanker will not consider this station into a bid. If the task station is reachable, the tanker will add the virtual position of station, the Collection state declaration and the estimated cost into the list of a bid.
In State Dispose, the tanker will move to a well to dispose waste. The generalsteps are same as state collection, except the target is changed to well.
In State Refuel, the tanker will move to pump station. If a previous target (collect or dispose) is finished and tanker still gets enough fuel to move to the nearest fuel pump to the target's position, it will add this fuel pump as a next target and the Refuel state declaration into the list of bid. Otherwise, it will ignore the target (station or well) and find the nearest fuel pump to the current position, then add this fuel pump as a new target, and add it with Refuel state into the bids list.

Implementation:



The general class relation is shown above.

1. Memory:

The function of memory is to store and classify the objects that tanker has observed in its view. The class VitrtualPosition is introduced to this system, Such position (x, y) is fixed and accessible even if the object is not in view any more. While the tanker is moving, virtual position is being updated according to the direction of tanker's movement. This is accomplished by updateMap() method in DemoTanker. The tanker will remember its last moving direction (movingDirect) and update the virtual position of current location according to this direction. Then the tanker can remember these objects with virtual position. The class for saving these position is called

MemoryMap. HashMap<VirtualPosition, Point> will be used to store the object. The key of hash map is a virtual position of object, and each key contains which type of the object. Every time when the view of tanker is updated, the tanker will search the whole view to save necessary objects, this is called drawmap in the program. Before storing, tanker will check whether the object has been saved before.

2. Bidding and plan

When tanker is moving, the BidGenerator class will gather information from both tanker and environment. The method makeBid in BidGenerator will take a task's virtual position, tanker's ID and the virtual map into consideration. The state of the tanker is initialized as Collection. Then the class goes through a loop until the state is changed to exploration. In the loop, the tanker first checks whether the fuel is sufficient, a tempPump virtual position variable is introduced here, when the tanker needs to refuel, the tempPump will be assigned with a nearest fuel pump. Then if the task has remaining waste and the tanker's waste level is not zero, then tanker changes to Dispose. Besides, if task is finished and tanker's waste level is smaller than the waste threshold, it keeps state of Collection, otherwise, the state is changed to Dispose.

In the state of Collection and Dispose, if the target (task or well) is reachable and this is the first reachable target for tanker, since in the loop the tanker may have several targets, this first target's virtual position and the current state should be recorded in the Bid (saved as firProperTar and firProperState). In the state of Refuel, the tanker first check whether the fuel is full or there is no proper fuel pump to go. If it is yes, the class will return invalid bid which will be vanished in other place. If not and the fuel pump is the first target, then the virtual position and current state Refuel should be saved in bId. In the end of the BidGenerator Class, the class will return the bid including the task virtual position, the cost, the first target and the corresponding state. The cost is calculated by the sum of total traveling distance during the loop and the total number of spending fuel.

When the system is running, the DemoFleet will ask DemoTanker to generate bids and ask the PlanManager to manage these bids. The PlanManager will ask Greedysearch to select the optimal bid with lowest cost and ask Exploration for tanker to explore. In GreedySearch, the makePlan method will vanish the invalid bids first. Then among the valid bids, the method will compare their costs with each other and choose the lowest one. Then the bid will be the moving plan including target and state. After the selection of specific plan, the tanker will try to move to the target, if the tanker has arrived at target, for Collection, it will load waste, for Dispose, it will dispose waste and for Refuel, it will get fuel.

Evaluation:

	Number of tankers	Total scores
1	2	182053
2	2	161913
3	3	158613
4	3	159052
5	4	133439
6	4	127867
7	5	127420
8	5	131491
9	6	134346

10	6	137714
Average:		145391

Conclusion:

In summary, the general performance of the system is good for multiple agents. That means, the central manager could select the optimal bids and the generated plans are optimal routes for each tanker. The tankers are all totipotent and they cooperate with each other properly under the control of manager. The manager will gather information from the whole environment, the memory and the features of tankers. In terms of the result, it is shown that when the number of tanker increases, the whole performance performs downtrend. That may because the quantity of tasks is not sufficient, in other words, tanker cannot get enough tasks and sometimes tankers are competing with each other. Such situation can influence the performance of the tankers.