Comprehensive Performance Report for Drone Communication Network Project

The drone communication network project simulates a fleet of drones organized into two distinct communication partitions. Each partition serves as a linked list network in which the drones communicate, perform search operations, and update their status based on nearby targets or hazards. The project not only tests the communication and coordination between drones but also measures the computational performance of common operations—deletion, search, and handling non-existent drone IDs.

The two communication partitions represent distinct functional groups within the network, with each partition assigned a unique role and a color for visual differentiation. This setup allows for efficient coordination as each partition's drones execute their specific tasks.

Partitioning of the Drone Communication Network

In this project, partitioning is employed to divide the drones into two groups, each responsible for a specific task:

1. **Partition 1 - Victim Search**: Drones in this partition are programmed to locate victim locations. These drones begin the simulation in **blue** to distinguish them from other drones and change to **red** upon discovering a victim. This color change signals that the drone has completed its task and remains at the victim location.
2. **Partition 2 - Hazard Search**: Drones in this partition are assigned to identify and locate hazard sites. These drones start in **green** and switch to **yellow** upon locating a hazard, indicating that they have found a hazard and will remain there as a marker.

This partitioning enables each drone group to specialize in a particular search task, enhancing the efficiency and organization of the network by avoiding overlap between drones looking for different targets.

How Partitioning Impacts Communication

Partitioning the drones into two linked list communication networks allows each group to operate semi-independently while still being part of an overarching communication system. The partition-specific roles facilitate task assignment and message propagation across groups, enabling drones within each partition to communicate status updates, such as the successful discovery of a victim or hazard.

Partition-specific message passing is managed through instances of a DroneCommunication class, which uses linked lists to hold each drone's ID and propagates messages through the list as needed. Each partition's drones can send and relay messages through this list structure, allowing for systematic and organized communication.

Drone Actions and Responsibilities

Each drone is assigned a specific task depending on its partition and operates autonomously to search for its target. Here's an overview of the main actions performed by drones in each partition:

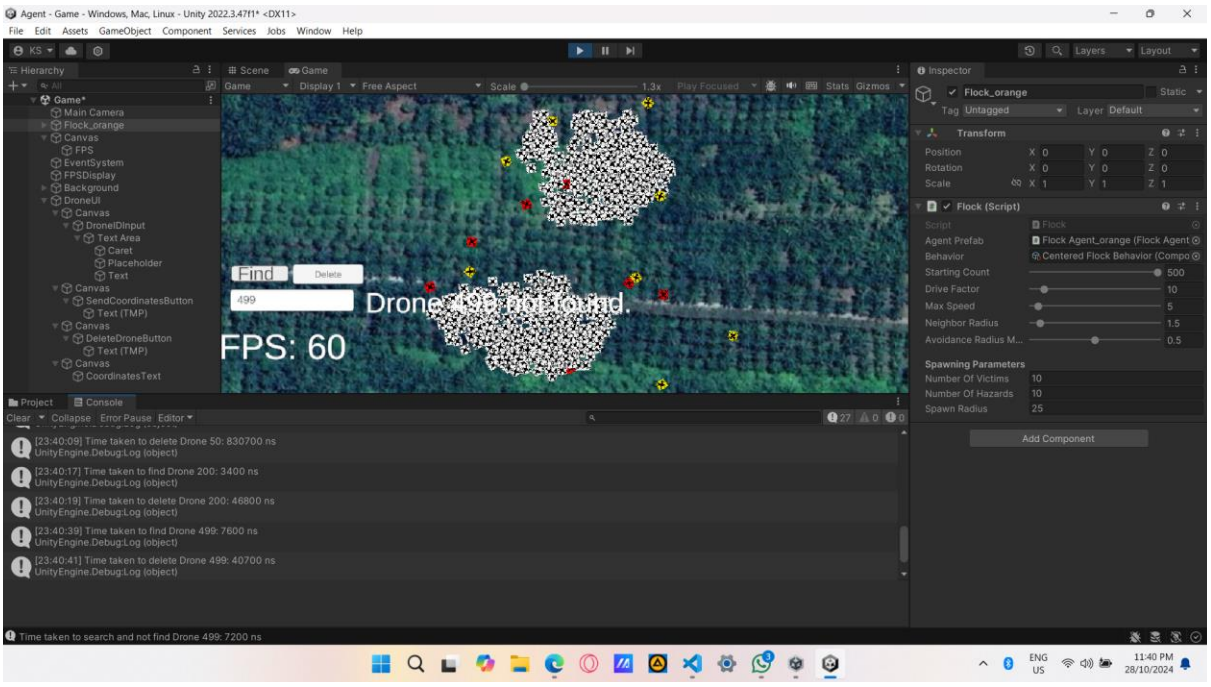
1. **Searching for Targets**:
    - Drones traverse their environment, moving systematically within a specified range to locate either victims (Partition 1) or hazards (Partition 2).
    - During each update cycle, drones check for targets within a certain detection radius. If they come within this range of a victim or hazard, they log the location and update their color to indicate successful discovery.
2. **Target Found Behavior**:
    - Once a target is located, the drone’s color changes to signal completion of its task—red for Partition 1 (victims) and yellow for Partition 2 (hazards).
    - The drone then ceases further movement and remains at the location, marking it as a designated zone, which could be important in a rescue or hazard-marking scenario.
3. **Message Propagation**:
    - Each partition uses a linked list to manage drone IDs, allowing for message propagation from one drone to the next within each partition. This structure supports scenarios where updates or status alerts need to be shared across the entire partition.

Operation Analysis

The specific operations measured in this performance test include:

1. **Delete Drone**: Removes a specified drone (based on its unique ID) from the communication list. The purpose is to evaluate the time required to search through and modify the linked list by removing an element.
2. **Find Drone**: Searches for a specified drone within the linked list to test retrieval performance and communication efficiency.
3. **Search Not Found Drone**: Attempts to locate a non-existing drone in the network, illustrating the time taken to traverse and confirm the absence of a specific drone ID within the list.

Each of these operations is performed on specific drones or IDs, and the corresponding times are recorded to provide an in-depth understanding of the system's performance across various scenarios.



```
[23:40:09] Time taken to delete Drone 50: 830700 ns
UnityEngine.Debug:Log (object)
[23:40:17] Time taken to find Drone 200: 3400 ns
UnityEngine.Debug:Log (object)
[23:40:19] Time taken to delete Drone 200: 46800 ns
UnityEngine.Debug:Log (object)
[23:40:39] Time taken to find Drone 499: 7600 ns
UnityEngine.Debug:Log (object)
[23:40:41] Time taken to delete Drone 499: 40700 ns
UnityEngine.Debug:Log (object)

Time taken to search and not find Drone 499: 7200 ns
```

```
Flock_orange
Tag Untagged    Layer Default
Transform
Position   X 0  Y 0  Z 0
Rotation   X 0  Y 0  Z 0
Scale      X 1  Y 1  Z 1
Flock (Script)
Script              Flock
Agent Prefab        Flock Agent_orange (Flock Agent
Behavior            Centered Flock Behavior (Compo
Starting Count      500
Drive Factor        10
Max Speed           5
Neighbor Radius     1.5
Avoidance Radius M... 0.5
Spawning Parameters
Number Of Victims   10
Number Of Hazards   10
Spawn Radius        25
```

Key Data Points and Observations

Below are some highlighted data points from the operations performed on the drone communication network:

- **Delete Drone (ID 50)**: Took **830,700 ns**, indicating a significant time investment to locate and delete a drone in the early portion of the list.
- **Find Drone (ID 200)**: Required only **3,400 ns**, showcasing efficient retrieval time when a drone ID is present within the list.
- **Delete Drone (ID 200)**: Took **46,800 ns**, illustrating that deletion times vary based on the drone's position in the list. This is significantly faster than the deletion of drone ID 50, likely due to differences in traversal requirements.
- **Find Drone (ID 499)**: Required **7,600 ns**, again showing a modest retrieval time for a large ID near the list's end, indicating that search efficiency is generally consistent.
- **Search Not Found Drone (ID 499)**: Took **7,200 ns**, which aligns closely with the previous find time, but reflects the additional time required to confirm absence as the search traverses the entire list.

Detailed Performance Analysis

1. **Deletion Performance**:
    - Deletion operations in a linked list are typically more time-consuming than search operations. In this project, deletion requires both a search to locate the specific drone ID and an additional step to remove the node from the list. The recorded times show that deleting drones positioned earlier in the list can be significantly more costly, likely due to traversal and pointer management.

    - For example, deleting Drone ID 50 took **830,700 ns**, while deleting Drone ID 200 took only **46,800 ns**. This suggests that deletion time can vary significantly with the drone's position, particularly if the list contains numerous drones.

2. **Search Efficiency**:
    - The "Find Drone" operations indicate that the linked list structure allows for relatively quick retrieval times. With linear traversal, search times increase in proportion to the distance of the target from the head of the list.
    - For instance, finding Drone ID 200 required only **3,400 ns**, a reflection of the linear traversal cost, whereas searching for a non-existent ID (499) took **7,200 ns**. This non-existent search time is slightly longer but suggests that unsuccessful searches do not excessively strain performance.

3. **Search for Non-Existent Elements**:
    - Unsuccessful searches were recorded, such as the search for Drone ID 499, which took **7,200 ns**. This is important because it demonstrates that even when the desired drone ID does not exist, the system requires time to complete a traversal and confirm absence.
    - Although the linked list approach ensures consistency in handling non-existent items, the linear nature means search times will increase with list size.

Complexity Analysis and Performance Implications

The results observed in this performance test align closely with the expected time complexity characteristics of linked list data structures:

- **Time Complexity of Find and Delete Operations**:
    - Both the "Find Drone" and "Delete Drone" operations display linear time complexity (O(n)), meaning time grows linearly with the number of nodes or drones in the list.
    - Search efficiency for existing items remains relatively high, but deletion suffers due to the dual need for traversal and pointer adjustments in a single-linked list.

Recommendations for Optimization

Given these findings, the following recommendations could potentially enhance the drone communication network's performance:

1. **Alternative Data Structures for Faster Lookups**:
    - Implementing auxiliary data structures, such as hash tables or dictionaries, could complement the linked list by providing an efficient way to index drone IDs. This hybrid approach might enable near-constant time (O(1)) lookups while still maintaining the sequential organization for communication purposes.

2. **Improved Partition Management**:

    - Experimenting with different partition sizes or utilizing multiple smaller lists might help distribute the workload more evenly, reducing the average traversal length required for each operation.

3. **Consider Doubly Linked Lists for Enhanced Deletion**:
    - Since deletions currently necessitate pointer adjustments, switching to a doubly linked list structure might improve performance by allowing easier navigation and node removal, especially for drones located toward the end of the list.

Summary

This performance report has evaluated the efficiency and time complexity of core operations within a linked list-based drone communication network. The project demonstrates the feasibility of using linked lists for drone ID management but highlights some limitations in terms of deletion performance and non-existent searches.

By addressing these limitations through recommended optimizations, such as auxiliary indexing structures or a hybrid linked list/hash table approach, the system could achieve faster and more scalable communication network management, ensuring a more efficient environment for drone coordination.