UNIVERSITI TEKNOLOGI PETRONAS

TEB1113/TFB2023:Algorithm and Data Structure

September 2024

FINAL REPORT

Group:

1. See Tho Soon Yinn (24000187)
2. Avinash Kumar A/L Jayaseelan (24000113)
3. Ashwin A/L Ravichandran (22012188)
4. Syukri Fadhli bin Ahmad (24000074)
5. Hamzah Muhsin Bin Hafiz Al-Asadi (22001057)
6. Aiman Naim bin Shaiful Zahrin (22005653)

Performance Analysis in Drone Management System Comparing Data Structures: Linked List, Binary Tree, and Graph + Djikstra

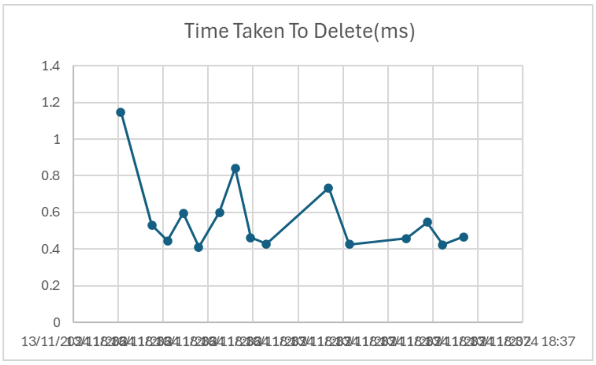
---

Introduction

Efficient drone communication and management are critical for real-time applications. In this project, a Unity-based system simulates a drone management network using three different data structures: linked list, binary tree, and graph. The primary goal is to evaluate the performance of these structures in terms of drone deletion operations. Each structure is implemented with unique traversal and deletion algorithms, and the time taken to delete a specific drone is recorded for comparison. This report summarizes the analysis and findings of the system's performance.

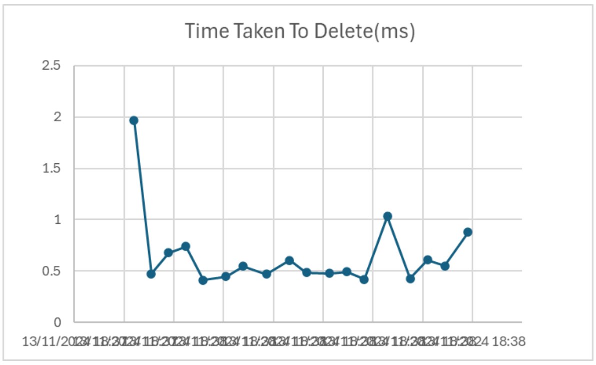
Analysis Data Comparison: Time Taken to Delete Drone

Linked List -



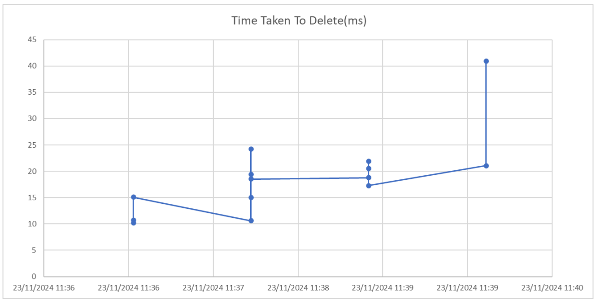
- Average deletion time: ~0.4-0.8ms
- More fluctuations in performance
- Initial high spike of 1.8ms
- Less consistent performance overall

Binary Tree -



- Average deletion time: ~0.5ms or less
- More stable and predictable performance
- Though has one spike to 2ms, generally maintains lower and more consistent times
- Tighter range of execution times

Graph + Djikstra -



- Average deletion time: ~10ms - 40ms
- Stable and low deletion times but shows fluctuations as the test progresses.
- A sudden spike to 40 ms is observed, likely due to deleting a node with multiple edges
- as the graph becomes more complex, deletion times show variability

Discussion and Analysis

The performance of each structure reflects its inherent algorithmic complexity:

Linked List is simple but suffers from linear search overhead. It is unsuitable for large-scale applications where frequent or indexed deletions are required. Time Complexity:

- Best Case: O(1) for deleting the first node.
- Worst Case: O(N) for deleting a node at the end of the list.

Binary Tree provides significant performance improvements over the linked list. However, tree balancing is critical for maintaining efficiency, as skewed trees can degrade performance. Time Complexity:

- Best Case: O(logN) for a balanced binary tree.
- Worst Case: O(N) for a highly unbalanced tree.

Graph demonstrates relatively stable performance but can become resource-intensive for larger networks due to adjacency management. Time Complexity:

- **Best Case**: O(VlogV+E), where V is the number of vertices and E is the number of edges. This occurs when using a priority queue (min-heap) to select the minimum distance node efficiently.
- **Worst Case**: O(V^2) if using a simple array for the priority queue, which results in slower performance for dense graphs.

Final Conclusion

The comparative analysis of linked list, binary tree, and graph structures highlights the importance of choosing appropriate data structures based on the application requirements.

- The linked list is best for simple, small-scale networks.
- The binary tree is optimal for balanced workloads, offering efficient deletion operations with proper maintenance.
- The graph is most suitable for dense and interconnected networks but requires careful handling of memory and complexity.