Data Structure Design and Implementation

Bikram Rumba

UC ID: 005030721

MSCS532: Algorithms and Data Structures

Medians and Order Statistics

Professor: Satish Penmatsa

February 09, 2025

Selection Algorithms

Selection Algorithms are fundamental concepts in computer science that involve finding the 'k' smallest element within an unsorted array. Two major selection algorithms are randomized selection (quicksort) and deterministic selection (median of medians). The randomized algorithm selects the random pivot element and partitions the array. The deterministic algorithm divides the array by finding the middle or median pivot element.

Implementation

1. Deterministic Algorithm (Median of Medians)

- It implements selection in wors-case linear time.
- It divides the array by finding medians and uses recursion to determine the pivot and sorts the array.
- Acquire execution time of O(n) for the worst-case complexity.

2. Randomized Algorithm (Quickselect)

- Randomly selects a pivot element and partitions the array.
- After partitions, it compares either left or right to select the position of elements with the selected pivot element.
- It will reach the average complexity time of O(n), but in the worst case, it would hit O(n^2).

Performance Analysis

1. Time Complexity:
    - Deterministic Selection: Worst-case complexity of O (n).
    - Randomized Selection: Average- scenario O(n) expected, and worst-case O(n^2).

2. Space Complexity: Both deterministic and randomized selection algorithms perform with O (1) additional space.
3. Empirical Analysis:
    - Different running times for random, sorted, and reverse-sorted input distributions. Both selection algorithms have different running times based on inputs.
    - It looks like expectations and observations are aligned.

Elementary Data Structures:

Elementary data structures provide a means to store and manage data efficiently. The primary data structures are arrays, matrices, stacks, queues, and linked lists. Arrays and matrices are fixed-size storage that allows random access and flexible for insertion, deletion, and access. Stacks works with the Last In, First Out (LIFO) principle with push and pop operations. Queues work with the First In, First Out (FIFO) principle with enqueue and dequeue operations. A linked list is a dynamic list of nodes that allows insertions, deletion, and traversal operations.

Implementation

1. Arrays and matrices implement the access, insertion, and delete operations.
2. Stack supports push (insertion) and pop (removal) operations.
3. Queue enqueue (insertion) and dequeue (removal) operations.
4. The linked list supports deletion, insertion, and traversal operations.

Performance

1. Time Complexity

- The array takes O (1) for access and O (n) for insertion/deletion.
- Stack takes O (1) for both push and pop operations.
- Queue time complexity is also O (1) for enqueue and dequeue.

- The linked list took O (1) time complexity for insertion and deletion and O(n) to traverse (search).

2. Space Complexity: Arrays require memory, whereas linked lists use dynamic memory allocation.
3. Arrays are more suitable for random access. On the other hand, linked lists are more efficient for dynamic insertions and deletions.

Selection algorithms and elementary data structures are crucial in optimizing computational efficiency in various applications. The deterministic Median of Medians algorithm ensures worst-case linear time, making it reliable for applications requiring consistent performance. In contrast, the randomized select algorithm is often faster in practice but can degrade in the worst case.

Similarly, elementary data structures like arrays, stacks, queues, and linked lists offer distinct advantages. Arrays provide fast random access, stacks, and queues efficiently manage ordered data, and linked lists offer flexibility in dynamic memory management. Understanding the trade-offs between these structures enables better algorithm and system design decision-making.

References

Cormen, T. H., & Leiserson, C. E. (2022). *Introduction to algorithms* (4th ed.). The MIT Press.

Erickson, J. (2023). *Algorithms*.

GeeksforGeeks. (n.d.). Data Structures and Algorithms. Retrieved from https://www.geeksforgeeks.org/data-structures/

Hoare, C. A. R. (1961). Algorithm 65: Find. Communications of the ACM, 4(7), 321-322.