**Assignment 6: Medians and Order Statistics & Elementary Data Structures**

Sailesh Limbu

University of the Cumberlands

Fall - Algorithms and Data Structures (MSCS-532-A01) - First Bi-term

Dr. Cooper

September 29, 2024

**Introduction:**

Fundamental data structures and techniques for selecting the kthk^{th}kth lowest element from a dataset are also covered in this research. We look at two important algorithms: the deterministic Median of Medians technique and the randomized Quickselect algorithm (Alpaydin, 2020). The former guarantees a time complexity of O(n)O(n)O(n), while the latter uses randomly selected pivots to give an anticipated performance of O(n)O(n)O(n). In addition to these methods, we study and compare the efficiency and appropriateness of fundamental data structures including linked lists, stacks, queues, and arrays.

**Part 1: Selection Algorithms**

**Implementation:**

Median of Medians is a deterministic technique for finding pivots that evenly distribute datasets; it guarantees worst-case linear time O(n)O(n)O(n). This ensures that the performance remains constant irrespective of the features of the input data. By comparison, picking a pivot at random allows the randomized Quickselect method to reach predicted linear time (Alpaydin, 2020). Although this approach works well in most cases, it could become less efficient under some conditions, including when using poorly distributed pivots.

A screenshot of a computer program

Description automatically generated

**Performance Analysis:**

The deterministic Median of Medians approach has a worst-case temporal complexity of O(n)O(n)O(n), which is guaranteed by its systematic pivot selection, which ensures that the data set is divided evenly. For each dataset, this gives it a trustworthy option for choosing the kthk^{th}kth smallest element. Conversely, in the majority of instances, the predicted time complexity of the randomized Quickselect method is O(n)O(n)O(n). Even though it's often quicker on average, it might fail miserably in certain scenarios due to how reliant its performance is on the quality of the randomly chosen pivot.

**Empirical Analysis:**

Empirical studies reveal that the deterministic method performs consistently with many kinds of input data, including sorted, reverse-sorted, and randomly ordered data. Applications requiring a guarantee of worst-case performance will benefit greatly from its predictability. The deterministic technique is usually faster than the random one, particularly with randomly ordered data (Alsuwaiyel, 2021). However, in the worst-case situation, where the pivot selection causes unequal partitions, the randomized process may decline dramatically.

A screen shot of a computer program

Description automatically generated

**Part 2: Elementary Data Structures**

**Implementation:**

Basic data structures are also covered in the study. The indexed structure of arrays allows for constant-time access to items but inserts and removals in arrays are linearly complicated. The use of dynamic arrays in the implementation of stacks and queues enables constant-time operations like pushing, popping, enqueuing, and dequeuing (Alsuwaiyel, 2021). Also created are linked lists, which are useful for dynamic data sets because they allow constant-time insertion at the head while giving linear-time navigation.

**Performance Analysis:**

Arrays are great with static datasets that don't change much and need data access often, but they become inefficient with frequent inserts and deletions since they need to relocate elements. However, linked lists' reliance on linear traversal makes them unsuitable for random access, while they excel at handling dynamic insertions at the head and tail (Alsuwaiyel, 2021). For fundamental operations like inserting and removing items, stacks and queues built using dynamic arrays continuously execute with O(1)O(1)O(1) time complexity.

**Practical Applications:**

Every data structure has real-world applications. For datasets of a fixed size that need data access often, arrays are the way to go. Recursion, function call management, and reversing algorithms often make use of stacks, which adhere to the last-in-first-out (LIFO) concept. Schedules, task buffers, and real-time data processing are all made possible by queues, which adhere to the first-in, first-out (FIFO) concept (Alsuwaiyel, 2021). Linked lists are better than other types of lists when systems that need dynamic memory allocation or real-time applications have to insert and remove items often.

**References**

Alpaydin, E. (2020). Introduction to machine learning. MIT press.

Alsuwaiyel, M. H. (2021). Algorithms: design techniques and analysis (Vol. 15). World Scientific.