Abstract

1 Introduction

2 Definitions

field A variable associated with an object that may be read from or written to. It is assumed that fields can be read from and written to in constant time.

get indexer A method that gets an item at a specified index in a list.

indexer Refers to a get indexer or set indexer.

property A function of an object that returns a value, runs in constant time, and does not mutate any state (i.e. by assigning new values to fields).

set indexer A method that sets an item at a specified index in a list.

trivial member A field or property of an object.

3 Predefined Functions

4 Trivial Members

In this section, I define **fields** and **properties** on dynamic and funnel arrays. These will be used later to implement operations on them in pseudocode.

5 Common Funnel Array Operations

In this section, I implement common operations for dynamic and funnel arrays in pseudocode. Then, I analyze and compare the time complexities of the implementations. If the operation allocates memory, I also compare their space complexities.

The following mathematical definitions will be used while analyzing time and space complexity:

$P(f(n))$ This is an alternative to big-O notation for time complexity; I will call it **big-P notation**.

The main purpose of big-P notation is to preserve the constant coefficient of the fastest-growing term in $f(n)$, while still maintaining many properties of big-O. For example, $O(2n) = O(n)$, but $P(2n) \neq P(n)$. This makes it possible to compare time complexities whose ratio converges to a constant as n becomes larger.

More formally, $P(f(n)) = P(g(n))$ iff

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$$

.

5.1 Adding

Description

Adding is the most common operation done on a dynamic array[1]. Funnel arrays improve the performance of adding in two ways: by allocating less memory, and reducing the amount of copying.

Dynamic array implementation

Time complexity

$$W(n) = P(2n)$$

Space complexity

$$S(n) = P(2^{\lceil \log_2 n \rceil + 1}) = O(n)$$

Funnel array implementation

Time complexity

$$W'(n) = P(n)$$

[1]

Space complexity

$$S'(n) = P(2^{\lceil \log_2 n \rceil}) = O(n)$$

Time complexity comparison

Space complexity comparison

$$r_S = \frac{P(2^{\lceil \log_2 n \rceil})}{P(2^{\lceil \log_2 n \rceil + 1})} = \frac{1}{2}$$

5.2 Indexing

Description

Indexing is another very common operation on a list. An **indexer** gets or sets an item at a specified index in a list.

Dynamic array implementation

Time complexity

$T(n) = O(1)$

Time complexity

$T'(n) = O(1)$

For the rest of this paper, I will not assume the indexer's time complexity is $O(1)$ nor $O(\log n)$. Instead, I will denote it by the function $I(n)$, where $I(n)$ is either 1 or $\log n$.

Since many list operations I detail can be implemented using the indexer in a loop, I make the following useful claim:

Let L be a list. Suppose LC is a loop construct that runs N times, where N is a fixed number. If the body of the loop contains only the indexer of L and statements that run in constant time, and the loop does not contain nested loops, then the time complexity of the loop is $O(N \cdot I(n))$.

To see this, note that the time complexity of the body of the loop is $O(I(n))+O(1) = O(I(n))$. Since the body is run N times, the total time complexity is $N \cdot O(I(n)) = O(N \cdot I(n))$.

Time complexity comparison

5.3 Iterating

Description

Iteration of a list is the process of performing some action on each of its elements. The standard implementation of iteration for dynamic arrays is also applicable to funnel arrays. When this happens in this and subsequent sections, the second heading will be titled "Common implementation".

Common implementation

By , the time complexity of iteration is $O(n \cdot I(n))$ for all lists[2].

For dynamic arrays, $I(n) = 1$ and this is the standard way of implementin. Thus, their time complexity for iteration equals $O(n)$.

Time complexity

Funnel array implementation

$T'(n) = O(n)$

Time complexity comparison

[2]