EECS3101 (Section E) Fall 2025
Tutorial: Week 3
Amortized/Average Analysis of Dynamic Arrays

Chen-Wei Wang

Release Date: Friday, September 19
No Submission Required: Complete for Learning & Test Prep

Contents

1 Analyzing the Doubling Strategy

Consider the *doubling* strategy for dynamic arrays:

```
1  public class ArrayStack<E> implements Stack<E> {
2    private int I;
3    private int capacity;
4    private E[] data;
5    public ArrayStack() {
6      I = 1000; /* arbitrary initial size */
7      capacity = I;
8      data = (E[]) new Object[capacity];
9      t = -1;
10   }
11   public void push(E e) {
12     if (size() == capacity) {
13       /* resizing by doubling */
14       E[] temp = (E[]) new Object[capacity * 2];
15       for(int i = 0; i < capacity; i ++) {
16         temp[i] = data[i];
17       }
18       data = temp;
19       capacity = capacity * 2;
20     }
21     t++;
22     data[t] = e;
23   }
24 }
```

(**Task 1**) Derive the tightest asymptotic upper bound on the average/amortized running time of the **push** operation.

(**Task 2**) Compare and contrast the *worst*-case vs. *average*-case running times of the **push** operation (implemented via a dynamic array using the *constant increments* vs. *doubling* strategies).