**CS 3100 / In-class Activity 6, Reductions and Worst Case Lower Bounds Proofs**

|  |  |
| --- | --- |
| **Name** | **Computing ID** |
| *Your Name: Carlos F Revilla* | cfr5spw |

**In class:**  You must work in teams of 2, 3 or 4. Each person writes answers and turns in the sheet at end of class.  **Missed class?**  Work alone and answer to the best of your ability. Submit to GradeScope by 9am on the 2nd day after in-class activity.

Today, we will prove the following:

Prove whether there exists a data structure where the operations INSERT (which inserts a given element into the data structure), DELETE (which removes a given element from the data structure, should it be present), and FINDMIN (which returns the minimum element from the data structure) require  *O(1)*  worst-case time each. The operations only use key-comparisons as their basic operation.

**1.**  Discuss with your team. Do you think such a data structure exists?

1. . List some data structures that have some of these properties. For example, INSERT into a Linked List is worst-case  *O(1)* , since we can insert at the beginning of the list. What about a Priority Queue?
2. . Assume we want to prove such a structure does not exist using a reduction. In that case, we need a problem. What problem should we use?
3. . Assume the problem we chose in question 3 is  **Problem A.**  What’s the direction of the reduction we want to make for a worst-case lower bound proof? Should we use an algorithm that solves  **Problem A**  to make an algorithm that uses this data structure to solve another problem ( **Problem B** ) ; or should we use this data structure to make an algorithm to solve  **Problem A** ? Additionally, write down the direction of the reduction:  *Problem X reduces to Problem Y*  (and stating what X and Y are).

**5.**  Describe the construction of the reduction. How do we take instances of one problem and convert them into instances of the other? How do we get the correct result for the first problem?   
  
  
**(Note to grader I forgot to print this out so I’m doing it on word and its formatted all weird so I’m just going to type my answers below)**

**Question 1:**

you cannot have an arbitrary data structure supporting INSERT, DELETE, and FINDMIN operations in O(1) time using only key comparisons because any comparison-based sorting algorithm has a lower bound of nlogn for sorting. So since finding the minimum is the same as finding the smallest element in a sorted list it can't be done in O(1) time using only key comparisons

**Question 2:**

- Linkelists: INSERT can be O(1) at the beginning.

- Priority Queues: insert is usually O(1) (depending on the implementation), DELETE and FINDMIN usually have at least O(log n) time complexity.

**Question 3:**

Need to use a Lower bound proof. The problem is the element distinctness problem

**Question 4:**

reduction direction would be: Element Distinctness reduces to Data Structure Operations.

IE : if you can solve the element distinctness problem with your proposed data structure then the data structure would have to perform at least as much work as solving the element distinctness problem.

**Question 5:**

- Each element in the set is inserted into the data structure using the INSERT operation.

- check for distinctness by performing the FINDMIN operation. If the minimum element is not unique then the elements in the set are not unique.

- If FINDMIN returns a unique minimum element DELETE that element from the data structure and repeat the process for the remaining elements.

- If at any point we find that the minimum element is not unique then we conclude that the elements are not distinct.

Using this reduction we have established a lower bound on the time comp of the data structure operations showing that they must be at least as hard as solving the element distinctness problem- which is known to have a lower bound of nlogn for comparison-based algorithms.