Data Structures

Fall 2019, Homework #5

Date: Dec. 30, 2019

1. (30 pts) Consider a hash table consisting of $M = 11$ slots, and suppose nonnegative integer key values are hashed into the table using the hash function $h1()$:

```
int h1 (int key) {
    int x = (key + 7)* (key + 7);
    x = x / 16;
    x = x + key;
    x = x % 11 ;
    return x
}
```

Suppose that collisions are resolved by using linear probing. The integer key values listed below are to be inserted, in the order given. Show the home slot (the slot to which the key hashes, before any probing), the probe sequence (if any) for each key, and the final contents of the hash table after the following key values have been inserted in the given order.

key value	home slot	probe sequence
43		
23		
1		
0		
15		
31		
4		
7		
11		
3		

Final Hash Table:

slot	0	1	2	3	4	5	6	7	8	9	10
content											

2. (20 pts) You have a set of elements a through j (i.e., $\{a\}$, ..., $\{j\}$) and perform the following sequence of operations on a Disjoint Sets ADT:

- find(d)
- union(d, a)
- union(b, c)
- union(h, j)
- find(c)
- union(h, b)
- find(j)
- union(b, a)

(a) Without weighted union or path compression, and choosing the root of a union by selecting the alphabetically smaller node, show the up-tree forest which results from the above operations. At what depth does node j end up?

(b) With weighted union, but without path compression, and breaking ties on unions by selecting the alphabetically smaller node, at what depth does node j end up? It is not necessary to show the resultant up-tree forest again.

(c) With weighted union and path compression, and breaking ties on unions by selecting the alphabetically smaller node, at what depth does node j end up? It is not necessary to show the resultant up-tree forest again.

3. (20 pts) Draw the decision tree for Selection-sort for an array A[0..2] of $n = 3$ elements.

4. (30 pts) Consider an undirected graph with n vertices and m edges.

   (a) Give an algorithm that counts the connected components using a union-find data structure.

   (b) How much time does the algorithm take?

   (c) Can you count the connected components in $O(n + m)$ time? Show how and analyze your algorithm.