Problem set 3

by Maksim Al Dandan

February 6, 2024

Week 3. Problem set (solutions)

1. Consider a hash table with 13 slots and the hash function $h(k) = \left(k^2 - 2k + 7\right) \mod 13$. Show the state of the hash table after inserting the keys (in this order)

$$5, 28, 19, 15, 20, 33, 12, 17, 10, 13, 3, 34$$

with collisions resolved by linear probing [Cormen, Section 11.4].

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
Key	3		17	20	33	19	34	28	15	5	12	10	13

2. Consider the following algorithm (see pseudocode conventions in [Cormen, Section 2.1]). The inputs to this algorithm are a map M and a key k. Additionally, assume the following:

(a) The map M uses the positive integers both for keys and for values.
(b) The map M is not empty and contains n distinct keys.
(c) The map M is represented as a hashtable with load factor α.
(d) The map M resolves collisions via double hashing [Cormen, Section 11.4].
(e) For all keys k, if a value in M at k exists, then it is smaller than k.

```
1  secret(M, k):                                   0
2      total := 0                                  1
3      v := M.get(k)                               1/(1-α)
4      for i=1 to v                                k
5          if M.hasKey(i)                          1/(1-α) * k
6              total := total + M.get(i)           1/(1-α) * k
7      return total                                1
```

Compute the average case time complexity of secret. The answer must use O-notation and may depend on n, k, and α. For the average case analysis, use independent uniform permutation hashing. Assume worst case for the contents of the input map M.

Briefly justify your answer ($2-3$ sentences). Detailed proof for the asymptotic complexity is not required for this exercise.

Average case time complexity:

$$O(\frac{1}{1-\alpha} + \frac{2k}{1-\alpha}) \tag{1}$$

$$O(\frac{1}{1-\alpha}(2k+1)) \tag{2}$$

$$O(\frac{k}{1-\alpha}) \tag{3}$$

The **third** equation is the answer.
Because of the clause (e) $v \leq k$, hence, the working time of the loop is k at most. The methods of map (such as *get* and $hasKey$) are getting $\frac{1}{1-\alpha}$ cost due to the lecture presentation.

3. In your own words, explain how it is possible to implement deletion of a key-value pair from a hashtable with $O(1)$ worst case time complexity if collision resolution is implemented using chaining? Specify the data structures and methods involved.

In a hashtable with chaining for collision resolution, each bucket in the hashtable contains a linked list of key-value pairs that hash to the same index. To achieve $O(1)$ worst-case time complexity for deletion, we can follow these steps:

1. **Hashing Function**: Use a good hashing function that distributes keys uniformly across the hashtable. This ensures that the distribution of elements is balanced, minimizing collisions.

2. **Bucket Organization:** Implement the hashtable as an array of linked lists (buckets). Each bucket holds the key-value pairs that hash to the same index.

3. **Find the Bucket:** To delete a key-value pair, first, we need to find the appropriate bucket based on the hashed index of the key.

4. **Search in the Bucket:** Once we locate the bucket, we need to traverse the linked list within that bucket to find the specific key-value pair we want to delete. This operation typically requires $O(k)$ time complexity, where k is the average number of elements per bucket.

5. **Delete Operation:** To delete the key-value pair, we modify the pointers in the linked list to remove the node containing the pair. This operation is constant time $O(1)$, as it involves changing pointers within the linked list.

By following these steps, we ensure that the time complexity of deleting a key-value pair from the hashtable remains $O(1)$ in the worst case. However, it's essential to note that this $O(1)$ time complexity assumes a well-designed hashing function, a good distribution of elements across the buckets, and a relatively small average number of elements per bucket.

References

[Cormen] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. Introduction to Algorithms, Fourth Edition. The MIT Press 2022