Assignment 1 - Hash Tables
COMP 251 - Algorithms and Data Structures
Prof. Jérôme Waldispühl
Fall 2018

LE, Nhat Hung
McGill ID: 260793376
Date: October 7 2018
Due date: October 9 2018

Effects of increasing number of keys on average collision:

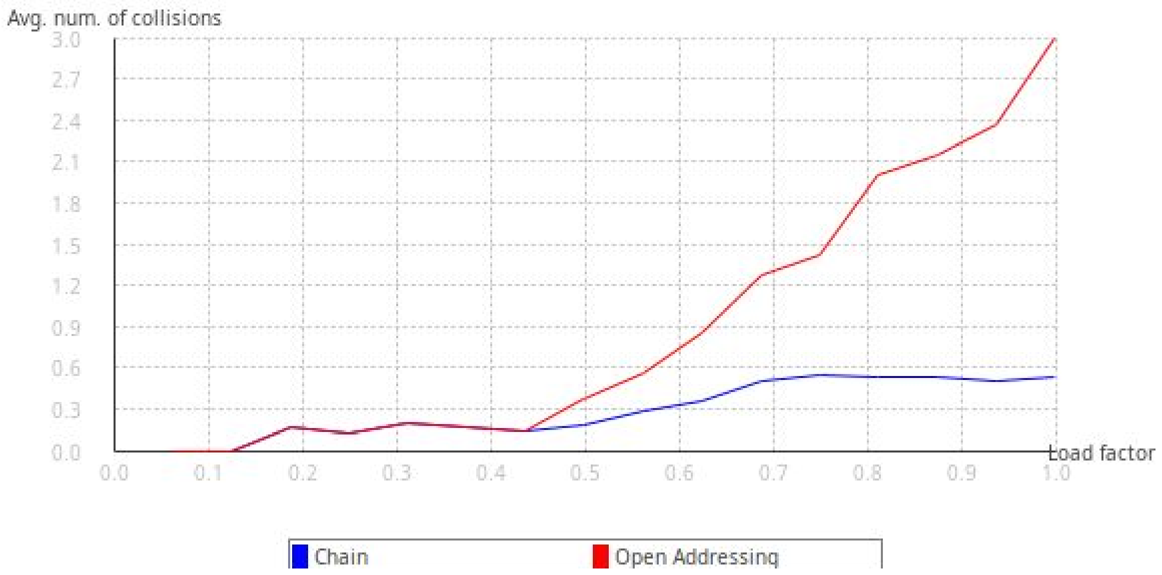

Figure 1: Average number of collisions in terms of hash table load factor

Open Addressing results in more collisions than Chaining the more the load factor increases, reaching about 5 times more when the load factor is 1 (number of keys = number of slots in table). But when the load factor is low (low n), the number of collisions is the same for both methods.

When load factor = 1, because Open Addressing attempts to insert in every single slot available in the table, the number of collisions is indeed higher than in Chaining.

When the load factor is lower, then there is less chance for collisions. Without collision, insertion is similar in both methods, which explains the also similar performances.

Observation of key removal in Open Addressing method:

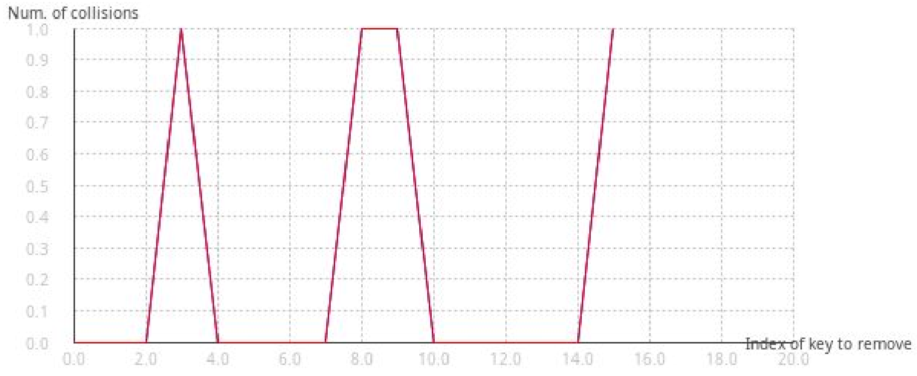

Figure 2: Number of collisions for each key to be removed

We observe that the maximum number of collisions is 1.

This performance is achieved by running the same hash function on the key to be removed, obtaining then the initial index of the key, moving downwards on the table if the key isn't found: just like how the key was inserted in the first place. Here the number of collisions is low because the load factor is only 0.5, but it would be higher if the load factor was itself closer to 1.

During this function, to indicate a key was removed, as opposed to the slot being empty in the first place, we set the slot to -2 instead of -1.

With this, if Table[hashValue] == -1, we know the key really isn't the table and stop the function.
Else if Table[hashValue] == -2, we know our wanted key might still be in the table and keep going.

Therefore, when trying to remove keys that aren't in the table, like

6, 8, 180, 3, 4, 5,

we know to stop immediately after running the hash function on them only once.

Effects of increasing w on average collision:

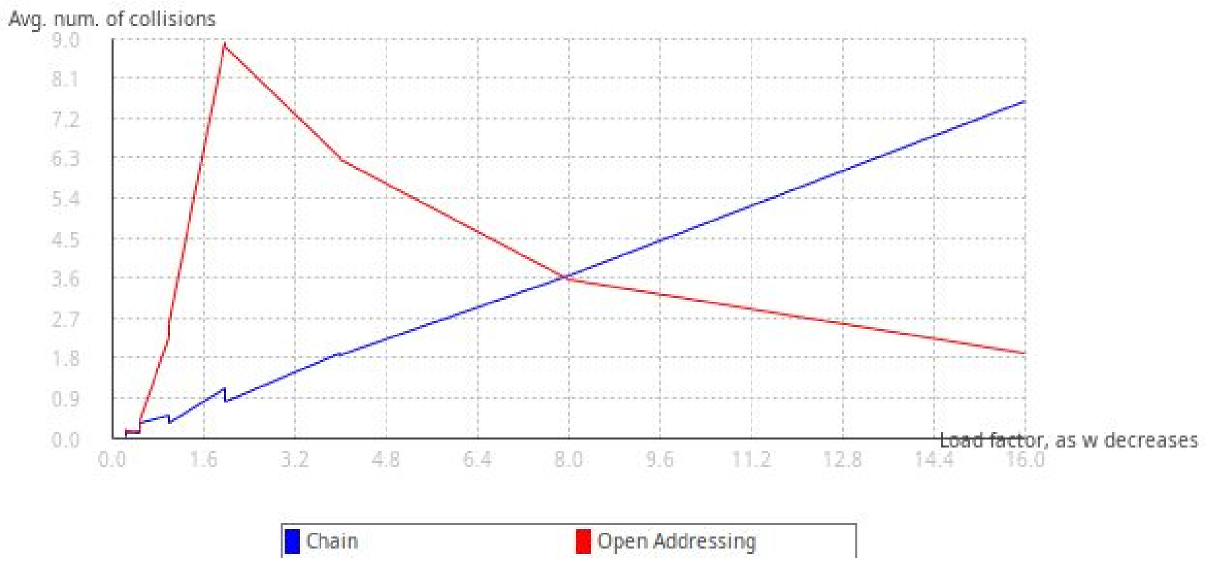

Figure 3: Values of w range from 12 to 1, with insertion of 32 non repeating keys

The size of the hash table, m, is calculated by

$$m = 2^r ,$$

$$\text{with } r = (w - 1) \div 2 + 1$$

Therefore, m increases as w increases. Because the load factor is inversely proportional to m, as w decreases, the load factor increases.

We observe that in Chaining, the number of collisions increases linearly as w decreases.
This is because Chaining always tries to insert a key even if every slot is taken in a linear fashion.

On the other hand, in Open Addressing, the number of collisions first spikes above the Chaining method's at load factor = 2 (w = 8 and 7), then decreases in a hyperbola pattern, crossing the blue line at load factor = 8 (w = 3).

For both methods, we see the number of collisions is almost zero when w is high, because if the load factor is low then there is more room for keys therefore fewer collisions.

Back to Open Addressing. We can explain the spike at load factor = 2, because this is when the number of keys is exactly twice the size of the table.

Say we have

6 keys
and the size of the table is 3.

Then, in the best case, we inserted the first 3 keys without collisions. Then, for each of the last 3 keys the number of collisions would be 3. The lowest number of collisions is thus:

$$3^2 = 9\text{, or } m^2\text{,}$$
$$\text{with } m \text{ the number of keys to insert}$$

When the size of the table decreases (load factor increases), we can apply the same logic. We still have 6 keys, but now the table only has 2 slots. The lowest number of collisions is then $4 \times 2 = 8, \; < 9$. When the table has 1 slot, the lowest number of collisions is $5 \times 1 = 5, \; < 9$.

In general, then, as the load factor increases, the best case number of collisions is always lower than m^2. This is the reason why in figure 3, as the load factor increases, the number of collisions in Open Addressing keeps decreasing.

In conclusion:

For this conclusion, we will define the quality of the hash table as filling out as occupying the most table slots possible. From above, we see that the Open Addressing method encounters the most collisions, but only because its nature is to handle the most collisions to fill out all table slots.

The number of collisions increases more stably with the the table size decreasing (load factor increasing) in the Chaining method, while Open Addressing would not be able to store all keys if the load factor exceeds 1. However, it is good practice in creating hash tables to keep the load factor lower than 1 anyway.

Therefore, according to this definition of quality, we conclude that Open Addressing is the better hashing method.

End of report.