CS124	**Lecture 15**	Spring 2015

1 Static dictionary problem

In this lecture we will study the static dictionary problem and describe two-level perfect hashing. Sometimes it is also called "FKS perfect hashing" after its inventors: Fredman, Komlós, and Szemerédi.

Recall in the static dictionary problem you are given n (key, value) pairs up front and want to create a data structure supporting query (but not insert and delete). We will show how to do this in $O(1)$ expected time and in linear space.

To begin, recall the birthday paradox where, assuming random birthdays, you shouldn't be surprised that two people have the same birthday when you have $\sqrt{365}$ people in one room. We will use a universal hash family $\mathcal{H}$ and a hash table where $m = n$ where m is the space of the hash table, and n is the number of values of the hashed elements.

1.1 Quadratic space

If we were willing to make a table whose size is quadratic in size n. Then we can easily construct a perfect hash value. Let $\mathcal{H}$ be a universal hash family and $m = n^2$.

Claim: If $\mathcal{H}$ is universal and $m = n^2$, then $P_h(\text{no collisions} \geq 1/2$ when using hash function $h \in \mathcal{H}$.

Proof: In order for a collision to occur, elements x, y must equal each other. Of the $\binom{n}{2}$ pairs, the chance they collide is $\leq 1/m$ by definition of universal hash family. Then the probability a collision occurs $P_h(\text{collision occurs}) \leq \binom{n}{2}/m < 1/2$.

This is the opposite of the birthday paradox since we are looking for the probability that no pair of people has the same birthday. The complement then shows that the probability of no collisions must $\geq 1/2$. Our method then involves trying a random h from $\mathcal{H}$. If we have any collisions, we pick another h. On average, we would only need to do this twice.

1.2 Linear space

Let's say we want to get a better space complexity. The general idea is that we are going to perform a 2-level hashing scheme: first we pick a random function $h_i \in \mathcal{H}$ from the universal hash family and hash all elements. Let B_i represent the number of items that hash to bucket i. We then wish to keep picking random $h_i \in \mathcal{H}$ until we find h_i such that

$$\sum_{i=1}^{m} B_i^2 \leq 4n$$

Note that we do not know how long it will take to find h_i that fulfills the above condition. Once we do find our satisfactory h_i, we can use the birthday paradox, which states that if we have n possible days in the year, once we

have $\sqrt{n}$, we can expect to find two people with the same birthday. Likewise, if we have significantly fewer than $T = \sqrt{n}$, if we taking values between 1 and T^2 and we have much less than T people, we can be pretty sure that there isn't a collision.

For bucket i, we can then hash all B_i of the values in it to $1, \ldots, B_i^2$ using $h_i : [U] \to [10B_i^2]$, then by the birthday paradox, there are no collisions. To summarize, the entire method is then

1) First we take our elements that take on values $v \in [n]$ and hash them using first-level hash function $h : [U] \to [m]$ such that
$$\sum_{i=1}^{m} B_i^2 \leq 4n$$
where B_i is the number of items in the bucket i. This creates first-level table A.

2) Next, we use m second-level hash functions $h_1, \ldots, h_m$ and m second-level tables $A_1, \ldots, A_m$. Note that we pick h_i such that there are no collisions in A_i.

Thus, we can ensure no collisions in the 2-level hash table structure. Our space complexity is $O(m + \sum_{i=1}^{m} 10B_i^2)$ since we need m first-level buckets and $\sum_{i=1}^{m} 10B_i^2$ second-level buckets. Note that since we chose h such that the sum of $\sum i = 1^m B_i^2 \leq 4m$, our space complexity is $\Theta(n)$.

Let's back up though. We initially said to keep picking h such that $\sum_{i=1}^{m} B_i^2 \leq 4n$. How long will that take?

Claim: $P_h(\sum_{i=1} B_i^2 > 4n) \leq 1/2)$

Proof: Let $Q_{ji} = 1$ if item j hashes to i and 0 otherwise. We can rewrite B_i as

$$\begin{aligned}
B_i &= \sum_{j=1}^{n} Q_{ji} \\
B_i^2 &= \left(\sum_{j=1}^{n} Q_{ji} \right)^2 = \sum_{j=1}^{n} Q_{ji}^2 + \sum_{j \neq k} Q_{ji} Q_{jk} \\
&= \sum_{j} Q_{ji} + \sum_{j \neq k} Q_{ji} Q_{ki} \qquad (1) \\
\sum_{i=1}^{m} B_i^2 &= \sum_{i=1}^{m} \sum_{j=1}^{n} Q_{ji} + \sum_{i=1}^{m} \sum_{j \neq k} Q_{ji} Q_{ki} \\
&= n + \sum_{i=1}^{m} \sum_{j \neq k} Q_{ji} Q_{ki} \qquad (2) \\
\mathbb{E}\left(\sum_{i=1}^{m} B_i^2 \right) &= n + \mathbb{E}\left(\sum_{i=1}^{m} \sum_{j \neq k} Q_{ji} Q_{ki} \right) \\
&= n + \mathbb{E}\left(\sum_{i=1}^{m} \sum_{j \neq k} Q_{ji} Q_{ki} \right) \\
&= n + n - 1 = 2n - 1 \qquad (3)
\end{aligned}$$

Above, we show a few significant and potentially non-intuitive steps. In equation 1, we can drop the square of $\sum_{j=1}^{n} Q_{ji}$ since Q_{ji} is either 0 or 1. In equation 2, we can reorder the summations: we know that $\sum_{i=1}^{m} Q_{ji} = 1$ since item j will hash to exactly one of the m values. We then find $\sum_{j=1}^{n} 1 = n$.

For equation 3, we know that $Q_{ji}Q_{ki}$ will have a product of 1 if and only if $h(j) = h(k)$, meaning j and k hash to the same bucket. Our resulting term is then

$$\mathbb{E} \sum_{j \neq k} 1 \text{ if } h(j) = h(k)$$

We can then linearize the expectation inside the summation, yielding the probability that the two item j,k collide, which is exactly universal hashing. The probability of collision $\leq 1/m$, and there are $n(n-1)$ possibilities since there are n choices for j and $n-1$ choices for k, giving us

$$\sum_{j \neq k} P(\text{collide}) = \frac{n(n-1)}{m}$$

Since $m = n$, we get that $\sum_{j \neq k} P(\text{collide}) = n - 1$.

From Markov's inequality, we get

$$P(X > \lambda \mathbb{E}X) < \frac{1}{\lambda}$$
$$P\left(\sum_{i=1}^{m} B_i^2 > 4n\right) < \frac{1}{2}$$

This clears up the mystery of why we chose 4 as a coefficient in $\sum_{i=1}^{m} B_i^2 \leq 4n$: because it's twice the expectation.

Now we know when picking h randomly from $\mathcal{H}$, we have to pick an expected two times to fulfill our condition. Picking h_i is a similar story.

Claim: When picking $h_i : [U] \to [10B_i^2]$, $P(\exists \text{ collision}) < 1/2$

Proof: We define again $X_{jk} = 1$ if j,k collide under h_i and 0 otherwise. The expected number of collisions is then

$$\begin{aligned}
\mathbb{E}(\#\text{ of collisions}) &= \mathbb{E} \sum_{j=1}^{B_i} \sum_{k=1}^{j-1} X_{jk} \\
&= \sum_{j=1}^{B_i} \sum_{k=1}^{j-1} \mathbb{E}X_{jk} \\
&= \sum_{j=1}^{B_i} \sum_{k=1}^{j-1} P(j,k \text{ collide under } h_i) \\
&\leq \sum_{k<j} \frac{1}{10B_i^2} \\
&< \frac{B_i^2}{2} \cdot \frac{1}{10B_i^2} = \frac{1}{20}
\end{aligned}$$

By Markov's inequality again, $P(\#\text{ of collisions} \geq 1) < 1/20$

We have now shown that we can construct a 2-level hash table using hash functions h and $h_1, \ldots, h_m$ to make a static dictionary since h and $h_1, \ldots, h_m$ can be chosen in constant time each and linear time overall.