CSE525: Randomized Algorithms and Probabilistic Analysis April 18, 2013

Lecture 5

Lecturer: Paris Koutris *Scribe: Jerry Li and Noah Siegel*

1 Introduction and Definitions

Fix a large universe U and let $|U| = N$. Fix some $m < N$ (in practice m will be much less than N). We want a strategy so that for all subsets $S \subseteq U$ with $|S| \leq m$ we can "look up" elements in S in constant time.

Heuristically, we want a hash function $h : U \to \{0, \ldots, n-1\}$ and we want that

- h is easily computable and doesn't take too much space to represent
- number of collisions are small; i.e. $\#\{x, y \in S : h(x) = h(y)\}$ is small
- $n = O(|S|)$

Given such an h, we hash all the elements of S into a hash table of size n using h, with collisions resolved using linked lists. Then, given some $x \in U$ we can test whether $x \in S$ by hashing x and checking all the elements in the linked list at that bucket. Using somewhat non-standard notation, we will say that h takes values in n bins.

We say that $h : U \to \{0, \ldots, n-1\}$ is a *perfect hash* for S with $|S| \leq m$ if for all $x, y \in S$ with $x \neq y$ we have $h(x) \neq h(y)$. We will show that we can efficiently construct perfect hashes that can be represented in polynomially many bits using randomization.

Ideally, if h mapped $x \in U$ uniformly at random to a bin, then the distribution of collisions would reduce to that of balls and bins, and the maximum number of collisions would be less than or equal to $\log m$ with high probability. However such a map would in general require a lookup table with $O(N)$ elements, each of size $O(\log n)$.

To get around this, we define a hash family $\mathcal{H} = \{h_i\}_{i=1}^{k}$ to be a set of hash functions where each h_i takes values in n bins, where n is some fixed constant $n < N$ independent of h_i. Notice that the naive procedure described above is equivalent to choosing a hash function uniformly at random from the hash family of all hash functions taking values in n bins, which is too large to do efficiently. Instead, we will try to approximate such a distribution by choosing a hash function from a much smaller hash family with certain properties that we will define.

2 Universal Hash Functions

Our strategy will be to define a hash family $\mathcal{H} = \{h_i\}_{i=1}^{k}$ where $h_i : U \to \{0, \ldots, n-1\}$ for each i.

Definition 1. *$\mathcal{H}$ is 2-universal if for all $x, y \in U$ such that $x \neq y$ we have*

$$\Pr_{h \sim \mathcal{H}}(h(x) = h(y)) \leq \frac{1}{n}$$

where $h \sim \mathcal{H}$ means that h is selected uniformly at random from $\mathcal{H}$.

Remark This definition only guarantees bounds for collisions between two variables. There may be a lot of correlation between groups of 3 or more variables. More generally, a hash family $\mathcal{H}$ is k-universal if for all $x_1, \ldots, x_k \in U$ such that $x_i \neq x_j$ for $i \neq j$ then

$$\Pr_{h \sim \mathcal{H}}(\exists i, j \text{ s.t. } i \neq j \text{ and } h_{x_i} = h(x_j)) \leq \frac{1}{n^{k-1}}.$$

However, we only need this for $k = 2$.

We first show that easily representable 2-universal hash families exist.

Theorem 2. *For all $n \leq N$, there exists a 2-universal hash family whose members take values in n bins, take $O(\log N)$ bits to represent uniquely, and take $O(1)$ time to evaluate.*

Proof We will define this family explicitly. It suffices to consider the case when $U = \{1, \ldots, N\}$ as any other set of size N can be mapped bijectively onto this set. Pick a prime $N \leq p \leq 2N$, which is possible by Chebyshev's theorem. For any any $a, b \in \mathbb{N}$ define

$$h_{a,b}(x) = [(ax + b) \mod p] \mod n$$

and we let

$$\mathcal{H} = \{h_{a,b} : a \in \{1, \ldots, p-1\}, b \in \{0, \ldots, p-1\}\}.$$

Notice that any member of this family can be represented uniquely by the 3-tuple (a, b, p) so each hash function takes at most $3\log(2N)$. bits Furthermore, the function contains a constant number of elementary operations which take constant time to evaluate, so each $h_{a,b}$ can be evaluated in constant time. We claim that this family is a 2-universal family. Fix some $x, y \in U$ with $x \neq y$. Without loss of generality assume $x > y$. For any $a \in \{1, \ldots, p-1\}$ and $b \in \{0, \ldots, p-1\}$ let $r = (ax + b) \mod p$ and $s = (ay + b) \mod p$.

Notice first that $r \neq s$ as otherwise we would have $ax + b \equiv ay + b \mod p$ so $p|a(x-y)$ which is impossible as both a and $(x - y)$ are positive and less than p. Furthermore, for $r \neq s$ fixed there exist unique solutions for a, b in the desired range as

$$a \equiv (r - s)(x - y)^{-1} \mod p$$

where x^{-1} here is the multiplicative inverse of x mod p which exists as p is prime and $x \neq 0$. This clearly yields a unique solution for $a \in \{1, \ldots, p-1\}$, and so

$$b \equiv r - ax \mod p$$

which yields a unique solution for $b \in \{0, \ldots, p-1\}$. Thus

$$h_{a,b}(x) = h_{a,b}(y) \Leftrightarrow r - s \equiv 0 \mod n.$$

Therefore the number of collisions is the size of the set

$$\begin{aligned} \#\{h \in \mathcal{H} : h(x) = h(y)\} &= \#\{(r, s) \in \{0, \ldots, p-1\} : r \neq s \ \& \ r \cong s \mod n\} \\ &\leq p\left(\lceil \frac{p}{n} \rceil - 1\right) \end{aligned}$$

as for any fixed r there are at most $\left(\lceil \frac{p}{n} \rceil - 1\right)$ $s \in \{0, \ldots, p-1\}$ with $r \neq s$ and $r \equiv s \mod p$. Since $\lceil \frac{p}{n} \rceil = \lceil \frac{p-1}{n} \rceil$ as p is prime and hence cannot divide n, we obtain that the expression above is bounded by $p\left(\lceil \frac{p-1}{n} \rceil - 1\right) \leq \frac{p(p-1)}{n}$. As $|\mathcal{H}| = p(p-1)$ we have

$$\Pr_{h \sim \mathcal{H}}(h(x) = h(y)) = \frac{\#\{h \in \mathcal{H} : h(x) = h(y)\}}{p(p-1)} \leq \frac{\frac{p(p-1)}{n}}{p(p-1)} = \frac{1}{n}$$

so $\mathcal{H}$ is 2-universal, as claimed. ■

We claim that a function selected uniformly at random from a 2-universal hash family hashes with few collisions on average. The following theorem is a rigorous statement of this intuition.

Theorem 3. *Let $\mathcal{H}$ be a 2-universal hash family taking values in n bins, and fix some subset $S \subset U$. Fix $u \in U$, and let X_u be the random variable which is the look-up time for x in S using $h \sim \mathcal{H}$ as our hash function. Then*

$$E(X_u) \leq 1 + \frac{|S| - 1}{n}$$

Proof Notice that X_u is bounded by the length of the linked list in the bucket of the hash table where h is hashed. There are now two cases:

Case 1: $u \in S$ Then we are guaranteed to see u in the bucket, so it suffices to count the remaining number of conflicts.

$$E(X_u) \leq 1 + E(\#\text{ of collisions with } u) = 1 + \sum_{y \in S, y \neq u} \Pr_{h \sim \mathcal{H}}(h(u) = h(y))$$
$$= 1 + \sum_{y \in S, y \neq u} \frac{1}{n} = 1 + \frac{|S| - 1}{n}$$

where the second line follows from our assumption of 2-universality.

Case 2: $u \notin S$ Here instead we are guaranteed not to find u, and so

$$E(X_u) = E(\#\text{ of collisions with } u) = \sum_{y \in S, y \neq u} \Pr_{h \sim \mathcal{H}}(h(u) = h(y)) \leq \frac{|S|}{n} \leq 1 + \frac{|S| - 1}{n}$$

and so we are done. ■

In fact, we can use 2-universal hash families to construct perfect hash functions with high probability.

Theorem 4. *Fix some $m < N$. Then for any $S \subseteq U$ with $|S| \leq m$ we can construct a perfect hash function taking value in $O(m)$ bins representable in $O(m \log N)$ bits with a Las Vegas algorithm that runs in expected time $O(m)$.*

We first need a technical lemma.

Lemma 5. *Fix some $m < N$. Let $\mathcal{H}$ be a 2-universal hash family taking value in n bins, and let $h \sim \mathcal{H}$. Let $S \subseteq U$ be fixed with $|S| \leq m$. Let X be a random variable representing the number of collisions, i.e.*

$$X = \frac{1}{2} \sum_{x, y \in S, x \neq y} C_{x,y}$$

where

$$C_{x,y} = \begin{cases} 1 & \text{if } h(x) = h(y); \\ 0 & \text{otherwise.} \end{cases}$$

(The $1/2$ term in the definition of X is to simply account for double counting; note that the maximum X can take is $m(m-1)/2$.) Then

$$\Pr\left[X \geq \frac{m^2}{n}\right] \leq \frac{1}{2}.$$

Proof By linearity of expectation

$$E(X) = \frac{1}{2} \sum_{x, y \in S, x \neq y} E(C_{x,y}) = \frac{1}{2} \sum_{x, y \in S, x \neq y} \Pr(h(x) = h(y)) = \frac{1}{2} \sum_{x, y \in S, x \neq y} \frac{1}{n}$$
$$\leq \frac{1}{n} \left(\frac{m(m-1)}{2}\right) \leq \frac{m^2}{2n}$$

so by Markov's inequality

$$\Pr\left(X \geq \frac{m^2}{n}\right) = \Pr(X \geq 2E(X)) \leq \frac{1}{2}$$

as claimed. ■

Proof of Theorem 4 Let X be defined as above, and for all $1 \leq k \leq N$ let $\mathcal{H}_k$ be the 2-universal hash family taking value in k bins as guaranteed by Theorem 2. By the lemma, notice that if we choose $n = m^2$

then $\Pr(X < 1) \geq \frac{1}{2}$. As X is integer-valued, when $X < 1$ there are no collisions. Thus when $n = m^2$ if we choose hash functions uniformly at random from the family defined in Theorem 2 then in an expected 2 trials we will find a perfect hash for any $S \subseteq U$ with $|S| \leq m$. This is because if we let X_i be the random variable which is number of collisions in the ith trial, then the X_i are i.i.d. and

$$E(\#\text{ of trials}) = \Pr(X_1 < 1) + \Pr(X_1 \geq 1)\Pr(X_2 < 1) + \ldots = \sum_{k=1}^{\infty} \frac{k}{2^k} = 2.$$

This allows us to find perfect hash functions quickly. However, we needed to set $n = m^2$, while we want $n = O(m)$. Thus, we will use a tiered hash function.

Choose $n = m$. Then we have $\Pr(X < m) \geq \frac{1}{2}$. Randomly choose hash functions from $\mathcal{H}$ until we find an $\tilde{h}$ with $X(\tilde{h}) < m$, which takes two trials in expectation by the same argument as above. Let c_i be the number of elements in bin i, for $i \in \{0, \ldots, n-1\}$. For each bin i with more than one element, we generate a perfect hash function $h_i : \{x : \tilde{h}(x) = i\} \to \{0...c_i^2 - 1\}$from the elements in that bin to bins in a new hash table. Since the number of bins is equal to the square of the number of elements to hash, by our previous argument, we can quickly (with an expected two trials) find such a perfect hash function. The total number of collisions then is exactly $\sum_{i=0}^{n-1} \binom{c_i}{2} = X(\tilde{h}) < m$ by our choice of $\tilde{h}$. The total number of bins for hash tables in this second tier will be $\sum_{i=0}^{n-1} c_i^2$, and the total number of bins overall will be:

$$m + \sum_{i=0}^{n-1} c_i^2 \leq m + 2\sum_{i=0}^{n-1} \binom{c_i}{2} + \sum_{i=0}^{n-1} c_i \leq m + 2m + m = 4m = O(m)$$

Our perfect hash function will be given as follows: given $x \in U$, first hash it by $\tilde{h}(x)$ to get a bucket i. If there are conflicts, we then hash it again to get another bucket j, and we hash it to the jth bucket in the secondary hash at bucket i. More rigorously, we let $h(x) = (\tilde{h}(x), h_{\tilde{h}(x)}(x))$, where bucket (i, j) indicates the jth bucket in the secondary hash of i, and we will define $h_i(x) = 0$ if h_i was not defined (i.e. there were no conflicts at bucket i in the primary hash). While this technically does not map into a set of the form $\{0, \ldots, k-1\}$, by the argument above h can take at most $O(m)$ distinct values, and thus by a trivial isomorphism we can modify h to be of the desired form, so it suffices to show that h satisfies the remaining conditions. Then h is a perfect hash function, so it achieves constant lookup time, since we can determine if an element is in the table in at most two lookups, and our total number of bins is $O(m)$ as desired. Furthermore, it is clear that h can be represented uniquely by identifying $\tilde{h}$ then h_i for $0 \leq i \leq n-1$. By our construction, this implies that h can be represented uniquely by $O(m \log N)$ bits.

It now suffices to check that this procedure runs in expected time $O(m)$. Notice that checking a hash function has at most k collisions for S takes $O(m)$ steps as it suffices to hash all the elements and keep track of the number of collisions we encounter. Thus, finding $\tilde{h}$ takes expected time $O(m)$, and finding each h_i takes expected time $O(c_i)$. Therefore finding h takes expected time $O(m)$ as $\sum_{i=0}^{m} c_i \leq m$. ■