

CSL 857: Randomized Algorithms

Ragesh Jaiswal
CSE, IIT Delhi

Hash Functions

Hash Functions: Universal Hashing

- Hashing: A set S of keys from a large universe $U = \{0, \dots, m - 1\}$ is stored in a table $T = \{0, \dots, n - 1\}$ using a hash function $h: U \rightarrow T$ so as to minimize the number of collisions. Collision being resolved using auxiliary data structure.
- If $m > n$, then any deterministic h is bad.
- Main idea: Choose h randomly from a hash function family H .
- 2-universal: A hash function family H is called 2-universal if

$$\forall x, y \in U, x \neq y, \Pr_{h \leftarrow H}[h(x) = h(y)] \leq \frac{1}{n}.$$

Hash Functions: Universal Hashing

- Claim: Consider hashing using a 2-universal hash function family. Consider t insert operations, the expected cost of each operation is at most $(1 + t/n)$.
- Proof: Consider any key x , the expected number of keys in location $h(x)$ is at most t/n .
- Question: Can you think of a 2-universal hash function family.

Hash Functions: Universal Hashing

- Claim: Consider hashing using a 2-universal hash function family. Consider t insert operations, the expected cost of each operation is at most $(1 + t/n)$.
- Proof: Consider any key x , the expected number of keys in location $h(x)$ is at most t/n .
- Question: Can you think of a 2-universal hash function family.
 - Simple answer: Consider the family containing all function from U to T .
 - Issue:

Hash Functions: Universal Hashing

- Claim: Consider hashing using a 2-universal hash function family. Consider t insert operations, the expected cost of each operation is at most $(1 + t/n)$.
- Proof: Consider any key x , the expected number of keys in location $h(x)$ is at most t/n .
- Question: Can you think of a 2-universal hash function family.
 - Simple answer: Consider the family containing all function from U to T .
 - Issue: The description of such a function family is huge.

Hash Functions: Universal Hashing

- A compact 2-universal family.
 - Let $m \leq p \leq 2m$, be a prime number.
 - $H = \{h_{a,b} \mid a \in \{1, \dots, p-1\}, b \in \{0, \dots, p-1\}\}$ and $h_{a,b}(x) = ((ax + b) \bmod p) \bmod n$.
 - H contains $p(p-1)$ functions. So, the description of a function is small.
- Claim 2: H is 2-universal.
- Proof:
 - Let $g_{a,b}(x) = (ax + b) \bmod p$.
 - Claim 2.1: $\forall \alpha, \beta, \Pr[g_{a,b}(x) = \alpha \text{ and } g_{a,b}(y) = \beta] = 1/p(p-1)$ if $\alpha \neq \beta$ and 0 otherwise.

Hash Functions: Universal Hashing

- A compact 2-universal family.
 - Let $m \leq p \leq 2m$, be a prime number.
 - $H = \{h_{a,b} \mid a \in \{1, \dots, p-1\}, b \in \{0, \dots, p-1\}\}$ and $h_{a,b}(x) = ((ax + b) \bmod p) \bmod n$.
 - H contains $p(p-1)$ functions. So, the description of a function is small.
- Claim 2: H is 2-universal.
- Proof:
 - Let $g_{a,b}(x) = (ax + b) \bmod p$.
 - Claim 2.1: $\forall \alpha, \beta, \Pr[g_{a,b}(x) = \alpha \text{ and } g_{a,b}(y) = \beta] = 1/p(p-1)$ if $\alpha \neq \beta$ and 0 otherwise.
 - Claim 2.2: $\forall x \neq y, \Pr[h_{a,b}(x) = h_{a,b}(y)] \leq \frac{1}{n}$.

Hash Functions: Universal Hashing

- Consider hashing using 2-universal family of hash functions. For any sequence of S operations with at most $S \leq n$ inserts, the expected cost of each operation is at most 2. What about the cost of the worst operation?
- Consider the simple scenario where there are no delete operations and all finds happen after all inserts. The goal is to hash with no collisions.
- Double hashing:
 - Use a *primary* hash function to all keys to map them to their respective buckets in the primary table.
 - Use a *secondary* hash function to all keys in the same bucket of the primary table to map it to buckets in the secondary table.

Hash Functions: Universal Hashing

- Double hashing:
 - Use a *primary* hash function to all keys to map them to their respective buckets in the primary table.
 - Use a *secondary* hash function to all keys in the same bucket of the primary table to map it to buckets in the secondary table.
 - The primary table entry holds the secondary hash function that was used for this bucket.
- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table to size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.

Hash Functions: Universal Hashing

- Double hashing:
 - Use a *primary* hash function to all keys to map them to their respective buckets in the primary table.
 - Use a *secondary* hash function to all keys in the same bucket of the primary table to map it to buckets in the secondary table.
 - The primary table entry holds the secondary hash function that was used for this bucket.
- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table to size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.
- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each buckets, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.

Hash Functions: Universal Hashing

- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table of size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.

Hash Functions: Universal Hashing

- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table of size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.
- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each bucket, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.

Hash Functions: Universal Hashing

- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table of size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.
- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each bucket, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.
- For every i , with probability at least $1/2$ a randomly chosen hash function from a 2-universal family gives no collisions.

Hash Functions: Universal Hashing

- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table of size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.
- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each bucket, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.
- For every i , with probability at least $1/2$ a randomly chosen hash function from a 2-universal family gives no collisions.
- For each i , we need to repeat constant number of times to find a hash function h_i with no collisions.

Hash Functions: Universal Hashing

- Claim: Suppose we use a 2-universal family of hash functions to hash a set of b keys into a table of size b^2 , then $\Pr[\textit{there is a collision}] \leq 1/2$.
- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each bucket, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.
- For every i , with probability at least $1/2$ a randomly chosen hash function from a 2-universal family gives no collisions.
- For each i , we need to repeat constant number of times to find a hash function h_i with no collisions.
- We store the description of h_i in the primary table.

Hash Functions: Universal Hashing

- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each buckets, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.
- For every i , with probability at least $1/2$ a randomly chosen hash function from a 2-universal family gives no collisions.
- For each i , we need to repeat constant number of times to find a hash function h_i with no collisions.
- We store the description of h_i in the primary table.

Hash Functions: Universal Hashing

- Suppose the primary hash function creates buckets of size $b_1, \dots, b_r$. For each buckets, we pick a 2-universal hash function that hashes to buckets of size $(b_i)^2$.
- For every i , with probability at least $1/2$ a randomly chosen hash function from a 2-universal family gives no collisions.
- For each i , we need to repeat constant number of times to find a hash function h_i with no collisions.
- We store the description of h_i in the primary table.
- The total size of the secondary tables is $\sum (b_i)^2$. One requirement of hashing is that the size of the table used should not exceed the number of elements $|S|$ hashed by more than a constant factor.

Hash Functions: Universal Hashing

- The total size of the secondary tables is $\sum (b_i)^2$. One requirement of hashing is that the size of the table used should not exceed the number of elements $|S|$ hashed by more than a constant factor.

Hash Functions: Universal Hashing

- The total size of the secondary tables is $\sum (b_i)^2$. One requirement of hashing is that the size of the table used should not exceed the number of elements $|S|$ hashed by more than a constant factor.
- Claim: If we use a 2-universal hash function family to hash a set of S keys into a table of size $n \geq |S|$, then the bucket sizes b_i satisfy $\Pr[\sum_i (b_i)^2 \geq 4|S|] \leq 1/2$.

Hash Functions: Universal Hashing

- The total size of the secondary tables is $\sum (b_i)^2$. One requirement of hashing is that the size of the table used should not exceed the number of elements $|S|$ hashed by more than a constant factor.
- Claim: If we use a 2-universal hash function family to hash a set of S keys into a table of size $n \geq |S|$, then the bucket sizes b_i satisfy $\Pr[\sum_i (b_i)^2 \geq 4|S|] \leq 1/2$.
- We repeatedly pick a primary hash function randomly from a 2-universal family and check if $\sum_i (b_i)^2 \leq 4|S|$. From above claim, we will only need to repeat a constant number of times.

End
