

Randomized algorithms II

Inge Li Gørtz

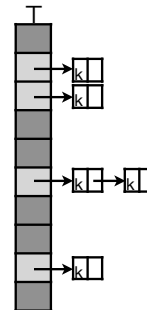
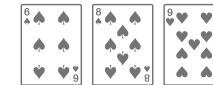
Randomized algorithms

- Last week

- Contention resolution
- Global minimum cut

- Today

- Expectation of random variables
 - Guessing cards
- Hash functions and hash tables



Random Variables and Expectation

Random variables

- A **random variable** is an entity that can assume different values.
- The values are selected “randomly”; i.e., the process is governed by a probability distribution.
- **Examples:** Let X be the random variable “number shown by dice”.
 - X can take the values 1, 2, 3, 4, 5, 6.
 - If it is a fair dice then the probability that $X = 1$ is $1/6$:
 - $\Pr[X=1] = 1/6$.
 - $\Pr[X=2] = 1/6$.
 - ...

Expected values

- Let X be a random variable with values in $\{x_1, \dots, x_n\}$, where x_i are numbers.
- The **expected value** (expectation) of X is defined as

$$E[X] = \sum_{j=1}^n x_j \cdot \Pr[X = x_j]$$

- The expectation is the theoretical average.
- Example:
 - X = random variable “number shown by dice”

$$E[X] = \sum_{j=1}^6 j \cdot \Pr[X = j] = (1 + 2 + 3 + 4 + 5 + 6) \cdot \frac{1}{6} = 3.5$$

Waiting for a first succes

- **Coin flips.** Coin is heads with probability p and tails with probability $1 - p$. How many independent flips X until first heads?

- Probability of $X = j$? (first succes is in round j)

$$\Pr[X = j] = (1 - p)^{j-1} \cdot p$$

- Expected value of X :

$$\begin{aligned} E[X] &= \sum_{j=1}^{\infty} j \cdot \Pr[X = j] = \sum_{j=1}^{\infty} j \cdot (1 - p)^{j-1} \cdot p = \frac{p}{1 - p} \sum_{j=1}^{\infty} j \cdot (1 - p)^j \\ &= \frac{p}{1 - p} \cdot \frac{1 - p}{p^2} = \frac{1}{p} \end{aligned}$$

$$\sum_{k=0}^{\infty} k \cdot x^k = \frac{x}{(1 - x)^2} \quad \text{for } x < 1.$$

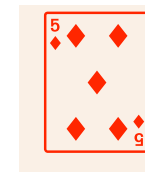
Properties of expectation

- If we repeatedly perform independent trials of an experiment, each of which succeeds with probability $p > 0$, then the expected number of trials we need to perform until the first success is $1/p$.
- If X is a 0/1 random variable, then $E[X] = \Pr[X = 1]$.
- **Linearity of expectation:** For two random variables X and Y we have

$$E[X + Y] = E[X] + E[Y]$$

Guessing cards

- **Game.** Shuffle a deck of n cards; turn them over one at a time; try to guess each card.
- **Memoryless guessing.** Can't remember what's been turned over already. Guess a card from full deck uniformly at random.



- **Claim.** *The expected number of correct guesses is 1.*

- $X_i = 1$ if i^{th} guess correct and zero otherwise.

- $X =$ the correct number of guesses $= X_1 + \dots + X_n$.

- $E[X_i] = \Pr[X_i = 1] = 1/n$.

- $E[X] = E[X_1 + \dots + X_n] = E[X_1] + \dots + E[X_n] = 1/n + \dots + 1/n = 1$.



Guessing cards

- **Game.** Shuffle a deck of n cards; turn them over one at a time; try to guess each card.
- **Guessing with memory.** Guess a card uniformly at random from cards not yet seen.
- **Claim.** *The expected number of correct guesses is $\Theta(\log n)$.*
 - $X_i = 1$ if i^{th} guess correct and zero otherwise.
 - $X =$ the correct number of guesses $= X_1 + \dots + X_n$.
 - $E[X_i] = \Pr[X_i = 1] = 1/(n - i + 1)$.
 - $E[X] = E[X_1] + \dots + E[X_n] = 1/n + \dots + 1/2 + 1/1 = H_n$.

$$\ln n < H(n) < \ln n + 1$$

Coupon collector

- **Coupon collector.** Each box of cereal contains a coupon. There are n different types of coupons. Assuming all boxes are equally likely to contain each coupon, how many boxes before you have at least 1 coupon of each type?
- **Claim.** *The expected number of steps is $\Theta(n \log n)$.*
 - Phase j = time between j and $j + 1$ distinct coupons.
 - X_j = number of steps you spend in phase j .
 - X = number of steps in total = $X_0 + X_1 + \dots + X_{n-1}$.
 - $E[X_j] = n/(n - j)$.
 - The expected number of steps:

$$E[X] = E\left[\sum_{j=0}^{n-1} X_j\right] = \sum_{j=0}^{n-1} E[X_j] = \sum_{j=0}^{n-1} n/(n - j) = n \cdot \sum_{i=1}^n 1/i = n \cdot H_n.$$

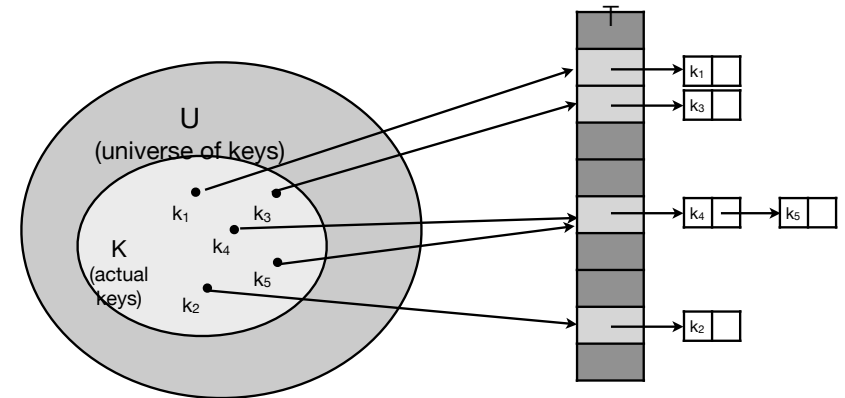
Hashing

Dictionaries

- **Dictionary problem.** Maintain a dynamic set of $S \subseteq U$ subject to the following operations:
 - **Lookup(x):** return true if $x \in S$ and false otherwise
 - **Insert(x):** Set $S = S \cup \{x\}$
 - **Delete(x):** Set $S = S \setminus \{x\}$
- **Universe size.** Typically $|U| = 2^{64}$ and $|S| \ll |U|$.
- **Satellite information.** Information associated with each element.
- **Goal.** A compact data structure with fast operations.
- **Applications.** Many! A key component in other data structures and algorithms.

Chained Hashing

- **Chained hashing** [Dumey 1956].
 - $n = |S|$.
 - **Hash function.** Pick some crazy, chaotic, random function h that maps U to $\{0, \dots, m-1\}$, where $m = \Theta(n)$.
 - Initialise an array $A[0, \dots, m-1]$.
 - $A[i]$ stores a linked list containing the keys in S whose hash value is i .



Uniform random hash functions

- *E.g.* $h(x) = x \bmod 11$. Not crazy, chaotic, random.
- **Suppose** $|U| \geq n^2$: For any hash function h there will be a set S of n elements that all map to the same position!
 - ⇒ we end up with a single linked list.
- **Solution:** randomization.
 - For every element $u \in U$: select $h(u)$ uniformly at random in $\{0, \dots, m-1\}$ independently from all other choices.
- **Claim.** *The probability that $h(u) = h(v)$ for two elements $u \neq v$ is $1/m$.*
- **Proof.**
 - m^2 possible choices for the pair of values $(h(u), h(v))$. All equally likely.
 - Exactly m of these gives a collision.

Chained Hashing with Random Hash Function

- Expected length of the linked list for $h(x)$?

- Random variable $L_x =$ length of linked list for x . $L_x = \{y \in S \mid h(y) = h(x)\}$

- Indikator random variable:

$$I_y = \begin{cases} 1 & \text{if } h(x) = h(y) \\ 0 & \text{otherwise} \end{cases} \quad L_x = \sum_{y \in S} I_y \quad E[I_y] = \Pr[h(y) = h(x)] = \frac{1}{m} \text{ for } x \neq y.$$

- The expected length of the linked list for x :

$$E[L_x] = E \left[\sum_{y \in S} I_y \right] = \sum_{y \in S} E[I_y] = 1 + \sum_{y \in S \setminus \{x\}} \frac{1}{m} = 1 + (n - 1) \cdot \frac{1}{m} = \Theta(1).$$

Chained Hashing with Random Hash Function

- Constant time and $O(n)$ space for the hash table.
- But:
 - Need $O(|U|)$ space for the hash function.
 - Need a lot of random bits to generate the hash function.
 - Need a lot of time to generate the hash function.
- Do we need a truly random hash function?
- When did we use the fact that h was random in our analysis?

Chained Hashing with Random Hash Function

- Expected length of the linked list for $h(x)$?

- Random variable $L_x =$ length of linked list for x . $L_x = \{y \in S \mid h(y) = h(x)\}$

- Indikator random variable:

$$I_y = \begin{cases} 1 & \text{if } h(x) = h(y) \\ 0 & \text{otherwise} \end{cases} \quad L_x = \sum_{y \in S} I_y \quad E[I_y] = \Pr[h(y) = h(x)] = \frac{1}{m} \text{ for } x \neq y.$$

- The expected length of the linked list for x :

$$E[L_x] = E \left[\sum_{y \in S} I_y \right] = \sum_{y \in S} E[I_y] = 1 + \sum_{y \in S \setminus \{x\}} \frac{1}{m} = 1 + (n - 1) \cdot \frac{1}{m} = \Theta(1).$$

Universal hash functions

- Universal hashing [Carter and Wegman 1979].

- Let H be a family of functions mapping U to the set $\{0, \dots, m - 1\}$.
- H is universal if for any $x, y \in U$, where $x \neq y$, and h chosen uniformly at random in H ,

$$\Pr[h(x) = h(y)] \leq 1/m.$$

- Require that any $h \in H$ can be represented compactly and that we can compute the value $h(u)$ efficiently for any $u \in U$.

Universal Hashing

- **Positional number systems.** For integers x and b , the **base- r representation** of x is x written in base b .
- **Example.**
 - $(10)_{10} = (1010)_2 \quad (1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0)$
 - $(107)_{10} = (212)_7 \quad (2 \cdot 7^2 + 1 \cdot 7^1 + 2 \cdot 7^0)$

Universal Hashing

- **Hash function.** Given a prime p and $a = (a_1 a_2 \dots a_r)_p$, define

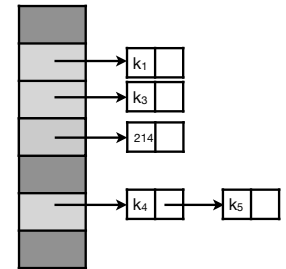
$$h_a((x_1 x_2 \dots x_r)_p) = a_1 x_1 + a_2 x_2 + \dots + a_r x_r \pmod{p}$$

- **Example.**

- $p = 7$
- $a = (107)_{10} = (212)_7$
- $x = (214)_{10} = (424)_7$
- $h_a(x) = 2 \cdot 4 + 1 \cdot 2 + 2 \cdot 4 \pmod{7} = 18 \pmod{7} = 4$

- **Universal family.**

- $H = \{h_a \mid (a_1 a_2 \dots a_r)_p \in \{0, \dots, p-1\}^r\}$
- Choose random hash function from $H \sim$ choose random a .
- H is universal (analysis next).
- $O(1)$ time evaluation.
- $O(1)$ space.
- Fast construction.



Uniform Hashing

- **Lemma 1.** For any prime p , any integer $z \not\equiv 0 \pmod{p}$, and any two integers α, β :

$$\alpha z = \beta z \pmod{p} \quad \Rightarrow \quad \alpha = \beta \pmod{p}.$$

- **Proof.**

- Show $(\alpha - \beta)$ is divisible by p :
 - $\alpha z = \beta z \pmod{p} \quad \Rightarrow \quad (\alpha - \beta)z = 0 \pmod{p}.$
 - By assumption z not divisible by p .
 - Since p is prime $\alpha - \beta$ must be divisible by p .
- Thus $\alpha = \beta \pmod{p}$ as claimed.

Universal Hashing

- **Goal.** For random $a = (a_1 a_2 \dots a_r)_p$, show that if $x \neq y$ then $\Pr[h_a(x) = h_a(y)] \leq 1/p$.

- Recall: $x = (x_1 x_2 \dots x_r)_p$ and $y = (y_1 y_2 \dots y_r)_p$:

$$x \neq y \Leftrightarrow (x_1 x_2 \dots x_r)_p \neq (y_1 y_2 \dots y_r)_p \Rightarrow x_j \neq y_j \text{ for some } j.$$

- **Lemma 2.** Let j be such that $x_j \neq y_j$. Assume the coordinates a_i have been chosen for all $i \neq j$. The probability of choosing a_j such that $h_a(x) = h_a(y)$ is $1/p$.

$$h_a(x) = h_a(y) \Leftrightarrow \sum_{i=1}^r a_i x_i \pmod p = \sum_{i=1}^r a_i y_i \pmod p \Leftrightarrow a_j (x_j - y_j) = \sum_{i \neq j} a_i (x_i - y_i) \pmod p = c$$

- *There is exactly one value $0 \leq a_j < p$ that satisfies $a_j z = c \pmod p$.*

fixed value $z \neq 0$

fixed value since all a_i fixed for $i \neq j$.

- Assume there was two such values a_j and a'_j .

- Then $a_j z = a'_j z \pmod p$.

- Lemma 1 $\Rightarrow a_j = a'_j \pmod p$. Since $a_j < p$ and $a'_j < p$ we have $a_j = a'_j$.

- Probability of choosing a_j such that $h_a(x) = h_a(y)$ is $1/p$.

Universal Hashing

- **Lemma 2.** Let j be such that $x_j \neq y_j$. Assume the coordinates a_i have been chosen for all $i \neq j$. The probability of choosing a_j such that $h_a(x) = h_a(y)$ is $1/p$.

- **Theorem.** For random $a = (a_1 a_2 \dots a_r)_p$, if $x \neq y$ then

$$\Pr[h_a(x) = h_a(y)] = 1/p.$$

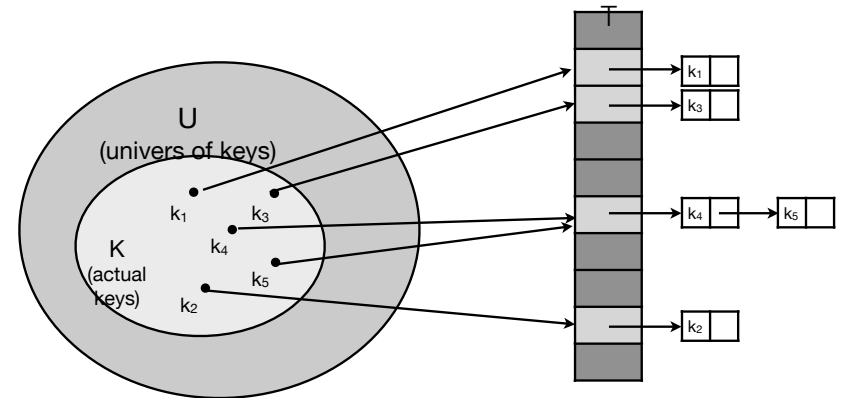
- **Proof.**

- E : the event that $h_a(x) = h_a(y)$.
- F_b : the event that the values a_i for $i \neq j$ gets the sequence of values b .
- Lemma 2 shows that $\Pr[E | F_b] = 1/p$ for all b .
- Thus

$$\Pr[E] = \sum_b \Pr[E | F_b] \cdot \Pr[F_b] = \sum_b \frac{1}{p} \cdot \Pr[F_b] = \frac{1}{p} \sum_b \Pr[F_b] = \frac{1}{p}$$

Dictionaries

- **Theorem.** We can solve the dictionary problem (without special assumptions) in:
 - $O(n)$ space.
 - $O(1)$ expected time per operation (lookup, insert, delete).



Universal Hashing

- Other universal families.

- For prime $p > 0$.

$$h_{a,b}(x) = ax \pmod{p}$$

$$H = \{h_{a,b} \mid a \in \{1, \dots, p-1\}, b \in \{0, \dots, p-1\}\}.$$

- Hash function from k -bit numbers to l -bit numbers.

$$h_a(x) = (ax \pmod{2^k}) \gg (k-l)$$

$$H = \{h_a \mid a \text{ is an odd integer in } \{1, \dots, 2^k - 1\}\}$$