CS 4800: Algorithms & Data

Lecture 23

April 13, 2018

Hashing

Birthday paradox

- Pr[2nd person has different birthday from 1st person]
  - $1 - \frac{1}{365}$
- Pr[3rd person has different birthday from first two people, provided that first two people have different birthdays]
  - $1 - \frac{2}{365}$
- Probability first k people have different birthdays is product of these terms
  - $\left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right) \dots \left(1 - \frac{k-1}{365}\right)$

Birthday paradox

- Probability first k people have different birthdays is product of these terms
  - $\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\ldots\left(1-\frac{k-1}{365}\right)$
- How large does k need to be for prob. < ½ ?
  - 23

Balls into bins

- n random birthdays among 365 choices
- n balls are thrown into m bins
- What is the distribution of the loads?
- Birthday paradox: what is minimum n so that the probability some bin has at least 2 balls is > ½?

Balls into bins

- n balls are thrown into m bins
- Expected number of empty bins?
- What is probability first bin is empty?
- Ball 1 misses bin 1 with probability $1 - 1/m$
- Probably n balls all miss bin 1 is $\left(1 - \frac{1}{m}\right)^n$
  - $\approx e^{-n/m}$

Hashing

- Assign a number to an object via a hash function
$$h: S \to \{0, 1, 2, \dots, m-1\}$$
- Make comparison easy
- Object u = object v only if $h(u) = h(v)$
- Downside: $h(u) = h(v)$ for some $u \neq v$ (collision)
- Idea: pick h randomly so that for any $u \neq v$, the chance $h(u) = h(v)$ is low
- Idealized: for all u and i, $\Pr[h(u) = i] = \frac{1}{m}$

Question

- Hash n objects to numbers in $\{0, 1, 2, \dots, m-1\}$
- How large should m be so that we expect less than 1 collision?

Password checker

- User picks a password
- Want to check if password is a common word
- Dictionary of n common words

Checker using hash function

- Use an array of m bits
- All bits are initialized to 0
- Hash every word w in dictionary
  - If hash(w)=i then set bit i of array to 1
- On query:
  - j=hash(password)
  - If bit j is 1, reject password

Checker using hash function

- On query:
  - j=hash(password)
  - If bit j is 1, reject password
- If password is common word, Pr[reject] = 1
- If password is not common,
  - $\Pr[accept] = \Pr[hash(w) \neq j \; for \; all \; common \; w]$

$$= \Pr[bin \; j \; is \; empty \; after \; n \; throws]$$

$$= (1 - 1/m)^n \approx \exp(-n/m)$$

Checker using hash function

- If password is not common,
  - $\Pr[accept] = \exp\left(-\frac{n}{m}\right)$
- Example, n=100000 common words
- m=1000000 bits
- $\Pr[accept] = 90\%$

Bloom filter

- t hash functions $h_1, h_2, \ldots, h_t$
- t bit arrays of size m/t each
- All bits initialized to 0
- Hash every word w in dictionary
  - If $h_3(w) = i$ then set bit i in array 3 to 1
  - Same for other tables
- On query q:
  - $j_1 = h_1(q), j_2 = h_2(q), \ldots$
  - If bit j_1 of array 1 is 0, accept password
  - If bit j_2 of array 2 is 0, accept password
  - …
  - If all those bits are 1, reject password

Bloom filter

- On query q:
  - $j_1 = h_1(q), j_2 = h_2(q), \ldots$
  - If bit j_1 of array 1 is 0, accept password
  - If bit j_2 of array 2 is 0, accept password
  - …
  - If all those bits are 1, reject password
- If password is common word, Pr[reject] = 1
- If password is not common,
  - $\Pr[reject] = \Pr[all\ arrays\ fail]$

$$= (\Pr[array\ 1\ fails])^t$$

$$= (1 - (1 - t/m)^n)^t$$

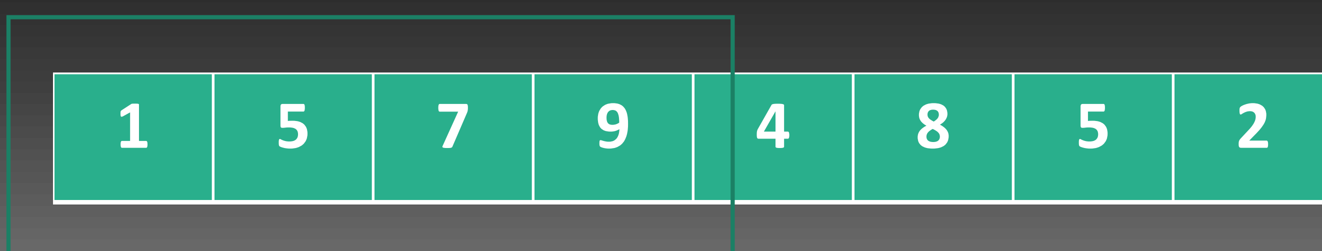
Bloom filter

- If password is not common,
  - $\Pr[reject] == (1 - (1 - t/m)^n)^t$
- Example, n=100000 common words
- m=500000 bits
- t=5 tables
- $\Pr[accept] = 90\%$

- m=1000000 bits
- *t=1*
- $\Pr[accept] = 90\%$

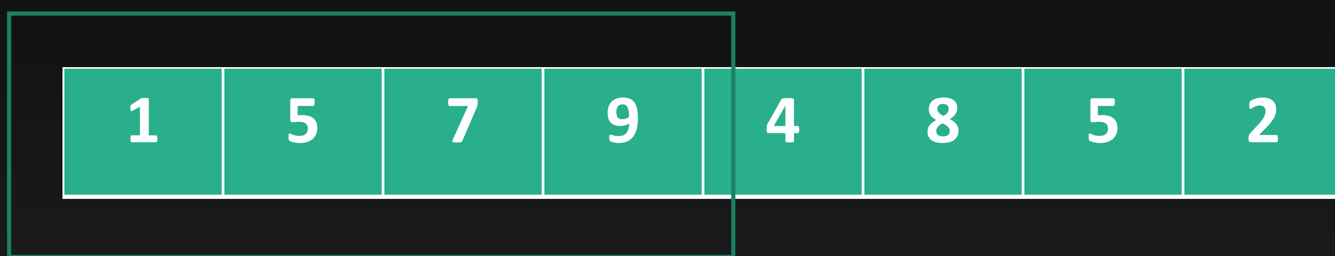
String matching

- Given a text T and a pattern P
- Find in the text T all occurrences of P
- Idea: view each character as a digit
- T is a long sequence of digits
- P is a |P|-digit number
- Each |P| consecutive characters in T form a |P|-digit number
- Want to compare these numbers against P



Streaming characters

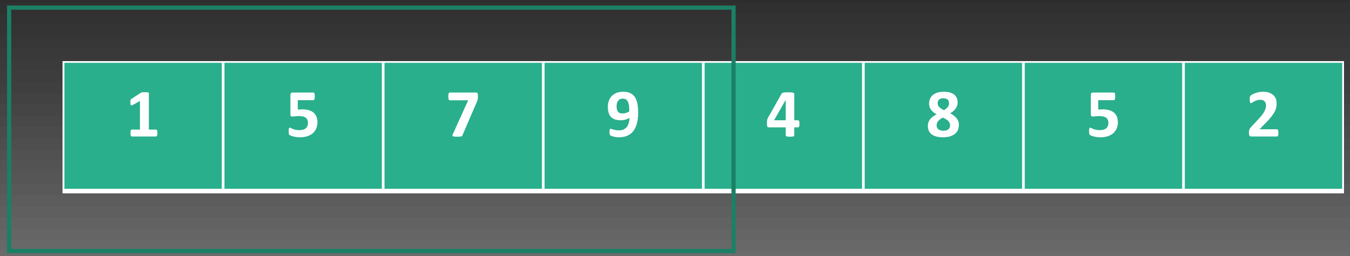
- Maintain the number formed by latest |P| characters of the text



- Slide the window one character at a time
- Need to update original number N to form new N’

Streaming characters

- $1579 \rightarrow 5794$
- Delete first digit a, multiply by 10, add last digit b
- $N' = 10\left(N - 10^{|P|-1}a\right) + b$
- Slide window from left to right, in every step
  - Form N’ from current N
  - Compare N’ with pattern P
- Time: O(T)
- N might be too large to fit in an int



Rabin-Karp/rolling hash

- Pick a prime p
- h(N) = N mod p
- Instead of keeping track of N, only keep h(N)

1	**5**	**7**	**9**	**4**	**8**	**5**	**2**

$$
\begin{aligned}
h(N') &= \left(10\left(N - 10^{|P|-1}a\right) + b\right) \mathit{mod}\ p \\
&= \left(10\left((N\ \mathit{mod}\ p) - \left(10^{|P|-1}\ \mathit{mod}\ p\right)a\right) + b\right)\ \mathit{mod}\ p
\end{aligned}
$$