CS7800: Advanced Algorithms

Lecture 18: Randomized III

- String Matching
- Start Hashing

Jonathan Ullman

11-15-2022

String Matching

Input: text $T[1:n] \in \Sigma^*$ ← e.g. $\{A, B, C, \ldots, Z\}$ or $\{0,1\}$

pattern $P[1:m] \in \Sigma^*$

Output: either s such that $T[s:s+m-1] = P[1:m]$

or "none" if there is no match

$$
\begin{array}{lllllllllll}
 & & 1 & 2 & 3 & 4 & & & & & \\
T & = & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
P & = & 0 & 1 & 0 & & & & & &
\end{array}
$$

String Matching Algorithm I

$T = 11001001$

$P = 010$

$O(n)$ iterations

for $s = 1, 2, \ldots, n-m+1$:

    if $(T[s : s+m-1] = P[1 : m])$:

        return s

return "none"

equality check takes $O(m)$ time

total time is $O(nm)$

and can be $\Omega(nm)$

$T =$ AAAA ... A

$P =$ AAA...AB

String Matching Algorithm II

$T = 11001001$

$P = 010$

$t_s = T[s:s+m-1]$ as a number

$p = P[1:m]$ as a number

compute p using

compute s_1 using

// $O(m)$ time

for $s = 1, 2, \dots, n-m+1$:

  if ($t_s == p$):

    return s

return "none"

Strings to Numbers

1	1	0	1
P[1]	P[2]	P[3]	P[4]

$$p = 13 = 2^3 \cdot P[1] + 2^2 \cdot P[2] + 2^1 \cdot P[3] + 2^0 \cdot P[4]$$

$$p = (((P[1]) \times 2 + P[2]) \times 2 + P[3]) \times 2 + P[4]$$

String Matching Algorithm II

T = 11001001

P = 010

t_s = T[s:s+m-1] as a number

p = P[1:m] as a number

compute p // O(m) time

compute t_1

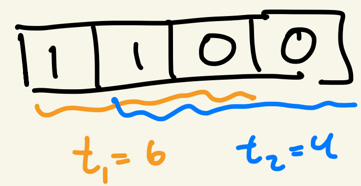
for s = 1, 2, ..., n-m+1 :

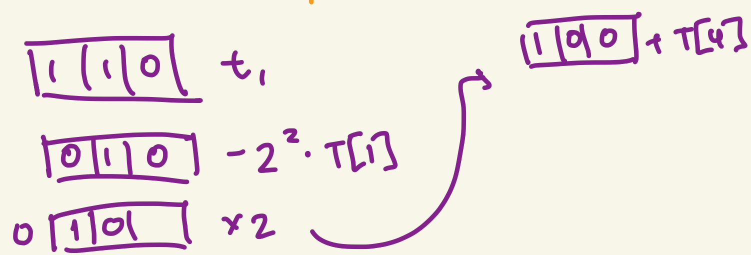
- if (t_s == p): O(m) time per iteration
  - return s
- $t_{s+1} = 2 \cdot (t_s - 2^{m-1} \cdot T[s]) + T[s+m]$

return "none"

No improvement yet

Sliding Window





Modular Arithmetic

- $x \bmod y$: "the remainder when you divide x by y"

  - $4 \bmod 3 = 1$
    $7 \bmod 2 = 1$
    $13 \bmod 5 = 3$

  - $x \bmod y = b$ means
    $x = ay + b$ for integer a, b

- If $x = x'$ then $(x \bmod y) = (x' \bmod y)$ for every y

String Matching Algorithm III

$T = 11001001$

$P = 010$

choose z ???

compute p mod z
compute t_1 mod z
$\sigma =$ compute 2^{m-1} mod z

$\Big\}$ time $O(m)$

for $s = 1, 2, \ldots, n-m+1$:

- if ($t_s == p$ mod z) $\Big]$ time $O(\log z)$
  - check if $t_s == p$ and if so, return s $\Big]$ time $O(m)$ if we have to do it
- $t_{s+1} = 2 \cdot (t_s - \sigma \cdot T[s]) + T[s+m]$ mod z $\Big]$ time $O(\log z)$

return "none"

$O(n \log z + m \cdot (\#\text{of matches}))$

Random Prime Numbers

① (Prime Number Theorem) The number of primes $\leq u$ is $\Theta\left(\frac{u}{\log u}\right)$

Difficult to prove

② Every integer u has at most $\log_2 u$ distinct prime factors

$$u = p_1^{a_1} \cdot p_2^{a_2} \cdot \ldots \cdot p_k^{a_k} \geq 2^k$$

$$\Rightarrow \quad k \leq \log_2 u$$

③ There is an efficient randomized algorithm to test primality

Difficult to prove

String Matching Algorithm III

$T = 11001001$

$P = 010$

$u \approx m^2 \log m$

choose z to be a random prime in $\{2, 3, ..., u\}$

think about running time of this step

compute $p \bmod z$ ← z has $O(\log m)$ digits

compute $t_1 \bmod z$

σ = compute $2^{m-1} \bmod z$

for $s = 1, 2, ..., n-m+1$:

- if ($t_s == p \bmod z$)
  - check if $t_s == p$ and if so, return s
- $t_{s+1} = 2 \cdot (t_s - \sigma \cdot T[s]) + T[s+m] \bmod z$

return "none"

$\frac{n}{m}$ in expectation

Time:

$O(n \log m + m \cdot \frac{n}{m})$

- $P(t_s == p \bmod z)$ when z is a random prime in $\{2, 3, ..., u\}$
- $t_s - p \geq 0$ is an m bit number
- $(t_s - p)$ has $\leq m$ prime factors
- $P(z$ is a factor of $(t_s - p))$

$\leq \frac{m}{\left(\frac{u}{\log u}\right)} \leq \frac{1}{m}$

- if $u = m^2 \log m$

Hash Tables

universe

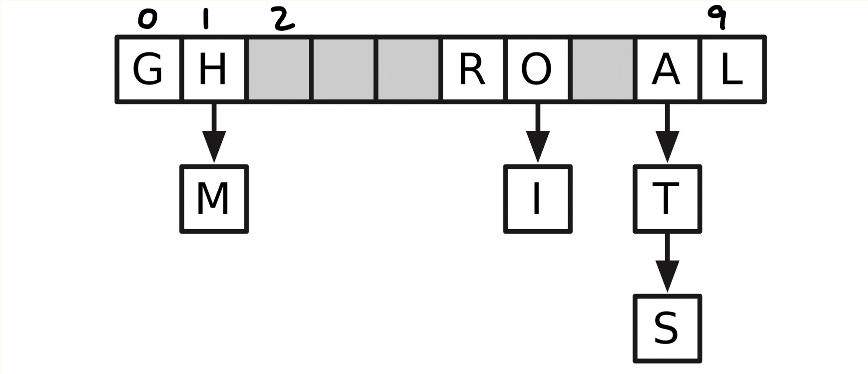
<u>Goal:</u> Store a set of n elements $S \subseteq U$ so that we can look up whether $x \in U$ is in S

- A dictionary also let's us associate a value with keys x

- A <u>hash table</u> $T[1:m]$ stores the elements
- A <u>hash function</u> $h: U \to \{0, 1, \dots, m-1\}$ maps elements to slots $x \to T[h(x)]$

Linear Chaining

A method for dealing with hash collisions



load factor n/m $\left[\begin{array}{l} m = 10 \\ n = 10 \end{array}\right.$

$U = \{A, B, C, \ldots, Z\}$

$h(G) = 0$
$h(M) = 1$
$h(A) = h(T) = 8$

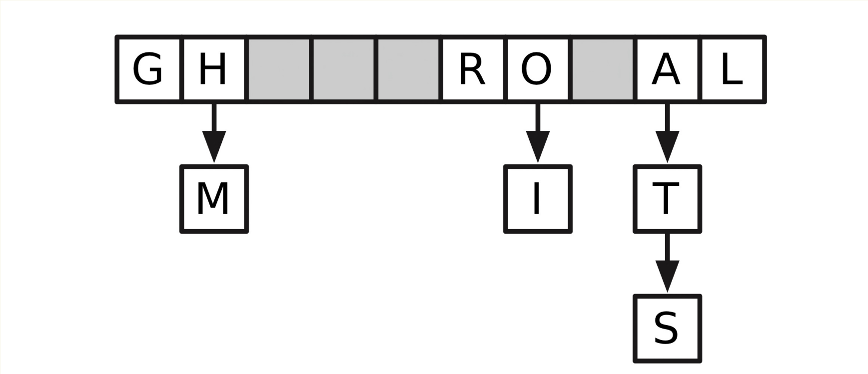
- Let $\ell(x)$ be the number of elements $y \in S$ such that $h(x) = h(y)$
- Time to lookup x is $O(\ell(x))$

Randomized Hash Functions

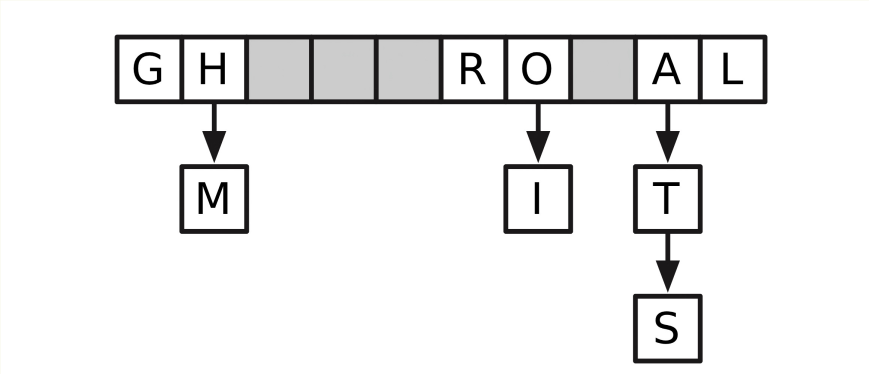
- A <u>hash family</u> $\mathcal{H} = \{ h : \mathcal{U} \to \{0, 1, \ldots, m-1\} \}$
- Choose a hash function h uniformly at random from $\mathcal{H}$.

- If $|\mathcal{H}| = 1$ (h is deterministic) then there is always a set of size $|\mathcal{U}|/m$ that all hash to the same bucket
- A uniformly random function is a "good hash family"
- There simple hash families that "good enough"

Linear Chaining with Ideal Hashing



Linear Chaining with Universal Hashing



Constructing Universal Hash Families

Fix some prime $p > |U|$ and table size m

$$\boxed{\begin{array}{l} h_{a,b}(x) = (ax + b \bmod p) \bmod m \\ \mathcal{H}_{p,m} = \{ h_{a,b} \text{ for } a \in \mathbb{Z}_p^+, b \in \mathbb{Z}_p \} \end{array}}$$

<u>Thm:</u> $\mathcal{H}_{p,m}$ is a universal hash family

Constructing Universal Hash Families

Thm: $\mathcal{H}_{p,m}$ is a universal hash family

$$h_{a,b}(x) = (ax + b \bmod p) \bmod m$$

$$\mathcal{H}_{p,m} = \{ h_{a,b} \text{ for } a \in \mathbb{Z}_p^+, b \in \mathbb{Z}_p \}$$

Lemma 1: If p is prime and $a \neq 0$ then there is a unique value $a^{-1} \in \{1, \ldots, p-1\}$ such that $a \cdot a^{-1} = 1 \bmod p$

(Division mod p is well defined)

Constructing Universal Hash Families

<u>Thm:</u> $\mathcal{H}_{p,m}$ is a universal hash family

$$h_{a,b}(x) = (ax + b \bmod p) \bmod m$$

$$\mathcal{H}_{p,m} = \{ h_{a,b} \text{ for } a \in \mathbb{Z}_p^+, b \in \mathbb{Z}_p \}$$

<u>Lemma 2:</u> If $x \neq y$ and $r \neq s$ then there is a unique solution (a,b) to the system

$$ax + b = r \bmod p$$

$$ay + b = s \bmod q$$

Constructing Universal Hash Families

$$h_{a,b}(x) = (ax + b \bmod p) \bmod m$$

$$\mathcal{H}_{p,m} = \{ h_{a,b} \text{ for } a \in \mathbb{Z}_p^+, b \in \mathbb{Z}_p \}$$

<u>Thm:</u> $\mathcal{H}_{p,m}$ is a universal hash family

<u>Proof:</u>