Polynomial hashes

$\mathbb{Z}_p^d \to \mathbb{Z}_p \quad \rightsquigarrow \quad x = (x_1, x_2, x_3, \dots, x_d)$

Def: $h_a(x) = \sum_{i=1}^{d} x_i \cdot a^{i-1}$

$= x_1 + x_2 \cdot a + x_3 \cdot a^2 + \dots x_d \cdot a^{d-1}$

≈ polynomial with coefs. x_i evaluated at point a

$h_a(x)$ can be calculated in $O(d)$ time

Def family of polynomial hash functions

$\mathcal{R} = \{ h_a \mid a \in \mathbb{Z}_p \}$

Only $O(1)$ time to pick h_a

(vs $\Theta(d)$ for scalar product family)

Thm $\mathcal{R}$ is d-universal

Proof

Fact polynomial p of degree d has $\le d$ roots unless $p \equiv 0$

- let $x \neq y \in \mathbb{Z}_p^d$

$\underbrace{h_a(x)}_{\text{poly}(a)} = \underbrace{h_a(y)}_{\text{poly}(a)} \iff \underbrace{h_a(x) - h_a(y) = 0}_{\text{poly}(a)}$

- def $p(t) = h_x(t) - h_y(t) = \sum_{i=1}^{d} (x_i - y_i) t^{i-1}$

‼ t is variable ‼ x, y constants

- $x \neq y \Rightarrow \exists i : x_i \neq y_i \Rightarrow p(t) \not\equiv 0$
- $h_a(x) = h_a(y) \iff a$ is a root of $p(t)$

→ $p(t)$ has $\le d$ roots

$\Rightarrow \Pr[h_a(x) = h_a(y)] \le \frac{d}{p}$ □

- Compose with linear functions to get 2-independence ← lemma 6

Corollary for prime p and #buckets m s.t. $p \ge 4 \cdot d \cdot m$ family $\mathcal{R} \circ \mathcal{L}$ is $(2, 5/2)$-independent

Variable lenght strings

- x is a string of lenght $\leq L$
  - → pad x with some character # to get lenght exactly L
    - → then use $\mathcal{R}$
- # must not appear elsewhere in x
  - ~ # is "new" character
- → good choice is #=0 (saves us work)

Application: Rabin-Karp algorithm

input: long text σ (haystack)
short string r (needle)
$|r|=d$, $|\sigma|=n$

output: all occurances of r in σ

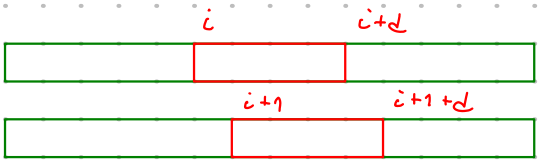
Naive algorithm:

for $i = 1 \dots (n-d)$: ⇒ time $O(n \cdot d)$

check $\sigma[i:i+d] \overset{?}{=} r$

$O(d)$

→ sliding window



Rabin-Karp:

- → first compare hash of $\sigma[i:i+d]$ ($= k$) with hash of r } $O(1)$:)
- → $h(r) \neq h(k) \Rightarrow r \neq k$
- → $h(r) = h(k) \Rightarrow$ check $r \overset{?}{=} k$

$\Rightarrow O(n+d+d\cdot(\#\text{occurances}+\#\text{collisions}))$ ($O(1)$ on average for reasonable h)

(Also works well for multiple needles)

Problem:

- we need $h(\sigma[i:i+d])$ (H_i) $\rightarrow h(\sigma[i+1:i+1+d])$ (H_{i+1}) in $O(1)$ time

Solution: use polynomial (rolling) hash $\mathcal{R}$

$$H_i = h_a(\sigma[i:i+d]) = \sigma_i \cdot a^{d-1} + \sigma_{i+1} \cdot a^{d-2} \dots + \sigma_{i+d-1} \cdot a^0$$

$$H_{i+1} = h_a(\sigma[i+1:i+1+d]) = \sigma_{i+1} \cdot a^{d-1} + \dots + \sigma_{i+d-1} \cdot a^1 + \sigma_{i+d} \cdot a^0$$

$$H_{i+1} = a \cdot H_i - \sigma_i \cdot a^d + \sigma_{i+d}$$