February 23, 2023 1:31 PM

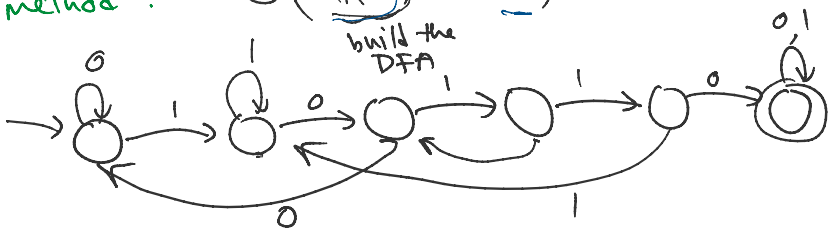
Problem Given strings $u = a_1 a_2 \cdots a_n \in \{0,1\}^*$ ← text Σ^*
$v = b_1 b_2 \cdots b_m \in \{0,1\}^*$ ← pattern Σ^*
$(m \ll n)$

is v a substring of u?

e.g. $u = 0110\boxed{10110}0$
$v = 10110$

brute force: $O(mn)$ time

"DFA method": $O(f(m) + n)$ time
($f(m)$: build the DFA)



Knuth-Morris-Pratt '77: $O(n)$ time
(by compressed version of DFA)
(regardless of $|\Sigma|$)

... a rand. $O(n)$-time alg'm
that is simpler ...

Subproblem Alice has a string $u \in \{0,1\}^*$
Bob has a string $v \in \{0,1\}^*$ $(|u| = |v| = n)$

Want to test whether $u = v$.

Goal - minimize communication complexity
i.e., # bits transmitted

Obvious idea - Alice transmits n bits

another idea - "checksum"
check # of 1's ($\log n$ bits)

Wrong! counterex: $101 \neq 110$

another idea -

use a mapping $F: \{0,1\}^* \to \{0,1,\ldots,p-1\}$

e.g. think of string as base-2 number, then mod p

randomized idea 1 - pick p randomly

randomized idea 2 - fix prime p.
use rand base x

let $F_x: \{0,1\}^* \to \{0,1,\ldots,p-1\}$

$$F_x(a_{n-1}a_{n-2}\cdots a_0) = \left(\sum_{i=0}^{n-1} a_i x^i\right) \bmod p$$

fingerprint function

Monte Carlo Alg'm: $\leq \log p$ bits transmitted

$x = \text{rand}(0, p-1)$

Alice transmits $F_x(u)$ to Bob

Bob says "probably equal" if $F_x(u) = F_x(v)$
"definitely not equal" else

Error Analysis:

if $u = v$, correct

if $u \neq v$,

say $u = a_{n-1}a_{n-2}\cdots a_0$
$v = b_{n-1}b_{n-2}\cdots b_0$,

alg'm errs $\iff$ $F_x(u) = F_x(v)$

$$\iff \sum_{i=0}^{n-1} a_i x^i \equiv \sum_{i=0}^{n-1} b_i x^i \pmod p$$

$$\iff \sum_{i=0}^{n-1} (a_i - b_i) x^i \equiv 0 \pmod p \quad (*)$$

Pr

Counting Lemma: A non-zero polynomial of deg $\leq n-1$ has $\leq n-1$ roots, mod p.

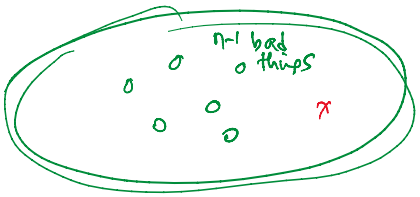
x's satisfying $(*)$

has $\leq n-1$ roots, mod p.

x's satisfying (*) / total # x's

$\Rightarrow \Pr[\text{alg'm errs}] \leq \frac{n-1}{p} \leq \frac{1}{n^d}$

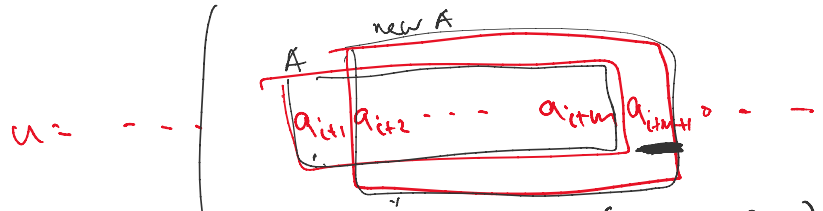
pick $p \approx n^{d+1}$ $\Rightarrow$ # bits transmitted $\log p \approx$ $O(\log n)$.



Back to string matching problem...

Rabin-Karp Rand. Alg'm (Monte Carlo)

```
x = rand(0, p-1)
B = F_x(b_1 b_2 ... b_m)
A = F_x(a_1 a_2 ... a_m)
for i = 0 to n-m {
    if A = B
        return "probably match at position i"
    // update A from F_x(a_{i+1} ... a_{i+m})
    //           to F_x(a_{i+2} ... a_{i+m+1})
    A = (A x + a_{i+m+1} - a_{i+1} x^m) mod p
}
// ... match"
```



precompute

x^m — $O(1)$ time

$O(1)$ time

} return "definitely no match"

$\Rightarrow$ $O(n)$ arithmetic op on $(\log p)$-bit #s

$O(\log n)$-bit #s

Error Analysis:

let E_i = (alg'm errs at i^{th} iteration)

by Alice-Bob, $Pr(E_i) \leq \frac{m-1}{p}$

$$Pr(\text{alg'm errs}) = Pr\left(\bigcup_{i=0}^{n-m} E_i\right)$$

$$\leq \sum_{i=0}^{n-m} Pr(E_i)$$

$$\leq n \frac{m}{p} \leq \frac{n^2}{p}$$

$$\lesssim \frac{1}{n^d}.$$

pick $p \simeq n^{d+2}$

$\Rightarrow$ $O(n)$ time

Las Vegas Version:

1. run Monte Carlo version of Karp-Rabin
2. if it says "probable match at i" {
3. verify $a_{i+1} \cdots a_{i+m} = b_1 \cdots b_m$ in $O(m)$ time
4. if so, return "match at i"
5. else run brute force

}

Always Correct

Prob $\geq 1 - \frac{1}{n^d}$

Always Correct

Expected runtime analysis:

if line 3 true, $O(n+m) = O(n)$ time (prob $\leq 1 - \frac{1}{n^d}$)

else $O(mn)$ time (prob $\leq \frac{1}{n^d}$)

expected runtime $\leq O(n) \cdot \left(1 - \frac{1}{n^d}\right) + O(mn) \cdot \frac{1}{n^d}$

$d=1$: $\leq \boxed{O(n)}$.