Given alphabet $\Sigma$, and two strings t, p such that that $t \in \Sigma^*$, $p \in \Sigma^*$, $|p| = m$, $|t| = n, m \leq n$

Goal: Find all positions $k$ in string $t$ ($1 \leq k \leq n - m + 1$) such that $t[k, k+m] = p$ where $x[a,b]$ is a substring of $x$ from $a$ to $b$.

Notation Examples:
- $t[1,n] = t$
- $x = \text{abcabcab}$
- $x[3,5] = \text{cab}$
- $x[i] = \text{ith element of } x$

Algorithm Examples:
- $t = \text{abababcababcab}$
- $p = \text{abc}$
- output = 3, 8

Definitions:
Prefix: $w$ is a prefix of $x$ ($w \sqsubset x$) if $x=wy$ for some $y \in \Sigma^*$
Suffix: $w$ is a suffix of $x$ ($w \sqsupseteq x$) if $x=yw$ for some $y \in \Sigma^*$

Example:
- $x = \text{abcab}$
- $w = \text{ab}$
- $w \sqsubset x$ and $w \sqsupseteq x$

Algorithm goal (take 2): Find all $K$ such that $p \sqsupseteq t[k,k+m]$

Naïve Matching:
Input $t, p$
$n = \text{length}(t)$
$m = \text{length}(p)$
for $i \leftarrow 1$ to $n-m+1$:
\hspace{1cm} if $t[i,i+m] = p$
\hspace{2cm} print “found one”
if $t[i,i+m] = p$
    for $j \leftarrow 1$ to $m$:
        if $t[i+j-1] \sim p[j]$ 
            return 0
    end
end
return true

Number of comparisons:

Best case:
“quick mismatch” ($t[i] \sim p[i]$) is $O(n)$

Worst case:
Always full substring match $O(n \times m)$

\[ e.g. t = a^n, p = a^m \]

Rabin-Karp:
Idea: exploit the fact that $t[i,i+m]$ and $t[i+1, i+1+m]$ are related.
Assume:
\[ \Sigma = \{0, 1, 2, \ldots, 9\} \]
\[ T=715123128 \]
\[ P = 12 \]
\[ T_v = 71, 15, 51, 12, 23, 31, 12, 28 \text{ (vector of integers)} \]
\[ P_v = 12 \]

\[ e.g. 71 = 7 \times 10^{1} = 1 \times 10^{0} \]

\[ |p| = m \rightarrow P_1 \times 10^{m-1} + P_2 \times 10^{m-2} + \cdots + P_m \]

Horner’s Rule
\[ P_v = P_m + 10((P_{m-1}) + 10(\cdots)) \]

E.g.
\[ 512 \]
\[ P_v = 5 \]
\[ P_v = 5 \times 10 + 1 \]
\[ P_v = 51 \times 10 + 2 \]

E.g.
\[ 71 \rightarrow 15 \]
\[ 71 - (7 \times 10^{m-1}) \times 10 + 5 \]

Complexity so far:
\[ O(m) \text{ to find } P_v \]
\[ O(m) \text{ to calculate } 10^{m-1} \]
\[ O(m+n) \text{ to calculate } T_v \]

\[ T_v[k+1] = 10(T_v[k] - 10^{m-1} \times T[k]) + t[k+m] \]
Note: $T[k]$ and $T[k+m]$ are integers
Problem: If m is large, we cannot use single precision, therefore calculation of \(Tv[k+1]\) cannot be done in constant time.

Idea: calculate everything modulo some number.

\[T = 715123128\]
\[Q = 3\]
\[P = 12\]
\[Tv[k+1] = (10 \times (Tv[k] - 10^{m-1} \mod Q \times T'[k]) + t[k+m]) \mod Q\]

\[Tv = 71, 15, 51, 12, 23, 31, 12, 28\]
\[Pv = 12\]
\[Tv^i = 2, 0, 0, 2, 1, 0, 1\]

Spurious Hits: Hits where the original strings are not matches but have the same answer modulo Q.

Hits: Actual hits.

Code:

Input T, P, Q (maybe d)
Pv \leftarrow 0
Tv(1) \leftarrow 0
C \leftarrow d^{n-1} \mod Q
For i \leftarrow 1 to m:
    Pv \leftarrow (d \times Pv + P(i)) \mod Q
    Tv(1) \leftarrow (d \times Tv(1) + T(i)) \mod Q
End
For k = 1 to n-m + 1:
    If Pv = Tv(k)
        If P = T[k, k+m]
            Print “found one”
        End
    End
Tv(k+1) \leftarrow (d \times (Tv(k) - T(k) \times c) + T(k+m)) \mod Q
End

Complexity of Rabin-Karp:
Preprocessing:
    O(m) for Pv
    O(m) for C
    O(m) for Tv(1)
Search Step:
    O(n-m+1) comparisons

Best case:
    O(n)
Worse case:
    O(n*m)

Average Case (Good hashing, (n/Q) spurious hits, v true hits):
    O(n) + O(m^* (v + n/Q)) = O(n) when n>>m and Q>m