Xiaoting Lian

CISC3220

String matching

The first solution for string match is called **Naïve**

Suppose we have a string A[A, S, D, F, G, H, J, K, L], the second string is B[D, F, G]

The idea of this algorithm is from beginning shift to the right until you match the string.

1. $n \leftarrow$ length [T]
2. $m \leftarrow$ length [P]
3. for $s \leftarrow 0$ to n-m do
4. $j \leftarrow 1$
5.   while $j \leq m$ and $T[s + j] = P[j]$ do
6. $j \leftarrow j + 1$
7.   If $j > m$ then
8.   return valid shift s
9. return no valid shift exist

the for-loop in line 3 is executed at most n - m +1 times, and the while-loop in line 5 is executed at most m times, The run time for naïve string matching is $O((n - m + 1)m)$

Example for Naïve string-matching algorithm

SHIFT =0

A= A. S. D. F. G. H, J, K, L

B= D, F, G

SHIFT=1

A= A. S. D. F. G. H, J, K, L

B=   D, F, G

SHIFT=2

A= A. S. D. F. G. H, J, K, L

B=   D, F, G

At this point the string is matched.

The second algorithm for string matching is **Rabin-Karp String Matching Algorithm**

Suppose we have a string S[1, 3, 4, 1, 4, 1, 2, 6]

Pattern P [4, 1, 4]

The basic idea is we consider P [4, 1, 4] as 4+1+4 =9, then we compare this value to every 3 elements in S string, until we find the matching string

To keep go on,

We take 1,2,4 from S, 1+2+4=7 which is not equal to 9,

we shift to right 3,4,1, 3+4+1= 8 which is not equal to 9

next we compare 4, 1 ,4 4+1+4 =9 valid match

as we keep go on, we can also find that 1+2+6 this is also equal to 9, but the string does not match, we call this **spurious hit.**

To solve this problem we need more variables for this algorithm to work

Example,

we set (4*10^2)+(1*10^1)+(4*10^0) this will give us 400+10+4=414

what we do next is take 1, 3, 4, (1*10^2)+(3*10^1)+(4*10^0) =100+30+4=134,

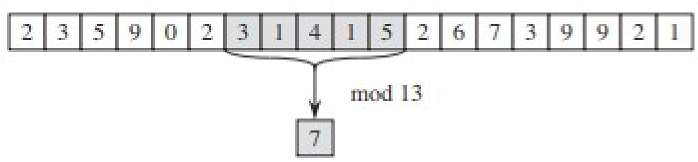
next is (134 – 1*10^2)*10+ 1*10^0 we minus 1*10^2 is to remove 1 from previous high order, multiply by 10 is to move all power by 1 with the previous then add the new digit.

As we keep shifting, we will notice that there will be only one substring in S that is match to P.

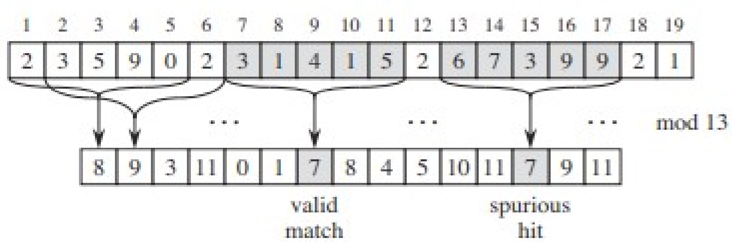
The book is giving different example that instead of adding all numbers, each character is converted in to decimal digit and strings of k-consecutive characters are represented as length-k decimal number, and find module for all numbers

Example: string 31415 is represented as 31,415

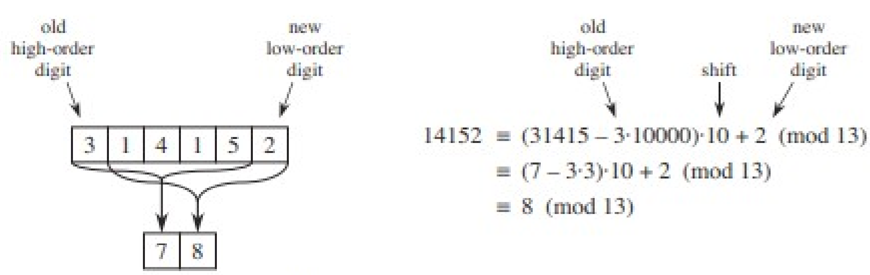
```
RABIN-KARP-MATCHER(T, P, d, q)
n = T.length
m = P.length
h = d^(m-1) mod q
p = 0
t_0 = 0
for i = 1 to m                         // preprocessing
    p = (dp + P[i]) mod q
    t_0 = (dt_0 + T[i]) mod q
for s = 0 to n - m                     // matching
    if p == t_s
        if P[1..m] == T[s + 1..s + m]
            print "Pattern occurs with shift" s
    if s < n - m
        t_{s+1} = (d(t_s - T[s + 1]h) + T[s + m + 1]) mod q
```



(a)



(b)



(c)

Average run time $O(n + m)$.

String matching with finite automata

Example

Sting S [a, b, a, b, a, b, a, c, a, b, a]

Pattern P [a, b, a, b, a, c, a]

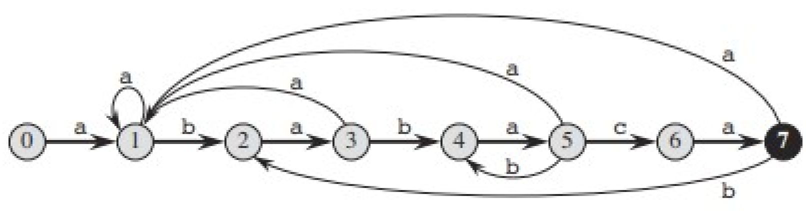
Numbers of type of element is a, b, c, d, 4 types

Total numbers of element of P is 6, the size of the table we will create is 6+1= 7.

State	a	b	c	P
0	1	0	0	a
1	1	2	0	b
2	3	0	0	a
3	1	4	0	b
4	5	0	0	a
5	1	4	6	c
6	7	0	0	a
7	1	2	0	

a match with a therefore the first box we put 1, b and c does not match we put zero.

Next row, we have ab and aa , only we only a and a match, we put 1 in the first box in sec row, then ab and ab, we get 2 match, so we put 2, do the same to the rest



(a)

Every time that arrow turn back that means string does not matching

```
FINITE-AUTOMATON-MATCHER(T, δ, m)
n = T.length
q = 0
for i = 1 to n
      q = δ(q, T[i])
      if q == m
            print "Pattern occurs with shift" i − m
```

The Knuth-Morris-Pratt algorithm

Example

We have string S[a,b,a,b,a,b,a,b,a,b,c]

The pattern is P[a,b,c,d]

Let S has a pointer i and P has pointer j

S

a	b	a	b	a	b	a	b	a	b	c
1	2	3	4	5	6	7	8	9	10	11

P

	a	b	a	b	c
	0	0	1	2	0
0	1	2	3	4	5

The second row for P table is when a and a are matching in P we note that second a with 1 and b appears twice we note that 2. c only appears once, we put 0, we consider ab as prefix.

Now we start comparing, when i is 1, we have a , j +1 is a too, then we move to the next, b and b are match as well, then we move again…. We see that when i=5 is a, j+1= 5 is c, doesn't match what we do here is not go all the way back, we move j to the noted number 2 in the p box which is the first b. then we keep comparing, j will go back to 2 three time. Then we fine the matched string.

```
KMP-MATCHER(T, P)
n = T.length
m = P.length
π = COMPUTE-PREFIX-FUNCTION(P)
q = 0                                  // number of characters matched
for i = 1 to n                         // scan the text from left to right
    while q > 0 and P[q + 1] ≠ T[i]
        q = π[q]                       // next character does not match
    if P[q + 1] == T[i]
        q = q + 1                      // next character matches
    if q == m                          // is all of P matched?
        print "Pattern occurs with shift" i − m
        q = π[q]                       // look for the next match
```

```
COMPUTE-PREFIX-FUNCTION(P)
m = P.length
let π[1 . . m] be a new array
π[1] = 0
k = 0
for q = 2 to m
    while k > 0 and P[k + 1] ≠ P[q]
        k = π[k]
    if P[k + 1] == P[q]
        k = k + 1
    π[q] = k
return π
```

Questions:

1. T[a,a,a,a,a,a,a,b], P[a,b] how many shift do we needed in naïve string maching algorithm.
   Ans:8-2=6 shift
2. Rabin Karp Algorithm makes use of elementary number theoretic notions.T or F
   Ans: T
3. What is worst case run time for Rabin-Karp string matching algorithm
   Ans: $O(n+m)$
4. What is average case run time for Rabin-Karp string matching algorithm
   Ans: $O(nm)$
5. What is it called when if the rolling hash produces a candidate match due to this hash collision, which turns out not to be a string match?
   Ans: spurious hit