问题求解（三）第17周作业

161220049　黄奕诚

December 25, 2017

TC Chapter 32

32.1-2

因为P中每一个字符都不同，所以当找到了一个匹配之后，可以直接跳—P—位，而不是1位，具体算法如下：

Algorithm 1 NAIVE-STRING-MATCHER(T, P)

```
n = T.length
m = P.length
k = 0
s = 0
while s ≤ n − m do
    i = 1
    if T[s] == P[1] then
        k = s
        i = 0
        while T[k + i] == P[i] and i < m do
            if i == m then
                Print ”Pattern occurs with shift” k
            end if
            i = i + 1
        end while
    end if
    s = s + i
end while
```

32.1-3

Proof. 因为$d \geq 2$，并且前$i-1$个字母匹配的概率为$\frac{1}{d^{i-1}}$，所以字母对比较的期望值为

$$
\begin{aligned}
(n-m+1)\sum_{i=1}^{m}\frac{1}{d^{i-1}} &= (n-m+1)\frac{1-(\frac{1}{d})^m}{1-\frac{1}{d}} && (1)\\
&= (n-m+1)\frac{1-d^{-m}}{1-d^{-1}} && (2)\\
&\leq (n-m+1)\frac{1}{1-d^{-1}} && (3)\\
&\leq (n-m+1)\frac{1}{1-\frac{1}{2}} && (4)\\
&= 2(n-m+1) && (5)
\end{aligned}
$$

□

32.1-4

假设P可以表示为$p_1 \diamond p_2 \diamond \cdots \diamond p_m$.可以用如下算法在$O(mn)$的时间内判断出$T$中是否存在这样的$P$：

Algorithm 2 GAP-STRING-MATCHER(T, P)

```
i = 1
while i ≤ m do
    exi = 0
    for j = i to n do
        if T[j] == P[i] then
            exi = 1
        end if
    end for
    if exi == 0 then
        return false
    end if
    i = i + 1
end while
return true
```

32.2-1

考虑到$15 \mod 11 = 4, 59 \mod 11 = 4, 92 \mod 11 = 4$，所以共有3次伪命中.

32.2-2

对于相同长度的k个模式，只要在Rabin-Karp算法的预处理模块中对每一个模式计算它的值即可，并且对每一个模式执行第10-14行(即加一个for循环)，此时渐进运行时间为$O(km(n-m+1))$.

对于不同长度的k个模式，只需要在执行第9行的for循环之前更新模式的长度即可，预先可以将每一个模式的长度存储在数组中，并在第9行的for循环外层再套一个for循环，以更新长度.

32.2-3

假设P是$m \times m$的矩阵，T是$n \times n$的矩阵.则对于每个$i \in [1, n-m+1], j \in [1, n-m+1]$计算$T$中每个规模为$m \times m$的子矩阵的值，求值时将其看作一维数组，由左到右，由上到下计算即可，并求出模式P的值.先比较值，若相等，为了摒除伪命中，则依次比对各个位上的字符，若全等则为一次occur.如此便完成了Rabin-Karp算法的拓展.

32.2-4

Proof. 假设$A(x) = B(x)$，则有$\sum_{i=0}^{n-1}(a_i - b_i)x^i \equiv 0 \mod q$.由Exercise 31.4-4及$q$为素数可知这个方程最多有$n-1$个解.从$q$中选中这$n-1$个解的概率为$P = \frac{n-1}{q} < \frac{n-1}{1000n} < \frac{1}{1000}$.所以若$A \neq B$，则最多有千分之一的概率得到$A(x) = B(x)$.若$A = B$，则必然有$A(x) = B(x)$. $\square$

32.3-2

现状态	输入a	输入b	现状态	输入a	输入b
0	1	0	11	1	12
1	1	2	12	3	13
2	3	0	13	14	0
3	1	4	14	1	15
4	3	5	15	16	8
5	6	0	16	1	17
6	1	7	17	3	18
7	3	8	18	19	0
8	9	0	19	1	20
9	1	10	20	3	21
10	11	0	21		

图太懒不画qaq......

32.3-3

考虑到P有非重叠性质，即若$P_k \sqsupseteq P_q$，则$k = 0$或$k = q$.于是相应的状态转移具有如下的特征：

(1) 如果当前输入使得状态机返回到除了0状态(初始状态)、1状态以外的任意状态，则已构造的一些后缀是要找的P的前缀

(2) 除去(1)中情况，若状态返回到0状态，则输入不是pattern的第一个字母

(3) 除去(1)(2)中情况，则会返回到1状态，即输入是pattern的第一个字母

32.3-5

设P可以表示为$p_1 \diamond p_2 \diamond \cdots \diamond p_m$，思路与32.1-4类似，先寻找$P_1$，找到后再往后找$P_2$，如此往复.此时涉及到状态机，将其修改为：$P_i$状态所能接受的状态不再是pattern中涉及的状态，而只有P_{i+1}.如此可以在$O(n)$的时间内在T中匹配P.