清华大学软件学院
算法分析与设计基础
2020 年春季学期

---

# 作业 8

吴佳龙 班级：软件 83 学号：2018013418 May 7, 2020

---

8.1. (CLRS Exercises 32.4-8)

**Lemma 1.** 若 $q=m$ 或 $P[q+1]\neq a$，则 $\delta(q,a)=\delta(\pi[q],a)$。

**证明**. 若 $q<m$ 且 $P[q+1]\neq a$，$\delta(q,a)=\sigma(P_qa),\delta(\pi[q],a)=\sigma(P_{\pi[q]}a)$，由于 $P_{\pi[q]}a$ 是 $P_qa$ 的后缀，因此 $\sigma(P_qa)\geq\sigma(P_{\pi[q]}a)$。若不等号严格成立，即
$\sigma(P_qa)=l_2>\sigma(P_{\pi[q]}a)=l_1$，则 $P_{l_2}$ 是 $P_qa$ 的后缀，$P_{l_2-1}$ 是 $P_q$ 的后缀，因此
$\pi[q]\geq l_2-1$，则 $|P_{\pi[q]}a|\geq l_2$，则 $\sigma(P_{\pi[q]}a)\geq l_2$，矛盾。因此，
$\delta(q,a)=\sigma(P_qa)=\delta(\pi[q],a)=\sigma(P_{\pi[q]}a)$。

若 $q=m$，令 $P'=P\#$ 为另一模式串，其中 $\#\notin\Sigma$，则
$\delta(q,a)=\delta'(q,a),\pi[q]=\pi'[q],\forall q\leq m$。同上讨论，可知，
$\delta(m,a)=\delta'(m,a)=\delta'(\pi'[m],a)=\delta(\pi[m],a)$ □

由以上引理，计算自动机转移函数的算法伪代码如下，该算法的时间复杂度为 $\Theta(m|\Sigma|)$。

```
COMPUTE−TRANSITION−FUNCTION(P, Sigma):
    COMPUTE−PREFIX−FUNCTION(P)
    for a in Sigma:
        delta(0,a) = P[1]==a?1:0
    for j = 1..m:
        for a in Sigma:
            if j<m and P[j+1]==a:
                delta(j,a) = j+1
            else:
                delta(j,a) = delta(pi[j],a)
```

8.2. (CLRS Problems 32-1)

(a) 算法的伪代码如下，该算法的复杂度是 $\Theta(m)$。

```
COMPUTE_RHO(P):
    COMPUTE−PREFIX−FUNCTION(P)
    for i = 1..m:
        if i mod (i−pi[i])==0:
            rho[i] = i/(i−pi[i])
        else:
            rho[i] = 1
```

(b) **证明**. $P_i$ 有重复因子 $r|i$ 的概率为 $(\frac{1}{2})^{i\frac{(r-1)}{r}}$，则
则当 $r>1$ 时，$\exists i,P_i$ 有重复因子 $r$ 的概率

$$\leq\sum_{j=1}^{\lfloor m/r\rfloor}(\frac{1}{2})^{jr(r-1)/r}=\sum_{j=1}^{\lfloor m/r\rfloor}(\frac{1}{2})^{j(r-1)}\leq(\frac{1}{2})^{(r-1)}$$

最后一个不等式依据 $\frac{q}{1-q} \le q \Leftarrow q \le \frac{\sqrt{5}-1}{2}$

则

$$
\begin{aligned}
E(\rho^*(P)) &= \sum_{r=1}^{m} r \cdot Prob(\rho^*(P) = r) \\
&\le Prob(\rho^*(P) = 1) + \sum_{r=2}^{m} r \cdot (\frac{1}{2})^{(r-1)} \\
&\le 1 + \sum_{r=2}^{\infty} r \cdot (\frac{1}{2})^{(r-1)} = 1 + O(1) = O(1)
\end{aligned}
$$

□

(c) 该算法是 brute-force 算法的变体，仅更改了当下一个字符不匹配时，$s$ 移动 $\max(1, \lceil q/k \rceil)$ 而不仅仅是 1。这是正确的。

因为，当 $T[s+1..s+q]$ 与 $P[1..q]$ 匹配时，若存在 $l > 0$ 使得 $T[s+l+1..s+l+m]$ 与 $P$ 匹配，且 $l < \lceil q/k \rceil$（也就是说 $lk < q$），则
$P[1..(k-1)l] = T[s+l+1, s+l+(k-1)l] = T[s+l+1, s+lk]$ 是
$P[1..lk] = T[s+1, s+lk]$ 的后缀，这就是说 $P[1..kl]$ 有重复因子 $k = \rho^*(P)+1$，这与 $\rho^*(P)$ 的定义矛盾。

该算法的时间复杂度分析类似 KMP 算法。在算法第 12 行进行 $s$ 增加 $\Delta s$ 时，$q$ 从上一次 $s$ 改变后至多增加了 $O(\Delta s \cdot k) = O(\Delta s \cdot \rho^*(P))$ 次，而 $\Delta s$ 的总和是 $O(n)$ 的。加上计算 $\rho(P_i)$ 和 $\rho^*(P)$ 的复杂度为 $O(m)$。该算法的总复杂度是 $O(\rho^*(P)n + m)$

8.3. 实现并测试 Brute-Force、KMP、BM 三种字符串匹配算法并提交实验报告。

详见源代码和实验报告。