傅申 PB20000051
2022 秋

算法基础 作业 13

25.2–5. 假定我们修改式 (25.7) 对等式的处理办法如下:

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & d_{ij}^{(k-1)} < d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \\ \pi_{kj}^{(k-1)} & d_{ij}^{(k-1)} \geqslant d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \end{cases} \tag{5.1}$$

请问这种前驱矩阵 Π 的定义正确吗?

解: 仍然是正确的. 在 $d_{ij}^{(k-1)} = d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$ 的情况下, 选择 $\pi_{kj}^{(k-1)}$ 相当于选择 $i \rightsquigarrow k \rightsquigarrow j$ 的路径, 而选择 $\pi_{ij}^{(k-1)}$ 相当于选择 $i \rightsquigarrow j$ 的路径, 且这两种路径的权值是相等且最小的, 都能构建出最短路径.

25.3–3. 假定对于所有的边 $(u, v) \in E$, 我们有 $w(u, v) \geqslant 0$. 请问权重函数 w 和 $\hat{w}$ 之间是什么关系?

解: 因为所有边的权值都是非负的, 所以 $\forall v \in V, h(v) = \delta(s, v) = 0$. 因此 $\forall (u, v) \in E, \hat{w}(u, v) = w(u, v) + h(u) - h(v) = w(u, v)$.

25.3–5. 假定在一个权重函数为 w 的有向图 G 上运行 Johnson 算法. 证明: 如果图 G 包含一条权重为 0 的环路 c, 那么对于环路 c 上的每条边 (u, v), $\hat{w}(u, v) = 0$.

解: 因为在重新赋值后, 有 $\hat{w}(c) = w(c) + h(v_0) - h(v_0) = w(c) = 0$. 且对于所有的边, 重新赋值后权值都是非负的. 所以, 对于环路 c 上的每条边 (u, v), 都有 $\hat{w}(u, v) = 0$.

32.1–2. 假设在模式 P 中所有的字符都不相同. 试说明如何对一段 n 个字符的文本 T 加速过程 Naive-String-Macher 的执行速度, 使其运行时间达到 $O(n)$.

解: 因为 P 中所有字符都不相同, 所以 P 的任何前缀的后缀都不会是 P 的前缀. 因此, 在匹配到 P_i 失败时 ($P_i \neq T_{s+i}$), 直接跳过前面的 $i - 1$ 个字符, 即给 s 赋值为 $s + i - 1$ 重新从头开始匹配.

32.2–2. 如何拓展 Rabin-Karp 算法, 使其能解决如下问题: 如何在文本字符串中搜寻出给定的 k 个模式中的任何一个出现? 起初假设所有 k 个模式都是等长的, 然后拓展你的算法以适用于不同长度的模式.

解: 先考虑所有 k 个模式都是等长的情况, 只需要先计算出所有模式的 p 值, 然后在匹配过程每次迭代都做 k 次比较即可.

然后考虑 k 个模式长度不同的情况, 设这 k 个模式长度分别为 $m_1, \cdots, m_k$. 使用两个数组 p 和 t 分别存储 k 个模式的 p 值和 t_s 值. 在预处理中对每个模式都计算出 p 和 t_0 并存储在数组中, 然后在匹配过程中每次迭代都分别进行 k 次比较即可.

32.2–3. 试说明如何拓展 Rabin-Karp 算法, 用于处理以下问题: 在一个 $n \times n$ 的二维字符数组中搜索一个给定的 $m \times m$ 模式. (该模式可以在水平方向和垂直方向移动, 但是不可以旋转)

解: 考虑将 $m \times m$ 的模式看作 m 行一维模式, 在 $m \times n$ 的滑动二维窗口中同步搜索这 m 个一维模式, 其中每行都使用 Rabin-Karp 算法. 只有当这 m 个一维模式的 p 值都与对应行的 t_s 值相匹配时, 才检测 $P[1..m, 1..m]$ 是否与 $T[s+1..s+m, s+1..s+m]$ 相匹配. 每次搜索完一个窗口后就将窗口向下移动一行从头开始搜索, 直到搜索完整个 $n \times n$ 的二维字符数组.

32.4–3. 试说明如何通过检查字符串 PT (由 P 和 T 连结形成的长度为 $m+n$ 的字符串) 的 π 函数来确定模式 P 在文本 T 中的出现位置.

解: 对于下标 i, 若 $\pi[i] = m$, 则说明 P 是 $(PT)_i$ 的真后缀, 即 $(PT)[i-m+1..i] = P$. 若 $i > 2m$, 则 $(PT)[i-m+1..i]$ 就是 $T[i-2m+1..i-m]$. 因此, 模式 P 在文本 T 中出现位置就是 $\{i-2m \mid \pi[i] = m, i > 2m\}$.

32.4–7. 写出一个线性时间的算法, 以确定文本 T 是否是另一个字符串 T' 的循环旋转. 例如 arc 和 car 是彼此的循环旋转.

解: 因为 T 的循环旋转一定是 TT 的字串, 所以只需要使用 KMP 算法在 $T'T'$ 中搜索 T 即可, 时间复杂度为 $\Theta(2n) = \Theta(n)$.