# Lyndon 数组，Lyndon 树，一个简介

EtaoinWu[1]

说明：这篇基本是[1]的翻译。

## 1 基本定义

给定一个有限集$\Sigma$，称作**字符集**。称$\Sigma^*$（$\Sigma$的任意长序列的集合，又称为“$\Sigma$上的自由幺半群”）的一个元素为一个**字符串**。借用自由幺半群的记法，用乘法（或者中间什么都不写）来表示字符串和/或字符的拼接。使用$|s|$表示字符串$s$的**长度**。使用$s[i]$表示$s$中第$i$号位置的元素，$1 \le i \le |s|$。称$s[l\,..\,r] = s[l]s[l+1]\cdots s[r]$为$s$的**子串**。$s[1\,..\,k], s[k\,..\,|s|]$ 分别称作 $s$ 的**前缀**、**后缀**。$\mathrm{pre}\,(s,k) := s[1\,..\,k]$, $\mathrm{suf}(s,k) := s[|s|-k+1, |s|]$。

对于一棵二叉树，我们定义一个**左结点**是根或其父亲的左孩子，**右结点**同理。

## 2 字符串的等价与自等价

若$\forall 1 \le x < |s| - p, s[x] = s[x+p]$，就称数字 $p$ 是 $s$ 的一个**周期**。若字符串 $t$ 既是 $s$ 的一个前缀，又是 $s$ 的一个后缀，称 $t$ 是 $s$ 的一个 **border**。

$p$ 是 $s$ 的一个周期当且仅当 $\mathrm{pre}(s, |s| - p)$是$s$的一个 border。

---

[1] 北京大学，信息管理系，wu.y@pku.edu.cn。

引理 甲：如果 $p, q$ 都是字符串$s$的周期，$p+q \le |s|$，则 $\gcd(p, q)$ 也是 $s$ 的周期。

证明可以考虑设 $p > q$，$s[i] = s[i+p-q] = s[i-q+p]$，从略。

推论：一个字符串 $s$ 的所有$\le \frac{|s|}{2}$的周期都是最小周期的整数倍。

推论：一个字符串的所有长度过半的 border 构成一个等差数列。

Border 的结构：考虑一个字符串的长度在$2^{-k}|s|$到$2^{1-k}|s|$之间的 border，所有这样的 border 都是其中最长者的“过半 border”。因此，一个字符串的所有 border 构成至多 $\log_2 n$ 段等差数列。

若两个字符串$s, t$满足存在字符串 $v, w$ 使得 $s = vw, t = wv$，那么我们称它们**循环同构**（有的文献上称作**共轭**）。一个充要条件是$s$是$tt$的子串。

## 2.1 Runs

定义一个字符串$w$里的一个 **run**，指其内部一段两侧都不能扩展的周期子串，且周期至少完整出现两次。严格地说，一个 run 是一个三元组$(i, j, p)$，满足 $p$ 是 $w[i..j]$的最小周期，$j-i+1 \ge 2p$，且满足如下两个条件：要么$i = 1$，要么$w[i-1] \ne w[i-1+p]$；要么 $j = n$，要么$w[j+1] \ne w[j+1-p]$。

对于一个 run，其任一长度为 $p$ 的子串称作其一个 root。

我们用 $Runs(w)$表示$w$的所有 run 的集合。本文的主要结果是如下定理。

定理 甲，Runs Theorem：$|Runs(w)| < |w|$。

为了证明这一定理，我们需要一些别的东西。

# 3 字符串的序

方便起见，我们设$\prec_0$代表$\Sigma$上的小于，$\prec_1$代表$\Sigma$上的大于。每种字符序都可以诱导一个字典序（逐一比较；一个串的前缀小于其自身），滥用符号也记为$\prec_\ell, \ell \in \{0,1\}$。举例来说，如果 $\mathrm{a} \prec_0 \mathrm{b}$，那么 $\mathrm{b} \prec_1 \mathrm{a}$，并且 $\mathrm{a} \prec_0 \mathrm{ab}, \mathrm{a} \prec_1 \mathrm{ab}$。

## 3.1 后缀数组

对于一个字典序$\prec$，我们可以对一个字符串$w$的所有后缀排序。把这些排序好的后缀的左端点的下标放到一个数组中，称作**后缀数组** $\mathrm{SA}(w)$。很明显这是一个排列；其逆排列记作**排名数组**$\mathrm{ISA}(w)$，OI 中常见称作 rank。对SA上相邻两个位置对应的后缀求最长公共前缀（LCP），得到 LCP 数组 $\mathrm{height}(w)$。

对 $\mathrm{height}_n$做区间最值查询（RMQ），可以求字符串任意两个子串的最长公共前缀，这是熟知的。

一个应用是，给定$i, p$，查询从$i$开始、周期为$p$的最长子串。很明显，$\mathrm{lcp}(w[i..], w[i+p..]) + p$就是答案。

## 3.2 Lyndon 串

给定一个$\prec$。如果一个字符串$s$小于其所有后缀，那么称它是一个**Lyndon 串**（**Lyndon word**）。其充要条件是，$s$小于其所有循环移位。其充要条件是，对于所有拆分$s = vw$，都有$v \prec w$。

引理 乙：如果$v, w$是两个 Lyndon 串，$v \prec w$，那么$vw$是一个 Lyndon 串。

证明：$v \prec w$的两种情况是$v \in \mathrm{pre}(w)$或$\exists i, v[i] \prec w[i]$。这两种情况下都有$vw \prec w$。$vw$的后缀要么是$\mathrm{suf}(v)w$，要么是$\mathrm{suf}(w)$，这二者都能证明$\prec vw$。

### 3.3 Lyndon 分解

定理 二，Chen-Fox-Lyndon Theorem：任何字符串可以唯一分解成若干个单调不上升的 Lyndon 串的续接。即对于任意字符串$s$，有唯一的$s = s_1 s_2 \cdots s_k$，其中$s_1 \succcurlyeq s_2 \succcurlyeq \cdots \succcurlyeq s_k$，且$s_i$都是 Lyndon 串。

这样的分解被称为 **Lyndon 分解**。

证明：先证存在性。最初$s = s[1]s[2] \cdots s[n]$（单字符都是 Lyndon 串），然后只要存在相邻两段满足前面$\prec$后面，就把它们合并起来（引理 乙）。唯一性的证明由下面的引理 丙自然可知。

引理 丙：一个串的 Lyndon 分解的第一个串一定是其最长 Lyndon 前缀。

应用反证法，设 $\tilde{s}$ 是 $s$ 的最长 Lyndon 前缀，则有 $\tilde{s} = s_1 s_2 \cdots s_k \text{pre}(s_{k+1}, l)$，则$\tilde{s} \prec \text{pre}(s_{k+1}, l) \preccurlyeq s_{k+1} \preccurlyeq s_1 \prec \tilde{s}$。

引理 丁：一个串的 Lyndon 分解的最后一个串一定是其最小后缀。
证明从略。

#### 3.3.1 Duval 算法

不断求一个串的最长 Lyndon 前缀，以计算其 Lyndon 分解。

引理 戊：设$w = u^k u' a$，其中$u$是一个 Lyndon 串，$u^k$表示$u$重复$k$次，$u'$是$u$的一个可能空的前缀，字符$a \neq u[|u'| + 1]$。若$a \succ u[|u'| + 1]$，那么$w$是一个 Lyndon 串。若$a \prec u[|u'| + 1]$，那么所有$w$开头的字符串的最长 Lyndon 前缀是$u$。

证明是十分简单的。

Duval 算法的主要过程是，在字符串中不断迭代，并不断保持引理戊的条件。最初$w = u = \varepsilon$。不断向$w$的最后添加字符，设向$w = u^k u'$中加入字符$a$。(1)如果$a = u[|u'| + 1]$则继续。(2)如果$a \succ$

$u[|u'|+1]$，我们令$u \leftarrow u^k u' a$。(3)如果$a \prec u[|u'|+1]$，输出$k$个$u$，然后从$u'$的开头处重新开始主算法。

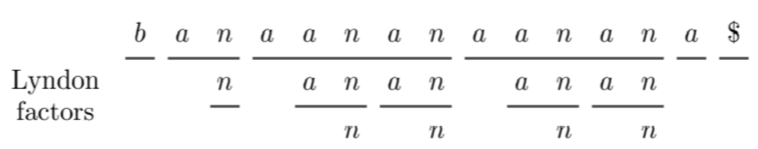
具体算法代码可以看 LibreOJ #129。

这一算法的时间复杂度是线性的，额外空间是常数的。时间线性可以如下证明：(3)的总输出量是$n$，并且每次向左重新开始的跳跃距离$|u'| < |u| < k|u|$即这一次的输出量，因此总的回跳量是$< n$的；因此添加字符的总次数$< 2n$。

### 3.4 Lyndon 数组

对于一个字符串$s$，我们定义$\mathrm{LA}(w)$是这样一个数组，这一数组的下标为$k$的位置为$w[k]$开头的最长 Lyndon 子串的长度。这称作 **Lyndon 数组**（**Lyndon array**）。注意，此节往后，我们讨论的字符串都加入了一个$\prec_0$意义下的无穷小字符\$。在$\prec_1$意义下，它便是无穷大。

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T = | $b$ | $a$ | $n$ | $a$ | $a$ | $n$ | $a$ | $n$ | $a$ | $a$ | $n$ | $a$ | $n$ | $a$ | \$ |
| SA = | 15 | 14 | 9 | 4 | 12 | 7 | 2 | 10 | 5 | 1 | 13 | 8 | 3 | 11 | 6 |
| ISA = | 10 | 7 | 13 | 4 | 9 | 15 | 6 | 12 | 3 | 8 | 14 | 5 | 11 | 2 | 1 |
| $\mathrm{NSV_{ISA}}$ = | 2 | 4 | 4 | 9 | 7 | 7 | 9 | 9 | 14 | 12 | 12 | 14 | 14 | 15 | 16 |
| LA = | 1 | 2 | 1 | 5 | 2 | 1 | 2 | 1 | 5 | 2 | 1 | 2 | 1 | 1 | 1 |




图表 3-1 Lyndon 数组的展示[2]

引理 己，SA ⇒ LA：如果$\mathrm{LA}(i) = j - i + 1$，那么$\mathrm{ISA}_{j+1} < \mathrm{ISA}_i$，且 $j$ 是最小的满足这一性质的数。

请自行验证。这说明了 Lyndon 数组可以在线性时间内构造。对于两个序$\prec_0, \prec_1$，我们可以定义$\mathrm{LA}_0$和$\mathrm{LA}_1$两个数组。

引理 庚：对于两个序$\prec_0, \prec_1$，$\mathrm{LA}_0(i)$和$\mathrm{LA}_1(i)$恰好有一个是1。

证明：设$w[i]$右边第一个不同于$w[i]$的字符是$w[j]$，且$w[j] \prec_\ell w[i]$。那么由引理 戊，$w[i\,..\,j]$是一个$\prec_\ell$ Lyndon 串，且$w[i]$是其开头最长的$\prec_{1-\ell}$ Lyndon 串。

### 3.5 Lyndon-ness 与 run

对于一个 run $r = (i, j, p)$，我们设$\ell_r \in \{0,1\}$满足$w[j+1] \prec_{\ell_r} w[j+1-p]$。如果其一个子串既是它的根（长度为$p$）又是 Lyndon word，那么称这个子串是它的一个 **L-root**。

引理 辛：一个 run 的任一 L-root 一定恰好是这个根的左端点的$\mathrm{LA}_{\ell_r}$。

证明和引理 庚几乎完全相同。由这一引理立即可以得出，$|Runs(w)| \le 2|w|$。因此我们转而分析一个 run 的所有 L-root 的左端点。设$\overline{B_r}$表示一个 run 的所有 L-root 的左端点集合，$B_r = \overline{B_r} \setminus \{i\}$。显然$B_r$非空。

我们可以发现如下的引理：

引理 壬：不同的 run 有不相交的$B_r$。

采用反证法，假设$i \in B_r \cap B_{r'}$。设$[i\,..\,j]$和$[i\,..\,j']$分别是$r, r'$的 L-root，由引理 辛可知，$j$和$j'$中恰有一个$= i$。不妨设$i = j' < j$。由$B$的定义去掉了run 的左端点，知$i-1$也包含于两个run 中。由$r'$的周期为1，知$w[i-1] = w[i]$；由$r$，知$w[i-1] = w[j]$。那么$w[i] = w[j]$，因此$w[i\,..\,j]$不是 Lyndon 串，矛盾。

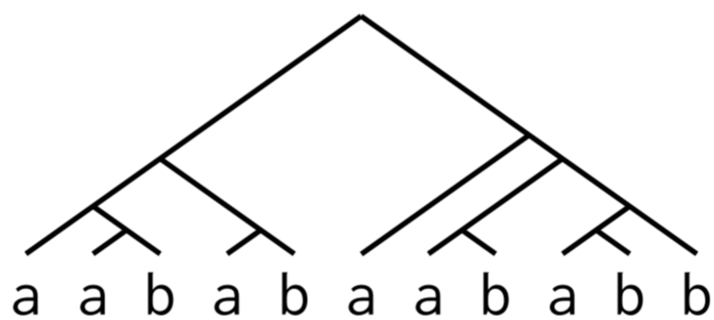
定理 甲，Runs Theorem：$|Runs(w)| < |w|$。

证 明 ： $|Runs(w)| = \sum_{r\in Runs(w)} 1 \le \sum_{r\in Runs(w)} |B_r| = \left|\bigcup_r B_r\right| \le |\{2,\cdots,|w|\}| = n-1 < n$。

利用以上结论，我们可以求所有 run 的位置：在两个LA数组中枚举位置，得到一个可能的 L-root，向两侧扩展，判断是否合法，然后去重即可。

# 4 Lyndon 树

对于一个长度大于一的 Lyndon 串 $w$，设$w=uv$，其中$v$是其所有真后缀中字典序最小的，可以证明$u,v$都是 Lyndon 串。我们称这样的分解方式为**标准分解**。递归进行这样的分解，可以得到一个树结构，称作 **Lyndon 树**（**Lyndon tree**）。




图表 4-1 aababaababb 的 Lyndon 树结构

对于 Lyndon 树的结构和 Lyndon 子串的关系，我们有如下引理：

引理 癸：设$w[l..r]$是一个 Lyndon 子串。对所有$[l..r]$范围内的叶子求 LCA，得到的节点对应的区间设为$[i..\ j]$。则$l=i$。

证明：设$[i..j]$这个结点的左右儿子分别是$[i..k],[k+1..j]$。注意到$i\le l\le k<k+1\le r\le j$。设$u=w[i..l-1],s=w[l..k],t=w[k+1..r],v=w[r+1..j]$。则$w[l..r]=st$是 Lyndon 串，那么$st\prec t$，则$stv\prec tv$，但$tv$是$w[i..j]=ustv$的最小真后缀，那么只能$u=\varepsilon$。

这是说，一个非叶右结点的左端点开头的 Lyndon 串都在这个非叶结点内部。因此，这个端点开头的最长 Lyndon 串就是这个非叶右结点本身。这告诉我们，已知LA$(s)$，用一个单调栈就可以求算其 Lyndon 树。

对于非 Lyndon 的串，我们可以在前面加一个（比最后那个更小的）无穷小字符，然后求 Lyndon 树。注意，为了保证 Lyndon 性，在有两个序的时候，两个无穷小是不同的，一个的无穷小是另一个的无穷大。严格地说，我们选取两个字符$\#_0, \#_1$，使$\forall x, \forall \ell$，应当有$\#_\ell \prec_\ell x$；也就是说，$\#_0 \prec_0 \$ \prec_0 x \prec_0 \#_1$，$\#_1 \prec_1 x \prec_1 \$ \prec_1 \#_0$。（这一串看起来容易眼花...）然后求 $\#_0 w\$$ 关于$\prec_0$的 Lyndon 树 $T_0$ 和 $\#_1 w\$$ 关于$\prec_1$的 Lyndon 树 $T_1$。

### 4.1 应用

设询问$exrun([l..r])$为包含$[l..r]$、周期不超过 $\frac{r-l+1}{2}$ 的 run，如果存在。他显然至多一个。这个东西是可以$O(1)$求的。

对于$exrun([l..r])$，设$m = \lceil \frac{l+r}{2} \rceil$。我们在$T_0$和$T_1$上分别寻找$[l..m]$的 LCA，记作$\alpha_0, \alpha_1$，对他们的右孩子$\beta_0, \beta_1$分别求以$\beta$为 L-root 的 run。答案如果存在，一定是这两个之一。

证明：假设答案存在$r = (i, j, p)$，设$\ell$满足$w[j+1] \prec_\ell w[j+1-p]$，由于$p \le \lfloor (j-i+1)/2 \rfloor$，我们有$l \le m-p < m+p-1 \le j$，因此$(i, j, p)$一定有一个 Lyndon 根包含$m$位置。设这一 Lyndon 根叫$\lambda$，它一定在 $\alpha_\ell$的子树内。我们用反证法证明$\lambda$就是$\alpha_\ell$的右孩子。设$\beta \ne \lambda$是$\alpha_\ell$的右孩子，由于三者都包含$m$位置，$\lambda$必然是$\beta$的子树内某个节点的右孩子。设$\lambda = [i_\lambda..j_\lambda], \beta = [i_\beta..j_\beta]$。有$i \le l < i_\beta < i_\lambda < r \le j$。讨论$j_\beta$和$j$的相对大小。如果$j_\beta \le j$，那么$\beta \subset [i..j]$，那么$\beta$就会有一个非平凡的周期，和 Lyndon 性矛盾；如果$j_\beta > j$，由$w[j+1] \prec_\ell w[j+1-p]$，可以得到$w[i_\lambda..j_\beta] \prec_\ell w[i_\beta..j_\beta]$，和 Lyndon 性矛盾。

## 5 参考文献


[0] 金策. 在 WC2017 上的讲课.

[1] BANNAI H, I T, INENAGA S, 等. The 《Runs》 Theorem[J]. SIAM Journal on Computing, 2017, 46(5): 1501–1514. DOI:10.1137/15M1011032.

[2] LOUZA F A, MANTACI S, MANZINI G, 等. Inducing the Lyndon Array[J/OL]. arXiv:1905.12987 [cs], 2019[2020–03–11]. http://arxiv.org/abs/1905.12987.