后缀自动机（姬）

算法思想

后缀自动机简称 SAM 。

我们记 $\sum$ 为字符集， $|\sum|$ 为字符集大小。

以下问题可以通过 SAM 在线性时间内解决。

1.在另一个字符串中搜索一个字符串的所有出现位置（ kmp 、哈希也可以）

2.计算给定的字符串中有多少个不同的子串。（后缀数组也可以线性做）

在直观上，我们可以把 SAM 理解为将字符串的所有子串压缩在一个树上。对于长度为 n 的字符串， SAM 的空间复杂度是 $O(n)$ 的，此外构造 SAM 的时间复杂度也是 $O(n)$ 的，小结论：一个 SAM 最多有 $2n-1$ 个结点和 $3n-4$ 条转移边。

我们把结点称作状态，边称为状态之间的转移。我们有一个初始的源点作为初始状态，其他各个结点都可以从这个源点出发到达。每个转移都标有一些字母，从一个结点出发的所有转移都不同。存在一个或多个终止状态，路径上所有转移连接起来一定是字符串的一个后缀，并且每一个后缀都可以用一条从源点到终止状态的路径构成。所以子串就是后缀的前缀也就是从源点开始到任意一个点的路径，所以可以说一个点对应着一个原字符串的子串。

结束位置 $endpos$ 考虑字符串 s 的任意非空子串 t ，我们记 $endpos(t)$ 为在字符串 s 中 t 的所有结束位置（从 0 开始），比如现在有一个字符串叫 $abcbc$ ，我们有 $endpos("bc")=2,4$ ，不同子串 t_{1} 和 t_{2} 的 $endpos$ 集合可能相等，比如刚才的 c 和 bc ，所以我们可以根据 $endpos$ 集合的不同将 s 的非空子串分为若干等价类。

SAM 中的每个状态对应一个 $endpos$ 相同的等价类，还有一个初始状态，所以 SAM 的状态个数等于等价类的个数 $+1$ 。

假设字符串 s 的两个非空子串 u 和 v 的 $endpos$ 相同，显然有短的字符串是长的字符串的后缀， $endpos$ 之间要么相交是空（不为后缀关系），要么是被包含的关系（其中一个是另一个的后缀），且满足这个等价类的子串的长度是一个连续的区间。

后缀链接 $link(v)$ 连接到对应于比 v 等价类最短的还短一个字符的字符串的等价类。所有的后缀链接构成一棵根节点为 t_{0} 的树。如果我们从任意状态 v_{0} 开始沿着后缀链接遍历，总会到达初始状态 t_{0} ，这种情况会得到一个互不相交的区间 $[minlen(v_{i}),len(v_{i})]$ ，并且他们的并集正好是 $ [0,len(v_{0})]$ 。