2020牛客暑期多校训练营（第一场）

牛客竞赛
AC.NOWCODER.COM

B–Suffix Array

> 给出一个字符串，定义一个字符串 $s_1 \sim s_m$ 对应的 $B_1 \sim B_m$ 满足 $B_i = \min_{1 \le j < i, s_j = s_i} \{i - j\}$，如果不存在这样的 j，那么 $B_i = 0$，现在要求将 s 的所有后缀按照其对应的 B 序列的字典序排列。

- Let C_i = min_{j > i and s_j = s_i} {j – i}
- The B–Suffix Array is equivalent to the suffix array of C_1 C_2 ... C_n

B-Suffix Array

证明：Bi相当于前面最近的一个与s_i相同的字符到 i 的距离，而 Ci相当于后面的最近的与 si 相同的字符到 i 的距离。

考虑一个特殊的情况。

对于一个后缀，考虑 BB 数组开头的一段，肯定是这样的： 0 1 1 1 0 ...

那两个 0 就是第一次出现的 a 和 b 字符，因为他们没有上一个相同的字符。

考虑两个 00 中间 11 序列的长度，他的长度越长，字典序就会越大，例如这两个 BB 数组： 0 1 1 1 0 ...

0 1 1 0 ...

显然下面那个字典序要小，因为第四位它是 0 而上面那个是 1。

然后考虑他们对应的 C 数组，是长这样的： 1 1 1 x 1 ...

1 1 y 1 ...

其中，x,y 都大于 1。

此时可以发现，上面那个的字典序反而要小了。

也就是说，对于开头的一段 1 长度不同的两个后缀，如果 X 的 B 比 Y 的 B 字典序要小，那么 X 的 C 一定比 Y 的 C 的字典序要大。

牛客竞赛 AC.NOWCODER.COM

B-Suffix Array

再考虑一般情况。

假设两个后缀 X,Y 的 B 序列有一段前缀是相同的，那么 X,Y 的这一段前缀肯定也是相同的，不妨设这段前缀以若干个 aa 结尾，那么对于第一个不同的位，肯定满足一个后缀的这一位是 a，另一个后缀的这一位是 b，像这样： ...baaaab...

...baab...

然后他们对应的 BB 序列就是： ...w1111x...

...w11y...

(x>y>1)后缀 X 的 B 的字典序比后缀 Y 的 B 的字典序要小。

再考虑他们的 C 序列： ...x111??...

...y1??..

然后你可以发现，在 x,y 之前的位都是相同的，由于x>y，所以此时 X 的 C 的字典序比 Y 的 C 的字典序要大。

Infinite Tree

一颗无限结点的树，任意大于1的点k与点$\frac{k}{mindiv(k)}$相连，其中$mindiv\left(k\right)$为k的最小质因子

记$\delta\left(u,v\right)$为树上$u-v$之间的距离，求$\min_u \sum_{i=1}^{m} w_i \delta\left(u,i!\right)$

Infinite Tree

题解

不考虑本题的树

我们先考虑这个题在已经知道树的结构下怎么解

显然$\delta(u,v)=dis(u,v)$

$$\min_u\sum_{i=1}^{m}w_i\delta(u,i!)=\min_u\sum_{i=1}^{m}w_i dis(u,i!)$$

在$root=1$的树中，假设现在$u=1$，那么当前答案$ans=\sum_{i=1}^{m}w_i dis(1,i!)$

记$f[u]=w[u]+\sum_{v\in son}f[v]$，那么在图中，显然$f[1]=\sum_{i=1}^{m}w_i$

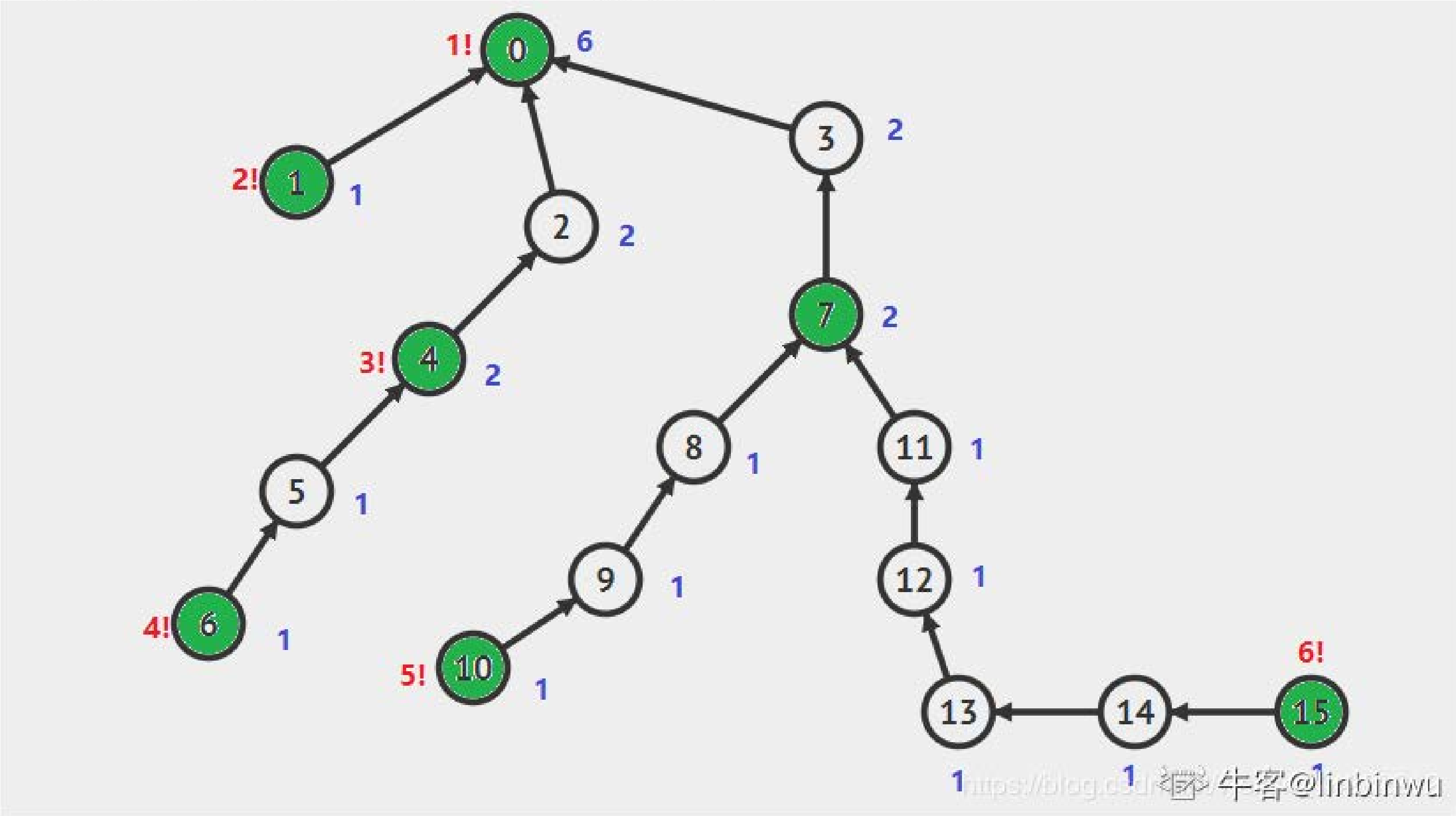
现在考虑当我们的点u转移到u的一个子节点v时，答案会发生什么变化

那么就有$f[1]-f[v]$多一段移动距离$dis(u,v)$，$f[v]$少一段移动距离$dis(u,v)$

所以当我们转移u点能够使答案变小的时候，即$(f[1]-f[v])-(f[v])=f[1]-2f[v]<0$时，我们就会移动$u$点，当不能继续移动时我们就找到了最终的答案

Infinite Tree

先来看看这个图(假设所有$w_i = 1$，蓝色的字为结点的f值)



Infinite Tree

对于结点i作质因数分解，记为$i = p_1^{k1} p_2^{k2} \cdots p_n^{kn}$

这棵树从根节点$1!$到点i的路径中，质因子由大变小，即经过的路径边上的质因数是形如$5,5,5,3,3,2,2,2,2$

且有$dis(i,1!) = \sum_{i=1}^{n} k_i$

而本题最大的点$m!, 1 \le m \le 1e5$是一个非常大的点，要将整棵树全部保存下来是不可能的

除了绿色的点之外，其他所有的点的f值都和他们的子节点相等!

也就是说，如果我们能够移动到点v，即有$f[1]-2f[v]<0$，如果$f[vv]=f[v]$那么肯定会继续移动下去

所以这些f值不变的点都是不重要的

只有我们的目标点$i!$和他们的最近公共祖先lca是有用的，这个就是虚树的概念

Infinite Tree

现在考虑怎么建虚树

目标点$i!$好说，就是他们之间的lca比较难求

之前我们说了：

这棵树从根节点$1!$到点i的路径中，质因子由大变小，即经过的路径边上的质因数是形如$5,5,5,3,3,2,2,2,2$

所以当$i!=p_1^{k1}p_2^{k2}\cdots p_n^{kn}$，当变成$(i+1)!$时，这条路径最先改变的地方就是$(i+1)$的最大质因子

如$2^4 3^2 5^3$乘$2^1 3^1$时

原来的路径：$5,5,5,3,3,2,2,2,2$

现在的路径：$5,5,5,3,3,3,2,2,2,2,2$

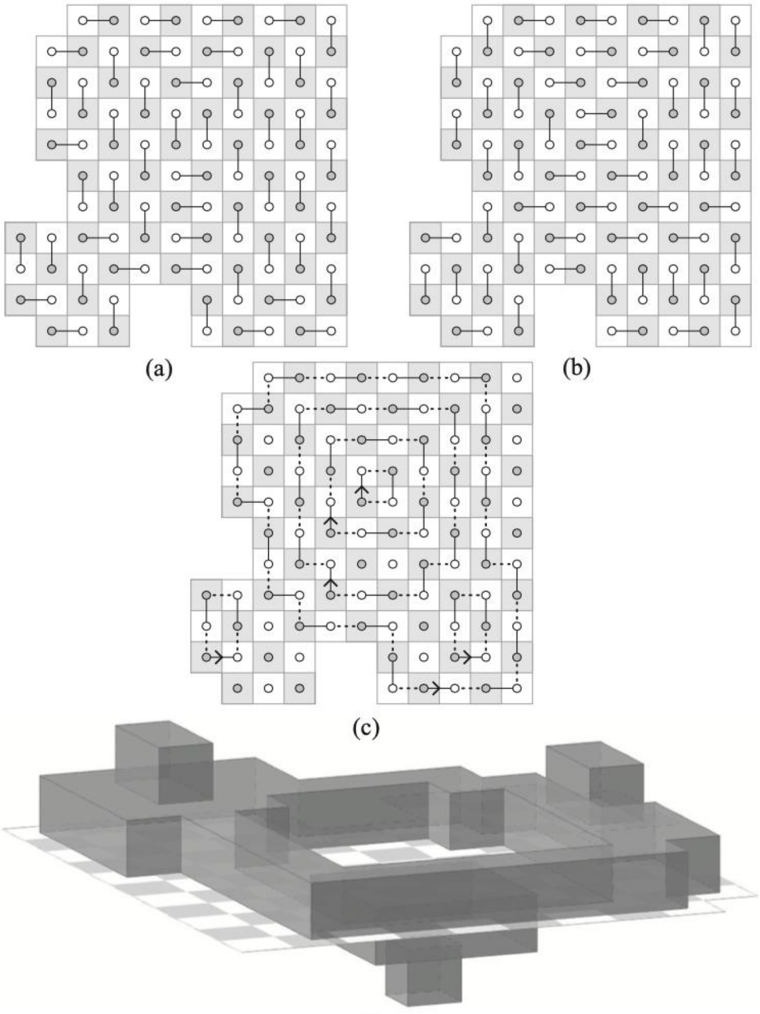
所以$dep[lca((i+1)!,i!)]=sum(maxdiv(i+1),n)$

这样我们能够顺利找到两个相邻点之间的lca

为什么不用考虑所有点呢，因为构造这棵树的时候已经是按dfs序排序了，所以考虑相邻的两个点即可

Domino

- See “Distances in Domino Flip Graphs”



(a) (b) (c) (d)

Quadratic Form

题意

$X=(x_1,x_2,\ldots,x_n)^T$，A为$n\times n$的正定二次型，b为$n\times 1$的列向量

求满足求$X^TAX\le 1$，$(X^Tb)^2$的最大的值

题解

带有不等式约束条件解极值问题，使用拉格朗日乘子法

设拉格朗日函数$L(X,\lambda)=X^Tb+\lambda\left(X^TAX-1\right)$

Quadratic Form

由KKT条件有

$$\begin{cases} \frac{\partial L}{\partial X} = b + 2\lambda AX = 0 \\ \lambda(X^T AX - 1) = 0 \\ X^T AX - 1 \le 0 \\ \lambda \le 0 \end{cases} \Rightarrow \begin{cases} b + 2\lambda AX = 0 \\ X^T AX - 1 = 0 \\ \lambda \le 0 \end{cases}$$

$\lambda(X^T AX - 1) = 0$，若$\lambda = 0$ 则$b = 0$，因此令$X^T AX - 1 = 0$

由$b + 2\lambda AX = 0$，得$X = -\frac{1}{2\lambda} A^{-1} b$

则

$$\begin{aligned} X^T AX &= 1 \\ (-\tfrac{1}{2\lambda} A^{-1} b)^T A(-\tfrac{1}{2\lambda} A^{-1} b) &= 1 \\ \tfrac{1}{4\lambda^2} b^T A^{-1} A A^{-1} b &= 1 \\ b^T A^{-1} b &= 4\lambda^2 \end{aligned}$$

Quadratic Form

$$
\begin{aligned}
X^T AX &= 1 \\
X^T(-\frac{1}{2\lambda}b) &= 1 \\
X^T b &= -2\lambda
\end{aligned}
$$

最终有 $\left(X^T b\right)^2 = 4\lambda^2 = b^T A^{-1} b$

Counting Spanning Trees

- The number of spanning trees is prod_{i >= 2} deg(x_i) deg(y_i)
- Detailed proof can be found in “Enumerative properties of Ferrers graphs” https://arxiv.org/pdf/0706.2918.pdf

Infinite String Comparision

- Compare the string a^{infty} and b^{infty} directly
- By the Periodicity Lemma, if there is no mismatches in the first a + b – gcd(a, b) characters, the two string are identical

BaXianGuoHai, GeXianShenTong

- **For simplicity, we denote the multiplication as +, and exponentiation as ***
- **Precompute B_{i, j} = 2^ {W * j} * v_i**
- **To compute sum_{i, j} (sum e'_{i, j} * 2^{W * j}) v_i**
  - = sum_x x sum_{i, j} [e'_{i, j} = x] B_{i, j} = sum_x x Q_x
- **To compute sum_x x Q_x**
  - = sum_x (sum_{y >= x} Q_y)
- **The overall complexity is O(nm / W + 2^W)**
- **Taking W = 16 yields a fast enough solution**

Minimum-cost Flow

- We denote the cost in a network with capacity c and flow f as cost(c, f).
- $cost(c, 1) = cost(c * 1/c, 1 / c) * c = cost(1, 1 / c) * c$
- For a network with unitary capacity, its cost grows linearly with the flow f, with at most O(m) pieces.
- Thus, we can compute O(m) pieces first, and query in O(log m) time.

1 or 2

- For an edge $e=(x, y)$ where $d_x = d_y = 2$, add the following edges:
  - (x, e) (x', e)
  - (y, e') (y', e')
  - (e, e')
- The problems is turned into to find a perfect matching in a general graph, which can be solved with Edmond's Algorithm.

Easy Integration

- The value is (n!)^2 / (2n+1)!
- Detailed proof can be found in “Wallis' integrals”. https://en.wikipedia.org/wiki/Wallis%27_integrals

Thanks

牛客竞赛
AC.NOWCODER.COM