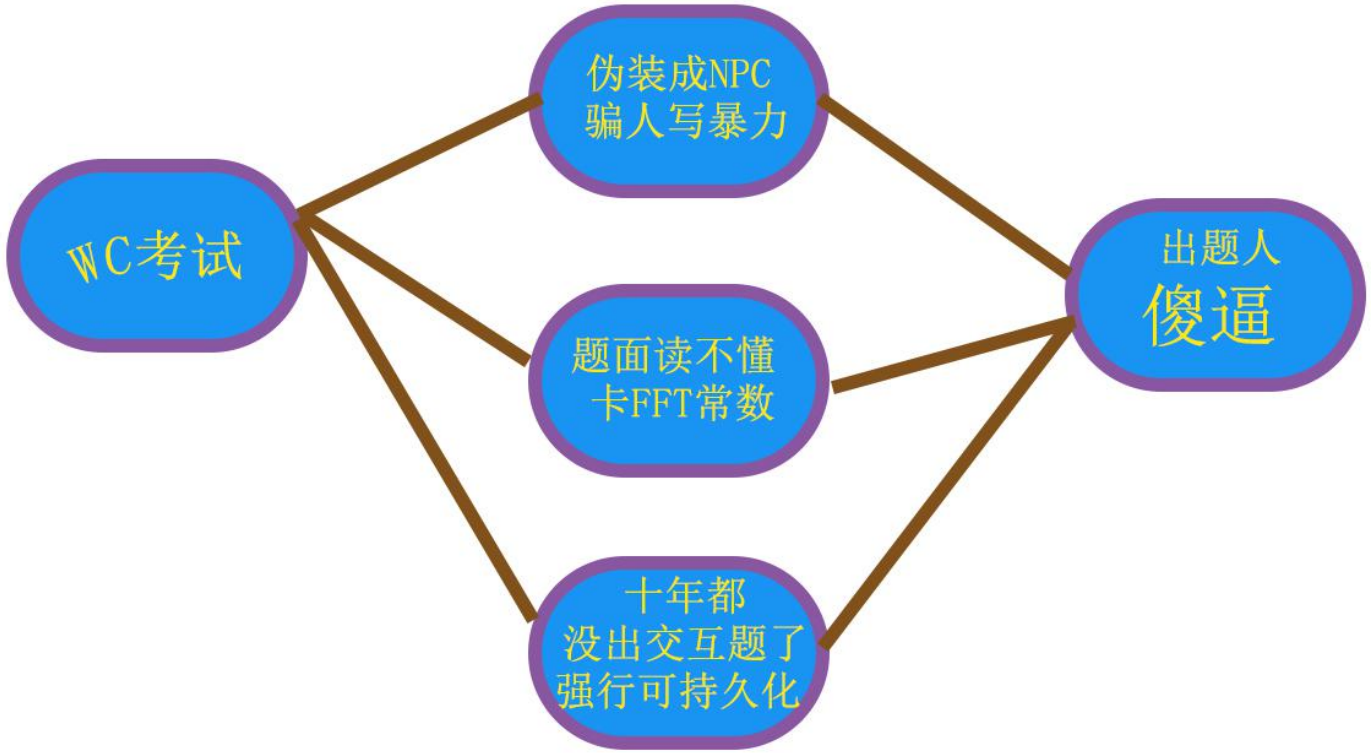
论战捆竹竿解题报告

清华大学交叉信息学院 王鉴浩

You can You up

出题人中傻逼多



冬令营里妹子少

简要题意

- 有一个仅包含小写英文字母的字符串s，s长度为n。现在把s串作为原串，然后不停在后面拼接串s来得到更长的串。
- 设目前串为t，在t后拼接串s时，可以选择一个t的后缀v，且v也要是s的前缀，来得到新串$t-v+s$，其中v可以为空串。
- 比如，$s=$aba，$t=s$。在t后拼接s可以得到 abaaba 或 ababa或 aba，其中v为空串或 a 或 aba。
- 给出串s，问可以通过s如上述拼接方式生成多少种不同**长度**的串，满足长度$\leq w$。注意，你开始的串就为s。此题共有T组数据。

数据范围

数据点	T	n	w	约束
1	5	≤ 10	≤ 10	s 仅包含字母 a 和 b
2		≤ 20	≤ 20	
3		≤ 100	$\leq 10^{18}$	无
4				
5		$\leq 10^3$		
6				
7		$\leq 5 \times 10^4$	$\leq 10^5$	
8				
9				
10				
11		$\leq 7 \times 10^4$	$\leq 10^{18}$	
12				
13		$\leq 10^5$		
14				
15				
16				
17		$\leq 5 \times 10^5$		
18				
19				
20				

得分情况

自由讨论

数据点1~2

- 直接dfs搜索所有情况，不重复搜索相同状态。时间复杂度为$O(2^w T)$。

数据点3~6

- 现在我们引入一个border的概念。对于串s，如果s的长为i的前缀和长为i的后缀相等，那么，成s长为i的前缀为s的一个border。特别，这里我们称空串和串s也为s的一个border。
- 由于题中任意一个生成的串的后缀都为s串。设原串长度为m，则对于一个长度为i的border，根据题意，可以发现，通过删去长为i这个border可以得到一个长度为$m+(n-i)$的串。此时，我们可以把问题一般化，先通过kmp算法，用线性的时间复杂度得到所有的border，把问题转化为：一开始我们的初始数字为n，共有m个border，每个border能使得串长增加a_i，问能生成多少个$\leq w$的不同的数字。
- 此题为经典问题，把数字$\bmod n$后跑最短路，得到$\bmod n$同余时最小生成的数是多少，然后计算答案，那么此刻，我们就可以通过时间复杂度为$O(Tn^2\log n)$来解决此问题了。

数据点3~6

- 事实上，这个暴力可以通过优化做到$O(n^2)$的时间复杂度。由于每个点连的边都一样，则我们一种增加border的顺序是任意的，于是，我们可以一个一个border更新最短路的值。对于一个border x，我们可以对于模x分组，每组我们可以直接通过递推来更新，于是这个解法的时间复杂度为$O(n^2)$。

数据点7~10：做法一

- 由于w很小，所以我们可以对m个a_i直接做完全背包，而这里完全背包只需要记录存在性，即记录长度k能不能被达到。
- 暴力的完全背包的时间复杂度为$O(Twn)$。
- 很自然地，我们可以选择压位来加速，通过手写bitset，设位长为v，当$a_i < v$时，暴力更新，当$a_i \geq v$时，通过压位加速。在这里位长为32，则时间复杂度为$O\left[nT\left(\frac{w}{v}+v\right)\right]$，即可解决此部分分。

数据点7~10：做法二

- 众所周知，大小为w的完全背包是可以通过分治+FFT来解决的。
- 这个做法的时间复杂度为$O(Tw \log w \log n)$。
- 但这并不是官方暴力，比较上一个算法，虽然理论复杂度好，但并没有什么卵用，代码复杂度却大大超过上一个算法，不保证能通过此类部分分。

数据点11~16

- 前面，我们都是把问题一般化之后再来解决问题。此刻，我们需要回归原问题，发掘此题的本质，来得到更优美的算法。
- 事实上，标准算法就是数据点3~6的算法的升级。
- 现在，我们先给出一个结论：border从小到大来分组，等差数列个数是$O(\log n)$的。

数据点11~16

- 现在我们来证明这个结论:现在设s串长为n，设t为s除了自己的最长的border，t的长度为m。由字符串性质可知，s的所有border就是t的所有border加上s自己。设$T(k)$为s串中长度为k能分成的等差数列的个数。
- 若$2m \le n$，则$T(n) = T(m) + 1$，那么$T(n) = O(\log n)$。
- 若$2m > n$，设p为s的长$n - m$的后缀，可以称s是以p为基的循环串。通俗地说s是$pp \ldots pp$的一个子串。这个可以由border的性质得到。设z为$n \bmod (n - m)$。那么$\{z, z + (n - m), \ldots, z + k(n - m), m\}$都是$s$的合法长度border。且由于$m$的最大性，$\{z, z + (n - m), \ldots, z + k(n - m), m\}$中间是不会有别的长度的border的，即这些border在分组的时候是可以分为一组的。
- （小证明：假设有一个border长度$l, l > z$，则可以发现$l + (n - m)$也是一个合法的border，即l可以生成一个长度大于m的border，则矛盾。）于是，$T(n) = T(n - m) + 1$，那么$T(n) = O(\log n)$。

数据点11~16

- 现在，考虑每一个等差数列，设首项为x，差为d，共y项。我们可以利用等差数列的性质来优化最短路的边。
- 首先n个数，我们讲其按模d分类，同余的为一组，对于同一组点我们构建线段树来维护。
- 对于每个点i，等差数列的边相当于，对从$(i+x) \bmod n$开始的y项进行连边，由于这是连续的一段，所以我们可以直接在线段树上连边。
- 于是，对于一个等差数列，我们用$O(n)$的点和$O\left(n \log \frac{n}{d}\right)$的边来构图。由于前面已证只有$O(\log n)$个等差数列，所以我们可以用$O(n \log n)$的点和$O(n \log n)$的边来构图。接下来，我们就可以通过经典最短路算法来解决此题。
- 由于用线段树构出的图没有负权，我们可以使用dijkstra算法在$O(n \log^2 n)$的时间复杂度内解决此题。

数据点11~16

- 另外，这张图其实可以分层跑，也就是说可以一个个等差数列分别构图跑最短路更新答案。
- 正确性证明与数据点3~6的$O(n^2)$算法类似，在那里我们是一个个border更新，这里只是加快到了一组组border更新而已。在这样分层跑的同时，如果我们不保存边，就可以做到使用$O(n)$的内存了。
  另外，分层跑的时候使用第二种构图跑spfa，会有更多的分数。

数据点17~20

- 上述算法还可以继续加强。我们注意到可以一个个等差数列更新最短路。那么一开始，模数为n。设目前这个等差数列首项为x，差为y，个数为z。
- 第一步，我们先把模数从n转为x。这个转模数是可以线性得到的。一开始，我们先把最短路放在模x下，然后通过加n来更新，就转化完毕了。
- 更新具体操作为：设$d = \gcd(x, n)$。从$0 \sim d - 1$我们可以通过加n来得到一个环的数。那么，只需要在这个循环上递推更新两边就可以了。
- 第二步，我们来更新这个等差数列，设$D = \gcd(x, y)$, 由于目前是模首项的，于是这个等差数列的更新，相当于在$0 \sim D - 1$由加y得到的环上，每个数向后$z - 1$个数进行更新。于是，这个更新只需要在环上找到权值最小的数，以这个数为开头进行一个单调队列就可以完成更新了。
- 于是，这个算法的时间复杂度就为$O(Tn \log n)$。

- 感谢CCF给我出题的机会
- 感谢超级幕后黑手杜瑜皓提供最后一个算法
- 感谢幕后黑手约大爷提供idea
- 安利一篇paper《浅谈字符串匹配的几种方法》

谢谢大家！