《圆滚滚的算术占卜》解题报告

中国人民大学附属中学　陈于思

September 2021

1　题目大意

定义关于自然数 n 的函数 $S(n)$ 为 n 的十进制表示的各位数字之和。

定义 $a \bigoplus b$ 表示两个自然数 a 和 b 的十进制表示顺次拼接的结果表示的数，例如 $123 \bigoplus 456 = 123456$ 。特别地，当 $b = 0$ 时，约定 $a \bigoplus 0 = a$ 。

令

$$f(n) = \begin{cases} 0, & n = 0 \\ n \bigoplus f(n - S(n)), & n > 0 \end{cases}$$

给定 T 组 (L, R) ，对每一组求出 $\sum_{n=L}^{R} f(n)$ ，结果对 998244353 取模。

2　数据范围

Subtask 1(10pts): $T \leq 500, R \leq 10^5$ 。

Subtask 2(10pts): $T \leq 5, L = R \leq 10^9$ 。

Subtask 3(20pts): $T \leq 500, L = R \leq 10^{12}$ 。

Subtask 4(25pts): $L = R$ 。

Subtask 5(10pts): $T \leq 500, R \leq 10^{12}$ 。

Subtask 6(25pts): 无特殊限制。

对于所有测试数据，满足 $1 \leq T \leq 5 \times 10^4, 1 \leq L \leq R \leq 10^{18}$ 。

3 解题过程

3.1 算法 1

记 $P = 998244353$ 。

计算 $(a \bigoplus b) \bmod P$ 时，只需要知道 $a \bmod P$,$b \bmod P$ 以及 b 的十进制位数，从而只需要维护 $f(n)$ 对 P 取模的结果以及位数就可以直接递推，线性预处理出前 $\max R$ 个 $f(n)$ 。再预处理前缀和，可以 $O(1)$ 回答每次询问。

复杂度 $O(T + \max R)$ ，可以通过子任务 1。

3.2 算法 2

$L = R \leq 10^9$ 时，可以不预处理每个 $f(n)$ ，而是直接根据定义递归计算 $f(L)$ ，递归次数最多只有约 2.5×10^7 次，结合算法 1 可以通过子任务 1,2。

3.3 算法 3

记 b 为进制数，w 为最大位数。在本题中，$b = 10, w = 18$ 。

考虑如何快速求一项 $f(n)$ 。

注意到 $S(n) \leq w(b-1)$ ，因此对于一段长度不小于 $w(b-1)$ 的区间 $[l, r]$ 以及 $n > r$ ，n 在不断减小的过程中一定落在 $[l, r]$ 上至少一次。

对 $n > 0$ ，如果 n 可以表示成形如 $(x+1) \times 10^k - y$ 的形式，其中 $k \geq 3$ 且 $1 \leq y \leq w(b-1)$ ，我们就称 n 是一个 “k-关键位置”。显然，所有上述 k-关键位置构成的集合是若干个长为 $w(b-1)$ 的不交区间的并，因此对其中任何一段极长区间，大于区间右端点的数在递归过程中一定会落在这段区间里。为了方便，再约定对任意的 k ，都认为 0 是一个 k-关键位置。

注意到 3-关键位置相当密集。事实上，只需要根据定义直接递归最多 80 次就可以到达一个 3-关键位置，从而只要能够预处理出关键位置的答案就可以快速求出一项的答案。

对于一个 k-关键位置 $n = (x+1) \times 10^k - y$ ，预处理出 n 第一次成为 $(k+1)$-关键位置时 n 的值 n' 以及目前拼接出的数 $g_0(k, x, y)$。

尽管 x 的数量可能很多，但 n 减小到 n' 所用的次数以及每次减小的值事实上只与 $S(x)$ 有关，并且在 x 的位数与 $S(x)$ 均固定时，$g_0(k, x, y)$ 是一

个关于 x 的一次函数。

于是，我们令 $g(k,l,s,u,y)$ 表示 x 位数为 l ，数码和为 $s+u$ ，末位为 u 时的关于 x 的一次函数 $g_0(k,x,y)$。

由于进行至少一次操作后，n 的值一定是 $b-1$ 的倍数，因此需要用到的 y 的个数是 $O(w)$ 的，于是状态数是 $O(b^2w^4)$ 的。

可以暴力求出 $k=3$ 的 g 。$k\geq 4$ 时:

1. 当 $u=0$ 时，可以用 $g(k-1,l+1,s,9,y)$ 转移到 $g(k,l,s,0,y)$ 。

2. 当 $1\leq u\leq 8$ 时，可以用 $g(k-1,l+1,s+u,9,y)$ 和 $g(k,l,s,u-1,y')$ 拼接成 $g(k,l,s,u,y)$ ，其中 y' 是 n' 对应的 y 。

3. 当 $u=9$ 时，x 本身就是一个 $(k+1)$-关键位置，g 值为 0 。

从而每次转移是 $O(1)$ 的，预处理的复杂度为 $O(b^2w^4)$ 。

查询时，先暴力转移到一个 3-关键位置，每次再从 k-关键位置向 $(k+1)$-关键位置转移，直到 n 变为 0 为止。形象地讲，这一过程实际上是从 10^3 位开始由低到高不断把每一位减到 9 ，直到无法退位时再将 n 减到 0 。于是除去最多转移 80 次的暴力部分，每次查询的复杂度是 $O(w)$ 的。

从而我们解决了 $L=R$ 的情形，结合算法 1 可以通过子任务 1,2,3,4。

3.4 算法 4

考虑如何求前 n 项 f 的和。

类似于求一项的做法，考虑预处理出每一段较高若干位固定时 f 的和。具体的，对于 $x\geq 0, 0\leq u\leq 9, k\geq 3$ ，考虑 $[10x\times 10^k,(10x+u+1)\times 10^k)$ 这些点第一次被减到小于 $10x\times 10^k$ 时被减成多少以及此时拼接出的数是多少。注意到这些数一定转移到了 $(k+1)$-关键点的位置，从而可以对每个关键点预处理转移到它的点的信息。

记 $size(k,x,u,y), h_0(k,x,u,y)$ 分别为转移到 $10x\times 10^k-y$ 的数的个数以及这些数对应的拼接出的数之和。类似于求一项，当 x 的位数、数码和固定时，$size$ 是一个常数且 h_0 是关于 x 的一次函数。转移与求一项是类似的，预处理部分复杂度为 $O(b^2w^4)$ 。

查询时，将区间 $[1,n]$ 最后 $(n+1) \bmod 10^3$ 个数用求一项的方法单独计算，并将区间的余下部分拆成至多 w 个形如 $[10x\times 10^k,(10x+u+1)\times 10^k)$ 的区间，每个这样的区间只会转移到至多 w 个不同的位置，对这些位置用求一项的方法单独计算。

于是每次查询需要调用 b^3+w^2 次算法 3 中求单项的方法，复杂度是 $O(w(b^3+w^2))$ ，结合算法 3 可以通过子任务 1,2,3,4,5。

3.5 算法 5

考虑一种线性求前缀和的方法。构造一棵以 0 为根的，结点编号为 0 到 n 的树，使 i 的父亲为 $i-S(i)$ 。在这颗树上 dp，对每个结点 v ，记 F_v 为子树中所有数减到 v 时拼接出的数的和，于是 $F_v=$ 所有儿子 F 之和 $\times 10^{v\text{的位数}}+v\times(v\text{的子树大小})$ 。

考虑基于上述算法优化算法 4。对于算法 4 中需要单独求解的 $O(b^3+w^2)$ 项，建立这些点的虚树。

对于较小的 $O(w^2)$ 个由连续区间转移到的项，它们在虚树上到父亲的边的信息恰好是求单项时预处理过的；对于较大的 $(n+1) \bmod 10^3$ 项，它们在虚树上的父亲就是在原树上的父亲，边的信息也是易于计算的。于是，可以类似于上文树形 dp 的方法在虚树上 dp，每次查询的复杂度是 $O(b^3+w^2)$ ，已经足以通过本题。实现时，并不需要显式建出虚树，只需要倒序扫描每个点并更新父亲信息即可。

当然，查询复杂度还可以进一步优化。注意到如果两个数仅有末位不同，那么它们的父亲是相同的。于是最后的 $(n+1) \bmod 10^3$ 项每连续 10 项都可以合并计算，复杂度可以做到 $O(b^2+w^2)$ 。

最终总的复杂度是 $O(b^2w^4+T(b^2+w^2))$ 或 $O(b^2w^4+T(b^3+w^2))$，可以通过本题。

4 参考资料

本题没有参考资料。

集训队队员许庭强参与了本题的验题工作。