解题报告

题目大意

求长度为 n、满足 $\forall i \in [0,n), 0 \le a_i \le R_i$ 且 $\forall x \ge 0, \left(\sum_{i=0}^{n-1} (\lfloor \frac{a_i}{2^x} \rfloor \bmod 2) \cdot 2^i\right) \in S$ 的非负整数数组 a 的取值方案数对 998244353 取模的结果。

数据范围

所有数据满足 $1 \le n \le 18, 0 \le R_i < 2^{60}, 0 \in S$。

解题过程

注意到第二个限制只和所有数的同一个二进制位上的值有关，所以可以考虑一位一位地填数。

考虑从高位向低位填所有数。当前每个数有两种状态：

- 当前数填的高位与上界高位不同，意味着此时这个数剩下的位填数不受上界限制；
- 当前数填的高位与上界高位相同，受到上界限制。

很快想到可以使用状压DP，设 $dp_{i,S}$ 表示填了 2^i 及之上的位，当前位所有数的状态为 S 的方案数。

一种直接的转移方式是枚举下一位上所有数填的值，这样的复杂度是 $O(n4^n \log R)$ 或者 $O(4^n \log R)$。

注意到第一种状态填数不受上界限制，所以做了预处理过后就只有一种转移，配合位运算技巧优化可以做到 $O(3^n \log R)$。

接下来尝试使用 FWT 进行优化。

不妨在DP状态里用`0`表示高位不同，`1`表示高位相同，然后用 t 表示下一位上所有数填的值的状态（同样压缩成二进制），

那么有两种转移方式：

- 上界 R_i 在这一位上是`0`：

  ```
  0 * 0 → 0
  0 * 1 → 0
  1 * 0 → 1
  ```

  上面的第一列表示DP状态里第 i 个数的状态，第二列表示 t 中第 i 个数填的值。

  原本 t 中第 i 个数的状态`0`和`1`就表示二进制位上填的值，此时我们将这个数的状态做一些处理，用`1`表示原来的状态`0`，用`0`表示原来状态`0`和状态`1`的和，那么就变成了一个普通点积。

- R_i 在这一位上是`1`：

  ```
  0 * 0 → 0
  0 * 1 → 0
  1 * 0 → 0
  1 * 1 → 1
  ```

  容易发现这个转移正好是我们熟悉的按位与卷积。

众所周知 FWT 可以将按位与卷积转化成点积计算，所以我们对上界处为 `0` 的数位做完状态处理之后，可以直接对上界处为 `1` 的部分数位做 FWT，最后全部转化为点积。

观察第一步的状态处理和第二步的 FWT，从线性变换的角度，我们知道每一位的 FWT 之间是满足交换律的。然后可以发现每一位的状态处理以及这两种线性变换之间也满足交换律，所以可以把这两步穿插在一次循环里做完。

这样的复杂度是 $O(n2^n \log R)$，并且不卡常。

参考资料

- OI Wiki，快速沃尔什变换