通过解码有界权重德布鲁因序列，解码 t-子集和 t-多重集的通用循环

Daniel Gabri´c圭尔夫大学，加拿大
Wazed Imam圭尔夫大学，加拿大
Lukas Janik-Jones圭尔夫大学，加拿大
Joe Sawada圭尔夫大学，加拿大

arXiv:2603.11934v1 [cs.DM] 12 Mar 2026

摘要

集合**S** 中的组合对象的通用循环，是一个长度为 |**S**| 的循环序列，其中每个代表都作为子串恰好出现一次。尽管文献中已知针对多种集合的许多通用循环构造，包括长度为 n的 k元字符串、 n阶排列、 n元集合的 t-子集，以及 n元集合的 t-多重集，但令人惊讶的是，只有极少数构造具有高效的解码（排名/反排名）算法。本文针对字母表大小为 k、长度为 n 的字符串的有界权重德布鲁因序列，开发了首批多项式时间/空间解码算法。随后，我们将这些结果用于解码 t-子集和 t-多重集的通用循环。

1 引言

一个通用循环是集合**S** 中组合对象的一个长度为 |**S**| 的循环序列，其中包含表示，使集合**S** 中的每个元素恰好作为一个子串出现一次。除非另有说明，假设每个元素的代表**s** $\in$ **S** 就是**s** 本身。若**s** 不是字符串，则会明确说明其代表。对于任意集合**S** [CDG92]，可以提出以下“基本问题”：

> (i) 是否存在针对*S*的通用循环？(ii) 针对**S**的通用循环有多少个？(iii) 能否（高效地）构造针对**S** 的通用循环？(iv) 是否存在具有某些性质、从而可以高效解码的针对**S** 的通用循环？换言之，针对**S**的某个特定通用循环，是否存在高效的排名算法和排名算法？

关于前面三个问题，针对各种有趣集合**S**，已有大量研究成果 [Fre82、CDG92、JSH09、BG11、Etz24、dbs25]。然而，解码问题一直更难以解决。

令 $\mathsf{S}_k(n)$ 表示由 {1, 2,. . ., k} 中的符号构成的所有长度为 n 的字符串的集合。$\mathsf{S}_k(n)$ 的通用循环称为德布鲁因序列。字符串的权重是其符号之和。令 $\mathsf{S}_k(n, w^{\uparrow})$ 表示 $\mathsf{S}_k(n)$ 中权重至少为 w 的字符串所构成的子集。令 $\mathsf{S}_k(n,w_{\downarrow})$ 表示 $\mathsf{S}_k(n)$ 中权重至多为 w 的字符串所构成的子集。$\mathsf{S}_k(n, w^{\uparrow})$ 和 $\mathsf{S}_k(n,w_{\downarrow})$ 的通用循环称为有界权重德布鲁因序列。本文提出针对有界权重德布鲁因序列的高效排名和反排名算法。这些算法可用于高效解码 t-子集和 t-多重集的通用循环。据我们所知，这是首批能够高效解码这些对象的特定通用循环的算法。

2 通用循环解码

通用循环解码。令 U $= u_1 u_2 \cdots u_m$ 为集合 S（由长度为 n 的字符串组成）的一个通用循环。字符串 s $\in$ S 的秩是 s 在 U 中出现时的起始索引。若 s 是 U 的前缀，则其秩为 1；若 s 从 U 的最后一个符号开始并绕回开头，则其秩为 |S|。给定秩 r，反排名过程会确定 U 中从索引 r 开始的字符串 s $\in$ S。排名和反排名算法结合起来，便可解码通用循环。通过遍历序列，直到找到给定字符串或秩，即可解码任何通用循环；然而，这需要 $O(|S|)$ 时间。同样，也可以使用查找表存储给定字符串的位置，但这需要 $\Theta(|S|)$ 空间。如果排名和反排名算法的运行时间为多项式时间，且所需空间为多项式空间（相对于字母表大小和每个字符串 s 的长度），我们就称解码算法是高效的。在机器人位置感知（视觉）应用 [BM93, DG11] 中，高效解码通用循环是必不可少的。

目前已有许多德布鲁因序列构造方法，采用了各种不同的算法思路；例如，参见 [Fre82, Etz24, dbs25]。然而，颇令人意外的是，目前已知只有 $S_k(n)$ 的字典序最小德布鲁因序列能够对所有 n 和 k [高效解码 KRR16, SW17]。我们将在第 3 节回顾这一德布鲁因序列的解码方法。递归构造的德布鲁因序列在二进制情形下可以高效解码，但并非对所有 n 和 k [都能如此 MEP96, Tul01]。另见 [Knu11, p.317, ex.97-99]。目前已知的其他高效解码算法，仅适用于采用简写表示的排列通用循环 [RW10, HRW12]。这凸显了发现此类解码算法的困难。

t-子集和 t-多重集。令 Subset(n, t) 表示所有 t-子集组成的集合，其中元素取自 {1, 2, . . . , n}。人们曾使用每个子集的标准字符串表示来研究 Subset(n, t) 的通用循环；例如，子集 {1, 2} 可用 12 或 21 表示 [CDG92, Jac93, Hur94, JSH09, Rud13]。使用这种表示时，只有当 n 整除 $\binom{n}{t}$ 时，t-子集的通用循环才存在。t-子集也可以表示为长度为 n 、含有 t 个 1 的二进制字符串。简写表示会省略末尾冗余的位。采用这种简写表示，对于所有 $1 \le t \le n$ [，都可以高效构造 t-子集的通用循环 RSW12]。目前尚无适用于这些通用循环的高效解码算法。在 [CGGP21], 中，使用标记图来表示 t-子集。借助这种表示，他们证明了 t-子集的通用循环存在，但没有给出高效构造方法。[CJJS25] 最近介绍了一种差分表示。给定一个元素按递增顺序排列的子集 $\{s_1, s_2, \ldots, s_t\}$，其差分代表元是一个长度为 t 的字符串，其中第一个符号为 s_1，第 j 个符号为 $s_j - s_{j-1}$，其中 $1 < j \le t$。例如，Subset(5, 3) 的差分代表元为 =

$$\{\{1,2,3\},\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,3,5\},\{1,4,5\},\{2,3,4\},\{2,3,5\},\{2,4,5\},\{3,4,5\}\}$$

分别表示为 111、112、113、121、122、131、211、212、221、和 311。一般而言，**子集**(n、t) 的差分代表元对应于 $\mathbf{S}_{n-t+1}(t、n\downarrow)$ 中的字符串。

令 Multiset(n, t) 表示所有 t-多重集的集合，其中元素取自 {0, 1, . . . , n − 1}。人们曾使用每个子集的标准字符串表示来研究 Multiset(n, t) 的通用循环；例如，多重集 {1, 1, 2} 可以用 112、121 或 211 中的任意一个来表示 [Jac93, HJZ09]。采用这种表示时，t-多重集的通用循环仅在 n 整除 $\binom{n+t-1}{t}$ 时存在。与 t- 子集类似，我们也可以对 t- 多重集应用差分表示 [CJJS25]。给定一个元素按非递减顺序排列的多重集 $\{m_1, m_2, \ldots, m_t\}$，其差分代表是一个长度为 t 的字符串，其中第一个符号为 m_1，第 j 个符号为 $m_j - m_{j-1}$，对于 $1 < j \le t$。例如，Subset(3, 3) 的差分代表元为 =

$$\{\{0,0,0\},\{0,0,1\},\{0,0,2\},\{0,1,1\},\{0,1,2\},\{0,2,2\},\{1,1,1\},\{1,1,2\},\{1,2,2\},\{2,2,2\}\}$$

分别为 000， 001， 002， 010， 011，020， 100， 101，110， 和 200。将差分代表元中每个符号 x 替换为 x + 1，对于 **多重集** (n， t)，所得字符串集合为 $\mathbf{S}_n(t,\ (n+t-1)\downarrow)$。

主要结果。 本文的主要结果如下：

1. 我们证明，对于 $\mathbf{S}_k(n, w^{\uparrow})$ ，字典序最小通用循环（有界权重德布鲁因序列）可以高效解码。**2.** 我们证明， t-子集的通用循环可以高效解码。**3.** 我们证明， t-多重集的通用循环可以高效解码。

正如我们将在本文后文提到的，针对 $\mathbf{S}_k(n, w^{\uparrow})$ 的可解码通用循环，可以轻松用于解码 $\mathbf{S}_k(n, w_{\downarrow})$ 中字符串的通用循环。本文所述所有算法的分析均假设采用单位成本 RAM 计算模型。我们解码算法的 C 语言实现可在以下网址获取：`https://debruijnsequence.org/db/subset_解码`。

概要。 本文其余部分安排如下。第 2 节介绍背景定义和符号。第 3 节回顾字典序最小德布鲁因序列的高效解码方法。第 4 节通过考虑字符串权重的下界来推广这些结果；第 5 节考虑权重的上界。第 6节将这些结果用于证明 t-子集和 t-多重集存在可高效解码的通用循环。第 7节证明技术性结果。

2 预备知识

回忆一下，$\mathrm{S}_k(n)$ 表示所有由 $\{1, 2, \ldots, \mathrm{k}\}$ 中的符号构成的长度为 n 的字符串的集合。考虑两个字符串 s 和 t。令 $\mathrm{s} \cdot \mathrm{t}$ 表示 s 与 t 的连接，令 t^j 表示将 j 个 t 连接在一起。项链是字符串在旋转下所属等价类中字典序最小的字符串。令 neck(s) 表示 s 的字典序最小旋转，即它的项链。令 ap(s) 表示 s 的非周期前缀，即满足 $\mathrm{s} = \mathrm{t}^j$ 的某个 $j \geq 1$ 所对应的最短字符串 t。例如，$\mathrm{ap}(1111) = 1$。如果 $\mathrm{ap}(\mathrm{s}) = \mathrm{s}$，则称字符串 s 是非周期的（本原的）；否则，s 是周期的（一个幂）。

令 $\mathbf{N}_k(n) = (\sigma_1, \sigma_2, \ldots, \sigma_t)$ 表示由 $\mathbf{S}_k(n)$ 中所有项链按字典序排列组成的元组。例如，

$\mathbf{N}_2(4) = (1111, 1112, 1122, 1212, 1222, 2222)$。

令人惊讶的是，将$\mathbf{N}_k(n)$ [FM78] 中各项链的非周期前缀连接起来，就得到了$\mathbf{S}_k(n)$ [FM78，Fre70]的字典序最小德布鲁因序列。设

$$\mathcal{G}_k(n) = \mathrm{ap}(\sigma_1)\,\mathrm{ap}(\sigma_2)\cdots\mathrm{ap}(\sigma_t),$$

表示 $S_k(n)$ 的字典序最小德布鲁因序列。例如，

$$\mathcal{G}_2(4) = 1 \cdot 1112 \cdot 1122 \cdot 12 \cdot 1222 \cdot 2.$$

序列 $\mathcal{G}_k(n)$ 也称为*Granddaddy*[1] 德布鲁因序列和*Ford* 序列[2]。

关于记号的说明。本文用粗体小写字母（如 $\mathbf{r}$、$\mathbf{s}$、$\mathbf{t}$）表示任意字符串，而用希腊字母（如 α、β、σ、γ）表示项链。

[1] 这个名称最初用于指 Martin 的优先取最大值的贪心构造 [Mar34]，由 Knuth [Knu11，第317]页所述。它通过对字母表符号进行简单映射，等价于 $\mathcal{G}_k(n)$。[2] 根据 Lester Ford 于 1957年发布的一份技术报告，他独立地通过一种等价的贪心构造发现了该序列。

2.1 界定权重

字符串 s 的权重记作 weight(s)，定义为其各符号权重之和。回顾一下，$\mathrm{S}_k(n, w^{\uparrow})$ 表示 $\mathrm{S}_k(n)$ 中权重至少为 w 的字符串所组成的子集。令 $\mathrm{N}_k(n, w^{\uparrow}) = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ 表示由 $\mathrm{S}_k(n, w^{\uparrow})$ 中所有项链按字典序排列构成的元组。例如，

$$\mathbf{N}_2(4, 6^{\uparrow}) = (1122, 1212, 1222, 2222).$$

值得注意的是，Granddaddy 构造可以推广到 $\mathbf{S}_k(n, w^{\uparrow})$。设

$$\mathcal{G}_k(n, w^{\uparrow}) = \mathrm{ap}(\alpha_1)\,\mathrm{ap}(\alpha_2)\cdots\mathrm{ap}(\alpha_m).$$

序列 $\mathcal{G}_k(n, w^{\uparrow})$ 是字典序最小的有界权重德布鲁因序列，对应于 $\mathbf{S}_k(n, w^{\uparrow})$ [SWW14,SWW16]。例如，

$$\mathcal{G}_2(4, 6^{\uparrow}) = 1122 \cdot 12 \cdot 1222 \cdot 2.$$

不幸的是，将 Granddaddy 构造改造为适用于具有权重上界的字符串，并不一定得到通用循环。例如，序列 $1 \cdot 1112 \cdot 1122 \cdot 12$ 并不是 $\mathbf{S}_2(4, 6_{\downarrow})$ 的通用循环：字符串 2211 缺失。

还有若干其他已知的有界权重德布鲁因序列构造（例如，参见 [SWW14, SWW16, GSWW20, SSTW24]）；然而，此前并不知道它们的高效解码算法。对于$\mathbf{S}_k(n)$ 中具有权重上下界的子集，也已知高效通用循环 [SWW13]。

2.2 项链性质

The 根据项链的定义，可以推导出项链的以下五个性质。

▶ **性质 1.** 若 $\alpha = \mathtt{a}_1 \cdots \mathtt{a}_{j-1}\mathtt{x}\mathtt{k}^{n-j}$ 属于$\mathbf{N}_k(n)$，其中 $j \geq 1$ 且符号为 $\mathtt{x} \neq \mathtt{k}$，则 $\mathtt{a}_1 \cdots \mathtt{a}_j(\mathtt{x}+1)\,\mathtt{k}^{n-j}$ 也属于项链。

▶ **性质 2.** 设 n、$k > 1$ 和 $k < w < kn$。$\mathbf{N}_k(n, w^{\uparrow})$ 中的第一个项链是 $1^{n-j-1}\mathtt{x}\mathtt{k}^j$，其中 $j = \lfloor \frac{w-n}{k-1} \rfloor$ 且 $\mathtt{x} = w - (n-j-1) - kj$；其权重为 w，且是非周期的。

▶ **性质 3.** 设 n、$k > 1$ 和 $w < kn$。$\mathbf{N}_k(n, w^{\uparrow})$ 中的最后两个项链是$(\mathtt{k}-1)\mathtt{k}^{n-1}$ 和 $\mathtt{k}^n$。

下一个性质可由周期项链的定义推出。

▶ **性质 4.** 设 α_i 和 α_{i+1} 是$\mathbf{N}_k(n, w^{\uparrow})$ 中的连续项链，其中 n、$k > 1$ 和 $w < kn$。若 α_i 是周期的，则 $\mathrm{ap}(\alpha_i)\cdot\mathrm{ap}(\alpha_{i+1})$ 以 α_i 为前缀。此外，α_{i+1} 是非周期的。

证明。由于 α_i 是周期的，它可以写成 $\alpha_i = (\mathtt{a}_1 \cdots \mathtt{a}_p)^j$，其中 $j > 1$满足某些条件。设 ℓ 是$[1, p]$ 中满足 $\mathtt{a}_\ell \neq \mathtt{k}$的最大索引。这样的索引存在，因为 α_i 不是 $\mathrm{N}_k(n, w^{\uparrow})$ 中最后一条项链 $\mathtt{k}^n$。根据性质 1，存在一条以 $(\mathtt{a}_1 \cdots \mathtt{a}_p)^{j-1}$ 为前缀的项链，它在字典序上大于 α_i，且也属于 $\mathrm{N}_k(n, w^{\uparrow})$。因此，由于 $\mathrm{N}_k(n, w^{\uparrow})$ 按字典序排列，α_{i+1} 也必定以 $(\mathtt{a}_1 \cdots \mathtt{a}_p)^{j-1}$ 为前缀。因此，$\mathrm{ap}(\alpha_i) \cdot \mathrm{ap}(\alpha_{i+1})$ 以 α_i为前缀。α_{i+1} 是非周期的这一事实已在 [SWW16, 引理 5]中证明。◀

下述性质证明见 [SW17]。

▶ **性质 5.** 设 $\alpha = \mathtt{a}_1\mathtt{a}_2 \cdots \mathtt{a}_n$ 是一条项链，且设 $1 \leq i \leq j \leq n$。那么，当 $\mathtt{x} > \mathtt{a}_j$时，$\mathtt{a}_i\mathtt{a}_{i+1} \cdots \mathtt{a}_{j-1}\mathtt{x}$ 的字典序大于 α。

3 Granddaddy de Bruijn 序列 $\mathcal{G}_k(n)$的排名

本节回顾如何基于 Kociumaka、Radoszewski 和 Rytter 的突破性成果，高效地解码 Granddaddy de Bruijn 序列 $\mathcal{G}_k(n)$ KRR [KRR16]。随后在第 4节，我们调整这一方法，以解码有界权重德布鲁因序列 $\mathcal{G}_k(n, w^{\uparrow})$。

设$\mathbf{s}$ 是$\mathbf{S}_k(n)$中的一个字符串。它可以写成$\mathbf{s} = \mathbf{pq}$，其中$\mathbf{q}$ 是$\mathbf{s}$ 的最长后缀，且$\mathbf{qp}$是一个项链。例如，若$\mathbf{s} = 221221$，则$\mathbf{p} = 22$ 且$\mathbf{q} = 1221$。令 $\mathbf{rank(s)}$ 表示$\mathbf{s}$ 在$\mathcal{G}_k(n)$中的秩。$\mathcal{G}_k(n)$中最后的 n 个长度为 n 的子串都是字符串 $\mathbf{s} = \mathtt{k}^{n-j}\mathtt{1}^j$，其中 $0 \le j < n$。对于这些字符串，$\mathbf{rank(s)} = k^n - (n - j) + 1$。对于所有其他字符串，高效计算 $\mathbf{rank(s)}$ 涉及两个关键步骤。

步骤 1： 确定连续项链 β_1, β_2和 β_3 在$\mathbf{N}_k(n)$ 中，使得$\mathrm{ap}(\beta_1)\mathrm{ap}(\beta_2)\mathrm{ap}(\beta_3)$ 包含$\mathbf{s}$ 作为子串。以下备注总结了 [FM78]的结果。

▶ 注 6。存在唯一的连续项链 β_1、 β_2 和 β_3，它们属于$\mathbf{N}_k(n)$（按循环方式考虑），满足$\mathbf{p}$ 是 β_1 的后缀，且$\mathbf{q}$ 是 β_2的前缀；也就是说，$\mathbf{s}$ 是 $\beta_1\beta_2$的子串。此外，$\mathbf{s}$ 是 $\mathrm{ap}(\beta_1)\mathrm{ap}(\beta_2)\ \mathrm{ap}(\beta_3)$ 的子串。

上述注释中的项链 β_1、 β_2和 β_3 可在$\mathcal{O}(n^2)$ 时间内，使用$\mathcal{O}(n)$ 空间 [KRR16]确定。

步骤 2： 计算 $T_k(n, \sigma_i) = |\mathrm{ap}(\sigma_1)\ \mathrm{ap}(\sigma_2) \cdots \mathrm{ap}(\sigma_{i-1})|$，它统计$\mathbf{S}_k(n)$中项链在字典序上小于 σ_i的字符串数量。例如，

$$T_2(4, \mathtt{1122}) = |\{\mathtt{1111}\}| + |\{\mathtt{1112}, \mathtt{1121}, \mathtt{1211}, \mathtt{2111}\}| = 5.$$

值 $T_k(n, \sigma_i)$ 可在$\mathcal{O}(n^2)$ 时间内，使用$\mathcal{O}(n^2)$ 空间 [KRR16]计算。

将这两个步骤结合起来，

$$\mathrm{rank}(\mathbf{s}) = \begin{cases} k^n - (n-j) + 1, & \text{if } \mathbf{s} = \mathtt{k}^{n-j}\mathtt{1}^j \text{ for some } 0 \le j < n; \\ T_k(n, \beta_2), -|\mathbf{p}| + 1, & \text{otherwise.} \end{cases}$$

例 1 回顾 $\mathcal{G}_2(4) = 1111211221212222$。考虑 $\mathbf{s} = 2112$。它是 $\beta_1 = 1112$ 与 $\beta_2 = 1122$的连接串中的一个子串，其中 $\mathbf{p} = 2$ 且$\mathbf{q} = 112$。由于 $T_2(4,1122) = 5$，$\mathbf{s}$ 的秩为$5 - 1 + 1 = 5$。2211 的秩为 $2^4 - 2 + 1 = 15$。

在 $\mathcal{G}_k(n)$ 中秩为 r 的字符串可通过二分查找并反复应用排名算法 [KRR16]，在 $\mathcal{O}(n^3 \log k)$ 时间内确定。另请参阅 [SW17]中对这些结果的另一种解读。

4 解码有界权重德布鲁因序列 $\mathcal{G}_k(n, w^{\uparrow})$

本节将 Granddaddy 的解码算法改编用于解码有界权重德布鲁因序列 $\mathcal{G}_k(n, w^{\uparrow})$。若 $w = k$，则 $\mathcal{G}_k(n, w^{\uparrow}) = \mathcal{G}_k(n)$。若 $w = kn$，则 $\mathcal{G}_k(n, w^{\uparrow})$ 是单个符号 $\mathtt{k}$。在本节余下部分中，假设 $k < w < kn$，并令：

- $\alpha_1 = \mathtt{a}_1\mathtt{a}_2 \cdots \mathtt{a}_n$ 是 $\mathbf{N}_k(n, w^{\uparrow})$ 中的第一个项链，$\mathbf{s} = \mathbf{pq}$ 是 $\mathbf{S}_k(n, w^{\uparrow})$ 中的一个字符串，其中 $\mathbf{q}$
- 是 $\mathbf{s}$ 的最长后缀，使得 $\mathbf{qp}$ 是一个项链，并且β_1 和 β_2 是备注 6 中定义的项链，对应于 $\mathbf{s}$。
-

回忆一下， β_1 以**p** 为后缀，而 β_2 以 **q** 为前缀。

为调整上一节中的**步骤 1**，请注意，项链 β_1 和 β_2 可能不满足所需的权重约束。这一点将在以下示例中说明。

例2 考虑 $n = 3$、 $k = 4$和 $w = 9$。由于

$$\mathbf{N}_4(3, 9^{\uparrow}) = 144,\ 234,\ 243,\ 244,\ 333,\ 334,\ 344,\ 444,$$

我们有

$$\mathcal{G}_4(3, 9^{\uparrow}) = 144 \cdot 234 \cdot 243 \cdot 244 \cdot 3 \cdot 334 \cdot 344 \cdot 4.$$

考虑 $\mathbf{s} = 423$，其中 $\mathbf{p} = 4$ 且 $\mathbf{q} = 23$。在 $\mathcal{G}_4(3)$ 中，$\mathbf{s}$ 是 $\beta_1 \cdot \beta_2 = 224 \cdot 233$ 的子串。然而，在 $\mathcal{G}_4(3, 9^{\uparrow})$ 中，它是 $144 \cdot 234$ 的子串。 ——

为了解决这一难题，首先考虑 $G_k(n, w^{\uparrow})$ 回绕处的字符串。回顾性质 2， α_1 是非周期的。因此， α_1 是 $G_k(n, w^{\uparrow})$ 的前缀。根据性质 3， $\alpha_{m-1} = (k-1)k^{n-1}$ 和 $\alpha_m = k^n$。因此，$G_k(n, w^{\uparrow})$ 以后缀 k^n 结尾。因此，若 s 的形式为 $k^{n-j}a_1 \cdots a_j$，其中某个 $0 \le j < n$，则它会作为 $G_k(n, w^{\uparrow})$ 最后 n 个子串之一出现。此类字符串 s 的秩为 $|S_k(n, w^{\uparrow})| - (n - j) + 1$。对于所有其他可能的字符串 s，考虑项链 δ_1、 δ_2和 δ_3，使得

- δ_1 是$N_k(n, w^{\uparrow})$中按字典序小于或等于 β_1的**最大**项链，δ_2 是$\mathbf{N}_k(n, w^{\uparrow})$中按字典序大于或
- 等于 β_2的最小项链，而δ_3 是在$\mathbf{N}_k(n, w^{\uparrow})$中紧随 δ_2 之后的项链。
-

根据定义， δ_1、 δ_2和 δ_3 在$\mathbf{N}_k(n, w^{\uparrow})$中连续出现。由于对**s** 和 w的限制，项链 δ_2 和 δ_3 是定义良好的。 δ_1 定义良好的这一事实可由下述引理的证明得出。最终，我们将证明**s** $= \mathbf{pq}$ 作为 $\mathrm{ap}(\delta_1)\ \mathrm{ap}(\delta_2)\ \mathrm{ap}(\delta_3)$ 的子串出现在$\mathcal{G}_k(n, w^{\uparrow})$中。

►**引理7**。 字符串$\mathrm{ap}(\delta_1)$ 以后缀**p**结尾。

证明。首先，我们证明 δ_1 以后缀 p 结尾。回忆一下，项链 β_1 以后缀 p 结尾。令 $j = |\mathbf{p}|$，并令 $\beta_1 = b_1 \cdots b_{n-j}\mathbf{p}$。若 $\mathbf{p} \ne k^j$，则根据性质 1， β_2 必定也以前缀 $b_1 \cdots b_{n-j}$ 开头；根据 β_2 的定义，这等于 q。因此， $\beta_1 = \mathbf{qp} \in \mathbf{N}_k(n, w^{\uparrow})$。于是根据 δ_1的定义，我们有 $\delta_1 = \beta_1$，这意味着 δ_1 以后缀 p 结尾。现在假设 $\mathbf{p} = k^j$。若 $\beta_1 \in \mathbf{N}_k(n, w^{\uparrow})$，则 $\delta_1 = \beta_1$，因而 δ_1 以后缀 p 结尾。否则， $\delta_1 < \beta_1$且 $\beta_1 \notin \mathbf{N}_k(n, w^{\uparrow})$。反设 δ_1 没有以 p 为后缀。若 δ_1 的长度为-$n - j$的前缀是 $b_1 \cdots b_{n-j}$，则 β_1 的权重大于 δ_1的权重，矛盾。若 δ_1 的长度为-$n - j$的前缀不是 $b_1 \cdots b_{n-j}$，则它必定小于 $b_1 \cdots b_{n-j}$。但根据性质 1，$\mathbf{N}_k(n, w^{\uparrow})$中存在一条项链，它大于 δ_1 但小于 β_1，矛盾。因此， δ_1 以后缀 p 结尾。

现在我们证明 $\mathrm{ap}(\delta_1)$ 以 p 为后缀。如果 δ_1 是非周期的，则我们已经证明 p 是 $\mathrm{ap}(\delta_1) = \delta_1$的后缀。如果 δ_1 是周期，则对某个 $j > 1$，它可以写成 $\delta_1 = \gamma^j$，其中 $\mathrm{ap}(\delta_1) = \gamma$。如果 $|\mathbf{p}| \le |\gamma|$，则 p 是 $\mathrm{ap}(\delta_1)$的后缀。否则，由性质 4，$\mathrm{ap}(\delta_1)\mathrm{ap}(\delta_2)$ 以 δ_1 为前缀。这意味着 $\mathrm{ap}(\delta_2)$ 以 γ^{j-1}开头。由于 $|\gamma| < |\mathbf{p}|$，我们有 $|\mathbf{q}| = |\delta_1| - |\mathbf{p}| < |\delta_1| - |\gamma| = |\gamma^{j-1}|$，这意味着 q 是 $|\gamma^{j-1}|$的前缀，因而也是 δ_1的前缀。但这样一来 $\delta_1 = \mathbf{qp}$，且 δ_1 是 s 的循环移位，这意味着 s 是周期，且与 δ_1具有相同的周期。这与 q 是 s 的使 qp 为项链的最长后缀相矛盾。因此 p 是 $\mathrm{ap}(\delta_1)$ 的后缀。◄

► **引理 8.** 字符串 $\mathrm{ap}(\delta_2)\mathrm{ap}(\delta_3)$ 以 **q** 为前缀。

证明。首先，我们证明 δ_2 以 $\mathbf{q}$ 为前缀。回想一下， β_2 以 $\mathbf{q}$ 为前缀。设 $j = |\mathbf{p}|$。由于 $\mathbf{s} = \mathbf{pq}$ 属于 $\mathsf{S}_k(n, w^\uparrow)$，因此 $\mathbf{q}\mathtt{k}^j$ 也属于 $\mathsf{S}_k(n, w^\uparrow)$。由于 $\mathbf{qp}$ 是一个项链，根据性质 1 的反复应用，$\mathbf{q}\mathtt{k}^j$ 也是一个项链。因此，由于 δ_2 是 $\mathbf{N}_k(n, w^\uparrow)$ 中字典序大于 β_2 的最小项链， δ_2必定以 $\mathbf{q}$ 为前缀。

现在证明 $\mathrm{ap}(\delta_2)\,\mathrm{ap}(\delta_3)$ 以 $\mathbf{q}$ 为前缀。如果 δ_2 是非周期的，那么显然 $\mathbf{q}$ 是 $\mathrm{ap}(\delta_2)$ 的前缀。如果 δ_2 是周期的，那么根据对 $\mathbf{s}$ 的限制，它不可能是 $\mathtt{k}^n$。此外，根据性质 4，$\mathrm{ap}(\delta_2)\mathrm{ap}(\delta_3)$ 以 δ_2 为前缀。因此，$\mathbf{q}$ 是 $\mathrm{ap}(\delta_2)\,\mathrm{ap}(\delta_3)$ 的前缀。◀

▶ **推论 9.** 字符串 $\mathbf{s} = \mathbf{pq}$ 是$\mathrm{ap}(\delta_1)\mathrm{ap}(\delta_2)\mathrm{ap}(\delta_3)$的子串。

为调整上一节中的步骤 2，我们计算 $T_k(n, w, \alpha)$，它统计 $\mathsf{S}_k(n, w^\uparrow)$ 中项链的字典序小于项链 α 的字符串数量。注意， $T_k(n, w, \alpha_i) \neq |\mathrm{ap}(\alpha_1)\,\mathrm{ap}(\alpha_2)\cdots\mathrm{ap}(\alpha_{i-1})|$。令 $\mathrm{rank}'(\mathbf{s})$ 表示 $\mathbf{s}$ 在 $\mathsf{G}_k(n, w^\uparrow)$ 中的秩，并令 $S_k(n, w^\uparrow) = |\mathsf{S}_k(n, w^\uparrow)|$。综合以上各项，我们可以如下计算 $\mathrm{rank}'(\mathbf{s})$。

$$\mathrm{rank}'(\mathbf{s}) = \begin{cases} S_k(n, w^\uparrow) - (n-j) + 1 & \text{if } \mathbf{s} = \mathtt{k}^{n-j}\mathtt{a}_1\cdots\mathtt{a}_j \text{ for some } 0 \le j < n, \\ T_k(n, w, \beta_2) - |\mathbf{p}| + 1 & \text{otherwise.} \end{cases}$$

注意，无需计算 δ_2，因为 $T_k(n, w, \delta_2) = T_k(n, w, \beta_2)$。

接下来分析计算 $\mathrm{rank}'(\mathbf{s})$ 所需的运行时间。可以应用以下递推关系来枚举 $S_k(n, w^\uparrow)$，其中 $n, k \ge 0$ 和 $0 \le w \le kn$:

$$S_k(n, w^\uparrow) = \sum_{j=1}^{k} S_k(n-1, \max(0, w-j)^\uparrow), \tag{1}$$

基本情形为 $S_k(n, w^\uparrow) = k^n$，当 $w \le n$ 且 $S_k(0, w^\uparrow) = 0$，当 $w > 0$。使用动态规划，可以在 $S_k(n, w^\uparrow)$ 时间内计算出该值，所需空间为 $\mathcal{O}(n^2k^2)$，运行时间为 $\mathcal{O}(n^2k)$。第 7 节将证明以下引理。

▶ **引理 10.** 设 $n, k > 1$。对于任意 $\alpha \in \mathbf{N}_k(n)$，值 $T_k(n, w, \alpha)$ 可以在 $O(n^3k^2)$ 时间内计算得到，所需空间为 $O(n^3k)$。

回顾一下， β_2 可以在 $\mathcal{O}(n^2)$ 时间内计算得到，所需空间为 $\mathcal{O}(n)$。综合以上结果，可得以下定理。

▶ **定理 11.** 通用循环 $G_k(n, w^\uparrow)$ 可以在 $O(n^3k^2)$ 时间内完成排名，使用 $O(n^3k)$ 空间。

4.1 反排名

本节将演示如何确定在 $\mathcal{G}_k(n, w^\uparrow)$ 中秩为 r 的字符串 $\mathbf{s} = \mathtt{s}_1\mathtt{s}_2\cdots\mathtt{s}_n$。根据 r 的值，我们分两种情况讨论。

情况 1: $S_k(n, w^\uparrow) - n + 1 < r \le S_k(n, w^\uparrow)$。在这种情况下，$\mathbf{s}$ 位于回绕部分。回顾一下， $\mathtt{a}_1\cdots\mathtt{a}_n$ 是 $\mathsf{G}_k(n, w^\uparrow)$ 中最前面的 n 个符号（如性质 2 所定义），而最后面的 n 个符号为 $\mathtt{k}^n$ （依据性质 3）。因此，可以在 $\mathrm{O}(n)$ 时间内返回 $\mathbf{s}$。

情况 2: $1 \le r \le S_k(n, w^\uparrow) - n + 1$。根据 $\mathcal{G}_k(n, w^\uparrow)$ 的构造和性质 4，项链在 $\mathbf{N}_k(n, w^\uparrow)$ 中出现时，其秩严格递增。令 $\mathrm{SMALLESTNECK}(r)$ 表示 $\mathbf{N}_k(n, w^\uparrow)$ 中秩大于或等于 r in $\mathcal{G}_k(n, w^\uparrow)$ 的最小项链。如果 $\gamma_1 = \mathrm{SMALLESTNECK}(r)$，那么令

r' 表示 γ_1 in $G_k(n, w^{\uparrow})$的秩。如果 $r = r'$，则 $s = \gamma_1$。否则，令 $\gamma_2 = \text{SMALLESTNECK}(r' - n)$。根据性质 4，$\gamma_2$ 和 γ_1 在 $N_k(n, w^{\uparrow})$ 中相邻出现，这两条项链中至多一条是周期的，并且 s 是通过将 γ_2 的最后 $r' - r$ 个元素与 γ_1的前 $n - (r' - r)$个元素拼接而成。

例 3 设 $n = 3, k = 4$ 和 $w = 9$，并考虑 $r = 3$。回忆

$$\mathcal{G}_4(3,\ 9^{\uparrow}) = 144 \cdot 234 \cdot 243 \cdot 244 \cdot 3 \cdot 334 \cdot 344 \cdot 4。$$

项链 $\gamma_1 = \text{SMALLESTNECK}(r) = 234$ 的秩为 $r' = 4$，而 $\gamma_2 = \text{SMALLESTNECK}(r' - n) = 144$。由于 $r' - r = 1$，秩为 $r = 3$ 的字符串是 γ_2 的最后一个符号与其后的前 $n - 1 = 2$个符号连接而成 γ_1。它$s = 423$。

项链 SMALLESTNECK(r) 可通过执行 O($n$log k) 次秩查询来计算，如算法 1 所示。它采用了与 [SW17] 中的算法相同的方法，并加入了权重约束。该算法从第一个符号开始，迭代确定 $\tau = t_1 \cdots t_n$ 中的每个符号。经过 i次迭代后，循环不变式保证：$t_1 \cdots t_i k^{n-i}$ 是一个秩大于或等于 r 的项链，并且 $t_1 \cdots t_{i-1}(t_i - 1)k^{n-i}$ 要么不属于 $N_k(n, w^{\uparrow})$，要么属于 $N_k(n, w^{\uparrow})$ 且其秩小于 r。通过二分查找确定第 i个符号，最多需要 log k 次排名查询。

算法 1 计算$t_1 t_2 \cdots t_n$ 中秩大于或等于 r 的最小项链$N_k(n, w^{\uparrow})$，其中$1 \le r \le S_k(n, w^{\uparrow}) - r + 1$

1: **函数**最小项链(r)

2: ▷使用二分查找查找每个 t_i3: 对于 i从 1 到 n执行 4: $min \leftarrow 1$5: $max \leftarrow k$6: $t_i \leftarrow k$7: 当 $min < max$时执行 8: $prev \leftarrow t_i$9: $v \leftarrow (min + max)/2$10: $t_i \leftarrow v$11: 如果 $t_1 \cdots t_i k^{n-i} \in N_k(n, w^{\uparrow})$ 且秩 $'(t_1 \cdots t_i k^{n-i}) \ge r$ 则 $max \leftarrow v$12: 否则 13: $t_i \leftarrow prev$14: $min \leftarrow v + 1$15: 返回 $t_1 t_2 \cdots t_n$

▶**定理 12.** 该 通用循环 $G_k(n, w^{\uparrow})$ 可在 $O(n^4 k^2 \log k)$ 时间内完成反排名，使用 $O(n^3 k)$ 空间 *e.*

证明。 判断一个字符串是否为项链可在 $\mathcal{O}(n)$ 时间/空间 [Boo80]。因此，所述反排名算法的运行时间和空间主要由 $\mathcal{O}(n \log k)$ 次对排名函数的调用决定。◀

5 权重的一个上界

回顾一下，$S_k(n, w\downarrow)$ 表示$S_k(n)$ 中权重至多为 w的字符串构成的子集。补字符串s记作comp(s)，其定义是将每个符号 x 替换为 k−x+1。若s 的权重为 w，则comp(s) 的权重为 $kn - w + n$。

例4 令 $n=5, k=4$，并考虑**字符串** $=12443$。其权重为 $w=14$。那么 comp(s) $=43112$的权重为 $20-14+5=11$；符号 1 和 4 互换，符号 2 和 3 互换。

通用循环 $\mathbf{S}_k(n,\ (kn-w+n)^{\uparrow})$ 的补是 $\mathbf{S}(n,\ w^{\downarrow})$ 的通用循环，反之亦然。因此，$\mathcal{G}_k(n,\ w^{\downarrow}) = \mathrm{comp}(\mathcal{G}_k(n,\ kn-w+n)^{\uparrow})$ 是 $\mathbf{S}_k(n,\ w^{\downarrow})$ 的通用循环。要确定字符串 s 在$\mathcal{G}_k(n,\ w^{\downarrow})$ 中的秩，只需返回 comp(s) 在 $\mathcal{G}_k(n,\ (kn-w+n)^{\uparrow})$ 中的秩。

▶ **推论 13。** 通用循环 $G_k(n,w^{\downarrow})$ 可以在 $O(n^3k^2)$ 时间内计算秩，并可在 $O(n^4k^2\log\ k)$时间内反向计算，所需空间为 $O(n^3k)$。

6 t-子集与 t-多重集的应用

回顾一下，Subset(n,t) 表示 $\{1,2,\ldots,\mathtt{n}\}$ 的所有 t-子集构成的集合。利用每个子集的差分代表元，Subset(n,t) 的通用循环恰好是 $\mathrm{S}_{n-t+1}(t,n^{\downarrow})$ [CJJS25] 的通用循环。因此，我们可以应用推论 13 的结果，高效地计算 Subset(n,t) 的通用循环的秩。

▶ **定理14.** 序列 $G_{n-t+1}(t,n^{\downarrow})$ 是通过差分代表元构造的**子集**(n,t) 的通用循环。使用 $O(nt^3)$ 空间，可以在 $O(n^2t^3)$ 时间内对该序列进行排名，并在 $O(n^2t^4\log n)$时间内进行逆排名。

例5 序列 $\mathcal{G}_3(3,5_{\downarrow})=3112212111$ 是使用差分代表元构造的**子集**(5,3)的通用循环。如下表所示，利用其补序列$\mathcal{G}_3(3,7^{\uparrow})=1332232333$ 可以高效地对其进行解码。

Rank	字符串 $\mathcal{G}_3(3,7^{\uparrow})$中的字符串	补（差分代表）	在$\{1,2,3,4,5\}$上的子集
1	133	311	{3, 4, 5}
2	332	112	{1, 2, 4}
3	322	122	{1, 3, 5}
4	223	221	{2,4,5}
5	232	212	{2,3,5}
6	323	121	{1,3,4}
7	233	211	{2,3,4}
8	333	111	{1, 2, 3}
9	331	113	{1,2,5}
10	313	131	{1,4,5}

回顾一下，Multiset(n,t) 表示由 $\{0,2,\ldots,\mathtt{n}-1\}$ 中的所有 t-多重集构成的集合。对每个多重集使用差分表示，并将所有符号加 1 后，Multiset(n,t) 的通用循环恰好是 $\mathrm{S}_n(t,(n+t-1)^{\downarrow})$ [CJJS25] 的通用循环。同样，我们可以应用推论 13 中的结果，高效地解码 Multiset(n,t) 的通用循环。

▶ **定理 15.** 序列 $G_n(t,(n+t-1)^{\downarrow})$ 是应用调整后的差代表元的**多重集**(n,t) 的通用循环。使用 $O(nt^3)$ 空间，该序列可在 $O(n^2t^3)$ 时间内求秩，并可在 $O(n^2t^4\log n)$ 时间内反求秩。

例 6 序列 2001101000 是使用差分代表元的**多重集**(3,3)的通用循环。将序列中的每个符号都加一，即可得到$\mathcal{G}_3(3,5_{\downarrow})$ = 3112212111。如以下表格所示，可使用其补$\mathcal{G}_3(3,7^{\uparrow})$ = 1332232333 对其进行高效解码。

Rank	字符串 $\mathcal{G}_3(3,7^{\uparrow})$	补减 1（差分表示）	在 $\{0,1,2\}$ 上的多重集
1	133	200	$\{2,\ 2,\ 2\}$
2	332	001	$\{0,0,1\}$
3	322	011	$\{0,1,2\}$
4	223	110	$\{1,2,2\}$
5	232	101	$\{1,1,2\}$
6	323	010	$\{0,1,1\}$
7	233	100	$\{1,1,1\}$
8	333	000	$\{0,\ 0,\ 0\}$
9	331	002	$\{0,0,2\}$
10	313	020	$\{0,2,2\}$

7 引理的证明 10：枚举 $T_k(n,w,\alpha)$

本节将展示如何高效计算 $T_k(n,w,\alpha)$，其中 α 是 $\mathbf{N}_k(n,w^{\uparrow})$中的一条项链。该方法沿用 [SW17、HS19]中的方法。为简化后续记号，假设 n,k以及项链 $\alpha=\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n$ 均为固定的。

7.1 枚举初步集合

为了高效计算 compute $T_k(n,w,\alpha)$，我们需要高效确定两个特殊字符串集合的基数，这些字符串同时具有前缀限制和后缀限制。对于 $t\geq j$，令 $\mathbf{B}(t,j,w)$ 表示 $\mathbf{S}_k(t,w)$ 中满足以下条件的字符串子集：每个字符串都以前缀 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_j$ 开头，且每个非空后缀都大于 α。当$j=0$时，这些字符串没有前缀限制。

例 7 设 $\alpha=112122$ 和 $k=3$。那么 $\mathbf{B}(6,3,11)$ 中的字符串为：

112133	112223	112313
	112232	112322
	112233	112323
		112332
		112333

第一列中的字符串可由 $\mathbf{B}(6,4,11)$ 给出，而第二列和第三列中带下划线的字符串分别对应集合 $\mathbf{B}(2,0,5)$ 和 $\mathbf{B}(2,0,4)$。

令 $B(t,j,w)=|\mathbf{B}(t,j,w)|$。基于上一示例所示的递归结构，可以推导出 $B(t,j,w)$ 的枚举公式，其中 $0\leq j\leq t\leq n$。考虑空字符串，若 $w\leq 0$，则 $B(0,0,w)=1$；若 $w>0$，则 $B(0,0,w)=0$。当 $t>0$ 时，$B(t,t,w)=0$，因为后缀 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_t$ 是

字典序更小或等于 α。否则，$\mathbf{B}(t,j,w)$ 中的字符串可以根据位置 $j{+}1$处的符号进行划分。

▷ 若第 $j{+}1$ 个符号小于 $\mathtt{a}_{j+1}$，则从第一个索引开始的后缀将小于α，矛盾。▷ 若第 $j{+}1$ 个符号为 $\mathtt{a}_{j+1}$，则此类字符串的数目为 $B(t,j+1,w)$。▷ 若第 $j{+}1$ 个符号大于 $\mathtt{a}_{j+1}$，则由引理 5，从索引 1、2、.....、$j+1$ 开始的任意后缀都大于 α。因此，对每个 $\mathtt{x} > \mathtt{a}_{j+1}$，剩余的 $t-j-1$ 个位置可以按 $B(t-j-1,0,w-\text{权重}(\mathtt{a}_1\cdots\mathtt{a}_t\mathtt{x}))$ 种方式填充。

因此，对于 $0 \le j \le t \le n$ 和 $w \ge 0$:

$$B(t,j,w) = \begin{cases} 1 & \text{if } \ j=t=0 \text{ and } w \le 0, \\ 0 & \text{if } \ j=t, \\ B(t,j+1,w) + \displaystyle\sum_{\mathtt{x}=\mathtt{a}_{j+1}+1}^{\mathtt{k}} B(t-j-1,0,w-\text{weight}(\mathtt{a}_1\cdots\mathtt{a}_j\mathtt{x})) & \text{otherwise.} \end{cases}$$

现考虑集合 $\mathbf{P}(t,j,w)$，它表示 $\mathbf{S}_k(t)$ 的子集，其中每个字符串的权重恰好 w，且前缀为 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_j$，使得每个非空后缀都大于 α。令 $P(t,j,w) = |\mathbf{P}(t,j,w)|$。采用与计算 $B(t,j,w)$ 时相同的递归策略，但使用不同的基本情形，我们得到 $0 \le j \le t$的如下递推关系:

$$P(t,j,w) = \begin{cases} 1 & \text{if } j=t=w=0, \\ 0 & \text{if } j=t \text{ or } w \le 0, \\ P(t,j+1,w) + \displaystyle\sum_{\mathtt{x}=\mathtt{a}_{j+1}+1}^{\mathtt{k}} P(t-j-1,0,w-\text{weight}(\mathtt{a}_1\cdots\mathtt{a}_j\mathtt{x})) & \text{otherwise.} \end{cases}$$

7.2 计算 $T_k(n,\ w,\ \alpha)$

令 $\mathbf{T}_k(n,w,\alpha)$ 表示 $\mathbf{S}_k(n,w^{\uparrow})$ 中项链字典序小于给定项链 α 的字符串子集。注意，$T_k(n,w,\alpha) = |\mathbf{T}_k(n,w,\alpha)|$。为了计算 $T_k(n,w,\alpha)$，我们采用 [SW17、HS19] 中使用的方法，将 $\mathbf{T}_k(n,w,\alpha)$ 划分为若干子集。

令 $\mathbf{A}(t,\ j)$ 表示 $\mathbf{T}_k(n,\ w,\ \alpha)$ 中形如 $\mathtt{s} = \mathtt{s}_1\mathtt{s}_2\cdots\mathtt{s}_n$ 的字符串子集，其中 t 是满足 $\mathtt{s}' = \mathtt{s}_t\mathtt{s}_{t+1}\cdots\mathtt{s}_1\mathtt{s}_2\cdots\mathtt{s}_{t-1} < \alpha$ 的最小整数，而 j 是满足 $\mathtt{a}_1\cdots\mathtt{a}_j$ 是$\mathtt{s}'$ 的前缀的最大整数。令 $A(t,\ j) = |\mathbf{A}(t,\ j)|$。于是

$$T_k(n,w,\alpha) = \sum_{t=1}^{n}\sum_{j=0}^{n} A(t,j).$$

例 8 集合 $\mathbf{T}_3(5,\ 10,\ 12313)$ 包含 40 个字符串，它们来自由以下八个项链表示的等价类：{11233，11323，11332，11333，12133，12223，12232，12233} 这些项链出现在 $\mathbf{N}_3(5,\ 10)$ 中的 12313 之前。这些字符串可以划分为 10 个形如 $\mathbf{A}(t,\ j)$ 的子集，如下所示:

$\mathbf{A}(t,j)$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$j=1$	11233, 11323 11332, 11333	31123, 31132 21133, 31113	33112, 23113 33112, 33112	22311, 32311 33211, 33311	12231, 13231 13321, 13331
$j=2$	12233, 12223 12232, 12232	31213, 31222 21223, 31223	33121, 33121 32122, 33122	13312, 22312 23212, 23312	21331, 22231 22321, 22331

对于 $j \in \{0, 3, 4, 5\}$，$\mathbf{A}(t, j) = \emptyset$ 对 t 的所有取值均成立。

显然 $A(t, n) = 0$ 对所有 $1 \le t \le n$。否则对于 $j < n$，每个集合 $\mathbf{A}(t, j)$ 属于以下两种情况之一。

情况 1 $(t + j \le n)$。每个 $\mathbf{s} \in \mathbf{A}(t, j)$ 形如 $\mathbf{r}\,\mathtt{a}_1\mathtt{a}_2 \cdots \mathtt{a}_j\,\mathtt{x}\,\mathbf{q}$，其中：$\triangleright\ \mathbf{r} \in \mathsf{S}_k(t-1, w')$，使得每个非空后缀都大于 α，$\triangleright\ \mathtt{x} < \mathtt{a}_{j+1}$，$\triangleright\ \mathbf{q} \in \mathsf{S}_k(n-t-j, w-w'-\text{权重}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x}))$，且$\triangleright$ 权重$(\mathbf{s}) \ge w$。

回顾可知，当权重固定为 w' 时，$\mathbf{r}$ 的可能数目由 $P(t-1, 0, w')$ 给出。对于每个权重为 w' 且给定 $\mathtt{x}$ 的 $\mathbf{r}$，$\mathbf{q}$ 有 $S_k(n-t-j, w-w'-\text{权重}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x}))$ 种可能。因此，在这种情况下

$$A(t,j) = \sum_{\mathtt{x}=1}^{\mathtt{a}_{j+1}-1} \sum_{w'=t-1}^{k(t-1)} P(t-1, 0, w') S_k(n-t-j, w-w'-\text{weight}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x})).$$

情况 2 $(t + j > n)$。每个 $\mathbf{s} \in \mathbf{A}(t, j)$ 的形式为 $\mathtt{a}_{n-t+2} \cdots \mathtt{a}_{j-1}\mathtt{a}_j\,\mathtt{x}\,\mathbf{r}\,\mathtt{a}_1\mathtt{a}_2 \cdots \mathtt{a}_{n-t+1}$ 其中：$\triangleright\ \mathtt{x} < \mathtt{a}_{j+1}$，$\triangleright\ \mathbf{r} \in \mathbf{S}(n-j-1, w-\text{权重}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x}))$，并且$\triangleright$ $\mathtt{a}_{n-t+2}$ 的每个非空后缀 $\ldots \mathtt{a}_{j-1}\mathtt{a}_j\,\mathtt{x}\,\mathbf{r}$ 都大于 α。[3]

设 $\mathbf{q} = \mathtt{a}_{n-t+2} \cdots \mathtt{a}_{j-1}\mathtt{a}_j$。由于 $\mathbf{q}$ 是项链 α 的一个子串，$\mathbf{q}$ 的任意后缀都必须按字典序大于或等于 α 中长度相同的前缀（根据项链的定义）。因此，我们确定最长的 $\mathbf{q}$ 后缀，该后缀等于 α 中长度相同的前缀。假设该后缀的长度为 z。这意味着 $\mathtt{a}_{j-z+1} \ldots \mathtt{a}_{j-1}\mathtt{a}_j = \mathtt{a}_1\mathtt{a}_2 \ldots \mathtt{a}_z$。这意味着，从索引小于或等于 $|\mathbf{q}| - z$ 开始的 $\mathbf{s}$ 的任意后缀都大于 α。根据 $\mathtt{x}$，考虑三种子情况。

- 若 $\mathtt{x} < \mathtt{a}_{z+1}$ 则 $\mathtt{a}_{j-z+1} \ldots \mathtt{a}_{j-1}\mathtt{a}_j\mathtt{x}$ 小于 α，矛盾。
- 若 $\mathtt{x} = \mathtt{a}_{z+1}$ 则 $\mathtt{a}_{j-z+1} \cdots \mathtt{a}_{z+1}\mathbf{r} \in \mathbf{B}(n-j+z, z+1, w-\text{权重}(\mathtt{a}_1 \cdots \mathtt{a}_{j-z}))$。
- 若 $\mathtt{x} > \mathtt{a}_{z+1}$ 则 $\mathbf{s}$ 从小于或等于 $|\mathbf{q}| + 1$ 的索引开始的任意后缀都将大于 α根据性质 5。因此，$\mathbf{r} \in \mathbf{B}(n-j-1, 0, w-\text{权重}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x}))$。

因此，若 $\mathtt{a}_{j+1} > \mathtt{a}_{z+1}$

$$A(t,j) = B(n-j+z, z+1, w-\text{weight}(\mathtt{a}_1 \cdots \mathtt{a}_{j-z})) + \sum_{\mathtt{x}=\mathtt{a}_{z+1}+1}^{\mathtt{a}_{j+1}-1} B(n-j-1, 0, w-\text{weight}(\mathtt{a}_1 \cdots \mathtt{a}_j\mathtt{x})).$$

否则 $A(t, j) = 0$。

7.3 分析

通过应用动态规划，可以在 $O(n^3k^2)$ 时间内计算所有必要的 $B(t, j, w)$ 和 $P(t, j, w)$，并使用 $O(n^3k)$。预先计算出这些值后，对于情况 1，可以在 $O(nk^2)$ 时间内计算 $A(t, j)$；对于情况 2，则可以在 $O(k)$ 时间内计算。因此，可以在 $O(n^3k^2)$ 时间内计算 $T_k(n, w, \alpha)$。这证明了引理 10。

[3] 这可由 t 的定义得出，注意到 $|a_{n-t+2} \cdots a_{j-1}a_j\,x\mathbf{r}| < n$。

8 致谢

Joe Sawada（资助编号 RGPIN-2025-03961）谨此感谢加拿大自然科学与工程研究理事会（NSERC）对本研究的资助。

参考文献

BG11 A. Blanca 和 A. P. Godbole。关于新型组合结构的通用循环。SIAM离散数学期刊，25(4):1832–1842，2011年。BM93 J. Burns 和 C. Mitchell。位置传感编码方案。载于《密码学与编码 III》（M.J. Ganley 编），第31–66页。牛津大学出版社，1993年。Boo80 K. S. Booth。字典序最小循环子串。《Inform. Process. Lett.》，10(4/5):240–242，1980年。CDG92 F. Chung、P. Diaconis 和 R. Graham。组合结构的通用循环。《离散数学》，110(1):43–59，1992年。CGGP21 A. Cantwell、J. Geraci、A. Godbole 和 C. Padilla。组合对象的图通用循环。《应用数学进展》，127:102166，2021年。CJJS25 C. Campbell、L. Janik-Jones 和 J. Sawada。一个 n 集合的 k-子集和 k-多重集的通用循环，审稿中，2025年。dbs25 德布鲁因序列与通用循环构造（2025）。http://debruijnsequence.org.，2025年。DG11 P. Diaconis 和 R. Graham。《神奇数学：伟大魔术技巧背后的数学思想》。普林斯顿大学出版社，2011年。Etz24 T. Etzion。《序列与德布鲁因图》。学术出版社，2024年。FM78 H. Fredricksen 和 J. Maiorana。由 k 种颜色珠子构成的项链及 k 元德布鲁因序列。《离散数学》，23(3):207–210，1978年。Fre70 H. Fredricksen。字典序最小德布鲁因循环。《J. Combinatorial Theory》，9:1–5，1970年。Fre82 H. Fredricksen。全长非线性移位寄存器循环算法综述。《Siam Review》，24(2):195–221，1982年。GSWW20 D. Gabric、J. Sawada、A. Williams 和 D. Wong。用于构造 k 元德布鲁因序列和通用循环的后继规则框架。IEEE信息论汇刊，66(1):679–687，2020年。HJZ09 G. Hurlbert、T. Johnson 和 J. Zahl。关于多重集的通用循环。《离散数学》，309(17):5321–5327，2009年。德布鲁因循环和格雷码的推广／图的非对称性／顶点传递（Cayley）图的哈密顿性问题。HRW12 A. E. Holroyd、F. Ruskey 和 A. Williams。排列的简写通用循环。Algorithmica，64(2):215–245，2012年。HS19 P. Hartman 和 J. Sawada。固定密度项链和 Lyndon 词的排名与反排名。《理论计算机科学》，791:36–47，2019年。Hur94 G. Hurlbert。关于一个 n 集合的 k-子集的通用循环。SIAM离散数学期刊，7(4):598–604，1994年。Jac93 B. Jackson。k-子集和 k-排列的通用循环。《离散数学》，117:141–150，1993年7月。JSH09 B. Jackson、B. Stevens 和 G. Hurlbert。关于格雷码和通用循环的研究问题。《离散数学》，309(17):5341–5348，2009年。Knu11 D. E. Knuth。《计算机程序设计艺术》第4A卷：组合算法，第1部分。Addison-Wesley，Upper Saddle River, NJ，2011年。KRR16 T. Kociumaka、J. Radoszewski 和 W. Rytter。Lyndon 词的高效排名及字典序最小德布鲁因序列的解码。SIAM离散数学期刊，30(4):2027–2046，2016年。Mar34 M. H. Martin。一个排列问题。《Bull. Amer. Math. Soc.》，40(12):859–864，1934年。

MEP96 C. Mitchell、T. Etzion 和 K. Paterson。构造可译码德布鲁因序列的一种方法。IEEE信息论汇刊，42(5):1472–1478，1996年。RSW12 F. Ruskey、J. Sawada 和 A. Williams。固定重量二元串的德布鲁因序列。SIAM离散数学期刊，26(2):605–617，2012年。Rud13 Y. Rudoy。通过归纳法构造 $[n]$ 的 k-子集上的通用循环。《电子组合学期刊》，20:P18，2013年。RW10 F. Ruskey 和 A. Williams。一个 n 集合的 (n-1)-排列的显式通用循环。ACM算法汇刊，6(3):1–12，2010年7月。SSTW24 J. Sawada、J. Sears、A. Trautrim 和 A. Williams。拼接树：高效构造通用循环和德布鲁因序列的框架。arXiv 预印本 arXiv:2308.12405，2024 年。SW17 J. Sawada 和 A. Williams。对项链、Lyndon 词和德布鲁因序列进行排名的实用算法。《离散算法期刊》，43:95–110，2017年。SWW13 J. Sawada、A. Williams 和 D. Wong。重量范围二元串的通用循环。载于《组合算法：第24届国际研讨会 IWOCA 2013，法国鲁昂，2013年7月10日至12日，修订后的精选论文24》，第388–401页。施普林格，2013年。SWW14 J. Sawada、A. Williams 和 D. Wong。具有最小指定重量的二元字符串的字典序最小通用循环。《离散算法期刊》，28:31–40，2014年。StringMasters 2012，2013年特刊（第1卷）。SWW16 J. Sawada、A. Williams 和 D. Wong。将德布鲁因序列和通用循环的经典贪心构造与项链构造推广。《Electron. J. Combin.》，23(1):论文1.24，20，2016年。Tul01 J. Tuliani。具有高效解码算法的德布鲁因序列。《离散数学》，226(1):313–336，2001年。