k 子集和 k 多重集的通用循环构造

Colin Campbell圭尔夫大学，加拿大
Luke Janik-Jones圭尔夫大学，加拿大
Joe Sawada
圭尔夫大学，加拿大

arXiv:2603.11954v1 [cs.DM] 12 Mar 2026

摘要

集合S 中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S 中的元素都恰好以子串的形式出现一次。若S 是 k-子集的集合，取自 $[n] = \{1, 2, \ldots, n\}$，众所周知，采用简单的字符串表示时，通用循环并非总是存在；例如，子集{1, 2}可以表示为12或21。类似地，若S 是k-多重集的集合，取自 $[n]$，则众所周知，采用类似的表示方式时，通用循环也并非总是存在；例如，多重集{1，1，2}可以表示为112、121或211。通过将这些集合映射到适当的带标号图族，已知通用循环存在，但尚无已知的高效构造方法。本文针对 k-子集和 k-多重集考虑一种新的表示方式，由此可为所有 n，$k \geq 2$构造高效的通用循环。我们给出后继规则算法，以每个符号$\mathcal{O}(n)$的时间构造此类通用循环，并使用$\mathcal{O}(n)$的空间；同时证明，项链拼接算法可使相同序列的生成达到每个符号$\mathcal{O}(1)$的摊销时间。这是已知的首批针对 k-多重集的高效通用循环构造。所得结果基于有界权重德布鲁因序列的构造。特别地，我们证明，有界权重的 Grandmama 德布鲁因序列推广形式可以按每个符号$\mathcal{O}(1)$的摊销时间构造。

1 引言

一个通用循环对于集合S 的组合对象来说，是一个长度为 |S| 的循环序列，其中每个元素的表示都恰好作为一个子串出现一次，且该元素属于 S 。本文重点讨论两个集合：(a)$\mathbf{S}_k(n)$，表示 k-子集的集合，其元素来自 $[n] = \{1, 2, \ldots, n\}$；以及 (b) $\mathbf{M}_k(n)$，表示 k-多重集的集合，元素来自 $[n]$。例如：

- $\mathbf{S}_2(3) = \Big\{\{1,2\},\{1,3\},\{2,3\}\Big\}$, and
- $\mathbf{M}_2(3) = \Big\{\{1,1\},\{1,2\},\{1,3\},\{2,2\},\{2,3\},\{3,3\}\Big\}$.

正如本文所展示的， k-子集或 k-多重集的表示方式，对于这些集合是否存在通用循环以及能否高效构造通用循环至关重要。

子集。关于 $\mathsf{S}_k(n)$ 的通用循环，主要通过考察标准字符串表示进行研究，即子集 {1, 2} 可以表示为 12 或 21（参见 [CDG92, JSH09, Lan12, Rud13]）。这种表示有一个重大缺陷：只有当 n 能整除 $\binom{n}{k}$，或等价地，当 k 能整除$\binom{n-1}{k-1}$时，通用循环才存在。满足这一条件时，Chung、Diaconis 和 Graham [CDG92] 猜想，对于固定的 k，当 n 足够大时，$\mathsf{S}_k(n)$ 的通用循环存在。该猜想已在 [Jac93, Hur94], 中针对 $k = 3$、4、6 得到验证；研究还表明，当 $k = n - 2$ [SBE+02] 时，子集的通用循环不存在。最近，Glock 等人 [GJKO20] 通过研究超图中的欧拉回路，肯定地回答了这一猜想。采用这种标准表示时，$\mathsf{S}_k(n)$ 的通用循环在大量情况下并不存在，因此人们开始研究子集装填与覆盖 [CHHM09, DL16]。

2 k-子集和 k-多重集的通用循环

另一种常见的 k-子集表示法，是用长度为 n 的二进制串表示，其中恰有 k 个 1。显然，采用这种表示法无法构造非平凡的通用循环。不过，若采用一种简写的二进制串表示法（省略末尾冗余的符号），每个 k-子集都对应一个唯一的长度为 $n-1$ 的子串，其中 1 的个数为 $k-1$ 或 k。这些字符串（因而也包括 k- 子集）的通用循环，可以通过项链拼接方法 [RSW12] 构造，摊销时间为 O(1)，空间复杂度为 O(n) [SW13]。[1] 同一序列也可以通过一种基于“缺失符号寄存器”的后继规则 [SW23]生成，其每个符号的生成时间为 O(n)。

本文探讨一种新颖的差分表示法：用长度为k 的字符串表示一个 k-子集，其中第一个符号是该子集中的最小元素，之后每个符号与字符串中的前一个符号相加，得到该子集中下一个更大的元素。例如，对于$n = 5$，子集{1, 3, 4}可表示为121。字符串的权重是其各符号之和。注意，采用差分表示法后，k-子集的集合对应于字母表{1, 2, . . . , n−k+1}上所有长度为 k的字符串，且每个字符串的权重均不超过 n。Jackson [Jac93] 和Hurlbert [Hur94], 此前研究过类似的差分表示法，但他们采用循环的方式处理，且未固定第一个符号。

的三种不同表示法见表1，其中$S_3(5)$。序列1101011100是采用简写表示法的$S_3(5)$ 通用循环，而序列1112122113 是采用差分表示法的$S_3(5)$ 通用循环。采用标准表示法时，不存在$S_3(5)$ 通用循环。

子集	简写	差集	标准
{1, 2, 3}	1110 (0)	111	123、132、.....，或 321
{1,2,4}	1101 (0)	112	124、142、.....，或 421
{1, 2, 5}	1100 (1)	113	125、152、. . .、或 521
{1, 3, 4}	1011 (0)	121	134, 143, . . . , 或 431
{1,3,5}	1010 (1)	122	135，153，.....，或 531
{1,4,5}	1001 (1)	131	145，154，.....，或 541
{2,3,4}	0111 (0)	211	234、243、.....、或432。
{2, 3, 5}	0110 (1)	212	235、253、.....、或 532
{2,4,5}	0101 (1)	221	245、254、.....、或 542
{3,4,5}	0011 (1)	311	345, 354, . . . , 或 543

表1 展示了$S_3(5)$中各元素的不同表示方式。对于简写代表元，省略的冗余符号以括号标出。

Cantwell 等人 [CGGP21]使用带标号图表示一个 k-子集。借助这种表示，他们证明了相应图族存在通用循环；然而，他们没有提供高效的构造方法。

多重集。与 k-子集一样，表示方式对于构造k-多重集的通用循环至关重要。众所周知，$|M_k(n)| = \binom{n+k-1}{k}$。Jackson [Jac93]以及 Hurlbert、Johnson 和 Zahl [HJZ09]首次使用标准字符串表示研究了 $M_k(n)$ 的通用循环，其中 112、121 或 211 都可以表示多重集 {1, 1,2}。与 k-子集类似，他们证明，$M_k(n)$ 的通用循环存在的必要条件是 n 整除 $\binom{n+k-1}{k}$。表示 k-多重集的另一种方法是使用频数映射，即一个长度为n 的字符串 $f_1 f_2 \cdots f_n$，其中每个 f_i 表示 i 在多重集中出现的次数。注意，$\sum_{i=1}^{n} f_i = k$。例如，在 $M_2(3)$ 中，多重集 {1, 2} 的频数映射表示为 110。这是一个

[1] 这种通用循环在子集中的应用最早由 [BG11]指出。

简单的练习，可以证明使用这种表示方式时，对于 n、$k \geq 2$，$\mathbf{M}_k(n)$ 的通用循环不存在。不过请注意，频数映射中的最后一个符号 f_n 是多余的：它的值可以由前面的 $n-1$ 个值确定，即 $f_n = k - \sum_{i=1}^{n-1} f_i$。因此，我们称 $f_1 \cdots f_{n-1}$ 为 $\mathbf{M}_k(n)$ 中多重集的简写频数表示。注意，这个集合对应于字母表 {0, 1,. . ., k} 上所有长度为$n-1$的字符串，其中每个字符串的权重不超过 k。

差分表示也可用于 k多重集。考虑将相同的定义应用于 k子集，但规定差分代表的第一个符号比多重集中最小的符号小一。等价地，直接应用差分代表的原始定义，但将k多重集定义在基集{0，1，……，$n-1$}上。注意，应用这种差分表示后，k多重集对应于字母表{0，1，……，n−1} 上的所有字符串 k，每个字符串的权重至多为 $n-1$。

这三种不同的表示方式用于 k多重集，在表2 中以$\mathbf{M}_3(3)$为例进行了说明。序列0011021203 是 $\mathbf{M}_3(3)$ 的一个通用循环，采用的是简写频数表示；而序列0001011002 是$\mathbf{M}_3(3)$ 的一个通用循环，采用的是差分表示。采用标准表示时，$\mathbf{M}_3(3)$ 不存在通用循环。

多重集	简写频数	差异	标准
{1, 1,1}	30 (0)	000	111
{1, 1,2}	21 (0)	001	112，121，或 211
{1，1，3}	20 (1)	002	113, 131,或 311
{1，2，2}	12 (0)	010	122，212，或 221
{1，2，3}	11 (1)	011	123、132、213、231、312、或321
{1，3，3}	10 (2)	020	133、313，或 331
{2, 2,2}	03 (0)	100	222
{2, 2,3}	02 (1)	101	223，232，或 322
{2, 3,3}	01 (2)	110	233，323，或 332
{3, 3,3}	00 (3)	200	333

表2 展示多重集 $\mathbf{M}_3(3)$ 的不同表示形式。对于简写频数代表，省略的冗余符号以括号标出。

多重集的通用循环 k在近距离传感器网络等领域有相关应用 [CWLW24]。如果多重集中每个元素的数量都不超过 k，那么这类受 t限制的多重集 t已在 [BG11] 中得到研究。与 k子集类似，Ca ntwell 等人 [CGGP21] 使用带标号图来表示多重集 k。他们利用这种表示证明了相应图族存在通用循环；不过，他们没有给出高效的构造方法。

主要结果。 回顾一下，$\mathbf{S}_k(n)$ 的差分代表，以及$\mathbf{M}_k(n)$ 的简写频数代表和差分代表，都是指定字母表上定长字符串的特例，且这些字符串的权重有上界。因此，可以应用有界权重德布鲁因序列的相关结果（见第 3 节）来获得本文的主要结果。特别是，我们证明了有界权重的 Grandmama 德布鲁因序列推广形式 [DHS+18] 可以通过构造实现，每个符号的摊销时间为 $\mathcal{O}(1)$。我们还介绍了一种应用缺失符号寄存器（MSR）的新颖构造方法。

1. 我们展示了一种算法，可利用差分代表构造通用循环$\mathcal{S}_1$，用于 $\mathbf{S}_k(n)$ 中的每个符号，且每个符号的摊销时间为 $\mathcal{O}(1)$。此外，给定 $\mathbf{S}_k(n)$ 中的任意一个子集及其差分代表$d_1 d_2 \cdots d_k$，即可在 $\mathcal{S}_1$ 中以 $\mathcal{O}(n)$ 时间计算出该字符串后面的符号。

2. 我们展示了一种算法，可利用简写频数代表构造通用循环 $\mathcal{M}_1$，用于 $\mathbf{M}_k(n)$ 中的每个符号，且每个符号的摊销时间为 $\mathcal{O}(1)$。此外，给定 $\mathbf{M}_k(n)$ 中的任意一个子集及其简写频数代表 $f_1 f_2 \cdots f_{n-1}$，即可在 $\mathcal{M}_1$ 中以 $\mathcal{O}(n)$时间计算出该字符串后面的符号。

3. 我们展示了一种利用差分代表构造通用循环 $\mathcal{M}_2$，适用于 $\mathbf{M}_k(n)$，其每个符号的摊销时间为$\mathcal{O}(1)$。此外，给定 $\mathbf{M}_k(n)$ 中的任意一个子集及其差分代表 $d_1 d_2 \cdots d_k$，可以在 $\mathcal{M}_2$ 中用 $\mathcal{O}(n)$ 时间计算出该字符串之后的符号。

最后两个结果是已知的首批高效 k-多重集通用循环构造。所有算法所需的空间均为关于 n 和 k的多项式空间。本文介绍的算法以及许多已知通用循环构造的实现均可在 http://debruijnsequence.org [dbs26]处获取。

论文结构。 第 2 节介绍预备定义和记号，并简要回顾循环合并过程与拼接树。第 3节描述一种已知算法，用于构造满足给定权重约束的长度为 t元字符串的通用循环，其中字符串的权重为 n，并对其中一种特定构造提出新的见解。我们还介绍一种具有有趣性质的新构造。接着在第 4节，我们将这些构造应用于 k-子集和 k-多重集。

2 预备知识

设 $\Sigma_t = \{0,1,2,\ldots,t-1\}$，并令 $\Sigma_t(n)$ 表示由 Σ_t 上所有长度为 n 的字符串组成的集合。回顾一下，字符串的权重是其符号之和。令 weight(α) 表示字符串 α 的权重。令 $\Sigma_t(n,w)$ 表示 $\Sigma_t(n)$ 中权重至多为 w 的所有字符串构成的子集。本文后续将关注按反字典序排列的字符串；当从右向左读取字符串时，这就是标准的字典序。例如，下面的有界权重字符串集合按反字典序排列：

$$\Sigma_3(3,2) = \{000, 100, 200, 010, 110, 020, 001, 101, 011, 002\}.$$

考虑两个字符串 α 和 β。令 $\alpha \cdot \beta$ 表示 α 和 β的拼接，并令 β^j 表示 j 个 β拼接在一起。 非周期前缀是指 α 的最短字符串 β，使得 $\alpha = \beta^j$ 对某个 $j \geq 1$成立。若一个字符串与其非周期前缀相同，则称其为 非周期的；否则称其为 周期的。

一个项链等价类是字符串在旋转下形成的一个等价类。一个项链是项链等价类中字典序最小的字符串。令$\mathbf{N}_t(n)$ 表示长度为n、以 Σ_t为字母表的项链集合。例如，

$\mathbf{N}_3(3) = \{000, 001, 002, 011, 012, 021, 022, 111, 112, 122, 222\}$。

Let $\mathbf{N}_t(n, w)$ 表示 $\mathbf{N}_t(n)$ 中所有权重不超过 w 的字符串所组成的子集。例如 ，

$\mathbf{N}_3(3, 2) = \{000, 001, 002, 011\}$。

反馈函数 f 将 $\Sigma_t(n)$中的字符串映射到 Σ_t。反馈移位寄存器是一个函数，对于给定的反馈函数 f，它将字符串 $\alpha = \mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n$映射为 $\mathtt{a}_2\cdots\mathtt{a}_n f(\alpha)$。纯循环寄存器（PCR）是反馈函数为 $f(\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n) = \mathtt{a}_1$ 的反馈移位寄存器。它将任意在旋转下封闭的集合划分为项链类，这些项链类可由 $\mathbf{N}_t(n)$ 中的项链表示。考虑 $\Sigma_t(n+1)$中字符串的一个子集S，其中每个字符串都具有唯一的简写代表，如第1节所述。令S′表示该简写代表元集合。则每个字符串 $\alpha = \mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n \in \mathrm{S}'$ 对应于唯一的字符串 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n\mathtt{z} \in \mathrm{S}$。我们称 z 为缺失符号。缺失符号寄存器（MSR）是一个反馈移位

此类集合的寄存器 $\mathbf{S}'$，其反馈函数为 $f(\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n)=\mathtt{z}$。MSR将任意简写集合 $\mathbf{S}'$在旋转下封闭者划分为项链类，且这些项链类可由位于 $\mathbf{N}_t(n+1)$中的项链表示 （参见示例 2，第3.2节）。MSR已被应用于排列、子集的简写代表元，以及更一般情形下具有固定内容的字符串，其所在集合为 [SW23]。

2.1 循环合并树与后继规则

在本节中，假定字符串集合 $\mathbf{S}$ 的代表元就是 $\mathbf{S}$ 本身。给定一个通用循环$\mathcal{U}=\mathtt{u}_1\mathtt{u}_2\cdots\mathtt{u}_m$，其对象为 $\mathbf{S}$，则后继规则 （针对 $\mathcal{U}$）是一个函数，它将每个字符串 α （属于 $\mathbf{S}$）映射到紧接在 α 之后、出现在 $\mathcal{U}$ 中的符号。因此，从 $\mathbf{S}$ 中的任意字符串出发，$\mathcal{U}$（按循环方式考虑）都可以通过反复应用其后继规则来构造。

设 $\mathbf{U}_i$ 表示 $\mathbf{S}_i\subseteq\Sigma_t(n)$ 的一个通用循环。若 $\mathbf{S}_1\cap\mathbf{S}_2=\emptyset$，则称两个通用循环 $\mathbf{U}_1$ 与 $\mathbf{U}_2$ 不相交。设 $\mathtt{x}$、$\mathtt{y}$ 为 Σ_t 中互不相同的符号。若 $\sigma=\mathtt{x}\mathtt{s}_2\cdots\mathtt{s}_n$ 且 $\hat{}\sigma=\mathtt{y}\mathtt{s}_2\cdots\mathtt{s}_n$，则称 σ 与 $\hat{}\sigma$ 互为共轭，并称 $(\sigma,\hat{}\sigma)$ 为一个共轭对。两个不相交的通用循环 $\mathbf{U}_1$ 与 $\mathbf{U}_2$ 可以通过交换共轭对 $(\sigma,\hat{}\sigma)$ 中每个字符串的后继而连接成单一的通用循环，其中 σ 出现在 $\mathbf{U}_1$ 中，$\hat{}\sigma$ 出现在 $\mathbf{U}_2$中。这就是著名的循环合并过程，即两个循环 $\mathbf{U}_1$ 与 $\mathbf{U}_2$ 经由共轭对 $(\sigma,\hat{}\sigma)$ 合并。圈连接树 $\mathbb{T}$ 是一棵无序树，其节点对应于一组互不相交的通用循环 $\mathbf{U}_1,\mathbf{U}2,\ldots,\mathbf{U}_\ell$，而 $\mathbf{U}_i$ 与 $\mathbf{U}j$ 之间的一条边由共轭对 $(\sigma,\hat{}\sigma)$ 定义，且满足 $\sigma\in\mathbf{S}_i$ 以及 $\hat{}\sigma\in\mathbf{S}j$。就我们的目的而言，我们将循环合并树视为有根的。在二元情形中，反复合并圈连接树中的相邻循环，可得到 $\mathbf{S}_1\cup\mathbf{S}_2\cup\cdots\cup\mathbf{S}_\ell$ 的唯一通用循环。对于非二进制字母表，根据循环被合并的顺序，可能得到不同的通用循环 [SSTW24]。因此，引入了下述性质，以便基于共轭对为非二进制字母表给出一种一般性的后继规则。

> **链性质**：如果圈连接树中的一个节点 $\mathbb{T}$ 有两个子节点，它们分别通过共轭对（$\mathtt{x}\mathtt{s}_2\cdots\mathtt{s}_n,\mathtt{y}\mathtt{s}_2\cdots\mathtt{s}_n$）和（$\mathtt{x}'\mathtt{t}_2\cdots\mathtt{t}_n,\mathtt{y}'\mathtt{t}_2\cdots\mathtt{t}_n$）连接，那么 $\mathtt{s}_2\cdots\mathtt{s}_n\neq\mathtt{t}_2\cdots\mathtt{t}_n$。

设 $\mathbb{T}$ 是一个满足底层字符串集合 $\mathbf{S}$ 的链性质的圈连接树，其中节点由一组共轭对 $\mathbf{C}$ 连接。如果该树包含 ℓ 个循环，那么 $\mathbf{C}$ 包含 $\ell-1$ 个共轭对，每个共轭对对应于 $\mathbb{T}$中的一条边。若 γ 属于共轭对（σ，$\hat{}\sigma$），则称其满足以下条件之一：$\gamma=\sigma$ 或 $\gamma=\hat{}\sigma$。令C_1、C_2、...、C_m 表示 $\mathbb{T}$ 中节点的一条最长路径，使得对于每个 $1\le i<m$，节点 C_i 是 C_{i+1} 的父节点，并且二者通过形如（$\mathtt{x}_i\beta$，$\mathtt{x}_{i+1}\beta$）的共轭对连接；β 在每个共轭对中都相同。将这样的路径称为长度为 m的链，并定义 g（$\mathtt{x}_i\beta$）$=\mathtt{x}_{i+1}$ 以及 $i<m$ 的 g（$\mathtt{x}_m\beta$）$=\mathtt{x}_1$，其中每个字符串都属于一个共轭对。则以下函数 h 是对应于 $\mathbf{S}$ 的通用循环的后继规则，其中 f（α）是底层反馈函数（例如 PCR 或 MSR），它在 $\mathbb{T}$中生成这些循环：

$$h(\alpha)=\begin{cases} g(\alpha) & \text{if } \alpha \text{ belongs to a conjugate pair in } \mathbf{C};\\ f(\alpha) & \text{otherwise.}\end{cases}$$

此规则对应于 函数 $\uparrow f_1(\alpha)$在 [SSTW24] 中，当 f 是 PC 的反馈函数时 R.

> 示例1 考虑由图1所示的 PCR 诱导出的、以节点为元素的圈连接树。节点路径 $\mathtt{000}\to\mathtt{001}\to\mathtt{002}\to\mathtt{003}\to\mathtt{004}$ 构成一条链，其中 $\beta=\mathtt{00}$。共轭对分别为 (000, 100)、(100,200)、(200,300) 和 (300, 400)。因此，h(000) $=1$、h(100) $=2$、h(2000) $=3$、h(300) $=4$，以及 h(400) $=0$。

直接应用该规则需要存储所有共轭对；然而，在我们的应用中，仅使用 $\mathcal{O}(n)$ 的空间，就可以相对简单地在 $\mathcal{O}(n)$ 时间内测试某个字符串是否属于共轭对。在本文中，我们将基于 PCR 和 MSR 定义圈连接树，然后应用 $h(\alpha)$ 高效地构造相应的通用循环。

2.2 拼接树

本节将介绍一种简化版的“左”拼接树，最初由 [SSTW24],定义；我们添加了一项假设：圈连接树中每个非根周期节点都是叶节点。

二分有序树（*BOT*）是一种有根树，其中一个节点的每个子节点都属于左子节点或右子节点之一，且每组中的节点都有序。设 $\mathbb{T}$ 是以 r 为根且满足链性质的基于PCR的圈连接树。基于 $\mathbb{T}$ 的拼接树会将 $\mathbb{T}$ 转换为一棵BOT：对子节点排序，将其分为左子节点和右子节点，并为每个节点（循环）分配可能的新标签。父子关系保持不变，且每个节点都有一个变化索引，即根据连接这两个节点的共轭对，节点标签与其父节点标签不同的唯一索引。在我们新增的假设下，除节点是周期根的子节点这一情况外，所有非根周期节点都是叶节点，因此该索引是唯一的。遇到这种情况时，我们将具体处理。根 r 可以被分配任意变化索引。若节点的变化索引为 c，则变化索引小于或等于 c 的所有子节点都归为左子节点，并按变化索引从小到大排序。变化索引大于 c 的所有子节点都归为右子节点，也按变化索引从小到大排序。RCL次序会从根开始递归遍历拼接树，先访问所有**右**子节点，再访问**当**前节点，最后访问所有**左**子节点。例如，图1 展示了一棵圈连接树、对应的拼接树及其RCL次序。

给定一棵拼接树 T，令$\mathcal{U}_T$ 为按 RCL 顺序访问各节点时，将每个节点的非周期前缀拼接而成的序列。以下结果对应于 [SSTW24]中的定理 3。

▶ **定理1**。设 T 是一棵基于 *PCR* 的循环拼接树，具有链性质，其底层集合为 **S**。则 U_T 是 **S**的一个通用循环，其后继规则为 $h(\alpha)$。

3 有界权重德布鲁因序列

对于 $\Sigma_t(n)$ 的通用循环称为德布鲁因序列。对于 $\Sigma_t(n)$ 的子集，若其权重具有上界或下界，则其通用循环称为有界权重德布鲁因序列。具有权重下界的字典序最小二进制德布鲁因序列可以通过每个符号 O(1) 摊还时间构造出来 [SWW14]。在 [SWW16]中，作者通过给出一种等价的贪心构造，将该算法推广到了非二进制字母表。 [GSWW20]中给出了任意字母表上有界权重德布鲁因序列的八种 O(n) 时间后继规则构造：其中四种具有权重下界，另外四种具有权重上界。[2] 上述每种算法都需要 O(n) 空间。我们研究的是 $\Sigma_t(n, w)$ 的通用循环，即权重上界为 w 的字符串。

本节重点介绍 [GSWW20] 中的一条后继规则，该规则为$\Sigma_t(n, w)$ 构造了一个具有有趣性质的通用循环。由于 $\Sigma_t(n, w)$ 在旋转下封闭，PCR 将该集合划分为项链循环。这些循环（节点）可以用$\mathbf{N}_t(n, w)$中的项链表示。例如， $\Sigma_5(3,4)$ 的循环可以用下列项链表示，因为它们是按反字典序列出的（这种顺序稍后会

[2] 直观来看，每条后继规则最多需要 t 次项链测试，每次测试需要 $\mathcal{O}(n)$ 的时间。不过，应用与接下来定理 6 的证明类似的优化后，这些后继规则可以通过常数次项链测试来实现。

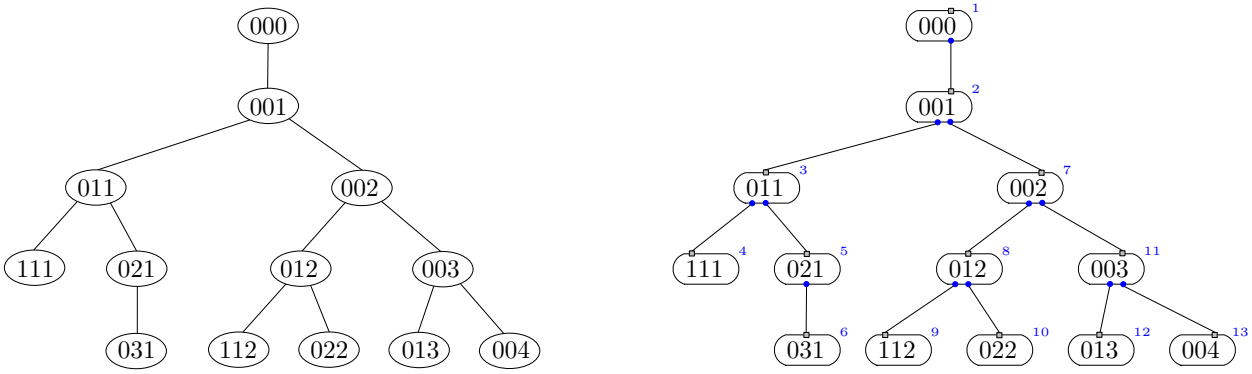
派上用场）：

$\mathbf{N}_5(3,\ 4)\ =\{000,\ 001,\ 011,\ 111,\ 021,\ 031,\ 002,\ 012,\ 112,\ 022,\ 003,\ 013,\ 004\}$。

以下父节点规则[3] 定义了一棵以 0^n 为根的圈连接树，其节点为由下列项链表示的循环$\mathbf{N}_t(n,\ w)$。

首个非零父节点规则（根为 0^n）：非根节点 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n$ 的父节点，其中 j 表示满足 $\mathtt{a}_j\neq 0$ 的最小索引，是 $0^{j-1}(\mathtt{a}_j-1)\mathtt{a}_{j+1}\cdots\mathtt{a}_n$。

例如，循环 基于此首个非零父节点规则的 $\Sigma_5(3,4)$的cle-连接树如图 1.



(a) 圈连接树 (b) 拼接树

图1 (a) $\Sigma_5(3,4)$的首个非零圈连接树，其中节点由$\mathbf{N}_5(3,4)$中的项链表示。(b) 相应的拼接树，其中根的变化索引被指定为最后一个符号，如节点顶部的小方框所示。所有子节点均为左子节点，且节点上的标签无需更改。对该树进行RCL遍历时，按反字典序访问项链。

▶ **引理 2.** 设 $\mathbb{T}$ 是由首个非零父节点规则诱导的基于*PCR*的环合并树，并设$\alpha=0^{i-1}\mathtt{x}\mathtt{a}_{i+1}\cdots\mathtt{a}_n$ 与 $\beta=0^{j-1}\mathtt{y}\mathtt{b}_{j+1}\cdots\mathtt{b}_n$ 是 $\mathbb{T}$ 中两个不同的节点（项链），其中 i、$j\leq n$ 以及$\mathtt{x}$、$\mathtt{y}>0$。则下列结论成立。

1. 若 $\alpha\neq 0^{n-1}1$，则 α 的父节点是一个非周期项链。（所有非根周期节点都是叶节点。）**2.** $\mathbb{T}$ 具有链性质。**3.** 若 α 和 β 都是周期的，则它们有不同的父节点。

证明。 **(1)** 由于 α 是一个项链，其前缀 $\mathtt{a}_1\cdots\mathtt{a}_i=0^{i-1}\mathtt{a}_i$ 按字典序小于或等于 α 中任何其他长度相同的子串（在 α 被视为循环时）。其父节点 γ 的前缀为 $0^{i-1}(\mathtt{a}_i-1)$，该前缀严格小于 γ 中所有其他循环考察的子串，这意味着 γ 是一个项链。此外，这也说明，只要 γ，$\gamma\neq 0^n$就是非周期的，而这仅在 $\alpha=0^{n-1}1$ 时成立。

(2-3) 假设 α 与 β 具有相同的父节点 γ。若 $i=j$ 则 $\alpha=\beta$，这与 α 和 β 互不相同相矛盾。由于它们是满足 $\mathtt{x},\mathtt{y}>0$的项链，我们有 $\mathtt{a}_n,\mathtt{b}_n>0$。不失一般性，假设 $i<j$。该

[3] 这条规则在 中被称为 第一个非 1 [GSWW20]，因为所考虑的字母表是 $\{1,2,\ldots,t\}$。

8 个 k子集和 k多重集的通用循环

连接 α 与 γ 的共轭对中的字符串以后缀 $\mathtt{a}_n0^{i-1}$ 结尾，而连接 β与 γ 的共轭对中的字符串以后缀 $\mathtt{b}_n0^{j-1}$ 结尾。因此，链性质成立。此外，假设 α 和 β 都是周期的。由于 β 是周期的，则 γ 必有一个子串 0^{j-1} 出现在第 j个符号之后。但这意味着α 也具有相同的子串，因为 α 和 β 共享同一父节点。这与 α 是一条项链这一事实相矛盾。因此 α 和 β 具有不同的父节点。◀

以下基于第一个非零圈连接树的后继规则对应于后继规则 g_4，见于 [GSWW20]。在其最初的表述中，所考虑的字母表为 {1、2、.....、 t}，而我们将该规则表述为字母表 $\Sigma_t = \{0、1、.....、 t-1\}$上的规则。它也是通用后继规则 $h(\alpha)$ 的一种节省空间的实现，该规则在第 2.1 节中给出，且不需要存储共轭对集合。

第一个非零（Grandmama）后继规则，适用于 $\alpha = \mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n$
令 j为满足 $\mathtt{a}_2\mathtt{a}_3\cdots\mathtt{a}_n$且 $\mathtt{a}_j \neq 0$的最大下标；若不存在这样的下标，则令 $j = 1$。令 $\mathtt{x}$为集合{1、2、... ...、 t−1} 中满足 $0^{n-j}\mathtt{x}\mathtt{a}_2\cdots\mathtt{a}_j$是项链且其权重为 $\mathtt{x}\mathtt{a}_2\mathtt{a}_3\cdots\mathtt{a}_n) \leq w$的最大符号；若不存在这样的符号，则令 $\mathtt{x} = -1$。

$$h_1(\alpha) = \begin{cases} 0 & \text{if } \mathtt{x} \neq -1 \text{ and } \mathtt{a}_1 = \mathtt{x}; \\ \mathtt{a}_1{+}1 & \text{if } \mathtt{x} \neq -1 \text{ and } \mathtt{a}_1 < \mathtt{x}; \\ \mathtt{a}_1 & \text{otherwise.} \end{cases}$$

令 $\mathcal{U}_t(n,\ w)$ 表示 $\Sigma_t(n,\ w)$ 的通用循环：从 0^n 开始，在最后 n 个符号上反复应用后继规则 $|\Sigma_t(n,\ w)| - n$ 次，以获得序列中的下一个符号。例如：

$$\mathcal{U}_5(3,4) = \underline{0\cdot 001}\cdot 011\cdot 1\cdot 021\cdot 031\cdot 002\cdot 012\cdot 112\cdot 022\cdot 003\cdot 013\cdot 004.$$

注意到，该序列具有一个有趣的性质：它对应于按照反字典序，将 $\mathbf{N}_5(3,\ 4)$ 中的项链的非周期前缀拼接起来。由于当权重没有上界，即 $w \geq n(t-1)$时，首个非零后继规则等价于*Grandmama deBruijn 序列* [DHS+18]的后继规则，因此这并不令人意外。Grandmama 序列还可以通过以下非常简单的拼接方案来描述：

拼接　　按照反字典序，将 $\mathbf{N}_t(n)$ 中的项链的非周期前缀拼接在一起　　*r.*

应用定理1，可将这一拼接构造推广到集合 $\Sigma_t(n,w)$。令 $\mathbb{T}$ 表示由首个非零父节点规则定义的 $\Sigma_t(n,w)$ 的圈连接树，其中循环（节点）由 PCR 诱导 —— 它们对应于 $\mathbf{N}_t(n,w)$ 中的项链。根据引理 2，$\mathbb{T}$ 满足链性质，且所有非根周期节点都是叶节点。因此，令 T 表示由 $\mathbb{T}$导出的拼接树，其中根节点 0^n 的变化索引是最右侧符号的索引。将根节点的唯一子节点的标签定义为 $0^{n-1}1$。这解释了我们简化的拼接树定义中的一个歧义，并且这一选择符合 [SSTW24] 中原始定义的条件。回顾一下，所有其他节点的标签均根据其父节点的标签递归定义。由引理 2 可知，所有其他节点的标签都是项链。

▶ **定理 3.** 对于 $n, t \geq 1$和 $w \geq 0$，$\Sigma_t(n,w)$ 的通用循环 $U_t(n,w)$ 可通过按反字典序连接 $\mathbf{N}_t(n,w)$ 中的项链的非周期前缀来构造。

证明。 根据定理 1，将 T 中节点的非周期前缀按 RCL 顺序依次拼接得到的序列，其后继规则为 $h_1(\alpha)$，它生成 $\mathcal{U}_t(n,w)$ 且以 0^n 开头。每个节点的子节点都是左子节点

。因此， T 的 RCL 排序对应于对 T进行前序遍历。由于父节点与子节点之间变化的符号总是从父节点到子节点递增，因此这种排序会按反字典序列出节点（项链 $\mathbf{N}_t(n,w)$）。◀

按 RCL 顺序遍历拼接树 T，即可生成 $\mathrm{U}_t(n,w)$。为高效完成此操作，我们从根节点 0^n开始遍历 T，并动态计算每个节点的子节点。考虑一个变化索引为 c、权重为 w'的节点$\alpha = a_1a_2\cdots a_n$。如果 $w' = w$，那么 α没有子节点。令 α_i表示将索引 i 处的符号加 1 后得到的字符串 α。如果 α_i 不是 $1 < i \le c$ 的项链，那么显然 α_{i-1}也不是项链。如果 $a_c < k-1$，我们就检验 α_c 是否为项链。如果它不是项链，那么 α 没有子节点。否则，我们检验 α_{c-1}，然后检验 α_{c-2}，依此类推，直到找到最大的下标 $i \le c$，使得 α_i 不是项链，或者满足 $i = 0$。因此， α的子节点为 α_{i+1}、$\cdots$、 α_{c-1}；若 $a_c < k-1$，还包括 α_c。这样，从根节点0^n 开始，我们便可以动态构建树 T，并按 RCL 顺序访问其节点。检验字符串是否为项链，以及确定项链的非周期前缀长度，均可在 $O(n)$ 时间内完成 [Boo80]。每次成功的项链检验都可以归属到相应的子节点；如果项链检验失败，则可以归属到当前节点。因此，访问各节点的摊销时间为 $O(n)$。附录中给出了用于构造 $\mathrm{U}_t(n,w)$ 的完整 C 语言实现，该实现采用了这一 RCL 遍历算法。

▶ **定理 4.** *$U_t(n,w)$ 可用每个符号 $O(1)$ 摊还时间构造，使用 $O(nt)$ 空间 e.*

证明。根据引理 2，非根周期项链的父节点唯一且是非周期的。忽略根节点，这意味着周期项链的数量小于或等于非周期项链的数量。由于每个非周期项链的非周期前缀都包含 n 个符号，因此可知$|\mathrm{U}_t(n,w)| \ge \frac{n}{2}|\mathrm{N}_t(n,w) - \{0^n\}|$。拼接树中的每条项链 T都以 $O(n)$ 摊还时间访问，因此 $\mathrm{U}_t(n,w)$ 可用每个符号 O(1) 摊还时间生成。访问每个节点α时，只需常量级的额外内存；若更新并恢复 α 中的值，即可测试给定的α_i 是否为项链，并访问每个子节点。由于递归深度不超过 nt，此构造的运行时栈需要 $O(nt)$ 空间。◀

3.1 固定权重与权重范围

某些应用可能同时需要对权重设定上界和下界。通过考虑一种简写表示 [RSW12]，可以构造二进制固定权重德布鲁因序列。简写字符串对应于所有长度为 $n-1$、权重在范围 $[w-1,w]$ 内的二进制串。这一结果被应用于[SWW13]，以构造适用于权重处于任意范围 $[w_1,w_2]$ 内的二进制串的二进制权重范围德布鲁因序列。这两种构造都能以每个符号 O(1) 摊还时间生成序列，并使用 $O(n)$ 空间。目前尚无已发表的针对非二进制字母表上的权重范围德布鲁因序列的构造方法。作为一个特例，使用简写表示构造的、权重恰好为 w 的 $\Sigma_t(n)$ 中字符串的固定权重德布鲁因序列，对应于权重处于范围$[max(0,w-t+1)\ w]$内的 $\Sigma_t(n-1)$ 中字符串的权重范围德布鲁因序列。如果 $w < t$，那么这类序列就对应于权重上界为 w 的有界权重德布鲁因序列。

▶ **定理5.** 当 $w < t$时，通用循环$U_t(n,w)$ 是使用简写代表元的固定重量德布鲁因序列。此外，将 0^n 子串替换为 0^{n-1} 的序列$U_t(n,w)$ 在 $w = t$时是使用简写代表元的固定重量德布鲁因序列。

3.2 应用MSR

本节将介绍另一种有界权重德布鲁因序列构造，用于 $\Sigma_t(n,\ w)$，并增加约束 $w < t$。回顾一下， $\Sigma_t(n,w)$ 中的字符串可以视为.....的简写代表元

10 个 k-子集和 k-多重集的通用循环

包含固定权重 $w < t$ 字符串的 $\Sigma_t(n+1)$ 子集。因此，考虑这样的集合 $\Sigma_t(n,\ w)$ 时，集合中给定字符串的缺失符号 z 是明确确定的。还需回顾，MSR 将 $\Sigma_t(n,\ w)$ 划分为等价类，这些等价类对应于 $\mathbf{N}_t(n+1)$ 中的项链。

例2 考虑 $\Sigma_5(3,4)$。MSR 将这 35 个字符串划分为以下 10 个等价类（列）：

000	001	010	002	011	020	003	012	021	111
004	013	103	022	112	202	031	121	211	
040	130	030	220	120		310	210	110	
400	300	301	200	201		100	101	102	

每个等价类（循环）都可以表示为 $\mathbf{N}_5(4)$ 中一个权重为 $w = 4$的项链。沿每列向下读取首个符号，即可标出每条项链。

使用 MSR，可以按照应用 PCR 时采用首个非零 父节点规则的类似方式，构造圈连接树和后继规则。在这种情况下，节点的长度为 $n+1$。将首个非零符号递减后，根据 MSR 将下一个符号递增，从而保持权重约束。

首个非零（MSR）（根为 $0^n\mathtt{w}$）：非根节点 $\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_{n+1}$ 的父节点为 $0^{j-1}(\mathtt{a}_j-1)(\mathtt{a}_{j+1}+1)$ $a_{j+2}\cdots\mathtt{a}_{n+1}$，其中 j 表示满足 $\mathtt{a}_j \neq 0$ 的最小索引。

该父节点规则是良定义的，因为权重约束意味着 $j < n+1$ 和 $\mathtt{a}_{j+1} < t-1$。例如，请参见图 2中 $\Sigma_k(n,w)$ 对应的圈连接树。

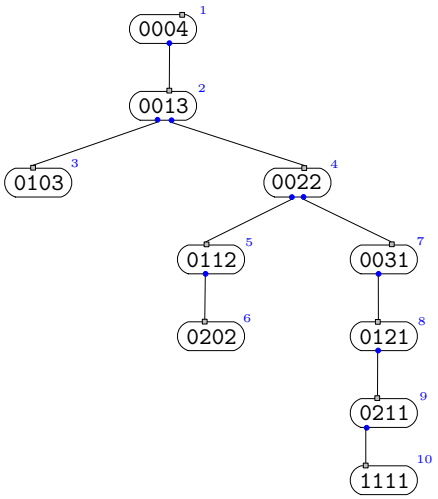


图2 基于 MSR 的 $\Sigma_5(3,\ 4)$ 中第一个非零循环合并树，其中节点由权重恰为 $w = 4$ 的$\mathbf{N}_5(4)$ 中的项链表示。该树的带标签前序遍历按逆反字典序访问这些项链。尽管基于 MSR 的循环合并树并未正式定义为拼接树，但此树仍以拼接树的样式绘制。

下面给出一个基于函数 $g(\alpha)$和 $h(\alpha)$ 的节省空间的后继规则（回顾第 2.1 节中的定义），该规则适用于这棵圈连接树，其构造基于相应的共轭对。

首个非零（MSR）后继规则，针对 $\alpha = \mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n$，缺失符号为 $\mathtt{z}$

设 j 为 $\mathtt{a}_2\mathtt{a}_3\cdots\mathtt{a}_n$ 中满足 $\mathtt{a}_j \neq 0$ 的最大下标，若不存在这样的下标，则为 $j = 1$。设 $\mathtt{x}$ 为{1，2，.....，$\mathtt{t}-1$} 中满足以下条件的最大符号：(i) $\mathtt{y} = \mathtt{z} - \mathtt{x} + \mathtt{a}_1$ 属于 Σ_t，且(ii) $0^{n-j}\mathtt{x}\mathtt{y}\mathtt{a}_2\cdots\mathtt{a}_j$ 是一条项链；若不存在这样的符号，则为 $\mathtt{x} = -1$。

$$h_2(\alpha) = \begin{cases} 0 & \text{if } \mathtt{x} \neq -1 \text{ and } \mathtt{z} = \mathtt{x}; \\ \mathtt{z}+1 & \text{if } \mathtt{x} \neq -1 \text{ and } \mathtt{z} < \mathtt{x}; \\ \mathtt{z} & \text{otherwise.} \end{cases}$$

设$\mathcal{V}_t(n,\ w)$表示 $\Sigma_t(n,\ w)$的通用循环，其构造方式为：从 0^n 开始，对最后 n 个符号反复应用上述后继规则 $|\Sigma_t(n,\ w)$共$| - n$ 次，以得到序列中的下一个符号。若采用朴素方法，为了确定 $\mathtt{x}$，函数 $h_2(\alpha)$ 最多需要测试 t 个字符串，以判断它们是否为项链。然而，借助一些相当直接的优化，我们可以将项链测试的次数减少到一次。

▶ **定理 6.** 对于 n、$t \geq 2$ 和 $0 \leq w < t$，$\Sigma_t(n, w)$ 的通用循环 $V_t(n, w)$ 可通过后继规则 $h_2(\alpha)$ 构造，每个符号耗时 $O(n)$，所用空间为 $O(n)$。

证明。我们证明，后继规则中的值 $\mathtt{x}$ 最多只需进行一次项链检验即可确定。设 $\beta = 0^{n-j}\mathtt{x}\mathtt{y}\mathtt{a}_2\cdots\mathtt{a}_j$。设 $\mathtt{x}'$ 表示 $\mathtt{a}_2\cdots\mathtt{a}_j$中紧随子串 0^{n-j} 之后的最小符号；若不存在这样的子串，则设 $\mathtt{x}' = \mathtt{t} - 1$。为使 β 成为项链，显然有 $\mathtt{x} \leq \mathtt{x}'$。此外，$\mathtt{x} + \mathtt{y} = \mathtt{a}_1 + \mathtt{z}$ 以维持权重约束，这意味着 $\mathtt{x} \leq \mathtt{a}_1 + \mathtt{z}$。将 $\mathtt{x}$ 视为 $\mathtt{x}'$与 $\mathtt{a}_1 + \mathtt{z}$ 的最小值。若该值为 0，则无需进行项链检验。若对于 $\mathtt{x}$的该取值，β 是一条项链，则无需更多检验，因为它是 $\mathtt{x}$的最大可能值。否则，将 $\mathtt{x}$ 的值减 1（并将y加 1）将得到一条项链，因为根据 $\mathtt{x}'$ 的取值以及 $\mathtt{y} \neq 0$这一事实，β 的长度为 $n - j + 1$ 的前缀将小于 β中所有其他相同长度的子串（循环地考虑）。◀

以下结果源于这样一个事实：$\Sigma_t(n, w)$ 中的字符串可以视为 $\Sigma_t(n + 1)$ 的简写代表元，其中包含权重固定为 $w < t$ 的字符串。

▶ **定理7。 </style}通用循环 V(n, w)** 在 $w < t$ 时，是一个使用简写代表元的固定重量德布鲁因序列。此外，将序列 $V_t(n, w)$ 中的 0^n 子串替换为 0^{n-1} 后，当 $w = t$ 时，该序列是一个使用简写代表元的固定重量德布鲁因序列。

有趣的是，$\mathcal{V}_t(n, w)$ 似乎也具有有趣的拼接性质。考虑：

$$\mathcal{V}_5(3{,}4) = \underline{0004} \cdot 0013 \cdot 0103 \cdot 0022 \cdot 0112 \cdot 02 \cdot 0031 \cdot 0121 \cdot 0211 \cdot 1.$$

该序列对应于下列项链的非周期前缀的拼接：$\mathbf{N}_5(4)$ 其权重恰为 $w = 4$，并按照 逆 colex 顺序排列！此外，注意到对相应的“类拼接树”（见图 2）进行前序遍历，也对应于按逆 colex 顺序访问这些项链。这些观察引出了以下猜想，其中 $\mathcal{V}'_t(n, w)$ 表示由以下项链的非周期前缀拼接而成的序列：$\mathbf{N}_t(n + 1)$ 权重为 w，并按照 逆 colex 顺序排列。

▶ **猜想 8。** 通用循环 $V_t(n,w)$ 等价于 $V'_t(n,w)$，其中 n、$t \geq 2$ 和 $w < t$。此外，通用循环可以按每个符号 $O(1)$ 的摊销时间生成。

我们认为，可以通过将（基于 PCR 的）拼接树框架 [SSTW24] 推广到其他反馈函数（如 MSR），然后对序列$\mathcal{U}_t(n,w)$ 进行类似的分析，从而证明这一猜想并得到运行时间结果。

4 对 k-子集和 k-多重集的应用

本节中，我们将上一节的结果应用于 k-子集，假设 $n \geq k \geq 1$；并应用于 k-多重集，假设 n，$k \geq 2$。回顾第 1 节中的以下观察。

- 当用差分代表表示 $\mathbf{S}_k(n)$ 的子集，且每个符号 x 映射到x−1 时，这些子集对应于 $\Sigma_{n-k+1}(k, n-k)$。
- 当用简写频数代表表示 $\mathbf{M}_k(n)$ 的多重集时，这些多重集对应于 $\Sigma_{k+1}(n-1,k)$。当用差分代表表示 $\mathbf{M}_k(n)$ 的多重集时，这些多重集对应于$\Sigma_n(k, n-1)$。

这些观察立即引出了以下三个结果。

▶ **定理 9.** $U_{n-k+1}(k, n-k)$ 和 $V_{n-k+1}(k, n-k)$ 是 $\mathbf{S}_k(n)$ 的通用循环，使用差分代表，其中代表中的每个符号 x 都映射为 x − 1。

例 3 考虑以下两个 $\Sigma_4(3,3)$ 的通用循环：

$\mathcal{U}_4(3,3)$ = 0 · 001 · 011 · 1 · 021 · 002 · 012 · 003，以及

$\mathcal{V}_4(3,3)$ = 0003 · 0012 · 0021 · 0111 · 0201。

将上述序列中的每个符号 x 映射为 x + 1 后，我们得到使用差分代表的 $\mathcal{S}_1$ 和 $\mathcal{S}_2$（属于 $\mathbf{S}_3(6)$）的通用循环：

$\mathcal{S}_1$ = 1 · 112 · 122 · 2· 132 · 113 · 123· 114，和

$\mathcal{S}_2$ = 1114· 1123· 1132 ·1222· 1312.

以上每个通用循环的长度均为预期值 $\binom{6}{3} = 20$。

▶ **定理10.** $U_{k+1}(n-1,k)$ 和 $V_{k+1}(n-1,k)$ 是使用简写频数代表的 $\mathbf{M}_k(n)$ 的通用循环。

▶ **定理11.** $U_n(k,n-1)$ 和 $V_n(k,n-1)$ 是使用差分代表的 $\mathbf{M}_k(n)$ 的通用循环。

例4 通用循环 $\mathcal{U}_5(3,4)$ 和 $\mathcal{V}_5(3,4)$ 是使用简写频数代表的 $\mathbf{M}_4(4)$ 的通用循环：

$\mathcal{U}_5(3,4)$ = 0 · 001 · 011 · 1 · 021 · 031 · 002 · 012 · 112 · 022 · 003 · 013 · 004，和

$V_5(3,4) = 0004 \cdot 0013 \cdot 0103 \cdot 0022 \cdot 0112 \cdot 02 \cdot 0031 \cdot 0121 \cdot 0211 \cdot 1$。通用循环 $U_4(4,3)$ 和 $V_4(4,3)$ 是使用差分代表的 $M_4(4)$ 的通用循环：$U_4(4,3) = 0 \cdot 0001 \cdot 01 \cdot 0011 \cdot 0111 \cdot 0021 \cdot 0002 \cdot 0102 \cdot 0012 \cdot 0003$，以及 $V_4(4,3) = 00003 \cdot 00012 \cdot 00102 \cdot 00021 \cdot 00111 \cdot 01011 \cdot 00201$。上述每个通用循环的长度均为预期的 $\binom{4+4-1}{4} = 35$。

上述三个结果表明，可以为 k个子集和 k个多重集构造通用循环，每个符号的摊销时间为 $\mathcal{O}(1)$；或者通过一个每个符号耗时 $\mathcal{O}(n)$ 的后继规则来构造。后两个结果是目前已知的首个针对 k个多重集的通用循环构造。

5 致谢

Joe Sawada（资助编号 RGPIN-2025-03961）衷心感谢加拿大自然科学与工程研究委员会（NSERC）对本研究的支持。

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 离散数学杂志》，25(4):1832–1842，2011。Boo80 K. S. Booth。字典序最小的循环子串。《信息处理快报》，10(4/5):240–242，1980。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。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。CHHM09 D. Curtis、T. Hines、G. Hurlbert 和 T. Moyer。子集存在近通用循环。《SIAM 离散数学杂志》，23(3):1441–9，2009。CWLW24 C. S. Chen、W. S. Wong、Y.-H. Lo 和 T.-L. Wong。用于近距离传感器网络的多重集组合格雷码，2024。dbs26 德布鲁因序列和通用循环构造（2026）。http://debruijnsequence.org.，2026。DHS+18 P. B. Dragon、O. I. Hernandez、J. Sawada、A. Williams 和 D. Wong。利用反字典序构造德布鲁因序列：k元 Grandmama 序列。《欧洲组合学杂志》，72:1–11，2018。DL16 M. Debski 和 Z. Lonc。 n元集合的 k子集的通用循环装填与覆盖。《图与组合》，32(6):2323–2337，2016年11月。GJKO20 S. Glock、F. Joos、D. Kühn 和 D. Osthus。超图中的欧拉回路。《组合学》，40(5):679–690，2020年11月。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。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。德布鲁因循环与格雷码的推广/图非对称性/顶点传递（凯莱）图的哈密顿性问题。

Hur94 G. Hurlbert。关于 n元集合的 k子集的通用循环。《SIAM 离散数学杂志》，7(4):598–604，1994。Jac93 B. Jackson。 k排列和 k子集的通用循环。《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。Lan12 M. Lanius。 n元集合的 k子集的通用循环。荣誉学位论文，韦尔斯利学院，2012。RSW12 F. Ruskey、J. Sawada 和 A. Williams。定重二进制串的德布鲁因序列。《SIAM 离散数学杂志》，26(2):605–617，2012。Rud13 Y. Rudoy。对 $[n]$的 k子集构造通用循环的归纳方法。《电子组合学杂志》，20:P18，2013。SBE+02 B. Stevens、P. Buskell、P. Ecimovic、C. Ivanescu、A. Malik、A. Savu、T. Vassilev、H. Verrall、B. Yang 和 Z. Zhao。解决一个悬而未决的猜想：不存在具有 $k = n - 2$的通用循环。《离散数学》，258:193–204，2002年12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。SW13 J. Sawada 和 A. Williams。定密度项链和林登词的常数摊销时间格雷码。《理论计算机科学》，502:46–54，2013。组合结构生成。SW23 J. Sawada 和 A. Williams。构造首个（也是最酷的）定内容通用循环。《Algorithmica》，85(6):1754–1785，2023年6月。SWW13 J. Sawada、A. Williams 和 D. Wong。重量范围二进制串的通用循环。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—12日，LNCS 8288，第388–401页，2013。SWW14 J. Sawada、A. Williams 和 D. Wong。具有指定最小重量的二进制串的字典序最小通用循环。《离散算法杂志》，28:31–40，2014。StringMasters 2012、2013 特刊（第1卷）。SWW16 J. Sawada、A. Williams 和 D. Wong。德布鲁因序列和通用循环的经典贪心构造与项链构造的推广。《电子组合学杂志》，23(1):论文1.24，20，2016。

有界重量“Grandmama”德布鲁因序列的 C 语言实现

```
//=========================================================================/
  /生成（上界）有界重量“Grandmama”德布鲁因序列
// 字母表为 {0,1, … , t-1}，且权重上界为 w。
// 可将其应用于构造 ［n］的 k-子集或 k-多重集的通用循环。
//=========================================================================#
```

```
//=============================================
```

16 k-子集与 k-多重集的通用循环

```
int 主函数()  {
    printf("输入 n t w:          ");  scanf("%d %d %d",  &n, &t, &w);
    对于 （int i=1; i<=n;  i++) a[i] = 0;
    RCL(a,1,n,0);
}
```