arXiv:2603.20790v1 [cs.DS] 21 Mar 2026

（集合）补散列因子

Duncan Adamson[1], Pamela Fleischmann[2], 和 Annika Huch[2]

[1] 英国圣安德鲁斯大学 duncan.adamson@st-andrews.ac.uk[2] 德国基尔大学， {fpa,ahu}@informatik.uni-kiel.de

摘要。自 20 世纪 70 年代 Imre Simon 的奠基性工作以来，散布因子（也称为子序列或散布子词）一直是备受持续关注的研究对象。关于散布因子的研究大多可分为两大类问题：给定一个词，哪些信息以散布因子的形式包含在其中，哪些不包含。本文考虑一个介于两者之间的问题，并引入补散布因子的概念。给定一个词 w 以及其散布因子 u， w的补散布因子是指相对于 u， w 的补散布因子集合 $C(w, u)$：从 w中移除 u 的任意一个嵌入后，可在 w 中形成的散布因子集合。这与交错运算密切相关；在交错运算中，两个词相互交织。换言之，我们利用关于散布因子的知识，扩展了以往有关交错运算符的研究。除了引入这些集合外，我们还给出了关于集合 $C(w, u)$ 大小的组合学结果；提出一种算法，可在 $O(|w| \cdot |u| \binom{w}{u})$ 时间内，根据 w 和 u 计算集合 $C(w, u)$，其中 $\binom{w}{u}$ 表示 u 嵌入 w 的嵌入数量；还提出一种算法，可在 $O(|w|^2 \binom{|w|}{|w|-|u|})$ 时间内，根据 w 和 $C(w, u)$ 构造 u；以及一种算法，可在 $O(|u| \cdot |w|^{|u|+1})$ 时间内，根据 u和 $C(w, u)$ 构造 w。

1 引言

对于给定的词 w，散布因子 （（散布）子词或子序列）是通过删除 w 中的某些字母，同时保留其余字母的顺序而得到的词。具体来说，长度为 n 的词 u 是词 w 的散布因子，当且仅当存在（可以为空的）词 $v_1, \ldots, v_{n+1}$，使得 $w = v_1 u[1] v_2 u[2] \cdots v_n u[n] v_{n+1}$，其中 $u[i]$ 表示 u 的第 i^{th} 个字母。例如，pile,nap 和 pipe 是 pineapple 的散布因子，而 peel 和 lane 则不是，因为这些字母的顺序不正确。由于散布因子可作为衡量一个词中信息缺失或存在情况的复杂性指标，组合词学、字符串学、语言理论和自动机理论都研究词与其散布因子之间的关系（有关这一主题的深入概述，请参见 [50,第 6 章：子词] ）。散布因子方面的研究始于 Imre Simon 在 20 世纪 70 年代的博士论文，其中包含关于分段可测试语言的奠基性工作 [60]。Simon 提出了这种关系，如今称为 Simon同余：若两个词具有相同的、长度不超过给定值的散布因子，则称它们同余。

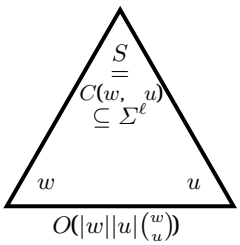
长度为 k。迄今为止，这种同余关系的指数仍然未知。要进一步了解该指数，可以研究 m-近乎 k-普适词 [8, 26, 17, 25, 58]：若一个词恰好有 m 个长度为 k 的词不是它的散布因子，则称该词为 m- 近乎 k- 普适 。普适性的概念与 k- 丰富性的概念密切相关（参见 [37–39]）。近年来，Simon同余在语言理论 [42, 43, 2, 24, 1] 和模式匹配 [28, 44] 中受到关注。此外，散布因子也在组合词学 [47, 53],、算法和字符串学 [6, 7, 11, 30, 51, 62], 、逻辑学 [9, 33, 48, 49], ，以及语言理论和自动机理论 [2, 37, 40, 59, 60, 64] 中得到研究。从更具应用性的角度来看，散布因子用于生物信息学 [19, 63], 、形式化软件验证 [33, 64] 和数据库理论 [4, 31, 45, 46, 57]。此外，在重构这一理论问题的背景下，散布因子也用于对损坏数据建模 [20, 29, 52]。

正如在简要讨论 m-近乎 k-普适词的概念时提到的，人们不仅可以研究一个词的散布因子，也可以根据其缺失的散布因子来刻画词。例如，lane 是 pineapple 的一个缺失的散布因子（参见 [3, 26, 27, 47, 61]）。在生物信息学中，人们不仅关注缺失的散布因子，也会考虑 DNA 通过（删除型）DNA 存储信道 [56, 32]时产生的删除（或插入、替换）后的剩余部分。经过对文献的全面检索，似乎迄今为止还没有人从组合学的角度研究两个给定词 u和 w 的剩余词，即散布因子的补集，也就是说，如果从 w中删除 u ，所得到的词的集合。举例来说，ala 在 alfalfa 中出现（嵌入）了三次，分别删除这三处后，我们得到$C(w, u)$ $=$ {flfa、falf、lfaf}。在这种情况下，补集的基数等于被删除词的嵌入数；但若考虑 peelwheel 和 peel，则得到 $|C(w, u)| = 4$ ，尽管其嵌入数为七。这个概念与交错算子的概念紧密相关，只不过视角相反。给定两个词 u 和 v，交错 $u \sqcup\!\sqcup v$ 是由所有满足以下条件的词 w 组成的集合：从 w 中删除 u 后得到 v ，并且从 w 中删除v 后得到 u，即
$u \sqcup\!\sqcup v = \{w \in \Sigma^* | \exists k, \ell \in \mathbb{N} \exists u_1, \ldots, u_k, v_1, \ldots, v_\ell \in \Sigma^* : \quad u = u_1 \cdots u_k$、
$v = v_1 \cdots v_k$、 $w = u_1 v_1 \cdots u_n v_n\}$。例如，给定 lop 和 apt，我们有 laptop $\in$ lop $\sqcup\!\sqcup$ apt：删除第一个字母和最后两个字母后得到 apt，而删除第二至第四个字母后得到 lop。交错在程序验证中有应用，例如可用作建模进程代数 [5]的工具。这个定义可以推广到语言，正则语言上的交错已得到充分研究 [23, 10]。关于交错与形式语言的更多结果见 [14, 16]。在交错研究领域中，出现了两个重要的分支：完美交错和自交错（或交错平方）。对于完美交错，必须从两个词中交替取字母（参见 [18, 22]）。请注意，与一般问题中两个词的交错是一个集合不同，两个词的完美交错只是一个词。例如，bnn 和 aaa 的完美交错得到 banana。对于自交错，研究的只是一个词与其自身的交错。因此，给定 $w =$ abc，我们得到 $w \sqcup\!\sqcup w =$ {abcabc、abacbc、ababcc、aabcbc、aabbcc}。

近年来，已有多篇论文研究自交错的概念（例如，见 [34, 35, 15]）。这些研究考察了无平方因子词的交错，即不含形如 vv 的因子的词，其中 v是一个非空词。 [13] 和 [55], 独立证明了判定一个词是否为自交错的问题是 NP完全，尽管当时对字母表有所限制。最近，[12] 证明了该结论对一般的二元字母表也成立。关于自交错近期进展和开放问题的综述，参见 [54]。回到补散布因子，我们关注集合 $C(w,\ u) = \{v \in \Sigma^* | \, w \in u \sqcup v\}$ 及其性质。

我们的贡献。本文提出补散列因子的概念，并从算法和组合两个角度展开研究。首先，我们基于动态规划方法计算给定 w 和 u 时的集合 $C(w, u)$。此外，我们针对两个逆问题给出算法：（1）给定一个词 u 和一个集合 S，判断是否存在一个词 w ，使得 $S = C(w, u)$；若存在，则计算 w；（2）给定一个词w 和一个集合 S，判断是否存在 w 的一个散布因子 u ，使得$S = C(w, u)$；若存在，则计算 u。图1直观展示了这三个相关问题的概况，其中运行时间指相应算法的运行时间；相邻边上的两个值分别是输入，而另一侧的值是输出。此外，我们还从词组合学的角度研究补散列因子。首先，我们借助拱或 α-β-因式分解等工具，给出恰有一个或多个补散列因子的特殊散布因子。接下来，我们研究满足以下条件的词集合： $C(w, u) = 1$ 当且仅当 $E(w, u) = 1$，其中 $E(w, u)$ 表示 u 在 w 中的嵌入数量。最后，我们将补散列因子的概念与（自）交错运算联系起来，并刻画通过完美交错运算得到的词中的补散列因子。

本文结构。 第2节介绍所有必要的定义。随后，在第3节中，我们介绍计算 $C(w,\ u)$ 的算法研究。接着，在第4节中，我们从组合词学的角度研究补散布因子这一对象。



定理 11

图 1. 定理11、14和20的算法结果概要。

2 预备知识

令 $\mathbb{N}$ 为自然数集合，$\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, $[n] = \{1, \ldots, n\}$, $[n]_0 := [n] \cup \{0\}$。令 M, N 为集合，且 $(f_i)_{i \in \mathbb{N}}$ 是从 M 到 N 的一族函数。对所有 $i \in \mathbb{N}$，令 $f_i(M) = \{f_i(m) \mid m \in M\}$。对于互不相同的 $i, j \in \mathbb{N}$，若 $f_i(M) \cap f_j(M) = \emptyset$，则称 f_i 和 f_j 不相交；若 $f_i(M) \cup f_j(M) = N$，则称它们穷尽。若两个函数 $f, g\colon M \to N$ 不相交且穷尽，则记作 $f \dot\cup g$。

字母表 Σ 是一个非空有限集合，其元素称为字母或符号。对于给定的 $\sigma \in \mathbb{N}$，我们固定 $\Sigma = \{\mathsf{a}1, \ldots, \mathsf{a}_\sigma\}$。词是由 Σ中的字母构成的有限序列。令 Σ^* 为 Σ 上所有有限词构成的集合，其中连接运算和空单词 ε 分别作为运算和的单位元。令 $\Sigma^+ := \Sigma^* \setminus \{\varepsilon\}$。令 $w \in \Sigma^*$。对所有的 $n \in \mathbb{N}_0$，归纳定义 $w^0 = \varepsilon$ 和 $w^n = ww^{n-1}$。w 的长度是其字母的数量；因此 $|\varepsilon| = 0$。对所有的 $k \in \mathbb{N}_0$，令$\Sigma^k := \{w \in \Sigma^* \mid |w| = k\}$（类似地定义 $\Sigma^{\leq k}$, $\Sigma^{\geq k}$）。我们将 w的i^{th}个字母记为 $w[i]$；若 $i < j$，则令 $w[i..j] = w[i] \cdots w[j]$；若对所有$i, j \in [|w|]$均有 $i > j$，则令 ε。令 $\mathsf{alph}(w) = \{\mathsf{a} \in \Sigma \mid \exists i \in [|w|]\colon w[i] = \mathsf{a}\}$。若存在 $x, y \in \Sigma^*$ 使得 $w = xuy$，则词 $u \in \Sigma^*$ 是 w 的因子。当 $x = \varepsilon$时，我们称u 为 w 的前缀；若 $y = \varepsilon$，则称其为后缀。对于某个 $\mathsf{a} \in \Sigma$，词$w \in \Sigma^*$ 的因子 aa 称为 w 的字母平方；对于某个 $y \in \Sigma^*$，若因子 $yy \in$ 属于 $\mathsf{Fact}(w)$，则称其为 w中的平方。将 w 的逆序定义为 $w^R = w[|w|] \cdots w[1]$。令 $<_\Sigma$为 Σ 上的全序，并将 Σ^* 上的字典序记为 $<$。更多定义请参见 [50]。介绍完这些基本记号后，我们将引入散布因子。

定义 1. 令 $w \in \Sigma^*$ 和 $n \in \mathbb{N}_0$。若词 $u \in \Sigma^n$ 满足……，则称其为 w 的散布因子，记作 $u \in \mathrm{ScatFact}(w)$。若存在 $v_1, \ldots, v_{n+1} \in \Sigma^*$ 使得$w = v_1u[1]v_2u[2] \cdots v_nu[n]v_{n+1}$，则成立。令 $\mathrm{ScatFact}_k(w) = \{u \in \mathrm{ScatFact}(w) \mid |u| = k\}$。

现在我们定义散布因子出现位置的集合，它由映射给出；这些映射将散布因子 u 的索引对应到 w中的相应索引。

定义 *2.* $u \in \Sigma^*$ 在 $w \in \Sigma^*$ 中的嵌入定义为映射$e\colon \{1, \ldots, |u|\} \to \{1, \ldots, |w|\}$，满足 $e(1) < \cdots < e(|u|)$，且 $u[i] = w[e(i)]$ 对所有 $i \in [|u|]$均成立。我们用 $\binom{w}{u}$ 表示 u在 w中的嵌入数量，即，$\binom{w}{u} = |\{e\colon \{1, \ldots, |u|\} \to \{1, \ldots, |w|\} \mid e$ 是 u 在 $w\}|$中的嵌入。对于给定的嵌入 e，令 $sf(e)$ 表示由 e定义的散布因子。u 在 w 中的所有嵌入的集合记为 $E(w, u)$。对于 $e \in E(w, u)$，令 e 表示词 v 的嵌入，使得 $e \dot\cup e$。记 $E_=(w, u)$为 u 在 w 中的嵌入，其中对所有 $e_1, e_2 \in E_=(w, u)$，均有 $sf(e_1) = sf(e_2)$。

词 cau，cafe、life 和 ufo 都是 cauliflower 的散布因子，但 flour 和 row 都不是。根据排列的元组表示法，我们用一个 $|u|$元组来表示散布因子 u 在词 w 中的一种嵌入，其中元组的第 i^{th} 个元素包含 w 中嵌入了 u 的第 i^{th} 个字母的位置索引。例如，cafe 在 cauliflower 中唯一的嵌入表示为 (1，2，6，10)。注意，散布因子可能有多种嵌入，例如 low 的嵌入为 (4，8，9) 和 (7，8，9)。在 bananaban 中，ban 有多种出现方式，有些彼此不相交，如 (1，2，3)、(7，8，9)，有些则不相交，如 (1，2，9)、(1，4，5)。

词 banana 和 ban 在 bananaban 中有一种互不相交且穷尽的嵌入。我们继续讨论 [8, 26] 中引入的（近乎）普适性概念。

定义 3。 若词 $w \in \Sigma^*$ 满足 m-近乎 k-普适，则称其为 $|\mathrm{ScatFact}_k(w)| = \sigma^k - m$。若 $m = 0$，则称这些词为 k-普适。使词成为 k-普适的最大 $k \in \mathbb{N}_0$ 称为普适性指标 $\iota(w)$。

在一元字母表中，所有词都具有 a^ℓ 的形式，其中 $\ell \in \mathbb{N}$，且散布因子很容易确定。因此，我们只考虑大小为二或更大的字母表。此外，对于给定的 w，我们假设 $\Sigma = \mathsf{alph}(w)$，除非另有说明。研究词的散布因子的主要工具之一是拱分解，它由 Hébrard [36] 引入。

定义 4。 对于词 $w \in \Sigma^*$，其拱分解定义为 $w = \mathrm{ar}_1(w) \cdots \mathrm{ar}_k(w)\, \mathrm{r}(w)$，其中 $k \in \mathbb{N}_0$，且
(1) $\iota(\mathsf{ar}_i(w)) = 1$ 对所有 $i \in [k]$, *(2)* $\mathsf{ar}_i(w)[|\mathsf{ar}_i(w)|] \notin \mathsf{alph}(\mathsf{ar}_i(w)[1..|\mathsf{ar}_i(w)| - 1])$ 对所有 $i \in [k]$, 以及 *(3)* $\mathsf{alph}(\mathsf{r}(w)) \subsetneq \Sigma$。

词 $ar_i(w)$ 称为 w 的弧，而 $r(w)$ 是 w的剩余部分。模态词 $m(w)$ 定义为 $ar_1(w)[|ar_1(w)|] \cdots ar_k(w)[|ar_k(w)|]$，也就是说，它是由每个拱的唯一末字母组成的词。$i^{th}$ 的拱的内部部分定义为 $ar_i(w)$ 的前缀，使得 $ar_i(w) = in_i(w)\, m(w)[i]$ 成立。

与散布因子的概念密切相关的是交错（参见 [21]）。

定义 5。 对于 $u, v \in \Sigma^*$，将 u 和 v 的交错定义为 $u \sqcup\!\sqcup v = \{w \in \Sigma^{|u|+|v|} \mid \exists e_1 \in E(w,u)\ \exists e_2 \in E(w,v) : e_1 \dot\cup e_2\}$。对于 $u, v \in \Sigma^m$，对 $m \in \mathbb{N}$ 定义 u和 v 的完美交错为 $u \sqcup\!\sqcup v = u[1]v[1]u[2]v[2] \cdots u[|u|]v[|u|]$。集合$u \sqcup\!\sqcup u$ 称为 u的自交错。

例如，设 $u = \mathsf{ban}$ 和 $v = \mathsf{ana}$。因此，交错包含长度为 6 的词，且我们有 $|u \sqcup\!\sqcup v| = 11$。一般而言，我们有 $|u \sqcup\!\sqcup v| \leq \binom{|uv|}{|u|}$。

定义 6. 对于 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$，我们将 u在 w中的补散布因子集定义为 $C(w,u) = \{v \in \Sigma^{|w|-|u|} \mid w \in u \sqcup\!\sqcup v\}$。每个$v \in C(w,u)$ 都称为 u 在 w中的一个补散布因子。

注意，根据定义，如果 $v \in C(w,u)$，则 $u \in C(w,v)$。本节最后给出一个关于自交错及其嵌入的引理（参见 [13,数学预备知识]）。

引理 7. 对于 $v \in \Sigma^*$ 和 $w \in v \sqcup\!\sqcup v$，存在 $e_1, e_2 \in E(w,v)$，满足 $e_1 \dot\cup e_2$，使得对于所有 $k \in [|v|]$，都有 $e_1(k) < e_2(k)$。若对于所有 $e_3 \in E(w,v)$，都存在 $j \in [|v|]$，使得 $e_1(j) \leq e_3(j)$，我们称 e_1 为 v 在 w 中的第一次出现，称 e_2 为第二次出现。

对于算法结果，我们采用具有对数级字长的标准计算模型随机存取机（RAM）（参见例如 [41]），即遵循字符串学中的一个标准假设：如果 w 是我们算法的输入字，那么我们假设$\Sigma = \mathsf{alph}(w) = \{1, 2, \ldots, \sigma\}$。

3 计算 $C(w,\ u)$

我们首先介绍算法 1，该算法通过动态规划方法，根据给定的两个词 $w \in \Sigma^*$ 和 $u \in \text{ScatFact}(w)$计算补散布因子集。我们维护一个动态规划矩阵 P，其中 $P[i,j]$ 包含 $u[1..i-1]$ 在 $w[1..j]$中的补散布因子集。

算法 1计算 $C(w,\ u)$

输入： $w \in \Sigma^*$ 和 $u \in \text{ScatFact}(w)$

输出： 集合 $C(w,\ u)$。

```
创建一个 |u|+1 × |w|—表 P 初始化 填充 使用 ∅
// 初始化 的 首个 列
P[1,1] = {w[1]}
if w[1] = u[1] 那么
  P[2,1] = {ε}
对于 j ∈ [2,|w|]:
  对于 i ∈ [2,|u|+1]:
    P[1,j] = {w[1..j]}
    如果 i−1 > j 那么
      // 这里的 前缀 比 u 的 前缀 w 的
      P[i,j] = ∅
    否则
      if w[j] = u[i−1] 那么
        P[i,j] = P[i−1,j−1] ∪ P[i,j−1]·w[j]
        // 第二项 并集为空 如果 P[i,j−1] = ∅
      否则
        P[i,j] = P[i,j−1]·w[j]
返回 P[|u|+1,|w|]
```

命题 8. 对于 $w \in \Sigma^*$、 $u \in \text{ScatFact}(w)$、 $i \in [|u|+1]$ 和 $j \in [|w|]$，有 $P[i,j] = C(w[1..j], u[1..i-1])$。

证明. 我们对 j 进行归纳来证明此命题。对于基例，考虑$j = 1$，即我们要证明 $P[i,1] = C(w[1], u[1..i-1])$。我们有

$$P[i,1] = \begin{cases} \{w[1]\}, & \text{if } i = 1 \\ \{\varepsilon\}, & \text{if } w[1] = u[1] = C(w[1], u[1..i-1]) \\ \emptyset, & \text{otherwise.} \end{cases}$$

对于归纳步骤，我们假设 $P[i,j] = C(w[1..j], u[1..i-1])$。根据算法 1，我们分三种情况讨论。

情况（$i - 1 > j + 1$）。根据算法 1，我们有 $P[i,j+1] = \emptyset = C(w[1..j+1], u[1..i-1])$，因为 $|w[1..j+1]| = j + 1 < i - 1 = |u[1..i-1]|$。

情况（$i-1 \le j+1$）。我们根据 $w[j+1]$ 相对于 $u[i-1]$ 的值对这种情况进行细分。

子情况（$w[j+1] = u[i-1]$）。这里有

$$\begin{aligned} P[i,j+1] &= P[i-1,j] \cup P[i,j] \cdot w[j+1] \\ &= C(w[1..j], u[1..i-2]) \cup C(w[1..j], u[1..i-1]) \cdot w[j+1] \\ &\overset{(*)}{=} C(w[1..j+1], u[1..i-1]). \end{aligned}$$

其中（$*$）可通过两个包含关系证明。首先，令 $v \in C(w[1..j],\ u[1..i-2]) \cup C(w[1..j],\ u[1..i-1]) \cdot w[j+1]$ 如果 $v \in C(w[1..j],\ u[1..i-2])$，那么，由于 w 的第（$j+1$）th 个字母与 u 的第（$i-1$）th 个字母相同，$v \in C(w[1..j],\ u[1..i-1])$。否则，如果 $v \in C(w[1..j],\ u[1..i-1]) \cdot w[j+1]$，则可得 $v \in C(w[1..j],\ u[1..i-1])$，这是因为 v 是包含字母 $u[i-1]$ 的补散布因子。

其次，令 $v \in C(w[1..j], u[1..i-1])$。那么，$v$ 可以通过使用匹配的字母构造，即 $v \in C(w[1..j], u[1..i-1])$，$\cdot w[j+1]$ 也可以通过跳过这个字母构造，即 $v \in C(w[1..j], u[1..i-2])$。

子情况 $(w[j+1] \neq u[i-1])$。此情况的论证与上一种情况中并集的第二部分相同，因为我们只能跳过 u 中不匹配的字母，即当 v 时使用它。

证明至此完成。 □

由命题 8 可立即得到关于每个条目 $P[i,j]$ 的长度以及仅存储空集的情形的以下两个结果（即，u 的前缀不是 w 的前缀的散布因子）。

推论 9。对于 $w \in \Sigma^*$、$u \in \mathrm{ScatFact}(w)$，$i \in [|u|+1]$ 和 $j \in [|w|]$，对于所有 $v \in P[i,j]$，我们有 $|v| = |w[1..j]| - |u[1..i-1]|$。

推论 10。对于 $w \in \Sigma^*$、$u \in \mathrm{ScatFact}(w)$，$i-1 > j$ 我们有 $P[i,j] = \emptyset$。

因此，通过计算表 P，我们可以得到两个给定词 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$ 的补散布因子集。

定理 11。给定 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$，我们可以在时间 $O(|w||u|\binom{w}{u})$ 内计算集合 $C(w,u)$，并使用空间 $O(|u|\binom{w}{u})$。

证明。根据命题 8，计算的正确性显然，因为我们有 $P[|w|, |u|+1] = C(w,\ u)$。为了推导运行时间，注意到 $P[i,j]$ 的值可以在 $O(\binom{w}{u})$ 时间内由 $P[i-1,j-1]$ 和 $P[i,j-1]$ 的值计算得出。由于表中有 $|w||u|$ 个条目，因此可得所述运行时间。□

表 1 给出了算法 1 在输入 $C(\text{ananas},\ \text{as})$ 上的示例。此外，算法 2 是算法 1 的扩展版本，不仅可以计算补散列因子的集合，还可以计算它们的重数（该算法并非通过匹配 w 和 u 的前缀，而是通过匹配它们的后缀来实现）。

	$w[1..6]$	$w[2..6]$	$w[3..6]$	$w[4..6]$	$w[5..6]$	$w[6]$
	a	an	ana	anan	anana_	ananas_
ε	{a}	{an}	{ana}	{anan}	{anana}	{ananas}
$u[1] =$ a_	$\{\varepsilon\}$ $\xrightarrow{\cdot n}$	{n}	$\xrightarrow{\cdot a}$ {an,na}	$\xrightarrow{\cdot n}$ {ann, nan}	$\xrightarrow{\cdot a}$ {anna, nana anan}	$\xrightarrow{\cdot s}$ {annas, nanas, anans}
$u[1..2] =$as_	$\emptyset$ $\xrightarrow{\cdot a}$	$\emptyset$ $\xrightarrow{\cdot n}$	$\emptyset$ $\xrightarrow{\cdot a}$	$\emptyset$ $\xrightarrow{\cdot n}$	$\emptyset$ $\xrightarrow{\cdot s}$	{anna, nana anan}

表 1. 计算 C(ananas，as)。

算法 2 通过后缀匹配和计数计算 $C(w,\ u)$

输入： $w \in \Sigma^*$ 和 $u \in$ ScatFact(w)

输出： 集合 $C(w,\ u)$，并为每个元素记录其出现次数的重数-素。

创建一个 $|u|+1 \times |w|$–表格 P 初始化 填满 使用 $\emptyset$
// 初始化 的 最后 列
$P[1, |w|] = \{(w[|w|], \mathbf{1})\}$
if $w[|w|] = u[|u|]$ then
  $P[2, |w|] = \{(\varepsilon, \mathbf{1})\}$
当 $j \in [|w|-1, 1]$:
  $P[1, j] = \{(w[j..|w|], \mathbf{1})\}$
  当 $i \in [2, |u|+1]$:
    如果 $i-1 > |w|-j+1$，那么
      // 这里的后缀 of u 比后缀更长 w 的
      //（其长度为 $|w|-j+1$）
      $P[i, j] = \emptyset$
    否则
      if $w[j] = u[i-1]$ 那么
        $P[i, j] =$带重数的并集 ($P[i-1, j+1]$, $P[i, j+1]$, $w[j]$)
        // 第二项 并集的为空 如果 $P[i, j-1] = \emptyset$
      否则
        $P[i, j] = \{(w,\ n) \mid w \in P[i, j+1]$.first() $\cdot\, w[j] \wedge P[i, j+1]$.snd()$\}$
return $P[|u|+1, |w|]$

// 类型 $\mathcal{P}(\Sigma^* \times \mathbb{N}) \times \mathcal{P}(\Sigma^* \times \mathbb{N}) \times \Sigma \to \mathcal{P}(\Sigma^* \times \mathbb{N})$
def 并集及其重数（compSet1, comp集合2 ,letter):
// 使用并集 查找 数据结构
  compSet= $\emptyset$
  对于 所有 $w \in$letter ·compSet1 .first()∪compSet2 .first():
    if $w \in$字母 ·compSet1 .first()∩compSet2 .first():
      compSet=compSet∪(w, compSet1 . snd ()+compSet2 . snd ( ))
    否则 if $w \in$letter·compSet1 . 首个 ():
      comp集合=comp集合∪(w, comp集合1 . snd ( ))
    否则:
      补集=补集∪(w, 补集2 . snd ( ))
  返回补集

例 *12.* 我们依次构造 $w = \mathsf{ababa}$ 和 $u = \mathsf{aba}$ 的补散布因子（为便于阅读，这里补充了表格说明，其中带下划线的字母是算法关注的对象）。首先是最后一列（代码的前三行）：

	$w[1..5]$ $\underline{\mathsf{a}}\mathsf{baba}$	$w[2..5]$ $\mathsf{b}\underline{\mathsf{a}}\mathsf{ba}$	$w[3..5]$ $\underline{\mathsf{a}}\mathsf{ba}$	$w[4..5]$ $\underline{\mathsf{b}}\mathsf{a}$	$w[5]$ $\underline{\mathsf{a}}$
ε	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{(\mathsf{a},\ 1)\}$
$u[3] = \mathsf{a}_$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{(\varepsilon, 1)\}$
$u[2..3] = \mathsf{b}\underline{\mathsf{a}}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$u[1..3] = \mathsf{a}\underline{\mathsf{b}}\mathsf{a}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

现在，我们遍历 w，然后遍历 u，也就是说，按列从右向左填充表。

	$w[1..5]$ $\underline{\mathsf{a}}\mathsf{baba}$	$w[2..5]$ $\mathsf{b}\underline{\mathsf{a}}\mathsf{ba}$	$w[3..5]$ $\underline{\mathsf{a}}\mathsf{ba}$	$w[4..5]$ $\underline{\mathsf{b}}\mathsf{a}$	$w[5]$ $\underline{\mathsf{a}}$
ε	$\emptyset$	$\emptyset$	$\emptyset$	$\{(\mathsf{ba},\ 1)\}$	$\{(\mathsf{a},\ 1)\}$
$u[3] = \mathsf{a}_$	$\emptyset$	$\emptyset$	$\emptyset$	$\{(\mathsf{b},\ 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\varepsilon, 1)\}$
$u[2..3] = \mathsf{b}\underline{\mathsf{a}}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{\varepsilon, 1\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$
$u[1..3] = \mathsf{a}\underline{\mathsf{b}}\mathsf{a}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$

请回想一下，只有当 u 和 w 中相应的字母匹配时，我们才使用对角值。

	$w[1..5]$ $\underline{\mathsf{a}}\mathsf{baba}$	$w[2..5]$ $\mathsf{b}\underline{\mathsf{a}}\mathsf{ba}$	$w[3..5]$ $\underline{\mathsf{a}}\mathsf{ba}$	$w[4..5]$ $\underline{\mathsf{b}}\mathsf{a}$	$w[5]$ $\underline{\mathsf{a}}$
ε	$\emptyset$	$\emptyset$	$\{(\mathsf{aba},\ 1)\}$	$\{(\mathsf{ba},\ 1)\}$	$\{(\mathsf{a},\ 1)\}$
$u[3] = \mathsf{a}_$	$\emptyset$	$\emptyset$	$\{(\mathsf{ba},\ 1),$ $(\mathsf{ab},\ 1)\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\{(\mathsf{b},\ 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\varepsilon, 1)\}$
$u[2..3] = \mathsf{b}\underline{\mathsf{a}}$	$\emptyset$	$\emptyset$	$\{\mathsf{a}\ 1\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\{\varepsilon, 1\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$
$u[1..3] = \mathsf{a}\underline{\mathsf{b}}\mathsf{a}$	$\emptyset$	$\emptyset$	$\{\varepsilon, 1\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\emptyset$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$

	$w[1..5]$ $\underline{\mathsf{a}}\mathsf{baba}$	$w[2..5]$ $\underline{\mathsf{b}}\mathsf{aba}$	$w[3..5]$ $\underline{\mathsf{a}}\mathsf{ba}$	$w[4..5]$ $\underline{\mathsf{b}}\mathsf{a}$	$w[5]$ $\underline{\mathsf{a}}$
ε	$\emptyset$	$\{(\mathsf{baba},\ 1)\}$	$\{(\mathsf{aba}, 1)\}$	$\{(\mathsf{ba},\ 1)\}$	$\{(\mathsf{a},\ 1)\}$
$u[3] = \mathsf{a}_$	$\emptyset$	$\{(\mathsf{bba},\ 1),$ $(\mathsf{bab},\ 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\mathsf{ba},\ 1),$ $(\mathsf{ab},\ 1)\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\{(\mathsf{b},\ 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\varepsilon, 1)\}$
$u[2..3] = \mathsf{b}\underline{\mathsf{a}}$	$\emptyset$	$\{(\mathsf{ba},\ 2),$ $(\mathsf{ab},\ 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\mathsf{a},\ 1)\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\{(\varepsilon, 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$
$u[1..3] = \mathsf{a}\underline{\mathsf{b}}\mathsf{a}$	$\emptyset$	$\{(\mathsf{b}, 1)\}$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\{(\varepsilon, 1)\}$	$\overset{\mathsf{a}\cdot}{\leftarrow}$ $\emptyset$	$\overset{\mathsf{b}\cdot}{\leftarrow}$ $\emptyset$

	$w[1..5]$ a̲baba		$w[2..5]$ b̲aba		$w[3..5]$ a̲ba		$w[4..5]$ b̲a	$w[5]$ a̲
ε	{(ababa，1)}		{(baba，1)}		{(aba，1)}		{(ba，1)}	{(a，1)}
$u[3]=$a_	{(baba，1)，(abba，1)，(abab，1)}	↙ a· ←	{(bba,1)，(bab，1)}	b· ←	{(ba，1)，(ab,1)}	↙ a· ←	{(b，1)}	b· ← {(ε,1)}
$u[2..3]=$ba̲	{(aba，2)，(aab,1)}	a· ←	{(ba，2)，(ab，1)}	↙ b· ←	{(a，1)}	a· ←	{(ε,1)}	↙ b· ← ∅
$u[1..3]=$ab̲a	{(ba，2)，(ab，2)}	↙ a· ←	{(b，1)}	b· ←	{(ε,1)}	↙ a· ←	∅	b· ← ∅

现在，补散布因子集为 C(ababa，aba) = {ab，ba}，其中两个补散布因子的重数均为二。

我们考虑两个相关问题：对于给定的 w 和 u，计算 $C(w,\ u)$：

问题 1. 给定 $w \in \Sigma^n$ 和 $S \subseteq \mathsf{ScatFact}_m(w)$，其中某个 $n,\ m \in \mathbb{N}$ 判定是否存在 $u \in \mathsf{ScatFact}_{n-m}(w)$，使得 $C(w,\ u) = S$.**P**

问题 2. 给定 $u \in \Sigma^*$ 和 $S \subseteq \Sigma^\ell$，其中 $\ell \in \mathbb{N}$ 为某个固定值，判断是否存在 $w \in \Sigma^{|u|+\ell}$，使得 $C(w,\ u) = S$。

现在假设给定一个词 $w \in \Sigma^n$，以及 w的散布因子集合 S，这些因子的长度均为 $m \in \mathbb{N}$，且它们是在移除某个未知词 u 后得到的（问题 1）。我们证明，可以在 $O(n^2\binom{n}{m})$时间内确定 u。注意，根据补散列因子的定义，所有 $s \in S$ 的长度都必须相同。我们的算法基于以下引理。

引理 13. 设 $n \in \mathbb{N}$、$w \in \Sigma^n$和 $S \subseteq \Sigma^m$，其中 $m \leq n$。对于所有 $u \in \Sigma^{n-m}$，都有 $S \subseteq C(w,u)$ 当且仅当 $u \in \bigcap_{v\in S} C(w,v)$。

证明。注意，根据补集的定义，如果 $v \in C(w,u)$，那么$u \in C(w,v)$。我们将分别证明两个方向。因此，设$S \subseteq C(w,u)$。根据定义，这意味着对于所有 $v \in S$，都有 $v \in C(w,u)$。根据前面所述的观察，对于所有 $v \in S$，都有 $u \in C(w,v)$，这意味着 $u \in \bigcap_{v\in S} C(w,v)$。

对于第二个方向，设 $u \in \bigcap_{v\in S} C(w,u)$，这意味着对于所有$v \in S$，都有 $u \in C(w,v)$。根据与前面相同的论证，我们得到，对于所有 $v \in S$，都有$v \in C(w,u)$，也就是说，$S \subseteq C(w,u)$。□

根据引理 13，我们可以从 $\bigcap_{v\in S} C(w,\ v)$ 中确定 u 的值。我们分两步进行。首先，确定 u的潜在取值候选集 $\mathcal{U} = \bigcap_{v\in S} C(w,\ v)$。接着，检查 $u \in \mathcal{U}$ 中的每个词，确定$C(w,\ u) = S$。如果找到任何满足 $C(w,\ u) = S$的 u，算法就会停止，并返回这个词作为解。否则，算法会继续执行，直到集合中的元素全部检查完毕。由此得到以下定理。

定理 14. 给定集合 $S \subseteq \Sigma^m$ 和 $w \in \Sigma^n$，我们可以在 $O(n^2\binom{n}{m})$ 时间和 $O(n\binom{n}{m})$ 空间内，确定是否存在一个词 u，使得 $C(w,u) = S$。

证明。注意，对于每个词 $v \in S$，我们可以根据命题 11，在 $O(n \cdot m\binom{n}{m})$ 时间和 $O(m\binom{n}{m})$ 空间内确定集合 $C(w,v)$。因此，确定集合 $\mathcal{U}$ 需要 $O(|S|n \cdot m\binom{n}{m})$ 时间，且无需增加空间。同样，由于我们可以在 $O(n \cdot (n-m)\binom{n}{n-m}) = O(n \cdot (n-m)\binom{n}{m})$ 时间和 $O((n-m)\binom{n}{n-m})$ 空间内确定 $C(w,u)$ 的值，并且可以在至多 $O(|S|m\log|S|)$ 时间内检查 $C(w,u)$ 是否 $= S$，这相当于先对每个集合排序，再检查 S 中的每个词是否与集合 $C(w,u)$ 中的词匹配。由此得到总时间复杂性为 $O(n \cdot (n-m)\binom{n}{m} + n \cdot m)\binom{n}{m}) = O(n^2\binom{n}{m})$，空间复杂性为 $O((n-m)\binom{n}{m} + m\binom{n}{m}) = O(n\binom{n}{m})$。□

现在我们转向问题 2：对于给定的词 u 和由长度为 ℓ 的词组成的集合 S，判断是否存在某个 w 使得 $C(w,u) = S$。我们关注一个更一般的问题：给定一个元组集合 $Z = \{(v_1,u_1),(v_2,u_2)\ldots,(v_r,u_r)\}$，其中 $|v_iu_i| = |v_ju_j|$，$\forall i, j \in [r]$，$r \in \mathbb{N}$，是否存在一个长度为 $|v_iu_i|$ 的词 w，使得对于每个元组 (v_i,u_i)，v_i 和 u_i 到 w 中都存在一对不相交嵌入。我们将寻找这样一个词的问题称为两两不相交嵌入问题。注意，当输入元组集合为 $\{(v_1,u),(v_2,u)\ldots,(v_r,u)\}$ 时，该问题就等价于原始问题。本节的主要结果是：对于输入集合 Z，其中包含 r 个长度为 n 的元组，即元组 $(v_i,u_i) \in Z, |v_iu_i| = n$，我们给出了一个运行时间为 $O(rn^r)$ 的算法来求解两两不相交嵌入问题。

我们通过动态规划方法解决两两不相交嵌入问题，构造一个大小为 $O(n^{|Z|+1})$ 的表 V（表示 Valid），并在每个单元格上用时 $O(|Z|)$ 填充该表。形式化地说，表 V 中的每个条目都由一个包含 $|Z|$ 个元组 $\{(z_1,y_1)、(z_2,y_2)\ldots\ldots、(z_r,y_r)\}$ 的集合进行索引，其中 z_i 是 v_i 的后缀，而 y_i 是 u_i 的后缀。我们进一步假设，对于每个这样的元组 $|z_iy_i| = |z_jy_j|$，对所有 i、$j \in [r]$ 均成立。注意，这一限制不适用于集合之间，因此我们可能有一对这样的集合 $(z_1,y_1)、(z_2,y_2)\ldots\ldots、(z_r,y_r)$ 和 $(z'_1,y'_1)、(z'_2,y'_2)\ldots\ldots、(z'_r,y'_r)$，其中 $|z_iy_i| \neq |z'_iy'_i|$，同时每个集合仍然都有一个条目。注意，这至多给出了 $\ell^{|Z|}$ 个集合，其中 $|z_iy_i| = \ell$，因此，令 $n = |v_iu_i|$ 表示输入集合 Z 中的某个元组 (v_i,u_i)，我们在 V 中有 $O(n^{|Z|+1})$ 个条目。我们将 $V[\{(z_1,y_1)、(z_2,y_2)\ldots\ldots、(z_r,y_r)\}]$ 定义为一个布尔值：当且仅当对于集合 $\{(z_1,y_1)、(z_2,y_2)\ldots\ldots、(z_r,y_r)\}$，两两不相交嵌入问题存在一个有效解时，该值为 true，否则为 false。作为基例，我们有 $V[\{(\varepsilon,\varepsilon)、(\varepsilon,\varepsilon)\ldots\ldots、(\varepsilon,\varepsilon)\}]$ 为 true。在给出用于定义我们的动态规划算法的递推关系之前，我们先给出一个关键观察。

引理 15. 给定一个元组集合 $Z = \{(z_1,y_1),(z_2,y_2)\ldots,(z_r,y_r)\}$，其中 $|z_iy_i| = |z_jy_j|$，$\forall i, j \in [r]$，$r \in \mathbb{N}$，存在一个长度为 $|z_iy_i|$ 的词 w，使得每个元组 $(z_i,y_i) \in Z$ 都可以不相交地嵌入，当且仅当对于所有 $i \in [r]$，我们有 $z_i[1] = w[1]$ 或 $y_i[1] = w[1]$。

证明。注意，由于 $|w| = |z_iy_i|$，w 中的每个符号都必须属于 z_i 或 y_i的嵌入。因此，$w[1]$ 必须是 z_i 或 y_i的第一个符号。□

根据引理 15，令 $X(\{(z_1, y_1) \ldots, (z_r, y_r)\}) = \bigcap_{i\in[r]}\{z_i[1], u_i[1]\}$。假设 $X(\{(z_1, y_1), (z_2, y_2) \ldots, (z_r, y_r)\})$ 非空且包含字母 x_1，我们定义集合 $T_1 = (z'_1, y'_1), (z'_2, y'_2) \ldots, (z'_r, y'_r)$，其中 (z'_i, y'_i) 或为 $(z_i[2, |z_i|], y_i)$（若 $z_i[1] = x_1$），或为 $(z_i, y_i[2, |y_i|])$。此外，若有 $X(\{(z_1, y_1), (z_2, y_2) \ldots, (z_r, y_r)\}) = \{x_1, x_2\}$，我们便构造集合 $T_2 = (\bar{z}'_1, \bar{y}'_1), (\bar{z}'_2, \bar{y}'_2) \ldots, (\bar{z}'_r, \bar{y}'_r)$，其中 $(\bar{z}'_i, \bar{y}'_i)$ 或为 $(z_i[2, |z_i|], y_i)$（若 $z_i[1] = x_2$），或为 $(z_i, y_i[2, |y_i|])$。注意，T_1 与 T_2 的差异在于 z_i 的第一个字母，而 $X(\{(z_1, y_1), (z_2, y_2) \ldots, (z_r, y_r)\})$ 的基数至多为二，这是由构造方式决定的。采用上述记号，我们现在定义递推关系，用 Z 表示集合 $\{(z_1, u_1), (z_2, u_2) \ldots, (z_r, u_r)\}$：

$$V[Z] = \begin{cases} \text{True}, & \text{if } Z = \{(\varepsilon, \varepsilon)\}, (\varepsilon, \varepsilon), \ldots, (\varepsilon, \varepsilon), \\ \text{False}, & \text{if } X(Z) = \emptyset, \\ V[T_1] & \text{if } X(Z) = \{x\}, \\ V[T_1] \wedge V[T_2] & \text{if } X(Z) = \{x_1, x_2\}. \end{cases} \tag{1}$$

引理 16. 方程（1）正确定义了表 V。

证明。我们分别考虑方程（1）中的每种情况。首先，如果有 $Z = \{(\varepsilon, \varepsilon), (\varepsilon, \varepsilon) \ldots, (\varepsilon, \varepsilon)\}$，那么每个元组都存在一个嵌入到空单词中的不相交嵌入，因此 $V[Z]$ 为真。其次，根据观察 15，如果集合 $X(Z)$ 中没有符号，那么就不可能存在一个词，将 Z 中的每个元组都作为不相交嵌入包含其中。

第三，如果存在这样一个共享符号，那么任何将 Z 中的每个元组都作为不相交嵌入包含其中的词 w，都必须以符号 x 开头。此外，w 必须包含 T_1 中每个元组的两两不相交嵌入。反过来，如果存在一个词 w，其中对于 T_1 中的每个元组，都存在 v_i，u_i 的一对不相交嵌入，那么词 xw 就允许对 Z 中的每个元组进行两两不相交嵌入。最后，如果 X、x_1 和 x_2 中有两个符号，那么根据与上述相同的论证，存在一个词允许对 Z 中的每个元组进行一对不相交嵌入，当且仅当存在某个以 x_1 开头的词，允许对 T_1 中的每个元组进行一对不相交嵌入；或者存在某个以 x_2 开头的词，允许对 T_2 中的每个元组进行一对不相交嵌入。□

暂且搁置运行时间分析，我们得到关于第三个主要问题的以下结果：对于给定的 $S \subseteq \Sigma^\ell$ 和 $u \in \Sigma^m$，判断是否存在（并确定）$w \in \Sigma^{\ell+m}$。

推论 17. 存在 $w \in \Sigma^*$，使得 $C(w, u) = \{(v_1, v_2, \ldots, v_r)\}$ 当且仅当 $V[\{(v_1, u), (v_2, u) \ldots, (v_r, u)\}]$ 为真。

在证明我们构造的运行时间之前，我们还需要以下内容。

引理18. 集合 $Z = \{(v_1, u_1), (v_2, u_2) \ldots, (v_r, u_r)\}$ 的表 V 可在 $O(rn^{r+1})$ 时间内计算，其中 $|Z| = r$ 且 $|v_1 u_1| = n$。

证明. 首先注意， $V[Z]$的值（其中每个元组 (z_i, y_i) 满足$|z_i y_i| = \ell$ ）可以在 $O(|Z|)$ 时间内计算，前提是已为每个 $Z' = \{(z'_1, y'_1), (z'_2, y'_2) \ldots, (z'_r, y'_r)\}$ 计算出 $V[Z']$ 的值，其中 $|z'_i y'_i| = \ell - 1$。因此，先按元组中词的总长度递增来确定 $V[Z]$ 的值，就可以在 $O(|Z| \cdot |V|)$ 时间内计算该表，其中$|V|$ 是 V中的条目总数。由于此类条目的数量不超过 $n^{|Z|+1}$，因此可得所述运行时间 $O(|Z|n^{|Z|})$。□

引理 19。 给定一个元组集合 $Z = \{(z_1, y_1)、(z_2, y_2).....(z_r, y_r)\}$，使得$V[Z]$ 为真且 $|z_i y_i| = |z_j y_j|$，对于所有 $i, j \in [r]$，我们可以重构出某个长度为 $n = |z_i y_i|$ 的词 w，使得集合 Z 中每个元组在 $O(|Z|n)$时间内都存在一对不相交嵌入。

证明。 我们以类似于算法 1 的方式完成此操作。具体而言，从 $V[Z]$ 开始，我们选择 $x \in X$（Z）作为第一个符号，使得从每个元组中某个词的开头移除符号 x 后形成的集合 T 满足 $V[T]$ 为真。我们重复这一过程，直到每个元组都由两个空单词组成。正确性由引理 16 和推论 17 得证。□

这引出了本节的最后一个定理，我们在其中给出了算法，用于根据其余两个信息确定 u、 w 和 C（w、 u）中的任意一个。

定理 20. 给定一个由 $V = \{v_1, v_2, \ldots, v_r\} \subseteq \Sigma^\ell$ 个词组成的集合，其中 n、 r、 $\ell \in \mathbb{N}$，以及一个词 $u \in \Sigma^{n-\ell}$，我们可以在 $C(w, u) = V$ 时间内确定是否存在某个词 $w \in \Sigma^n$，使得 $C(w, u) = V$ 属于 $O(rn^{r+1})$；此外，若存在这样的词，我们可以在 $O(rn^{r+1})$ 时间内构造它。

证明。 由引理 16 和 19 以及推论 17 可得。 □

推论 21. 给定一个由元组 $Z = \{(z_1, y_1)、(z_2, y_2) \ldots, (z_r, y_r)\}$ 组成的集合，使得 $|z_i y_i| = |z_j y_j|$，对于所有 i、 $j \in [r]$，我们可以在 $O(rn^{r+1})$ 时间内确定是否存在一个长度为 n 的词 w，使每个元组都能在 Z 中不相交嵌入。此外，若存在这样的词，我们可以在 $O(rn^{r+1})$ 时间内确定它。

4 补散因子

本节将介绍关于补散布因子集的组合学见解。由于我们的算法运行时间取决于 $\binom{w}{u}$，也就是说， u 在 w 中的嵌入越多，算法所需时间就越长，因此我们关注与嵌入数量无关的 $C(w, u)$相关见解。我们先列出一系列观察结果。首先，我们观察到，在包含字母平方的词中，存在一个散布因子，它只有一个对应的补散布因子，却有多个嵌入。

观察 22 设 $w \in \Sigma^*$ 是一个包含字母平方 aa （其中a $\in \Sigma$）的词，即存在 x、 $y \in \Sigma^*$，使得 $w = x\mathtt{aa}y$。若 $u \in \mathrm{ScatFact}(w)$ 是

形式为 $x_u\mathtt{a}y_u$，其中 $x_u \in \mathrm{ScatFact}(x)$ 和 $y_u \in \mathrm{ScatFact}(y)$，则有 $|C(w,u)| \lhd |E(w,u)|$。原因是，u 在 w 中的嵌入 e_1、e_2，连同 $e_1(|x_u|+1) = |x|+1$ 和 $e_2(|x_u|+1) = |x|+2$，会产生相同的补散布因子。

此外，我们想强调一个简单的事实：具有唯一嵌入意味着补集是单元素补集，但反过来并不成立。

观察 23 如果 $u \in \mathrm{ScatFact}(w)$ 对于某个 $w \in \Sigma^*$ 恰好有一个嵌入，那么我们有 $|C(w,u)| = 1$。请注意，反方向通常并不成立：考虑 $w = \mathtt{aaa}$ 和 $u = \mathtt{a}$，则 $C(w,u) = \{\mathtt{aa}\}$，但 $|E(w,u)| = 3$。

接下来，我们利用给定词 $w \in \Sigma^*$ 的普适性指标 $\iota(w)$（从而也利用其拱因子分解），来考察长度小于 $\iota(w)$的给定散布因子的补散列因子数量。

引理 24. 对于 $w \in \Sigma^*, u \in \mathrm{ScatFact}(w)$，若 $|u| \le \iota(w)$，则有 $|E(w,u)| \ge \binom{\iota(w)}{|u|}$。

证明。根据普适性的定义，每个散布因子 u 都包含在一个$|u|$-普适词中。因此，每个 $|u|$-普适词 $\mathrm{ar}_{i_1}(w)\,\mathrm{ar}_{i_2}(w)\cdots\mathrm{ar}_{i_{|u|}}(w) \in \mathrm{ScatFact}(w)$ with $i_j \in [\iota(w)]$, $j \in [|u|]$ and $i_1 < i_2 < \cdots < i_{|u|}$ does contain $u$as a scattered factor. 通过组合论论证，可以形成这类 $|u|$-普适词的方式有 $\binom{\iota(w)}{|u|}$ 种，因此在 w中至少有 $\binom{\iota(w)}{|u|}$ 个不同的 u 嵌入。□

请注意，引理 24 的思想无法用于深入理解词 $|u| > \iota(w)$：这些嵌入可能属于同一个补散列因子。例如，$w = \mathtt{ababbaba}$ 中 $u = \mathtt{ab}$ 的补散列因子为 ababba（出现次数为 1）、abbaba（出现次数为 3）、abbbaa（出现次数为 1）、bababa（出现次数为 2）和 babbaa（出现次数为 1），即$|C(w,\ u)| = 5 \not\ge 6 = \binom{4}{2}$。

现在我们为 w 和 u 指定性质，使得 $|C(w,\ u)| \ge 2$。

引理 *25.* 给定 $w \in \Sigma^*$，其中 $\iota(w) = 2$，并令 $u \in \mathit{alph}(w)$，我们有$|C(w,u)| = 1$ 当且仅当 $u = \mathit{m}(w)[1]$，且对于某些 $x \in \Sigma^*$ 和 $r \in \mathbb{N}$，有 $\mathit{ar}_2(w) = (\mathit{m}(w)[1])^r x$，其中 $\mathit{m}(w)[1] \notin \mathit{alph}(x)$。

证明。首先，令 $|C(w,\ u)| = 1$。若 $u \ne \mathrm{m}(w)[1]$，那么分别取自第一拱且保持第二拱不变的 u 的嵌入，以及取自第二拱且保持第一拱不变的 u 的嵌入，会产生两个不同的补散布因子。反过来，由于 w 的形式为 $y(\mathrm{m}(w)[1])^r x$，其中存在某个 $r \in \mathbb{N}$，且 $\mathrm{m}(w)[1]$ 既不出现在 y 中，也不出现在x中，因此结论成立。□

引理 26. 设 $w \in \Sigma^*$ 满足 $\iota(w) \ge 2$。设 $u \in \mathrm{ScatFact}(w)$ 满足 $|u| < \iota(w)$。那么，若 $u[1] \ne \mathrm{m}(w)[1]$，则 $|C(w,u)| > 1$。

证明. 设 $u[1] \neq \mathsf{m}(w)[1]$，并假设 $C(w,u) = \{v\}$。由于 u 严格短于 w 的普适性指标，因此存在一个嵌入 $e_1 \in E(w,u)$，使得

$$e_1(1) \leq |\operatorname{ar}_1(w)| \quad \text{and} \quad |\operatorname{ar}_{i-1}(w)| + 1 \leq e_1(i) \leq |\operatorname{ar}_i(w)|$$

对所有 $i \in [|u|]$，也就是说，我们从前 $|u|$ 个弧中选取 u 的字母。令 e_{1i} 为 $\mathsf{ar}_i(w)$ 中在 w 内为 $e_1(i)$ 的那个弧的索引，因此它表示 $e(i)$ 相对于 $\mathsf{ar}_i(w)$ 的位置。对所有 $i \in [|u|]$，令 $x_i = \mathsf{ar}_i(w)[1..e_{1i}-1]\mathsf{ar}_i(w)[e_{1i}+1..|\mathsf{ar}_i(w)|]$，也就是说，$w$ 的第一个不含我们为 u 选取的字母的弧。用同样的论证，我们可以找到一个嵌入 e_2，它只使用 w 的最后 $|u|$ 个弧。对所有 $\iota(w) - |u| + 1 \leq i \leq \iota(w)$，定义 y_i，方式类似于 x_i。

若 $\iota(w) = 2$，则有 $|u| = 1$。由于 v 是唯一的补散布因子，我们知道 $x_1 \mathsf{ar}_2(w)$，$\mathsf{ar}_1(w)y_2$ 是 v 的前缀。因此有 $x_1[|x_1|] = \mathsf{m}(w)[1] \neq \mathsf{ar}_1(w)[|x_1|]$，因为第一个弧的倒数第二个字母不可能是众数字母——这就产生了矛盾。

如果 $\iota(w) > 2$，我们有 $|u| \geq 1$。由于 v 是唯一的补散列因子，我们知道不仅 x_1x_2，而且 $\mathsf{ar}_1(w)\mathsf{ar}_2(2)$ 也是 v的前缀。然而，根据 $u[1] \neq \mathsf{m}(w)[1]$，我们再次得到 $x_1[|x_1|] = \mathsf{m}(w)[1] \neq \mathsf{ar}_1(w)[|x_1|]$ ——这就产生了矛盾。□

请注意，引理 26 的反向结论并不成立（由逆否命题可知）：如果 $w = \mathsf{abba}$ 且 $u = \mathsf{b}$，我们有 $C(w, u) = \{\mathsf{aba}\}$。不过，我们可以将这一结果推广到普适性指标至少为三的词：如果 Σ中至少有三段拱和三个字母，我们就能给出关于 w 和 u 的一个条件，使得 w中至少有两个补散列因子 u。

定理 27. 设 $|\Sigma| \geq 3$。如果 $w \in \Sigma^*$ 满足 $\iota(w) > 2$，且 $u \in \operatorname{ScatFact}(w)$ 满足 $|u| < \iota(w)$，$u \neq \varepsilon$ 那么 $|C(w,u)| > 1$。

证明。首先，注意到我们有 $\iota(w) > 1$，否则 $|u|$ 将为 0，因此$u = \varepsilon$。令 $u \in \mathsf{ScatFact}(w)$，其中 $|u| < \iota(w)$。若 $u[1] \neq \mathsf{m}(w)[1]$，则由引理 26 可得结论。因此，假设 $u[1] = \mathsf{m}(w)[1]$。

假设 $C(w, u) = \{v\}$。由于 u 严格短于普适性指标，因此存在一个嵌入 $e_1 \in E(w, u)$，使得

$$e_1(1) \leq |\operatorname{ar}_1(w)| \quad \text{and} \quad |\operatorname{ar}_{i-1}(w)| + 1 \leq e_1(i) \leq |\operatorname{ar}_i(w)|$$

对于所有 $i \in [|u|]$，也就是说，我们从前 $|u|$ 个弧中选取 u 的字母。令 e_{1i} 为 $\mathsf{ar}_i(w)$ 的索引，该索引在 w 中为 $e_1(i)$，因此也是 $e(i)$ 相对于 $\mathsf{ar}_i(w)$ 的位置。对所有 $i \in [|u|]$，设 $\mathsf{ar}_i(w)[1..e_{1i}-1]\mathsf{ar}_i(w)[e_{1i}+1..|\mathsf{ar}_i(w)|]$，也就是说，$w$ 的 i^{th} 弧去掉我们为 u 选取的字母。用同样的论证，我们可以找到一个嵌入 e_2，它只使用 w 的最后 $|u|$ 个弧。对所有满足 $\iota(w) - |u| + 1 \leq i \leq \iota(w)$ 的情况，定义 y_i，其方式与 x_i 类似。

由于 $\iota(w) > 2$ 且 v 是唯一的补散布因子，我们知道不仅 x_1x_2，而且 $\mathsf{ar}_1(w)\,\mathsf{ar}_2(2)$ 也是 v 的前缀。注意，对于所有 $i \in [|u|]$，都有 $|x_i| + 1 = |\mathsf{ar}_w(i)|$。由 $u[1] = \mathsf{m}(w)[1],$，我们有 $x_1 = \mathsf{in}_1(w)$。

由于 $\alpha_1\alpha_2$ 是 $\mathsf{ar}_1(w)\mathsf{ar}_2(w)$ 的前缀，我们得到 $x_2[1] = \mathsf{ar}_2(w)[1]$。由于 x_2 是从 $\mathsf{ar}_2(w)$ 中删除 $\mathsf{ar}_2(w)$ 中恰好一个字母后得到的，因此有以下几种情况：

情况 1：$e_1(2) = |\mathsf{ar}_1(w)| + 1$（我们为 u 选取第二个拱的第一个字母）于是我们有 $\mathsf{ar}_2(w)[2..|\mathsf{ar}_2(w)|-1] = \mathsf{ar}_2(w)[1..|\mathsf{ar}_2(w)|-2]$。因此，存在 $\mathsf{a}、\mathsf{b}、\mathsf{c} \in \Sigma$ 和 $y \in \Sigma^*$，使得 $\mathsf{ar}_2(w) = \mathsf{a}y\mathsf{bc}$，从而 $y\mathsf{b} = \mathsf{a}y$。这意味着对于某个 $r \in \mathbb{N}_0$，$\mathsf{a} = \mathsf{b}$ 且 $y = \mathsf{a}^r$，即 $\mathsf{ar}_2(w) = \mathsf{a}^{r+2}\mathsf{c}$。因此产生矛盾，因为我们有 $|\Sigma| \geq 3$。
情况 2：$|\mathsf{ar}_1(w)| + 1 < e_1(2) < |\mathsf{ar}_1(w)\,\mathsf{ar}_2(w)|$ 因此，我们为 u 选取第二个拱中较靠后的一个字母，但不选取众数字母。这意味着 $\mathsf{m}(w)[2]$ 是 x_2 的最后一个字母，并且 $\mathsf{m}(w)[2] \in$ 位于$_2(2)$中；根据 $|\Sigma| \geq 3$，这意味着 $|\mathsf{ar}_2(w)| \geq 3$。**情况 3**：$e_1(2) = \mathsf{m}(w)[2]$

在这种情况下，我们再次得到 $|$字母表$(\mathsf{ar}_2(w))| = 2$ ——这与结论矛盾。 □

注意，定理 27 中的条件 $|u| < \iota(w)$ 是绝对必要的：任何具有空余部且长度为 $|u| = \iota(w)$ 的词 $w \in \Sigma^*$，其 $u = \mathsf{m}(w)$ 按定义都只有一个唯一嵌入，这便是明证。因此，我们有 $|C(w, \mathsf{m}(w))| = 1$，若没有上述条件，这与定理 27 的结论相矛盾。此外，定理 27 得出长度为 $\iota(w)$ 的唯一散布因子 -1，满足 $|C(w, u)| = 1$。

推论 *28*。设 $w \in \Sigma^*$，对于所有 $i \in [\iota(w) - 1]$，均有 $m(w)[i] = ar^{i+1}(w)[1]$，且 $r(w) = \varepsilon$。散布因子 $u = m(w)[1..|m(w)|-1]$ 是唯一一个满足 $|C(w,u)| = 1$ 的长度为 $\iota(w)$ 的散布因子。

下面的结果使用了在研究最短缺失散布因子（SAS）的背景下于 [26] 中引入的 α-β-因子分解 [47]。这里我们使用给定词的 SAS 定义：若 $v \notin$ScatFact(w) 且 ScatFact$(v) \setminus \{v\} \subseteq$ ScatFact(w)，则词 v 是 w 的一个 SAS。

命题 29。设 $w \in \Sigma^*$。若 $u \in \mathrm{SAS}(w)$，$|\mathrm{ar}_i(w)|_{u[i]} = 1$ 对所有 $i \in [|u|-1]$ 以及 $u[|u|] \notin$ 字母表$(\mathsf{r}(w))$ 均成立，则 $u[1..|u|-1]$ 是 w 的一个散布因子，且满足 $C(w, u[1..|u|-1]) = 1$。

证明。令 $w = \alpha_1\beta_1 \cdots \alpha_{\iota(w)}\beta_{\iota(w)}\alpha_{\iota(w)+1}$ 为 w 的 α-β 因子分解。由命题 6] 可知，对于所有 $i \in [|u|-1]$ 和 $u[|u|] \notin \mathsf{alph}(\mathsf{r}(w))$，$u[1] \in \mathsf{alph}(\beta_1)\backslash\mathsf{alph}(\alpha_1)$ $u[i] \in \mathsf{alph}(\beta_i)$ $u[i]u[i+1] \notin \mathsf{ScatFact}_2(\beta_i\alpha_{i+1})$。根据 SAS 的定义，我们首先得到 $u[1..|u|-1] \in \mathsf{ScatFact}(w)$。现在假设 $u[1..|u|-1]$ 有两个补散布因子，即 $|C(w, u[1..|u|-1])| > 1$。因此，$u[1..|u|-1]$ 在 w 中必定有两种嵌入。由于 $|u[1..|u|-1]| = \iota(w)$ 且 $|\mathsf{ar}_i(w)|_{u[i]} = 1$ 对所有 $i \in [|u|-1]$ 成立，这意味着要么某个嵌入中的两个字母位于同一条弧内，要么最后一个字母 $u[i]$ 也出现在剩余部分中。由于 $u[|u|] \notin \mathsf{alph}(\mathsf{r}(w))$，后一种情况不可能发生。因此，对于某个 $i \in [\iota(w)]$，有 $u[i]u[i+1] \in \mathsf{ar}_i(w)$。根据命题 6]，我们有 $u[i] \in \mathsf{alph}(\beta_i)$。由于 $|\mathsf{ar}_i(w)|_{u[i]} = 1$，$\mathsf{ar}_i(w)$ 中没有 $u[i]$ 的其他出现次数。因此，$u[i]u[i+1] \in \mathsf{ar}_i(w)$ 意味着 $u[i]u[i+1] \in \mathsf{ScatFact}_2(\beta_i)$，这与关于缺失散布因子的条件 $u[i]u[i+1] \notin \mathsf{ScatFact}_2(\beta_i\alpha_{i+1})$ 矛盾。因此，我们可以得出结论：$C(w, u[1..|u|-1]) = 1$。□

下面的观察描述了这样的情况： w 包含重复 y^k，而 u 包含其基 y。考虑 w = bananaananas u = ana。这里，由平方 na 导致的两种嵌入具有相同的补散布因子 bnanaanas，其索引分别为 (2，8，9) 和 (2，10，11)。

观察30 设 $w \in \Sigma^*$ 和 $u \in$ ScatFact(w)。如果 $w = xy^kz$ 且 $u = x'yz'$对于 $x,\ z \in \Sigma^*$, $y \in \Sigma^+,\ k \in \mathbb{N}$满足 alph($x'$) $\subseteq$alph(x) 以及 alph(z') $\subseteq$alph(z)那么 $|E_=(w,\ u)| \geq k$。

我们继续研究 $|C(w,\ u)| = 1$ 和 $|E(w,\ u)| = 1$，并证明：对于任何不包含平方的词，具有两个不同嵌入的散布因子的补散布因子集的大小总是至少为 2。

引理 31。 如果 $w \in \Sigma^*$ 是无平方的，那么对于每个 $u \in \mathrm{ScatFact}(w)$，满足$|E(w,\ u)| \geq 2$，都有 $|C(w,\ u)| > 1$。

证明。设 w 为无平方的，且 $u \in$ ScatFact(w) 满足 $|E(w,\ u)| \geq 2$。现在假设 $|C(w,\ u)| = 1$。由于存在两个嵌入，因此存在$v_1,\ .....,\ v_{|u|+1},\ v'_1,\ .....,\ v'_{|u|+1} \in \Sigma^*$ 使得 $w = v_1u[1]v_2u[2]v_3 \cdots v_{|u|}u[|u|]v_{|u|+1} = v'_1u[1]v'_2u[2]v'_3 \cdots v'_{|u|}u[|u|]v'_{|u|+1}$且 $v_1v_2 \cdots v_{|u|+1} = v'_1v'_2 \cdots v'_{|u|+1}$。由于这些嵌入彼此不同，存在一个最左侧的索引 $j \in [|u|+1]$，使得$v_j \neq v'_j$。不妨设 $|v_j| < |v'_j|$。因此，存在 $z \in \Sigma^*$ 使得 $v_ju[j]z = v'_j$。

由于 $v_1v_2 \cdots v_{|u|+1} = v'_1v'_2 \cdots v'_{|u|+1}$这意味着 $v_{j+1}[1] = u[j]$但这样一来 w就包含平方 $u[j]u[j]$ ——这与 w的前置条件矛盾。□

命题 32. 对于 $w \in \Sigma^*$， w 不包含任何字母平方，当且仅当每个 $u \in \mathrm{ScatFact}(w)$ 且满足 $C(w, u) = 1$ 的项都只有一个嵌入。

证明。 首先，我们证明从左到右的蕴含关系。设 w中不存在字母平方。反设存在一个 $u \in$ ScatFact(w)，满足 $|C(w, u)| = 1$，且它至少有两个嵌入。

如果这些嵌入不属于任何平方，我们可以应用引理 31，由此得出 $|C(w, u)| > 1$，这与 $|C(w, u)|$的前置条件矛盾。

因此，假设这两个嵌入属于一个平方。因此存在w'、 $w'' \in \Sigma^*$、 $v \in \Sigma^+$ 使得 $w = w'vvw''$。此外，存在 $j_1,\ \ldots,\ j_\ell \in [|v|]$递增，且 $w' \in$ ScatFact(w') $w'' \in$ ScatFact(w'')，使得

$$\overline{w'} \cdot v[1..j_1 - 1]v[j_1 + 1..j_2 - 1] \cdots v[j_{\ell-1} + 1..j_\ell - 1]v[j_{\ell+1}..|v|] \cdot v \cdot \overline{w''},$$
$$\overline{w'} \cdot v \cdot v[1..j_1 - 1]v[j_1 + 1..j_2 - 1] \cdots v[j_{\ell-1} + 1..j_\ell - 1]v[j_{\ell+1}..|v|] \cdot \overline{w''} \in C(w, u).$$

由于 $C(w,\ u) = 1$两个词相等，即 $v[1..j_1-1]v[j_1+1..j_2-1] \cdots v[j_{\ell-1}+1..j_\ell-1]v[j_{\ell+1}..|v|] \cdot v = v \cdot v[1..j_1-1]v[j_1+1..j_2-1] \cdots v[j_{\ell-1}+1..j_\ell-1]v[j_{\ell+1}..|v|]$。我们得到 $v[j_1] = v[j_1 + 1]$，这意味着 $v[j_1]v[j_1 + 1] = v[j_1 + 1]v[j_1 + 1]$和 w 包含一个字母平方，这与矛盾相矛盾。

现在我们证明从左到右的蕴含关系。因此，假设每个$u \in \mathsf{ScatFact}(w)$（其中 $C(w, u) = 1$）都只有一个嵌入。为导出矛盾，假设 w 中存在一个字母平方，即我们可以将$w = w'\mathsf{a}\mathsf{a}w''$改写为 w'，$w'' \in \Sigma^*$，$\mathsf{a} \in \Sigma$。散布因子 $w'\mathsf{a}w'' \in \mathsf{ScatFact}(w)$ 以 a 为补散布因子，因此 $C(w, w'\mathsf{a}w'') = \{\mathsf{a}\}$ 有两个嵌入，因为可以任选两个 a 中的一个。这与以下事实矛盾：任何补集为单元素补集的散布因子都只有一个嵌入。□

既然我们现在已经完全刻画了对于无平方因子词 $C(w, u) = 1$ 蕴含 $E(w, u) = 1$ 以及 $u \in \mathsf{ScatFact}(w)$ 的情形，接下来我们将继续考察包含字母平方的 w。由于包含某个字母平方中任一部分所用字母的散布因子 u，会立即至少有两种嵌入，因此我们得到以下结果。在下面的命题中，我们考察 w 的一个字母平方；当我们说 u 使用它的字母时，指的就是该平方的这次出现。

命题 33. 设 $w \in \Sigma^*$ 包含字母平方。对于每个 $u \in \mathrm{ScatFact}(w)$，若 $|C(w, u)| = 1$ 仅使用 w 中某些字母平方里的一个字母，则有 $|E(w, u)| > 1$。反过来，对于每个 $u' \in \mathrm{ScatFact}(w)$，若 $|C(w, u')| = 1$不使用或使用 w 中相应字母平方的全部字母，则有 $|E(w, u')| = 1$。

证明。首先，令 $u \in \mathsf{ScatFact}(w)$，其中 $|C(w, u)| = 1$，且它在 w 中只使用某些字母平方中的一个字母。我们有 $|E(w, u)| > 1$。由此立即可知，u 有两种嵌入：一种使用该平方的第一个字母，另一种使用第二个字母。

其次，令 $u' \in \mathsf{ScatFact}(w)$，其中 $|C(w, u')| = 1$，且它在 w 中不使用或使用字母平方中的所有字母。先考虑不使用任何字母平方的情形。因此，u' 在 w 中的每个嵌入也都出现在 $\mathsf{cond}(w)$ 中，其中 $\mathsf{cond}(w)$ 是 $w = x_1^{k_1} x_2^{k_2} \cdots x_\ell^{k_\ell} \in \Sigma^*$ 的压缩形式，且 $k_i, \ell \in \mathbb{N}$；$i \in [\ell]$ 在假设 $x_j \neq x_{j+1}$ 对于 $j \in [\ell - 1]$ 成立时由 $\mathsf{cond}(w)$ 定义。现在，对于 $\mathsf{cond}(w)$，命题 32 适用，因为 $|C(w, u')| = 1$ 意味着 $|C(\mathsf{cond}(w)\ u')| = 1$，因此我们可以得出结论：$u'$ 在 $\mathsf{cond}(w)$ 中恰有一种嵌入，因而在 w 中也恰有一种嵌入。

现在假设 u' 使用了每个字母平方中的所有字母。假设$|E(w, u')| > 1$。因此，必定存在 w 中 u' 的两个嵌入。考虑 $\mathsf{cond}(w)$ 中 $\mathsf{cond}(u')$ 的嵌入。根据 u' 的构造，我们可以得出，$\mathsf{cond}(w)$ 中必定有 $\mathsf{cond}(u')$ 的两个嵌入。由于 $\mathsf{cond}(w)$ 不含字母平方，我们可以应用命题 32，得出 $\mathsf{cond}(w)$ 中 $\mathsf{cond}(u')$ 只有一个嵌入。由于现在 u' 完全穷尽了 w 的每个字母平方，我们也得到 u' 只有一个嵌入 w，这就产生了矛盾。□

注记 *34.* 注意，命题 33 中不考虑长度至少为二的平方，因为在这一特定情形下，对于嵌入的像中不含该平方索引的散布因子，$|C(w, u)| = 1$ 的前置条件不成立。理由如下：设 $w = xy^2z$，其中 $x, y, z \in \Sigma^*$，$|$属于字母表 $\mathsf{alph}(y)$，且 y 是本原的。现在令 $u \in \mathsf{ScatFact}(w)$ 具有如下形式：$w = x'y'z'$，其中 $x' \in \mathsf{ScatFact}(x)$ $y' \in \mathsf{ScatFact}(y)$ $z' \in \mathsf{ScatFact}(z)$。令

$y[i]$ 令 $i \in [|y|]$ 为 $y'[1]$ 在 y 中最左边的一次出现，并且 $x'' \in C(x, x')$ $y'' \in C(y, y')$ $z'' \in C(z, z')$，我们得到 $x'yy''z', x'y[1..i-1]y[i+1..|y|]y''[1..i-1]y[i]y''[i+1..|y''|]z' \in C(w, u)$。为导出矛盾，假设 $|C(w, u)| = 1$。因此，$x'yy''z' = x'y[1..i-1]y[i+1..|y|]y''[1..i-1]y[i]y''[i+1..|y''|]z'$，这意味着 $yy'' = y[1..i-1]y[i+1..|y|]y''[1..i-1]y[i]y''[i+1..|y''|]$。此外，我们可以将左侧改写为 $yy'' = y[1..i-1]y[i]y[i+1..|y|]y''[1..i-1]y''[i..|y''|]$，从而得到

$$\begin{aligned}&(y[1..i-1])(y[i]y[i+1..|y|]y''[1..i-1])(y''[i..|y''|])\\ =\ &(y[1..i-1])(y[i+1..|y|]y''[1..i-1]y[i])(y''[i+1..|y''|])\end{aligned}$$

这化简为 $y[i]y[i+1..|y|]y''[1..i-1] = y[i+1..|y|]y''[1..i-1]y[i]$. 根据著名的 Lyndon-Schützenberger 定理，我们得到 $y[i+1..|y|]y''[1..i-1] \in y[i]^*$，而且由于 $y''[1..i-1] = y[1..i-1]$ （根据索引 i 的选取），我们得出 $y \in y[i]^*$——这与 $|\mathsf{alph}(y)| \geq 2$ 矛盾。

我们从补散布因子集的角度，完成对给定 $u \in \Sigma^*$ 的自洗牌 $u \sqcup u$ 的研究，即对于某个 $w \in \Sigma$，有 $u \in C(w, u)$。首先注意到，$u \in C(w, u)$ 并不意味着 $|C(w, u)| = 1$：考虑 $w = \mathsf{abaabaaa} \in \mathsf{abaa} \sqcup \mathsf{abaa}$，这里嵌入 $(1, 5, 6, 7) \in E(w, u)$ 得到 $\mathsf{baaa} \in C(w, u)$。

现在我们继续讨论自交错中所包含的词的刻画。

引理 35. $w, u \in \Sigma^*$ 成立，当且仅当存在 $e_1 = (i_1, \ldots, i_{|u|}), e_2 = (j_1, \ldots, j_{|u|}) \in E(w, u)$，满足 $e_1 = e_2$ 且

- $2 \leq j_1 \leq |w|$ 在 w 中是极小的，使得 $w[j_1] = u[1]$，
- 对所有 $r \in [|u|]_{>1}$，都有 $j_{r-1} < j_r \leq |w|$ 在 w 中是极小的，使得 $w[j_r] = u[r]$。

证明。首先，设 $w \in u \sqcup u$。我们必须证明，满足右侧条件的 e_1 和 e_2 是 u 在 w 中两个不相交且穷尽的嵌入。由于 $w \in u \sqcup u$，我们有 $w[1] = u[1]$，并且可以设 $i_1 = 1$。根据 j_1 的选取，我们有

$$|w[2..j_1 - 1]|_{u[1]} = 0.$$

令 $i_s = s$ 对所有 $s \leq j_1 - 1$ 都成立。则根据 $w \in u \sqcup u$，有 $w[i_s] = u[s]$。此外，我们有 $u[1] = w[j_1]$。现在假设对于一个固定但任意的 $r - 1 \in [|u|]_{>1}$，我们有

- $w[j_1]w[j_2] \cdots w[j_{r-1}] = u[1..r-1]$ and
- $w[1..(j_1 - 1)]w[(j_1 + 1)..(j_2 - 1)] \cdots w[(j_{r-2} + 1)..(j_{r-1} - 1)] = u[1..(j_{r-1} - r + 1)]$.

根据 j_r 的选取，我们有 $u[r] \notin$ 字母表$(w[(j_{r-1} + 1)..(j_r - 1)])$。由于 $w \in u \sqcup u$，我们有 $w[[1..(j_1 - 1)]w[(j_1 + 1)..(j_2 - 1)] \cdots w[(j_{r-1} + 1)..(j_r - 1)] = u[1..j_r - r]$，并且可以对所有 $j_{r-1} < s < j_r$ 令 $i_s = j_{r-1} + s$。这就完成了该方向的证明。

另一方向可直接得出，因为 e_1, e_2 是 u 在 w 中互不相交的穷尽嵌入。□

利用前面的刻画，我们现在能够在线性时间内判定 u 是否属于 $C(w,u)$。

命题 36。 给定 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$ ，可以在时间$O(|w|)$ 内判定 $u \in C(w,u)$是否成立。

证明。若 $u \notin C(w,u)$，则有 $w \notin u \sqcup u$。这意味着，要么 $|w| \neq 2|u|$ ，可在 $O(|w|)$ 时间内检验；要么所有满足 $|v| = |u|$ 的 $v \in C(w,u)$ 都不同于 u。根据引理 35，我们可以遍历一次 w ，并逐字母构造 u 的嵌入以及某个$v \in C(w,u)$。因此，我们有三个计数器： i 用于 w， $c1$用于 u 的一次出现次数， $c2$ 用于第二次出现次数。对于每个 i，若无法设置某个 j_r，则将 c_1 设为 i ，并递增 c_1 和 i。如果满足 j_r 的条件，就将 $c2$ 设为 i ，并递增两个计数器。一旦无法将 w的某个字母指派给 u，我们就知道 u 不属于 $C(w,u)$（参见算法 3）。□

算法 3 检查 $u \in C(w,\ u)$

输入： $w \in \Sigma^*$ 和 $u \in$ ScatFact(w)

输出： 若 $u \in C(w,\ u)$ 则为 True，否则为 False

定义 findEmbeddings (w, u):

```
c1=0
c2=0
i=0

当 i < len (w ):
    如果c1 < c2 并且u [c1]==w[i ]:
        c1 = c1+1
    否则如果 c1 < c2 and u [c1]!=w[i ]:
        if u[c2]==w[i ]:
            c2 = c2+1
        否则 :
            返回                    错假月球彩票娱乐
    否则如果 u[c2]==w[i ]:
        c2 = c2+1
    否则:
        返回          错假月球
                      彩票娱乐
    i = i+1
返回 真
```

对于 $w \in \Sigma^*$, $u \in$ ScatFact(w)， $v \in C(w,\ u)$ 满足 $|v| < |u|$，我们有 $u \notin C($ $w,\ u)$。在 $|v| > |u|$ 的情况下， u 可能是这种 $v \in C(w,\ u)$ 的散布因子。这里，我们可以对引理 35 中的思路稍作调整。

命题37。 给定 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$ ，可以在时间$O(|w|)$ 内判定是否存在 $v \in C(w,u)$ ，使得 $u \in \mathrm{ScatFact}(v)$。

证明。我们按如下方式修改引理 35：对于 $w,\ u \in \Sigma^*$ 我们有 $w \in u \sqcup\!\sqcup v$，且 $u \in \mathrm{ScatFact}(v)$ 当且仅当 $e_1 = (i_1, \ldots, i_{|u|})$ $e_2 = (j_1, \ldots, j_{|u|}) \in E(w,u)$，其中

- $2 \le j_1 \le |w|$ 在 w 中是极小的，使得 $w[j_1] = u[1]$，– </style>对于所有 $r \in [|u|]_{>1}$，我们有 $j_{r-1} < j_r \le |w|$ 在 w 中是极小的，使得$w[j_r] = u[r]$。

与引理 35 相比，算法上唯一的区别是，我们不要求 e_2 是 e_1 的补嵌入。因此，如果满足 $c1 < c2$ 和 $w[i] = u[c1]$，我们就将 w 的实际字母分配给 u 的第一次出现；如果满足 $c1 > c2$ 和 $w[i] = u[c2]$，则分配给第二次出现；如果满足 $u[c1] \ne w[i] \ne u[c2]$，则跳过这个字母。□

下面的引理表明，词 w 中的 u 是其自身唯一的补散布因子，当且仅当 w 是 u 的完美交错。

定理38。设 $w \in \Sigma^*$ 和 $u \in \mathrm{ScatFact}(w)$。那么，我们有 $C(w,u) = \{u\}$ 当且仅当 $w \in u \sqcup\!\sqcup u$。

证明。设首先有 $1 = |C(w,u)|$ 和 $u \in C(w,u)$。根据 $C(w,u)$ 的定义，我们得到 $w \in u \sqcup\!\sqcup u$。假设 $w \notin u \sqcup\!\sqcup u$。那么存在 $i \in [|w|]$ 使得$w[i] \ne u[\lceil \frac{i}{2} \rceil]$，取最小的 i。设 $j \in [|u|]$ 满足 $w[i] = u[j]$。那么 $u[1..\lceil \frac{i}{2} \rceil - 1]u[j] \ne u[1..\lceil \frac{i}{2} \rceil]$ 是词 $v \in C(w,u)$ 的一个前缀——这就产生了矛盾。

对于第二个方向，设 $w \in u \sqcup\!\sqcup u$ 和 $|u| = n$。假设存在$u \ne v \in C(w,u)$。设 $e_1 \in E(w,u)$ 和 $e_2 \in E(w,v)$ 满足 $e_1 \cup e_2$。由$w \in u \sqcup\!\sqcup u$ 和 $e_2 \notin E(w,u)$，我们得到 $e_1 \notin \{(e_{11}, e_{12}, \ldots, e_{1n}) \mid \forall i \in [n]\colon\ e_{1i} \in \{2(i-1)+1, 2i\}\}$，其中函数值写成一个元组。因此，存在 $j \in [n]$ 使得 $e_1(j) \notin \{2(j-1)+1, 2j\}$。取最小的 j。因此，我们有

$$
\begin{array}{llllll}
e_1(1) = 1, & e_1(2) = 3, & \ldots, & e_1(j-1) = 2(j-2)+1, & & \\
e_2(1) = 2, & e_2(2) = 4, & \ldots, & e_2(j-1) = 2(j-1), & e_2(j) = 2j-1 & e_2(j+1) = 2j.
\end{array}
$$

相对于 w 绘制后，我们得到

w	1	2	3	4	$\cdots$	$2j-3$	$2j-2$	$2j-1$	$2j$	$2j+1$	$\cdots$	$2n-1$	$2n$
u	1	1	2	2	$\cdots$	$j-1$	$j-1$	j	j	$j+1$	$\cdots$	n	n
e_1	1		2		$\cdots$	$j-1$							
e_2		1		2			$j-1$	j	$j+1$				

因此，我们有 $v[j] = v[j+1] = u[j]$，从而 $u[1..j]u[j] = u[1..j-1]u[j]^2$ 是 v 的前缀。根据 $e_1(j) \ge 2j+1$，存在 $2j+1 \le k_1 < k_2 < \ldots < k_{n-j+1} \le 2n$，使得 $e_1(j+r) = k_{r+1}$ 对于 $r \in [n-j]_0$ 成立。因此，对于 $r \in [n-j]_0$，我们有 $u[j+r] = w[k_{r+1}]$。根据 $w[k_1] = \overline{u[\lceil \frac{k_1}{2} \rceil]}$ 和 $\overline{\lceil \frac{k_1}{2} \rceil} > j$，我们有 $\overline{u[\lceil \frac{k_1}{2} \rceil]} = u[j]$。注意，根据鸽巢原理，存在 $r \in [n-j+1]_0$，使得 $k_r + 1 = k_{r+1}$。

选择 t，使得 $(k'_i)_{i \in [t]}$ 是 $(k_i)_{i \in [n-j+1]}$ 的严格单调递增子序列，并且满足 $k_1 = k'_1$ 和 $k'_{i+1} = e_1\left(\left\lceil \frac{k'_i}{2} \right\rceil\right)$.。由此得到序列 $s' = \left(\left\lceil \frac{k'_i}{2} \right\rceil\right)_{i \in [t]}$，它继承了严格递增的单调性。若对所有 $r \in [n-j]_0]$，都存在 $i \in [t]$ 满足 $k'_i = k_r$，那么就存在一个 $\in \Sigma$ 满足 $u[j..n] =$、一个 $^{n-j+1}$，因此 $u = v$，这就产生了矛盾。假设

存在 $p \in [n-j+1]$，使得对所有 $i \in [t]$，都有 $k'_i \neq k_p$。这就得到第二个序列$\hat{s}\hat{}k_1 = k_p$，以及当 $i \in [$时的$\hat{}k_{i+1} = e_1\left(\left\lceil \frac{\hat{k}_i}{2} \right\rceil\right)$和某个$\hat{}t \leq 2n$的$\hat{}t]$。由于这两个序列都包含 u 和 $w = u \sqcup\!\sqcup u$ 的索引，因此存在$\hat{}\ell, _\ell' \leq 2n$，使得 $k'_\ell = k_{\ell'}$。对于序列 s' 中未出现的所有索引，都可以应用基于$\hat{s}$ 的这一论证；我们再次得到 $u[j..n] =$ 、一个$^{n-j+1}$，因此$u = v$——再次产生矛盾。□

5 结论

在这项工作中，我们结合了关于散布因子和交错算子的知识，以研究补散布因子集。由于一个散布因子在一个词中可能有多个嵌入，因此对于每个这样的散布因子，我们都会得到一个（不一定是新的）补散布因子。这种方法引出了三个自然问题，分别涉及 $w \in \Sigma^*$、 $u \in$ ScatFact(w) 和 $C(w,\ u)$：给定其中两个，能否计算出第三个？针对这三个问题，我们都提出了算法；又由于（正如预期的那样）其运行时间并不适用于实际问题，我们还从组合角度研究了 $C(w,\ u)$。借助关于拱分解和α-β-分解的知识，我们研究了在什么情况下会有一个或多个补散布因子。此外，我们解决了以下问题：何时 u 是 u 的补散布因子，以及何时它是唯一的补散布因子。一个尚待解决的问题是，研究比给定词的普适性指标更长的散布因子。此外，$|C(w,\ u)|$ 的上界，乃至特定情况下的闭式公式，也仍有待研究。

参考文献

1. D. Adamson、P. Fleischmann、A. Huch、T. Koß 和 F. Manea。重新审视正则语言的k-普适性。载于算法学前沿，第 16–32 页，2025年。2. D. Adamson、P. Fleischmann、A. Huch、T. Koß、F. Manea 和 D. Nowotka。正则语言的 k-普适性。载于*ISAAC 2023*，*LIPIcs*第 283 卷，第 4:1–4:21 页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2023年。3. D. Adamson、P. Fleischmann、A. Huch、F. Manea、P. Sarnighausen-Cahn 和 M. Wiedenhöft。缺失子序列数量的紧确界。16106:15–29，2025年。4. A. Artikis、A. Margara、M. Ugarte、S. Vansummeren 和 M. Weidlich。复杂事件识别语言：教程。载于*DEBS*，第 7–10 页，2017年。5. J.C.M. Baeten 和 W.P. Weijland。进程代数。剑桥理论计算机科学论丛。剑桥大学出版社，1990年。6. R. A. Baeza-Yates。子序列搜索。*TCS*，78(2):363–376，1991年。7. H. Bannai、T. I、T. Kociumaka、D. Köppl 和 S. J. Puglisi。计算最长 Lyndon 子序列和最长公共 Lyndon 子序列。*Algorithmica*，86(3):735–756，2024年。8. L. Barker、P. Fleischmann、K. Harwardt、F. Manea 和 D. Nowotka。词的散布因子全性。载于*DLT 2020*，*LNCS*第 12086 卷，第 14–28 页。Springer，2020年。

9. P. Baumann、M. Ganardi、R. S. Thinniyam 和 G. Zetzsche。子词序上的存在可定义性。*Log. Methods Comput. Sci.*，19(4)，2023年。10. J. Berstel、L. Boasson、O. Carton、J.-E. Pin 和 A. Restivo。交错乘积的表达能力。*Inf. Comput.*，208(11):1258–1272，2010年。11. K. Bringmann 和 M. Künnemann。最长公共子序列的多变量精细复杂性。载于*SODA 2018*，第 1216–1235 页。SIAM，2018年。12. L. Bulteau 和 S. Vialette。识别二元交错平方是NP难的。*TCS*，806:116–132，2020年。13. S.R. Buss 和 M. Soltys。将平方反交错是NP难的。*J. Comput. Syst. Sci.*，80:766–776，2014年。14. C. Câmpeanu、K. Salomaa 和 S. Vágvölgyi。正则语言的交错分解。*Int. J. Found. Comput. Sci.*，13(6):799–816，2002年。15. J. Currie 和 K. Saari。具有无平方因子自交错的无平方因子词。***The Electronic Journal of Combinatorics***，21，2014年1月。16. W. Czerwinski 和 S. Lasota。部分交换上下文无关语言。载于*EXPRESS/SOS 2012, Newcastle upon Tyne, UK, September 3, 2012*，*EPTCS*第 89 卷，第 35–48 页，2012年。17. J. D. Day、P. Fleischmann、M. Kosche、T. Koß、F. Manea 和 S.Siemer。到 k-子序列普适性的编辑距离。载于*STACS 2021*，*LIPIcs*第 187 卷，第 25:1–25:19 页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2021年。18. P. Diaconis、R.L. Graham 和 W.M. Kantor。完美交错的数学。*Advances in Applied Mathematics*，4(2):175–196，1983年。19. D. T. Do、T. T. Q. Le 和 N.-Q.-K. Le。利用深度神经网络和生物学子词检测蛋白质S-亚磺酰化位点。*Briefings Bioinform.*，22(3)，2021年。20. A. W. M. Dress 和 P. L. Erdős。在线性时间内从子词重构词。*Annals of Combinatorics*，8:457–462，2005年。21. S. Eilenberg 和 S. Mac Lane。关于群 H(Π, n)，I。*Annals of Mathematics*，58(1):55–106，1953年。22. J.A. Ellis、H. Fan 和 J. Shallit。多路完美交错置换的循环。*Discret. Math. Theor. Comput. Sci.*，5:169–180，2002年。23. Z. Ésik 和 I. Simon。用交错对字面态射建模。*Semigroup Forum*，56:225–227，1998年。24. S. Z. Fazekas、T. Koß、F. Manea、R. Mercas 和 T. Specht。形式语言的子序列匹配与分析问题。载于*ISAAC 2024*，*LIPIcs*第 322 卷，第 28:1–28:23 页。Schloss Dagstuhl - Leibniz-Zentrum für Informatik，2024年。25. P. Fleischmann、S.B. Germann 和 D. Nowotka。散布因子普适性——剩余部分的力量。预印本 *arXiv:2104.09063*（发表于 *RuFiDim*），2021年。26. P. Fleischmann、L. Haschke、J. Höfer、A. Huch、A. Mayrock 和 D. Nowotka。近k-普适词——探究 Simon同余的一部分。*TCS*，974:114113，2023年。27. P. Fleischmann、J. Höfer、A. Huch 和 D. Nowotka。 α-β-分解与 Simon同余的二元情形。载于*FCT 2023*，*LNCS*第 14292 卷，第 190–204 页。Springer，2023年。28. P. Fleischmann、S. Kim、T. Koß、F. Manea、D. Nowotka、S. Siemer 和 M. Wiedenhöft。在 Simon同余下匹配含变量模式。载于*RP 2023*，*LNCS*第 14235 卷，第 155–170 页。Springer，2023 年。29. P. Fleischmann 、 M. Lejeune 、 F. Manea 、 D. Nowotka 和 M. Rigo。从右有界块词重构词。*Int. J. Found. Comput. Sci.*，32(6):619–640，2021年。30. P. J. Forrester 和 A. Mays。与最长递增子序列长度分布相关的有限尺寸修正。*Adv. Appl. Math.*，145:102482，2023年。

31. A. Frochaux and S. Kleest-Meißner. 关于子序列查询扩展的思考：析取与广义间隙。载于*AMW 2023*，收入*CEUR Workshop Proceedings*第3409卷，2023年。 32. J. Haghighat and T. M. Duman. 用于删除信道的半标记码及其在 DNA 存储中的应用。*IEEE Communications Letters*，2025年。 33. S. Halfon, P. Schnoebelen, and G. Zetzsche. 子词序上一阶逻辑的可判定性、复杂性与表达能力。载于*LICS 2017*，第1–12页。IEEE计算机学会，2017年。 34. T. Harju. 关于词的无平方交错的一则短札。载于*WORDS 2013*，收入*LNCS*第8079卷，第154–160页。Springer，2013年。 35. T. Harju and M. Müller. 词的无平方交错。*TCS*，601：29–38，2015年。 36. J.-J. Hébrard. 一种根据子序列高效区分比特串的算法。*TCS*，82(1)：35–49，1991年。 37. P. Karandikar, M. Kufleitner, and P. Schnoebelen. 分段可测试性的Simon同余的指标。信息处理快报，115(4)：515–519，2015年。 38. P. Karandikar and P. Schnoebelen. 分段可测试语言的高度及其在逻辑复杂性中的应用。载于*Proc. CSL*，收入*LIPIcs*第62卷，第37:1–37:22页，2016年。 39. P. Karandikar and P. Schnoebelen. 分段可测试语言的高度与子词逻辑的复杂性。*LICS*，15(2)，2019年。 40. P. Karandikar and P. Schnoebelen. 分段可测试语言的高度与子词逻辑的复杂性。*Log. Methods Comput. Sci.*，15(2)，2019年。 41. J. Kärkkäinen, P. Sanders, and S. Burkhardt. 线性工作量的后缀数组构造。《*ACM*期刊》，53(6)：918–936，2006年。 42. S. Kim, Y.-S. Han, S.-K. Ko, and K. Salomaa. 关于字符串的Simon同余闭包。*TCS*，972：114078，2023年。 43. S. Kim, Y.-S. Han, S.-K. Ko, and K. Salomaa. 关于语言的Simon同余邻域。载于*DLT*，收入*LNCS*第13911卷，第168–181页。Springer，2023年。 44. S. Kim, S.-K. Ko, and Y.-S. Han. Simon 同余模式匹配。*TCS*，994：114478，2024年。 45. S. Kleest-Meißner, R. Sattler, M. L. Schmid, N. Schweikardt, and M. Weidlich. 从轨迹中发现事件查询：为带通配符和间隙大小约束的子序列查询奠定基础。载于*ICDT*，收入*LIPIcs*第220卷，第18:1–18:21页，2022年。 46. S. Kleest-Meißner, R. Sattler, Markus L. Schmid, N. Schweikardt, and M. Weidlich. 从轨迹中发现多维子序列查询——从理论到实践。载于*BTW*，收入*LNI*第P-331卷，第511–533页，2023年。 47. M. Kosche, T. Koß, F. Manea, and S. Siemer. 词中的缺失子序列。《基础信息学》，189(3-4)：199–240，2022年。 48. D. Kuske. 子轨迹序与计数一阶逻辑。载于计算机科学——理论与应用，第289–302页。施普林格国际出版，2020年。 49. D. Kuske and G. Zetzsche. 按子词序排列的语言。载于软件科学与计算结构基础，第348–364页。施普林格国际出版，2019年。 50. M. Lothaire. 组合词学。剑桥数学文库。剑桥大学出版社，1997年。 51. D. Maier. 关于子序列与超序列的若干问题的复杂性。《*ACM*期刊》，25(2)：322–336，1978年。 52. J. Manuch. 由子词刻画一个词。载于*DLT 1999*，第210–219页。世界科技出版公司，1999年。

53. A. Mateescu, A. Salomaa, and S. Yu. 子词历史与Parikh矩阵。*J. Comput. Syst. Sci.*，68(1)：1–21，2004年。 54. A. Restivo. 交错乘积：新的研究方向。载于语言与自动机理论及其应用，第70–81页。施普林格国际出版，2015年。 55. R. Rizzi and S. Vialette. 关于识别交错乘积意义下为平方的词。载于计算机科学——理论与应用，第235–245页。施普林格柏林海德堡，2013年。 56. O. Sabary, A. Yucovich, G. Shapira, and E. Yaakobi. DNA存储系统的重构算法。《科学报告》，14(1)：1951，2024年。 57. R. Sattler, S. Kleest-Meißner, S. Lange, M. L. Schmid, N. Schweikardt, and M. Weidlich. Disces：系统地发现事件流查询。*ACM*数据管理会议录，3(1)，2025年。 58. P. Schnoebelen and J. Veron. 词与压缩词的弓形分解和子词普遍性。载于*WORDS 2023*，收入*LNCS*第13899卷，第274–287页。Springer，2023年。 59. I. Simon. 点深度一的事件层级。博士论文，滑铁卢大学，1972 年。 60. I. Simon. 分段可测试事件。载于自动机理论与形式语言，第*2*届*GI*会议，凯撒斯劳滕，*1975*年*5*月*20*日至*23*日，收入*LNCS*第33卷，第214–222页。Springer，1975年。 61. Z. Tronícek. 关于缺失子序列相关问题。载于*COCOA 2023*，收入*LNCS*第14462卷，第351–363页。Springer，2023年。 62. R. A. Wagner and M. J. Fischer. 字符串到字符串校正问题。《*ACM*期刊》，21(1)：168–173，1974年。 63. C. Wang, K. Cho, and J. Gu. 使用字节级子词的神经机器翻译。载于*EAAI 2020*，第9154–9160页。AAAI出版社，2020年。 64. G. Zetzsche. 向下闭包比较的复杂性。载于*ICALP 2016*，收入*LIPIcs*第55卷，第123:1–123:14页。达格施图尔城堡——莱布尼茨信息学中心，2016年。